CS342: Networks Lab
Assignment 1

Gunjan Dhanuka - 200101038
Department of Computer Science and Engineering

August 14, 2022

Question 1: ping

(a) ping -c <NUMBER OF ECHO REQUESTS>

(b) ping -i <TIME INTERVAL>

(c) ping -l <NO OF PACKETS>
The number of **ECHO_REQUEST** packets that can be sent one after another without waiting for a reply by a normal user (not super user) is **atmost 3**, and can be sent only after a **time interval of 200 ms** using the option -i (as mentioned above).

(d) ping -s <ECHO REQUEST PACKET SIZE in bytes>
If the payload size of the packet is 32 bytes, then the total packet size would be **60 bytes** (IP header: 8 bytes + ICMP header: 20 bytes + Payload size: 32 bytes).

Question 2: RTT

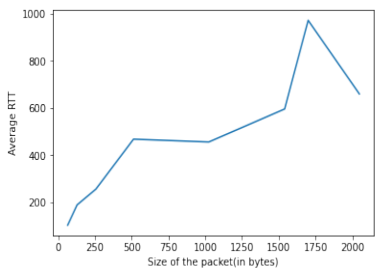
(a)

Sr. No	Host	Avg RTT 4 PM	Avg RTT 10 PM	Avg RTT 8 AM	Avg RTT
1	google.com	153.340 ms	154.421 ms	160.586 ms	156.115 ms
2	youtube.com	151.423 ms	152.890 ms	159.675 ms	154.662 ms
3	instagram.com	129.535 ms	130.022 ms	135.324 ms	131.625 ms
4	amazon.in	287.676 ms	304.887 ms	292.034 ms	294.865 ms
5	medium.com	183.482 ms	185.546 ms	190.387 ms	186.447 ms
6	bitcoin.org	158.158 ms	160.475 ms	170.945 ms	163.192 ms

(b) **Packet Loss**: There were NO cases that showed more than 0% packet loss. However, generally packet loss can occur if there is some network congestion. Some packets might collide with others. If the server drops all the ICMP packets then we have a 100% packet loss.

(c) **RTT and correlation with Geographical distance**: The above collected examples show weak relation of Average RTT with Geographical distance. To establish a strong connection, we need to repeat the experiment in a standardized way for a large number of iterations to ascertain. However, in theory the RTT should generally increase with Geographical distance since the packet passes through more nodes and thus the processing delay adds up. However there are other factors like bandwidth and congestion along a path, which also affect the RTT.

(d) Variation of RTT with packet size for **bitcoin.org**:



Packet size	64	128	256	512	1024	1540	1700	2048
RTT(ms)	103.2	189.8	256.7	468.9	456.2	596.5	972.4	660.2

The data shows that the RTT increases steadily till around 1500 but then we see a huge jump. The MTU is set of 1500 bytes by default. If the packet size is less than 1500 bytes, it pads to make their size equal to 1500 bytes, hence keeping the RTTs almost similar. If the packet is greater than 1500 bytes, it is split into multiple packets each of size 1500 bytes, and hence the increase in time.

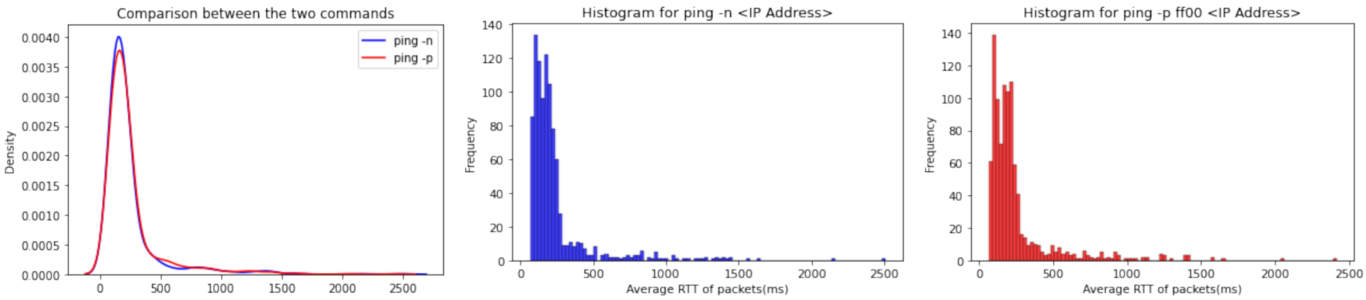
Question 3: ping commands

(a) Using **8.8.8.8** as the host. Packet loss rate using **-n** is 2.6% while packet loss using **-p ff00** is 2.9%.

(b)

Command	Min latency	Max latency	Mean latency	Median latency
ping -n <IPAddress>	67.0 ms	2506.0 ms	237.792 ms	175.0 ms
ping -p ff00 <IPAddress>	64.0 ms	2415.0 ms	248.380 ms	183.0 ms

(c)



(d) The two commands are similar and both represent a normal distribution of values. However, **-p ff00** sends the packet filled with **1111111100000000**, i.e. 8 ones and 8 zeros. Thus synchronisation of the clocks can have problems, since there is one transition (i.e. 1 to 0) present in the bit pattern, and hence we observe a higher packet loss rate.
In the **-n** option, no attempt is tried to get the host names, and thus the mean latency is lower than **-p** case.

Question 4: ifconfig and route

(a) **ifconfig:** ifconfig is used to view and change the configuration of the network interfaces on a UNIX-based system.

```
(base) gunjan@gunjan-TUF:~$ ifconfig
enp2s0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 3c:7c:3f:59:db:2f  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

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 20307  bytes 1928920 (1.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20307  bytes 1928920 (1.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.122.1  netmask 255.255.255.0  broadcast 192.168.122.255
        ether 52:54:00:6d:93:0a  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

wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.150.35.99  netmask 255.255.240.0  broadcast 10.150.47.255
        inet6 fe80::9e19:89e7:6ba3:3869  prefixlen 64  scopeid 0x20<link>
        ether 94:08:53:3a:1f:8b  txqueuelen 1000  (Ethernet)
        RX packets 230525  bytes 75767415 (75.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 123302  bytes 32398371 (32.3 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Network Interface: The software interface to networking hardware. We can see two types of network interfaces: physical and virtual.

(i) **Physical network interface:** The actual network hardware such as the network interface controller (NIC). In my case, I observed **enp2s0** and **wlp3s0**.
enp2s0: Ethernet, PCI Bus 2, Slot 0
wlp3s0: Wireless LAN, PCI bus 3, Slot 0

(ii) **Virtual network interface:** It is linked to a hardware device but doesn't represent one. In our case, we observe **lo** and **virbr0**.
lo: Loopback in the output. **virbr0:** Virtual Bridge 0, used for NAT (Network Address Translation).

Interface details:

(i) **UP**: Shows the interface-related kernel modules have been loaded.

(ii) **LOOPBACK**: Tells that the interface is loopback mode. Packets tran

(iii) **RUNNING**: Tells that the interface is ready to accept data.

(iv) **MULTICAST**: Tells that the interface support multicasting - sending packets to a select group of systems.

(v) **BROADCAST**: Tells that the system supports broadcasting.

(vi) **ether**: Ethernet Interface

(vii) **inet**: IPv4 address assigned to the interface.

(viii) **netmask**: network mask for the interface

(ix) **broadcast**: broadcast address for the interface.

(x) **inet6**: IPv6 address assigned to the interface.

(xi) **mtu**: Maximum transmission unit.

(xii) **scope**: It is the scope of the IPv6 address. It can be either link-local and global.

Interface stats:

(i) **RX packets**: The total number of packets received.

(ii) **RX error**: The total number of packets received with errors.

(iii) **RX bytes**: The total number of bytes received.

(iv) **RX dropped**: The total number of dropped packets because of unintended VLAN tags or receiving IPv6 frames when interface is not configured for IPv6.

(v) **RX overruns**: The number of packets received that experienced FIFO overrruns.

(vi) **RX frame**: Number of misaligned frames.

(vii) **TX packets**: Number of packets transmitted.

(viii) **TX bytes**: Number of bytes transmitted.

(ix) **TX txqueulen**: The length of transmission queue.

(x) **TX carriers**: Number of packets that experienced loss of carriers.

(xi) **TX collisions**: Number of transmitted packets that experienced Ethernet collisions.

(b) The options which can be used with ifconfig are:

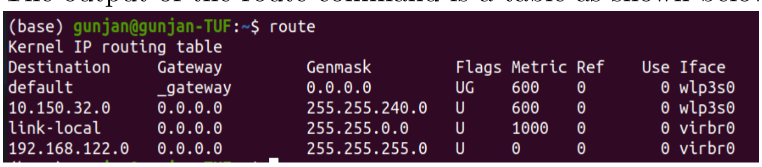
(i) -a: Display all the available interfaces, even if they are down.

(ii) -s: Display a short list, instead of a detailed one.

(iii) up: Activate the driver for the given interface.

(iv) down: Deactivate the driver for the given interface.

(c) The output of the route command is a table as shown below:



```
(base) gunjan@gunjan-TUF:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         _gateway        0.0.0.0         UG    600    0        0 wlp3s0
10.150.32.0     0.0.0.0         255.255.240.0   U     600    0        0 wlp3s0
link-local      0.0.0.0         255.255.0.0     U     1000   0        0 virbr0
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 virbr0
```

The various columns are:

(i) **Destination**: Destination network or the destination host.

(ii) **Gateway**: Gateway address

(iii) **Genmask**: The netmask for the destination net; 255.255.255.255 for a host destination and 0.0.0.0 for the default route.

(iv) **Flags**:
U - indicates that the route is up
G - indicates that the route is to a gateway
H - Indicates that the destination is a fully qualified host address, rather than a network

(v) **Metric**: The distance to the target (measured in loops)

(vi) **Ref**: Number of references to this route.

(vii) **Use**: Count of lookups for the route.

(viii) **Iface**: Interface to which packets for this route will be sent.

(d) The options that can be used with route command are:

(i) -n: Display routing table in full numeric form.

(ii) -C: Display routing cache.

(iii) -e: Display other/more information.

(iv) -F: Display Forwarding information base.

```
(base) gunjan@gunjan-TUF:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.12.0.254     0.0.0.0         UG    100    0        0 enp2s0
10.12.0.0       0.0.0.0         255.255.192.0   U     100    0        0 enp2s0
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 virbr0
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 virbr0
(base) gunjan@gunjan-TUF:~$ route -C
Kernel IP routing cache
Source          Destination     Gateway         Flags Metric Ref    Use Iface
(base) gunjan@gunjan-TUF:~$ route -e
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
default         _gateway        0.0.0.0         UG        0 0          0 enp2s0
10.12.0.0       0.0.0.0         255.255.192.0   U         0 0          0 enp2s0
link-local      0.0.0.0         255.255.0.0     U         0 0          0 virbr0
192.168.122.0   0.0.0.0         255.255.255.0   U         0 0          0 virbr0
(base) gunjan@gunjan-TUF:~$ route -F
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         _gateway        0.0.0.0         UG    100    0        0 enp2s0
10.12.0.0       0.0.0.0         255.255.192.0   U     100    0        0 enp2s0
link-local      0.0.0.0         255.255.0.0     U     1000   0        0 virbr0
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 virbr0
```

Question 5: netstat

(a) **netstat (network statistics)** is used to display various network related information such as network connections, routing tables, interface statistics, masquerade connections, multicast memberships, etc.

(b) We can use **grep** with the output of **netstat -a** to get the list of all established connections.
Answer: netstat -at | grep -n "ESTABLISHED"

```
(base) gunjan@gunjan-TUF:~$ netstat -at | grep -n ESTABLISHED
11:tcp        0      0 gunjan-TUF:34882        sd-in-f188.1e100.n:5228 ESTABLISHED
12:tcp        0      0 gunjan-TUF:43408        64.52.120.34.bc.g:https ESTABLISHED
13:tcp        0      0 gunjan-TUF:33226        server-18-66-63-1:https ESTABLISHED
14:tcp        0      0 gunjan-TUF:49748        lb-140-82-112-25-:https ESTABLISHED
16:tcp        0      0 gunjan-TUF:41204        server-108-158-24:https ESTABLISHED
18:tcp        0      0 gunjan-TUF:35558        server-108-158-24:https ESTABLISHED
19:tcp        0      0 gunjan-TUF:33862        104.21.58.234:https     ESTABLISHED
20:tcp        0      0 gunjan-TUF:39198        whatsapp-cdn-shv-:https ESTABLISHED
21:tcp        0      0 gunjan-TUF:38034        172.17.1.1:domain       ESTABLISHED
22:tcp        0      0 gunjan-TUF:51476        stackoverflow.com:https ESTABLISHED
23:tcp        0      0 gunjan-TUF:56368        server-18-66-85-5:https ESTABLISHED
24:tcp        0      0 gunjan-TUF:55894        server-18-66-78-1:https ESTABLISHED
26:tcp        0      0 gunjan-TUF:48588        52.111.252.0:https      ESTABLISHED
28:tcp        0      0 gunjan-TUF:51582        52.109.56.78:https      ESTABLISHED
29:tcp        0      0 gunjan-TUF:43692        ec2-13-232-193-23:https ESTABLISHED
31:tcp        0      0 gunjan-TUF:58106        ec2-54-184-13-11.:https ESTABLISHED
32:tcp        0      0 gunjan-TUF:51478        stackoverflow.com:https ESTABLISHED
33:tcp        0      0 gunjan-TUF:43410        64.52.120.34.bc.g:https ESTABLISHED
35:tcp        0      0 gunjan-TUF:50202        ec2-3-108-35-59.a:https ESTABLISHED
36:tcp        0      0 gunjan-TUF:37194        server-18-66-78-1:https ESTABLISHED
```

(c) netstat -r shows the kernel routing information. The various columns are:

(i) **Destination**: Destination network or the destination host.

(ii) **Gateway**: Gateway address

(iii) **Genmask**: The netmask for the destination net; 255.255.255.255 for a host destination and 0.0.0.0 for the default route.

(iv) **Flags**:
U - indicates that the route is up
G - indicates that the route is to a gateway
H - Indicates that the destination is a fully qualified host address, rather than a network

(v) **MSS**: Default max segment size for TCP connections over this route.

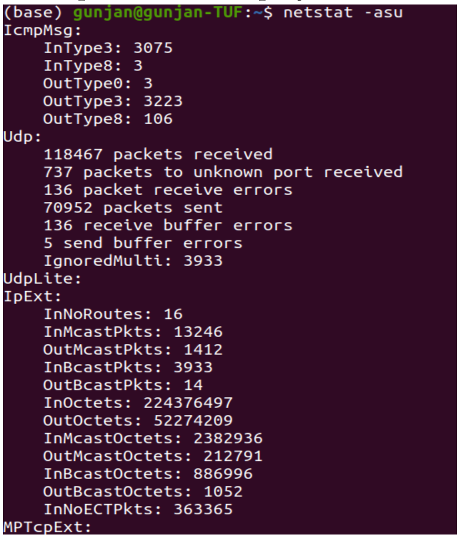
(vi) **Window**: Default window size for TCP conections over this route.

(vii) **irtt**: Initial Round Trip Time (RTT).

(viii) **Iface**: Interface to which packets for this route will be sent.

(d) The option **-ai** displays the status of all network interfaces. To find out the number of interfaces, we can use the command: **echo $[$(netstat -ai | wc -l)-2]** We calculate the number of lines in the table output, and do a **-2** to remove the table name line and column name line.

(e) The option **-asu** displays the statistics of all the UDP connections.



```
(base) gunjan@gunjan-TUF:~$ netstat -asu
IcmpMsg:
    InType3: 3075
    InType8: 3
    OutType0: 3
    OutType3: 3223
    OutType8: 106
Udp:
    118467 packets received
    737 packets to unknown port received
    136 packet receive errors
    70952 packets sent
    136 receive buffer errors
    5 send buffer errors
    IgnoredMulti: 3933
UdpLite:
IpExt:
    InNoRoutes: 16
    InMcastPkts: 13246
    OutMcastPkts: 1412
    InBcastPkts: 3933
    OutBcastPkts: 14
    InOctets: 224376497
    OutOctets: 52274209
    InMcastOctets: 2382936
    OutMcastOctets: 212791
    InBcastOctets: 886996
    OutBcastOctets: 1052
    InNoECTPkts: 363365
MPTcpExt:
```

(f) Loopback interface is a virtual interface used by the machine to communicate with itself. The loopback interface is used to identify the device. While any interface address can be used to determine if the device is online, the loopback address is the preferred method. Whereas interfaces might be removed or addresses changed based on network topology changes, the loopback address never changes. It is used for **network diagnosis and troubleshooting**.

For example, we can examine all the web documents on our web server and can view them file by file on the local machine. For IPv4, the loopback interface is assigned to all the IPs in the 127.0.0.0/8 address block.

Question 6: traceroute

Traceroute is a network diagnostic tool used to track in real-time, the pathway taken by a packet on an IP network from source to destination, reporting the IP Addresses of all the routers it pinged in between. Traceroute also records the time taken for each hop the packet makes during its route to the destination.

(a)

Sr. no	Host	Hops at 4PM	Hops at 10PM	Hops at 8AM
1	youtube.com	17	16	17
2	google.com	18	17	17
3	instagram.com	13	13	13
4	medium.com	13	13	14
5	amazon.in	timed out	timed out	timed out
6	bitcoin.org	13	11	13

The common hops are my device (192.168.37.166) and also 10.72.163.18 and 172.17.182.196 which can be the IP Address of my service provider. If the routes to the destinations pass through the same devices, then we have common hops.

(b) The route to the hosts changes at different times of the day in the experiments because of reasons like network congestion. The packets are redirected by the nodes to take a route having less traffic. Then load balancing is done to reduce the load of the congested path.

(c) Traceroute fails to find a complete path to a host sometimes, because there might be servers/hosts along the path which have not been configured to respond to the ICMP Traffic or have firewalls which block the ICMP Traffic. Under heavy loads, many network providers turn off the ICMP traffic.

(d) It is possible to find the route to certain hosts using traceroute which fail to respond with ping experiment. It depends on the implementation of the Traceroute command. If we use *tracert* (Windows), it uses ICMP with incrementing TTL field to map the hops to the final destination address.

Ping is direct ICMP from point A to point B, that traverses networks via routing rules. *tracert* works by targeting the final hop, but limiting the TTL and waiting for a time exceeded message, and then increasing by one for the next iteration. Therefore, the response it gets is not an ICMP echo reply to the ICMP echo request from the host along the way, but a time exceeded message from that host.

In UNIX based systems, *traceroute* application is used, which uses UDP packets with an incrementing TTL field to map the hops to the final destination. So if the ICMP is blocked by some network, then *Ping* and *tracert* will fail but *traceroute* from a Linux machine would succeed.

Question 7: ARP

(a) **arp -a** is used to display the complete ARP (Address Resolution Protocol) table on the machine. The columns in the output are:

(i) **Hostname**: It is the hostname. If it cannot be resolved, we get a ?

(ii) **MAC Adress**: It is a six part hexadecimal number, also known as hardware address or ethernet address.

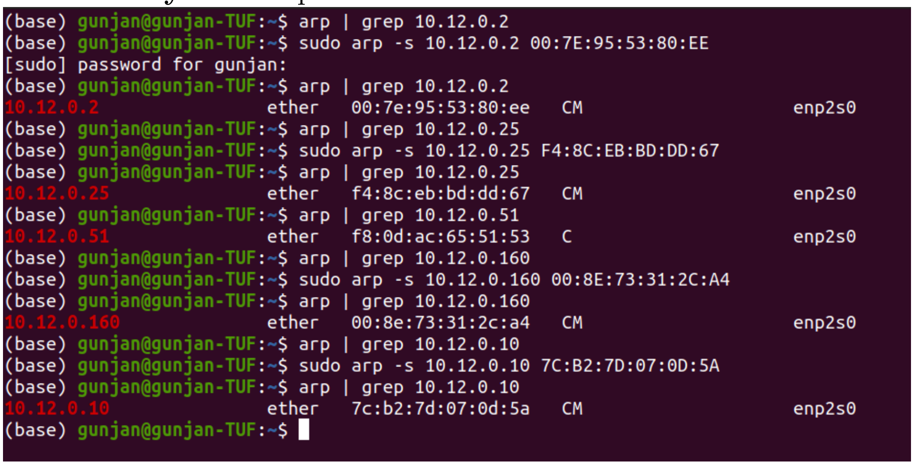
(iii) **IP Address**: IP Address of the host.

(iv) **HWtype**: It is hardware type, for eg: ether i.e. ethernet.

(v) **Flags**:
C - Complete Entry
M - Permanent Entry
P - Published Entry

(vi) **Iface**: Network Interface

(b) **Delete a entry**: sudo arp -d `<IP ADDRESS>`
Add a entry: sudo arp -s `<IP ADDRESS> <MAC ADDRESS>`



```
(base) gunjan@gunjan-TUF:~$ arp | grep 10.12.0.2
(base) gunjan@gunjan-TUF:~$ sudo arp -s 10.12.0.2 00:7E:95:53:80:EE
[sudo] password for gunjan:
(base) gunjan@gunjan-TUF:~$ arp | grep 10.12.0.2
10.12.0.2                ether   00:7e:95:53:80:ee   CM                    enp2s0
(base) gunjan@gunjan-TUF:~$ arp | grep 10.12.0.25
(base) gunjan@gunjan-TUF:~$ sudo arp -s 10.12.0.25 F4:8C:EB:BD:DD:67
(base) gunjan@gunjan-TUF:~$ arp | grep 10.12.0.25
10.12.0.25               ether   f4:8c:eb:bd:dd:67   CM                    enp2s0
(base) gunjan@gunjan-TUF:~$ arp | grep 10.12.0.51
10.12.0.51               ether   f8:0d:ac:65:51:53   C                     enp2s0
(base) gunjan@gunjan-TUF:~$ arp | grep 10.12.0.160
(base) gunjan@gunjan-TUF:~$ sudo arp -s 10.12.0.160 00:8E:73:31:2C:A4
(base) gunjan@gunjan-TUF:~$ arp | grep 10.12.0.160
10.12.0.160              ether   00:8e:73:31:2c:a4   CM                    enp2s0
(base) gunjan@gunjan-TUF:~$ arp | grep 10.12.0.10
(base) gunjan@gunjan-TUF:~$ sudo arp -s 10.12.0.10 7C:B2:7D:07:0D:5A
(base) gunjan@gunjan-TUF:~$ arp | grep 10.12.0.10
10.12.0.10               ether   7c:b2:7d:07:0d:5a   CM                    enp2s0
(base) gunjan@gunjan-TUF:~$
```

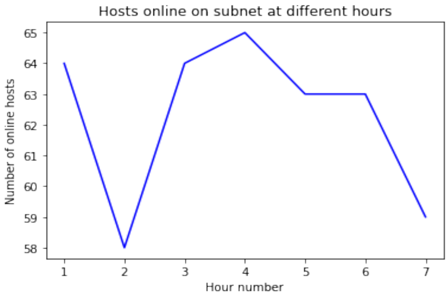
In the above figure, we have added 4 new entries to the ARP table using their IP address and MAC Address which was obtained using the nmap command (with superuser privileges).

(c) The setting **gc_stale_time** effects how often the ARP cache is checked for stale entries. Also, the parameter **base_reachble_time_ms** controls the amount of time for which an entry remains valid. Once a neighbor is found, the entry is considered valid for atleast a random value between *base_reachable_time_ms/2* and *3*base_reachable_time_ms/2*. The entry's validity can be extended if it receives positive feedback from higher level protocols. The default value is **30 seconds**.
We might use the **bisection method** and do trial-and-error. For example, if we guess the timeout value of 40 minutes and then make the system clock 40 mins faster, we can check to see if there are any changes. If the ARP cache has cleared, try 20 mins, or try a larger value if it hasn't. (*Divide the Search space by 2 every time*).
Another possible way can be to check the files at */proc/sys/net/ipv4/neigh/* and see exactly what happens to the ARP table after reaching those value in the files.

(d) If there are 2 IP Addresses mapping to the same Ethernet interface, then that interface will answer to both of the addresses. Packets to both of the addresses will be sent to the same Interface. It is very useful in a lot of cases, like hosting virtual private servers or virtual hosts without multiple domain names. This is called IP Aliasing.
Ethernet is layer 2 and IP is layer 3, and hence no real relationship with regards to host capacity and method of data transfer.

Question 8: nmap

nmap is a network mapping tool. It works by sending various network messages to the IP addressses in the range we're going to provide it with.
To check which hosts are online in our subnet, we use the following command:
nmap -sn `172.16.112.0/26`



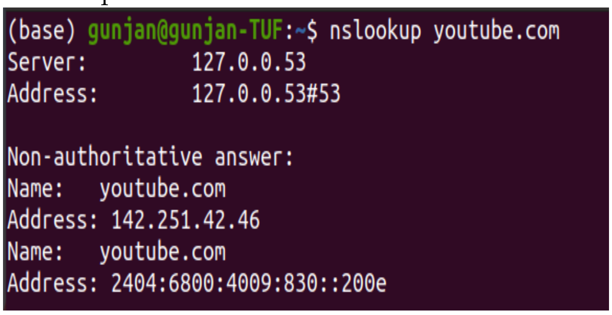
Hour number	1	2	3	4	5	6	7
Hosts online	64	58	64	65	63	63	59

We can observe that the number of hosts that are online at a given hour is always between 58 and 65.

Question 9: nslookup

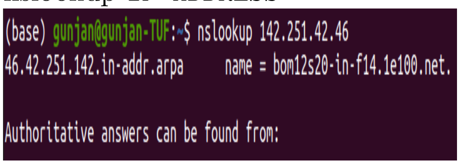
nslookup (Name Server Lookup) is a network administration tool for querying Domain Name System to obtain domain name or IP address mapping.

(a) To find IP address of a host:
nslookup `HOST NAME`



Here we see the address of the host (youtube.com)

(b) To find the domain name of an IP address:
nslookup `IP ADDRESS`



Here we can do a reverse DNS lookup.

(c) To find mail servers for a domain:
nslookup -type=mx `DOMAIN NAME`