CS 342 : Assignment - 1

Name: VISHISHT PRIYADARSHI
Roll No: 180123053

Question 1:

a) The option to specify the number of echo requests is **– c** followed by no. For example, *ping -c 5 google.com*

b) The option to set time interval (in sec) is **– i.** We further need to give the interval after specifying – i. For example, *ping - i 2 google.com*

c) The command is : **ping -l preload <destination>**. The limit for sending such packets by normal users is **3**. There are 2 more ways to do this:

i) Flood ping: Using **– f** option, we can achieve the task. If the interval is not specified, it sets interval to 0, which is required.
ii) Using – i option: We can specify the time interval as 0 after – i, to achieve the task.

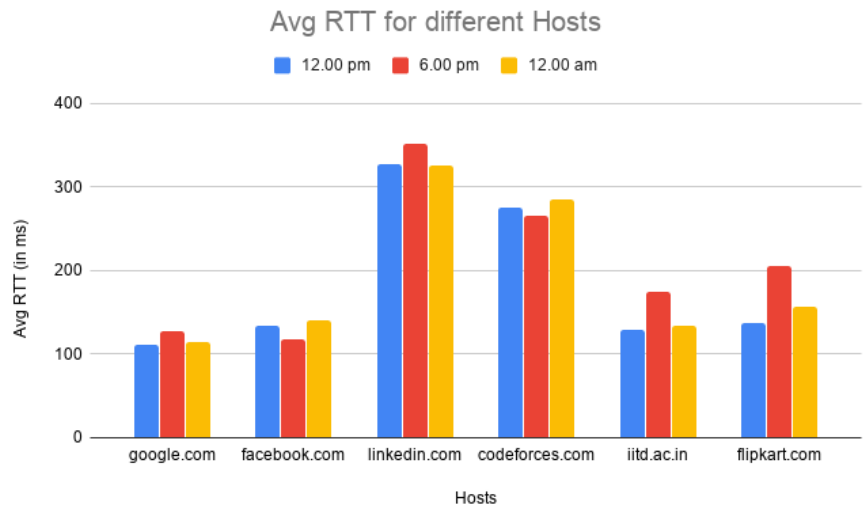
The limit for sending such packets by normal user is **0.2 s or 200 ms.**

d) The command to set the ECHO_REQUEST payload/data size (in bytes) is **– s**. The packet size is larger than the payload size due to the addition of different headers like ICMP header and IPv4 header. The total packet size, when the payload size is 32 bytes, is **40 bytes** when **ICMP header** is considered, and **60 bytes** when **IPv4 header** is also included.

Question 2:

a) The table is as follows :

Host	Location	Distance (in km)	Avg RTT (at 12 pm) (in ms)	Avg RTT (at 6 pm) (in ms)	Avg RTT (at 12 am) (in ms)	Overall Avg RTT (in ms)
google.com	California, USA	12,610	111	128	115	118
facebook.com	Dublin, Ireland	7837	133	117	140	130
linkedin.com	Kansas, USA	12,814	327	351	326	334.67
codeforces.com	Moscow, Russia	5066	275	265	285	275
iitd.ac.in	Delhi, India	1090	129	175	133	145.67
flipkart.com	Maharashtra, India	1415	137	205	156	166



It is evident from the data that there exists some **positive correlation** on the geographical distance between source and destination. This is due to the fact that larger distance increases **propagation delay** and the **number of hops** required. Also there is **node processing delay** since larger distance means more nodes to pass through. But there are several other factors too like network conditions (traffic), internet speed, server capabilities which may also affect the RTT, which is also evident from the data above *(comparing google.com and linkedin.com)*.

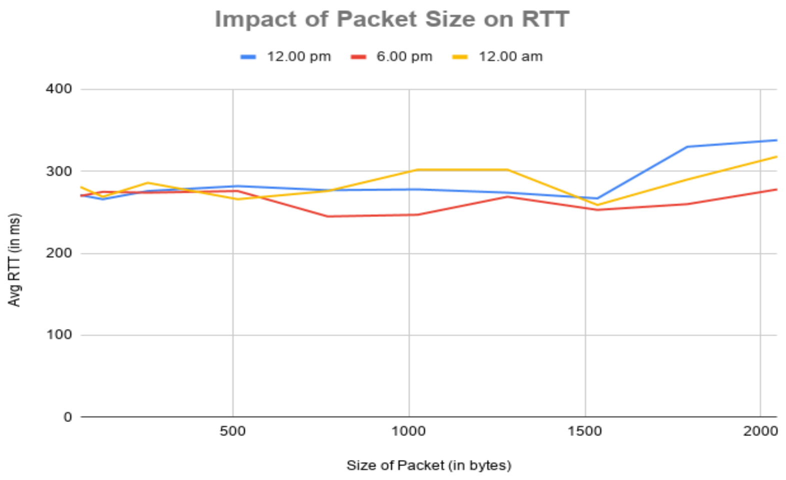
b)

Host	Packet Loss (at 12 pm)	Packet Loss (at 6 pm)	Packet Loss (at 12 am)
google.com	0 %	4 %	0 %
facebook.com	0 %	8 %	0 %
linkedin.com	0 %	0 %	0 %
codeforces.com	0 %	4 %	0 %
iitd.ac.in	0 %	4 %	0 %
flipkart.com	0 %	8 %	0 %

It is seen that around 6 pm, there are packet losses. It implies that the network traffic was higher around that time. So, due to network congestion, some packets got lost. There may be packets collision in the network due to which packets got dropped. At rest of the times, the network traffic was smooth since all packets were correctly transmitted without incurring any loss.

c) Avg RTT variations for different packet sizes : (**host:** codeforces.com)

	64 bytes	128 bytes	256 bytes	512 bytes	768 bytes	1024 bytes	1280 bytes	1536 bytes	1792 bytes	2048 bytes
12 pm	271 ms	266 ms	276 ms	282 ms	277 ms	278 ms	274 ms	267 ms	330 ms	338 ms
6 pm	270 ms	275 ms	274 ms	276 ms	245 ms	247 ms	269 ms	253 ms	260 ms	278 ms
12 am	281 ms	269 ms	286 ms	266 ms	276 ms	302 ms	302 ms	259 ms	290 ms	318 ms



d) The default MTU (Maximum Transmission Unit) is 1500 bytes. When the packets exceed this value, a single frame can't be used to transmit the packet. As a result, the packet is fragmented over multiple frames (which increase RTT). If the packet size is less than MTU, the packets are padded sufficiently to make size 1500 bytes.

This fact also agrees with the data, since upto 1500 bytes, the RTT almost remains same. And after 1500 there is steep rise in RTT values. After further increase in packet size, the RTT value increases.

At different times of the day the network traffic is different. The number of online hosts and congestion in network vary. So, this also affects the RTT values. But the overall trend is more dominated by the packet size. There are slight variations in the observed data since the data was collected using sequential pinging strategy (not parallel) and network speed may change between each ping in an unpredictable way.

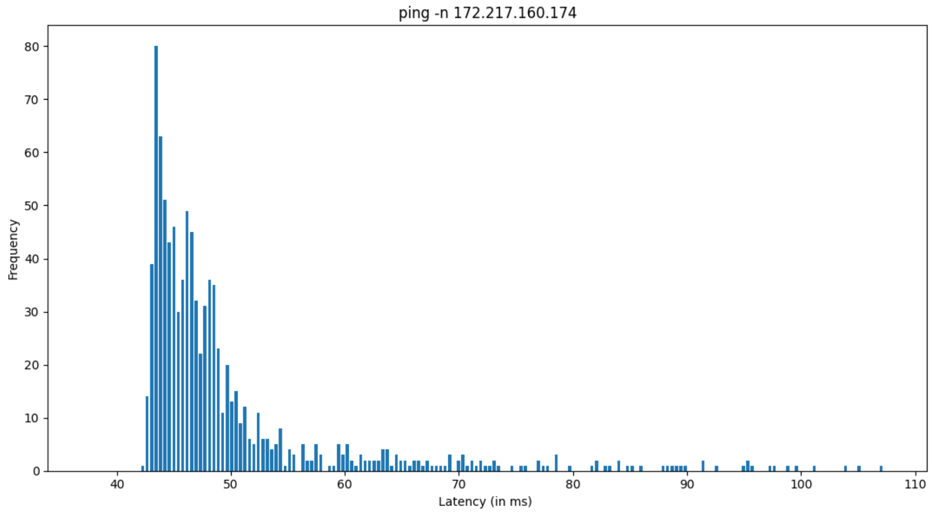
Question 3:

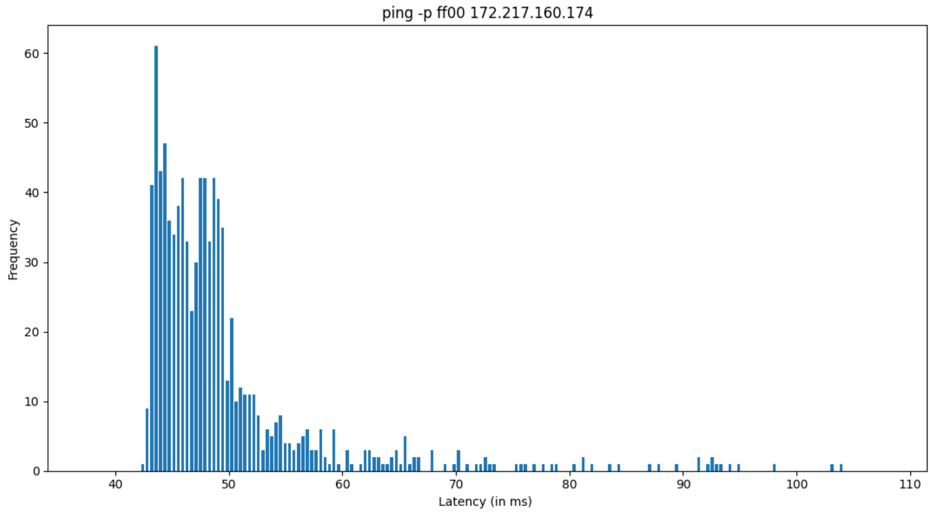
a) Packet Loss for **ping -n 172.217.160.174** is **1.2 %**, while that for **ping -p ff00 172.217.160.174** is **3.7 %**.

b) The statistics is summarised below :

	ping -n 172.217.160.174	ping -p ff00 172.217.160.174
Minimum Latency	42.4 ms	42.5 ms
Maximum Latency	820.0 ms	821.0 ms
Mean Latency	57.61 ms	58.04 ms
Median Latency	46.70 ms	47.7 ms

c) The plots are as follows *(both plots are **right-skewed**)* :





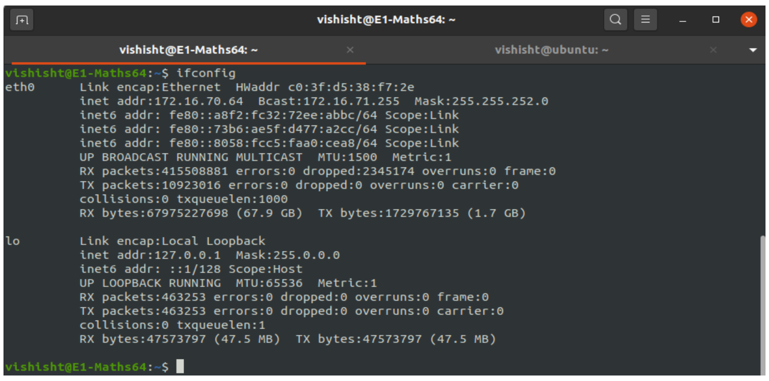
d) When the ping is used with -p flag, all the quantities including the packet loss rate **higher** than that of -n flag. Using -n flag will cause no attempt to be made to look up symbolic names for host addresses, i.e, no DNS resolution takes place. Hence, mean latency is lower in -n flag case. Also -p flag is generally used for diagnosing data-dependent problems in a network. Here, it will fill out the packets with ff00 (as specified) - 16 bytes.

Question 4:

a) The output of **ifconfig** is as follows :

There are **2 network interfaces** on the machine: **eth0** is a physical interface representing Ethernet network card while **lo** is the virtual loopback interface. **Link encap** shows how packets are encapsulated for transmission. **HWaddr** is the hardware address of the Ethernet interface. **inet addr** is IPv4 address assigned to the respective interface. **Bcast** is the broadcast address for this interface. **inet6 addr** is the IPv6 address and **Scope** is its scope. **UP** means that kernel modules for this interface is loaded and interface is activated. **BROADCAST** shows that it can handle broadcast packets. **RUNNING** shows the interface is ready to accept data. **MULTICAST** means it supports multicasting. **MTU** is maximum transmission unit. **Metric** helps in deciding the priority of the interface.

RX packets is the total number of packets received. **errors** shows number of damaged packets received. **dropped** is the number of dropped packets. **TX packets** is the total number of transmitted packets. **collisions** means the number of transmitted packets that experienced Ethernet collisions. **txqueuelen** is the length of transmission queue. **RX bytes** is the total number of bytes received over interface, and **TX bytes** is the total number of transmitted bytes over the interface.



```
vishisht@E1-Maths64:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr c0:3f:d5:38:f7:2e
          inet addr:172.16.70.64  Bcast:172.16.71.255  Mask:255.255.252.0
          inet6 addr: fe80::a8f2:fc32:72ee:abbc/64 Scope:Link
          inet6 addr: fe80::73b6:ae5f:d477:a2cc/64 Scope:Link
          inet6 addr: fe80::8058:fcc5:faa0:cea8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:415508881 errors:0 dropped:2345174 overruns:0 frame:0
          TX packets:10923016 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:67975227698 (67.9 GB)  TX bytes:1729767135 (1.7 GB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:463253 errors:0 dropped:0 overruns:0 frame:0
          TX packets:463253 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:47573797 (47.5 MB)  TX bytes:47573797 (47.5 MB)

vishisht@E1-Maths64:~$
```

b) The following options can be used with ifconfig :

i) **-a :** It displays all the interfaces which are currently available, even if they are down.

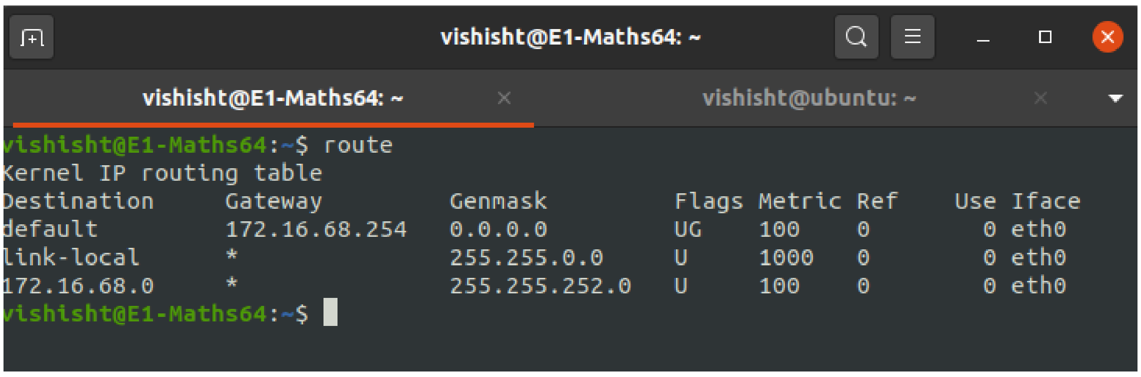
ii) **-s :** It displays a short list of interfaces. (like netstate –i).

iii) **-v :** It allows the output to be more verbose for some error condition.

iv) **[-] arp :** It enables/disables the use of the ARP protocol on this interface.

v) **mtu N :** This parameter sets the Maximum Transfer Unit (MTU) of an interface.

c) The output is as follows :



```
vishisht@E1-Maths64:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         172.16.68.254   0.0.0.0         UG    100    0        0 eth0
link-local      *               255.255.0.0     U     1000   0        0 eth0
172.16.68.0     *               255.255.252.0   U     100    0        0 eth0
vishisht@E1-Maths64:~$
```

The **Destination** column identifies the destination network. The **Gateway** column identifies the gateway for the specified network. An **asterisk (*)** appears in this column if no forwarding gateway is needed for the network. The 0.0.0.0 means that the network is locally connected on that interface and no more hops are needed to get to it. The **Genmask** column shows the netmask on the network. 0.0.0.0 in Genmask means there is no mask. Under the **Flags** section, the **U flag** means the route is up and the **G flag** means that the specified gateway should be used for this route. **Metric** is the distance to the target (usually counted in hops). **Ref** is the number of references to this route. The **IFace** column shows the network interface. **eth0** is the Ethernet device.

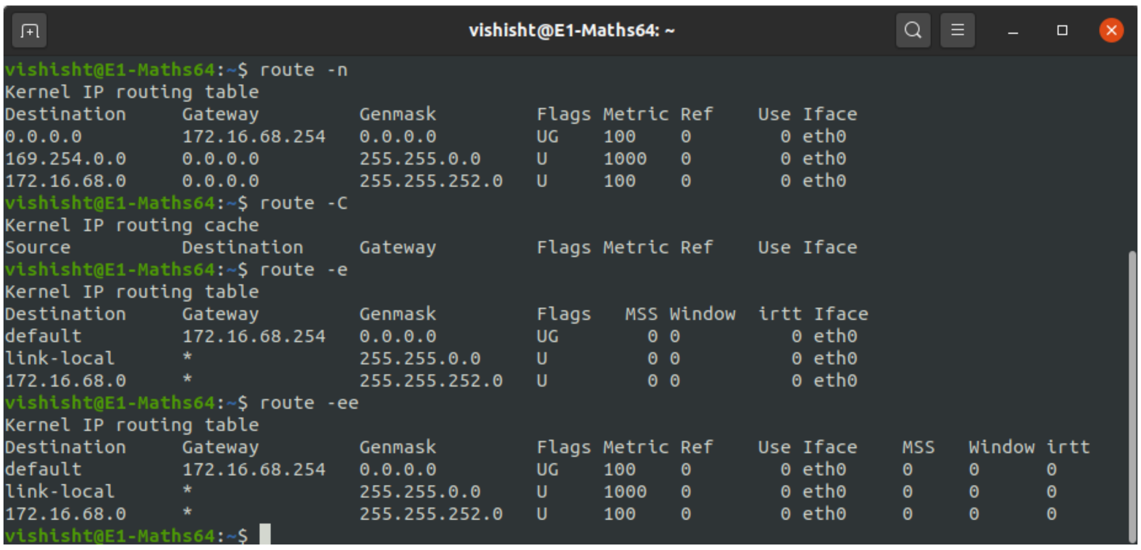
d) The following options can be used with route :

i) **- n :** It is used to display the numerical IP addresses.

ii) **- C :** It will list the kernel's routing cache information.

iii) **- e :** It will allow route command to use netstat-format for displaying the routing table.

iv) **- ee :** It will show all parameters from the routing table, generating a long line.

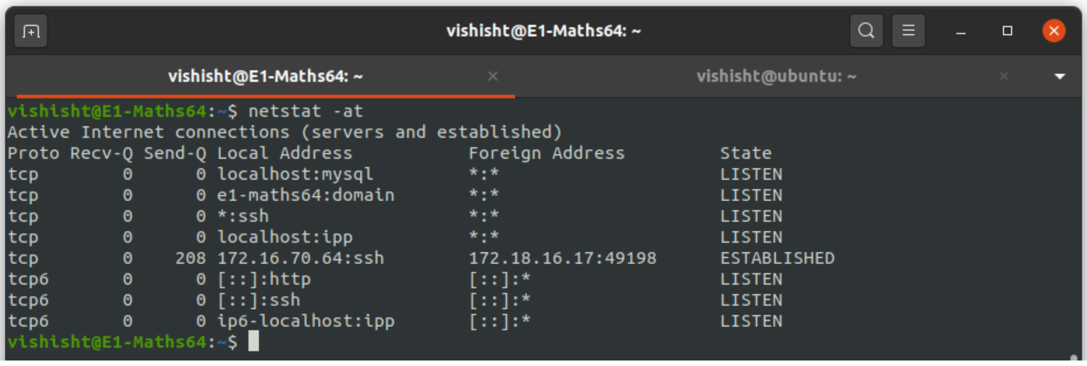


```
vishisht@E1-Maths64:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.16.68.254   0.0.0.0         UG    100    0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 eth0
172.16.68.0     0.0.0.0         255.255.252.0   U     100    0        0 eth0
vishisht@E1-Maths64:~$ route -C
Kernel IP routing cache
Source          Destination     Gateway         Flags Metric Ref    Use Iface
vishisht@E1-Maths64:~$ route -e
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
default         172.16.68.254   0.0.0.0         UG        0 0          0 eth0
link-local      *               255.255.0.0     U         0 0          0 eth0
172.16.68.0     *               255.255.252.0   U         0 0          0 eth0
vishisht@E1-Maths64:~$ route -ee
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface    MSS   Window irtt
default         172.16.68.254   0.0.0.0         UG    100    0        0 eth0     0     0      0
link-local      *               255.255.0.0     U     1000   0        0 eth0     0     0      0
172.16.68.0     *               255.255.252.0   U     100    0        0 eth0     0     0      0
vishisht@E1-Maths64:~$
```

Question 5:

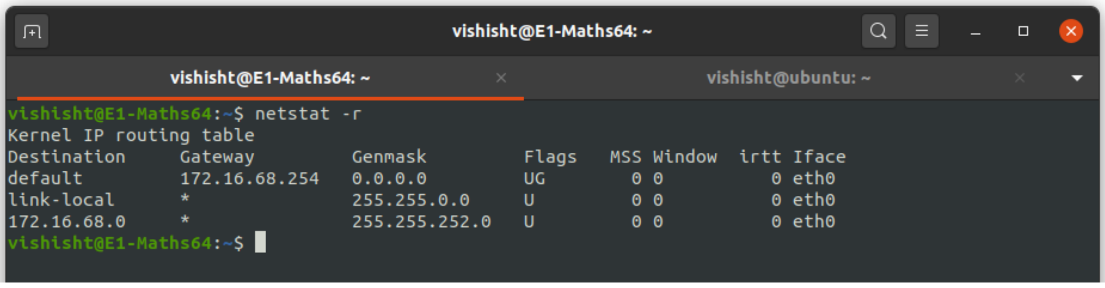
a) netstat (network statistics) is a command line tool for monitoring network connections both incoming and outgoing as well as viewing routing tables, interface statistics etc. It prints network connections, routing tables, interface statistics, masquerade connections, and multicast memberships.

b) The established TCP connections can be shown by using : **netstat -at**.



```
vishisht@E1-Maths64:~$ netstat -at
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 localhost:mysql         *:*                     LISTEN
tcp        0      0 e1-maths64:domain       *:*                     LISTEN
tcp        0      0 *:ssh                   *:*                     LISTEN
tcp        0      0 localhost:ipp           *:*                     LISTEN
tcp        0    208 172.16.70.64:ssh        172.18.16.17:49198      ESTABLISHED
tcp6       0      0 [::]:http               [::]:*                  LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
tcp6       0      0 ip6-localhost:ipp       [::]:*                  LISTEN
vishisht@E1-Maths64:~$ 
```

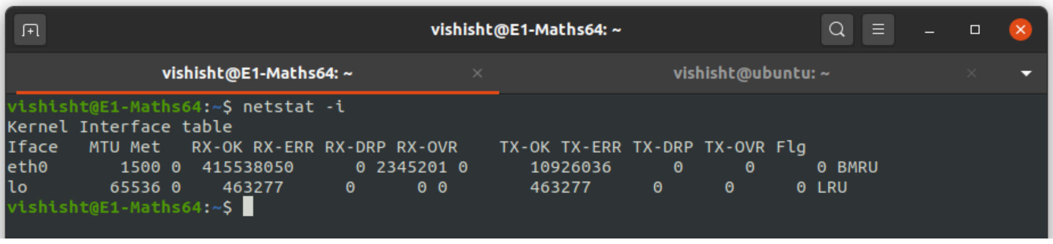
c) The output is as follows:



```
vishisht@E1-Maths64:~$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
default         172.16.68.254   0.0.0.0         UG        0 0          0 eth0
link-local      *               255.255.0.0     U         0 0          0 eth0
172.16.68.0     *               255.255.252.0   U         0 0          0 eth0
vishisht@E1-Maths64:~$ 
```

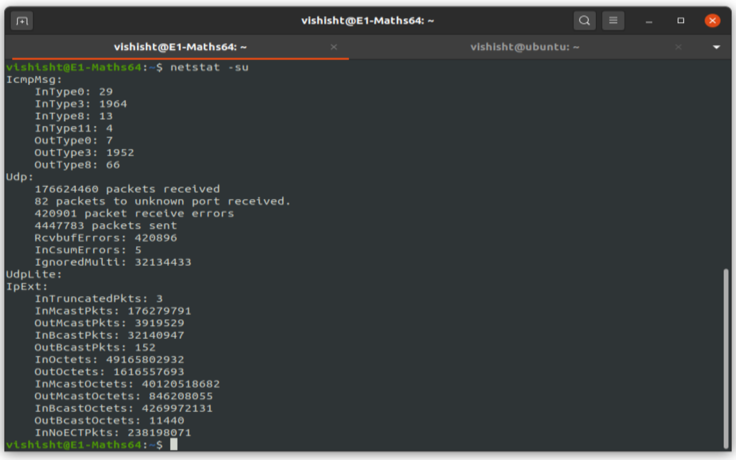
The **Destination** column identifies the destination network. The **Gateway** column identifies the gateway for the specified network. An **asterisk (*)** appears in this column if no forwarding gateway is needed for the network. The 0.0.0.0 means that the network is locally connected on that interface and no more hops are needed to get to it. The **Genmask** column shows the netmask on the network. 0.0.0.0 in Genmask means there is no mask. Under the **Flags** section, the **U flag** means the route is up and the **G flag** means that the specified gateway should be used for this route. **MSS** stands for Maximum Segment Size which is the size of the largest datagram the kernel will construct for transmission via this route. **Window** represents the maximum amount of data the system will accept in a single burst of transfer from a remote host. **irtt** stands for initial round trip time. The **IFace** column shows the network interface. **eth0** is the Ethernet device.

d) The status of all network interfaces can be found put by **netstat -i**. There are 2 interfaces on my computer – (i) **eth0**(Ethernet device), and (ii) **lo** (loopback interface).



```
vishisht@E1-Maths64:~$ netstat -i
Kernel Interface table
Iface      MTU Met    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0      1500 0   415538050      0 2345201 0      10926036      0      0      0 BMRU
lo       65536 0      463277      0      0 0        463277      0      0      0 LRU
vishisht@E1-Maths64:~$ 
```

e) The statistics of all UDP connections can be found out by executing **netstat -su**.



```
vishisht@E1-Maths64:~$ netstat -su
IcmpMsg:
    InType0: 29
    InType3: 1964
    InType8: 13
    InType11: 4
    OutType0: 7
    OutType3: 1952
    OutType8: 66
Udp:
    176624460 packets received
    82 packets to unknown port received.
    420901 packet receive errors
    4447783 packets sent
    RcvbufErrors: 420896
    InCsumErrors: 5
    IgnoredMulti: 32134433
UdpLite:
IpExt:
    InTruncatedPkts: 3
    InMcastPkts: 176279791
    OutMcastPkts: 3919529
    InBcastPkts: 32140947
    OutBcastPkts: 152
    InOctets: 49165802932
    OutOctets: 1616557693
    InMcastOctets: 40120518682
    OutMcastOctets: 846208055
    InBcastOctets: 4269972131
    OutBcastOctets: 11440
    InNoECTPkts: 238198071
vishisht@E1-Maths64:~$ 
```

f) **loopback interface** is a special virtual interface which is used by the computer to communicate with itself. When a network interface is disconnected, no communication is possible on that interface, even between the computer and itself. In such scenarios, loopback interface comes into the picture. It is also used for diagnostics purposes, troubleshooting, and to connect to the servers running on the local machine. The loopback interface does not represent an actual hardware, but is a logical, virtual interface.

Question 6:

a) **traceroute** tracks the route packets taken from an IP network on their way to a given host. It utilizes the IP protocol's TTL field find the route a packet takes to reach the host.

b) The hop counts at different time of the day for different hosts are as follows :

	google.com	facebook.com	codeforces.com	linkedin.com	iitd.ac.in	flipkart.com
12 pm	9	9	19	18	14	11
6 pm	9	9	19	18	14	11
12 am	9	9	20	18	14	11

The hops which were common to all the host were **192.168.0.1** (my IP address), **103.206.8.62** (ISP provider) and **103.206.8.61**. **14.143.172.17** was common to flipkart.com and linkedin.com. **203.192.196.30** was common to facebook.com and google.com. **10.248.2.61** was common to codeforces.com and google.com. **14.143.59.13** was common to codeforces.com and iitd.ac.in.

The hops were common because the packets travelled through the same routes for a part of their journey and so were handled by same nodes.

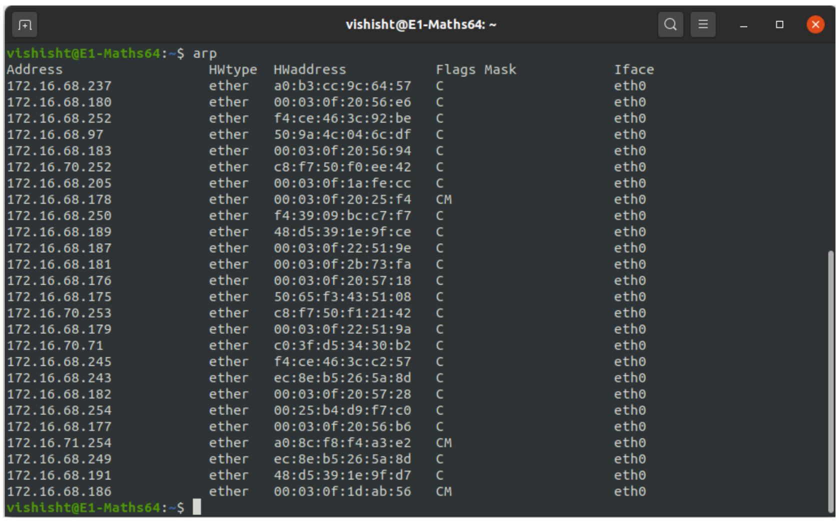
c) It is possible for the route to the hosts to change at different times of the day. It was also evident from the data collected. Due to the network congestion and traffic, the packets are redirected to the nodes with less traffic to reduce the congestion. Also destination host may utilize multiple servers to handle the incoming packets, thereby showing different IP addresses when the command is executed multiple times.

d) **traceroute** in Linux systems use UDP packets. Sometimes hosts on the path are configured to block the ICMP/UDP packets or they may have a firewall set-up which blocks the packets. As a result, they do not respond. Nevertheless, they send data to the next hops since there are nodes in the results which follows * * *. A hop that outputs * * * means that the router at that hop does not respond to the type of packet we were using for the traceroute. Such nodes are configured to prevent DoS attacks which are generated using UDP/ICMP packets. Also sometimes due to heavy huge networks traffic, the nodes are disabled for receiving these packets.

e) Yes, it is possible because **ping** uses ICMP echo requests, while **traceroute** implementations provide a wide range of protocols including ICMP echo request, TCP SYN, and UDP packets. ping is straight ICMP from point A to point B, that traverses networks via routing rules. traceroute works by targeting the final hop, but limiting the TTL and waiting for a time exceeded message, and then increasing it 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 - so even though it is using ICMP, it is using it in a very different way. Traceroute looks for the ICMP Time exceeded packet and not the ICMP Reply Packet. Hence, it is possible to discover those hosts using wide variety of protocols available with different implementations of traceroute.

Question 7:

a) The full arp table can be seen by this command: **arp -a**

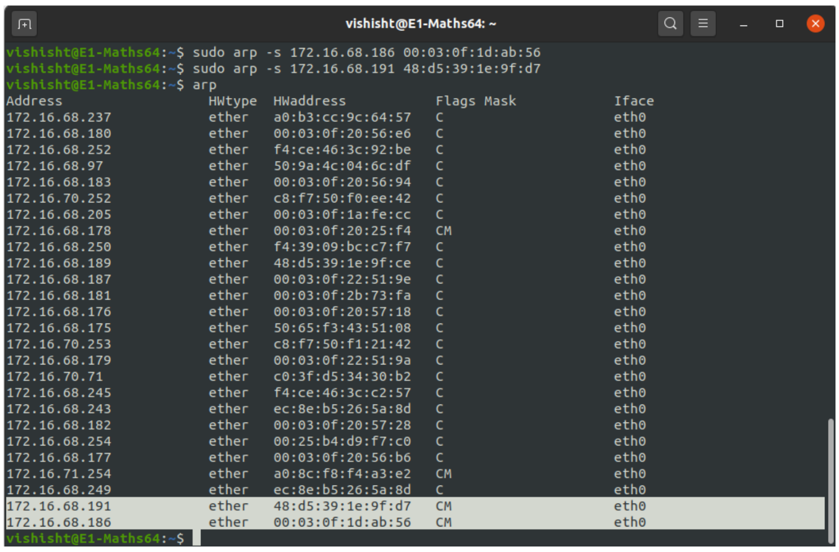


```
vishisht@E1-Maths64:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
172.16.68.237            ether   a0:b3:cc:9c:64:57   C                     eth0
172.16.68.180            ether   00:03:0f:20:56:e6   C                     eth0
172.16.68.252            ether   f4:ce:46:3c:92:be   C                     eth0
172.16.68.97             ether   50:9a:4c:04:6c:df   C                     eth0
172.16.68.183            ether   00:03:0f:20:56:94   C                     eth0
172.16.70.252            ether   c8:f7:50:f0:ee:42   C                     eth0
172.16.68.205            ether   00:03:0f:1a:fe:cc   C                     eth0
172.16.68.178            ether   00:03:0f:20:25:f4   CM                    eth0
172.16.68.250            ether   f4:39:09:bc:c7:f7   C                     eth0
172.16.68.189            ether   48:d5:39:1e:9f:ce   C                     eth0
172.16.68.187            ether   00:03:0f:22:51:9e   C                     eth0
172.16.68.181            ether   00:03:0f:2b:73:fa   C                     eth0
172.16.68.176            ether   00:03:0f:20:57:18   C                     eth0
172.16.68.175            ether   50:65:f3:43:51:08   C                     eth0
172.16.70.253            ether   c8:f7:50:f1:21:42   C                     eth0
172.16.68.179            ether   00:03:0f:22:51:9a   C                     eth0
172.16.70.71             ether   c0:3f:d5:34:30:b2   C                     eth0
172.16.68.245            ether   f4:ce:46:3c:c2:57   C                     eth0
172.16.68.243            ether   ec:8e:b5:26:5a:8d   C                     eth0
172.16.68.182            ether   00:03:0f:20:57:28   C                     eth0
172.16.68.254            ether   00:25:b4:d9:f7:c0   C                     eth0
172.16.68.177            ether   00:03:0f:20:56:b6   C                     eth0
172.16.71.254            ether   a0:8c:f8:f4:a3:e2   CM                    eth0
172.16.68.249            ether   ec:8e:b5:26:5a:8d   C                     eth0
172.16.68.191            ether   48:d5:39:1e:9f:d7   C                     eth0
172.16.68.186            ether   00:03:0f:1d:ab:56   CM                    eth0
vishisht@E1-Maths64:~$
```

Address shows the IP addresses of the network connections. **HWType** shows the type of Ethernet device (hardware) used on the machine. **HWaddress** shows the MAC address (or hardware address) for that network connection. **Flags Mask** is used to describe each entry in the ARP table, for example, M means permanent entries,P means published entries and C means complete entry. **IFace** shows the respective network interface.

b) To add an entry in ARP table, we use: **arp -s <IP_address> <MAC_address>**

To delete an entry in ARP table, we use: **arp -d <IP_address>**
But since complete deletion of an entry is expensive, the MAC address of the entry is changed to **<incomplete>** instead to invalidate the entry.



```
vishisht@E1-Maths64:~$ sudo arp -s 172.16.68.186 00:03:0f:1d:ab:56
vishisht@E1-Maths64:~$ sudo arp -s 172.16.68.191 48:d5:39:1e:9f:d7
vishisht@E1-Maths64:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
172.16.68.237            ether   a0:b3:cc:9c:64:57   C                     eth0
172.16.68.180            ether   00:03:0f:20:56:e6   C                     eth0
172.16.68.252            ether   f4:ce:46:3c:92:be   C                     eth0
172.16.68.97             ether   50:9a:4c:04:6c:df   C                     eth0
172.16.68.183            ether   00:03:0f:20:56:94   C                     eth0
172.16.70.252            ether   c8:f7:50:f0:ee:42   C                     eth0
172.16.68.205            ether   00:03:0f:1a:fe:cc   C                     eth0
172.16.68.178            ether   00:03:0f:20:25:f4   CM                    eth0
172.16.68.250            ether   f4:39:09:bc:c7:f7   C                     eth0
172.16.68.189            ether   48:d5:39:1e:9f:ce   C                     eth0
172.16.68.187            ether   00:03:0f:22:51:9e   C                     eth0
172.16.68.181            ether   00:03:0f:2b:73:fa   C                     eth0
172.16.68.176            ether   00:03:0f:20:57:18   C                     eth0
172.16.68.175            ether   50:65:f3:43:51:08   C                     eth0
172.16.70.253            ether   c8:f7:50:f1:21:42   C                     eth0
172.16.68.179            ether   00:03:0f:22:51:9a   C                     eth0
172.16.70.71             ether   c0:3f:d5:34:30:b2   C                     eth0
172.16.68.245            ether   f4:ce:46:3c:c2:57   C                     eth0
172.16.68.243            ether   ec:8e:b5:26:5a:8d   C                     eth0
172.16.68.182            ether   00:03:0f:20:57:28   C                     eth0
172.16.68.254            ether   00:25:b4:d9:f7:c0   C                     eth0
172.16.68.177            ether   00:03:0f:20:56:b6   C                     eth0
172.16.71.254            ether   a0:8c:f8:f4:a3:e2   CM                    eth0
172.16.68.249            ether   ec:8e:b5:26:5a:8d   C                     eth0
172.16.68.191            ether   48:d5:39:1e:9f:d7   CM                    eth0
172.16.68.186            ether   00:03:0f:1d:ab:56   CM                    eth0
vishisht@E1-Maths64:~$
```

c) ARP (Address Resolution Protocol) is for use within **a single network only**. Computers use it to map IP addresses to MAC addresses within a network. ARP table helps in discovering link layer address associated with an internet layer address. So, there **cannot be an entry** from different subnet in ARP table of my PC. Yet there is a concept of ARP proxy in which a device on a given network answers the ARP queries for an IP address that is not on that network.

d) After performing the given steps, the IP whose Ethernet Address was changed completely failed to respond to the pings, resulting in 100 % packet loss while the other IP responded.
When ping is sent from one device to another in the network, the destination IP address must be resolved to MAC address for transmission in data link layer. To achieve this, a broadcast packet is sent out in the network, known as **ARP request**. The destination machine with the IP in the ARP request then responds with an **ARP reply** that contains the MAC address for that IP. When 2 devices share the same MAC address, it creates confusion in the network. MAC address is supposed to be unique since it identifies the hardware in a network. Due to the tampering with the MAC address (Ethernet address), the packets were unable to reach that particular host, since its MAC address was not available in the ARP table and hence, it failed to respond.

Question 8:

a) The command to check which PCs in sub-net are up is: **nmap -sP <subnet_address>**

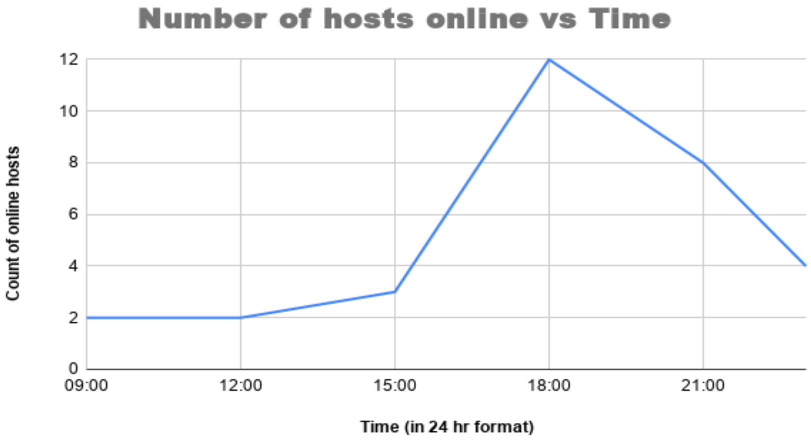
In my case, it was : **nmap -sP 192.168.0.0/24**

b) The command to detect firewall setting is : **nmap -sA <IP_address>**

It will detect whether the packets can pass through the firewall unfiltered (ACK Scan). We can also perform SYN scan with **nmap -sS <IP_address>** .

c) The data is as follows :

Time	9 am	12 pm	3 pm	6 pm	9 pm	11 pm
No of hosts online	2	2	3	12	8	4



It can be deduced that the number of online hosts are less in the morning and the count increases as the day progresses with maximum hosts online during evening around 6 pm. Thereafter, the number of online hosts decreases.