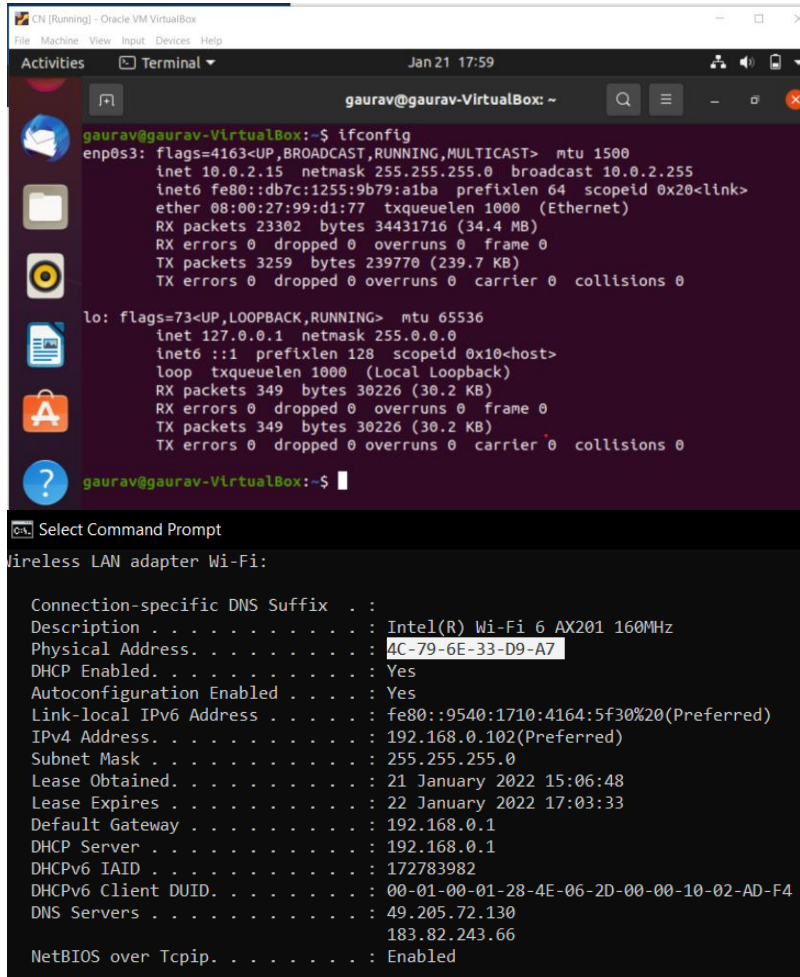


Name: GAURAV MAHAJAN	SRN: PES1UG20CS150	Section: C
	Date: 19-01-2022	WEEK 1

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



```

gaurav@gaurav-VirtualBox:~$ 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::db7c:1255:9b79:a1ba prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:99:d1:77 txqueuelen 1000 (Ethernet)
    RX packets 23302 bytes 34431716 (34.4 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 3259 bytes 239770 (239.7 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 349 bytes 30226 (30.2 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 349 bytes 30226 (30.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

gaurav@gaurav-VirtualBox:~$
  
```

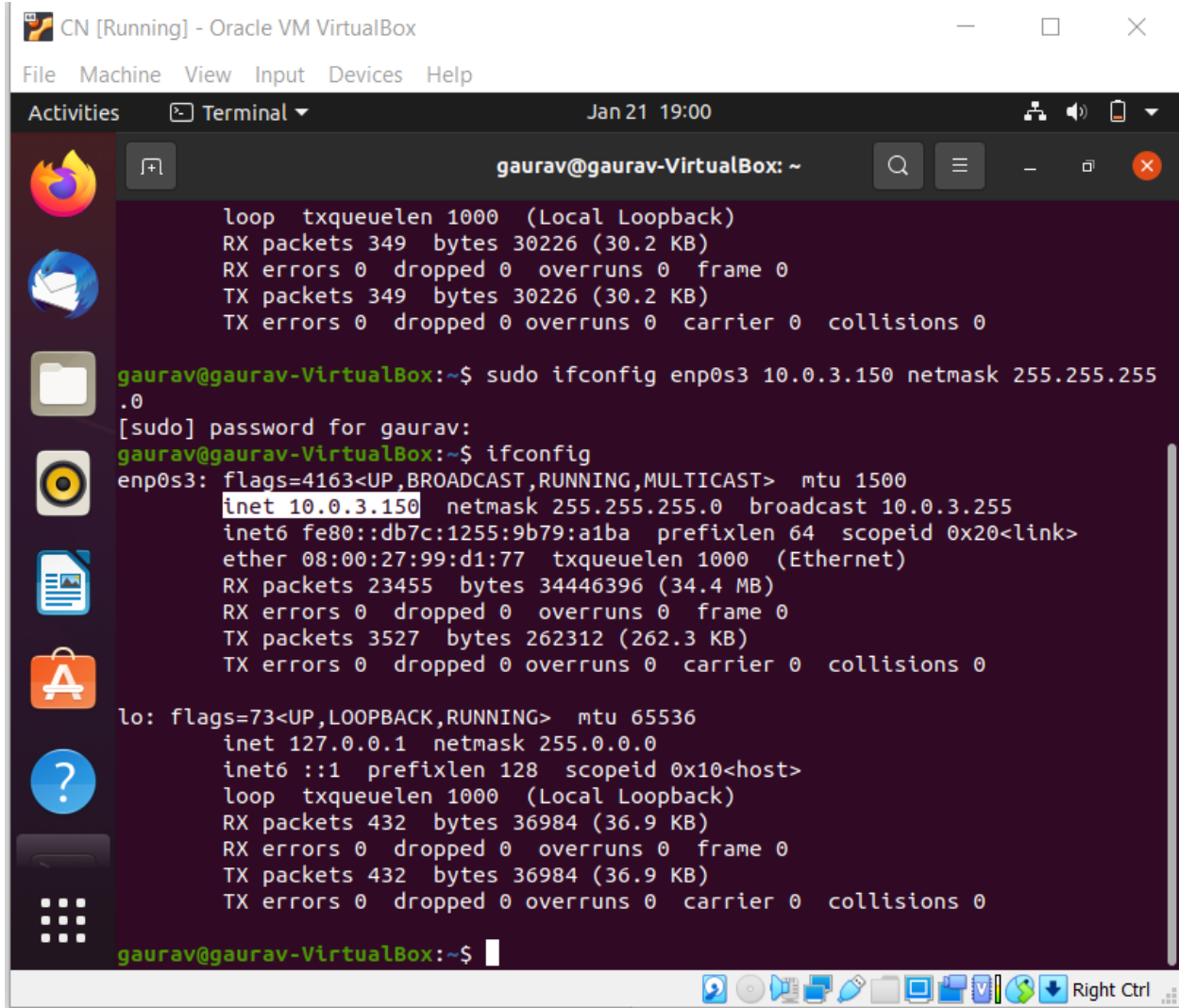
```

Select Command Prompt
Wireless LAN adapter Wi-Fi:

Connection-specific DNS Suffix . : 
Description . . . . . : Intel(R) Wi-Fi 6 AX201 160MHz
Physical Address. . . . . : 4C-79-6E-33-D9-A7
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . . : Yes
Link-local IPv6 Address . . . . : fe80::9540:1710:4164:5f30%20(Preferred)
IPv4 Address. . . . . : 192.168.0.102(Preferred)
Subnet Mask . . . . . : 255.255.255.0
Lease Obtained. . . . . : 21 January 2022 15:06:48
Lease Expires . . . . . : 22 January 2022 17:03:33
Default Gateway . . . . . : 192.168.0.1
DHCP Server . . . . . : 192.168.0.1
DHCPv6 IAID . . . . . : 172783982
DHCPv6 Client DUID. . . . . : 00-01-00-01-28-4E-06-2D-00-00-10-02-AD-F4
DNS Servers . . . . . : 49.205.72.130
                        183.82.243.66
NetBIOS over Tcpip. . . . . : Enabled
  
```

INTERFACE	IPV4	MAC ADDRESS
enp0s3	10.0.2.15	08:00:27:99:d1:77
Wi-fi	192.168.0.10	4C:79:6E:33:D9:A7

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**



```
gaurav@gaurav-VirtualBox: ~  
loop txqueuelen 1000 (Local Loopback)  
RX packets 349 bytes 30226 (30.2 KB)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 349 bytes 30226 (30.2 KB)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
gaurav@gaurav-VirtualBox:~$ sudo ifconfig enp0s3 10.0.3.150 netmask 255.255.255  
.0  
[sudo] password for gaurav:  
gaurav@gaurav-VirtualBox:~$ ifconfig  
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
inet 10.0.3.150 netmask 255.255.255.0 broadcast 10.0.3.255  
inet6 fe80::db7c:1255:9b79:a1ba prefixlen 64 scopeid 0x20<link>  
ether 08:00:27:99:d1:77 txqueuelen 1000 (Ethernet)  
RX packets 23455 bytes 34446396 (34.4 MB)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 3527 bytes 262312 (262.3 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 432 bytes 36984 (36.9 KB)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 432 bytes 36984 (36.9 KB)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
gaurav@gaurav-VirtualBox:~$
```

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

```
C:\Users\Gaurav>ping 192.168.0.1

Pinging 192.168.0.1 with 32 bytes of data:
Reply from 192.168.0.1: bytes=32 time=7ms TTL=64
Reply from 192.168.0.1: bytes=32 time=59ms TTL=64
Reply from 192.168.0.1: bytes=32 time=10ms TTL=64
Reply from 192.168.0.1: bytes=32 time=12ms TTL=64

Ping statistics for 192.168.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss)
    Approximate round trip times in milli-seconds:
        Minimum = 7ms, Maximum = 59ms, Average = 22ms

C:\Users\Gaurav>
```

Capturing from Wi-Fi

No.	Time	Source	Destination	Protocol	Length	Info
21	6.477510	192.168.0.102	192.168.0.1	ICMP	74	Echo (ping) request id=0x0001, seq=107/27392, ttl=64 (reply in 22)
22	6.484387	192.168.0.1	192.168.0.102	ICMP	74	Echo (ping) reply id=0x0001, seq=107/27392, ttl=64 (request in 21)
29	7.483781	192.168.0.102	192.168.0.1	ICMP	74	Echo (ping) request id=0x0001, seq=108/27648, ttl=64 (reply in 30)
30	7.543042	192.168.0.1	192.168.0.102	ICMP	74	Echo (ping) reply id=0x0001, seq=108/27648, ttl=64 (request in 29)
33	8.493986	192.168.0.102	192.168.0.1	ICMP	74	Echo (ping) request id=0x0001, seq=109/27904, ttl=64 (reply in 34)
34	8.504421	192.168.0.1	192.168.0.102	ICMP	74	Echo (ping) reply id=0x0001, seq=109/27904, ttl=64 (request in 33)
37	9.505310	192.168.0.102	192.168.0.1	ICMP	74	Echo (ping) request id=0x0001, seq=110/28160, ttl=64 (reply in 38)
38	9.517504	192.168.0.1	192.168.0.102	ICMP	74	Echo (ping) reply id=0x0001, seq=110/28160, ttl=64 (request in 37)

- TTL =64
- Protocol used by ping =ICMP
- Time =22ms(AVERAGE)

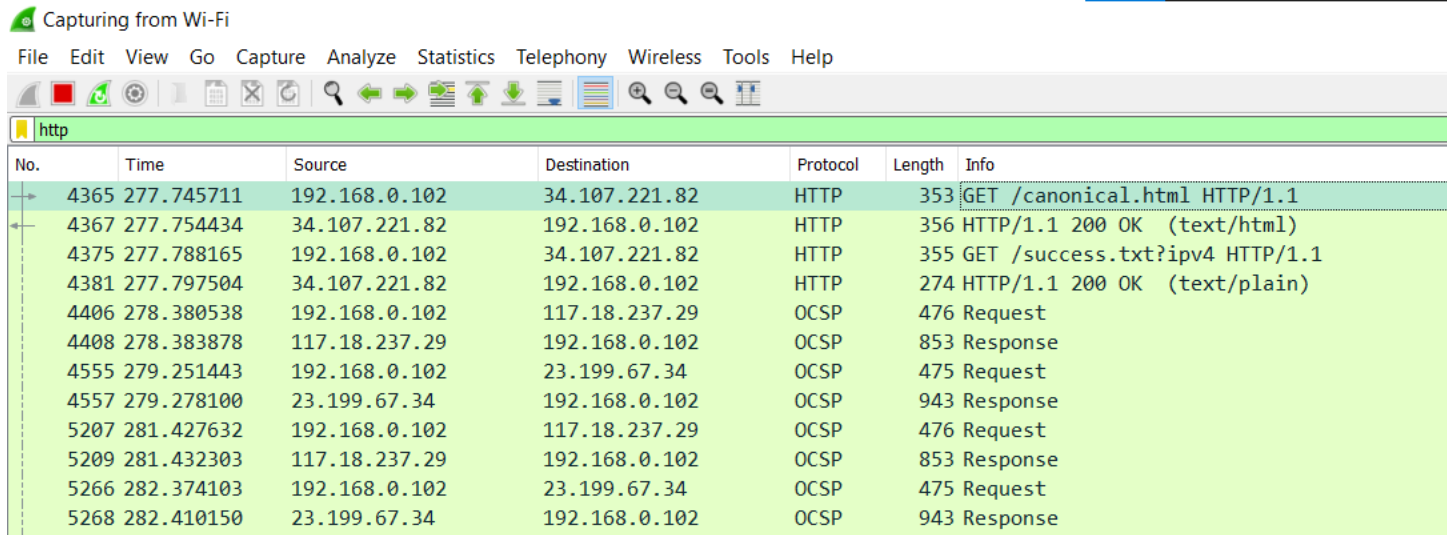
Details	First Echo Request	First Echo Reply
Frame Number	21	22
Source IP address	192.168.0.102	192.168.0.1
Destination IP address	192.168.0.1	192.168.0.102
ICMP Type Value	8	0
ICMP Code Value	0	0
Source Ethernet Address	4c:79:6e:33:d9:a7	6c:5a:b0:a7:63:fe
Destination Ethernet Address	6c:5a:b0:a7:63:fe	4c:79:6e:33:d9:a7
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



The screenshot shows the Wireshark interface with the filter 'http' applied. The packet list shows several HTTP requests and responses. The first request (frame 4365) is a GET request for /canonical.html. The second frame (4367) is the corresponding 200 OK response. Subsequent frames show more requests and responses, including a successful text/plain response (frame 4375).

No.	Time	Source	Destination	Protocol	Length	Info
4365	277.745711	192.168.0.102	34.107.221.82	HTTP	353	GET /canonical.html HTTP/1.1
4367	277.754434	34.107.221.82	192.168.0.102	HTTP	356	HTTP/1.1 200 OK (text/html)
4375	277.788165	192.168.0.102	34.107.221.82	HTTP	355	GET /success.txt?ipv4 HTTP/1.1
4381	277.797504	34.107.221.82	192.168.0.102	HTTP	274	HTTP/1.1 200 OK (text/plain)
4406	278.380538	192.168.0.102	117.18.237.29	OCSP	476	Request
4408	278.383878	117.18.237.29	192.168.0.102	OCSP	853	Response
4555	279.251443	192.168.0.102	23.199.67.34	OCSP	475	Request
4557	279.278100	23.199.67.34	192.168.0.102	OCSP	943	Response
5207	281.427632	192.168.0.102	117.18.237.29	OCSP	476	Request
5209	281.432303	117.18.237.29	192.168.0.102	OCSP	853	Response
5266	282.374103	192.168.0.102	23.199.67.34	OCSP	475	Request
5268	282.410150	23.199.67.34	192.168.0.102	OCSP	943	Response

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	4365	4367
Source Port	52541	80
Destination Port	80	64020
Source IP address	192.168.0.102	34.107.221.82
Destination IP address	34.107.221.82	192.168.0.102
Source Ethernet Address	4c:79:6e:33:d9:a7	6c:5a:b0:a7:63:fe
Destination Ethernet Address	6c:5a:b0:a7:63:fe	4c:79:6e:33:d9:a7

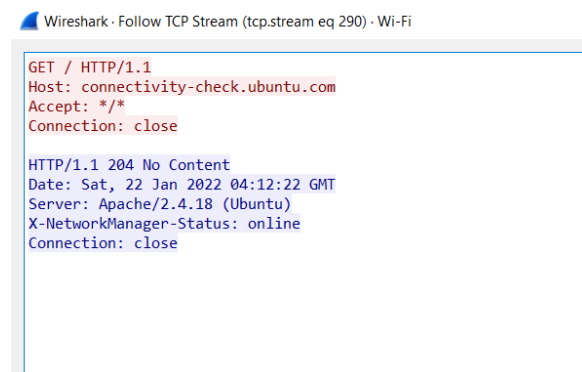
Step 4: Analyze the HTTP request and response and complete the table below.

HTTP Request		HTTP Response	
Get	/canonical.html HTTP/1.1	Server	nginx
Host	detectportal.firefox.com	Content-Type	text/html
User-Agent	Mozilla/5.0	Date	Fri,21 Jan 2022 03:34:40 GMT
Accept-Language	en-US	Location	-
Accept-Encoding	gzip,deflate	Content-Length	356
Connection	keep-alive	Connection	close

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.



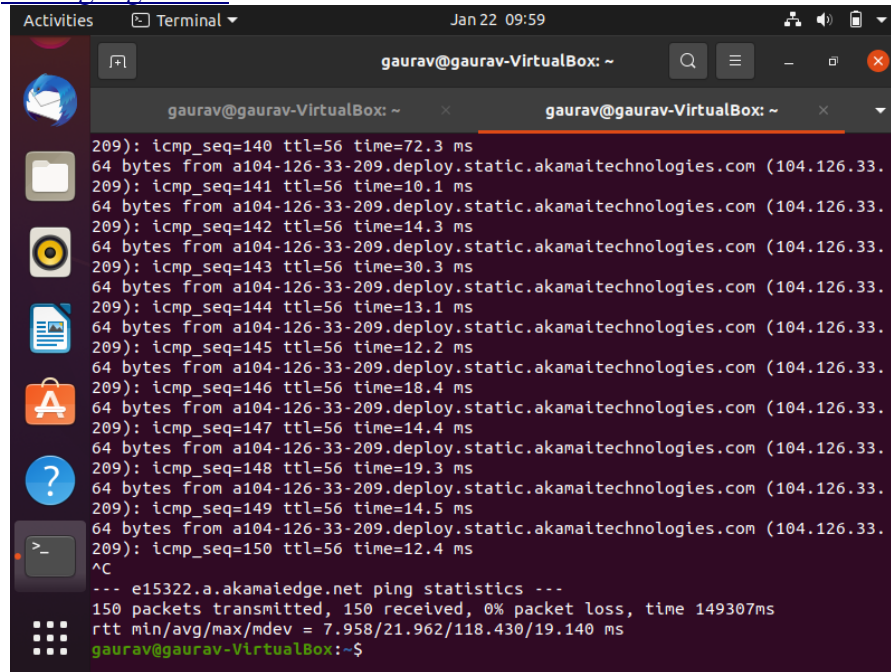
Task 4: Capturing packets with tcpdump

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:

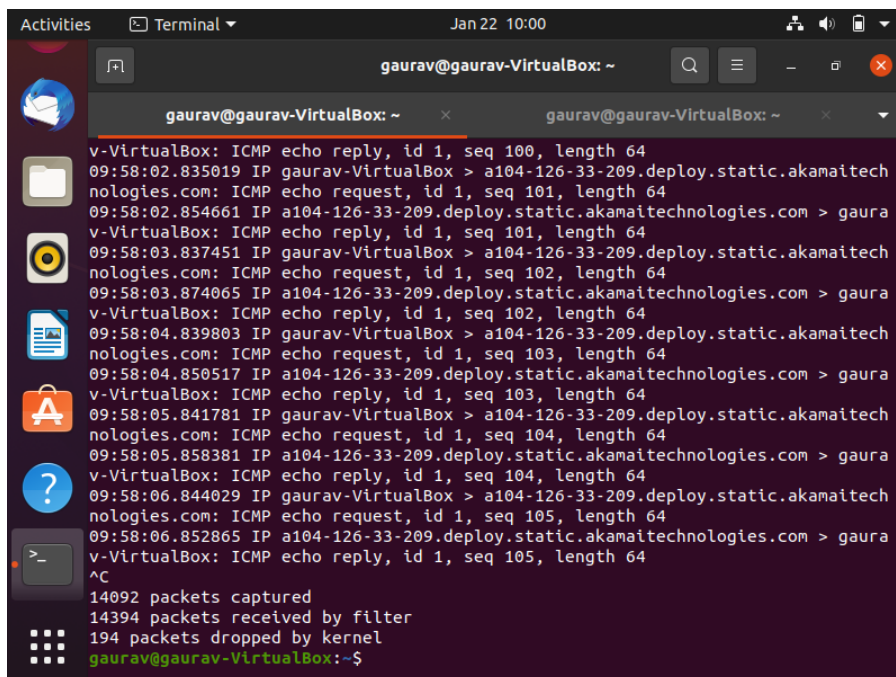
sudo tcpdump -i any

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



```

gaurav@gaurav-VirtualBox: ~
209): icmp_seq=140 ttl=56 time=72.3 ms
64 bytes from a104-126-33-209.deploy.static.akamaitechnologies.com (104.126.33.209): icmp_seq=141 ttl=56 time=10.1 ms
64 bytes from a104-126-33-209.deploy.static.akamaitechnologies.com (104.126.33.209): icmp_seq=142 ttl=56 time=14.3 ms
64 bytes from a104-126-33-209.deploy.static.akamaitechnologies.com (104.126.33.209): icmp_seq=143 ttl=56 time=30.3 ms
64 bytes from a104-126-33-209.deploy.static.akamaitechnologies.com (104.126.33.209): icmp_seq=144 ttl=56 time=13.1 ms
64 bytes from a104-126-33-209.deploy.static.akamaitechnologies.com (104.126.33.209): icmp_seq=145 ttl=56 time=12.2 ms
64 bytes from a104-126-33-209.deploy.static.akamaitechnologies.com (104.126.33.209): icmp_seq=146 ttl=56 time=18.4 ms
64 bytes from a104-126-33-209.deploy.static.akamaitechnologies.com (104.126.33.209): icmp_seq=147 ttl=56 time=14.4 ms
64 bytes from a104-126-33-209.deploy.static.akamaitechnologies.com (104.126.33.209): icmp_seq=148 ttl=56 time=19.3 ms
64 bytes from a104-126-33-209.deploy.static.akamaitechnologies.com (104.126.33.209): icmp_seq=149 ttl=56 time=14.5 ms
64 bytes from a104-126-33-209.deploy.static.akamaitechnologies.com (104.126.33.209): icmp_seq=150 ttl=56 time=12.4 ms
^C
--- e15322.a.akamaiedge.net ping statistics ---
150 packets transmitted, 150 received, 0% packet loss, time 149307ms
rtt min/avg/max/mdev = 7.958/21.962/118.430/19.140 ms
gaurav@gaurav-VirtualBox:~$
  
```



```

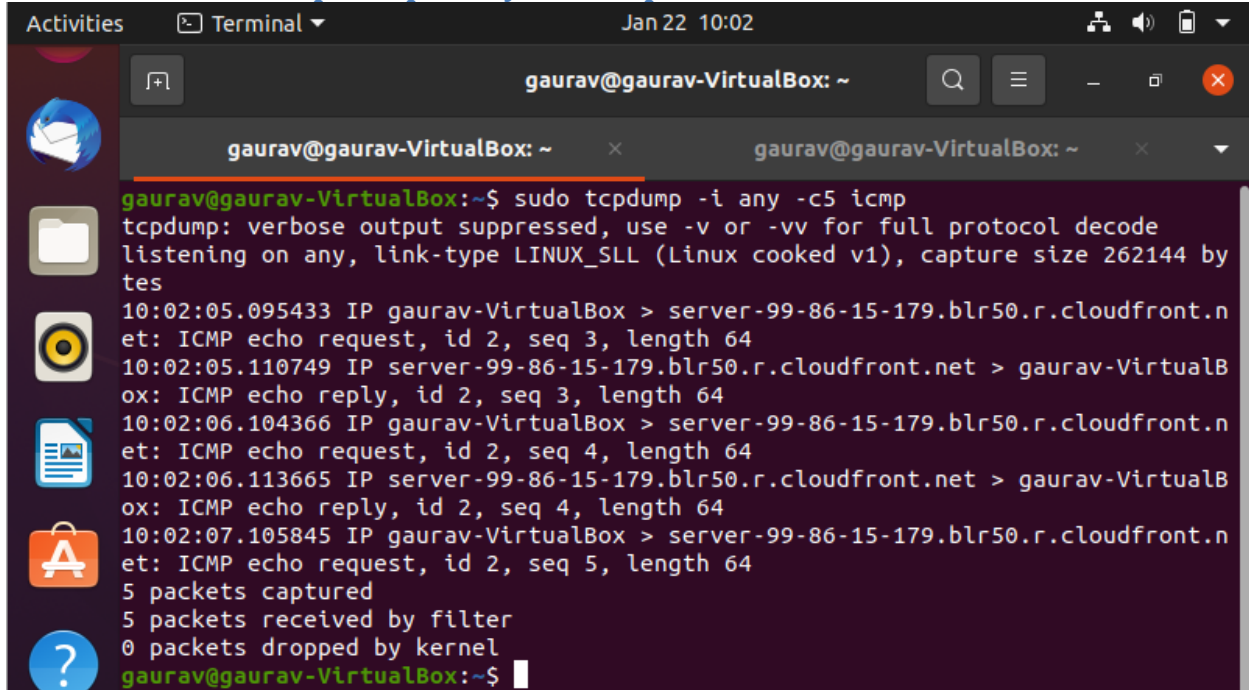
gaurav@gaurav-VirtualBox: ~
v-VirtualBox: ICMP echo reply, id 1, seq 100, length 64
09:58:02.835019 IP gaurav-VirtualBox > a104-126-33-209.deploy.static.akamaitech
nologies.com: ICMP echo request, id 1, seq 101, length 64
09:58:02.854661 IP a104-126-33-209.deploy.static.akamaitechnologies.com > gaura
v-VirtualBox: ICMP echo reply, id 1, seq 101, length 64
09:58:03.837451 IP gaurav-VirtualBox > a104-126-33-209.deploy.static.akamaitech
nologies.com: ICMP echo request, id 1, seq 102, length 64
09:58:03.874065 IP a104-126-33-209.deploy.static.akamaitechnologies.com > gaura
v-VirtualBox: ICMP echo reply, id 1, seq 102, length 64
09:58:04.839803 IP gaurav-VirtualBox > a104-126-33-209.deploy.static.akamaitech
nologies.com: ICMP echo request, id 1, seq 103, length 64
09:58:04.850517 IP a104-126-33-209.deploy.static.akamaitechnologies.com > gaura
v-VirtualBox: ICMP echo reply, id 1, seq 103, length 64
09:58:05.841781 IP gaurav-VirtualBox > a104-126-33-209.deploy.static.akamaitech
nologies.com: ICMP echo request, id 1, seq 104, length 64
09:58:05.858381 IP a104-126-33-209.deploy.static.akamaitechnologies.com > gaura
v-VirtualBox: ICMP echo reply, id 1, seq 104, length 64
09:58:06.844029 IP gaurav-VirtualBox > a104-126-33-209.deploy.static.akamaitech
nologies.com: ICMP echo request, id 1, seq 105, length 64
09:58:06.852865 IP a104-126-33-209.deploy.static.akamaitechnologies.com > gaura
v-VirtualBox: ICMP echo reply, id 1, seq 105, length 64
^C
14092 packets captured
14394 packets received by filter
194 packets dropped by kernel
gaurav@gaurav-VirtualBox:~$
  
```

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:

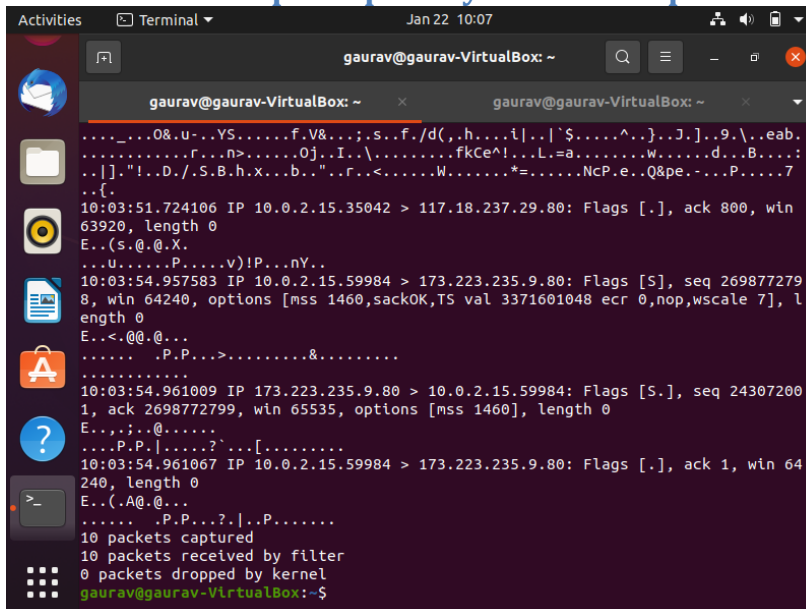
sudo tcpdump -i any -c5 icmp



```
gaurav@gaurav-VirtualBox:~$ sudo tcpdump -i any -c5 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked v1), capture size 262144 by tes
10:02:05.095433 IP gaurav-VirtualBox > server-99-86-15-179.blr50.r.cloudfront.net: ICMP echo request, id 2, seq 3, length 64
10:02:05.110749 IP server-99-86-15-179.blr50.r.cloudfront.net > gaurav-VirtualBox: ICMP echo reply, id 2, seq 3, length 64
10:02:06.104366 IP gaurav-VirtualBox > server-99-86-15-179.blr50.r.cloudfront.net: ICMP echo request, id 2, seq 4, length 64
10:02:06.113665 IP server-99-86-15-179.blr50.r.cloudfront.net > gaurav-VirtualBox: ICMP echo reply, id 2, seq 4, length 64
10:02:07.105845 IP gaurav-VirtualBox > server-99-86-15-179.blr50.r.cloudfront.net: ICMP echo request, id 2, seq 5, length 64
5 packets captured
5 packets received by filter
0 packets dropped by kernel
gaurav@gaurav-VirtualBox:~$
```

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

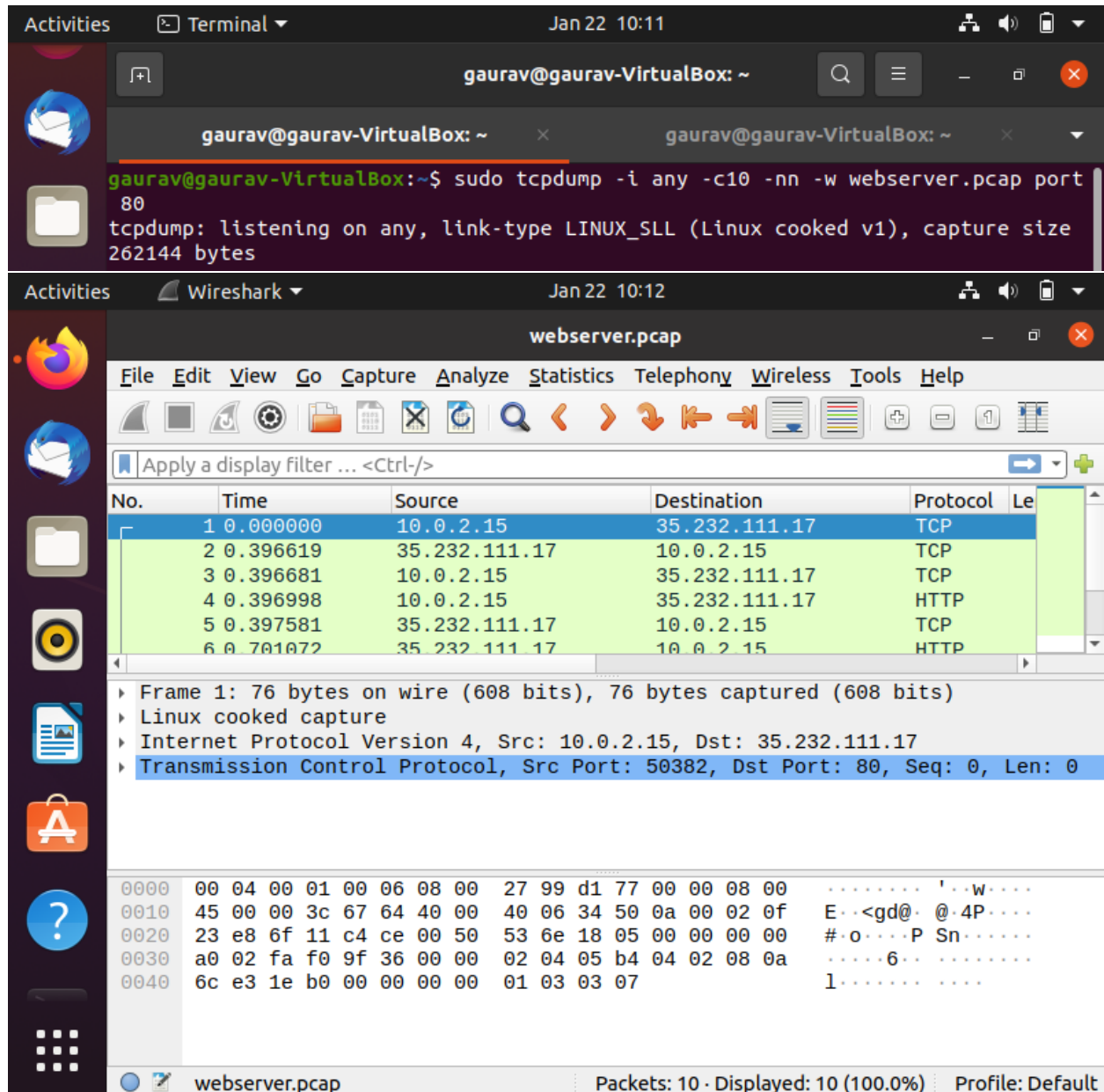
sudo tcpdump -i any -c10 -nn -A port 80



```
gaurav@gaurav-VirtualBox:~$ sudo tcpdump -i any -c10 -nn -A port 80
10:03:51.724106 IP 10.0.2.15.35042 > 117.18.237.29.80: Flags [S], seq 269877279, win 63920, length 0
E..(s.@.X.
...U.....P.....v)IP...nY..
10:03:54.957583 IP 10.0.2.15.59984 > 173.223.235.9.80: Flags [S], seq 269877279, win 64240, options [mss 1460,sackOK,TS val 3371601048 ecr 0,nop,wscale 7], length 0
E..<.@.@...
.....P.P...>.....&.....
10:03:54.961009 IP 173.223.235.9.80 > 10.0.2.15.59984: Flags [S], seq 24307200, ack 269877279, win 65535, options [mss 1460], length 0
E.,;..@.....
....P.P.|.....?`...[.....
10:03:54.961067 IP 10.0.2.15.59984 > 173.223.235.9.80: Flags [S], seq 24307200, ack 1, win 64, length 0
E..(A@.@...
.....P.P...?..P.....
10 packets captured
10 packets received by filter
0 packets dropped by kernel
gaurav@gaurav-VirtualBox:~$
```

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



The screenshot shows a terminal window and the Wireshark network protocol analyzer. The terminal window displays the command `sudo tcpdump -i any -c10 -nn -w webserver.pcap port 80` and its output: `tcpdump: listening on any, link-type LINUX_SLL (Linux cooked v1), capture size 262144 bytes`. The Wireshark window shows the captured packets in the packet list pane, with the first packet selected. The packet details pane shows the structure of the first packet: Frame 1: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface 0, Linux cooked capture. The packet is an Internet Protocol Version 4, Src: 10.0.2.15, Dst: 35.232.111.17. The packet is a Transmission Control Protocol, Src Port: 50382, Dst Port: 80, Seq: 0, Len: 0. The packet bytes pane shows the raw data of the packet.

No.	Time	Source	Destination	Protocol	Length
1	0.000000	10.0.2.15	35.232.111.17	TCP	76
2	0.396619	35.232.111.17	10.0.2.15	TCP	76
3	0.396681	10.0.2.15	35.232.111.17	TCP	76
4	0.396998	10.0.2.15	35.232.111.17	HTTP	76
5	0.397581	35.232.111.17	10.0.2.15	TCP	76
6	0.701072	35.232.111.17	10.0.2.15	HTTP	76

Frame 1: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface 0, Linux cooked capture
 Internet Protocol Version 4, Src: 10.0.2.15, Dst: 35.232.111.17
 Transmission Control Protocol, Src Port: 50382, Dst Port: 80, Seq: 0, Len: 0

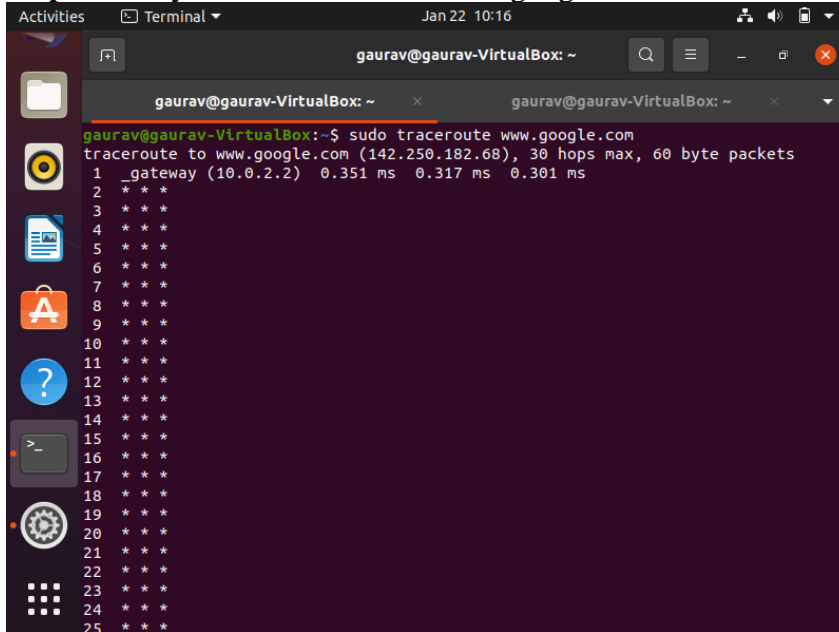
0000 00 04 00 01 00 06 08 00 27 99 d1 77 00 00 08 00 'w....
 0010 45 00 00 3c 67 64 40 00 40 06 34 50 0a 00 02 0f E..<gd@. @.4P....
 0020 23 e8 6f 11 c4 ce 00 50 53 6e 18 05 00 00 00 00 #.o...P Sn.....
 0030 a0 02 fa f0 9f 36 00 00 02 04 05 b4 04 02 08 0a6.....
 0040 6c e3 1e b0 00 00 00 00 01 03 03 07 1.....

Task 5: Perform Traceroute checks

Step 1: Run the traceroute using the following command.

sudo traceroute www.google.com

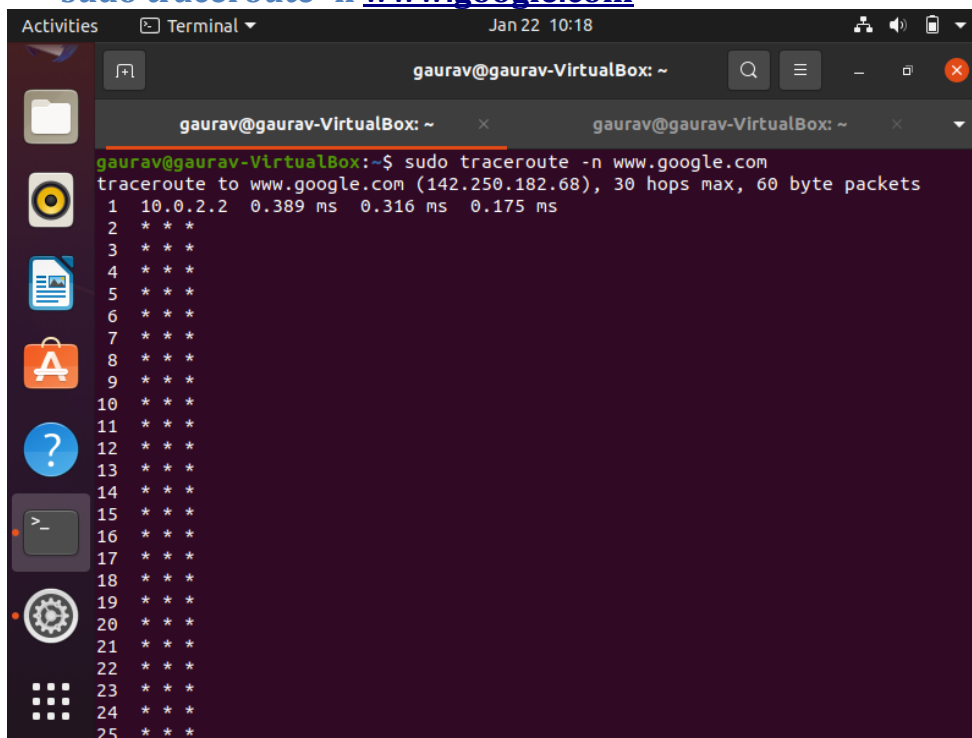
Step 2: Analyze destination address of google.com and no. of hops



```
gaurav@gaurav-VirtualBox: ~  
gaurav@gaurav-VirtualBox: ~  
gaurav@gaurav-VirtualBox:~$ sudo traceroute www.google.com  
traceroute to www.google.com (142.250.182.68), 30 hops max, 60 byte packets  
1  _gateway (10.0.2.2)  0.351 ms  0.317 ms  0.301 ms  
2  * * *  
3  * * *  
4  * * *  
5  * * *  
6  * * *  
7  * * *  
8  * * *  
9  * * *  
10 * * *  
11 * * *  
12 * * *  
13 * * *  
14 * * *  
15 * * *  
16 * * *  
17 * * *  
18 * * *  
19 * * *  
20 * * *  
21 * * *  
22 * * *  
23 * * *  
24 * * *  
25 * * *
```

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



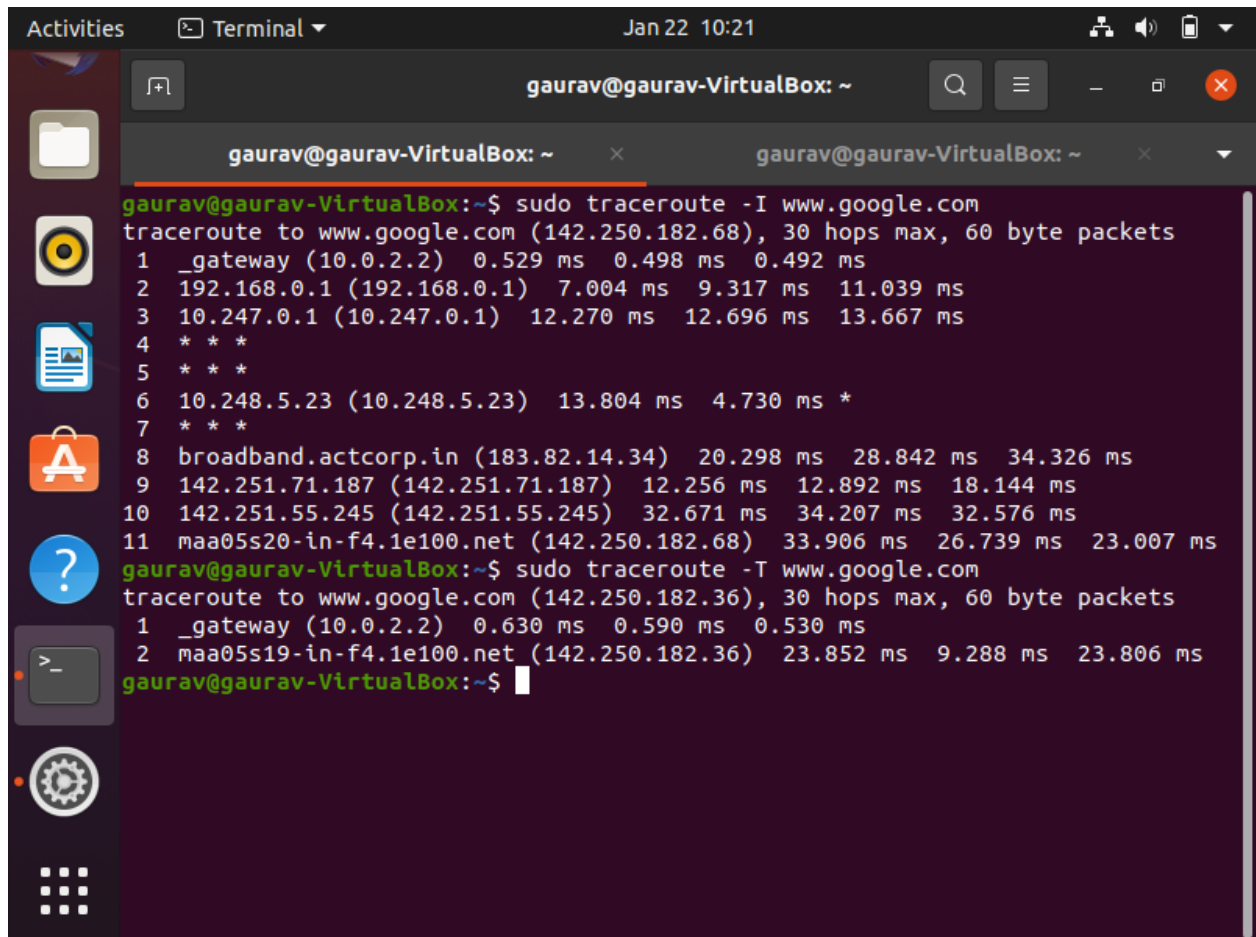
```
gaurav@gaurav-VirtualBox: ~  
gaurav@gaurav-VirtualBox: ~  
gaurav@gaurav-VirtualBox:~$ sudo traceroute -n www.google.com  
traceroute to www.google.com (142.250.182.68), 30 hops max, 60 byte packets  
1  10.0.2.2  0.389 ms  0.316 ms  0.175 ms  
2  * * *  
3  * * *  
4  * * *  
5  * * *  
6  * * *  
7  * * *  
8  * * *  
9  * * *  
10 * * *  
11 * * *  
12 * * *  
13 * * *  
14 * * *  
15 * * *  
16 * * *  
17 * * *  
18 * * *  
19 * * *  
20 * * *  
21 * * *  
22 * * *  
23 * * *  
24 * * *  
25 * * *
```

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

sudo traceroute -I www.google.com

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



```
gaurav@gaurav-VirtualBox: ~$ sudo traceroute -I www.google.com
traceroute to www.google.com (142.250.182.68), 30 hops max, 60 byte packets
 1 _gateway (10.0.2.2)  0.529 ms  0.498 ms  0.492 ms
 2 192.168.0.1 (192.168.0.1)  7.004 ms  9.317 ms  11.039 ms
 3 10.247.0.1 (10.247.0.1)  12.270 ms  12.696 ms  13.667 ms
 4 * * *
 5 * * *
 6 10.248.5.23 (10.248.5.23)  13.804 ms  4.730 ms *
 7 * * *
 8 broadband.actcorp.in (183.82.14.34)  20.298 ms  28.842 ms  34.326 ms
 9 142.251.71.187 (142.251.71.187)  12.256 ms  12.892 ms  18.144 ms
10 142.251.55.245 (142.251.55.245)  32.671 ms  34.207 ms  32.576 ms
11 maa05s20-in-f4.1e100.net (142.250.182.68)  33.906 ms  26.739 ms  23.007 ms
gaurav@gaurav-VirtualBox:~$ sudo traceroute -T www.google.com
traceroute to www.google.com (142.250.182.36), 30 hops max, 60 byte packets
 1 _gateway (10.0.2.2)  0.630 ms  0.590 ms  0.530 ms
 2 maa05s19-in-f4.1e100.net (142.250.182.36)  23.852 ms  9.288 ms  23.806 ms
gaurav@gaurav-VirtualBox:~$
```

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.

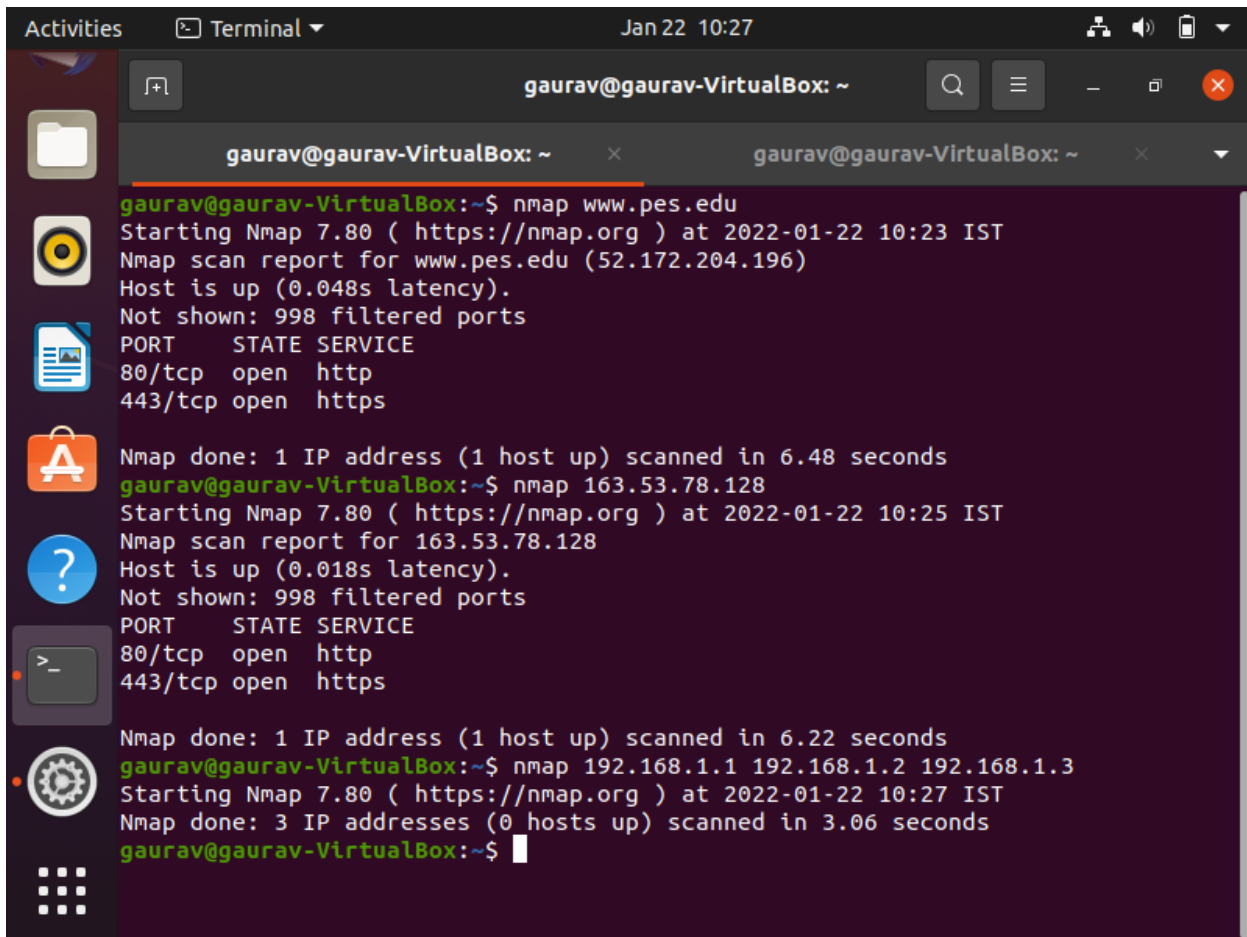
nmap www.pes.edu

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



```
gaurav@gaurav-VirtualBox: ~  
gaurav@gaurav-VirtualBox: ~$ nmap www.pes.edu  
Starting Nmap 7.80 ( https://nmap.org ) at 2022-01-22 10:23 IST  
Nmap scan report for www.pes.edu (52.172.204.196)  
Host is up (0.048s 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 6.48 seconds  
gaurav@gaurav-VirtualBox:~$ nmap 163.53.78.128  
Starting Nmap 7.80 ( https://nmap.org ) at 2022-01-22 10:25 IST  
Nmap scan report for 163.53.78.128  
Host is up (0.018s 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 6.22 seconds  
gaurav@gaurav-VirtualBox:~$ nmap 192.168.1.1 192.168.1.2 192.168.1.3  
Starting Nmap 7.80 ( https://nmap.org ) at 2022-01-22 10:27 IST  
Nmap done: 3 IP addresses (0 hosts up) scanned in 3.06 seconds  
gaurav@gaurav-VirtualBox:~$
```

Questions on above observations:

- 1) Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server?
- 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?
- 4) How will you identify remote host apps and OS?

Answers on above observations:

- 1)Browser is running a HTTP version 1.1 and server is also of the same version
- 2)After much observation,it has been observed that the Last modified date has been not specified.
- 3)By using the command ping -c <number > <website>
- 4)By using nmap