

Assignment 1 - COL334

Aayush Goyal

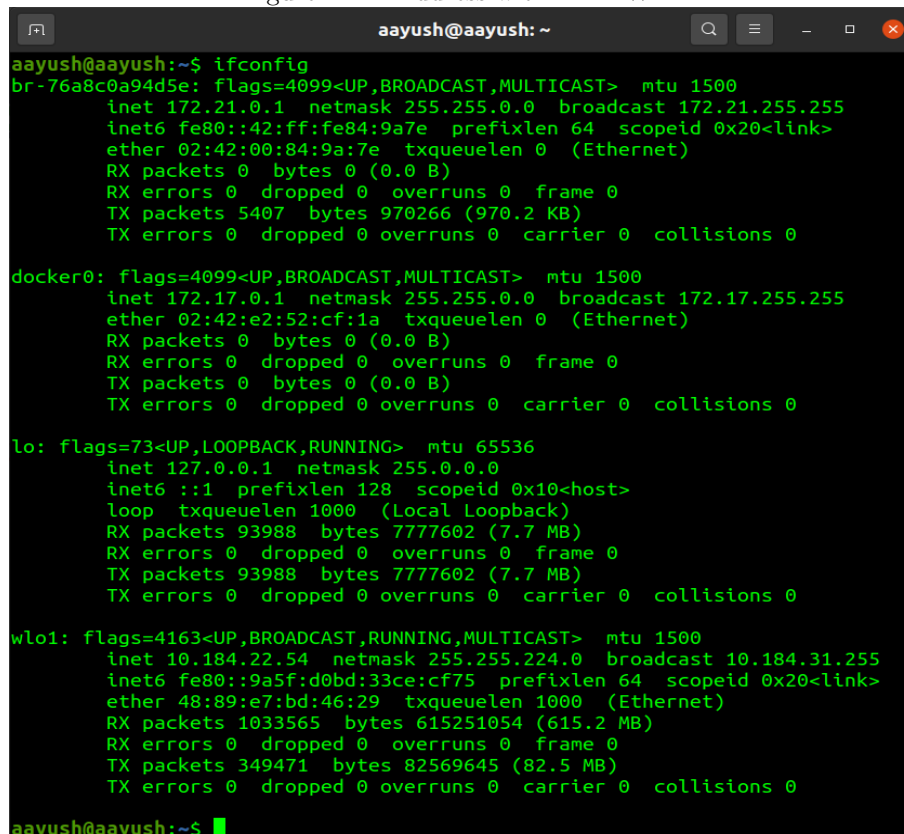
August 2021

1 Networking Tools

1.1 ifconfig

For this part I have considered the private IP address of my machine provided by the router. The router provides each machine a IP address and this changes when the Internet service provider is changed. When the command "ifconfig" is run in the terminal, once with Jio Mobile Hotspot and second time with IITD WiFi the IPv4 and IPv6 both were different as we can see in the below images (IPv4 - inet value in the first line and IPv6 - inet6 value in the third line)

Figure 1: IP Address with IITD WiFi

A terminal window titled 'aayush@aayush: ~' showing the output of the 'ifconfig' command. The output lists network interfaces: 'br-76a8c0a94d5e' (bridge), 'docker0' (Docker interface), 'lo' (loopback), and 'wlo1' (wireless). Each interface shows its flags, MTU, and IP addresses (inet for IPv4, inet6 for IPv6). The 'wlo1' interface shows an IPv4 address of 10.184.22.54 and an IPv6 address of fe80::9a5f:d0bd:33ce:cf75. The terminal text is as follows:

```
aayush@aayush:~$ ifconfig
br-76a8c0a94d5e: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.21.0.1 netmask 255.255.0.0 broadcast 172.21.255.255
    inet6 fe80::42:ff:fe84:9a7e prefixlen 64 scopeid 0x20<link>
    ether 02:42:00:84:9a:7e txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 5407 bytes 970266 (970.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    ether 02:42:e2:52:cf:1a txqueuelen 0 (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 93988 bytes 7777602 (7.7 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 93988 bytes 7777602 (7.7 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.184.22.54 netmask 255.255.224.0 broadcast 10.184.31.255
    inet6 fe80::9a5f:d0bd:33ce:cf75 prefixlen 64 scopeid 0x20<link>
    ether 48:89:e7:bd:46:29 txqueuelen 1000 (Ethernet)
    RX packets 1033565 bytes 615251054 (615.2 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 349471 bytes 82569645 (82.5 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

aayush@aayush:~$
```

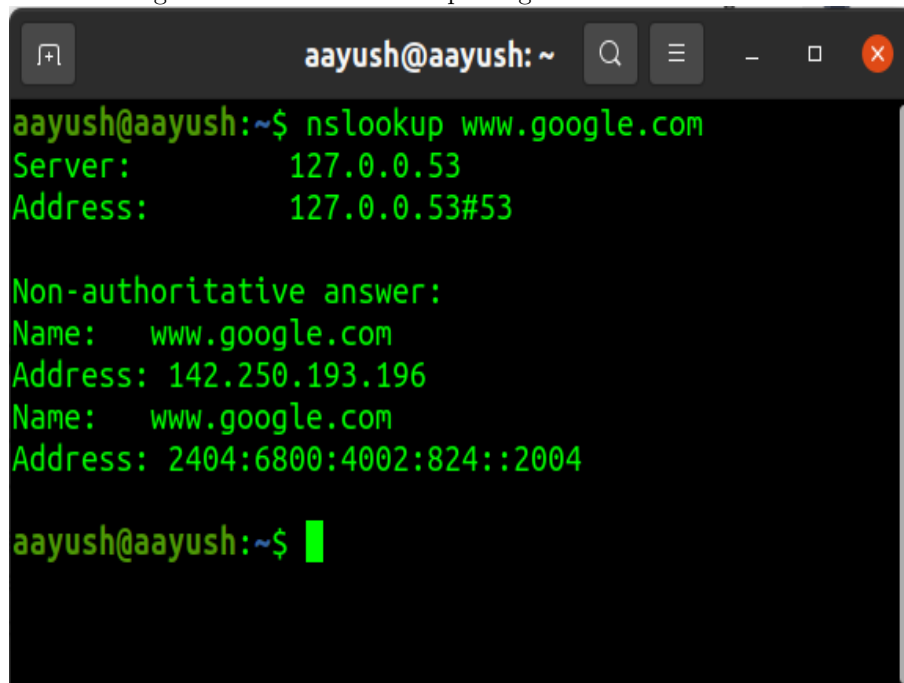
Figure 2: IP Address with Jio Mobile Hotspot

```
aayush@aayush: ~  
aayush@aayush:~$ ifconfig  
br-76a8c0a94d5e: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500  
    inet 172.21.0.1 netmask 255.255.0.0 broadcast 172.21.255.255  
    inet6 fe80::42:ff:fe84:9a7e prefixlen 64 scopeid 0x20<link>  
    ether 02:42:00:84:9a:7e txqueuelen 0 (Ethernet)  
    RX packets 0 bytes 0 (0.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 5411 bytes 970786 (970.7 KB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500  
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255  
    ether 02:42:e2:52:cf:1a txqueuelen 0 (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 94336 bytes 7804966 (7.8 MB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 94336 bytes 7804966 (7.8 MB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.43.228 netmask 255.255.255.0 broadcast 192.168.43.255  
    inet6 fe80::d8d7:455c:85f9:d3b2 prefixlen 64 scopeid 0x20<link>  
    inet6 2405:204:1024:7090:8c70:5c90:d6bb:8a3e prefixlen 64 scopeid 0x0<global>  
    inet6 2405:204:1024:7090:e908:45fd:13ea:50ca prefixlen 64 scopeid 0x0<global>  
    ether 48:89:e7:bd:46:29 txqueuelen 1000 (Ethernet)  
    RX packets 1034147 bytes 615447270 (615.4 MB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 350019 bytes 82768143 (82.7 MB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
aayush@aayush:~$
```

1.2 nslookup

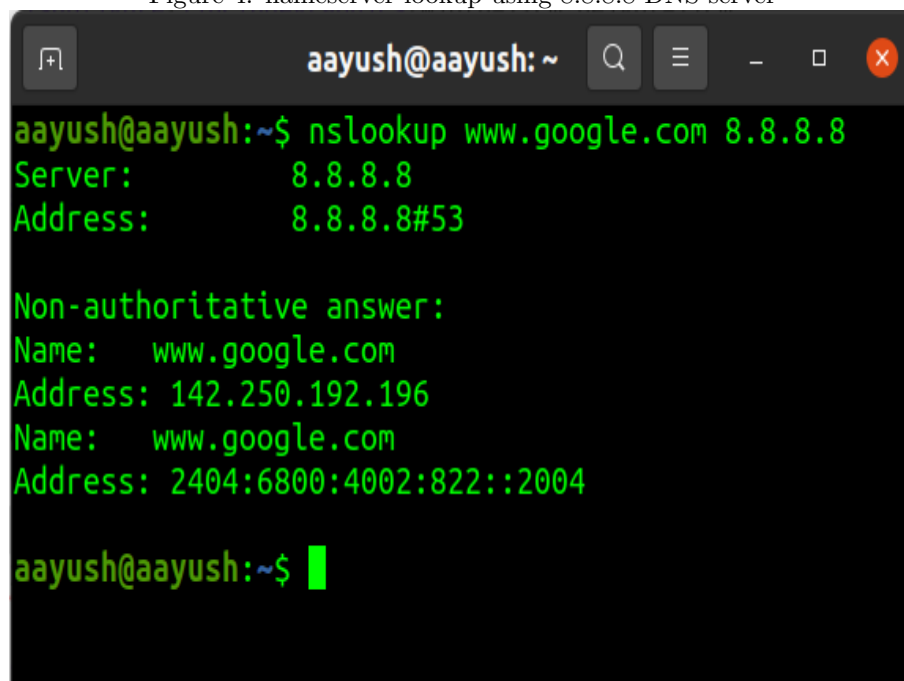
The "nslookup" command is used to find the IPv4 and IPv6 address associated with a domain name. This keeps changing even for the same DNS server. For human beings it is not easy to remember the IP address of a server, so they are given a domain name, but for computer we need the IP address and hence the DNS servers help in this conversion from domain name to its IP. Domain names like www.google.com have many IP addresses reserved with it, thus depending on the DNS server to which we are asking, the resultant IPv4 and IPv6 values reported maybe different.

Figure 3: nameserver lookup using 127.0.0.53 DNS server



```
aayush@aayush: ~  
aayush@aayush:~$ nslookup www.google.com  
Server:          127.0.0.53  
Address:         127.0.0.53#53  
  
Non-authoritative answer:  
Name:   www.google.com  
Address: 142.250.193.196  
Name:   www.google.com  
Address: 2404:6800:4002:824::2004  
  
aayush@aayush:~$
```

Figure 4: nameserver lookup using 8.8.8.8 DNS server



```
aayush@aayush: ~  
aayush@aayush:~$ nslookup www.google.com 8.8.8.8  
Server:          8.8.8.8  
Address:         8.8.8.8#53  
  
Non-authoritative answer:  
Name:   www.google.com  
Address: 142.250.192.196  
Name:   www.google.com  
Address: 2404:6800:4002:822::2004  
  
aayush@aayush:~$
```

1.3 ping

The ping command is used to check whether a particular host is reachable across an IP network, by sending an ICMP packet and checking the response. This can be used to communicate with various IPs.

To find the maximum packet size for which the domain responds, binary search on the final possible value can be done. For this we send the domain some packets of a size and based on the response we can determine whether it is the optimal value or the actual value is smaller or bigger.

For this I made a python script which will ping the domain and by doing a binary search found the maximum size for which we can get back a response.

The maximum data packet size that we can send turned out to be 65399. The maximum data packet size that we can receive a response for is same for both www.facebook.com and www.google.com and the value turned out to be 1472. Whereas for www.iitd.ac.in, since I am connected on the IITD WiFi, that is the Local Network, I was able to receive a response for the maximum packet size that we can send ie 65399.

Figure 5: Binary search for www.google.com



```
aayush@aayush: ~/Desktop/COL334-Computer-Networks/Assignment-1
aayush@aayush:~/Desktop/COL334-Computer-Networks/Assignment-1$ sudo python 1c.py www.google.com
1 4294967296
1 2147483647
1 1073741823
1 536870911
1 268435455
1 134217727
1 67108863
1 33554431
1 16777215
1 8388607
1 4194303
1 2097151
1 1048575
1 524287
1 262143
1 131071
1 65535
1 32767
1 16383
1 8191
1 4095
1 2047
1025 2047
1025 1535
1281 1535
1409 1535
1473 1535
1473 1503
1473 1487
1473 1479
1473 1475
1473 1473
Maximum packet size: 1472
aayush@aayush:~/Desktop/COL334-Computer-Networks/Assignment-1$
```

Figure 6: Binary search for www.facebook.com

```

aayush@aayush: ~/Desktop/COL334-Computer-Networks/Assignment-1
aayush@aayush:~/Desktop/COL334-Computer-Networks/Assignment-1$ sudo python 1c.py www.facebook.com
1 4294967296
1 2147483647
1 1073741823
1 536870911
1 268435455
1 134217727
1 67108863
1 33554431
1 16777215
1 8388607
1 4194303
1 2097151
1 1048575
1 524287
1 262143
1 131071
1 65535
1 32767
1 16383
1 8191
1 4095
1 2047
1025 2047
1025 1535
1281 1535
1409 1535
1473 1535
1473 1503
1473 1487
1473 1479
1473 1475
1473 1473
Maximum packet size: 1472
aayush@aayush:~/Desktop/COL334-Computer-Networks/Assignment-1$

```

Figure 7: Binary search for www.iitd.ac.in

```

aayush@aayush: ~/Desktop/COL334-Computer-Networks/Assignment-1
aayush@aayush:~$ cd Desktop/COL334-Computer-Networks/Assignment-1/
aayush@aayush:~/Desktop/COL334-Computer-Networks/Assignment-1$ sudo python 1c.py www.iitd.ac.in
[sudo] password for aayush:
1 4294967296
1 2147483647
1 1073741823
1 536870911
1 268435455
1 134217727
1 67108863
1 33554431
1 16777215
1 8388607
1 4194303
1 2097151
1 1048575
1 524287
1 262143
1 131071
1 65535
32769 65535
49153 65535
57345 65535
61441 65535
63489 65535
64513 65535
65025 65535
65281 65535
65281 65407
65345 65407
65377 65407
65393 65407
65393 65399
65397 65399
65399 65399
Maximum packet size: 65399
aayush@aayush:~/Desktop/COL334-Computer-Networks/Assignment-1$

```

1.4 traceroute

When the packet is sent from the source to the destination IP address, it goes through a number of routers on the way. Traceroute command helps us to identify the the IPs of the routers we have encountered on the way. -4 flag can be used to print the IPv4 address forcibly. With this we can determine the IP of a router 'x' hops away from the source and also the Round trip time from the source to that router. The route taken to reach the final destination address is different with different ISPs, like say when we are reaching www.iitd.ac.in using "Jio" the routers are different from the routers we get when we reach www.iitd.ac.in using Local Area Network.

Figure 8: Trace route using IITD WiFi

```
traceroute to www.iitd.ac.in (10.10.211.212), 64 hops max
1  10.184.0.14      4.020ms  1.653ms  1.740ms
2  10.254.236.18    1.610ms  1.822ms  1.845ms
3  10.10.211.212    1.653ms  1.357ms  1.553ms
```

Figure 9: Trace route using Jio Mobile Hotspot

```
traceroute to www.iitd.ac.in (103.27.9.24), 64 hops max
1  192.168.43.1      2.553ms  3.486ms  2.447ms
2  * * *
3  10.71.71.18       42.439ms 49.814ms 29.793ms
4  172.26.105.5      39.943ms 39.870ms 41.410ms
5  172.26.105.19     53.789ms 24.727ms 39.859ms
6  192.168.44.44     49.700ms 30.666ms 38.858ms
7  192.168.44.43     39.769ms 42.336ms 39.796ms
8  172.26.14.75      40.134ms 39.588ms 40.161ms
9  172.26.14.75      39.805ms 39.920ms 603.074ms
10 115.249.187.169    77.142ms 39.434ms 52.686ms
11 115.255.253.18     47.267ms 39.457ms 40.298ms
12 115.249.198.97     41.916ms 40.355ms 39.458ms
13 * * *
14 * * *
15 * * *
16 * * *
17 * * *
18 * * *
19 103.27.9.24       98.386ms 39.790ms 39.917ms
```

2 Packet Analysis

2.1

The time taken for DNS request-response to complete was around 3.2ms

2.2

Number of http requests were around 27. This shows that the whole web page is not loaded at once. Parts of the webpage load and we can't tell what the web page is going to be. Also the requests were going to different servers and thus we can see that all the data is not stored on the same server, and thus we have to send http requests to the servers where data is present. For images, gifs we have to send additional requests.

2.3

Time to download the web page is:
Start time (first DNS request): 23:00:02.822
End time (Time at which last content object was received): 23:00:04.257
Total time: 1.435s

2.4

There was no http traffic. There was only one http request. When this request is made to the server, the server responds with the code "website moved permanently" and the objects are moved to the https server as cse.iitd.ac.in deals only with encrypted data. The client then converses with the location using TLS encryption.

3 Implementing Traceroute using Ping

3.1 Approach used

The time to live value of the packet that is sent reduces by 1 everytime it hits a router on the route. Thus by making the TTL value of the packet as 'x' we can determine the the IP of the router that is 'x' hops away from the source (here 'x' is less than the minimum ttl value required to successfully reach the final destination).

I used a ICMP socket and created ICMP packets which will be the send to the final destination. Everytime the packet dies we get the response back and the IP address of the router where it died. Using this we have obtained the IP address of the Router on the route. And within the code, by starting the clock just before sending the packet and stopping the clock just after we have received the packet, we can know the Round trip time value. This way RTT value can be determined for every Hop number.

Sometimes the router doesn't respond back and in that case, code might be stuck at the line where we are waiting for the response. For that we can set the timeout value for the socket. So if we are not able to get the response in some "TIMEOUT" time, then raise the exception for that packet and report the RTT value for that packet to be 0.

3.2 Hop number and their RTT values

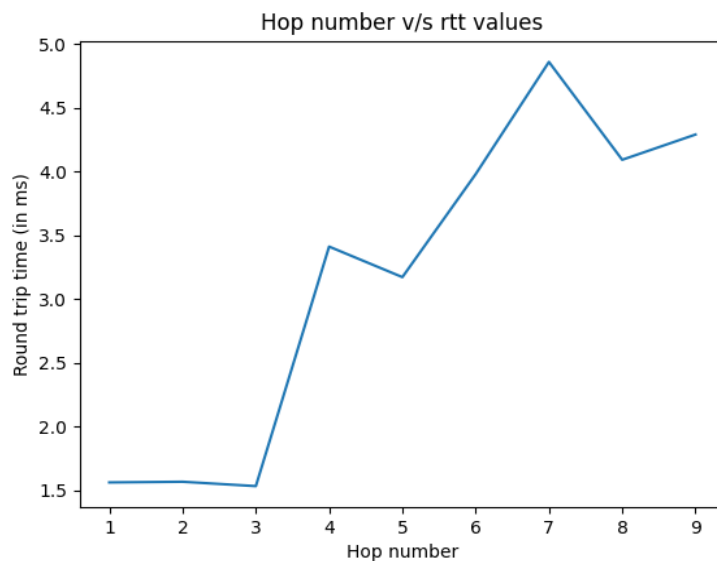
3.2.1 www.google.com

Figure 10: Routers on route to www.google.com

```
aayush@aayush: ~/Desktop/COL334-Compu...
aayush@aayush:~/Desktop/COL334-Computer-Networks/Assignment-1$
sudo python traceroute.py www.google.com

traceroute to www.google.com (142.250.77.196), 64 hops max
 1  10.184.0.14      1.732ms  1.565ms  1.806ms
 2  10.255.1.34      1.575ms  1.808ms  1.570ms
 3  10.119.233.65    1.864ms  1.688ms  1.535ms
 4  10.1.207.69      3.856ms  3.455ms  3.413ms
 5  10.119.234.162   3.477ms  3.173ms  4.584ms
 6  72.14.195.56     8.195ms  4.475ms  3.980ms
 7  74.125.243.97    5.027ms  4.860ms  5.628ms
 8  142.250.225.249  4.132ms  4.093ms  4.110ms
 9  142.250.77.196   4.291ms  4.684ms  5.079ms
aayush@aayush:~/Desktop/COL334-Computer-Networks/Assignment-1$
```

Figure 11: Hop number v/s RTT graph for www.google.com



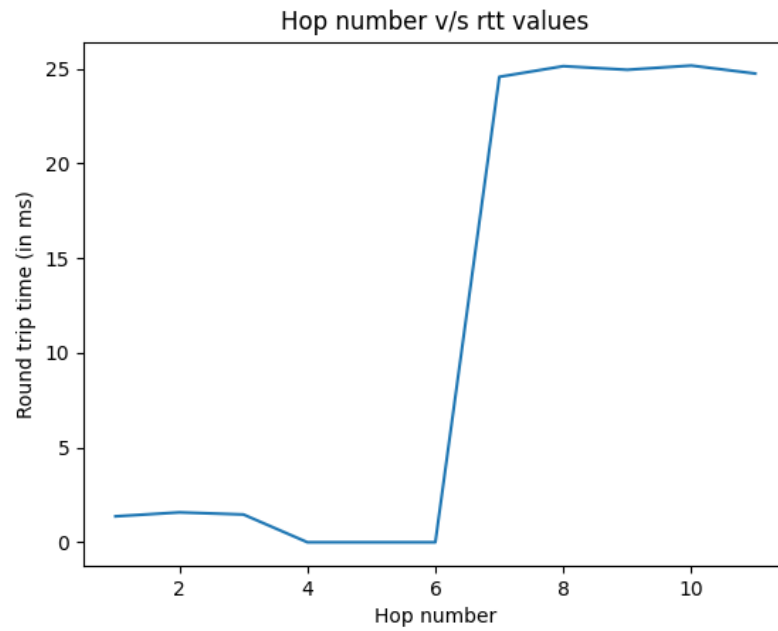
3.2.2 www.facebook.com

Figure 12: Routers on route to www.facebook.com

```
aayush@aayush: ~/Desktop/COL334-Compu...
aayush@aayush:~/Desktop/COL334-Computer-Networks/Assignment-1$
sudo python traceroute.py www.facebook.com

traceroute to www.facebook.com (157.240.16.35), 64 hops max
 1  10.184.0.14          1.443ms  1.368ms  1.379ms
 2  10.255.1.34          1.739ms  1.613ms  1.578ms
 3  10.119.233.65        1.465ms  1.619ms  1.739ms
 4  * * *
 5  * * *
 6  * * *
 7  10.152.7.38          26.485ms 25.285ms 24.568ms
 8  157.240.68.238       44.315ms 25.131ms 26.101ms
 9  157.240.40.229       25.158ms 25.229ms 24.941ms
10  157.240.38.205       27.009ms 25.163ms 25.291ms
11  157.240.16.35        24.742ms 26.251ms 25.174ms
aayush@aayush:~/Desktop/COL334-Computer-Networks/Assignment-1$
```

Figure 13: Hop number v/s RTT graph for www.facebook.com



3.2.3 www.iitd.ac.in

Figure 14: Routers on route to www.iitd.ac.in

```
aayush@aayush: ~/Desktop/COL334-Compu...  
aayush@aayush:~/Desktop/COL334-Computer-Networks/Assignment-1$  
sudo python traceroute.py www.iitd.ac.in  
  
traceroute to www.iitd.ac.in (10.10.211.212), 64 hops max  
1  10.184.0.14      1.523ms  1.646ms  1.561ms  
2  10.254.236.18    2.056ms  1.971ms  1.614ms  
3  10.10.211.212    1.582ms  1.296ms  1.492ms  
aayush@aayush:~/Desktop/COL334-Computer-Networks/Assignment-1$
```

Figure 15: Hop number v/s RTT graph for www.iitd.ac.in

