

Computer Networks Lab -1

Name : P K Navin Srinivas

Section : D

SRN : PES2UG20CS237

Week : 1

Note : All written answers are in bold.

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

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

Analyze and fill the following table:

```
<navin>~> ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eno1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether bc:e9:2f:8c:1c:4e brd ff:ff:ff:ff:ff:ff
        altname enp2s0
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether e0:d4:e8:32:73:8c brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.10/24 brd 192.168.1.255 scope global dynamic noprefixroute wlan0
            valid_lft 80014sec preferred_lft 80014sec
        inet6 2401:4900:1f25:c59:7194:3464:c58:daea/64 scope global dynamic noprefixroute
            valid_lft 86351sec preferred_lft 86351sec
        inet6 fe80::7473:30dc:5d7e:cafa/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
<navin>~>
```

ip-address table:

Interface name	IP addresses (IPv4 / IPv6)	MAC address	
Lo	127.0.0.1	00:00:00:00:00:00	Loop back device
Eno1 [enp2s0]	-	bc:e9:2f:8c:1c:4e	Ethernet, disconnected
Wlan0	192.168.1.10	e0:d4:e8:32:73:8c	Wireless LAN

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

```
(navin)~>> sudo ip addr add 10.0.4.237/24 dev wlan0
[sudo] password for navin:
(navin)~>> ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eno1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether bc:e9:2f:8c:1c:4e brd ff:ff:ff:ff:ff:ff
        altname enp2s0
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether e0:d4:e8:32:73:8c brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.10/24 brd 192.168.1.255 scope global dynamic noprefixroute wlan0
            valid_lft 79729sec preferred_lft 79729sec
        inet 10.0.4.237/24 scope global wlan0
            valid_lft forever preferred_lft forever
        inet6 2401:4900:1:f25:c59:7194:3464:c58:daea/64 scope global dynamic noprefixroute
            valid_lft 86067sec preferred_lft 86067sec
        inet6 fe80::fe80:1c4e%wlan0/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
(navin)~>>
```

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

```
sudo ifconfig interface_name down
sudo ifconfig
interface_name up
```

```
(navin)~>> sudo ip link set dev wlan0 down
(navin)~>> ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eno1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether bc:e9:2f:8c:1c:4e brd ff:ff:ff:ff:ff:ff
        altname enp2s0
3: wlan0: <BROADCAST,MULTICAST> mtu 1500 qdisc noqueue state DOWN group default qlen 1000
    link/ether e0:d4:e8:32:73:8c brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.10/24 brd 192.168.1.255 scope global dynamic noprefixroute wlan0
            valid_lft 79589sec preferred_lft 79589sec
        inet 10.0.4.237/24 scope global wlan0
            valid_lft forever preferred_lft forever
(navin)~>> sudo ip link set dev wlan0 up
(navin)~>> sudo ip link set dev wlan0 up
(navin)~>> ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eno1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether bc:e9:2f:8c:1c:4e brd ff:ff:ff:ff:ff:ff
        altname enp2s0
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether e0:d4:e8:32:73:8c brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.10/24 brd 192.168.1.255 scope global dynamic noprefixroute wlan0
            valid_lft 86396sec preferred_lft 86396sec
        inet 10.0.4.237/24 scope global wlan0
            valid_lft forever preferred_lft forever
        inet6 2401:4900:1:f25:c59:7194:3464:c58:daea/64 scope global dynamic noprefixroute
            valid_lft 85910sec preferred_lft 85910sec
        inet6 fe80::fe80:1c4e%wlan0/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
(navin)~>>
```

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

```
ip neigh
```

```
(navin)~>> ip neigh
192.168.1.1 dev wlan0 lladdr f8:c4:f3:00:91:f8 DELAY
fe80::1 dev wlan0 lladdr f8:c4:f3:00:91:f8 router STALE
(navin)~>>
```

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.

Done in previous tasks, hence pinging the same machine causes a loopback, this can be seen with indiffrentiating ip add in the table. Can also be seen with the mac address of device used.

Step 2: Launch Wireshark and select ‘any’ interface

Step 3: In terminal, type

```
ping 10.0.your_section.your_sno
```

```
(navin|~)➤ ping 10.0.4.237
PING 10.0.4.237 (10.0.4.237) 56(84) bytes of data.
64 bytes from 10.0.4.237: icmp_seq=1 ttl=64 time=0.039 ms
64 bytes from 10.0.4.237: icmp_seq=2 ttl=64 time=0.118 ms
64 bytes from 10.0.4.237: icmp_seq=3 ttl=64 time=0.085 ms
^C
--- 10.0.4.237 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2033ms
rtt min/avg/max/mdev = 0.039/0.080/0.118/0.032 ms
(navin|~)➤
```

288 209.95121/149 192.168.1.1	224.0.0.1	IGMPv3	52 Membership query, general
289 209.956193198 192.168.1.10	224.0.0.22	IGMPv3	56 Membership Report / Join group 224.0.0.251 for any sources
290 210.638634027 192.168.1.17	239.255.255.250	SSDP	169 M-SEARCH * HTTP/1.1
291 210.945437087 192.168.1.17	239.255.255.250	SSDP	169 M-SEARCH * HTTP/1.1
292 211.203399261 10.0.4.237	10.0.4.237	ICMP	100 Echo (ping) request id=0x0002, seq=1/256, ttl=64 (reply in 293)
293 211.203412261 10.0.4.237	10.0.4.237	ICMP	100 Echo (ping) reply id=0x0002, seq=1/256, ttl=64 (request in 292)
294 211.252737245 192.168.1.17	239.255.255.250	SSDP	169 M-SEARCH * HTTP/1.1
295 212.222698456 10.0.4.237	10.0.4.237	ICMP	100 Echo (ping) request id=0x0002, seq=2/512, ttl=64 (reply in 296)
296 212.222723390 10.0.4.237	10.0.4.237	ICMP	100 Echo (ping) reply id=0x0002, seq=2/512, ttl=64 (request in 295)
297 213.236277453 10.0.4.237	10.0.4.237	ICMP	100 Echo (ping) request id=0x0002, seq=3/768, ttl=64 (reply in 298)
298 213.236360851 10.0.4.237	10.0.4.237	ICMP	100 Echo (ping) reply id=0x0002, seq=3/768, ttl=64 (request in 297)
299 214.837412836 192.168.1.18	239.255.255.250	SSDP	169 M-SEARCH * HTTP/1.1
300 215.142655866 102.168.1.10	239.255.255.250	SSDP	169 M-SEARCH * HTTP/1.1

Observations to be made

Step 4: Analyze the following in Terminal

- TTL : 64
- Protocol used by ping : ICMP : Internet Control Message Protocol
- Time : ~2000ms , observed from ping command

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	292	293
Source IP address	10.0.4.237	10.0.4.237
Destination IP address	10.0.4.237	10.0.4.237
ICMP Type Value	8	0
ICMP Code Value	0	0
Source Ethernet Address	00:00:00:00:00:00	00:00:00:00:00:00
Destination Ethernet Address	00:00:00:00:00:00	00:00:00:00:00:00
Internet Protocol Version	4	4
Time To Live (TTL) Value	64	64

First ping request :

```

▶ Frame 292: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface any, id 0
└▶ Linux cooked capture v1
    Packet type: Unicast to us (0)
    Link-layer address type: Loopback (772)
    Link-layer address length: 6
    Source: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Unused: 0a0d
    Protocol: IPv4 (0x0800)
└▶ Internet Protocol Version 4, Src: 10.0.4.237, Dst: 10.0.4.237
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 84
    Identification: 0xdb8c (56204)
    ▶ Flags: 0x40, Don't fragment
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0x4143 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.0.4.237
    Destination Address: 10.0.4.237
└▶ Internet Control Message Protocol

```

First Ping response :

```

▶ Frame 293: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface any, id 0
└▶ Linux cooked capture v1
    Packet type: Unicast to us (0)
    Link-layer address type: Loopback (772)
    Link-layer address length: 6
    Source: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Unused: 4000
    Protocol: IPv4 (0x0800)
└▶ Internet Protocol Version 4, Src: 10.0.4.237, Dst: 10.0.4.237
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 84
    Identification: 0xdb8d (56205)
    ▶ Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0x8142 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.0.4.237
    Destination Address: 10.0.4.237
└▶ Internet Control Message Protocol

```

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

www.flipkart.com does not do http requests and hence wireshark can not analyze them, thus I would be doing my analysis using :

<http://info.cern.ch/>

http					
No.	Time	Source	Destination	Protocol	Length Info
+ 17	7.065784814	192.168.1.10	188.184.21.108	HTTP	624 GET / HTTP/1.1
+ 20	7.373776527	188.184.21.108	192.168.1.10	HTTP	196 HTTP/1.1 304 Not Modified

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	17	20
Source Port	58694	80
Destination Port	80	58964
Source IP address	192.168.1.10	188.184.21.108
Destination IP address	188.184.21.108	192.148.1.10
Source Ethernet Address	e0:d4:e8:32:73:8c	f8:c4:f3:00:91:f8
Destination Ethernet Address	f8:c4:f3:00:91:f8	e0:d4:e8:32:73:8c

HTTP First request :

HTTP First response :

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	Apache
Host	info.cern.ch	Content-Type	text/plain
User-Agent	Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.4692.71 Safari/537.36	Date	Thu, 20 Jan 2022 12:09:16 GMT
Accept-Language	en-GB,en-US;q=0.9,en;q=0.8	Location	
Accept-Encoding	gzip, deflate	Content-Length	25
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.

Entire conversation :

```
GET / HTTP/1.1
Host: info.cern.ch
Connection: keep-alive
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.4692.71 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-GPC: 1
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
If-None-Match: "286-4f1aadb3105c0"
If-Modified-Since: Wed, 05 Feb 2014 16:00:31 GMT

HTTP/1.1 304 Not Modified
Date: Thu, 20 Jan 2022 10:46:01 GMT
Server: Apache
Connection: close
ETag: "286-4f1aadb3105c0"
```

Sadly this connection did not yield any TCP connections to follow.

Task 4: Capturing packets with tcpdump

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

```
sudo tcpdump -D
```

```
<navinl~>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.eno1 [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]
```

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.

```
fish /home/navin
3821: 51619 1/0/0 PTR maa05s22-in-f4.1e100.net. (84)
17:51:00.667167 IP _gateway.domain > navin-omenlaptop15en0xxx.3
9056: 61972 1/0/0 PTR static-182-18-160-106.ctrls.in. (89)
17:51:00.668192 IP navin-omenlaptop15en0xxx.35571 > _gateway.do
main: 61577+ PTR? 201.169.37.52.in-addr.arpa. (44)
17:51:01.532626 IP navin-omenlaptop15en0xxx.43802 > ec2-52-18-2
13-104.eu-west-1.compute.amazonaws.com.https: Flags [.], ack 42
win 501, options [nop,nop,TS val 111952698 ecr 150377
0594], length 0
17:51:01.596562 IP navin-omenlaptop15en0xxx.44574 > _gateway.do
main: 29163+ PTR? 104.213.18.52.in-addr.arpa. (44)
17:51:01.681118 IP ec2-52-18-213-104.eu-west-1.compute.amazonaws.com.https > navin-omenlaptop15en0xxx.43802: Flags [.], ack 1,
win 129, options [nop,nop,TS val 1503816672 ecr 11186106], length 0
17:51:02.139812 IP navin-omenlaptop15en0xxx > www.google.com: ICMP echo request, id 11, seq 6, length 64
17:51:02.152887 IP www.google.com > navin-omenlaptop15en0xxx: ICMP echo reply, id 11, seq 6, length 64
17:51:02.153473 IP navin-omenlaptop15en0xxx.41323 > _gateway.do
main: 51466+ PTR? 132.182.256.142.in-addr.arpa. (46)
17:51:02.159235 IP _gateway.domain > navin-omenlaptop15en0xxx.4
1323: 51466 1/0/0 PTR maa05s22-in-f4.1e100.net. (84)
17:51:02.305081 IP _gateway.domain > navin-omenlaptop15en0xxx.4
4574: 29163 1/0/0 PTR ec2-52-18-213-104.eu-west-1.compute.amazonaws.com. (107)
^C
107 packets captured
120 packets received by filter
13 packets dropped by kernel
<navinl~>
```



```
fish /home/navin
64 bytes from maa03s47-in-f4.1e100.net (142.250.196.164): icmp_
seq=2 ttl=59 time=12.4 ms
64 bytes from maa03s47-in-f4.1e100.net (142.250.196.164): icmp_
seq=3 ttl=59 time=13.3 ms
64 bytes from maa03s47-in-f4.1e100.net (142.250.196.164): icmp_
seq=4 ttl=59 time=13.3 ms
64 bytes from maa03s47-in-f4.1e100.net (142.250.196.164): icmp_
seq=5 ttl=59 time=12.2 ms
^C
--- www.google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 11.829/12.679/13.470/0.670 ms
<navinl~> ping www.google.com
PING www.google.com (142.250.182.132) 56(84) bytes of data.
64 bytes from www.google.com (142.250.182.132): icmp_seq=1 ttl=11
time=12.9 ms
64 bytes from maa05s22-in-f4.1e100.net (142.250.182.132): icmp_
seq=2 ttl=11 time=12.4 ms
64 bytes from maa05s22-in-f4.1e100.net (142.250.182.132): icmp_
seq=3 ttl=11 time=11.0 ms
64 bytes from maa05s22-in-f4.1e100.net (142.250.182.132): icmp_
seq=4 ttl=11 time=9.89 ms
64 bytes from maa05s22-in-f4.1e100.net (142.250.182.132): icmp_
seq=5 ttl=11 time=12.6 ms
64 bytes from maa05s22-in-f4.1e100.net (142.250.182.132): icmp_
seq=6 ttl=11 time=13.1 ms
^C
--- www.google.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 9.892/11.987/13.120/1.148 ms
<navinl~>
```

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
```

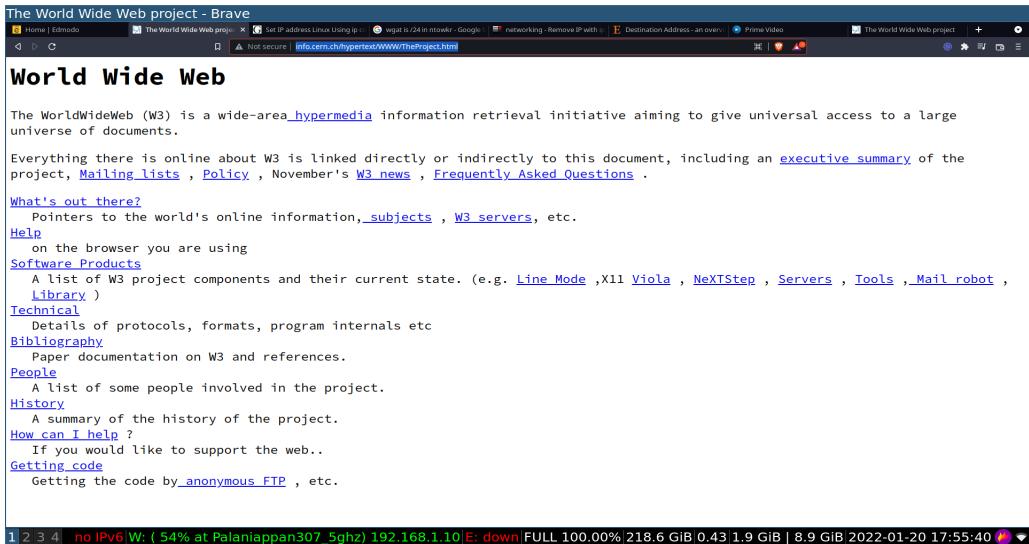
Step 6: To save packets to a file instead of displaying them on screen, use the option **-w**:

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

PING TO IN BROWSER :

<http://info.cern.ch/hypertext/WWW/TheProject.html>

A ping command was not used as filter command was for HTTP and not ICMP



```
(navin㉿) ~ > sudo tcpdump -i any -c10 -nn -w first.pcap port 80
tcpdump: data link type LINUX_SLL2
tcpdump: listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
10 packets captured
14 packets received by filter
0 packets dropped by kernel
```

Opening the above file in wireshark :

No.	Time	Source	Destination	Protocol	Length: Info
1	0:000000	192.168.1.10	188.184.21.108	TCP	88 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TStamp=2856479228 TSectr=0 WS=128
2	0.256523	192.168.1.10	188.184.21.108	TCP	88 58704 - 88 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TStamp=2856479478 TSectr=0 WS=128
3	1.016654	192.168.1.10	188.184.21.108	TCP	88 [TCP Retransmission] [TCP_Port numbers reused] 58702 - 88 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TStamp=2856480244 TSectr=0 WS=128
4	1.228666	188.184.21.108	192.168.1.10	TCP	88 80 - 58702 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1452 SACK_PERM=1 TStamp=2864238293 TSectr=2856480244 TSectr=2664238293
5	1.228907	192.168.1.10	188.184.21.108	TCP	72 58702 - 88 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TStamp=2856480456 TSectr=2664238293
6	1.235635	192.168.1.10	188.184.21.108	TCP	88 [TCP Retransmission] [TCP_Port numbers reused] 58704 - 88 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TStamp=2856480484 TSectr=0 WS=128
7	1.256619	192.168.1.10	188.184.21.108	TCP	72 80 - 58702 [ACK] Seq=1 Ack=586 Win=30280 Len=0 TStamp=2864238504 TSectr=2856480457
8	1.433321	188.184.21.108	192.168.1.10	HTTP	208 HTTP/1.1 304 Not Modified
9	1.433859	188.184.21.108	192.168.1.10	HTTP	72 58702 - 88 [ACK] Seq=586 Ack=129 Win=64128 Len=0 TStamp=2856480661 TSectr=2664238504
10	1.433901	192.168.1.10	188.184.21.108	TCP	

```

Frame 6: 657 bytes on wire (5256 bits), 657 bytes captured (5256 bits)
Link layer: Linux cooked capture v2
Internet Protocol Version 4, Src: 192.168.1.10, Dst: 188.184.21.108
Transmission Control Protocol, Src Port: 58702, Dst Port: 80, Seq: 1, Ack: 1, Len: 585
Hypertext Transfer Protocol
    > GET /hypertext/WWW/TheProject.html HTTP/1.1\r\n
        Host: info.cern.ch\r\n
        Connection: keep-alive\r\n
        Cache-Control: max-age=0\r\n
        Upgrade-Insecure-Requests: 1\r\n
        User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.4692.71 Safari/537.36\r\n
        Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9\r\n
        Sec-GPC: 1\r\n
        Accept-Encoding: gzip, deflate\r\n
        Accept-Language: en-GB,en-US;q=0.9,en;q=0.8\r\n
        If-None-Match: "8a9-291e721995900"\r\n
        If-Modified-Since: Thu, 03 Dec 1992 08:37:20 GMT\r\n
    \r\n
[Full request URL: http://info.cern.ch/hypertext/WWW/TheProject.html]
[HTTP request 1/1]
[Response in frame: 9]
```

Task 5: Perform Traceroute checks

Step 1: Run the traceroute using the following command.

```
sudo traceroute www.google.com
```

```
(navin㉿) ~ % sudo traceroute www.google.com
traceroute to www.google.com (142.250.182.132), 30 hops max, 60 byte packets
 1 * _gateway (192.168.1.1)  6.348 ms  6.553 ms local LAN gateway
 2  223.178.56.1 (223.178.56.1)  6.390 ms  6.379 ms  7.429 ms Some hops to ISP
 3  nsg-corporate-101.95.187.122.airtel.in (122.187.95.101)  7.419 ms nsg-corporate-97.95.187.122.airtel.in (122.187.95.97)  8.
215 ms  8.206 ms  My ISP
 4  * 182.79.198.0 (182.79.198.0)  23.047 ms 182.79.142.222 (182.79.142.222)  45.746 ms
 5  72.14.216.192 (72.14.216.192)  16.805 ms 72.14.208.234 (72.14.208.234)  13.598 ms  13.588 ms
 6  * * *
 7  142.251.55.232 (142.251.55.232)  13.419 ms 108.170.253.97 (108.170.253.97)  14.172 ms 142.250.228.80 (142.250.228.80)  12.4
90 ms
 8  maa05s22-in-f4.1e100.net (142.250.182.132)  13.626 ms 216.239.56.63 (216.239.56.63)  11.377 ms  11.335 ms
(navin㉿) ~ %
```

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

Destination : 142.250.182.132

Number of hops : 8

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
```

```
(navin㉿) ~ % sudo traceroute -n www.google.com
[sudo] password for navin:
traceroute to www.google.com (142.250.182.4), 30 hops max, 60 byte packets
 1  192.168.1.1  3.098 ms  3.059 ms  6.017 ms
 2  223.178.56.1  6.017 ms  6.001 ms  6.092 ms
 3  122.187.95.101  6.618 ms  6.604 ms  6.586 ms
 4  * * 116.119.57.201  12.156 ms
 5  72.14.216.192  13.064 ms 72.14.208.234  14.314 ms  14.030 ms
 6  * * *
 7  142.251.55.40  10.661 ms 142.251.55.228  10.741 ms 108.170.253.97  11.001 ms
 8  108.170.253.122  14.167 ms 142.251.55.219  12.488 ms 108.170.253.105  11.211 ms
 9  74.125.242.145  12.704 ms 142.250.182.4  12.021 ms 74.125.242.145  10.796 ms
(navin㉿) ~ %
```

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

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

```
(navin㉿) ~ % sudo traceroute -I www.google.com
traceroute to www.google.com (142.250.182.4), 30 hops max, 60 byte packets
 1  _gateway (192.168.1.1)  1.927 ms  1.879 ms  2.503 ms
 2  223.178.56.1 (223.178.56.1)  10.468 ms  10.458 ms  10.738 ms
 3  nsg-corporate-97.95.187.122.airtel.in (122.187.95.97)  10.799 ms  10.790 ms  10.782 ms
 4  * * *
 5  72.14.208.234 (72.14.208.234)  10.965 ms  11.203 ms  11.194 ms
 6  72.14.232.71 (72.14.232.71)  12.310 ms  11.927 ms  11.880 ms
 7  142.251.55.217 (142.251.55.217)  11.164 ms  11.081 ms  10.982 ms
 8  maa05s18-in-f4.1e100.net (142.250.182.4)  10.251 ms  10.240 ms  10.232 ms
(navin㉿) ~ %
```

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
```

```
(navin㉿) ~ % sudo traceroute -T www.google.com
traceroute to www.google.com (142.250.182.4), 30 hops max, 60 byte packets
 1  gateway (192.168.1.1)  2.496 ms  2.423 ms  2.937 ms
 2  223.178.56.1 (223.178.56.1)  5.142 ms  5.343 ms  6.310 ms
 3  nsg-corporate-97.95.187.122.airtel.in (122.187.95.97)  6.322 ms  6.308 ms  6.291 ms
 4  182.79.198.0 (182.79.198.0)  25.202 ms * *
 5  72.14.216.192 (72.14.216.192)  14.252 ms  15.000 ms  72.14.208.234 (72.14.208.234)  13.212 ms
 6  142.251.227.213 (142.251.227.213)  13.678 ms  142.251.227.211 (142.251.227.211)  12.297 ms  72.14.234.9 (72.14.234.9)  12.965
ms
 7  142.251.55.219 (142.251.55.219)  12.230 ms  142.251.55.217 (142.251.55.217)  10.315 ms  9.600 ms
 8  maa05s18-in-f4.1e100.net (142.250.182.4)  11.502 ms  11.986 ms  11.971 ms
(navin㉿) ~ %
```

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
```

```
<navin|~>> nmap www.pes.edu
Starting Nmap 7.92 ( https://nmap.org ) at 2022-01-20 18:13 IST
Nmap scan report for www.pes.edu (52.172.204.196)
Host is up (0.026s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https

Nmap done: 1 IP address (1 host up) scanned in 5.47 seconds
<navin|~>> nmap 192.168.1.1
Starting Nmap 7.92 ( https://nmap.org ) at 2022-01-20 18:14 IST
Nmap scan report for Broadcom.Home (192.168.1.1)
Host is up (0.0037s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https

Nmap done: 1 IP address (1 host up) scanned in 3.97 seconds
<navin|~>> nmap 192.168.1.11
Starting Nmap 7.92 ( https://nmap.org ) at 2022-01-20 18:14 IST
Nmap scan report for server1 (192.168.1.11)
Host is up (0.0057s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT      STATE SERVICE
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 0.30 seconds
```

TASK 7 A) NETCAT AS CHAT TOOL

a) Intra system communication (Using 2 terminals in the same system)

Step 1: Open a terminal (Ctrl+Alt+T). This will act as a Server.

Step 2: Type nc -l any_portnum (For eg., nc -l 1234)

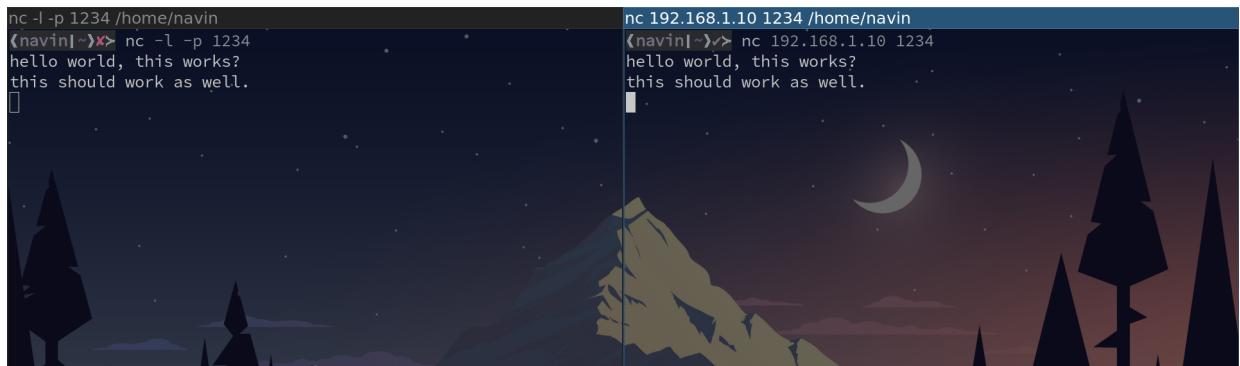
Note: It will goto listening mode

Step 3: Open another terminal and this will act as a client.

Step 4: Type nc <your-system-ip-address> portnum

Note: portnum should be common in both the terminals (for eg., nc 10.0.2.8 1234)

Step 5: Type anything in client will appear in server



```
nc -l -p 1234 /home/navin
<navin|~>x> nc -l -p 1234
hello world, this works?
this should work as well.

nc 192.168.1.10 1234 /home/navin
<navin|~>x> nc 192.168.1.10 1234
hello world, this works?
this should work as well.
```

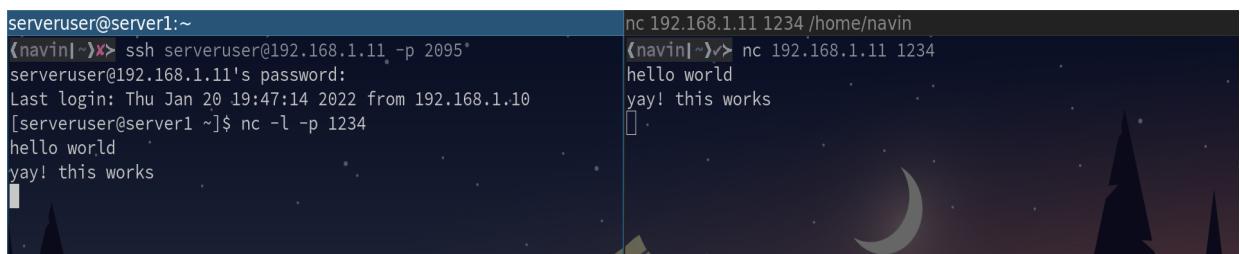
b) Inter system communication Setup a simple switched network of 2 PCs with one acting as Web server. Assign IP addresses for both PCs. Set the capture option as described above.

Step 1: Open terminal on Server machine (Machine 1).

Step 2: Type nc -l any_portnum

Step 3: Open terminal on the Client machine (Machine 2)

Step 4: Type nc



```
serveruser@server1:~
<navin|~>x> ssh serveruser@192.168.1.11 -p 2095
serveruser@192.168.1.11's password:
Last login: Thu Jan 20 19:47:14 2022 from 192.168.1.10
[serveruser@server1 ~]$ nc -l -p 1234
hello world
yay! this works

nc 192.168.1.11 1234 /home/navin
<navin|~>x> nc 192.168.1.11 1234
hello world
yay! this works
```

INFO :

The two machines are :

192.168.1.10 : local machine

192.168.1.11 : my server withing my LAN, which I SSH'ed into.

TASK 7 B): USE NETCAT TO TRANSFER FILES

The netcat utility can also be used to transfer files.

Step 1: At the server side, create an empty file named 'test.txt' sudo nc -l 555 > test.txt

Note: 2 students can combine for the following tasks (switch and cables can be taken from Lab technicians)

Step 2: At the client side, we have a file 'test.txt'. Add some contents to it.

Step 3: Run the client as: sudo nc 10.0.2.8 555 < test.txt here, 10.0.2.8 is the IP address of server and 555 is the port number.

Step 4: At server side, verify the file transfer using the command cat test.txt

serveruser@server1:~	fish /home/navin
[serveruser@server1 ~]\$ touch test.txt	← create a file
[serveruser@server1 ~]\$ cat test.txt	
[serveruser@server1 ~]\$ cat > test.txt	← write to the file
hello world, this has been transferred	
[serveruser@server1 ~]\$ nc -l -p 1234 < test.txt	
[serveruser@server1 ~]\$ nc -l -p 1234 < test.txt	← pass the file to netcat
^C[serveruser@server1 ~]\$	
	← write contents from netcat to a file
	← read the file
	hello world, this has been transferred.
	{navin ~}>

Questions on above observations:

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

HTTP 1.1

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

304 Not Modified

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

ping ip -v number_of_pings

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

The nmap -O -v ip_addr

The above command will show remote hosts and apps