Submitted by :

NAME : PREM SAGAR J S

SRN : PES1UG20CS825

SEC : 'H'

Week #1

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

1. Wireshark

2. Tcpdump

3. Ping

4. Traceroute

5. Nmap

Note : Operating System Being used is Kali Linux.

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:

IP Address Table :

Interface name	IP address (IPv4 / IPv6)	MAC address
eth0	—	c8:d9:d2:ec:e2:cb
lo	127.0.0.1	—
wlan0	192.168.69.63	74:40:bb:4c:61:23

Step 2: To assign an IP address to an interface.

Section H and Serial No 61 :

```
prem@Kraken: ~
┌──(prem㉿Kraken)-[~]
└─$ sudo ifconfig eth0 10.0.8.61 netmask 255.255.255.0

┌──(prem㉿Kraken)-[~]
└─$ ifconfig eth0
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 10.0.8.61  netmask 255.255.255.0  broadcast 10.0.8.255
        ether c8:d9:d2:ec:e2:cb  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

┌──(prem㉿Kraken)-[~]
└─$
```

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

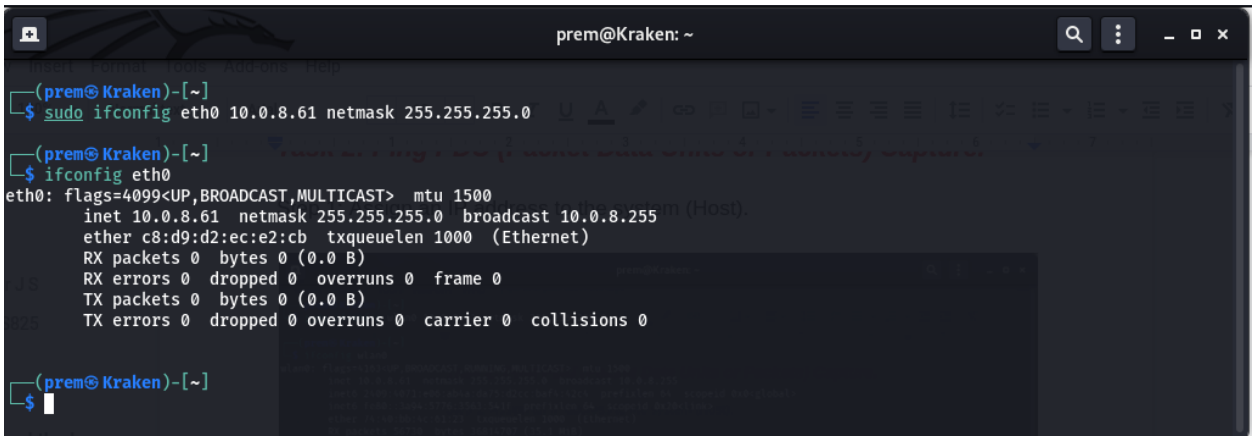
```
prem@Kraken: ~
┌──(prem㉿Kraken)-[~]
└─$ sudo ifconfig wlan0 down

┌──(prem㉿Kraken)-[~]
└─$ ifconfig -s
Iface      MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0      1500        0      0      0      0         0      0      0      0 BMU
lo       65536      174      0      0      0       174      0      0      0 LRU

┌──(prem㉿Kraken)-[~]
└─$ sudo ifconfig wlan0 up

┌──(prem㉿Kraken)-[~]
└─$ ifconfig -s
Iface      MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0      1500        0      0      0      0         0      0      0      0 BMU
lo       65536      174      0      0      0       174      0      0      0 LRU
wlan0     1500    50726      0      0      0     46069      0      0      0 BMU

┌──(prem㉿Kraken)-[~]
└─$
```

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

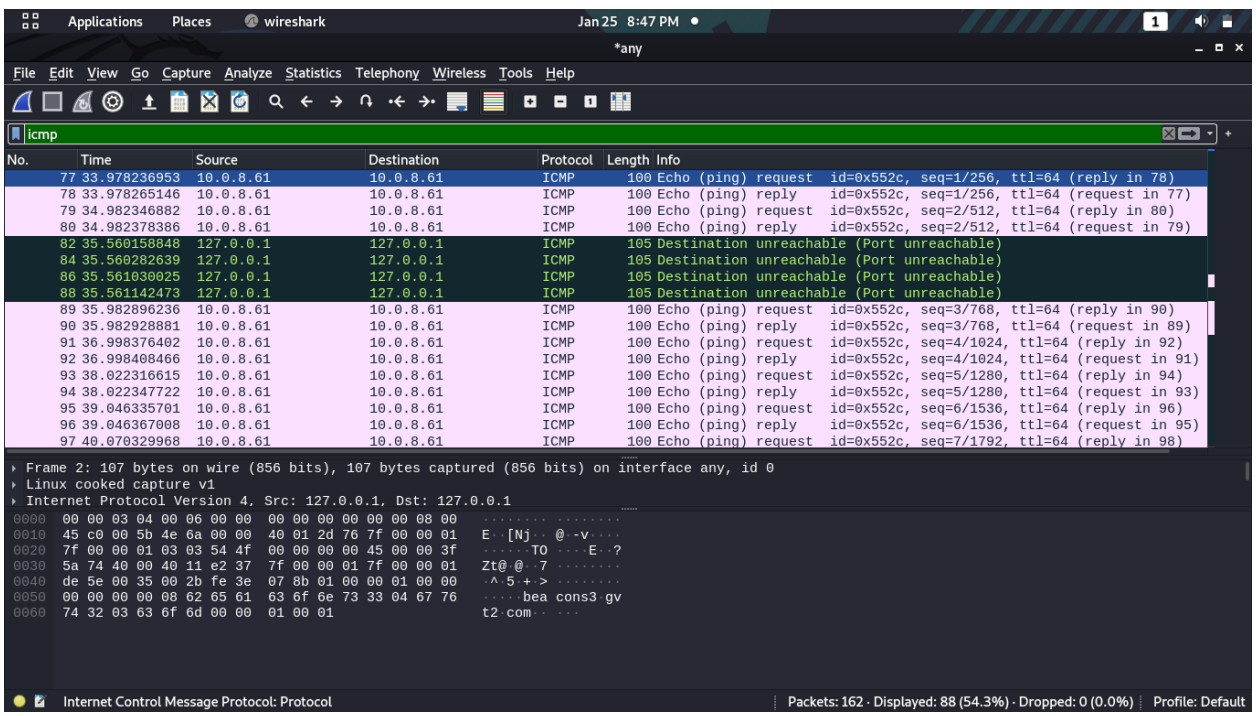
```
prem@Kraken: ~
┌──(prem㉿Kraken)-[~]
└─$ ip neigh
192.168.69.147 dev wlan0 lladdr 86:8e:3c:00:66:f5 DELAY
192.168.69.225 dev wlan0 lladdr 30:32:35:51:03:f6 REACHABLE
2409:4071:e06:ab4a:da75:d2cc:baf4:42c4 dev wlan0  FAILED
2409:4071:e06:ab4a::42 dev wlan0 lladdr 86:8e:3c:00:66:f5 router STALE
fe80::848e:3cff:fe00:66f5 dev wlan0 lladdr 86:8e:3c:00:66:f5 router REACHABLE

┌──(prem㉿Kraken)-[~]
└─$
```

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

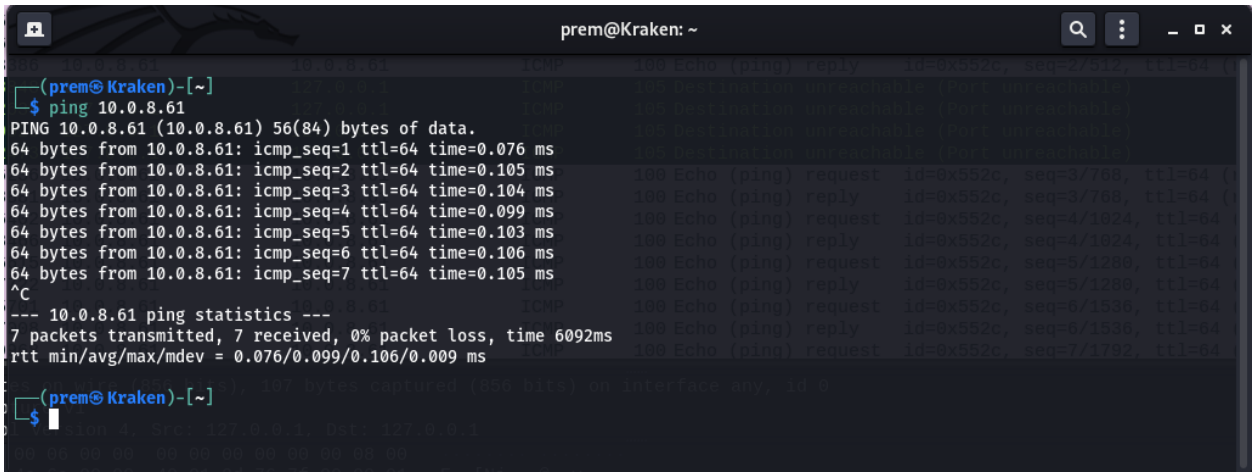
Step 1: Assign an IP address to the system (Host).



Step 2: Launch Wireshark and select ‘any’ interface



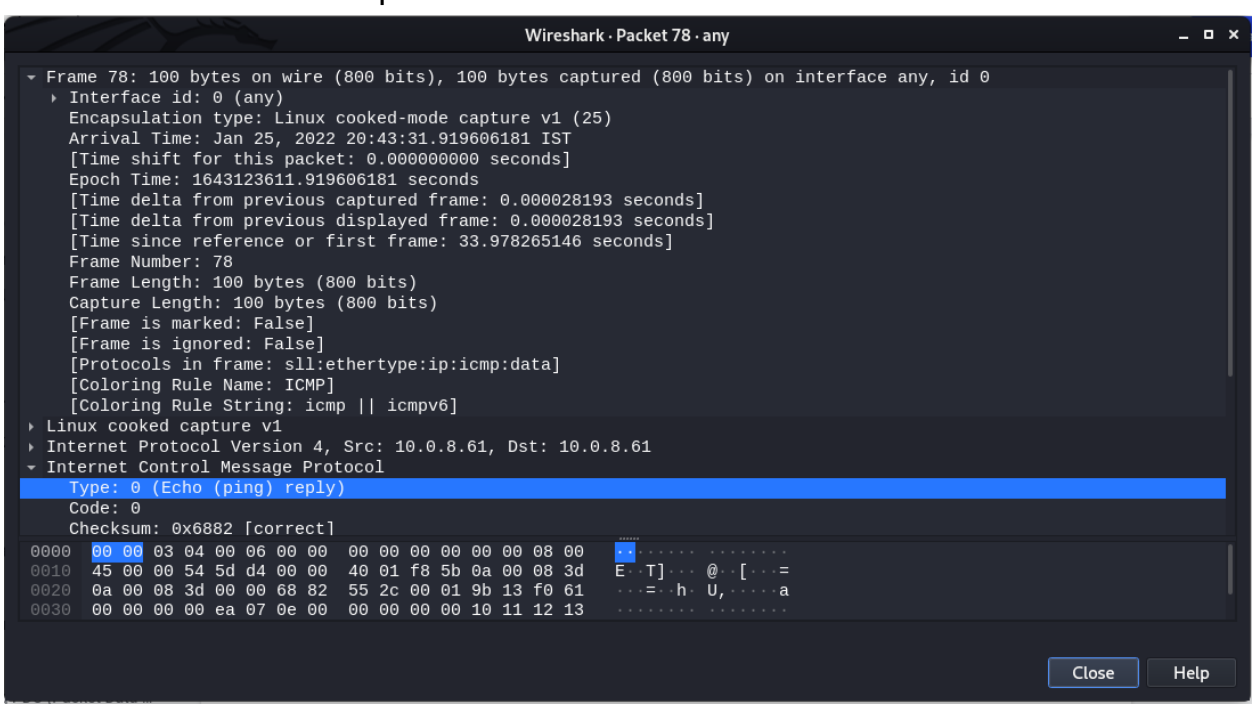
Step 3: In terminal, type ping 10.0.your_section.your_sno
Observations to be made



Step 4: Analyze the following in Terminal

- TTL = 64
- Protocol used by ping = ICMP
- Time = 0.076 to 0.105 ms

Step 5: Analyze the following in Wireshark 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.

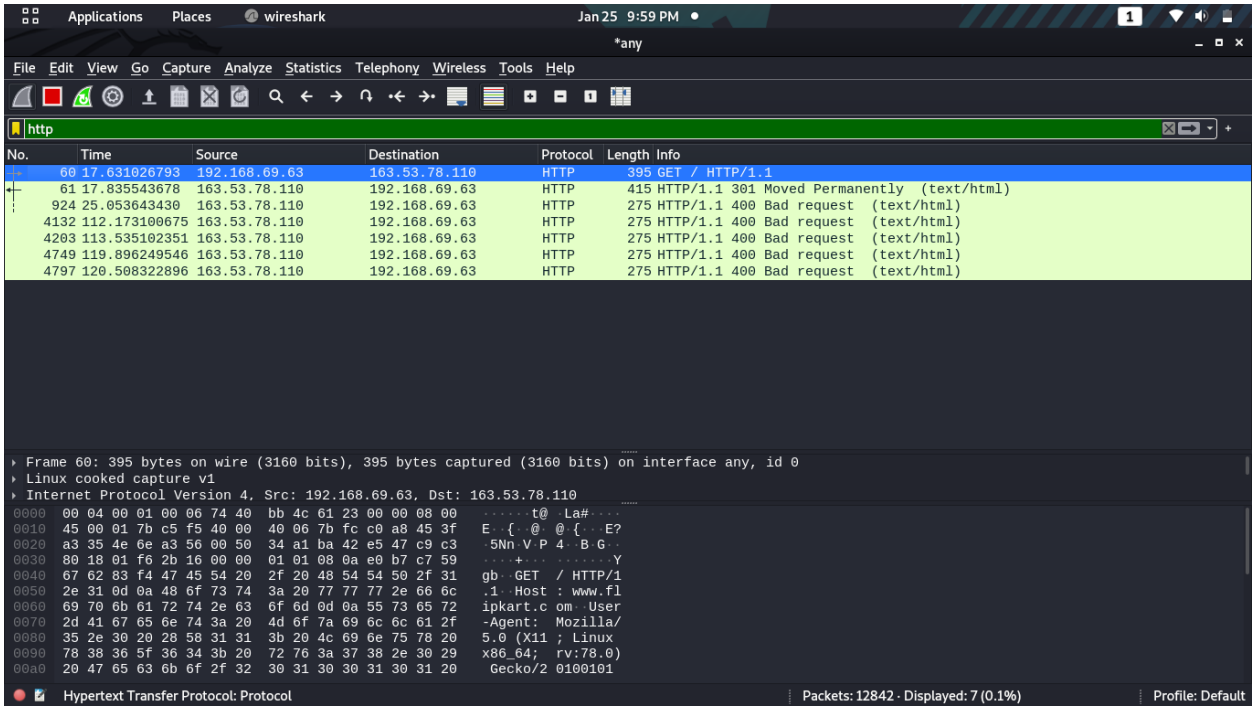
Details	**First Echo Request**	**First Echo Reply**
Frame Number	77	78
Source IP address	10.0.8.61	10.0.8.61
Destination IP address	10.0.8.61	10.0.8.61
ICMP Type Value	Type: 8 (Echo (ping) request)	Type: 0 (Echo (ping) reply)
ICMP Code Value	Code: 0	Code: 0
Source Ethernet Address	—	—
Destination Ethernet Address	—	—
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



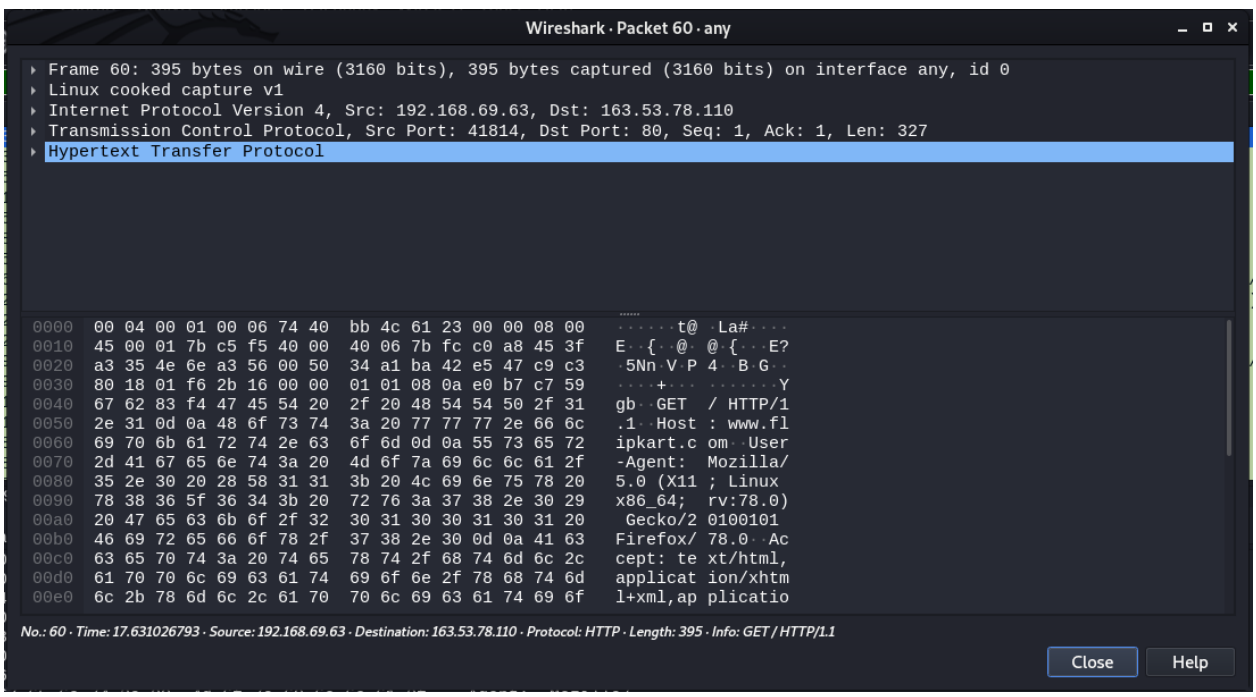
Step 3: Analyze the first (interaction of host to the web server) and second frame (response of server to the client).

By analyzing the filtered frames, complete the table below:

Details	**First Echo Request**	**First Echo Reply**
Frame Number	60	61
Source Port	41814	80
Destination Port	80	41814
Source IP address	192.168.69.63	163.53.78.110
Destination IP address	163.53.78.110	192.168.69.63
Source Ethernet Address	74:40:bb:4c:61:23	86:8e:3c:00:66:f5
Destination Ethernet Address	86:8e:3c:00:66:f5	74:40:bb:4c:61:23

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

HTTP Request		HTTP Response	
Get	GET / HTTP/1.1\r\n	Server	nginx
Host	www.flipkart.com	Content-Type	text/html
User-Agent	Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0	Date	Tue, 25 Jan 2022 16:23:20 GMT
Accept-Language	en-US,en;q=0.5	Location	https://www.flipkart.com/
Accept-Encoding	gzip, deflate	Content-Length	178
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

```
┌──(prem㉿Kraken)-[~]
└─$ sudo tcpdump -D
1.wlan0 [Up, Running, Wireless, Associated]
2.any (Pseudo-device that captures on all interfaces) [Up, Running]
3.lo [Up, Running, Loopback]
4.eth0 [Up, Disconnected]
5.bluetooth0 (Bluetooth adapter number 0) [Wireless, Association status unknown]
6.bluetooth-monitor (Bluetooth Linux Monitor) [Wireless]
7.nflog (Linux netfilter log (NFLOG) interface) [none]
8.nfqueue (Linux netfilter queue (NFQUEUE) interface) [none]
9.dbus-system (D-Bus system bus) [none]
10.dbus-session (D-Bus session bus) [none]

┌──(prem㉿Kraken)-[~]
└─$
```

Step 2: Capture all packets in any interface by running this command:
sudo tcpdump -i any

```
┌──(prem㉿Kraken)-[~]
└─$ ping www.google.co.in -4
PING  (142.250.182.99) 56(84) bytes of data.
64 bytes from maa05s21-in-f3.1e100.net (142.250.182.99): icmp_seq=7 ttl=111 time=1461
ms
64 bytes from maa05s21-in-f3.1e100.net (142.250.182.99): icmp_seq=8 ttl=111 time=1473
ms
64 bytes from maa05s21-in-f3.1e100.net (142.250.182.99): icmp_seq=10 ttl=111 time=1540
 ms
64 bytes from maa05s21-in-f3.1e100.net (142.250.182.99): icmp_seq=14 ttl=111 time=1534
 ms
64 bytes from maa05s21-in-f3.1e100.net (142.250.182.99): icmp_seq=17 ttl=111 time=1470
 ms
64 bytes from maa05s21-in-f3.1e100.net (142.250.182.99): icmp_seq=19 ttl=111 time=1332
 ms
64 bytes from maa05s21-in-f3.1e100.net (142.250.182.99): icmp_seq=21 ttl=111 time=1348
 ms
^C
---  ping statistics ---
23 packets transmitted, 7 received, 69.5652% packet loss, time 33376ms
rtt min/avg/max/mdev = 1332.316/1451.286/1540.257/76.030 ms, pipe 2

┌──(prem㉿Kraken)-[~]
└─$
```

```
┌──(prem㉿Kraken)-[~]
└─$ sudo tcpdump -i any
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
22:43:23.983789 wlan0 In  IP maa03s45-in-f3.1e100.net.https > Kraken.39921: UDP, lengt
h 33
22:43:23.984173 wlan0 Out IP Kraken.39921 > maa03s45-in-f3.1e100.net.https: UDP, lengt
h 1250
22:43:24.031002 wlan0 Out IP Kraken.40027 > _gateway.domain: 11888+ PTR? 63.69.168.192
.in-addr.arpa. (44)
22:43:24.054087 wlan0 Out IP Kraken.39921 > maa03s45-in-f3.1e100.net.https: UDP, lengt
h 1250
22:43:24.393487 wlan0 In  IP maa03s45-in-f3.1e100.net.https > Kraken.39921: UDP, lengt
h 36
22:43:24.393812 wlan0 Out IP Kraken.39921 > maa03s45-in-f3.1e100.net.https: UDP, lengt
h 1250
22:43:24.415383 wlan0 Out IP6 Kraken.50910 > maa05s14-in-x0e.1e100.net.https: UDP, len
gth 1230
22:43:24.463024 wlan0 Out IP Kraken.39921 > maa03s45-in-f3.1e100.net.https: UDP, lengt
h 1250
22:43:24.507568 wlan0 Out IP Kraken.36174 > maa05s21-in-f14.1e100.net.https: Flags [P.
], seq 1473457129:1473457193, ack 1995280514, win 501, options [nop,nop,TS val 1756922
219 ecr 3593528158], length 64
22:43:24.893601 wlan0 In  IP maa03s45-in-f3.1e100.net.https > Kraken.39921: UDP, lengt
h 38
22:43:24.894001 wlan0 Out IP Kraken.39921 > maa03s45-in-f3.1e100.net.https: UDP, lengt
h 1250
22:43:24.962204 wlan0 Out IP Kraken.39921 > maa03s45-in-f3.1e100.net.https: UDP, lengt
h 1250
22:43:25.212663 wlan0 In  IP maa05s21-in-f14.1e100.net.https > Kraken.36176: Flags [F.
], seq 140017087, ack 3801213609, win 256, options [nop,nop,TS val 1856697610 ecr 1756
911660], length 0
22:43:25.212989 wlan0 Out IP Kraken.36176 > maa05s21-in-f14.1e100.net.https: Flags [F.
], seq 601, ack 1, win 502, options [nop,nop,TS val 1756922924 ecr 1856697610], length
```

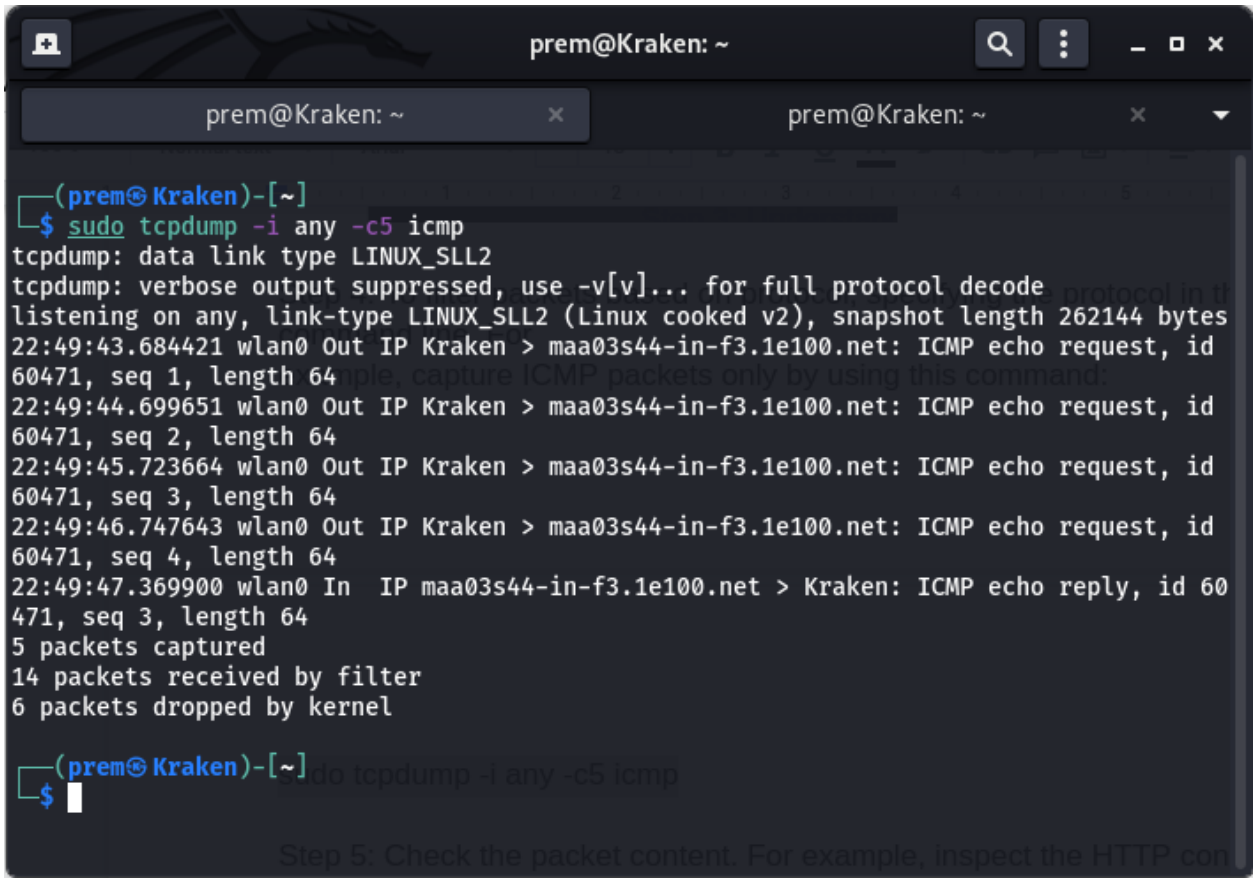
Step 3: Understand the output format.

```
26 packets captured
1285 packets received by filter
1243 packets dropped by kernel
```

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

```
┌──(prem㉿Kraken)-[~]
└─$ 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
22:51:42.145486 wlan0 Out IP 192.168.69.63.55294 > 192.124.249.41.80: Flags [S], seq 4
155706268, win 64240, options [mss 1460,sackOK,TS val 1894899468 ecr 0,nop,wscale 7],
length 0
E..<..@.@.....E?.|.)...P............a..........
p...........
22:51:42.396015 wlan0 Out IP 192.168.69.63.55296 > 192.124.249.41.80: Flags [S], seq 2
898817857, win 64240, options [mss 1460,sackOK,TS val 1894899718 ecr 0,nop,wscale 7],
length 0
E..<.b@.@.....E?.|.)...P..sA........D..........
p...........
22:51:42.474403 wlan0 Out IP 192.168.69.63.55298 > 192.124.249.41.80: Flags [S], seq 1
377839248, win 64240, options [mss 1460,sackOK,TS val 1894899797 ecr 0,nop,wscale 7],
length 0
E..<..@.@..@..E?.|.)...PR (..........6.........
p..U........
```

```
22:51:42.692787 wlan0 Out IP 192.168.69.63.55296 > 192.124.249.41.80: Flags [P.], seq
1:364, ack 1, win 502, options [nop,nop,TS val 1894900015 ecr 3590218806], length 363:
 HTTP: POST / HTTP/1.1
E....d@.@..f..E?.|.)...P..sB.eTo.....K.....
p../..d6POST / HTTP/1.1
Host: ocsp.godaddy.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/ocsp-request
Content-Length: 76
Connection: keep-alive

0J0H0F0D0B0     ..+...........lkv...8..f..R34N..@..'..4.0.3..l...,...  ..V...0.!
10 packets captured
11 packets received by filter
0 packets dropped by kernel

┌──(prem㉿Kraken)-[~]
└─$
```

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

```
┌──(prem㉿Kraken)-[~]
└─$ 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 leng
th 262144 bytes
10 packets captured
10 packets received by filter
0 packets dropped by kernel

┌──(prem㉿Kraken)-[~]
└─$ ls
Desktop    Downloads  Pictures  Templates  webserver.pcap
Documents  Music      Public    Videos

┌──(prem㉿Kraken)-[~]
└─$
```

Task 5: Perform Traceroute checks

Step 1: Run the traceroute using the following command.
sudo traceroute www.google.com

```
┌──(prem㉿Kraken)-[~]
└─$ sudo traceroute www.google.com
traceroute to www.google.com (142.250.205.228), 30 hops max, 60 byte packets
 1  _gateway (192.168.69.147)  3.602 ms  3.556 ms  3.516 ms
 2  * * *
 3  56.8.40.45 (56.8.40.45)  834.716 ms 56.8.40.25 (56.8.40.25)  837.616 ms 56.8.40.33 (56.8.40.33)  837.623 ms
 4  192.168.65.248 (192.168.65.248)  943.411 ms  943.427 ms 192.168.65.250 (192.168.65.250)  943.400 ms
 5  192.168.65.253 (192.168.65.253)  943.374 ms 192.168.65.251 (192.168.65.251)  943.397 ms 192.168.65.253 (192.168
.65.253)  958.712 ms
 6  172.26.74.24 (172.26.74.24)  978.301 ms  990.701 ms  1005.196 ms
 7  172.26.77.242 (172.26.77.242)  1127.341 ms * *
 8  * * *
 9  * * *
10  * * *
11  * * 72.14.196.126 (72.14.196.126)  198.634 ms
12  * * *
13  142.251.60.185 (142.251.60.185)  204.222 ms  204.038 ms 142.251.55.224 (142.251.55.224)  287.778 ms
14  108.170.253.103 (108.170.253.103)  287.764 ms 142.251.60.187 (142.251.60.187)  287.728 ms maa05s28-in-f4.1e100.
net (142.250.205.228)  287.630 ms

┌──(prem㉿Kraken)-[~]
└─$
```

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

Destination address : 142.250..205.228 , no of hops : 14

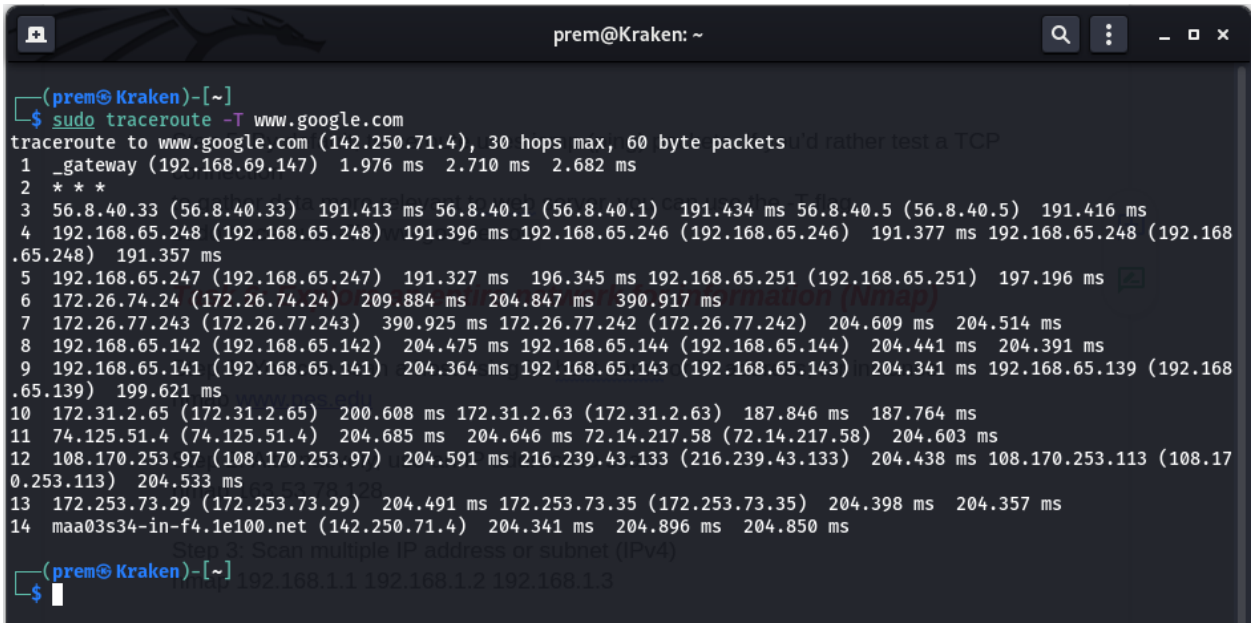
Step 3: To speed up the proess, you can disable the mapping of IP addresses with hostnamesby using the -n option
sudo traceroute -n www.google.com

```
┌──(prem㉿Kraken)-[~]
└─$ sudo traceroute -n www.google.com
traceroute to www.google.com (142.250.71.4), 30 hops max, 60 byte packets
 1  192.168.69.147  2.444 ms  2.351 ms  2.271 ms
 2  * * *
 3  56.8.40.33  830.582 ms 56.8.40.49  830.534 ms 56.8.40.1  830.474 ms
 4  192.168.65.250  830.410 ms 192.168.65.246  830.336 ms 192.168.65.250  830.279 ms
 5  192.168.65.251  830.191 ms 192.168.65.249  830.110 ms 192.168.65.253  830.031 ms
 6  172.26.74.24  829.952 ms  830.363 ms  830.277 ms
 7  172.26.77.243  848.684 ms  101.282 ms 172.26.77.242  204.389 ms
 8  192.168.65.142  204.394 ms 192.168.65.138  204.364 ms 192.168.65.142  204.306 ms
 9  192.168.65.141  205.961 ms  221.018 ms 192.168.65.139  237.891 ms
10  172.31.2.63  408.929 ms  408.930 ms 172.31.2.65  407.477 ms
11  72.14.217.58  407.451 ms 74.125.50.202  388.975 ms 72.14.196.126  307.968 ms
12  * 108.170.253.113  216.242 ms *
13  172.253.73.29  370.172 ms 74.125.242.129  368.694 ms 216.239.47.142  353.466 ms
14  74.125.242.154  336.646 ms 142.250.71.4  165.490 ms 74.125.242.147  165.844 ms

┌──(prem㉿Kraken)-[~]
└─$
```

Step 4: The -I option is necessary so that the traceroute uses ICMP.
sudo traceroute -I www.google.com

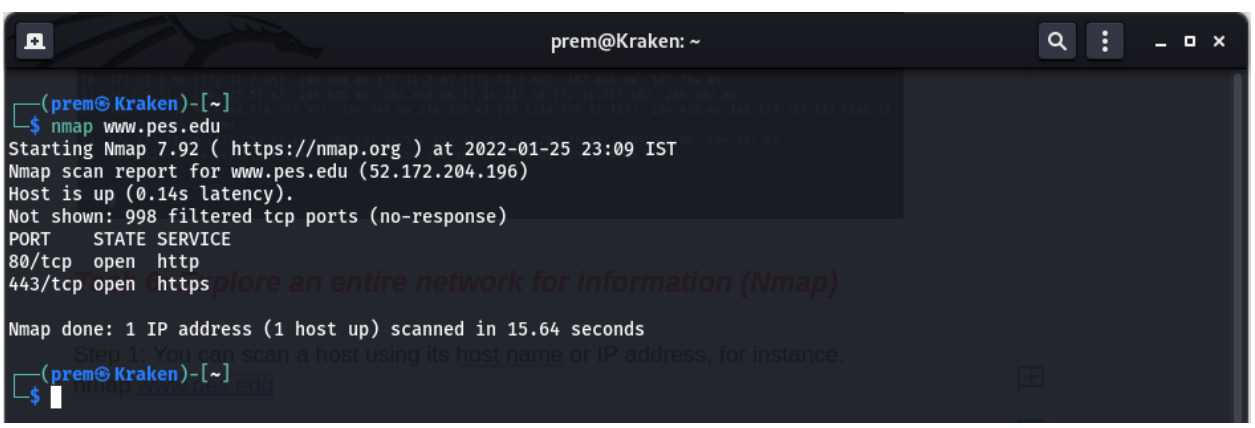
```
┌──(prem㉿Kraken)-[~]
└─$ sudo traceroute -I www.google.com
traceroute to www.google.com (142.250.71.4), 30 hops max, 60 byte packets
 1  _gateway (192.168.69.147)  2.166 ms  2.718 ms  2.714 ms
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  172.26.74.24 (172.26.74.24)  252.028 ms  246.862 ms  246.812 ms
 7  172.26.77.243 (172.26.77.243)  246.802 ms  204.848 ms  204.848 ms
 8  192.168.65.138 (192.168.65.138)  204.841 ms 192.168.65.140 (192.168.65.140)  204.836 ms 192.168.65.138 (192.168
.65.138)  172.130 ms
 9  192.168.65.141 (192.168.65.141)  172.092 ms  172.078 ms  172.064 ms
10  172.31.2.65 (172.31.2.65)  237.134 ms  237.126 ms  237.123 ms
11  74.125.50.202 (74.125.50.202)  237.118 ms  204.717 ms  204.679 ms
12  216.239.43.133 (216.239.43.133)  204.615 ms  204.650 ms  204.404 ms
13  172.253.73.35 (172.253.73.35)  204.390 ms  204.372 ms  204.359 ms
14  maa03s34-in-f4.1e100.net (142.250.71.4)  204.711 ms  204.699 ms  204.692 ms

┌──(prem㉿Kraken)-[~]
└─$
```

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



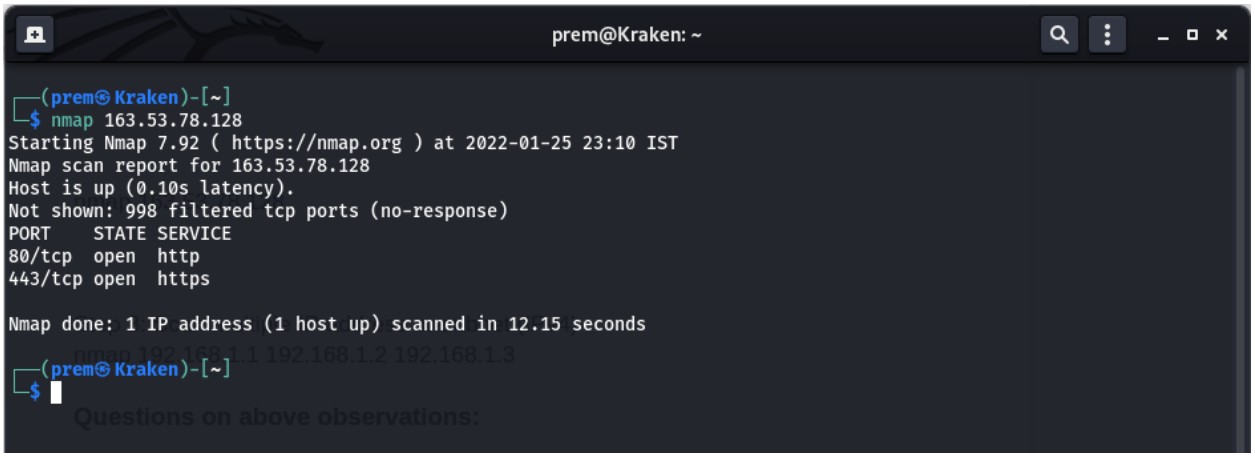
Task 6: Explore an entire network for information (Nmap)

Step 1: You can scan a host using its host name or IP address, for instance.
nmap www.pes.edu

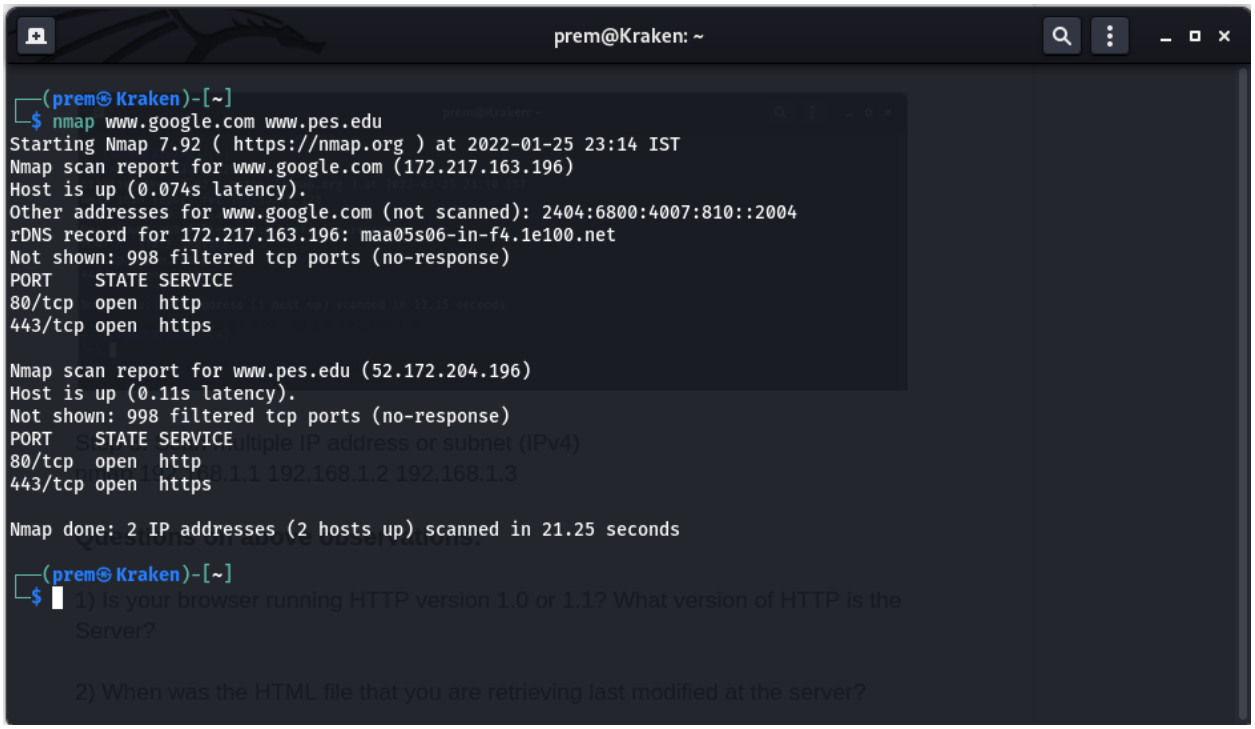


Step 2: Alternatively, use an IP address to scan.

nmap 163.53.78.128



Step 3: Scan multiple IP address or subnet (IPv4)
nmap 192.168.1.1 192.168.1.2 192.168.1.3



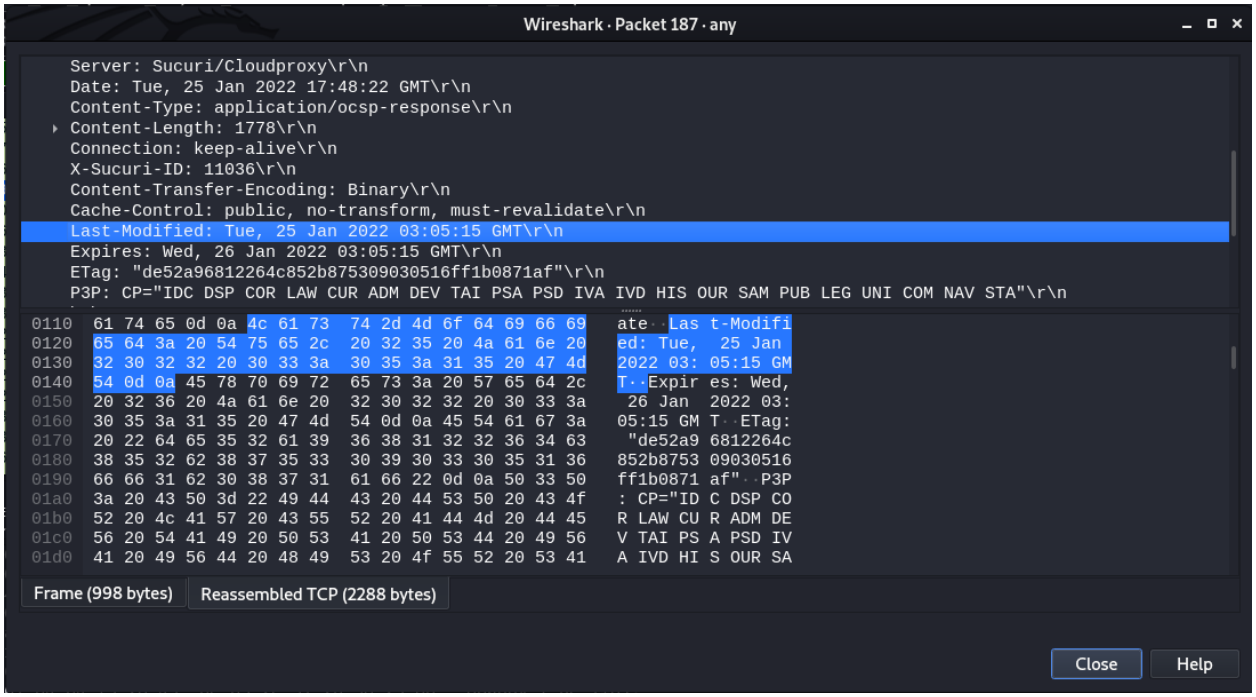
Questions on above observations:

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

- HTTP version 1.1

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

- Tue, 25 Jan 2022 03:05:15 GMT



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

- By specifying -c count or –w deadline

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

- Using Nmap tool