# Week #1

**Study and understand the basic networking tools - Wireshark, Tcpdump, Ping, Traceroute.**

NAME = RACHAPPA

SRN = PES1UG19CS359

ROLL NO = 1

## Task 1: Linux Interface Configuration (ifconfig / IP command)

**Step 1:** To display status of all active network interfaces.

**ifconfig** (or) **ip addr show**

Analyze and fill the following table:

```
rachappa@balaji:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::835d:883e:3f1c:a0b5  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:37:41:ee  txqueuelen 1000  (Ethernet)
        RX packets 199  bytes 240941 (240.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 151  bytes 16957 (16.9 KB)
        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 128  bytes 10895 (10.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 128  bytes 10895 (10.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

**ip address table:**

| **Interface name** | **IP address (IPv4 / IPv6)** | **MAC address** | |
|---|---|---|---|
| Enp0s3 | 10.0.2.15 | 08-00-27-37-41-ee | |
| lo | 127.0.0.1 | | |
| | | | |

**Step 2:** To assign an IP address to an interface, use the following command.

**sudo ifconfig interface_name 10.0.your_section.your_sno netmask 255.255.255.0** (or)

**sudo ip addr add 10.0.your_section.your_sno /24  dev interface_name**

```
root@balaji:/home/rachappa# sudo ifconfig enp0s3 10.0.1.1
root@balaji:/home/rachappa# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.1.1  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::835d:883e:3f1c:a0b5  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:37:41:ee  txqueuelen 1000  (Ethernet)
        RX packets 260  bytes 283022 (283.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 214  bytes 31295 (31.2 KB)
        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 137  bytes 11660 (11.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 137  bytes 11660 (11.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

**Step 3:** To activate / deactivate a network interface, type.

**sudo ifconfig interface_name down**

```
root@balaji:/home/rachappa# sudo ifconfig enp0s3 down
root@balaji:/home/rachappa# ifconfig
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 138  bytes 11733 (11.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 138  bytes 11733 (11.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@balaji:/home/rachappa#
```

**sudo ifconfig interface_name up**

```
root@balaji:/home/rachappa# sudo ifconfig enp0s3 up
root@balaji:/home/rachappa# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::835d:883e:3f1c:a0b5  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:37:41:ee  txqueuelen 1000  (Ethernet)
        RX packets 268  bytes 284424 (284.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 243  bytes 34775 (34.7 KB)
        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 138  bytes 11733 (11.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 138  bytes 11733 (11.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

**Step 4:** To show the current neighbor table in kernel, type

**ip neigh**

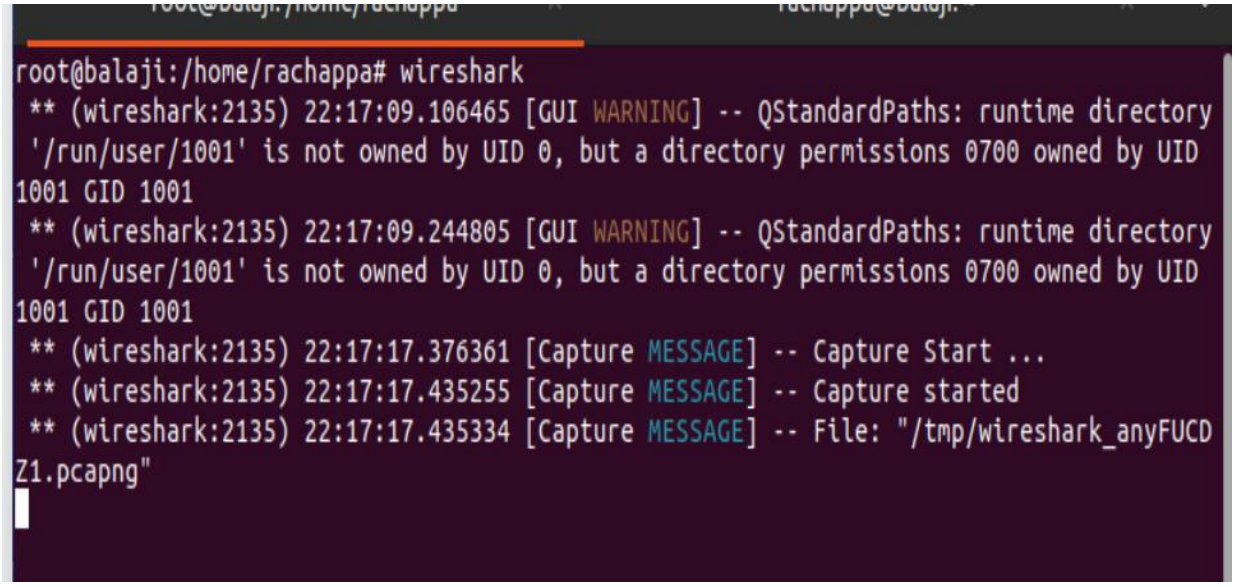
```
root@balaji:/home/rachappa# ip neigh
10.0.2.2 dev enp0s3 lladdr 52:54:00:12:35:02 REACHABLE
root@balaji:/home/rachappa#
```

## Task 2: Ping PDU (Packet Data Units or Packets) Capture

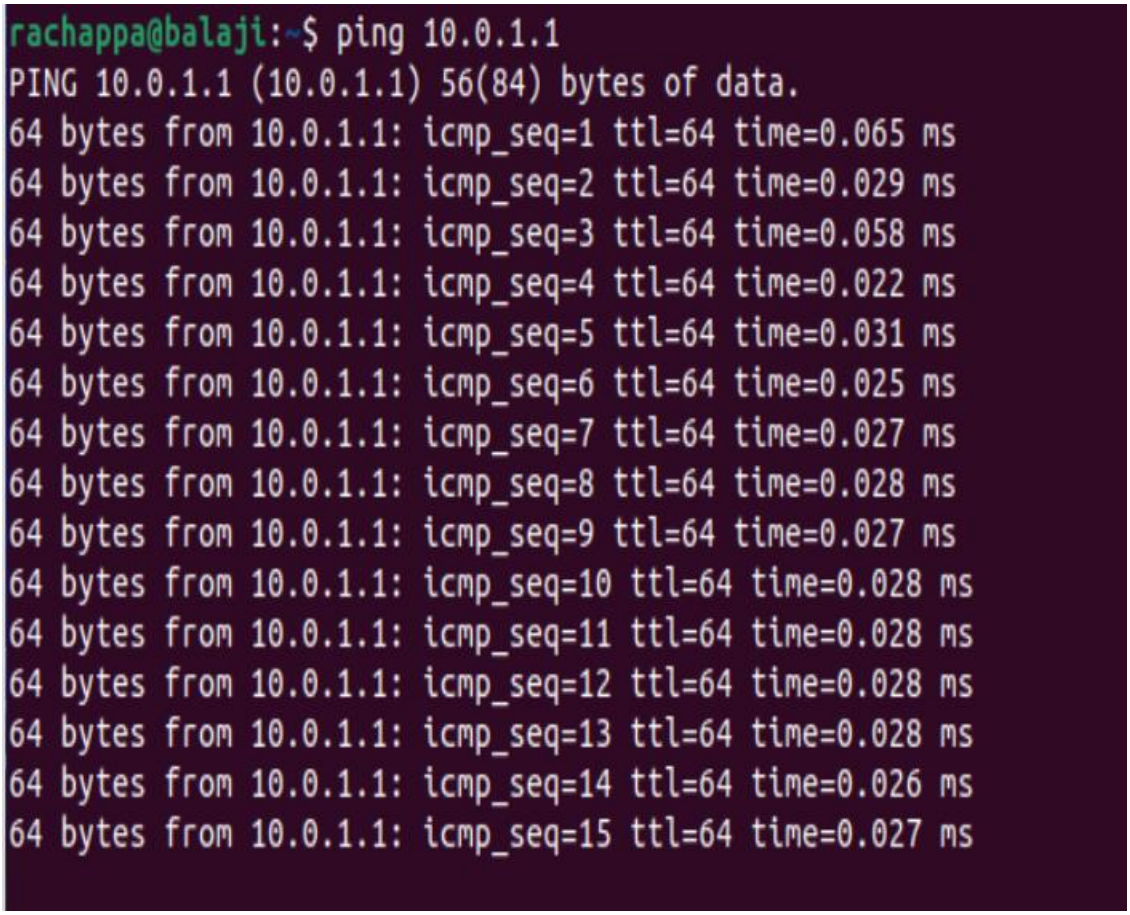
**Step 1:** Assign an IP address to the system (Host).

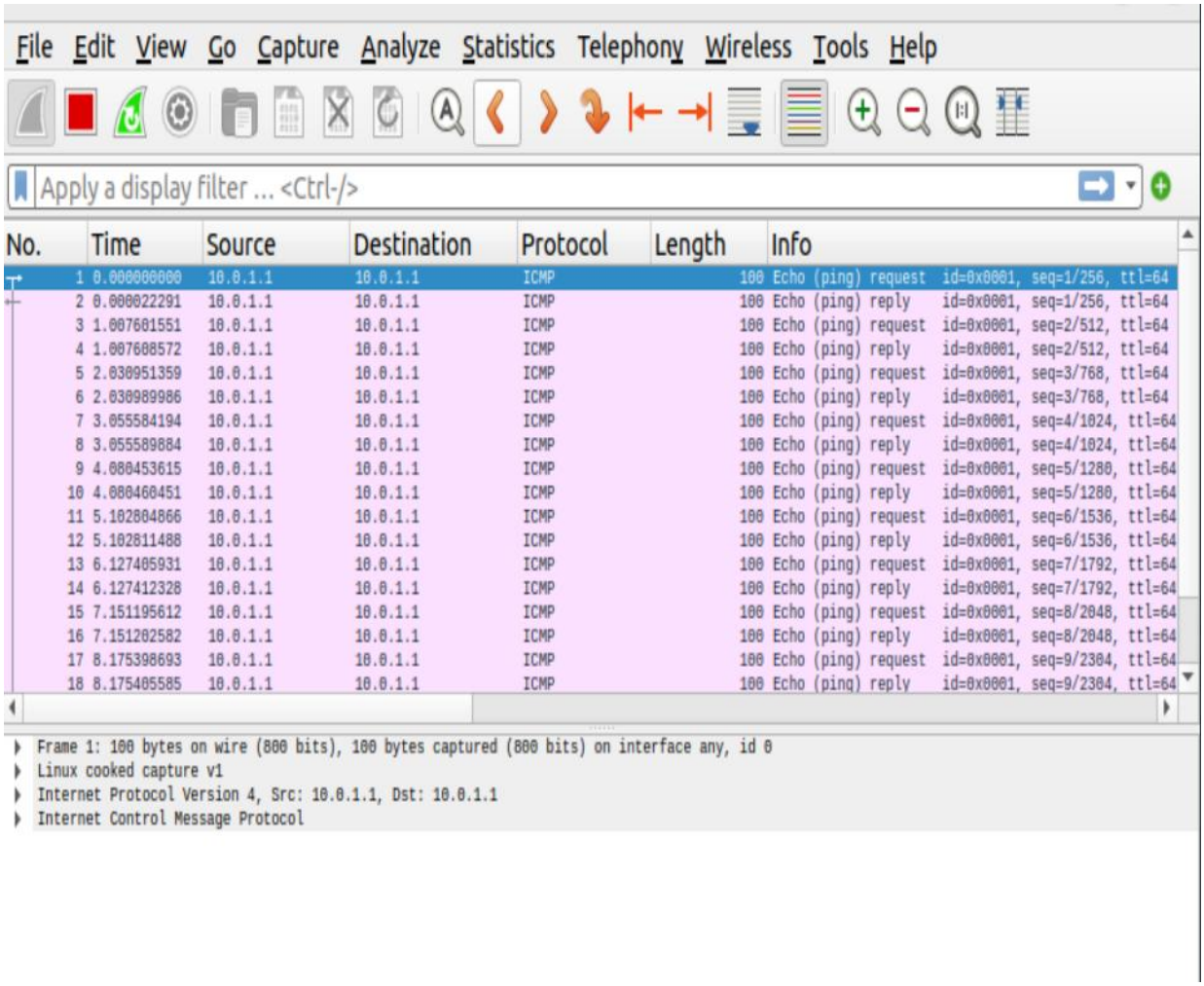
Note: IP address of your system should be 10.0.your_section.your_sno.

**Step 2:** Launch Wireshark and select ‘any’ interface




**Step 3:** In terminal, type **ping 10.0.your_section.your_sno**






**Observations to be made**

**Step 4:** Analyze the following in Terminal

- TTL

TTL IS TTL IS 64

- Protocol used by ping

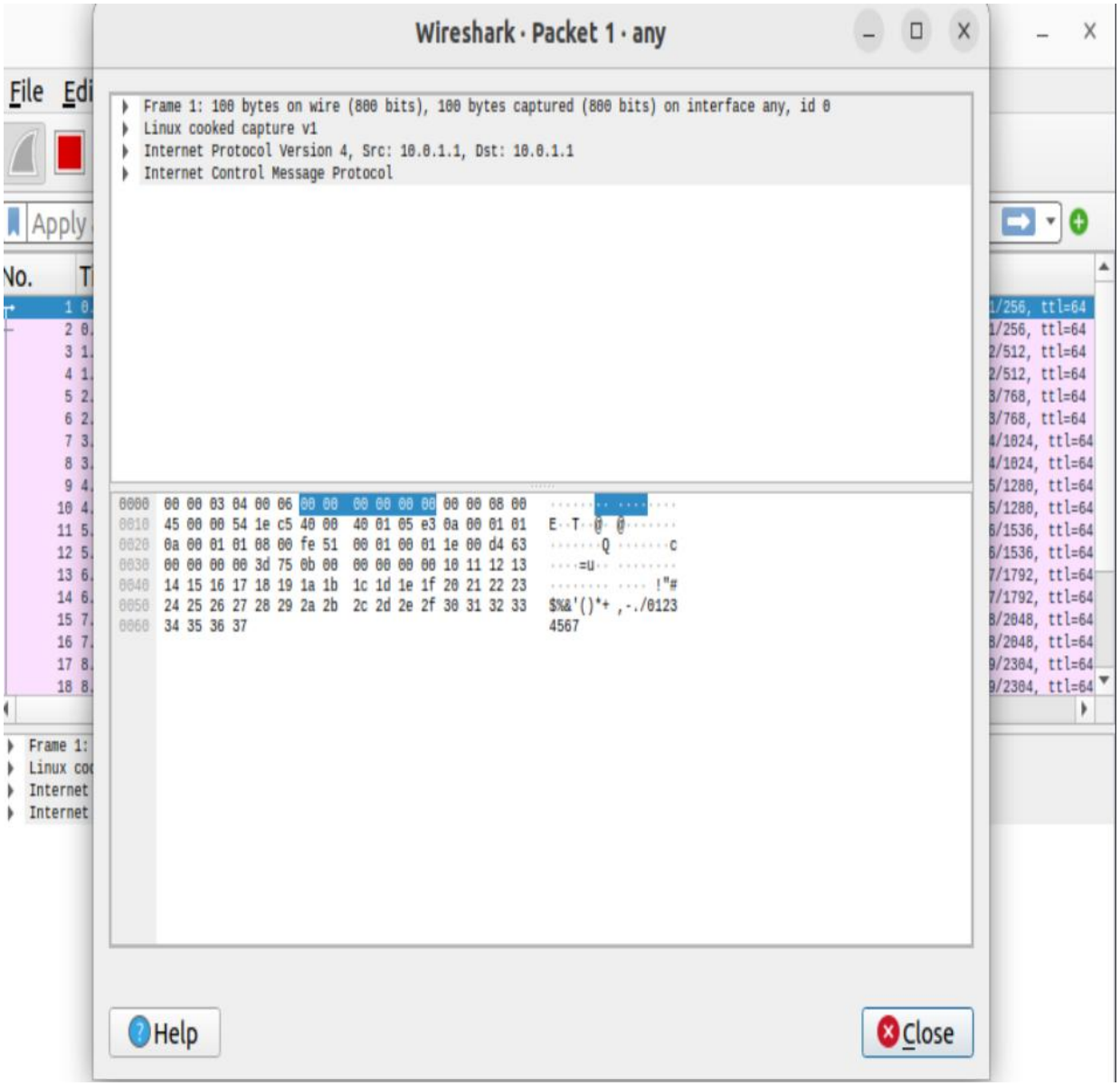
PING PROTOCOL WORK ON ICMP

- Time

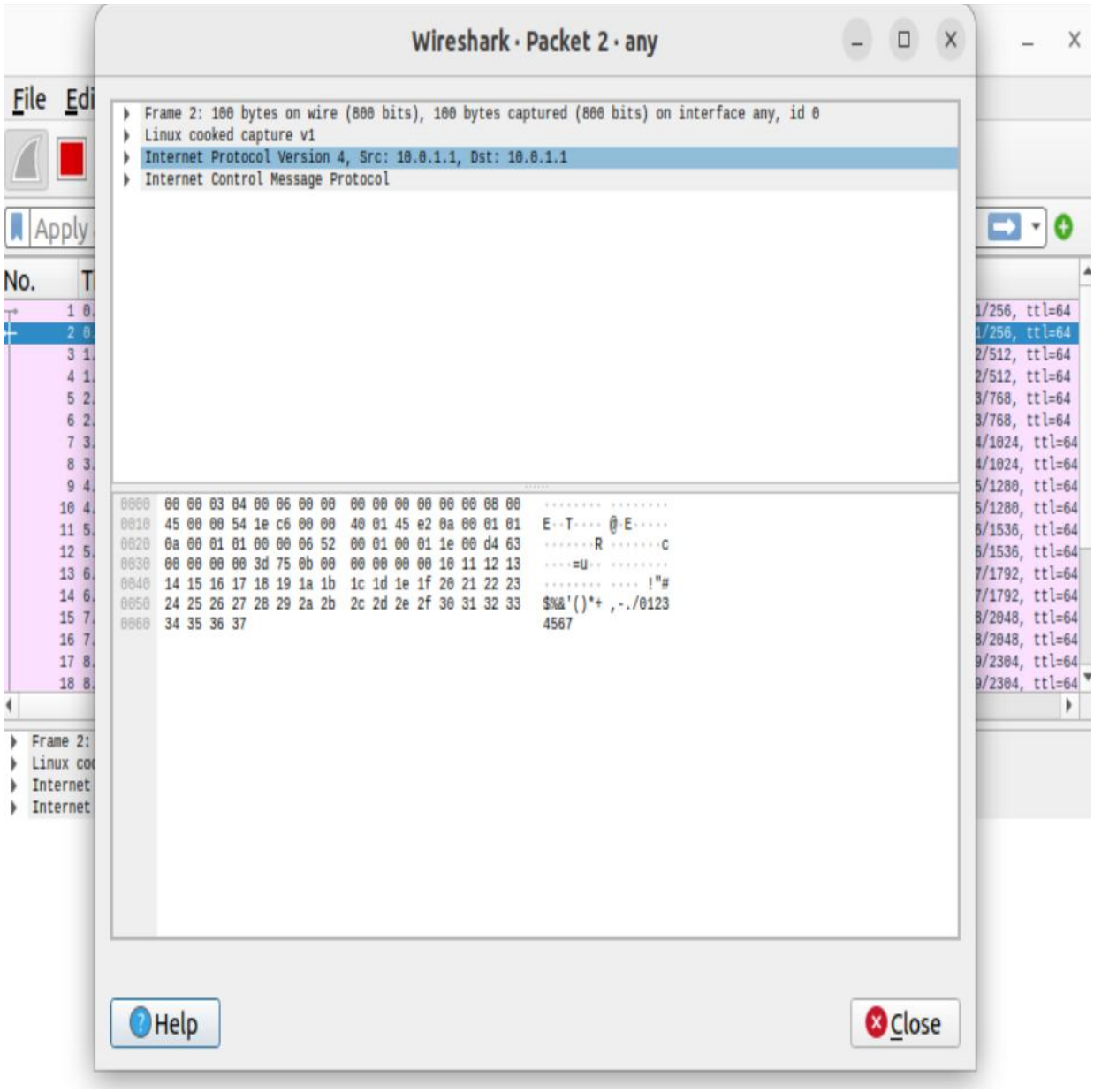
Rtt min/avg/max/mdev = 0.022/0.035/0.119/0.019

**Step 5:** Analyze the following in Wireshark

Frame 1


Wireshark · Packet 1 · any
Frame 1: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 10.0.1.1, Dst: 10.0.1.1
Internet Control Message Protocol
0000 00 00 03 04 00 06 00 00 00 00 00 00 08 00
0010 45 00 00 54 1e c5 40 00 40 01 05 e3 0a 00 01 01
0020 0a 00 01 01 08 00 fe 51 00 01 00 01 1e 00 d4 63
0030 00 00 00 00 3d 75 0b 00 00 00 00 00 10 11 12 13
0040 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23
0050 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33
0060 34 35 36 37
Help
Close

Frame 2


Wireshark · Packet 2 · any
Frame 2: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 10.0.1.1, Dst: 10.0.1.1
Internet Control Message Protocol
0000 00 00 03 04 00 06 00 00 00 00 00 00 00 00 08 00
0010 45 00 00 54 1e c6 00 00 40 01 45 e2 0a 00 01 01
0020 0a 00 01 01 00 00 06 52 00 01 00 01 1e 00 d4 63
0030 00 00 00 00 3d 75 0b 00 00 00 00 00 10 11 12 13
0040 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23
0050 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33
0060 34 35 36 37
Help
Close

On Packet List Pane, select the first echo packet on the list. On Packet Details Pane, click on each of the four "+" to expand the information. Analyze the frames with the first echo request and echo reply and complete the table below.

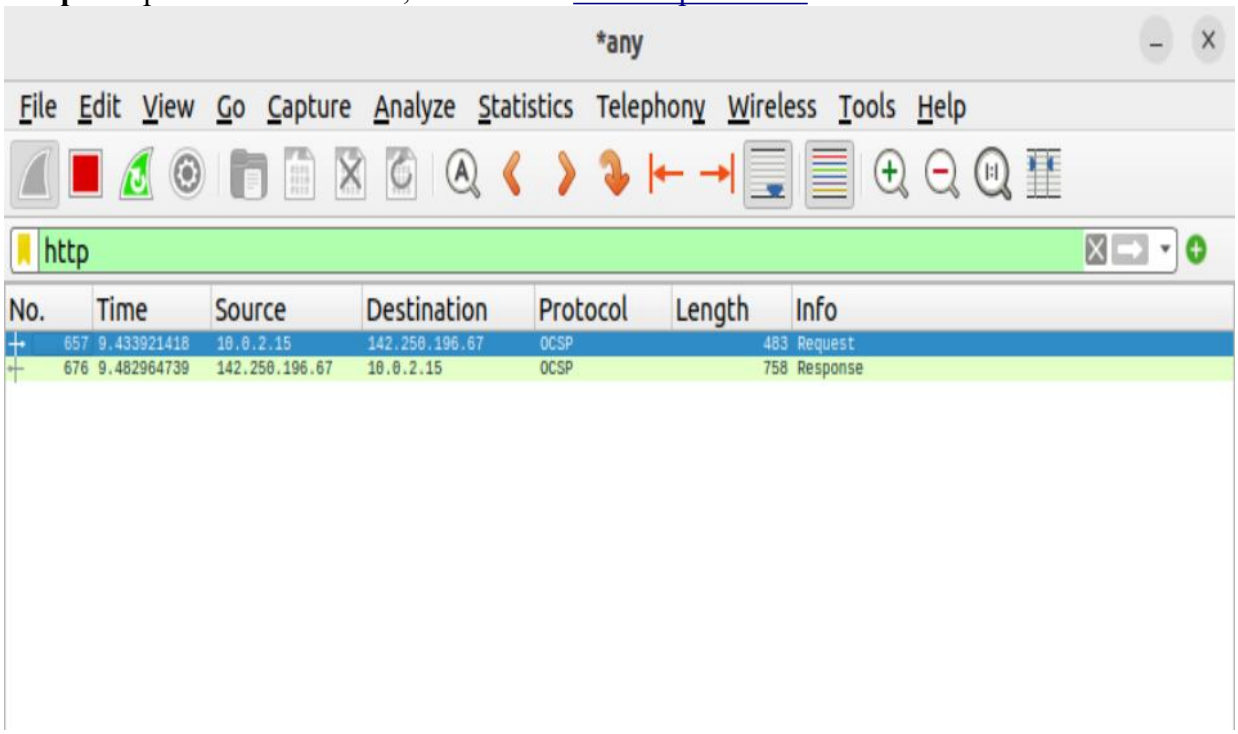
| Details | First Echo Request | First Echo Reply |
|---|---|---|
| Frame Number | 1 | 2 |
| Source IP address | 10.0.1.1 | 10.0.1.1 |
| Destination IP address | 10.0.1.1 | 10.0.1.1 |
| ICMP Type Value | 8 | 0 |
| ICMP Code Value | 0 | 0 |
| Source Ethernet Address | 10.0.1.1 | 10.0.1.1 |
| Destination Ethernet Address | 10.0.1.1 | 10.0.1.1 |
| Internet Protocol Version | 4 | 4 |
| Time To Live (TTL) Value | 64 | 64 |

## Task 3: HTTP PDU Capture

### Using Wireshark's Filter feature

**Step 1:** Launch Wireshark and select 'any' interface. On the Filter toolbar, type-in 'http' and press enter

**Step 2:** Open Firefox browser, and browse www.flipkart.com




**Observations to be made**

**Step 3:** Analyze the first (interaction of host to the web server) and second frame (response of server to the client). By analyzing the filtered frames, complete the table below:

| Details | First Echo Request | First Echo Reply |
|---|---|---|

| | | |
|---|---|---|
| Frame Number | 657 | 676 |
| Source Port | 60892 | 80 |
| Destination Port | 180 | 60892 |
| Source IP address | 10.0.2.15 | 142.250.196.67 |
| Destination IP address | 142.250.196.67 | 10.0.2.15 |
| Source Ethernet Address | 10.0.2.15 | 142.250.196.67 |
| Destination Ethernet Address | 142.250.196.67 | 10.0.2.15 |

**Step 4:** Analyze the HTTP request and response and complete the table below.

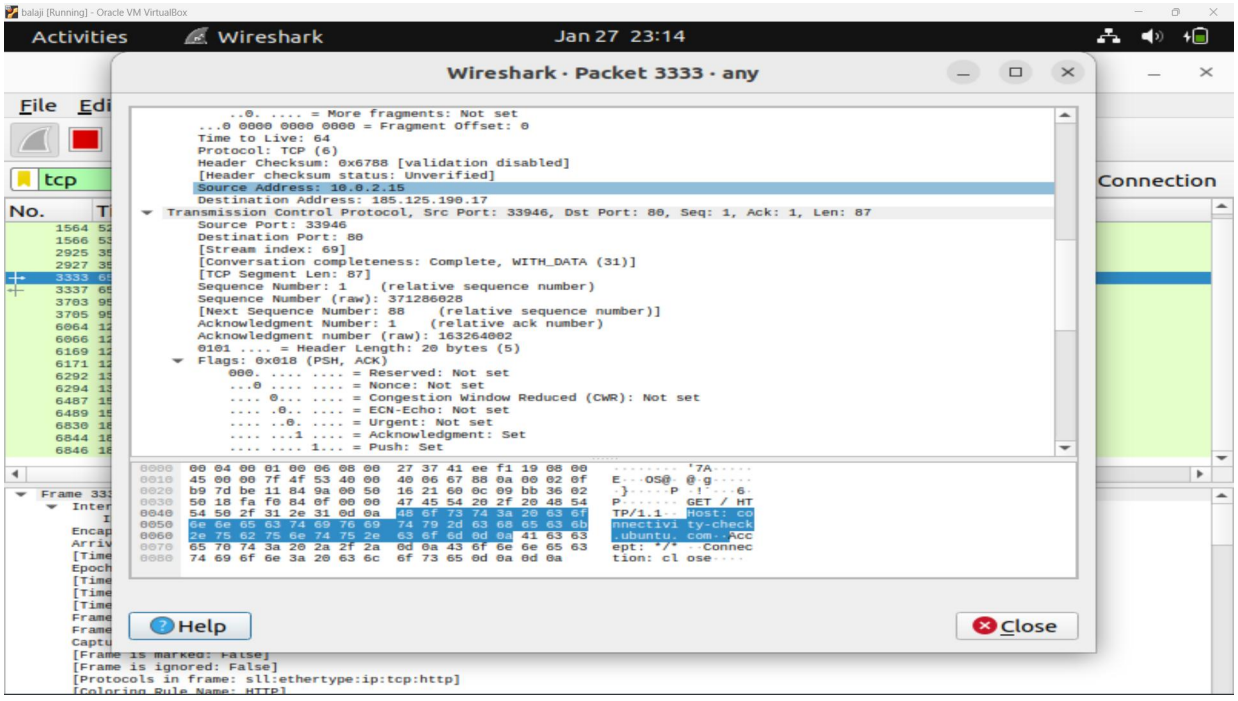
| **HTTP Request** | | **HTTP Response** | |
|---|---|---|---|
| Get | | Server | |

| | | | |
|---|---|---|---|
| Host | Ocsp.pki.goog \r\n | Content-Type | Application/ocsp-request |
| User-Agent | Mozilla/5.0 | Date | Jan 27 2023-22 |
| Accept-Language | En-us | Location | |
| Accept-Encoding | Gzip,deflate | Content-Length | 472\r\n |
| Connection | Keep-alive | Connection | |

## Using Wireshark's Follow TCP Stream

**Step 1:** Make sure the filter is blank. Right-click any packet inside the Packet List Pane, then select 'Follow TCP Stream'. For demo purpose, a packet containing the HTTP GET request "GET / HTTP / 1.1" can be selected.

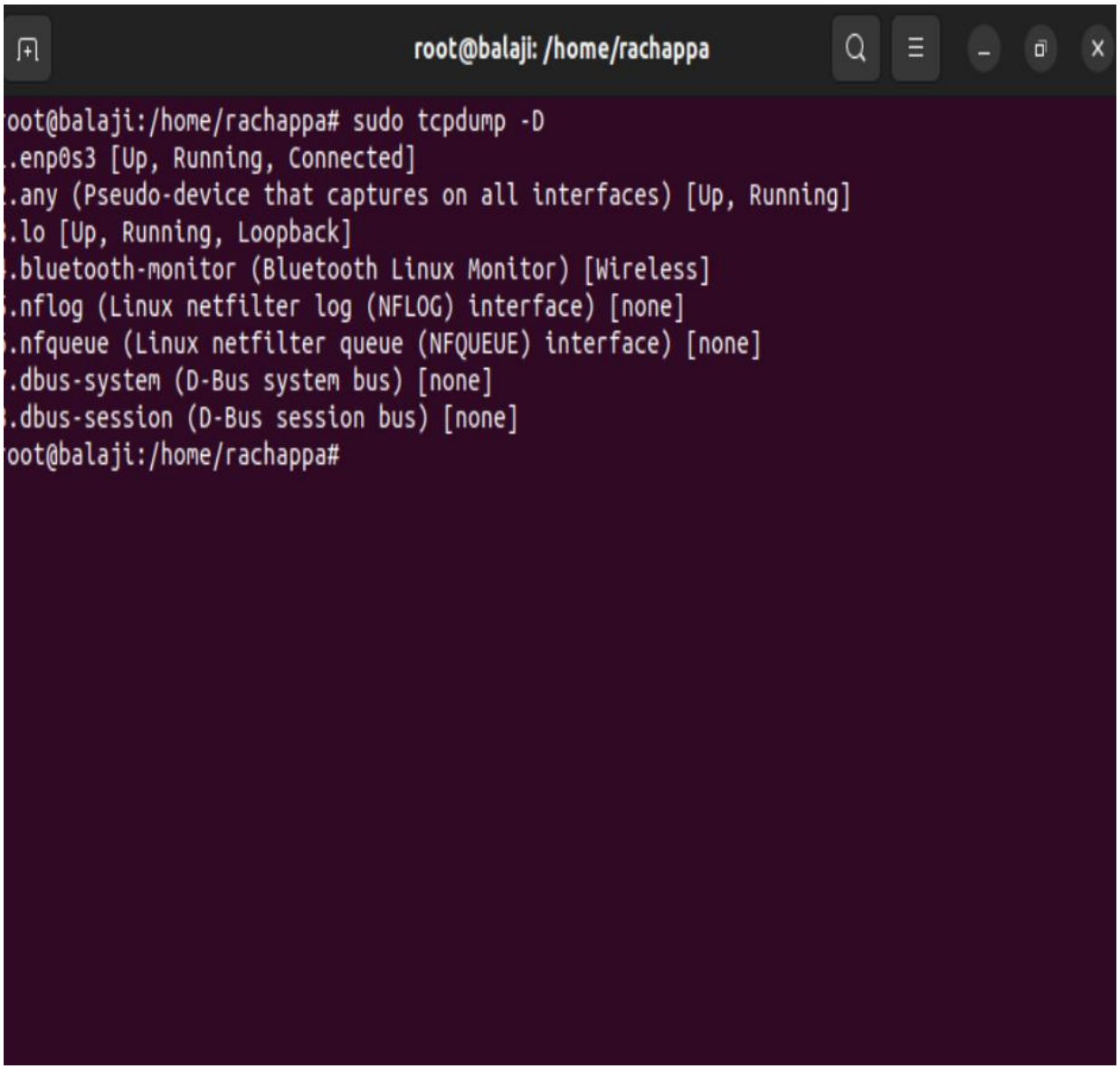
Step 2: Upon following a TCP stream, screenshot the whole window.




## Task 4: Capturing packets with tcpdump

**Step 1:** Use the command **tcpdump -D** to see which interfaces are available for capture.

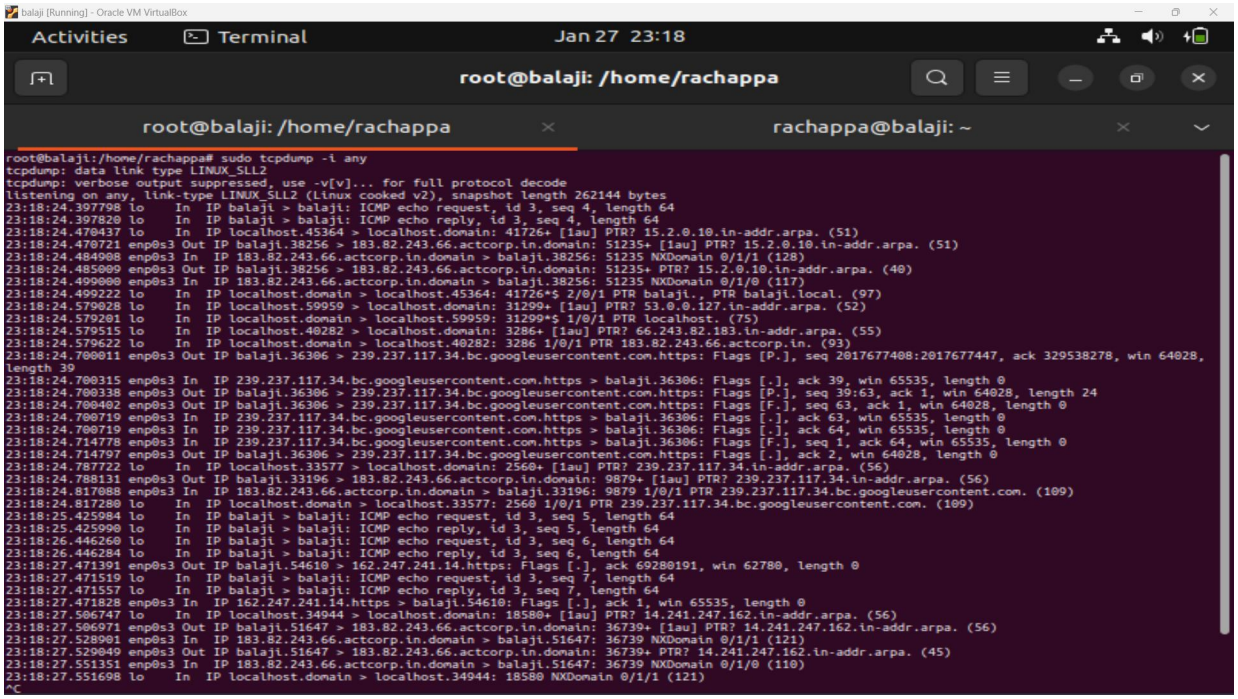
**sudo tcpdump -D**




**Step 2:** Capture all packets in any interface by running this command:

**sudo tcpdump -i any**

Note: Perform some pinging operation while giving above command. Also type www.google.com in browser.

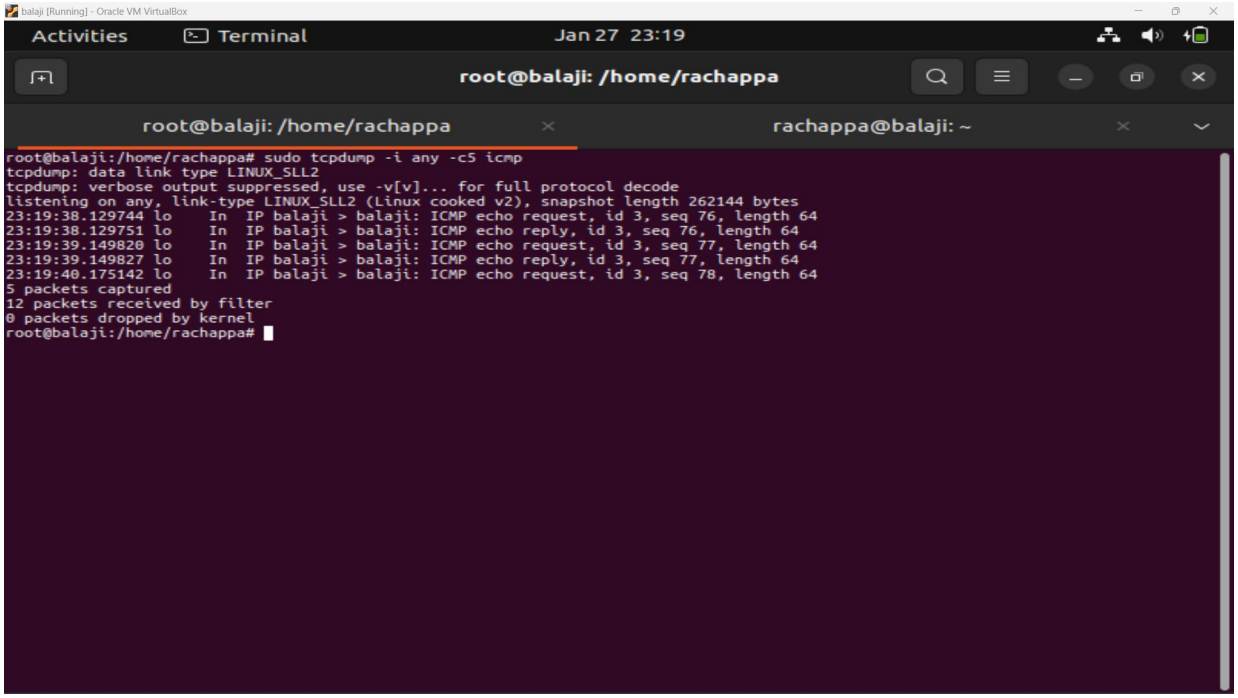



**Observation**

**Step 3:** Understand the output format.

**Step 4:** To filter packets based on protocol, specifying the protocol in the command line. For example, capture ICMP packets only by using this command:

**sudo tcpdump -i any -c5 icmp**




**Step 5:** Check the packet content. For example, inspect the HTTP content of a web request like this:

**sudo tcpdump -i any -c10 -nn -A port 80**

```
root@balaji:/home/rachappa# sudo tcpdump -i any -c10 -nn -A port 80
tcpdump: data link type LINUX_SLL2
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
23:21:46.809110 enp0s3 Out IP 10.0.2.15.47508 > 142.250.182.67.80: Flags [S], seq 18074
29992, win 64240, options [mss 1460,sackOK,TS val 2966105543 ecr 0,nop,wscale 7], lengt
h 0
E..<0.@.@...
......C...Pk.1h........Q{.........
..-.........
23:21:46.820942 enp0s3 In  IP 142.250.182.67.80 > 10.0.2.15.47508: Flags [S.], seq 3776
64001, ack 1807429993, win 65535, options [mss 1460], length 0
E..,iX..@..'...C
....P......k.1i`...'=........
23:21:46.821015 enp0s3 Out IP 10.0.2.15.47508 > 142.250.182.67.80: Flags [.], ack 1, wi
n 64240, length 0
E..(0.@.@...
......C...Pk.1i....P...Qg..
23:21:46.821128 enp0s3 Out IP 10.0.2.15.47508 > 142.250.182.67.80: Flags [P.], seq 1:42
8, ack 1, win 64240, length 427: HTTP: POST /gts1c3 HTTP/1.1
E...0.@.@...
......C...Pk.1i....P...S...POST /gts1c3 HTTP/1.1
Host: ocsp.pki.goog
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/10
9.0
```

```
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/ocsp-request
Content-Length: 84
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache

0R0P0N0L0J0     ..+.........y...a4...GB....$.c...t.......=...F..q5.'....,\...V1
.2.I..4
23:21:46.824971 enp0s3 In  IP 142.250.182.67.80 > 10.0.2.15.47508: Flags [.], ack 428,
win 65535, length 0
E..(iY..@..*...C
....P......k.3.P...=O........
23:21:46.863162 enp0s3 In  IP 142.250.182.67.80 > 10.0.2.15.47508: Flags [P.], seq 1:70
3, ack 428, win 65535, length 702: HTTP: HTTP/1.1 200 OK
E...ih..@..]...C
....P......k.3.P....Z..HTTP/1.1 200 OK
Content-Type: application/ocsp-response
Date: Fri, 27 Jan 2023 17:51:46 GMT
Cache-Control: public, max-age=14400
Server: ocsp_responder
Content-Length: 472
X-XSS-Protection: 0
X-Frame-Options: SAMEORIGIN
```

**Step 6:** To save packets to a file instead of displaying them on screen, use the option -w:

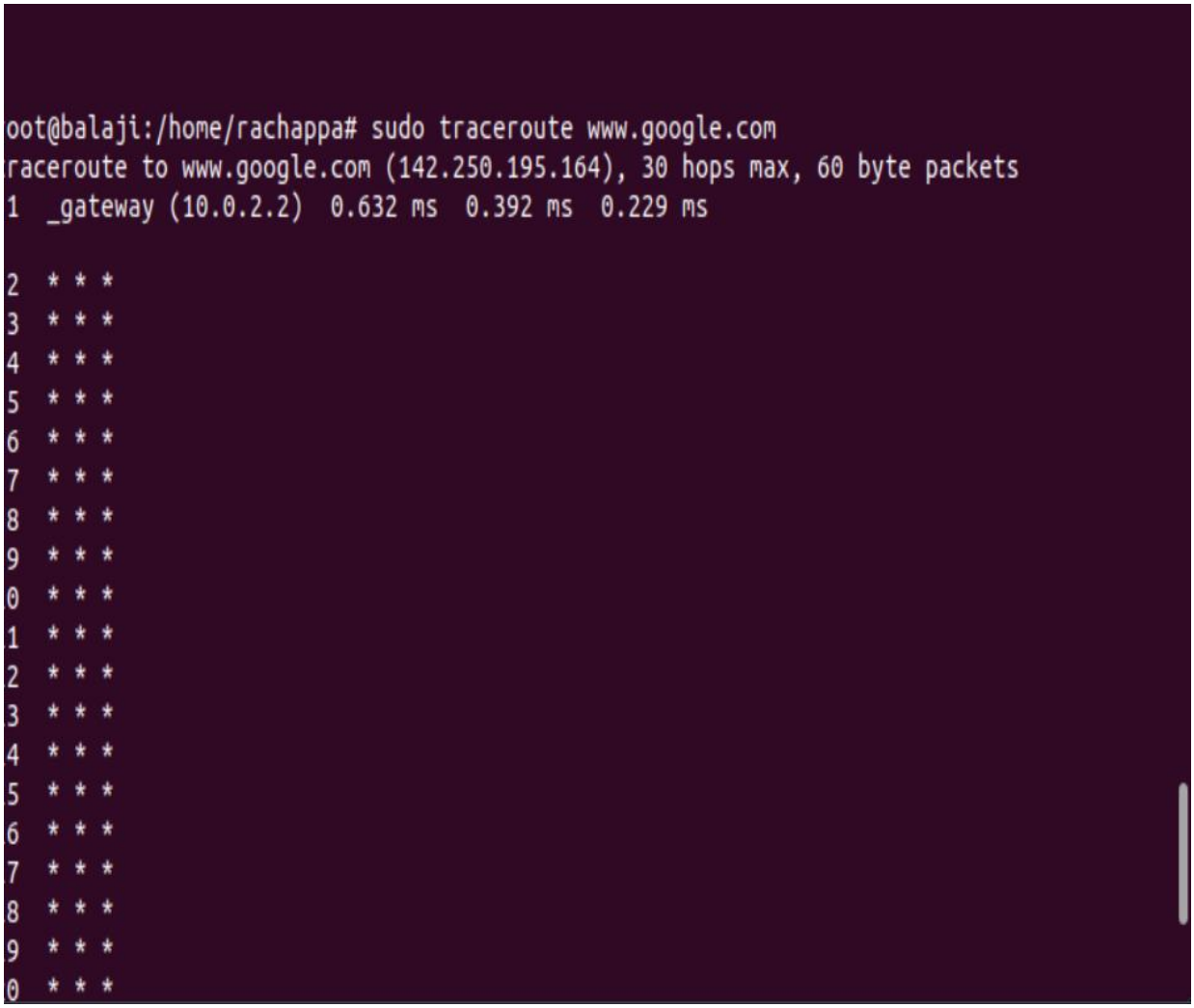
**sudo tcpdump -i any -c10 -nn -w webserver.pcap port 80**

```
root@balaji: /home/rachappa
root@balaji: /home/rachappa          rachappa@balaji: ~
root@balaji:/home/rachappa# sudo tcpdump -i any -c10 -nn -w webserver.pcap port 80
tcpdump: data link type LINUX_SLL2
tcpdump: listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 2621
44 bytes
10 packets captured
11 packets received by filter
0 packets dropped by kernel
root@balaji:/home/rachappa#
```

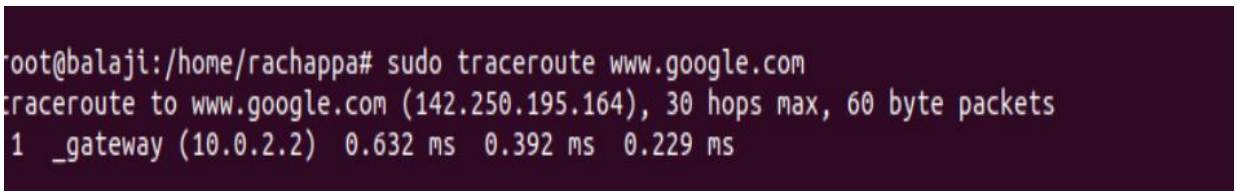
## Task 5: Perform Traceroute checks

**Step 1:** Run the traceroute using the following command.

**sudo traceroute www.google.com**




**Step 2:** Analyze destination address of google.com and no. of hops




**Step 3:** To speed up the process, you can disable the mapping of IP addresses with hostnames by using the *-n* option

**sudo traceroute -n www.google.com**

```
oot@balaji:/home/rachappa# sudo traceroute -n www.google.com
raceroute to www.google.com (142.250.195.164), 30 hops max, 60 byte packets
1  10.0.2.2  0.583 ms  0.498 ms  0.453 ms
2  * * *
3  * * *
4  * * *
5  * * *
6  * * *
7  * * *
8  * * *
9  * * *
0  * * *
1  * * *
2  * * *
3  * * *
4  * * *
5  * * *
6  * * *
7  * * *
8  * * *
9  * * *
0  * * *
1  * * *
2  * * *
3  *
```

**Step 4:** The -I option is necessary so that the traceroute uses ICMP.

**sudo traceroute -I www.google.com**

```
root@balaji:/home/rachappa# sudo traceroute -i  www.google.com
Specify "host" missing argument.
root@balaji:/home/rachappa# sudo traceroute -I  www.google.com
traceroute to www.google.com (142.250.195.164), 30 hops max, 60 byte packets
 1  _gateway (10.0.2.2)  0.427 ms  0.188 ms  0.360 ms
 2  192.168.0.1 (192.168.0.1)  5.538 ms  5.331 ms  5.125 ms
 3  10.240.0.1 (10.240.0.1)  13.900 ms  13.689 ms  13.483 ms
 4  49.205.72.35.actcorp.in (49.205.72.35)  13.229 ms  12.974 ms  12.675 ms
 5  * * *
 6  * 10.248.5.23 (10.248.5.23)  7.553 ms  7.536 ms
 7  49.205.72.39.actcorp.in (49.205.72.39)  11.687 ms  8.581 ms  8.484 ms
 8  72.14.243.242 (72.14.243.242)  20.412 ms  19.974 ms  20.215 ms
 9  108.170.227.7 (108.170.227.7)  12.466 ms  12.174 ms  12.418 ms
10  142.251.55.91 (142.251.55.91)  11.365 ms  11.495 ms  12.214 ms
11  maa03s41-in-f4.1e100.net (142.250.195.164)  16.899 ms  11.622 ms  11.350 ms
root@balaji:/home/rachappa#
```

**Step 5:** By default, traceroute uses icmp (ping) packets. If you'd rather test a TCP connection to gather data more relevant to web server, you can use the -T flag.

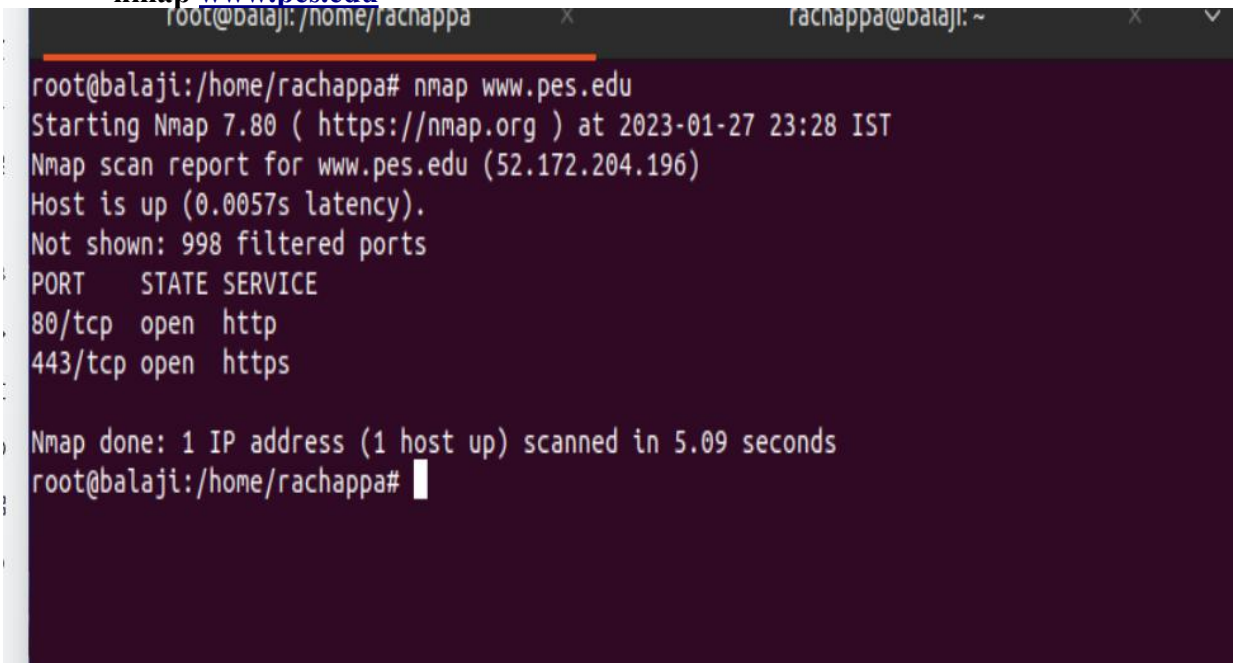
**sudo traceroute -T www.google.com**

```
root@balaji:/home/rachappa# sudo traceroute -T  www.google.com
traceroute to www.google.com (142.250.195.164), 30 hops max, 60 byte packets
 1  _gateway (10.0.2.2)  0.897 ms  0.840 ms  0.780 ms
 2  maa03s41-in-f4.1e100.net (142.250.195.164)  15.437 ms  16.198 ms  16.143 ms
root@balaji:/home/rachappa#
```

## Task 6: Explore an entire network for information (Nmap)

**Step 1:** You can scan a host using its host name or IP address, for instance.

**nmap www.pes.edu**




**Step 2:** Alternatively, use an IP address to scan.

**nmap 163.53.78.128**

```
Nmap done: 1 IP address (1 host up) scanned in 5.09 seconds
root@balaji:/home/rachappa# nmap 163.53.78.128
Starting Nmap 7.80 ( https://nmap.org ) at 2023-01-27 23:29 IST
Nmap scan report for 163.53.78.128
Host is up (0.0035s latency).
Not shown: 998 filtered ports
PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 4.72 seconds
root@balaji:/home/rachappa#
```

**Step 3:** Scan multiple IP address or subnet (IPv4)

**nmap 192.168.1.1 192.168.1.2 192.168.1.3**

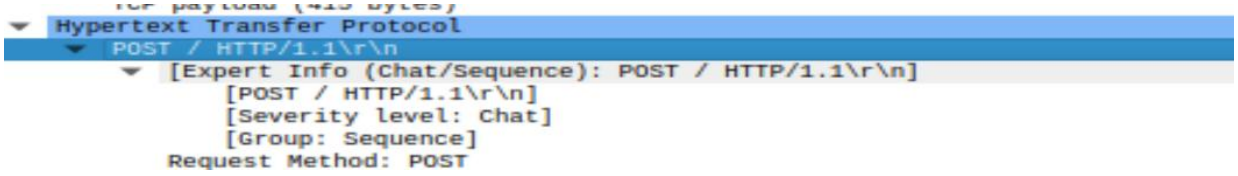
C

```
root@balaji:/home/rachappa# nmap 192.168.1.1 192.168.1.2 192.168.1.3
Starting Nmap 7.80 ( https://nmap.org ) at 2023-01-27 23:31 IST
Nmap scan report for 192.168.1.1
Host is up (0.0011s latency).
All 1000 scanned ports on 192.168.1.1 are filtered

Nmap scan report for 192.168.1.2
Host is up (0.0016s latency).
All 1000 scanned ports on 192.168.1.2 are filtered

Nmap scan report for 192.168.1.3
Host is up (0.0015s latency).
All 1000 scanned ports on 192.168.1.3 are filtered

Nmap done: 3 IP addresses (3 hosts up) scanned in 28.74 seconds
root@balaji:/home/rachappa#
```

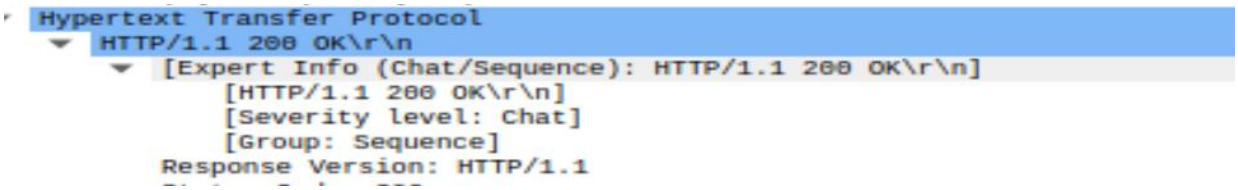
**Questions on above observations:**

1) Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server?

CLIENT

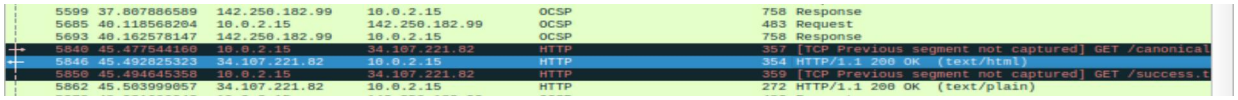



SERVER




SO they both run on http 1.1 version

2) When was the HTML file that you are retrieving last modified at the server?




3) How to tell ping to exit after a specified number of ECHO_REQUEST packets?

ping -c 5 example.com

Where -c is use to count the number of ping

```
ping -c 5 example.com
```

4) How will you identify remote host apps and OS?

Using nmap