

Computer Network - Assignment 1

Aditya Gupta(2022031)

20 August 2024

1 Question - 1

1.1 Part (A)

- In this question, we are asked to learn about the **ifconfig** command and figure out the IP address of our network interface . When we write the command **ifconfig -a** , we get all the network interfaces , the screenshot for the same is as follows :

```
aduop@LAPTOP-ULPEDJ2F:~$ ifconfig -a
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 172.28.140.121  netmask 255.255.240.0  broadcast 172.28.143.255
    inet6 fe80::215:5dff:fe83:1f34  prefixlen 64  scopeid 0x20<link>
    ether 00:15:5d:83:1f:34  txqueuelen 1000  (Ethernet)
    RX packets 3219  bytes 1102390 (1.1 MB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 451  bytes 76504 (76.5 KB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    inet6 ::1  prefixlen 128  scopeid 0x10<host>
    loop txqueuelen 1000  (Local Loopback)
    RX packets 383  bytes 36565 (36.5 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 383  bytes 36565 (36.5 KB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
```

Figure 1: Ifconfig Command

Thus using the Ifconfig command , we get the Ip address as **172.28.140.121** in Ipv4 format as it is written in front of inet. The Ip address in the Ipv6 format is **fe80::215:5dff:fe83:1f34** as it is written in front of inet6 .

- Now , we can see that there are two networks are showcased :

```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 172.28.140.121  netmask 255.255.240.0  broadcast 172.28.143.255
    inet6 fe80::215:5dff:fe83:1f34  prefixlen 64  scopeid 0x20<link>
    ether 00:15:5d:83:1f:34  txqueuelen 1000  (Ethernet)
    RX packets 3219  bytes 1102390 (1.1 MB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 451  bytes 76504 (76.5 KB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
```

From this one we got the IP address for our machine . Also , we have another network interface which is :

```
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 383 bytes 36565 (36.5 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 383 bytes 36565 (36.5 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

This is loop back address(localhost) . This is used for redirecting the traffic to the system .

1.2 Part (B)

The Ip Address shown on the website <https://www.whatismyip.com/> is **103.25.231.126** . The screenshot for the same is as follows :

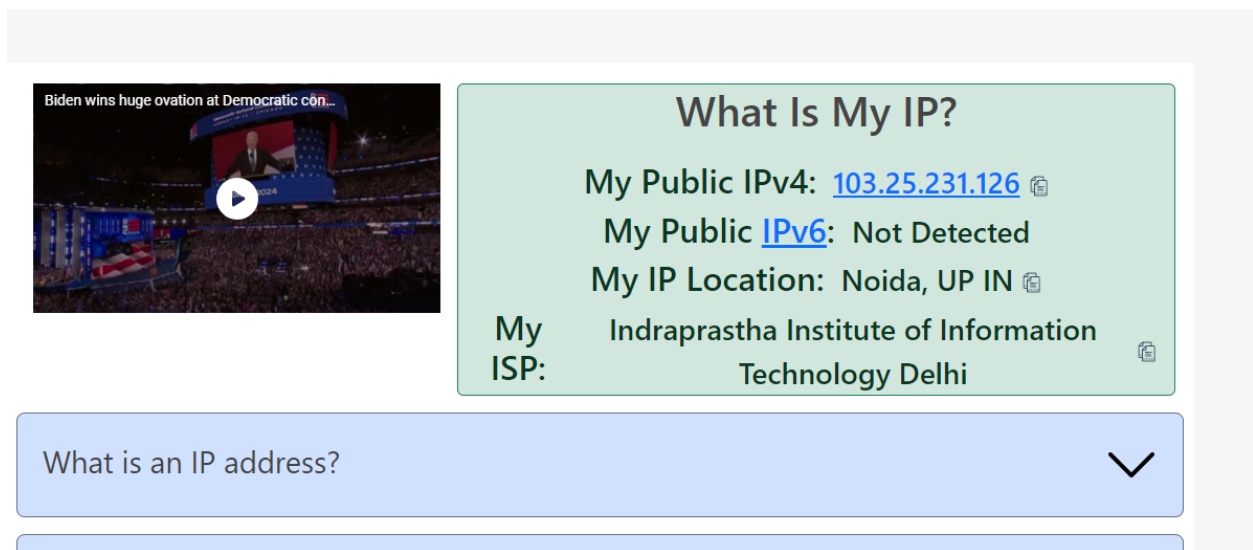


Figure 2: <https://www.whatismyip.com/>

The Ip address found in part (a) and part (b) are **different** . This is because the ip address shown by the website is the public IP address and the IP address shown by the system is the private IP address (which we found in part (a)) . The **public IP** address is **globally** available .

- The private IP address of my machine is **172.28.140.121 (Ipv4)** as found in part (a) . This address is used to communicate with the systems on the local network .
- The public IP address of my machine is **103.25.231.126** as found in part (b) . This address is visible on the internet and it is used to identify the network globally on Wide Area Network . We can get the public ip address using the command **curl ifconfig.me** to get the ip shown on the website on the terminal also .

```
aduop@LAPTOP-ULPEDJ2F:~$ curl ifconfig.me
103.25.231.126aduop@LAPTOP-ULPEDJ2F:~$ |
```

Figure 3: Public IP on terminal

2 Question - 2

2.1 Part (A)

In this question , we are asked to change the IP address of the network interface using the command line . Now , we have the network interface **eth0** . Its initial IP address is **172.28.140.121 (Ipv4)** which we get by using **ifconfig eth0** command . The screenshot of the result is as shown :

```
aduop@LAPTOP-ULPEDJ2F:~$ ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.28.140.121 netmask 255.255.240.0 broadcast 172.28.143.255
    inet6 fe80::215:5dff:fe83:1571 prefixlen 64 scopeid 0x20<link>
    ether 00:15:5d:83:15:71 txqueuelen 1000 (Ethernet)
    RX packets 6195 bytes 1527863 (1.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 123 bytes 17787 (17.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

aduop@LAPTOP-ULPEDJ2F:~$ |
```

Figure 4: Initial_IP_address

Now , we can change the IP address of this network interface using the command **sudo ifconfig eth0 172.28.140.40** . We can see that the IP address of the network interface changes . The screenshot for the same is as follows :

```
aduop@LAPTOP-ULPEDJ2F:~$ ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.28.140.121 netmask 255.255.0.0 broadcast 172.28.255.255
    inet6 fe80::215:5dff:fe83:1571 prefixlen 64 scopeid 0x20<link>
    ether 00:15:5d:83:15:71 txqueuelen 1000 (Ethernet)
    RX packets 6261 bytes 1541226 (1.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 123 bytes 17787 (17.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

aduop@LAPTOP-ULPEDJ2F:~$ sudo ifconfig eth0 172.28.140.40
aduop@LAPTOP-ULPEDJ2F:~$
aduop@LAPTOP-ULPEDJ2F:~$ ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.28.140.40 netmask 255.255.0.0 broadcast 172.28.255.255
    inet6 fe80::215:5dff:fe83:1571 prefixlen 64 scopeid 0x20<link>
    ether 00:15:5d:83:15:71 txqueuelen 1000 (Ethernet)
    RX packets 6261 bytes 1541226 (1.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 123 bytes 17787 (17.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

aduop@LAPTOP-ULPEDJ2F:~$ |
```

Figure 5: Changed Ip address(in front of inet)

Thus , the IP address of the network interface changes to **172.28.140.40** , which we gave as an argument in command above .

Now to revert to the original IP address , we can use the same command which we used above and give the argument of the `new_address` as the previous IP address and revert back the changes . Thus we use the command **sudo ifconfig eth0 172.28.140.121** , thus the IP address again changes to **172.28.140.121** , which was the old IP address . The screenshot for the same is as follows :

```
aduop@LAPTOP-ULPEDJ2F:~$ sudo ifconfig eth0 172.28.140.40
aduop@LAPTOP-ULPEDJ2F:~$
aduop@LAPTOP-ULPEDJ2F:~$ ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.28.140.40 netmask 255.255.0.0 broadcast 172.28.255.255
    inet6 fe80::215:5dff:fe83:1571 prefixlen 64 scopeid 0x20<link>
    ether 00:15:5d:83:15:71 txqueuelen 1000 (Ethernet)
    RX packets 6261 bytes 1541226 (1.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 123 bytes 17787 (17.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

aduop@LAPTOP-ULPEDJ2F:~$ sudo ifconfig eth0 172.28.140.121
aduop@LAPTOP-ULPEDJ2F:~$
aduop@LAPTOP-ULPEDJ2F:~$ ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.28.140.121 netmask 255.255.0.0 broadcast 172.28.255.255
    inet6 fe80::215:5dff:fe83:1571 prefixlen 64 scopeid 0x20<link>
    ether 00:15:5d:83:15:71 txqueuelen 1000 (Ethernet)
    RX packets 6378 bytes 1564174 (1.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 124 bytes 17857 (17.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

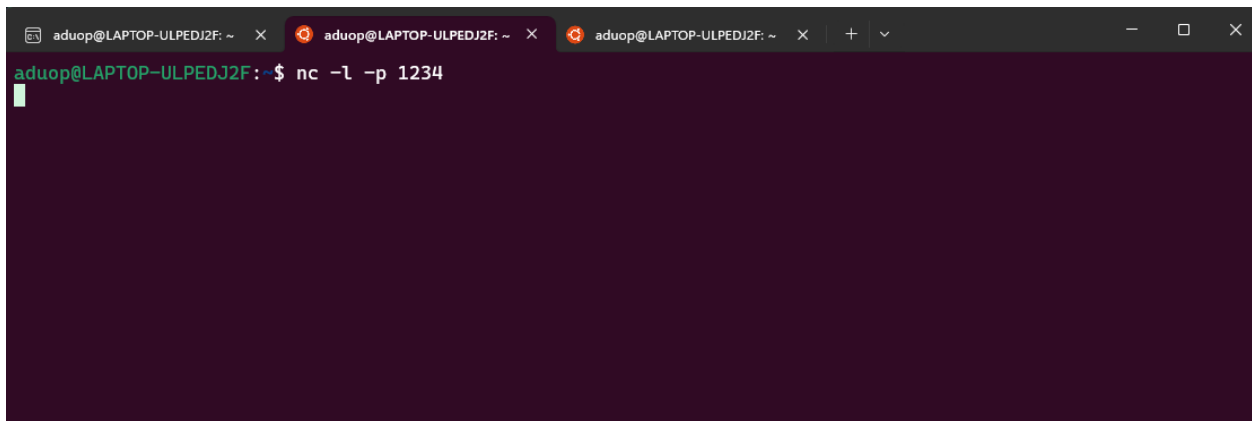
aduop@LAPTOP-ULPEDJ2F:~$
```

Figure 6: reverted_changes

3 Question - 3

3.1 Part (A)

In this part of the question , we will Set Up a TCP Client/Server Connection with localhost using **netcat** command . For this purpose lets first open two ubuntu terminals on the host machine . Now , on the first terminal we write the command **nc -l -p 1234** . This command tells netcat to listen for incoming connections on port 1234 . The screenshot for the same is follows :

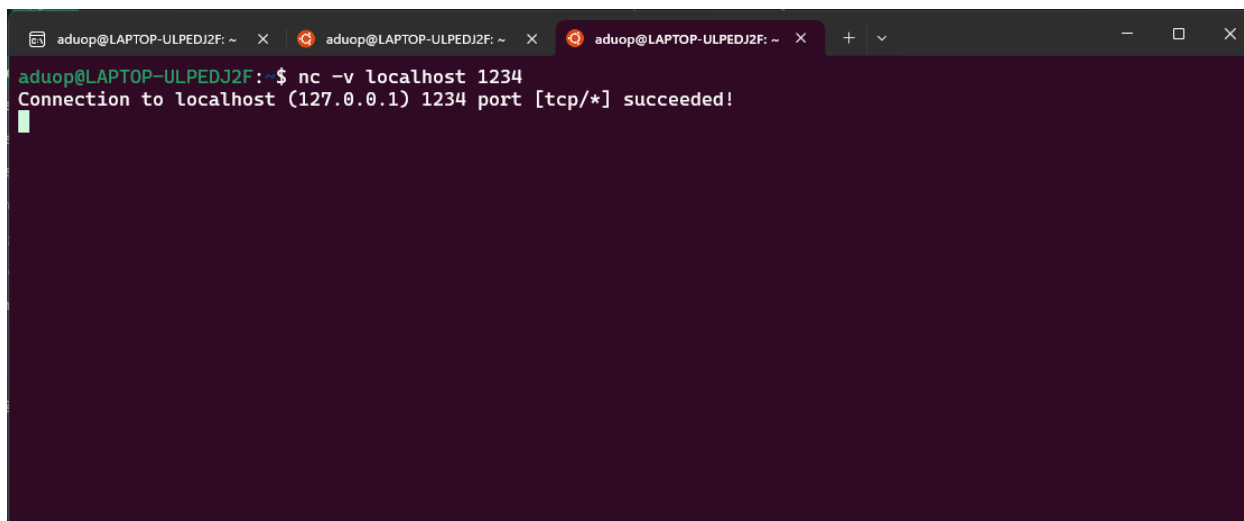


```
aduop@LAPTOP-ULPEDJ2F: ~  x  aduop@LAPTOP-ULPEDJ2F: ~  x  aduop@LAPTOP-ULPEDJ2F: ~  x  +  v  -  □  x
aduop@LAPTOP-ULPEDJ2F:~$ nc -l -p 1234
```

Figure 7: terminal-1 ready for communication

Now , lets open the second terminal and run the command **nc -v localhost 1234** . This command initiates a connection to the server running on 'localhost' on port 1234 . The connection to localhost

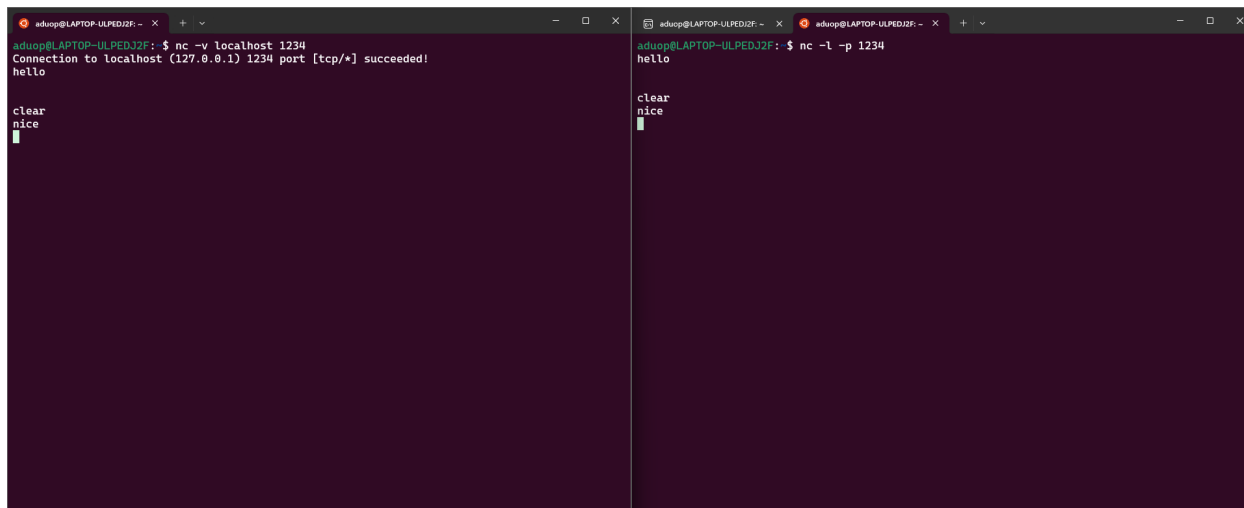
(127.0.0.1) gets succeeded as a result . The screenshot for the same is as follows :

A terminal window with a dark background and light green text. The prompt is 'aduop@LAPTOP-ULPEDJ2F: ~'. The command entered is 'nc -v localhost 1234'. The output is 'Connection to localhost (127.0.0.1) 1234 port [tcp/*] succeeded!'. The cursor is on a new line below the output.

```
aduop@LAPTOP-ULPEDJ2F: ~  
$ nc -v localhost 1234  
Connection to localhost (127.0.0.1) 1234 port [tcp/*] succeeded!  
█
```

Figure 8: Terminal-2 ready for communication

Thus the connection is established . Now if we write anything on one terminal then it will be reflected in the other terminal also . A screenshot for the same is as follows :

Two terminal windows side-by-side. The left window shows the command 'nc -v localhost 1234' and output 'Connection to localhost (127.0.0.1) 1234 port [tcp/*] succeeded!'. It then shows 'hello' being typed and received. The right window shows the command 'nc -l -p 1234' and output 'hello' being received. Both windows show 'clear' and 'nice' commands being entered and executed.

```
aduop@LAPTOP-ULPEDJ2F: ~  
$ nc -v localhost 1234  
Connection to localhost (127.0.0.1) 1234 port [tcp/*] succeeded!  
hello  
  
clear  
nice  
█  
  
aduop@LAPTOP-ULPEDJ2F: ~  
$ nc -l -p 1234  
hello  
  
clear  
nice  
█
```

Figure 9: Connection checking between the two terminals

3.2 Part (B)

Now, in this part of the quesiton , we need to check the state of this TCP connection(s) at the client node . For checking this we can open another terminal while the other two terminal(which have established) are communicating . In this new terminal , we write the command **netstat -an | grep 1234** . Now this command will return the state of the TCP connection . The result for the same is as follows :

```

aduo@LAPTOP-ULPEDJ2F:~$ netstat -an | grep 1234
tcp        0      0 0.0.0.0:1234        0.0.0.0:*           LISTEN
tcp        0      0 127.0.0.1:1234      127.0.0.1:58684      ESTABLISHED
tcp        0      0 127.0.0.1:58684     127.0.0.1:1234      ESTABLISHED
aduo@LAPTOP-ULPEDJ2F:~$

```

Figure 10: State of the connection at client node

We can now see that the connection state is listen and established here . We see these states because of the following reasons :

- **LISTEN State (Server)** : When we run `nc -l -p 1234` on the server, netcat starts listening on port 1234. When the client (`nc localhost 1234`) connects to the server, the connection is established, and you will see **ESTABLISHED** state entries in both the client and server. for incoming connections. Thus it is on the server side .
- **ESTABLISHED State (Client and Server)**: When the client (`nc -v localhost 1234`) connects to the server, the connection is established, and you will see **ESTABLISHED** state entries in both the client and server. Thus this is the reason for 2 **ESTABLISHED** being shown when we run the command `netstat -an | grep 1234` . Thus on the **client side** , the TCP connection state is **ESTABLISHED** .

This is how we checked the connection state .

4 Question - 4

4.1 Part (A)

In this part of the question , we need to get an authoritative result for "**google.in**" using **nslookup** command . Now to do this we do the following steps (along with the screenshots for each step as we progress) :

- First , we write the command `nslookup -type=ns google.in` . This command queries the DNS for the Name Servers (NS) responsible for "google.in" . We also get the authoritative name servers from the response . The screenshot for the same is as follows :

```

aduo@LAPTOP-ULPEDJ2F:~$ nslookup -type=ns google.in
Server:      10.255.255.254
Address:     10.255.255.254#53

Non-authoritative answer:
google.in    nameserver = ns1.google.com.
google.in    nameserver = ns3.google.com.
google.in    nameserver = ns4.google.com.
google.in    nameserver = ns2.google.com.

Authoritative answers can be found from:
ns1.google.com internet address = 216.239.32.10
ns3.google.com internet address = 216.239.36.10
ns4.google.com internet address = 216.239.38.10
ns2.google.com internet address = 216.239.34.10
aduo@LAPTOP-ULPEDJ2F:~$

```

Figure 11: Getting authoritative name servers

Now , we have the information that from where we can get the authorized results .

- Let's select one of the authoritative name servers from the list showcased before . We select **ns1.google.com** . Now we run the command **nslookup google.in ns1.google.com** , we get the following result :

```
aduop@LAPTOP-ULPEDJ2F:~$ nslookup google.in ns1.google.com
Server:      ns1.google.com
Address:     216.239.32.10#53

Name:   google.in
Address: 142.250.194.228
Name:   google.in
Address: 2404:6800:4002:825::2004

aduop@LAPTOP-ULPEDJ2F:~$ |
```

Figure 12: authoritative result for “google.in”

This step will query one of the authoritative name servers directly to get information about the ”google.in” domain .

4.2 Part (B)

In this part of the question , we need to find out the time to live for any website on the local DNS . Thus we need to find TTL(Time To Live) for any website . We can use the command **nslookup -debug www.google.com** . We use -debug flag to get the information about the ttl time for the domain . We get the following result from the terminal :

```
aduop@LAPTOP-ULPEDJ2F:~$ nslookup -debug www.google.com
Server:      10.255.255.254
Address:     10.255.255.254#53

-----
QUESTIONS:
  www.google.com, type = A, class = IN
ANSWERS:
-> www.google.com
   internet address = 142.250.77.228
   ttl = 11
AUTHORITY RECORDS:
ADDITIONAL RECORDS:
-----
Non-authoritative answer:
Name:   www.google.com
Address: 142.250.77.228

-----
QUESTIONS:
  www.google.com, type = AAAA, class = IN
ANSWERS:
-> www.google.com
   has AAAA address 2404:6800:4002:814::2004
   ttl = 12
AUTHORITY RECORDS:
ADDITIONAL RECORDS:
-----
Name:   www.google.com
Address: 2404:6800:4002:814::2004
```

Figure 13: ttl values

Thus we can see the ttl values in the screenshots , to make it look more clear , we have combine it with grep command using pipe to showcase the ttl values . Thus , we run the command **nslookup -debug www.google.com | grep ttl**

```
aduop@LAPTOP-ULPEDJ2F:~$ nslookup -debug www.google.com | grep ttl
    ttl = 118
    ttl = 40
aduop@LAPTOP-ULPEDJ2F:~$ |
```

Figure 14: ttl values finding

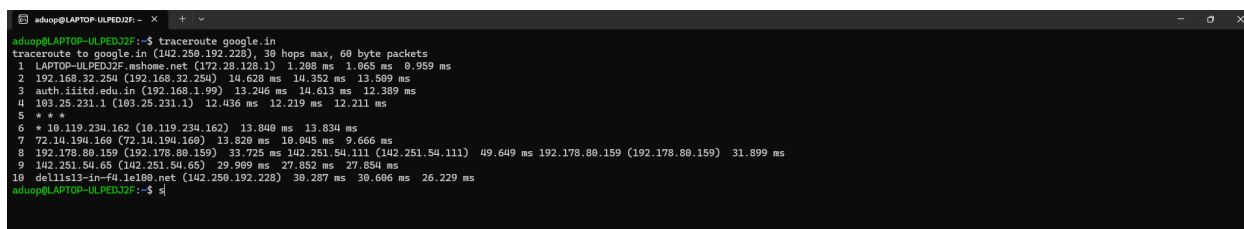
Now we can see **two ttl values** , this is because of the fact that one ttl value is for **type = A** , which is for **IPv4** address and the other is for **type = AAAA** which is for **IPv6** address. Thus this is the reason for two ttl values . Thus the ttl values shown in figure.14 is **118** seconds(A) and **40** seconds(AAAA) respectively . Thus the entry for the IPv4 entry will **expire** in **118** seconds while the IPv6 address will expire in **40** seconds .

Reference - Check DNS TTL Record Guide

5 Question - 5

5.1 Part (A)

In this part of the question , first we need to run the command **traceroute google.in** , then find the number of intermediate hosts . The screenshot after running the command is as follows :



```

aduop@LAPTOP-ULPEDJ2F:~$ traceroute google.in
traceroute to google.in (142.250.192.228), 30 hops max, 60 byte packets
 1 LAPTOP-ULPEDJ2F.mshome.net (172.28.128.1)  1.208 ms  1.065 ms  0.959 ms
 2 192.168.32.254 (192.168.32.254)  14.628 ms  14.352 ms  13.509 ms
 3 auth.iitd.edu.in (192.168.1.99)  13.246 ms  14.613 ms  12.389 ms
 4 103.25.231.1 (103.25.231.1)  12.436 ms  12.219 ms  12.211 ms
 5 * * *
 6 * 10.119.234.162 (10.119.234.162)  13.840 ms  13.834 ms
 7 72.14.194.160 (72.14.194.160)  13.820 ms  10.045 ms  9.666 ms
 8 192.178.80.159 (192.178.80.159)  33.725 ms  142.251.54.111 (142.251.54.111)  49.649 ms  192.178.80.159 (192.178.80.159)  31.899 ms
 9 142.251.54.65 (142.251.54.65)  29.989 ms  27.852 ms  27.854 ms
10 del11a13-in-f4.1e100.net (142.250.192.228)  30.287 ms  30.606 ms  26.229 ms
aduop@LAPTOP-ULPEDJ2F:~$

```

Figure 15: traceroute google.in

Lets list the intermediate hosts from the output now , they are as follows :

1. Hop 1 - The IP address of the Host 1 is 172.28.128.1 . This is the first intermediate host with average latency :

$$(1.208ms + 1.065ms + 0.959ms)/3 = 1.077ms \quad (1)$$

2. Hop 2 - The IP address of the Host 2 is 192.168.32.254 . This is the second intermediate host with average latency of :

$$(14.628ms + 14.352ms + 13.509ms)/3 = 14.163ms \quad (2)$$

3. Hop 3 - The IP address of the Host 3 is 192.168.1.99 . This is the third intermediate host with average latency of :

$$(13.246ms + 14.613ms + 12.389ms)/3 = 13.416ms \quad (3)$$

4. Hop 4 - The IP address of the Host 4 is 103.25.231.1 . This is the fourth intermediate host with average latency of :

$$(12.436ms + 12.219ms + 12.211ms)/3 = 12.289ms \quad (4)$$

5. Hop 5 - The IP address of the Host 5 is unknown as it is * * * * . Thus the average latency is N/A (not applicable) .

6. Hop 6 - The IP address of the Host 6 is 10.119.234.162 .This is the sixth intermediate host with average latency of :

$$(13.840ms + 13.834ms)/2 = 13.837ms \quad (5)$$

7. Hop 7 - The IP address of the Host 7 is 72.14.194.160 . This is the seventh intermediate host with average latency of :

$$(13.820ms + 10.045ms + 9.666ms)/3 = 11.844ms \quad (6)$$

8. Hop 8 - In this case , there are 3 intermediate hosts , this occurs to reduce the load on a single host/router , thus the average latency for each of them are as follows :

- Host IP - 192.178.80.159 and average latency - 33.725 ms .
- Host IP - 142.251.54.111 and average latency - 49.649 ms .
- Host IP - 192.178.80.159 and average latency - 31.899 ms .

Thus , we can calculate the average for these three sub-hosts and find the net average latency for this hop as :

$$(33.725ms + 49.649ms + 31.899ms)/3 = 38.424ms \quad (7)$$

We call this as the eighth intermediate host .

- Hop 9 - The IP address of the Host 9 is 142.251.54.65 . This is the ninth intermediate host with average latency of :

$$(29.909ms + 27.852ms + 27.854ms)/3 = 28.538ms \quad (8)$$

- Hop 10 - The IP address of the Host 10 is 142.250.192.228 . This is the ninth intermediate host with average latency of :

$$(30.287ms + 30.606ms + 26.229ms)/3 = 29.04ms \quad (9)$$

This is the final location host (Host 10) .

These were all the intermediate hosts along with there average latencies . The last line is the final destination host . Thus we have **10** intermediate hosts including the last final host.

5.2 Part (B)

For this part let us first run the command **ping -c 50 google.com** . The screenshot for the same is as follows :

```
aduope@LAPTOP-ULPEDJ2F:~$ ping -c 50 google.com
PING google.com (142.250.256.174) 56(84) bytes of data:
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=1 ttl=113 time=26.9 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=2 ttl=113 time=26.3 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=3 ttl=113 time=26.1 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=4 ttl=113 time=26.3 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=5 ttl=113 time=26.0 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=6 ttl=113 time=26.2 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=7 ttl=113 time=26.5 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=8 ttl=113 time=26.1 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=9 ttl=113 time=26.7 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=10 ttl=113 time=26.1 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=11 ttl=113 time=26.2 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=12 ttl=113 time=26.2 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=13 ttl=113 time=26.8 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=14 ttl=113 time=77.5 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=15 ttl=113 time=27.0 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=16 ttl=113 time=45.0 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=17 ttl=113 time=26.1 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=18 ttl=113 time=65.8 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=19 ttl=113 time=26.3 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=20 ttl=113 time=26.6 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=21 ttl=113 time=26.2 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=22 ttl=113 time=26.1 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=23 ttl=113 time=26.8 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=24 ttl=113 time=33.4 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=25 ttl=113 time=26.5 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=26 ttl=113 time=26.1 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=27 ttl=113 time=28.3 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=28 ttl=113 time=26.5 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=29 ttl=113 time=26.7 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=30 ttl=113 time=27.1 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=31 ttl=113 time=26.2 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=32 ttl=113 time=26.2 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=33 ttl=113 time=27.9 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=34 ttl=113 time=26.4 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=35 ttl=113 time=26.0 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=36 ttl=113 time=26.3 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=37 ttl=113 time=26.2 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=38 ttl=113 time=26.2 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=39 ttl=113 time=26.2 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=40 ttl=113 time=27.4 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=41 ttl=113 time=31.3 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=42 ttl=113 time=39.3 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=43 ttl=113 time=26.2 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=44 ttl=113 time=26.4 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=45 ttl=113 time=26.2 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=46 ttl=113 time=26.2 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=47 ttl=113 time=26.7 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=48 ttl=113 time=26.3 ms
64 bytes from dell1522-in-f14.1e100.net (142.250.256.174): icmp_seq=49 ttl=113 time=26.1 ms
```

```

64 bytes from dell1s22-in-f14.1e100.net (142.250.206.174): icmp_seq=44 ttl=113 time=26.4 ms
64 bytes from dell1s22-in-f14.1e100.net (142.250.206.174): icmp_seq=45 ttl=113 time=26.2 ms
64 bytes from dell1s22-in-f14.1e100.net (142.250.206.174): icmp_seq=46 ttl=113 time=26.2 ms
64 bytes from dell1s22-in-f14.1e100.net (142.250.206.174): icmp_seq=47 ttl=113 time=26.7 ms
64 bytes from dell1s22-in-f14.1e100.net (142.250.206.174): icmp_seq=48 ttl=113 time=26.3 ms
64 bytes from dell1s22-in-f14.1e100.net (142.250.206.174): icmp_seq=49 ttl=113 time=26.1 ms
64 bytes from dell1s22-in-f14.1e100.net (142.250.206.174): icmp_seq=50 ttl=113 time=26.5 ms

--- google.com ping statistics ---
50 packets transmitted, 50 received, 0% packet loss, time 49882ms
rtt min/avg/max/mdev = 26.010/28.956/77.530/9.257 ms
aduo@LAPTOP-ULPEDJ2F:~$

```

Figure 16: Running ping command

We can see that the average rtt comes out to be **28.956 ms**. Here rtt is round trip time. Thus we consider latency = rtt (assumption). Thus average latency comes out to be **28.956 ms**. This is asked in the question.

5.3 Part (C)

In this part of the problem, first we add up the ping latency of all the intermediate hosts obtained in (a). This comes out to be :

$$(1.077ms + 14.163ms + 13.416ms + 12.289ms + 13.837ms + 11.844ms + 38.424ms + 28.538ms + 29.04ms) = 162.628ms \quad (10)$$

Thus after adding up all the avg latencies from (a) we get the value as **162.628 ms**. This is a lot more than the value we get from part (b) i.e 28.956 ms. The reasons for the same are as follows :

- Traceroute measures the time taken for each hop in the network path from your computer to the destination server. It does this by sending packets with increasing TTL (Time to Live) values, and each router along the path sends a "Time Exceeded" message when it decreases the TTL to 0. This process measures the latency to each intermediate hop individually and accumulates delays from all the hops.
- Ping sends ICMP Echo Request packets directly to the destination server and measures the round-trip time (RTT) for these packets. The result gives you the time it takes for a packet to go from your computer to the destination server and back, effectively summarizing the latency of the entire path in a single value.
- This is the reason why the sum of values is much larger than what we got in part b.

Reference - Check difference in ping and traceroute

5.4 Part (D)

The maximum latency from the part (a) is **49.649 ms**. This is not matching with the value we get in part (b) which is **28.956 ms**. The maximum latency is calculated because in case of bottleneck the latency is maximum as it takes the maximum time to be pinged.

The average latency is dependent on several factors like internet bandwidth and wifi signals hence these values differ when we run them after some time. Also some of reasons listed in part (c) play an important role in this.

5.5 Part (E)

The multiple entries for a single hop occur while using traceroute command because of the following reasons :

- Load balancing - In some networks , routers may employ load balancing , where different probes take slightly different paths to the destination . This cause 'traceroute' to show different results for the same hop as the packets may traverse different routers . Thus the traffic is distributed across different paths or routers .
 - Network Behavior - If you see significantly different times for the same hop, it might indicate network congestion, varying routes, or inconsistent performance on that hop.
 - Latency Information - The multiple entries show the round-trip time (RTT) for each of the probes to reach that hop and return. This helps in identifying variability or potential issues in the network.
- Thus these are some of the reasons why this phenomenon occurs in traceroute command .

Reference - Check for the reason

5.6 Part (F)

In this part , lets first run the command **ping stanford.edu** . The screenshot of the result is as follows :

```

adun@LAPTOP-ULPEDJ2F:~$ ping -c 50 stanford.edu
PING stanford.edu (171.67.215.200) 56(84) bytes of data:
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=1 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=2 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=3 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=4 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=5 ttl=234 time=288 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=6 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=7 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=8 ttl=234 time=290 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=9 ttl=234 time=293 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=10 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=11 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=12 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=13 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=14 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=15 ttl=234 time=285 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=16 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=17 ttl=234 time=289 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=18 ttl=234 time=291 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=19 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=20 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=21 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=22 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=23 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=24 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=25 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=26 ttl=234 time=289 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=27 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=28 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=29 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=30 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=31 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=32 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=33 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=34 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=35 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=36 ttl=234 time=288 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=37 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=38 ttl=234 time=285 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=39 ttl=234 time=297 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=40 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=41 ttl=234 time=289 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=42 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=43 ttl=234 time=286 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=44 ttl=234 time=290 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=45 ttl=234 time=292 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=46 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=47 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=48 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=49 ttl=234 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=50 ttl=234 time=286 ms

--- stanford.edu ping statistics ---
50 packets transmitted, 50 received, 0% packet loss, time 49069ms
rtt min/avg/max/ndev = 285.375/287.211/296.809/2.099 ms
adun@LAPTOP-ULPEDJ2F:~$

```

Figure 17: ping stanford.edu

Now ,we can see that the average rtt is 287.211 ms . Thus using the formulae that avg latency = avg rtt(assumption) , we get that the average latency is **287.211 ms** .

5.7 Part (G)

Lets first run the command **traceroute stanford.edu** . Thus the result of the same is as follows :

```
aduop@LAPTOP-ULPEDJ2F:~$ traceroute stanford.edu
traceroute to stanford.edu (171.67.215.200), 30 hops max, 60 byte packets
 1 LAPTOP-ULPEDJ2F.mshome.net (172.28.128.1)  0.445 ms  0.415 ms  0.694 ms
 2 192.168.32.254 (192.168.32.254)  121.951 ms  121.680 ms  121.632 ms
 3 vpn.iitd.edu.in (192.169.1.99)  12.720 ms  12.689 ms  12.678 ms
 4 103.25.231.1 (103.25.231.1)  12.292 ms  12.284 ms  12.258 ms
 5 10.1.200.201 (10.1.200.201)  35.366 ms  31.674 ms  29.444 ms
 6 10.1.200.137 (10.1.200.137)  32.599 ms  32.599 ms  32.572 ms
 7 10.255.238.254 (10.255.238.254)  53.929 ms  47.942 ms  48.950 ms
 8 180.149.48.18 (180.149.48.18)  29.130 ms  31.148 ms  29.797 ms
 9 * * *
10 * * *
11 * * *
12 * * *
13 * * *
14 * * *
15 * * *
16 * * *
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * * *
23 * * *
24 * campus-ial-nets-b-vl1120.SUNet (171.66.255.232)  289.499 ms *
25 * * campus-mw-rtt-vl1104.SUNet (171.66.255.200)  290.258 ms
26 web.stanford.edu (171.67.215.200)  284.759 ms * *
aduop@LAPTOP-ULPEDJ2F:~$
```

Figure 18: traceroute stanford.edu

Thus we can see the number of hops in stanford.edu is 26 including the **** ones and the number of hops in google.in is 10 including the **** ones from . Thus the number of hops in stanford.edu one are more as a lot more distance has to be covered to reach to that domain as compared to google.in .

5.8 Part (H)

In part b , we find the latency for google.in to be **28.956 ms**. While from part f , we found the latency for stanford.edu to be **287.211 ms** . Thus the latency for stanford is much higher than google.in .

The reason for this is the packets have to travel more distance and cover more number of hosts hence this leads to increased latency and delay . The propagation and the processing time increases in case of the stanford.edu case .

These are the primary reasons why stanford.edu take more latency time .

6 Question - 6

In this question , we need to make the ping command fail for 127.0.0.1(with 100 % packet loss) . For this we will follow the following steps :

- The 'lo' (loopback) interface is a virtual network on the system that allows the machine to communicate with itself . The IP address '127.0.0.1' is bound to this interface . It is used for testing and communication within the local machine .
- Thus , we will run the command **sudo ifconfig lo down** . This brings the loopback interface down , effectively disabling it . This means that the system can no longer route traffic to or from 127.0.0.1. The screenshot of running the command is as follows :

```
aduop@LAPTOP-ULPEDJ2F: ~$ sudo ifconfig lo down
aduop@LAPTOP-ULPEDJ2F:~$
```

Figure 19: disabling the lo interface

- Now that the lo interface is disabled , we try running the command **ping 127.0.0.1** . The results are as follows :

```
aduop@LAPTOP-ULPEDJ2F:~$ sudo ifconfig lo down
aduop@LAPTOP-ULPEDJ2F:~$
aduop@LAPTOP-ULPEDJ2F:~$ 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 ---
4 packets transmitted, 0 received, 100% packet loss, time 3137ms

aduop@LAPTOP-ULPEDJ2F:~$
```

Figure 20: 100% packet loss on running ping 127.0.0.1 command

We see that that 100% packet loss occurs now as the 'lo' interface is down .

- Now for reverting back to the original setting , we run the command **sudo ifconfig lo up** . Now we start getting 0% packet loss again . The screenshot for the same is as follows :

```
aduop@LAPTOP-ULPEDJ2F:~$ sudo ifconfig lo up
aduop@LAPTOP-ULPEDJ2F:~$
aduop@LAPTOP-ULPEDJ2F:~$
aduop@LAPTOP-ULPEDJ2F:~$ ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.018 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.032 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.028 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.020 ms
64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.038 ms
^C
--- 127.0.0.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4143ms
rtt min/avg/max/mdev = 0.018/0.027/0.038/0.007 ms
aduop@LAPTOP-ULPEDJ2F:~$
```

Figure 21: Reverting back to 0% packet loss