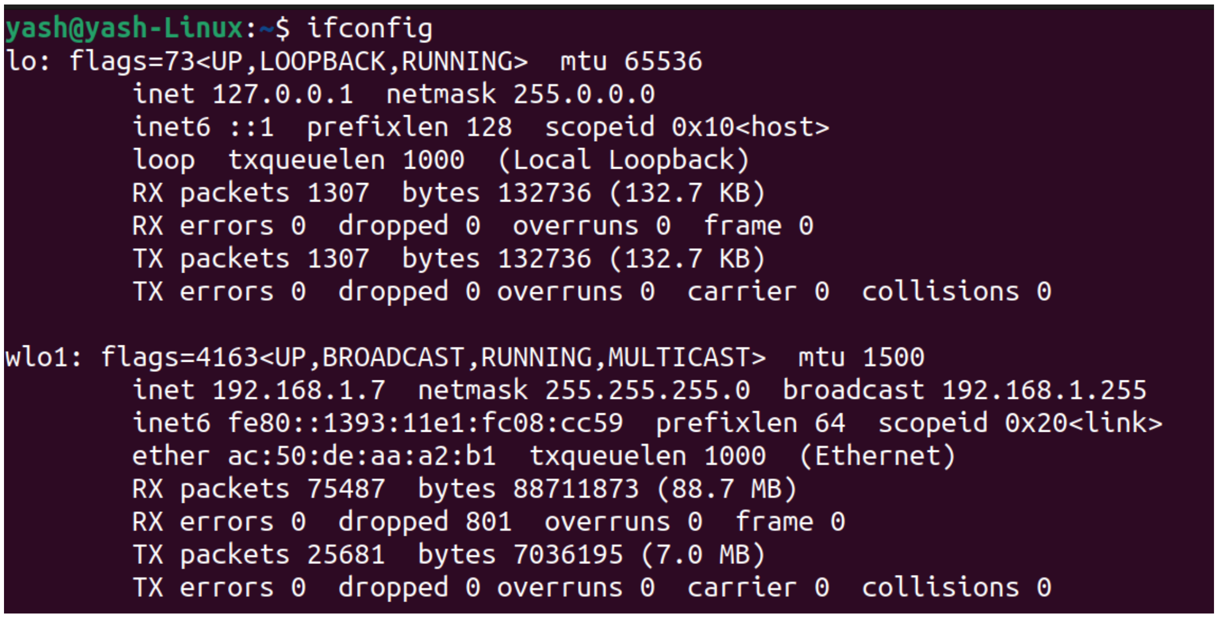
CSE 232: Programming Assignment 1

Using command-line utilities for network debugging

Yash Singh
2022589
Github Repo Link

Q1.

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

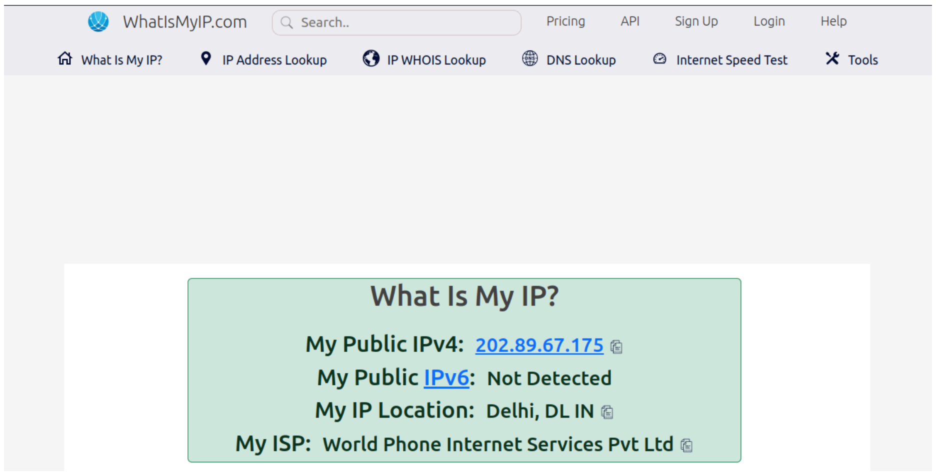


```
yash@yash-Linux:~$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1307  bytes 132736 (132.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1307  bytes 132736 (132.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.7  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::1393:11e1:fc08:cc59  prefixlen 64  scopeid 0x20<link>
        ether ac:50:de:aa:a2:b1  txqueuelen 1000  (Ethernet)
        RX packets 75487  bytes 88711873 (88.7 MB)
        RX errors 0  dropped 801  overruns 0  frame 0
        TX packets 25681  bytes 7036195 (7.0 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

`Lo`: This is the loopback interface used for internal communication within the host. The IP address here is `127.0.0.1`, which is the standard loopback IP address.
`wlo1`: This is a wireless network interface, and the IP address here is `192.168.1.7`. This is the local IP address assigned to my device on the network by the router.

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 https://www.whatismyip.com is different from the one displayed by the `ifconfig` command.



What Is My IP?

My Public IPv4: 202.89.67.175
My Public IPv6: Not Detected
My IP Location: Delhi, DL IN
My ISP: World Phone Internet Services Pvt Ltd

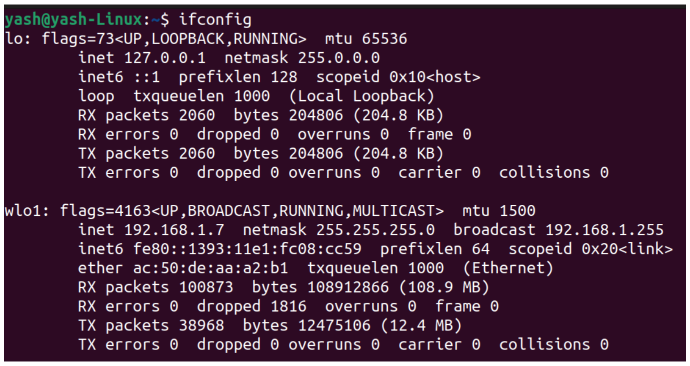
Local vs. Public IP Address:

- **Local IP Address:** The IP address seen using the `ifconfig` command (e.g., `192.168.1.7`) is my local IP address. This address is assigned to my device by router and is used to identify device on local network (e.g., home or office network).
- **Public IP Address:** The IP address shown on a website like `whatismyip.com` is my public IP address. This address is assigned to router by the Internet Service Provider (ISP) and is used to identify my network on the internet. So, `202.89.67.175` is my public ip address.

Q2.

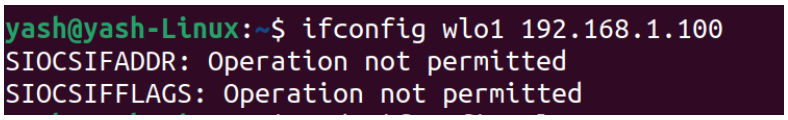
a) **Change the IP address of your network interface using the command line. Put a screenshot that shows the change. Revert to the original IP address.**

Initially IP address of wlo1 interface is show below inet `192.168.1.7`



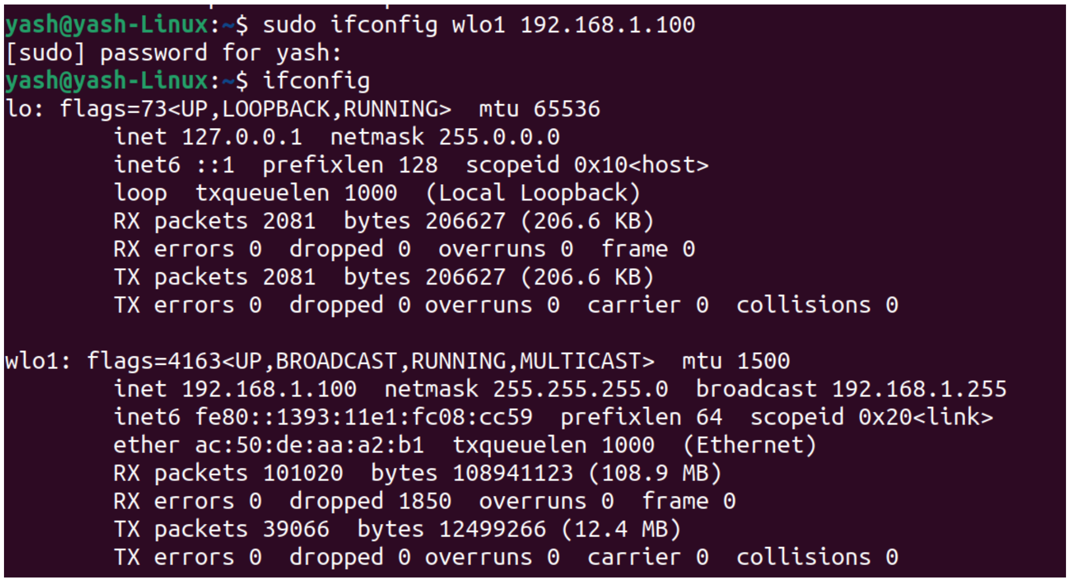
```
yash@yash-Linux:~$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 2060  bytes 204806 (204.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2060  bytes 204806 (204.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.7  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::1393:11e1:fc08:cc59  prefixlen 64  scopeid 0x20<link>
        ether ac:50:de:aa:a2:b1  txqueuelen 1000  (Ethernet)
        RX packets 100873  bytes 108912866 (108.9 MB)
        RX errors 0  dropped 1816  overruns 0  frame 0
        TX packets 38968  bytes 12475106 (12.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The command to change the IP address of interface is `ifconfig wlo1 192.168.1.100` but this gives error SIOCSIFADDR: Operation not permitted SIOCSIFFLAGS: Operation not permitted



```
yash@yash-Linux:~$ ifconfig wlo1 192.168.1.100
SIOCSIFADDR: Operation not permitted
SIOCSIFFLAGS: Operation not permitted
```

The `sudo` command is required to execute these changes with administrative privileges.



```
yash@yash-Linux:~$ sudo ifconfig wlo1 192.168.1.100
[sudo] password for yash:
yash@yash-Linux:~$ ifconfig
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 2081  bytes 206627 (206.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2081  bytes 206627 (206.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.100  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::1393:11e1:fc08:cc59  prefixlen 64  scopeid 0x20<link>
        ether ac:50:de:aa:a2:b1  txqueuelen 1000  (Ethernet)
        RX packets 101020  bytes 108941123 (108.9 MB)
        RX errors 0  dropped 1850  overruns 0  frame 0
        TX packets 39066  bytes 12499266 (12.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Now the ip address of network interface wlo1 is change and the new inet is `192.168.1.100`

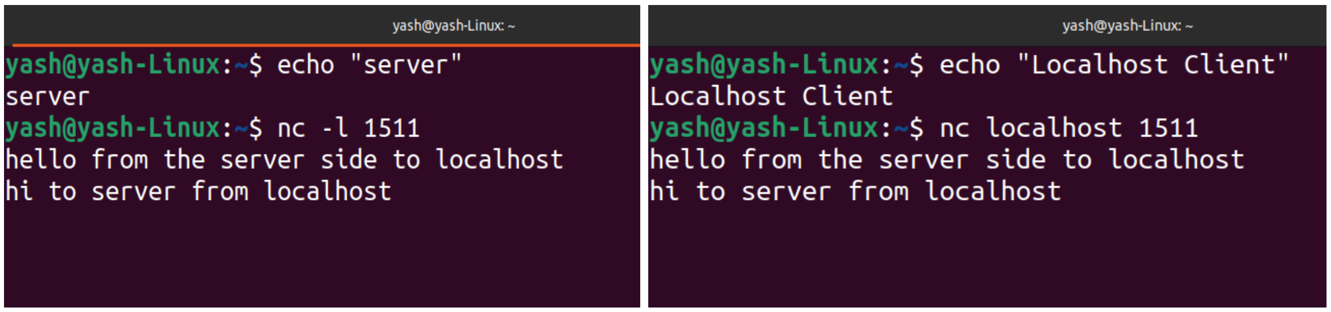
Using the same command `sudo ifconfig wlo1 192.168.1.7` we can revert to the original ip address.

Q3.

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.**

I am working on the linux system so i'll use one terminal for server and one terminal for client. Start the server `nc -l 1511` nc - calls netcat. -l - tells netcat to listen for incoming connections (i.e., sets it as a server). 1511 - the port number on which the server is listening.

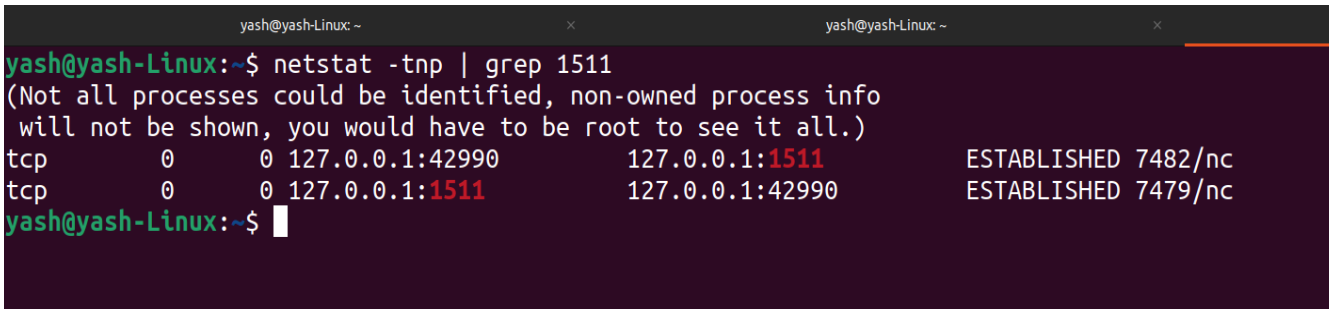
Here our client is localhost using the command `nc localhost 1511.`



```
yash@yash-Linux:~$ echo "server"
server
yash@yash-Linux:~$ nc -l 1511
hello from the server side to localhost
hi to server from localhost
```

```
yash@yash-Linux:~$ echo "Localhost Client"
Localhost Client
yash@yash-Linux:~$ nc localhost 1511
hello from the server side to localhost
hi to server from localhost
```

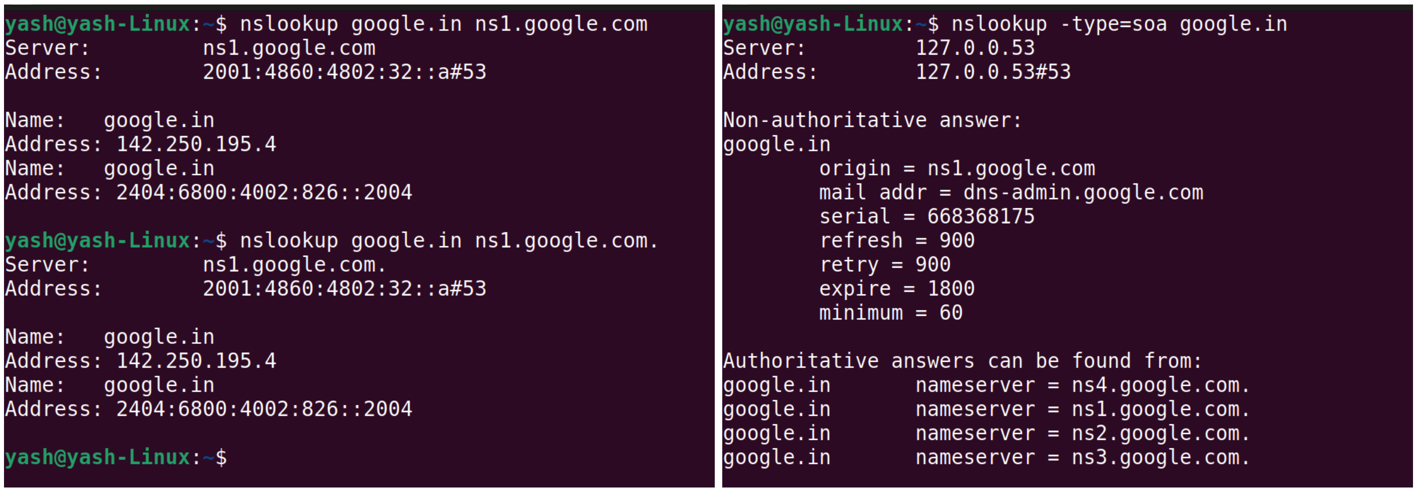
b) Determine the state of this TCP connection(s) at the client node. Put a screenshot.



```
yash@yash-Linux:~$ netstat -tnp | grep 1511
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
tcp        0      0 127.0.0.1:42990         127.0.0.1:1511          ESTABLISHED 7482/nc
tcp        0      0 127.0.0.1:1511          127.0.0.1:42990         ESTABLISHED 7479/nc
yash@yash-Linux:~$
```

Q4.

a) Get an authoritative result for "google.in" using nslookup. Put a screenshot. Explain how you did it.



```
yash@yash-Linux:~$ nslookup google.in ns1.google.com
Server:         ns1.google.com
Address:        2001:4860:4802:32::a#53

Name:   google.in
Address: 142.250.195.4
Name:   google.in
Address: 2404:6800:4002:826::2004

yash@yash-Linux:~$ nslookup google.in ns1.google.com.
Server:         ns1.google.com.
Address:        2001:4860:4802:32::a#53

Name:   google.in
Address: 142.250.195.4
Name:   google.in
Address: 2404:6800:4002:826::2004

yash@yash-Linux:~$
```

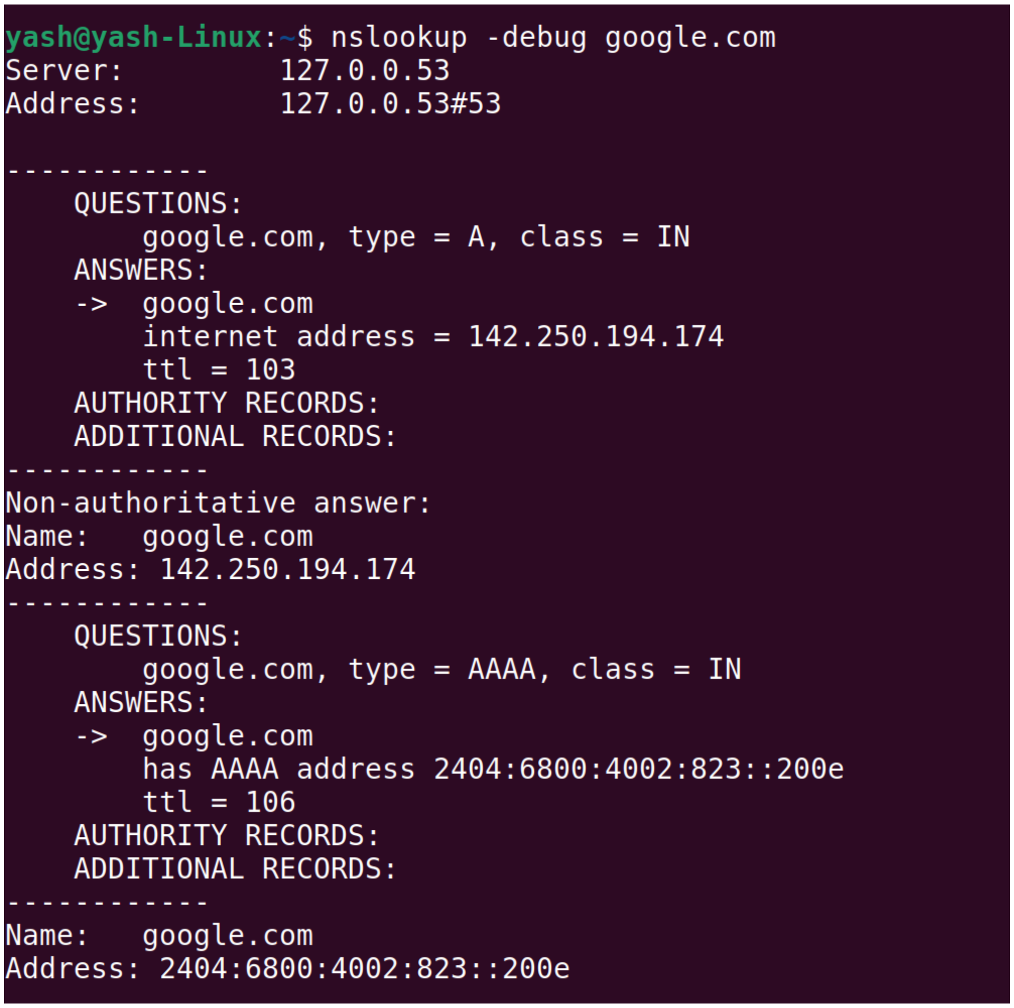
```
yash@yash-Linux:~$ nslookup -type=soa google.in
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
google.in
        origin = ns1.google.com
        mail addr = dns-admin.google.com
        serial = 668368175
        refresh = 900
        retry = 900
        expire = 1800
        minimum = 60

Authoritative answers can be found from:
google.in       nameserver = ns4.google.com.
google.in       nameserver = ns1.google.com.
google.in       nameserver = ns2.google.com.
google.in       nameserver = ns3.google.com.
```

To get an authoritative result for google.in we need to use the command nslookup -type=soa google.in which will give us an authoritative answers server.This will display the Start of Authority (SOA) record, including the primary name server. Then, with command nslookup google.in ns1.google.com. We can get the server details.

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.**

Using the command nslookup -debug google.com in the output we can see the TTL value given. It will be displayed in seconds. The TTL value indicates how long the DNS entry will remain in the local DNS cache before it expires. DNS resolver will need to fetch the record from the authoritative DNS server again.

Below 103 seconds is the ttl and after 103 seconds dns resolver will fetch the new record form the dns server again for google.com

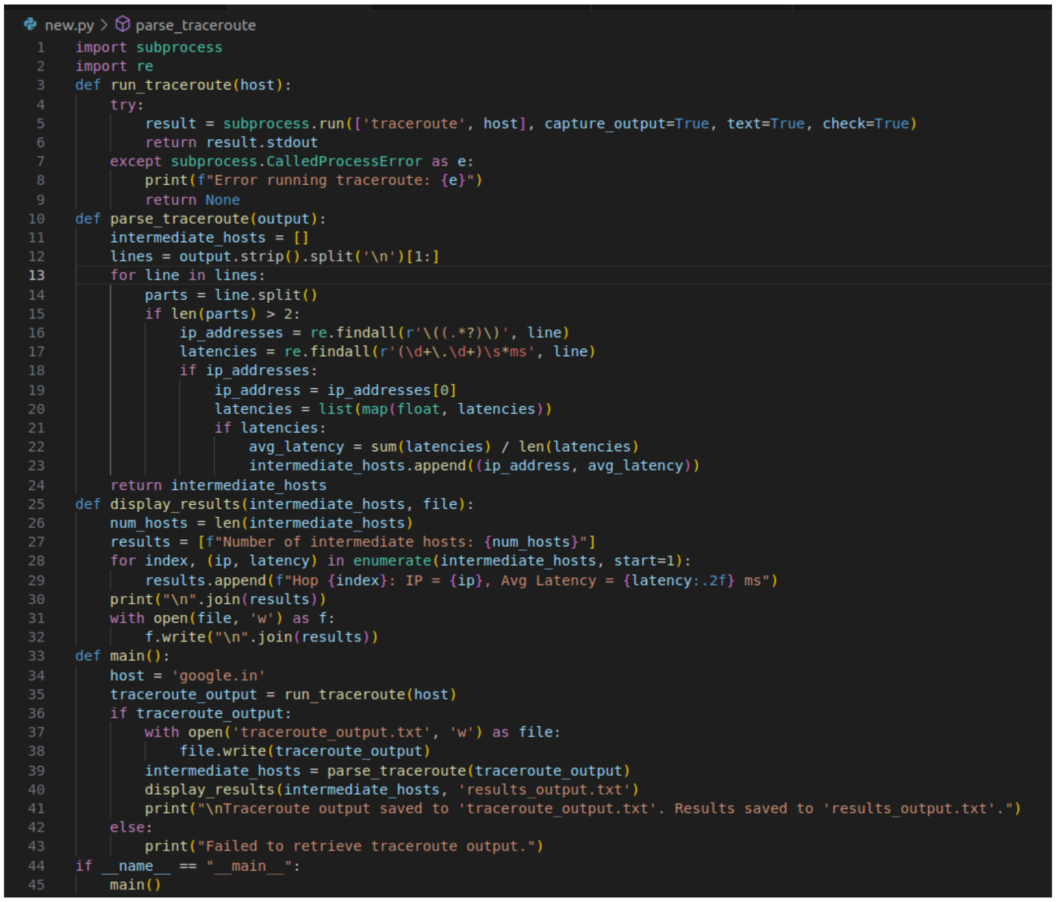


```
yash@yash-Linux:~$ nslookup -debug google.com
Server:         127.0.0.53
Address:        127.0.0.53#53

------------
    QUESTIONS:
        google.com, type = A, class = IN
    ANSWERS:
    ->  google.com
        internet address = 142.250.194.174
        ttl = 103
    AUTHORITY RECORDS:
    ADDITIONAL RECORDS:
------------
Non-authoritative answer:
Name:   google.com
Address: 142.250.194.174
------------
    QUESTIONS:
        google.com, type = AAAA, class = IN
    ANSWERS:
    ->  google.com
        has AAAA address 2404:6800:4002:823::200e
        ttl = 106
    AUTHORITY RECORDS:
    ADDITIONAL RECORDS:
------------
Name:   google.com
Address: 2404:6800:4002:823::200e
```

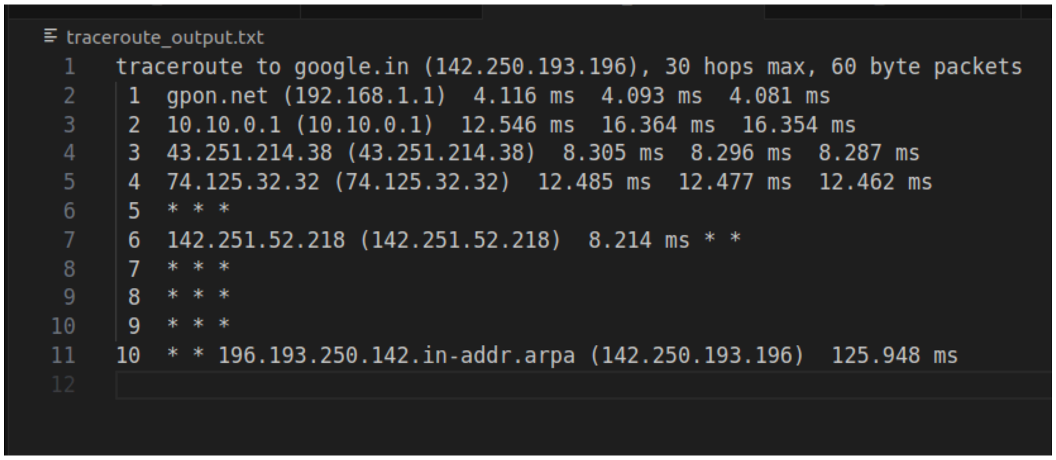
Q5.

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.**

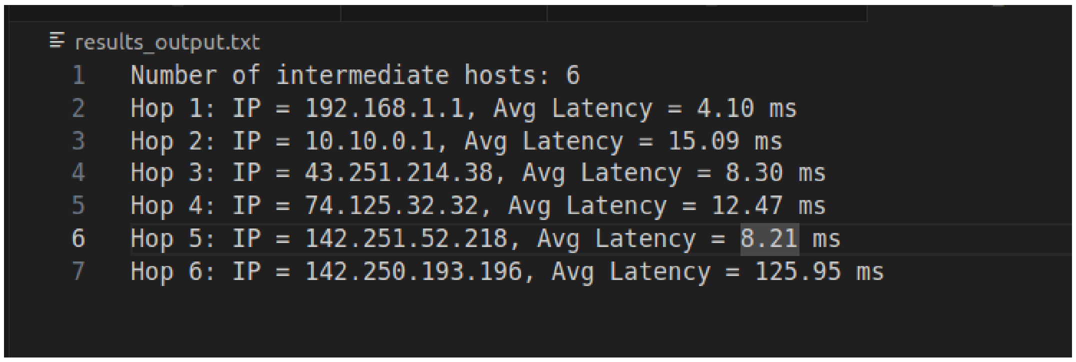
To do this section, I have written a Python script which will run the subprocess for command traceroute google.in and then from the output of this command will find out the number of intermediate hosts and the average latency to each host.


new.py > parse_traceroute

```python
import subprocess
import re
def run_traceroute(host):
    try:
        result = subprocess.run(['traceroute', host], capture_output=True, text=True, check=True)
        return result.stdout
    except subprocess.CalledProcessError as e:
        print(f"Error running traceroute: {e}")
        return None
def parse_traceroute(output):
    intermediate_hosts = []
    lines = output.strip().split('\n')[1:]
    for line in lines:
        parts = line.split()
        if len(parts) > 2:
            ip_addresses = re.findall(r'\((.*?)\)', line)
            latencies = re.findall(r'(\d+\.\d+)\s*ms', line)
            if ip_addresses:
                ip_address = ip_addresses[0]
                latencies = list(map(float, latencies))
                if latencies:
                    avg_latency = sum(latencies) / len(latencies)
                    intermediate_hosts.append((ip_address, avg_latency))
    return intermediate_hosts
def display_results(intermediate_hosts, file):
    num_hosts = len(intermediate_hosts)
    results = [f"Number of intermediate hosts: {num_hosts}"]
    for index, (ip, latency) in enumerate(intermediate_hosts, start=1):
        results.append(f"Hop {index}: IP = {ip}, Avg Latency = {latency:.2f} ms")
    print("\n".join(results))
    with open(file, 'w') as f:
        f.write("\n".join(results))
def main():
    host = 'google.in'
    traceroute_output = run_traceroute(host)
    if traceroute_output:
        with open('traceroute_output.txt', 'w') as file:
            file.write(traceroute_output)
        intermediate_hosts = parse_traceroute(traceroute_output)
        display_results(intermediate_hosts, 'results_output.txt')
        print("\nTraceroute output saved to 'traceroute_output.txt'. Results saved to 'results_output.txt'.")
    else:
        print("Failed to retrieve traceroute output.")
if __name__ == "__main__":
    main()
```

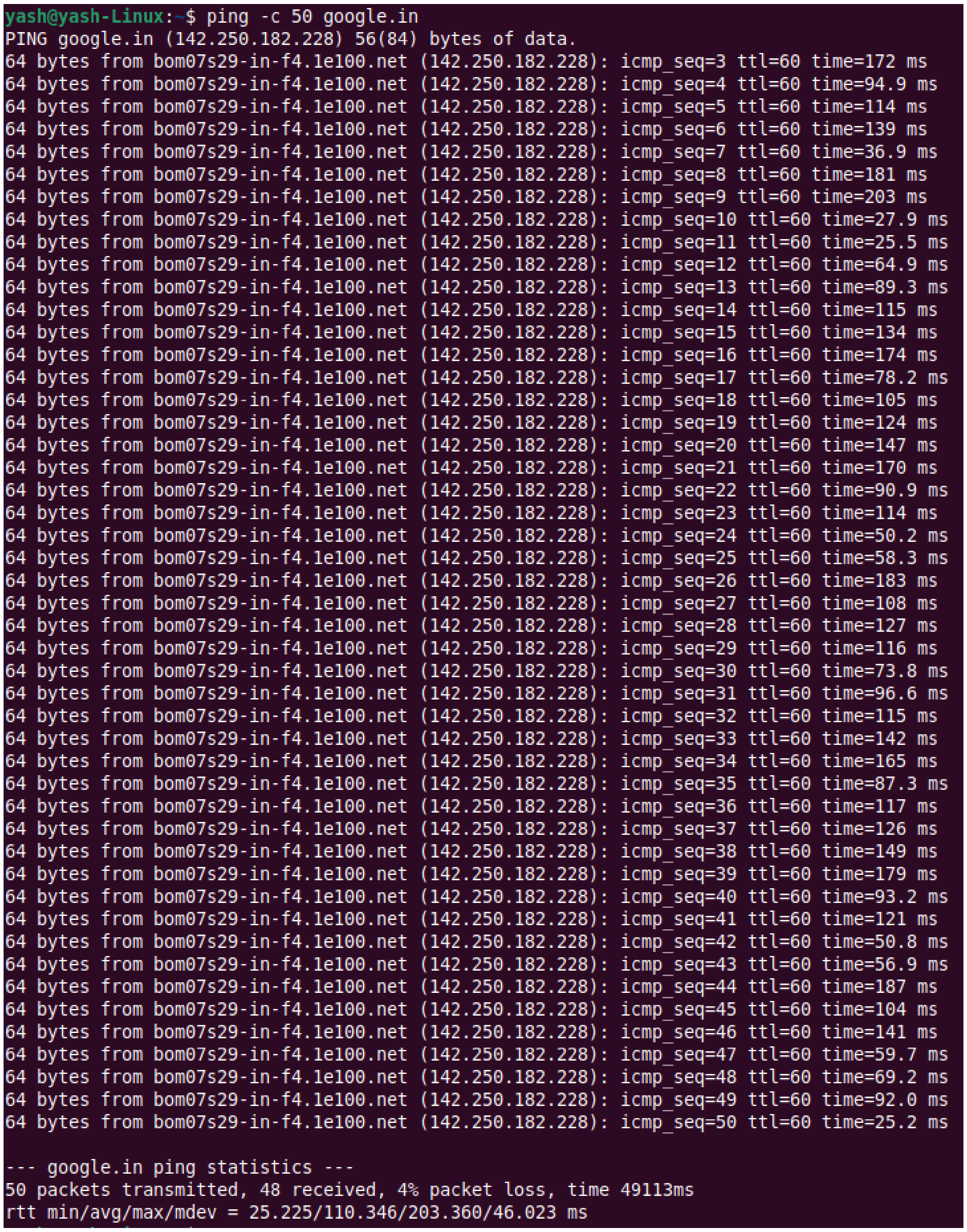

traceroute_output.txt

```
traceroute to google.in (142.250.193.196), 30 hops max, 60 byte packets
 1  gpon.net (192.168.1.1)  4.116 ms  4.093 ms  4.081 ms
 2  10.10.0.1 (10.10.0.1)  12.546 ms  16.364 ms  16.354 ms
 3  43.251.214.38 (43.251.214.38)  8.305 ms  8.296 ms  8.287 ms
 4  74.125.32.32 (74.125.32.32)  12.485 ms  12.477 ms  12.462 ms
 5  * * *
 6  142.251.52.218 (142.251.52.218)  8.214 ms * *
 7  * * *
 8  * * *
 9  * * *
10  * * 196.193.250.142.in-addr.arpa (142.250.193.196)  125.948 ms
```


results_output.txt

```
Number of intermediate hosts: 6
Hop 1: IP = 192.168.1.1, Avg Latency = 4.10 ms
Hop 2: IP = 10.10.0.1, Avg Latency = 15.09 ms
Hop 3: IP = 43.251.214.38, Avg Latency = 8.30 ms
Hop 4: IP = 74.125.32.32, Avg Latency = 12.47 ms
Hop 5: IP = 142.251.52.218, Avg Latency = 8.21 ms
Hop 6: IP = 142.250.193.196, Avg Latency = 125.95 ms
```

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



```
yash@yash-Linux:~$ ping -c 50 google.in
PING google.in (142.250.182.228) 56(84) bytes of data.
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=3 ttl=60 time=172 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=4 ttl=60 time=94.9 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=5 ttl=60 time=114 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=6 ttl=60 time=139 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=7 ttl=60 time=36.9 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=8 ttl=60 time=181 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=9 ttl=60 time=203 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=10 ttl=60 time=27.9 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=11 ttl=60 time=25.5 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=12 ttl=60 time=64.9 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=13 ttl=60 time=89.3 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=14 ttl=60 time=115 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=15 ttl=60 time=134 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=16 ttl=60 time=174 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=17 ttl=60 time=78.2 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=18 ttl=60 time=105 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=19 ttl=60 time=124 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=20 ttl=60 time=147 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=21 ttl=60 time=170 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=22 ttl=60 time=90.9 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=23 ttl=60 time=114 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=24 ttl=60 time=50.2 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=25 ttl=60 time=58.3 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=26 ttl=60 time=183 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=27 ttl=60 time=108 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=28 ttl=60 time=127 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=29 ttl=60 time=116 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=30 ttl=60 time=73.8 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=31 ttl=60 time=96.6 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=32 ttl=60 time=115 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=33 ttl=60 time=142 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=34 ttl=60 time=165 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=35 ttl=60 time=87.3 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=36 ttl=60 time=117 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=37 ttl=60 time=126 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=38 ttl=60 time=149 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=39 ttl=60 time=179 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=40 ttl=60 time=93.2 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=41 ttl=60 time=121 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=42 ttl=60 time=50.8 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=43 ttl=60 time=56.9 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=44 ttl=60 time=187 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=45 ttl=60 time=104 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=46 ttl=60 time=141 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=47 ttl=60 time=59.7 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=48 ttl=60 time=69.2 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=49 ttl=60 time=92.0 ms
64 bytes from bom07s29-in-f4.1e100.net (142.250.182.228): icmp_seq=50 ttl=60 time=25.2 ms

--- google.in ping statistics ---
50 packets transmitted, 48 received, 4% packet loss, time 49113ms
rtt min/avg/max/mdev = 25.225/110.346/203.360/46.023 ms
```

The average latency is 110.346 ms.

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

The sum of the ping latency is 4.0966+15.088+8.296+12.474+8.214+125.948 = 174.1166, which does not match with (b) average latency.

The latencies from `ping` and `traceroute` do not match. This is because `ping` provides the total latency for the entire path (known as Round-Trip Time or RTT), while `traceroute` measures the latency for each individual hop along the path.

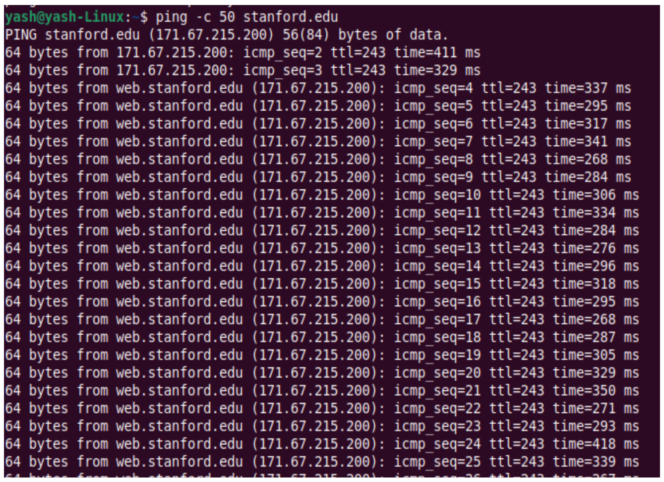
d) Take the maximum ping latency amongst the intermediate hosts (in (a)) and compare it with (b). Are they matching, explain?

The maximum ping latency for an intermediate host in (a) is 125.95 ms, while for the entire trip in (b), it's 203.360 ms. They differ because (a) measures latency to a single hop, whereas (b) reflects the total round-trip time to the destination, including all hops. The cumulative nature of (b) naturally results in a higher latency.

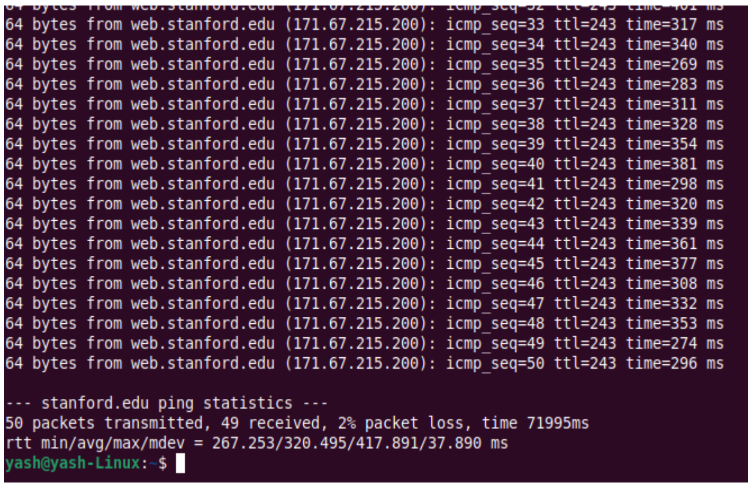
e) You may see multiple entries for a single hop while using the traceroute command. What do these entries mean?

Multiple entries for a single hop in a traceroute result indicate that the packets are taking different paths to reach the same intermediate router or that the router is load-balancing across multiple paths. Each entry represents a different response from the router, showing variations in latency due to the different routes taken or the router's processing.

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



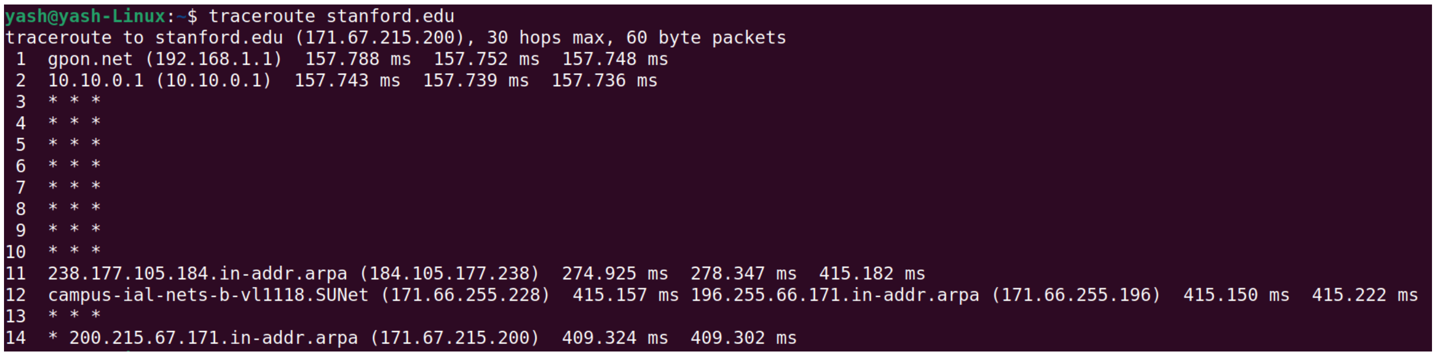
```
yash@yash-Linux:~$ ping -c 50 stanford.edu
PING stanford.edu (171.67.215.200) 56(84) bytes of data.
64 bytes from 171.67.215.200: icmp_seq=2 ttl=243 time=411 ms
64 bytes from 171.67.215.200: icmp_seq=3 ttl=243 time=329 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=4 ttl=243 time=337 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=5 ttl=243 time=295 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=6 ttl=243 time=317 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=7 ttl=243 time=341 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=8 ttl=243 time=268 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=9 ttl=243 time=284 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=10 ttl=243 time=306 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=11 ttl=243 time=334 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=12 ttl=243 time=284 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=13 ttl=243 time=276 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=14 ttl=243 time=296 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=15 ttl=243 time=318 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=16 ttl=243 time=295 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=17 ttl=243 time=268 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=18 ttl=243 time=287 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=19 ttl=243 time=305 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=20 ttl=243 time=329 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=21 ttl=243 time=350 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=22 ttl=243 time=271 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=23 ttl=243 time=293 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=24 ttl=243 time=418 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=25 ttl=243 time=339 ms
```



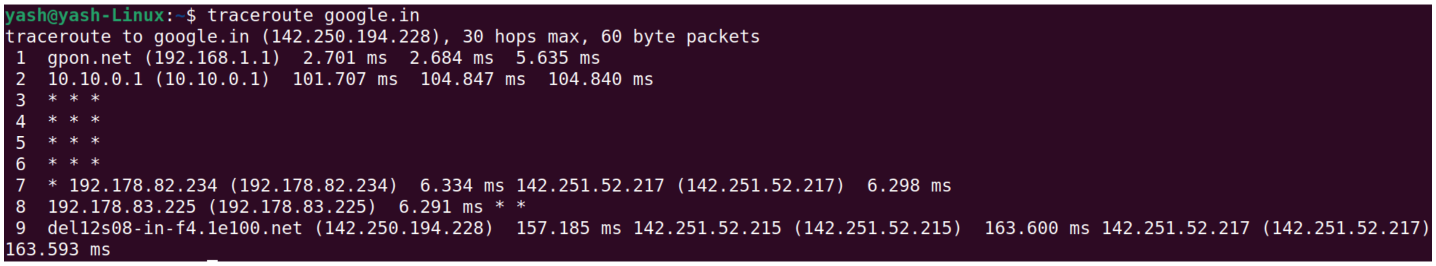
```
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=33 ttl=243 time=317 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=34 ttl=243 time=340 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=35 ttl=243 time=269 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=36 ttl=243 time=283 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=37 ttl=243 time=311 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=38 ttl=243 time=328 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=39 ttl=243 time=354 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=40 ttl=243 time=381 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=41 ttl=243 time=298 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=42 ttl=243 time=320 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=43 ttl=243 time=339 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=44 ttl=243 time=361 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=45 ttl=243 time=377 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=46 ttl=243 time=308 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=47 ttl=243 time=332 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=48 ttl=243 time=353 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=49 ttl=243 time=274 ms
64 bytes from web.stanford.edu (171.67.215.200): icmp_seq=50 ttl=243 time=296 ms

--- stanford.edu ping statistics ---
50 packets transmitted, 49 received, 2% packet loss, time 71995ms
rtt min/avg/max/mdev = 267.253/320.495/417.891/37.890 ms
yash@yash-Linux:~$ 
```

The average latency is 320.495 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).



```
yash@yash-Linux:~$ traceroute stanford.edu
traceroute to stanford.edu (171.67.215.200), 30 hops max, 60 byte packets
 1  gpon.net (192.168.1.1)  157.788 ms  157.752 ms  157.748 ms
 2  10.10.0.1 (10.10.0.1)  157.743 ms  157.739 ms  157.736 ms
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  238.177.105.184.in-addr.arpa (184.105.177.238)  274.925 ms  278.347 ms  415.182 ms
12  campus-ial-nets-b-vl1118.SUNet (171.66.255.228)  415.157 ms 196.255.66.171.in-addr.arpa (171.66.255.196)  415.150 ms  415.222 ms
13  * * *
14  * 200.215.67.171.in-addr.arpa (171.67.215.200)  409.324 ms  409.302 ms
```



```
yash@yash-Linux:~$ traceroute google.in
traceroute to google.in (142.250.194.228), 30 hops max, 60 byte packets
 1  gpon.net (192.168.1.1)  2.701 ms  2.684 ms  5.635 ms
 2  10.10.0.1 (10.10.0.1)  101.707 ms  104.847 ms  104.840 ms
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * 192.178.82.234 (192.178.82.234)  6.334 ms 142.251.52.217 (142.251.52.217)  6.298 ms
 8  192.178.83.225 (192.178.83.225)  6.291 ms * *
 9  del12s08-in-f4.1e100.net (142.250.194.228)  157.185 ms 142.251.52.215 (142.251.52.215)  163.600 ms 142.251.52.217 (142.251.52.217)
163.593 ms
```

No of hops for (stanford.edu) = 14
No of hops for (google.in) = 9

No of hops for stanford.edu is more than google.in

h) Can you explain the reason for the latency difference between google.in and stanford.edu (see (b) & (f))? [1]

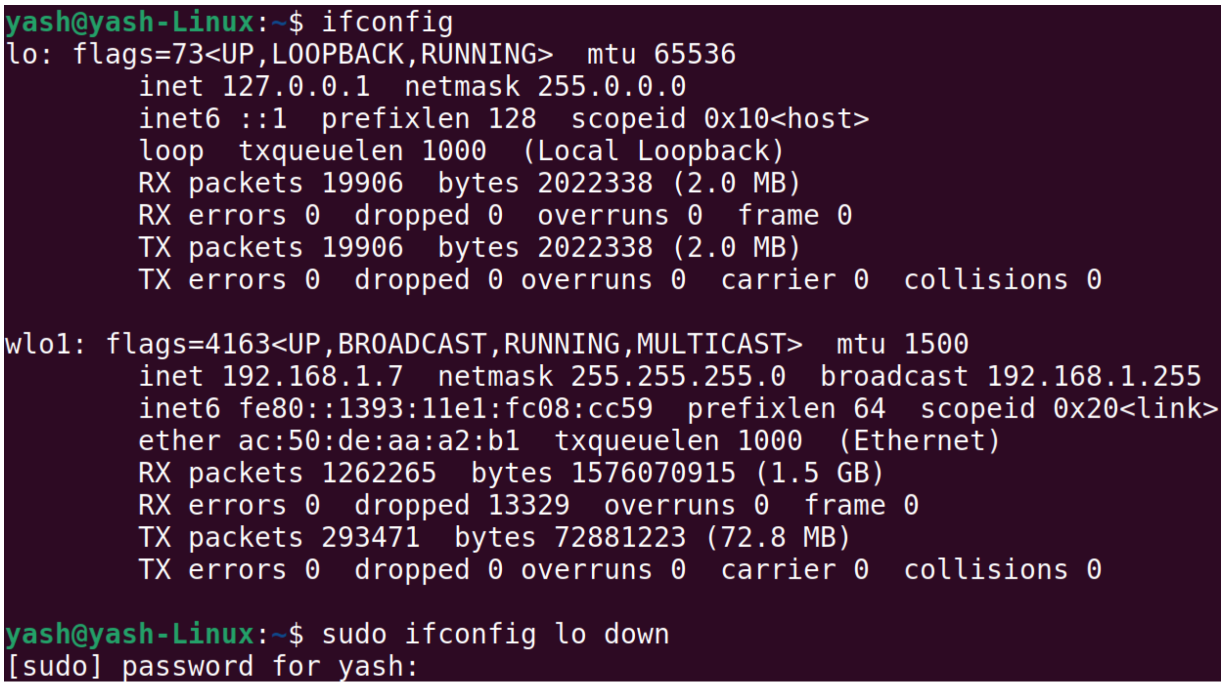
The latency for stanford.edu is much higher than for google.in primarily due to geographical factors. Google India servers are located in India, which is closer to me, while Stanford server is in the USA, much farther away. This means that data packets have to travel a longer distance and pass through more intermediate devices and routers to reach the Stanford server.

Google is a company with a vast infrastructure, which helps them respond to queries quickly and efficiently. Their extensive network and optimization significantly reduce latency. On the other hand, the path to Stanford involves more intermediate networks and potentially less optimized infrastructure, which contributes to the higher latency.

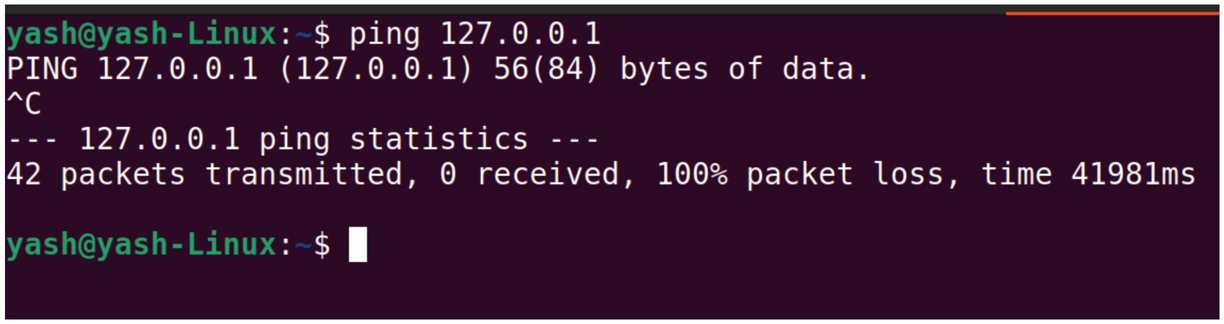
Q6)

a) **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.**

To make the `ping` command fail for 127.0.0.1, first disable the loopback interface by running `sudo ifconfig lo down` in the terminal. Then, execute the `ping 127.0.0.1` command. This will result in 100% packet loss, as the loopback interface is down and cannot respond.



```
yash@yash-Linux:~$ ifconfig
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 19906  bytes 2022338 (2.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 19906  bytes 2022338 (2.0 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.7  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::1393:11e1:fc08:cc59  prefixlen 64  scopeid 0x20<link>
        ether ac:50:de:aa:a2:b1  txqueuelen 1000  (Ethernet)
        RX packets 1262265  bytes 1576070915 (1.5 GB)
        RX errors 0  dropped 13329  overruns 0  frame 0
        TX packets 293471  bytes 72881223 (72.8 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

yash@yash-Linux:~$ sudo ifconfig lo down
[sudo] password for yash:
```



```
yash@yash-Linux:~$ 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 ---
42 packets transmitted, 0 received, 100% packet loss, time 41981ms

yash@yash-Linux:~$ 
```