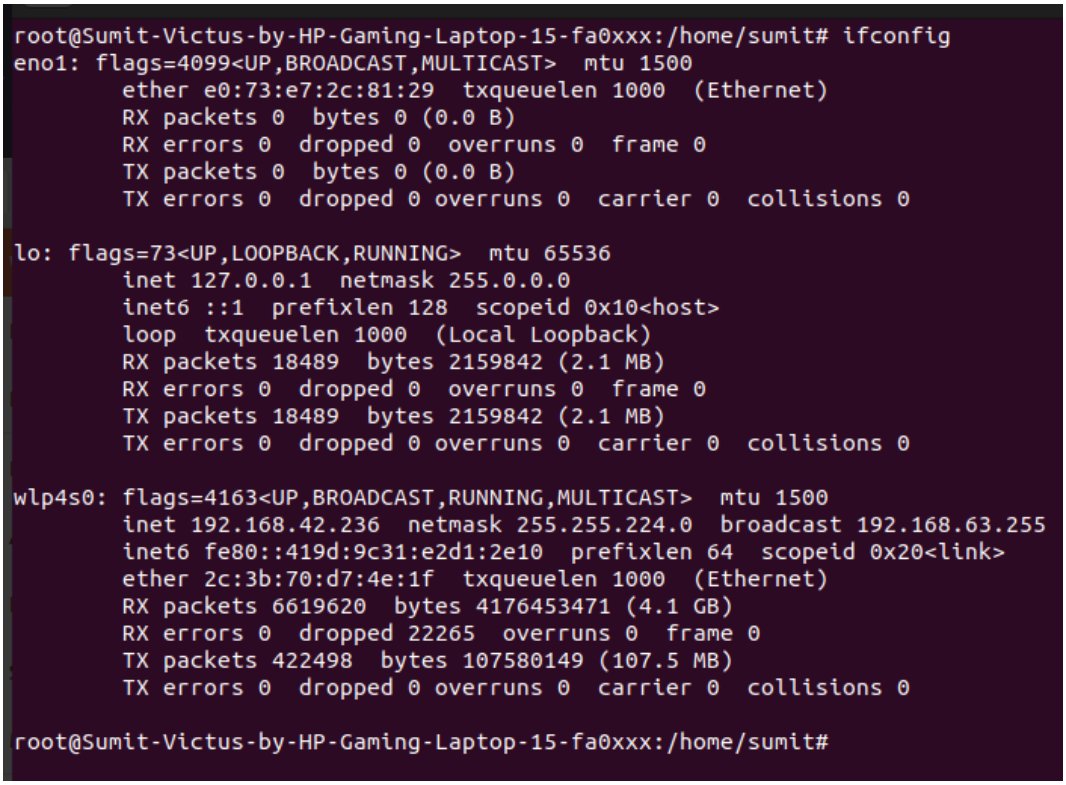
Computer Network - Assignment - 1

Q1. a) Learn to use the ifconfig command, and figure out the IP address of your network interface. Put a screenshot.

→ The *ifconfig* command is used to configure and display network interfaces in Linux. The IP address listed under *inet* for the relevant interface represents your device's internal IP within your local network. This address allows devices within the same network to communicate.

Running the command '*ifconfig*' in linux terminal.

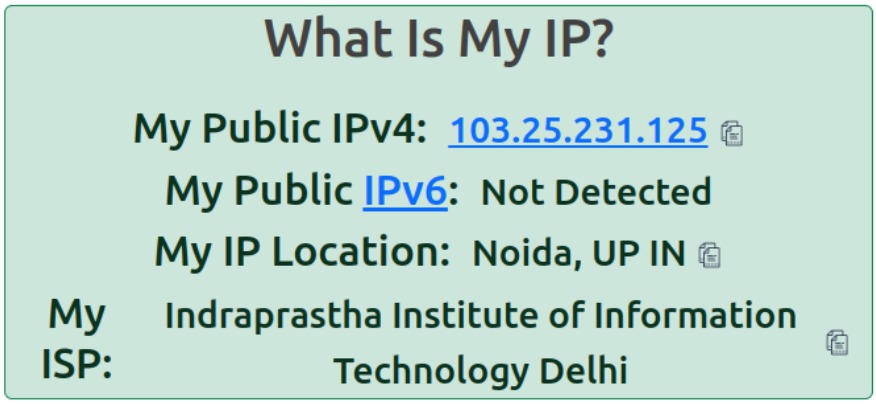


My IP address was found as 192.168.42.236.

b) Go to the webpage https://www.whatismyip.com and find out what IP is shown for your machine. Are they identical or different? Why?

→ The IP address shown on whatismyip.com is our public IP address, which is assigned by our Internet Service Provider (ISP) and is visible to the outside world. The IP address shown by *ifconfig* is your local IP address within our network. They are different because our local IP is

used within your private network, and the public IP is used for communication on the broader internet.



Q.2. a) Change the IP address of your network interface using the command line. Put a screenshot that shows the change. Revert to the original IP address.

→ The following command temporarily changes the IP address of your network interface, which gets reset after a reboot. My initial IP was 192.168.42.236, changing to 10.0.0.1 now.

sudo ifconfig <interface> <new_IP_address>

```
sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$ sudo ifconfig wlp4s0 10.0.0.1
[sudo] password for sumit:
sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$ ifconfig
eno1: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether e0:73:e7:2c:81:29  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 24410  bytes 2759150 (2.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 24410  bytes 2759150 (2.7 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp4s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.1  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::419d:9c31:e2d1:2e10  prefixlen 64  scopeid 0x20<link>
        ether 2c:3b:70:d7:4e:1f  txqueuelen 1000  (Ethernet)
        RX packets 10262646  bytes 6333874849 (6.3 GB)
        RX errors 0  dropped 33577  overruns 0  frame 0
        TX packets 575607  bytes 146734770 (146.7 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$ sudo ifconfig wlp4s0 down
sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$ ifconfig
eno1: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether e0:73:e7:2c:81:29  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 24859  bytes 2792944 (2.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 24859  bytes 2792944 (2.7 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$ sudo ifconfig wlp4s0 up
sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$ ifconfig
eno1: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether e0:73:e7:2c:81:29  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 25143  bytes 2813692 (2.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 25143  bytes 2813692 (2.8 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp4s0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.42.236  netmask 255.255.224.0  broadcast 192.168.63.255
        inet6 fe80::4fff:a76a:58d4:480a  prefixlen 64  scopeid 0x20<link>
        ether 2c:3b:70:d7:4e:1f  txqueuelen 1000  (Ethernet)
        RX packets 10264558  bytes 6334391112 (6.3 GB)
        RX errors 0  dropped 33587  overruns 0  frame 0
        TX packets 575618  bytes 146735373 (146.7 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The original IP address is restored as the change was temporary.

The IP address can be permanently changed, for which we need to edit the network configuration file.

Q.5. a.)
Run the command, traceroute google.in. How many intermediate hosts do you see? What are the IP addresses? Compute the average latency to each intermediate host. Put a screenshot. Note that some of the intermediate hosts might not be visible; their IP addresses will come as " *** ", ignore those hosts for this assignment.

→ Command used : *" traceroute google.in "*

```
root@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:/home/sumit# traceroute google.in
traceroute to google.in (142.250.193.4), 30 hops max, 60 byte packets
 1  192.168.32.254 (192.168.32.254)  22.807 ms  22.774 ms  22.756 ms
 2  auth.iiitd.edu.in (192.168.1.99)  10.426 ms  10.405 ms  10.361 ms
 3  103.25.231.1 (103.25.231.1)  23.821 ms  23.777 ms  23.725 ms
 4  * * *
 5  10.119.234.162 (10.119.234.162)  23.567 ms  23.555 ms  23.531 ms
 6  72.14.195.56 (72.14.195.56)  23.533 ms  13.273 ms 72.14.194.160 (72.14.194.160)  13.247 ms
 7  142.251.54.111 (142.251.54.111)  26.558 ms 192.178.80.159 (192.178.80.159)  100.952 ms  100.884 ms
 8  142.251.54.87 (142.251.54.87)  100.852 ms 142.251.54.89 (142.251.54.89)  88.401 ms 142.251.54.87 (142.251.54.87)  88.362 ms
 9  del11s14-in-f4.1e100.net (142.250.193.4)  88.397 ms  88.333 ms  88.368 ms
```

Upon trying traceroute command to google.in, we encountered 9 hops. If we argue that destination does not count to intermediary hosts, it becomes 8. Among the 8, one host was hidden as seen by that *** thing. As said in qn ignoring that.

IP addresses of the hops are:

1. 192.168.32.254
2. 192.168.1.99
3. 103.25.231.1
4. * * * (hidden / unresponsive)
5. 10.119.234.162
6. 72.14.195.56 / 72.14.194.160
7. 142.251.54.111 / 192.178.80.159
8. 142.251.54.87 / 142.251.54.89
9. 142.250.193.4 (The final destination)

Average latencies of each visible ntermediate hosts:

1. 192.168.32.254 (22.807 + 22.774 + 22.756)/3 = **22.779**ms
2. 192.168.1.99 (10.426 + 10.405 + 10.361)/3 = **10.397** ms
3. 103.25.231.1 (23.821 + 23.777 + 23.725)/3 = **23.635** ms
4. 10.119.234.162 (23.567 + 23.555 + 23.531)/3 = **23.551** ms
5. 72.14.195.56 / 72.14.194.160 (23.533 + 13.273 + 13.247)/3 = **16.684** ms
6. 142.251.54.111 / 192.178.80.159 (26.558 + 100.952 + 100.884/3 = **76.131** ms

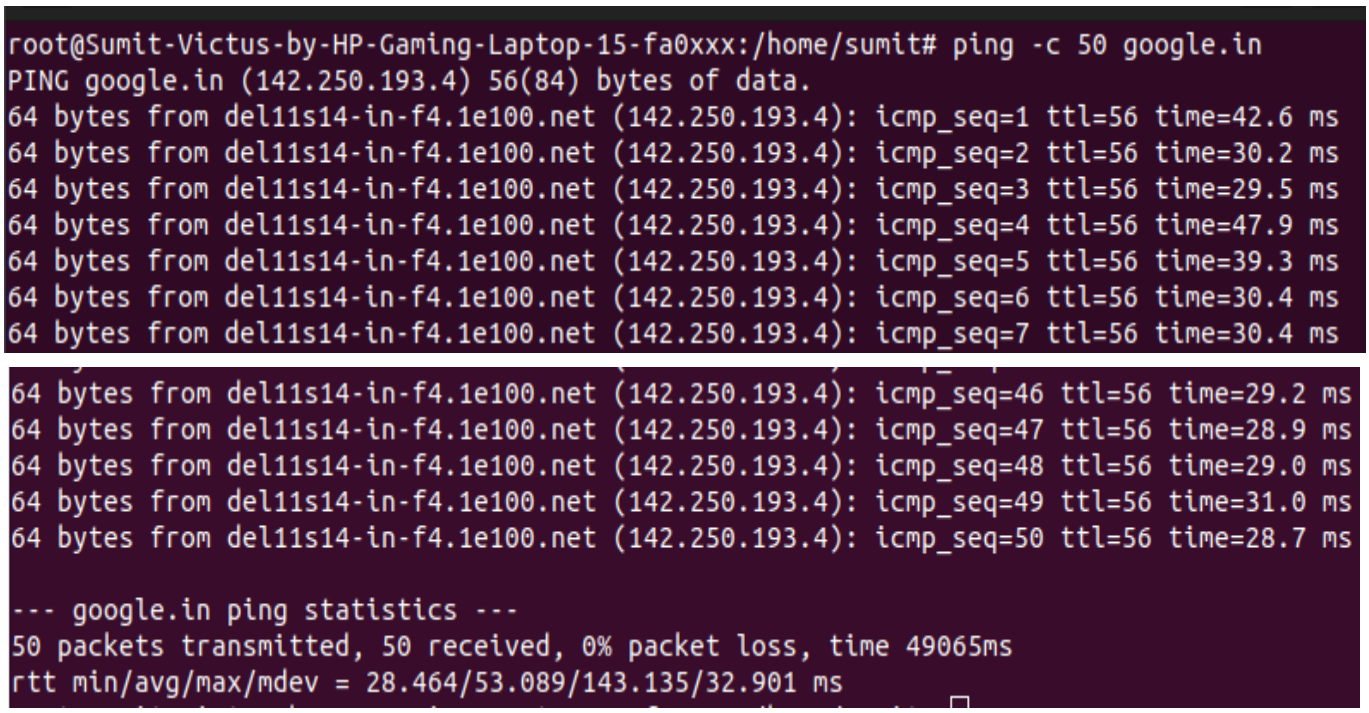
7. 142.251.54.87 / 142.251.54.89 (100.852 + 88.401+ 88.362)/3 = **92.538** ms

Destination avg latency 142.250.193.4 (88.397 + 88.333 + 88.368) = **88.364** ms

Sum of ping latency of all visible intermediate hosts : **265.715** ms

b) Send 50 ping messages to google.in, Determine the average latency. Put a screenshot.

Command used : *ping -c 50 google.in*



The statistics is given in the last line where avg. latency is shown by avg rtt(round trip time). According to which, the average latency is found out to be 53.089 ms.

c) Add up the ping latency of all the intermediate hosts obtained in (a) and compare with (b). Are they matching, explain?

→ No, they are not matching. The summed latency from traceroute is not directly comparable to the latency reported by ping because it aggregates the latencies of all hops, which include return times, whereas ping measures the direct round-trip time between your machine and the destination, which might be less than the summed latency of intermediate hops.

d) Take the maximum ping latency amongst the intermediate hosts (in (a)) and compare it with (b). Are they matching, explain? [1+1]

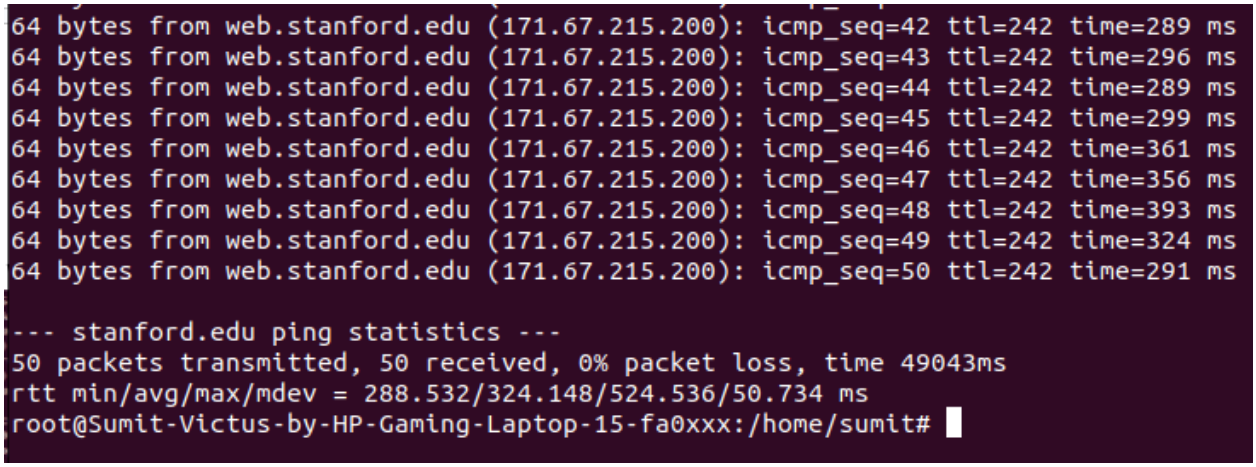
→ The maximum latency in traceroute usually reflects the slowest link in the path. If this matches the ping latency, it could indicate that this hop is the most significant source of delay.

e) You may see multiple entries for a single hop while using the traceroute command. What do these entries mean?

Yes this happens, we do see multiple entries for a single hop while using the traceroute command. This is because by default traceroute sends three packets to each hop. So, there is no guarantee that each packet will be routed in exactly the same way, they can take different paths due to network conditions (congestion or load balancing) or routing changes. In this case, the secons and third packets ended up taking slightly different paths.

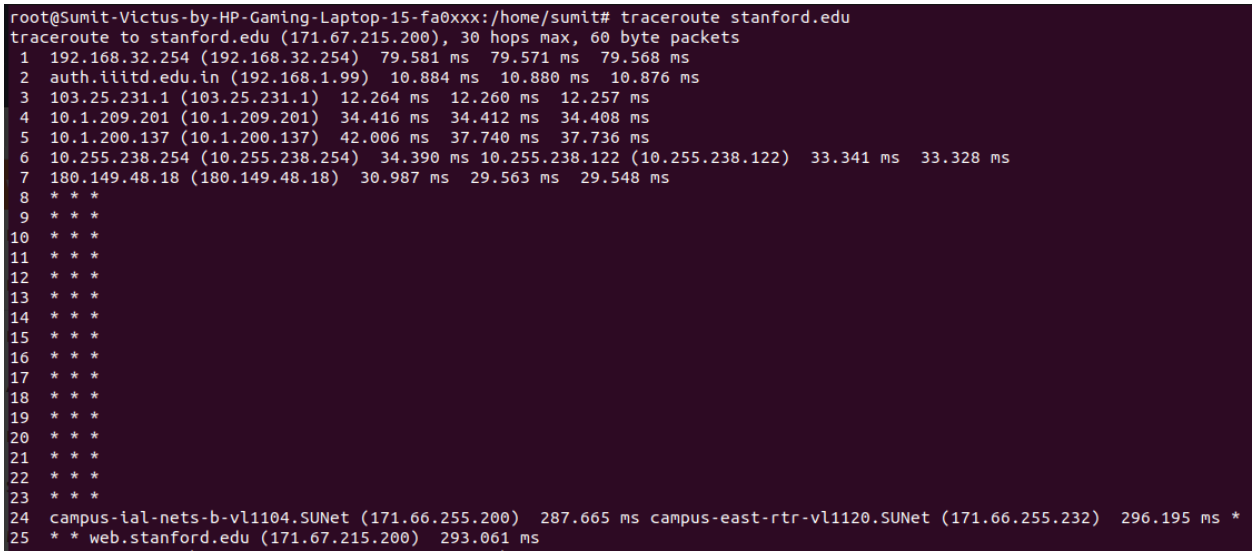
f) Send 50 ping messages to stanford.edu, Determine the average latency. Put a screenshot.

Command used : ping -c 50 stanford.edu



Again, the average latency is given by rtt (avg) i.e **324.148 ms.**

g) Run the command, traceroute stanford.edu. Compare the number of hops between google.in and stanford.edu (between the traceroute result of google.in and stanford.edu).



Number of hops in google.in was 9, while in stanford.edu has 25.

h) Can you explain the reason for the latency difference between google.in and stanford.edu (see (b) & (f)) ?

→ Latency differences arise because ‘google.in’ and ‘stanford.edu’ are hosted on different networks, have different geographical locations, and may have distinct levels of server optimization and network paths.

Q.3.
a) Use “netcat” to set up a TCP client/server connection between your VM and host machine. If you are not using a VM, you can set up the connection with localhost. Put a screenshot.

→ Here, we used ‘*netcat*’ to set up the TCP client/server connection. I have not been using VM, so set up the communication between two different terminals.

```
root@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:/home/sumit# nc -l 12345
SERVER: Hi
CLIENT: HI, there
```

```
sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$ nc localhost 12345
SERVER: Hi
CLIENT: HI, there
```

b) Determine the state of this TCP connection(s) at the client node. Put a screenshot.

→ The TCP connection states are part of the TCP state machine, which controls the establishment and termination of a connection. Common states include *LISTEN*, *ESTABLISHED*, and *CLOSE_WAIT*. We can observe these states using `netstat` to verify that the connection is active.

```
sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$ netstat -an | grep 12345
tcp        0      0 0.0.0.0:12345           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:12345         127.0.0.1:52216         ESTABLISHED
tcp        0      0 127.0.0.1:52216         127.0.0.1:12345         ESTABLISHED
```

The connection is tested on a different terminal than client or server, to check if communication is established. If the state is asked at the client node, the following shows :

```
sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$ nc localhost 12345
hi
Client : Its client here
Server : Its server this side
^C
sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$ netstat -an | grep 12345
tcp        0      0 127.0.0.1:36874         127.0.0.1:12345         TIME_WAIT
sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$
```

Q.4. nslookup
a) Get an authoritative result for “google.in” using nslookup. Put a screenshot. Explain how you did it.

→ The `nslookup` command queries DNS servers to find the IP address associated with a domain name. An authoritative answer comes directly from the DNS server responsible for the domain. This server is the definitive source of information for the domain.

```
sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$ nslookup -type=soa google.in
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
google.in
        origin = ns1.google.com
        mail addr = dns-admin.google.com
        serial = 669259768
        refresh = 900
        retry = 900
        expire = 1800
        minimum = 60

Authoritative answers can be found from:
ns1.google.com  internet address = 216.239.32.10
ns1.google.com  has AAAA address 2001:4860:4802:32::a
```

```
sumit@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:~$ nslookup -type=soa google.in ns1.google.com
Server:         ns1.google.com
Address:        216.239.32.10#53

google.in
        origin = ns1.google.com
        mail addr = dns-admin.google.com
        serial = 668858537
        refresh = 900
        retry = 900
        expire = 1800
        minimum = 60
```

I attempted to obtain authoritative results for nslookup google.in using the -type=soa flag (which stands for Start of Authority). It indicated that authoritative answers are available from ns1.google.com. Next, I ran nslookup on the DNS server ns1.google.com for google.com to retrieve authoritative results from Google.

Non-authoritative responses come from a DNS cache or resolver and are not sourced from the original authoritative server. In contrast, authoritative responses come directly from the server responsible for the domain, providing the most accurate and current information.

b) Find out the time to live for any website on the local DNS. Put a screenshot. Explain in words (with unit) after how much time this entry would expire from the local DNS server.

→The time to live (TTL) value is the amount of time in seconds that a DNS record is cached by DNS servers and clients before it needs to be refreshed. After this period, the DNS server will discard the cached record and query the authoritative server again for updated information.

When tried in google.in, we get :

184 seconds for the IPv4 address (A record).

300 seconds for the IPv6 address (AAAA record).

Q.6.) Make your ping command fail for 127.0.0.1 (with 100% packet loss). Explain how you do it. Put a screenshot that it failed.

→ 127.0.0.1 is the ip address for *'lo'* - loop back interface . Normally pinging to it, gets the packets successfully passed. However, in this qn, we want the ping command to fail with 100% packet loss. For this, the lo network interface is sent down using *'ifconfig lo down'* . Now pinging lo results to 100% packets loss.

```
root@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:/home/sumit# ifconfig lo down
root@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:/home/sumit# ifconfig
eno1: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether e0:73:e7:2c:81:29  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

wlp4s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.42.236  netmask 255.255.224.0  broadcast 192.168.63.255
        inet6 fe80::419d:9c31:e2d1:2e10  prefixlen 64  scopeid 0x20<link>
        ether 2c:3b:70:d7:4e:1f  txqueuelen 1000  (Ethernet)
        RX packets 8483332  bytes 5248855614 (5.2 GB)
        RX errors 0  dropped 28942  overruns 0  frame 0
        TX packets 504444  bytes 130862509 (130.8 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:/home/sumit# ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
^C
--- 127.0.0.1 ping statistics ---
8 packets transmitted, 0 received, 100% packet loss, time 7207ms

root@Sumit-Victus-by-HP-Gaming-Laptop-15-fa0xxx:/home/sumit#
```