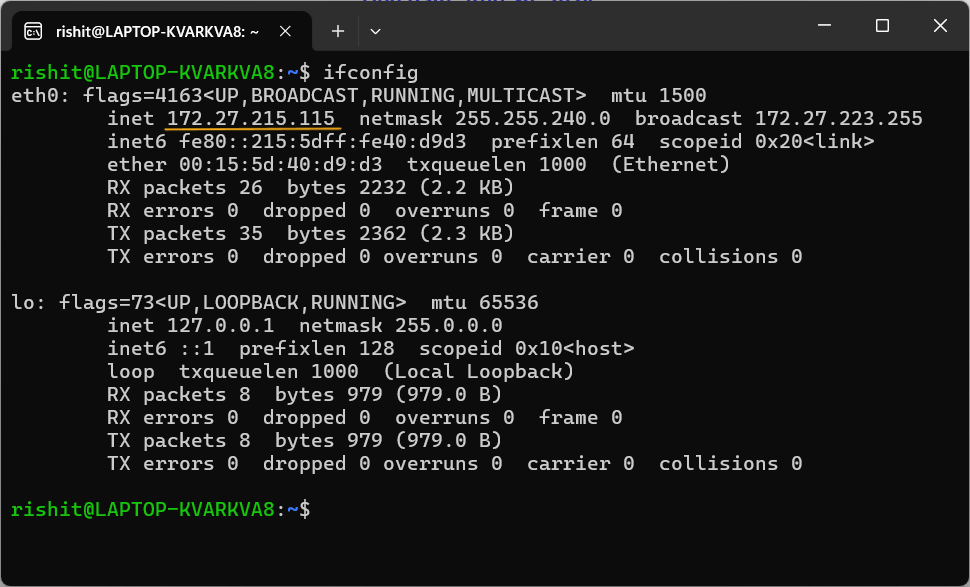
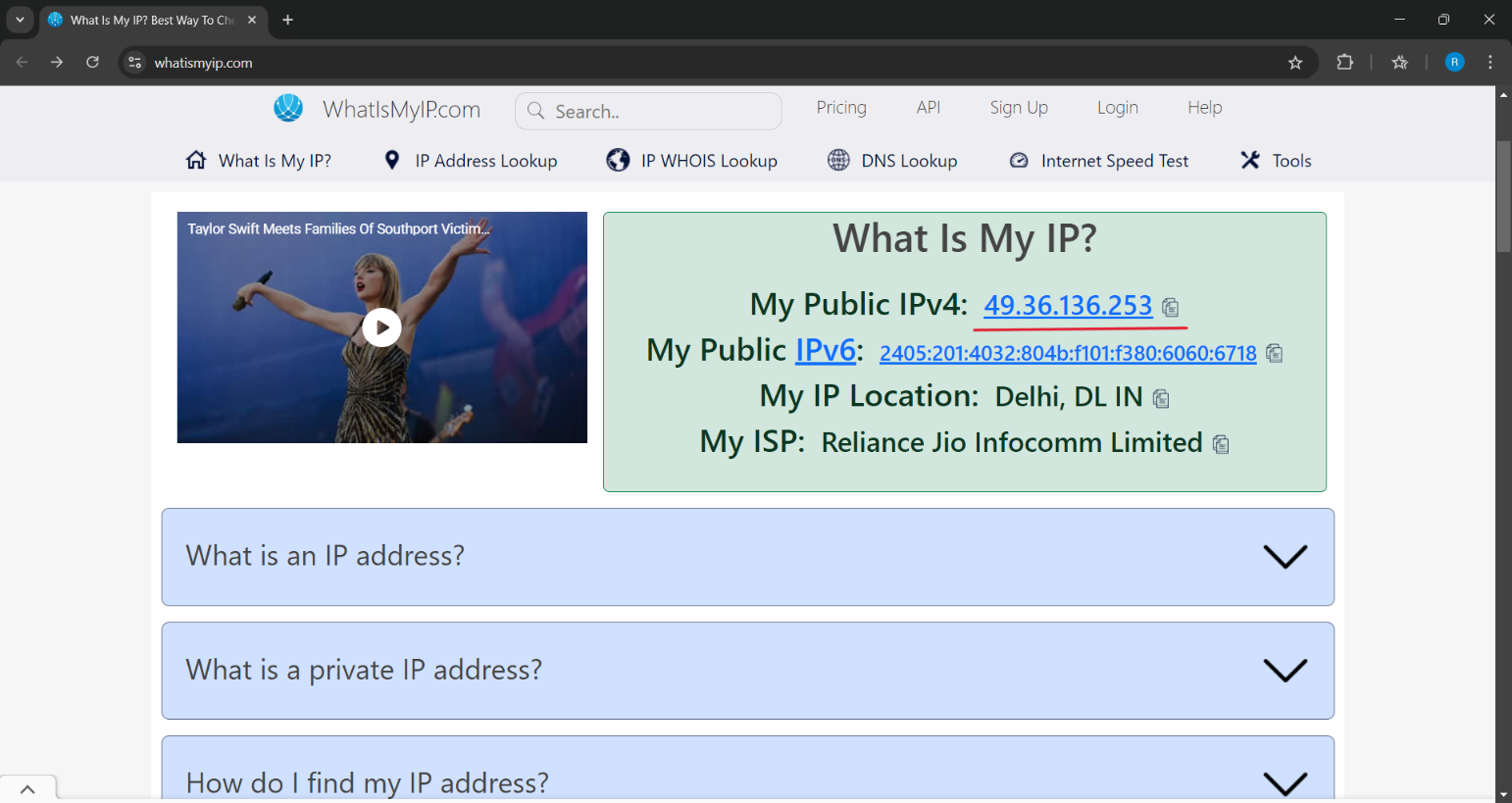
**Computer Networks Assignment-1**

**Using command-line utilities for network debugging**

**Ans 1.a)** Using **ifconfig** command: 

IP address of my “eth0” interface: **172.27.215.115**

**b)** Using [WhatIsMyIP](https://www.whatismyip.com/) website:



IP address shown on the website: **49.36.136.253**

Both of the IP addresses are different. This is because:

There are 2 types of IP addresses: **Public IP** and **Private IP** addresses.

My Public IP is a part of a **Wide Area Network (WAN).** It covers a large area. For eg, internet.

A **Local Area Network (LAN)** refers to my internal network. It consists of private IP addresses.

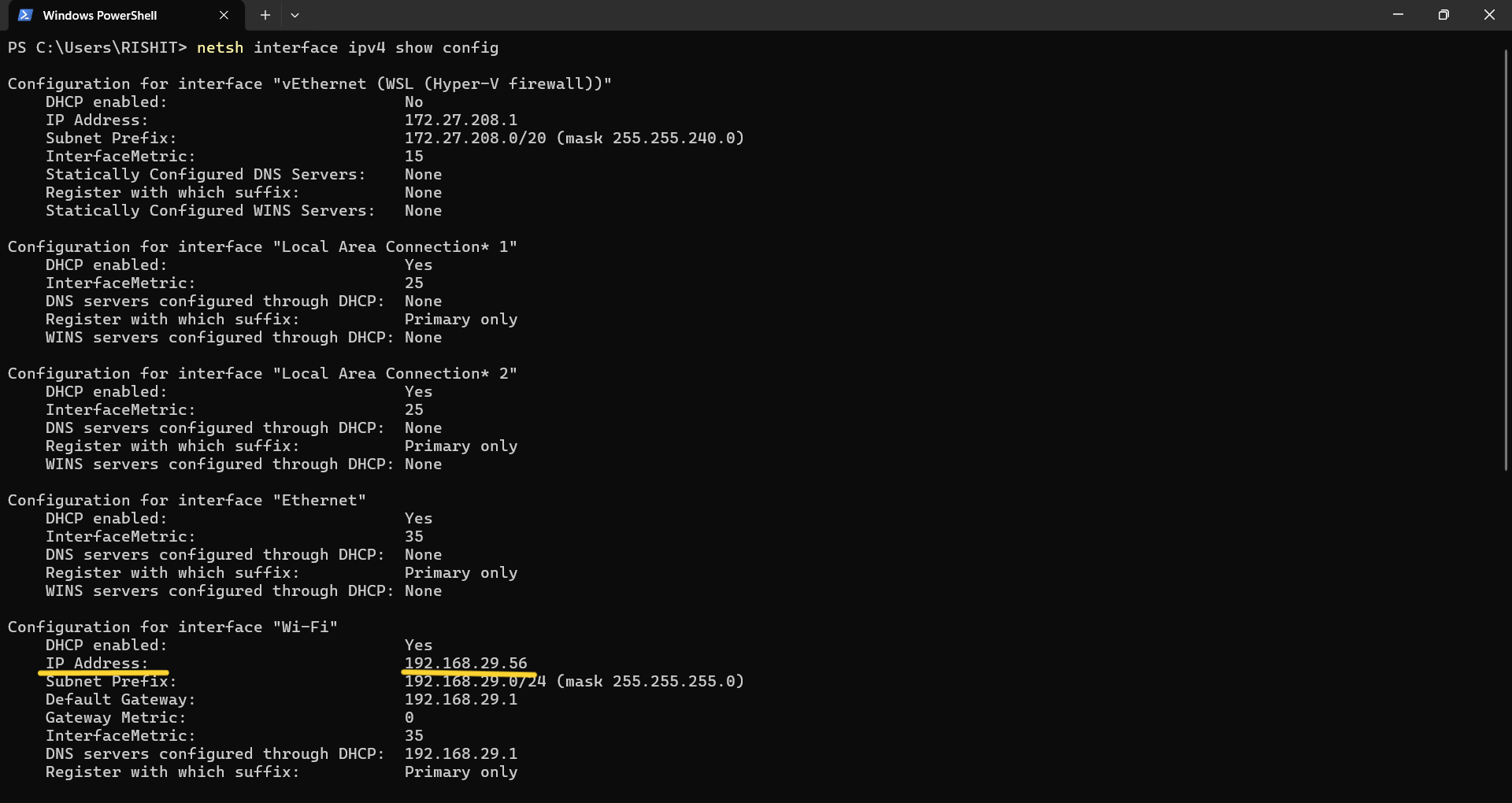
The IP address shown on [WhatIsMyIP](https://www.whatismyip.com/) website is the Public IP, specifically the router or modem.

The IP address shown with the **ifconfig** command is the Private IP assigned to my device specifically. **ifconfig** gives private IP.

**Ans 2. a)** Original IP address for my “Wi-Fi” interface: **192.168.29.56**

Command used: **netsh interface ipv4 show config**

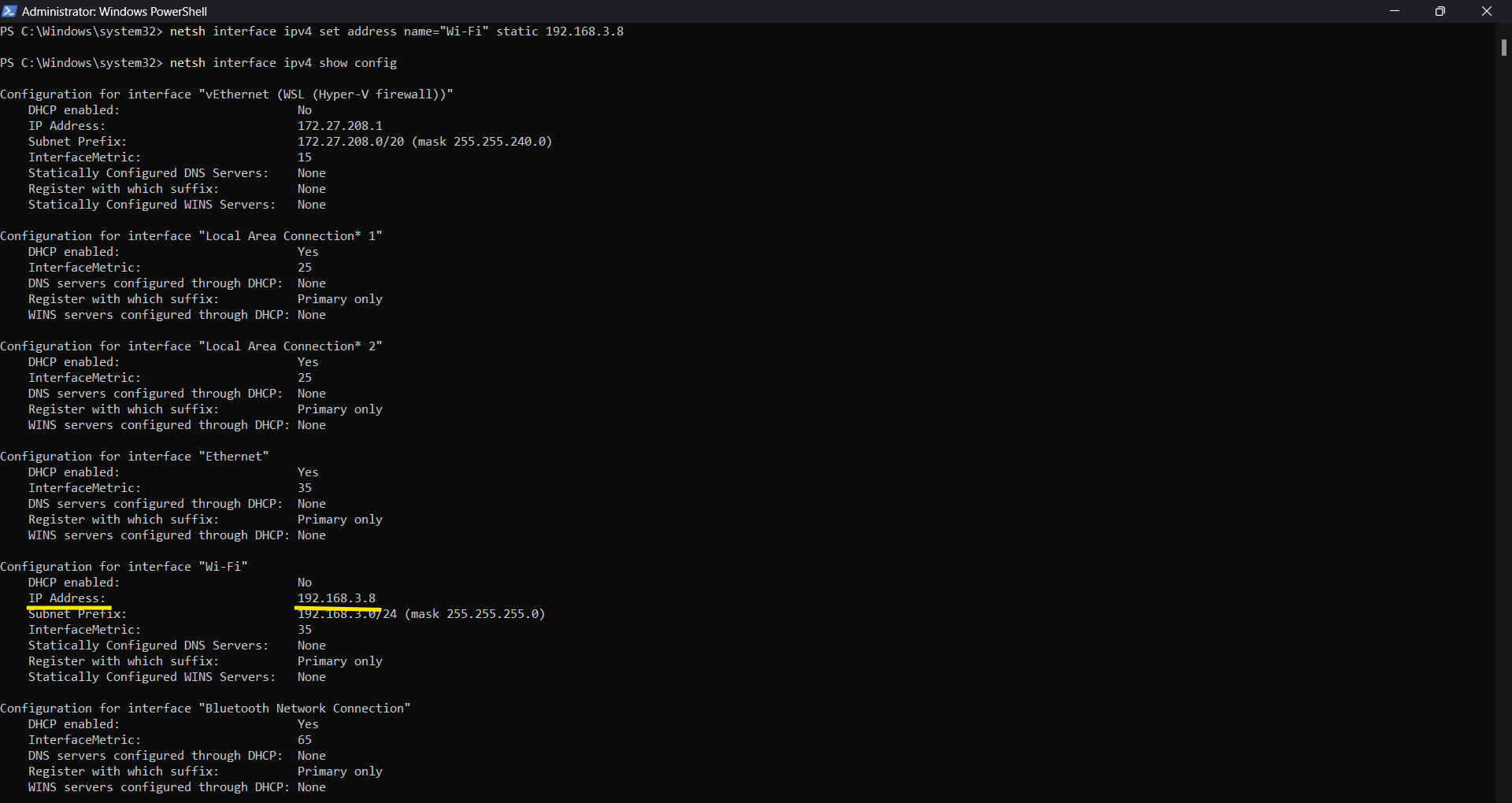
It displays the system’s current IP configuration settings.



Changed IP address for my “Wi-Fi” interface to **192.169.3.8**

Command used: **netsh interface ipv4 set address name="Wi-Fi" static 192.168.3.8**

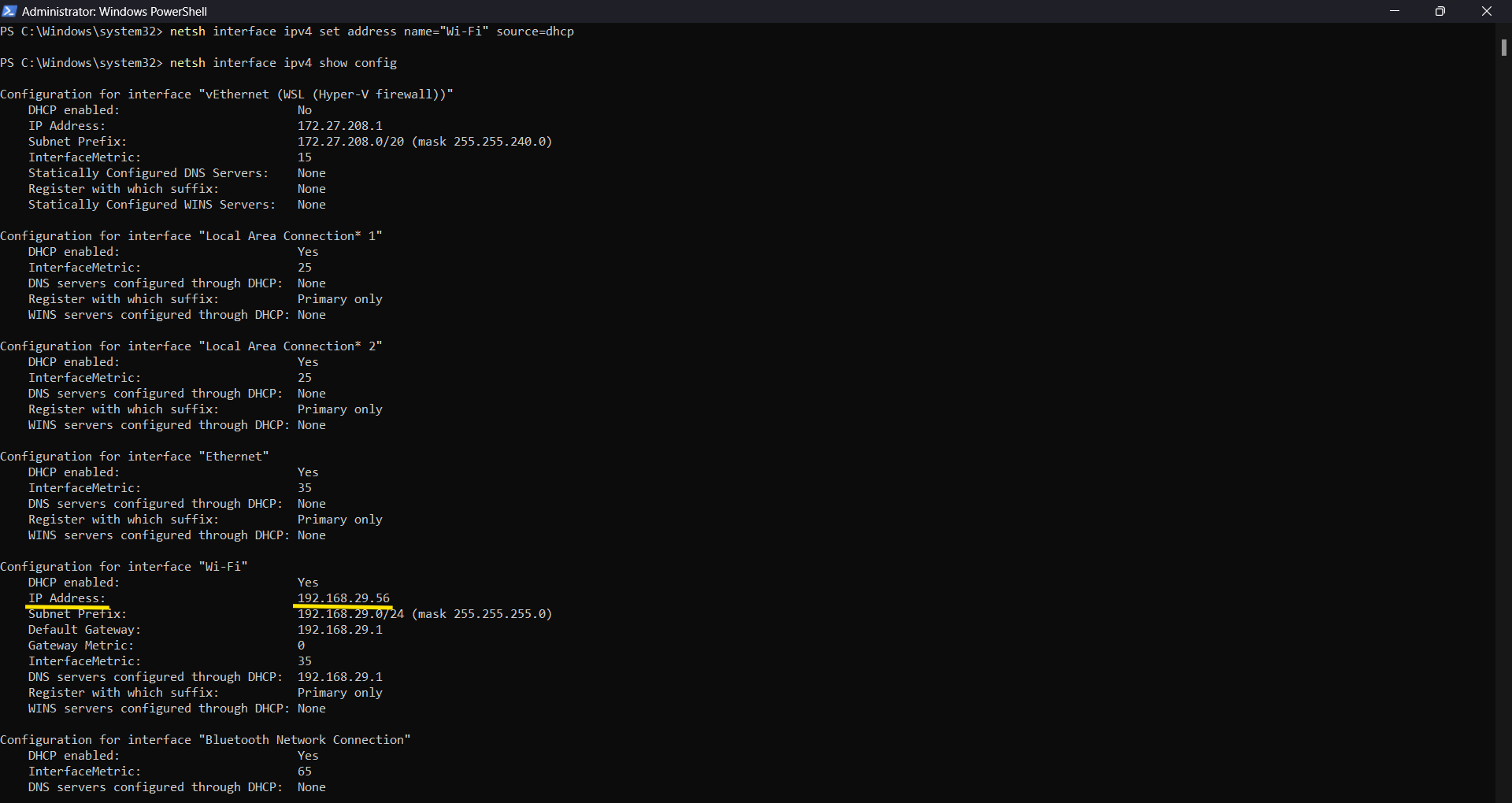
It sets the IPv4 address of the interface “Wi-Fi” to static **192.168.3.8**

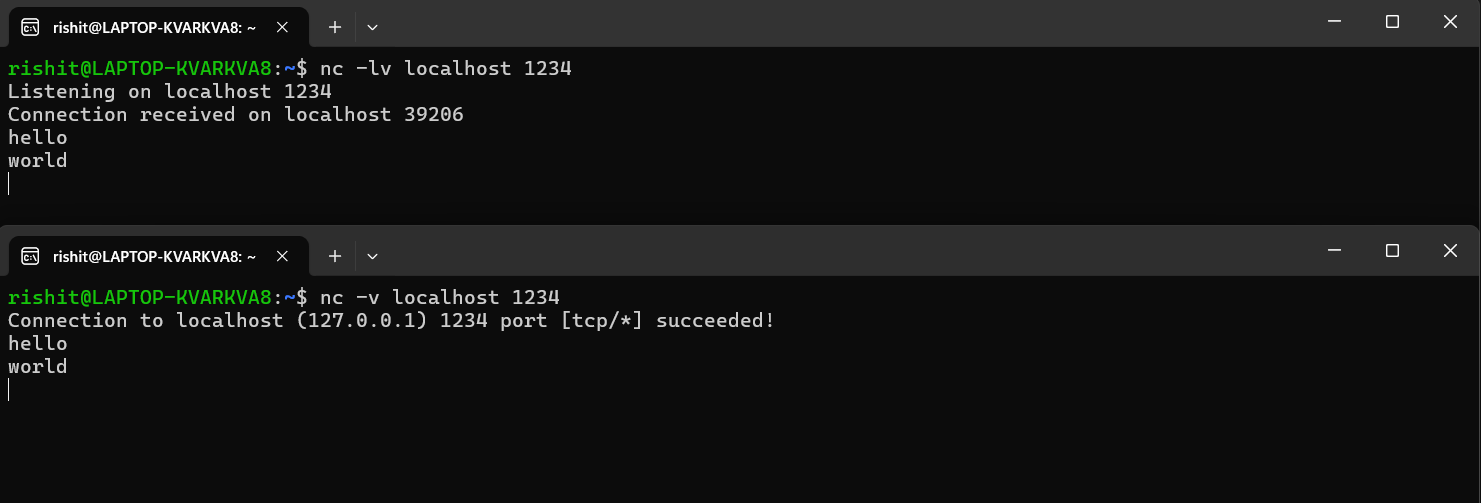


Restored IP address: **192.168.29.56**

Command used: **netsh interface ipv4 set address name="Wi-Fi" source=dhcp**

It restores the original IP by switching to an IP assigned automatically by a DHCP server, like my router.



**Ans 3. a)** 

We listen for any incoming connections on terminal 1 (first screen) using:

**nc -lv localhost 1234**

where, **localhost** is the IP address of the server side and **1234** is the port we listen on.

Terminal 1 acts as a server node.

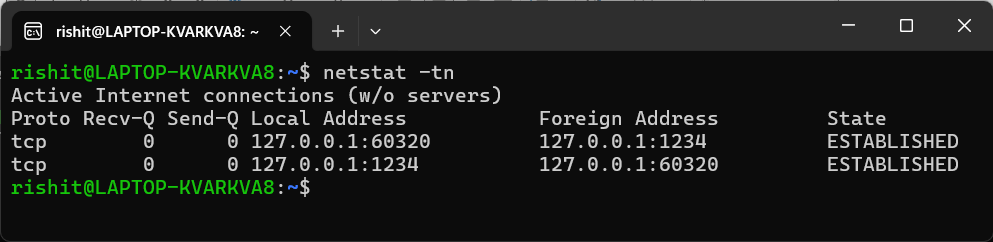
We establish a connection to this server on terminal 2 (second screen) using:

**nc -v localhost 1234**

where, **localhost** is the IP address of the server and **1234** is the port we want to connect to.

Terminal 2 acts as a client node.

By default, the connection is TCP.

**b)** 

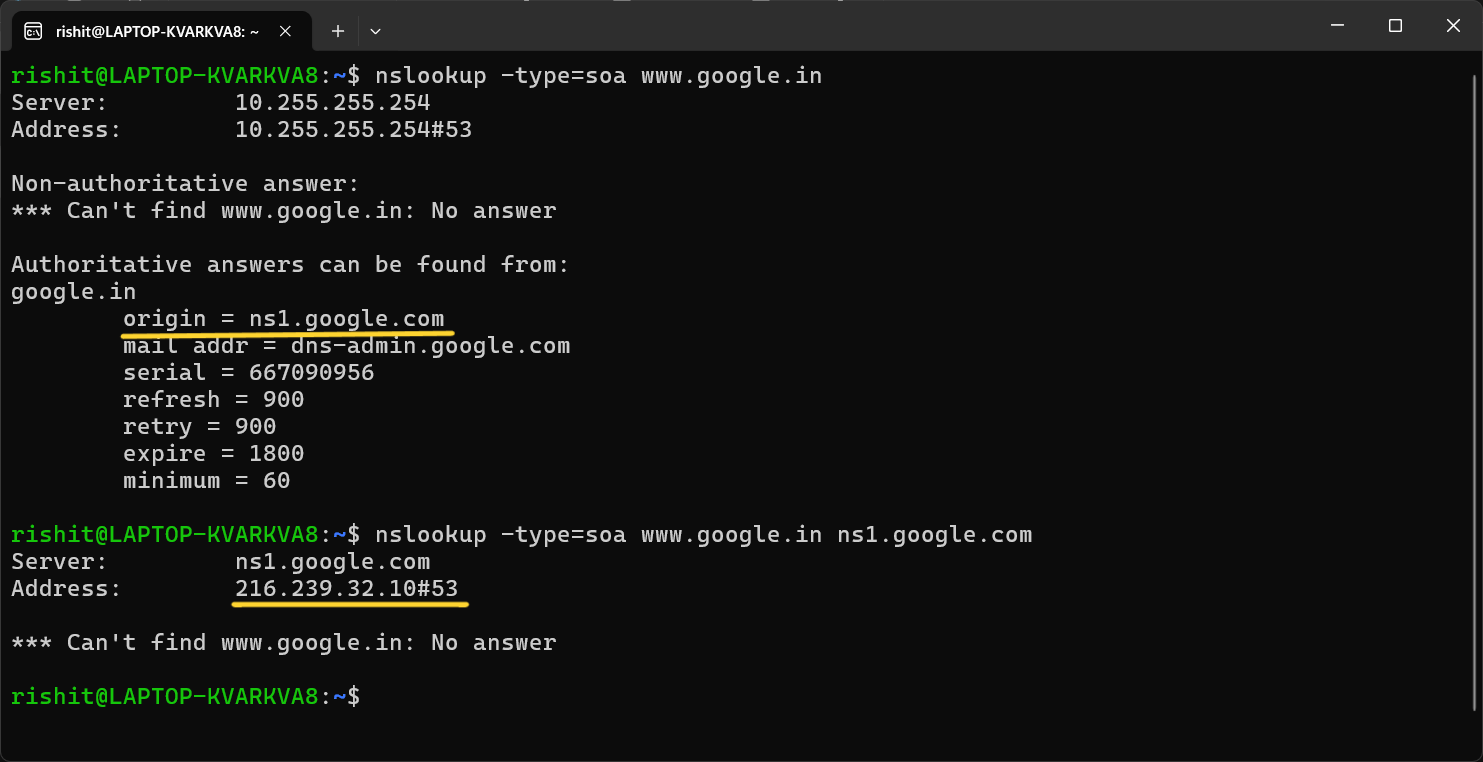
In some 3rd terminal, we run the following command:

**netstat -tn**

This will display a list of active TCP connections on your system.

It shows the socket address: **127.0.0.1:60320** of the client, where **60320** is the port of client.

**127.0.0.1:1234** is the socket address of the server, where **1234** is the port, the server listens on.

**Ans 4. a)** 

First, lookup the authoritative information about the domain **“www.google.in”** using:

**nslookup -type=soa www.google.in**

It will show the origin server for the domain, which is **ns1.google.com**, but not exactly its IP address.

To get the IP address of the origin server, lookup the exact server of the domain using:

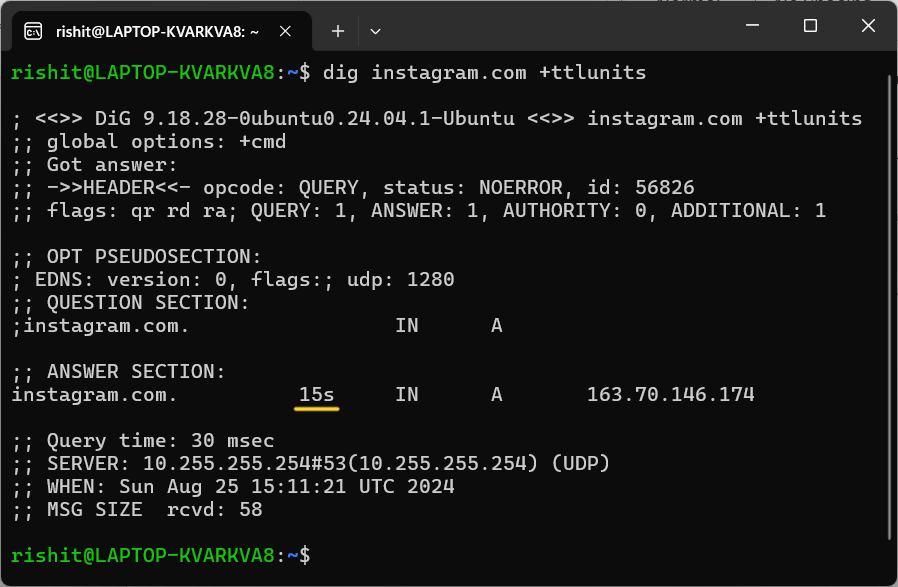
**nslookup -type=soa www.google.in ns1.google.com**

This will show the IP address and port number of the exact server of the domain, which is

**216.239.32.10#53**

where, **216.239.32.10** is the IP address of the origin server and **53** is the port number.

**b)** Let us try **instagram.com** using: **dig instagram.com +ttlunits**

****

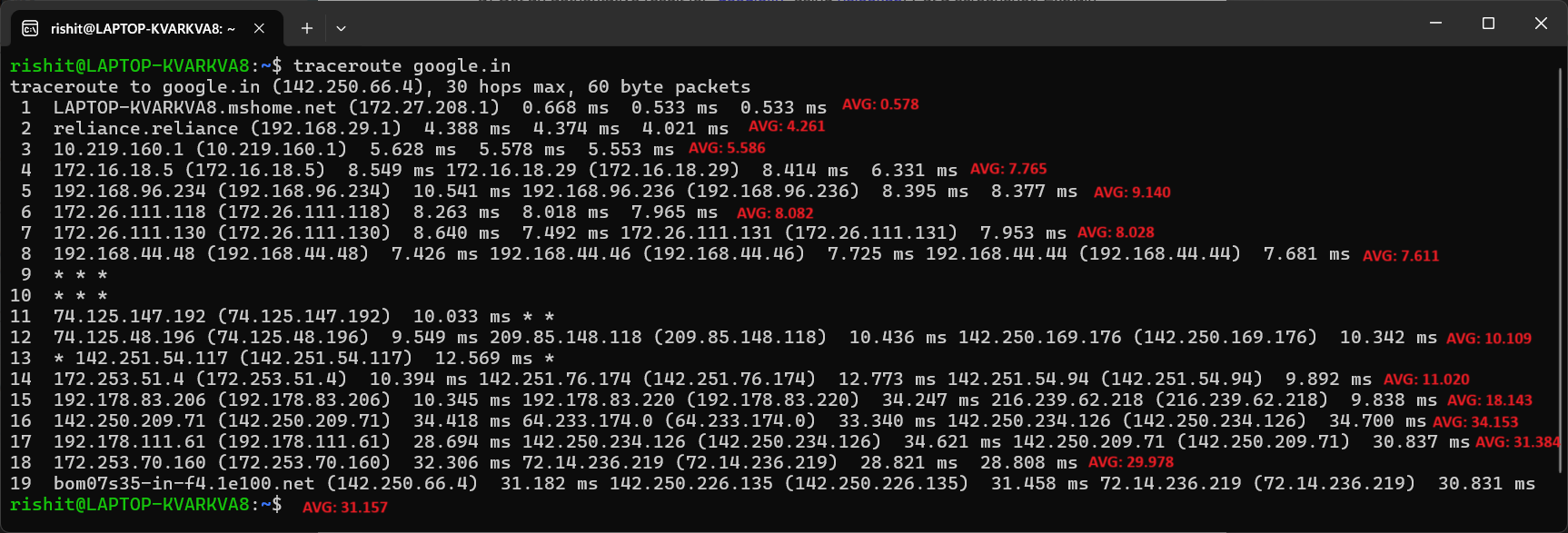
**TTL (Time-To-Live)** value for a DNS record determines how long the local DNS resolver is allowed to cache the particular record. Once the TTL expires, the resolver requests for a fresh value of the record.

We get information about a domain TTL using **dig.** It is present in the **ANSWER** section.

If we use **+ttlunits,** it will show the TTL in human-readable form (eg, mins, secs etc.)

In the **HEADER** section, **ra** flag indicates the query was answered from the cache by the default local resolver on the system. For a non-cached answer by the authoritative nameserver, we would see **aa.**

In the case of **instagram.com**, TTL is **15s.** This means that this entry would expire from the local DNS server after 15s. After which, it would issue a fresh request.

**Ans 5. a)** 

**NOTE:** I’ve calculated the average latency of each hop and edited it onto the screenshot itself.

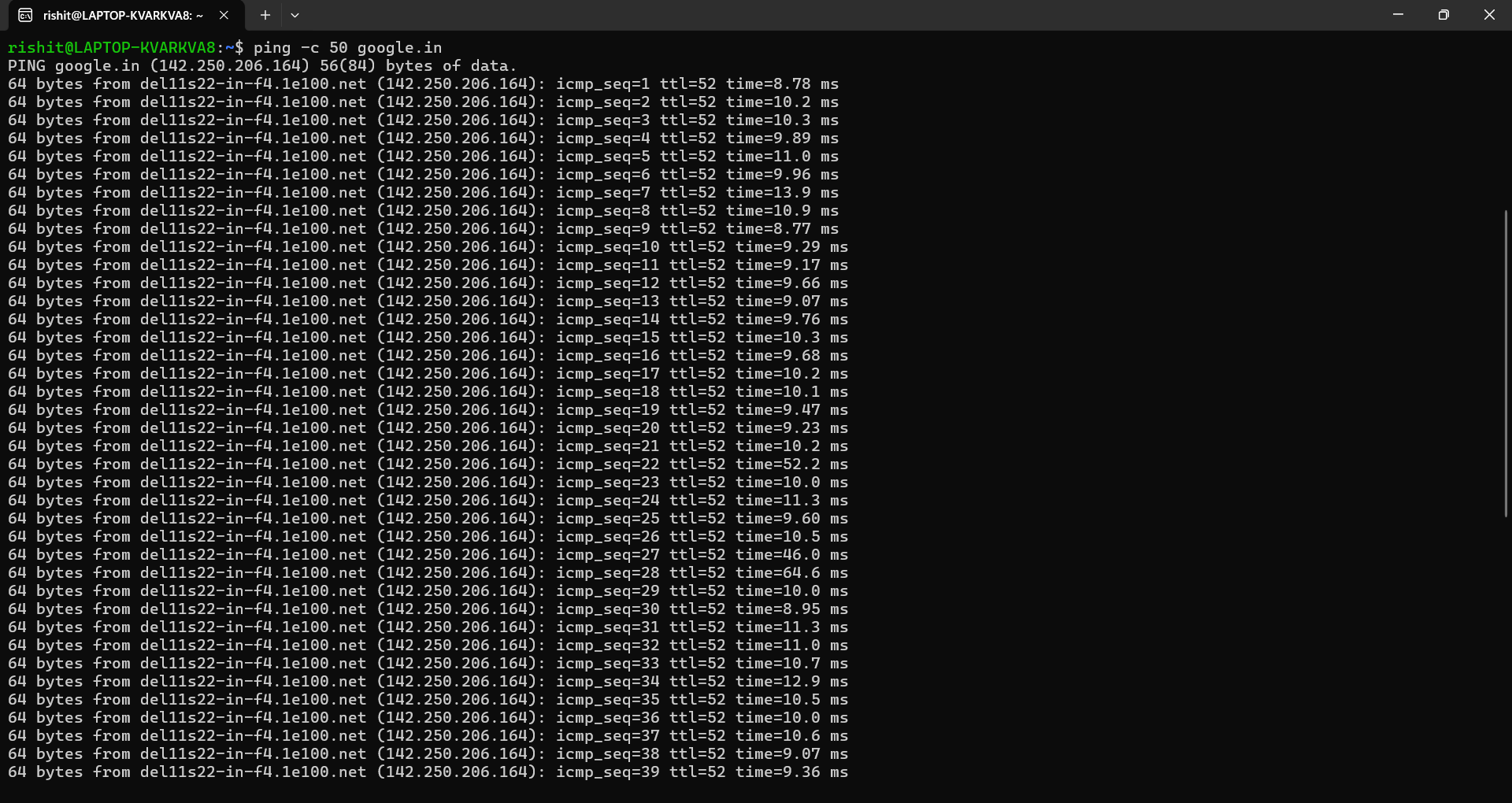
**NOTE:** I’m not considering those routers that contain even a single star (\*).

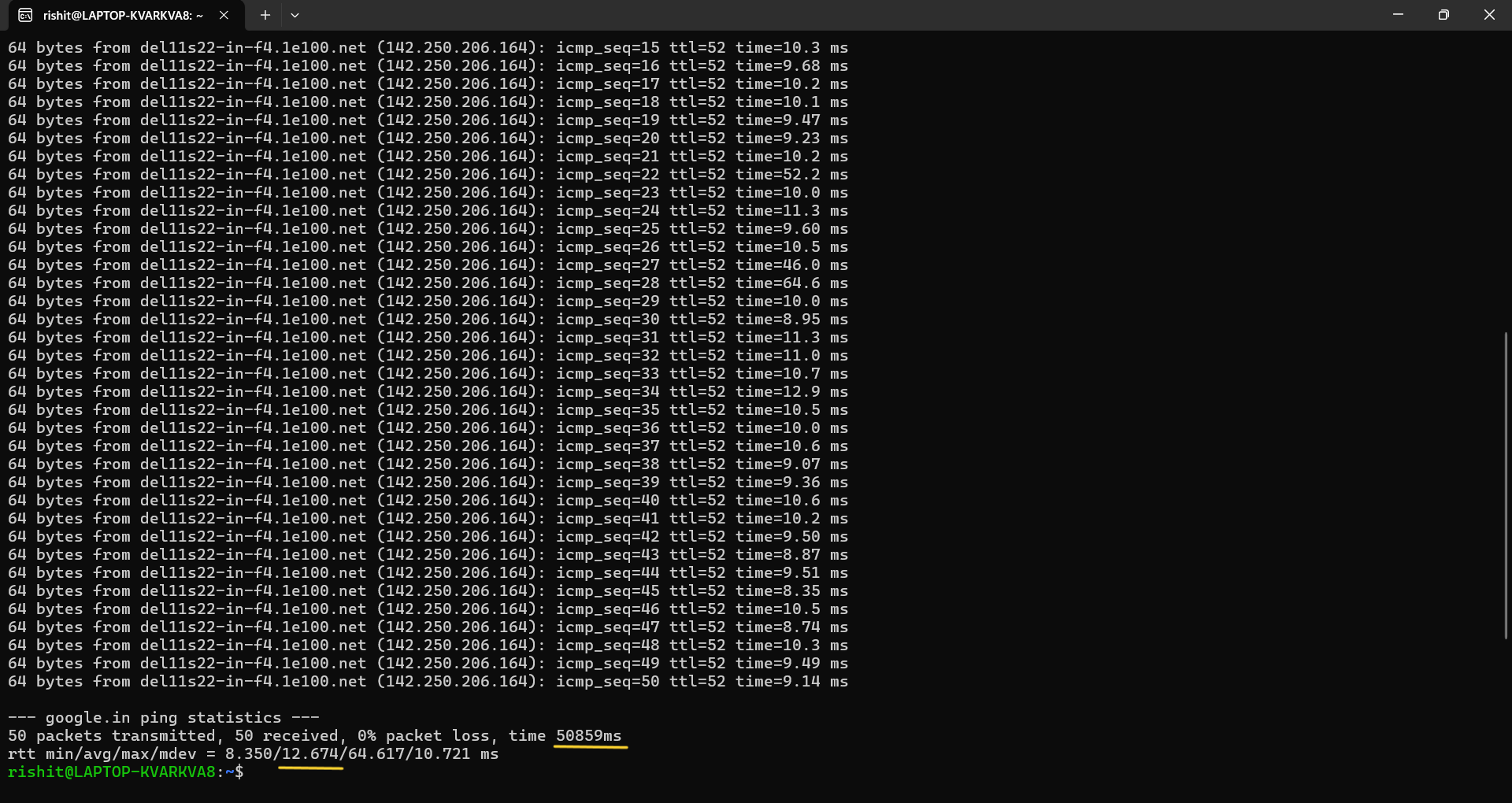
**NOTE:** I’m not considering the last hop (19th hop) to be the intermediate host, since it is the destination.

There are **14** intermediate hosts (hops). Their IP addresses and the computed average latency for each host is given in the screenshot above.

**traceroute** sends 3 packets of data for each hop. This gives us 3 **RTTs** (Round-Trip times)

**b)**

****



Average latency using **ping** by sending 50 packets: **12.674 ms**

**c) NOTE:** Since, the question is not very clear (☹), I’ll be comparing:

1. Sum of **traceroute** hop latencies v/s Sum of **ping** latencies

Sum of average latencies of each hop in **a),** **185.838 ms**

Sum of latencies of all data packets in **b),** **50859 ms**

This difference is because of the no. of packets that we are sending using **ping** is 50, so their sum will consequently be very high compared to the minimum required no. of hops we get using **traceroute**.

1. Avg. of **traceroute** latencies v/s Avg. of **ping** latencies

Average ping latency of intermediate hosts in **a)** is **13.27 ms**

We calculated this by summing up the latencies of each hop, **185.838 ms**

Then, dividing this sum by the no. of hops, **14**

Average latency in **b)** is **12.674 ms**

These values obtained by **ping** and **traceroute** are similar, but different. This is because **ping** measures the **RTT** between your device and a target IP address by sending ICMP Echo Request packets and waiting for the Echo Reply.

**traceroute** measures the latency to each hop along the path to the target IP address. It sends multiple packets with incrementally increasing **TTL** values and records the time it takes for each hop to return a "Time Exceeded" message or the final Echo Reply from the target.

Because **traceroute** measures the time to each intermediate hop, it often provides higher latency values for hops closer to the destination due to the accumulated delays at each hop. Also, network traffic or routing policies at intermediate hops can affect **traceroute** results, making the average latency different from that observed with **ping**.

Hence, **ping** provides the average round-trip latency directly to the target, whereas **traceroute** provides latency to each hop along the route, which can result in different average values.

**d) NOTE:** I’ll be comparing the average latency in **b)**

Maximum ping latency among all intermediate hosts in **a)** = **34.153 ms**

Average ping latency in **b)** = **12.674 ms**

These two latencies do not match because **traceroute** includes the latencies of each hop (router or gateway) between your machine and the destination.

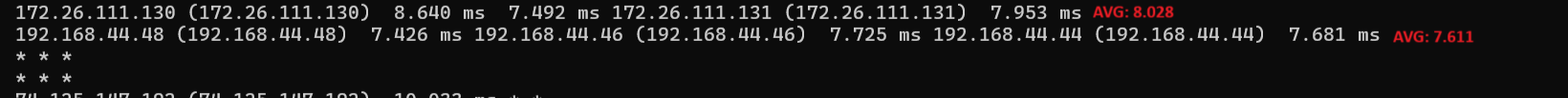
The maximum latency reported by **traceroute** comes from the slowest hop along the path, which could be a router or server temporarily experiencing higher load or delay in processing the packets.

**ping** directly measures the round-trip time to the target server, reflecting a more straightforward and possibly optimized route with less variability.

Intermediate hops might experience congestion or higher processing load, resulting in higher latencies at certain points in the **traceroute**.

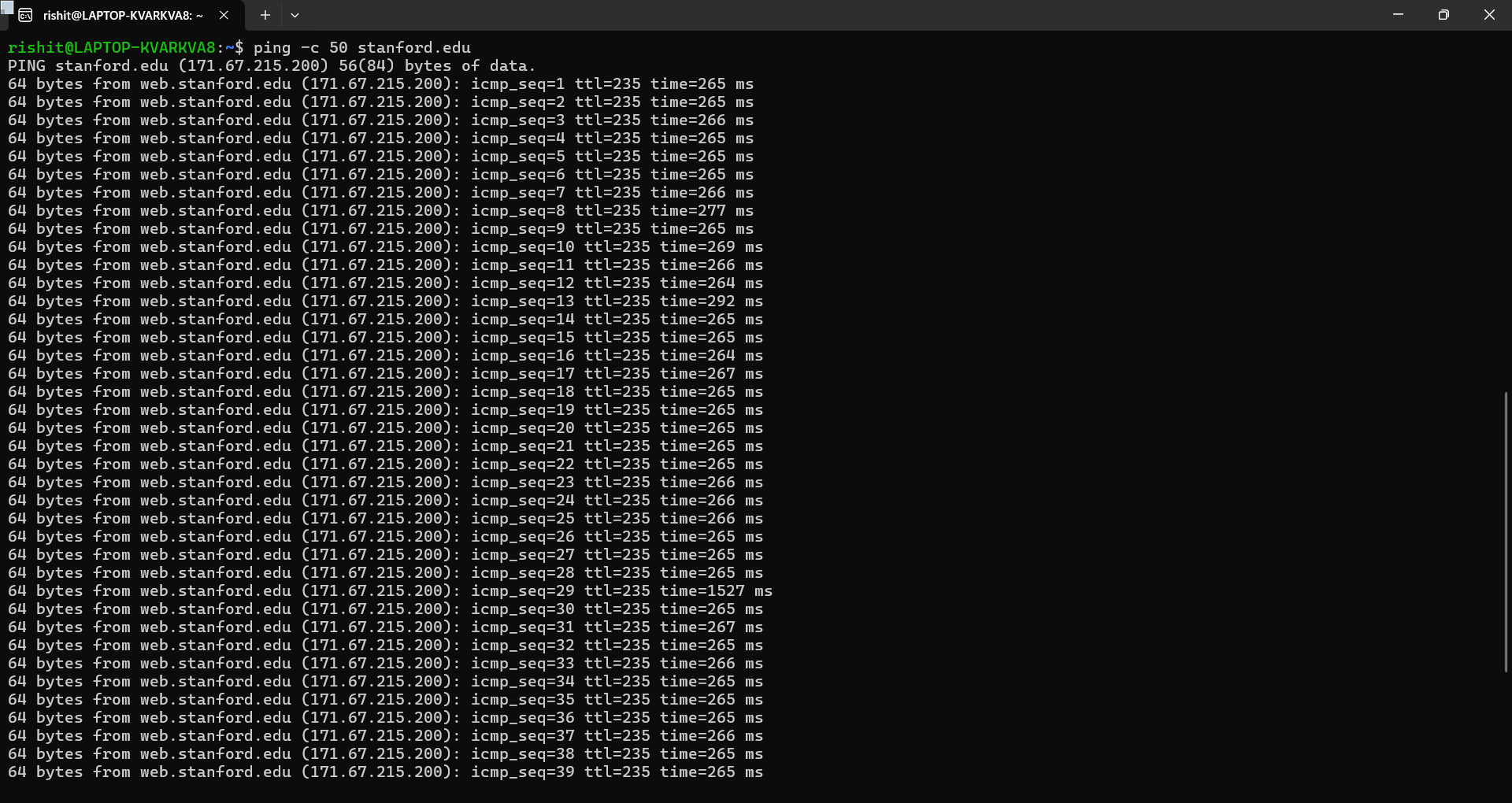
The final destination (e.g., **google.in**) may be optimized for handling network traffic, resulting in lower average latency during the **ping** test.

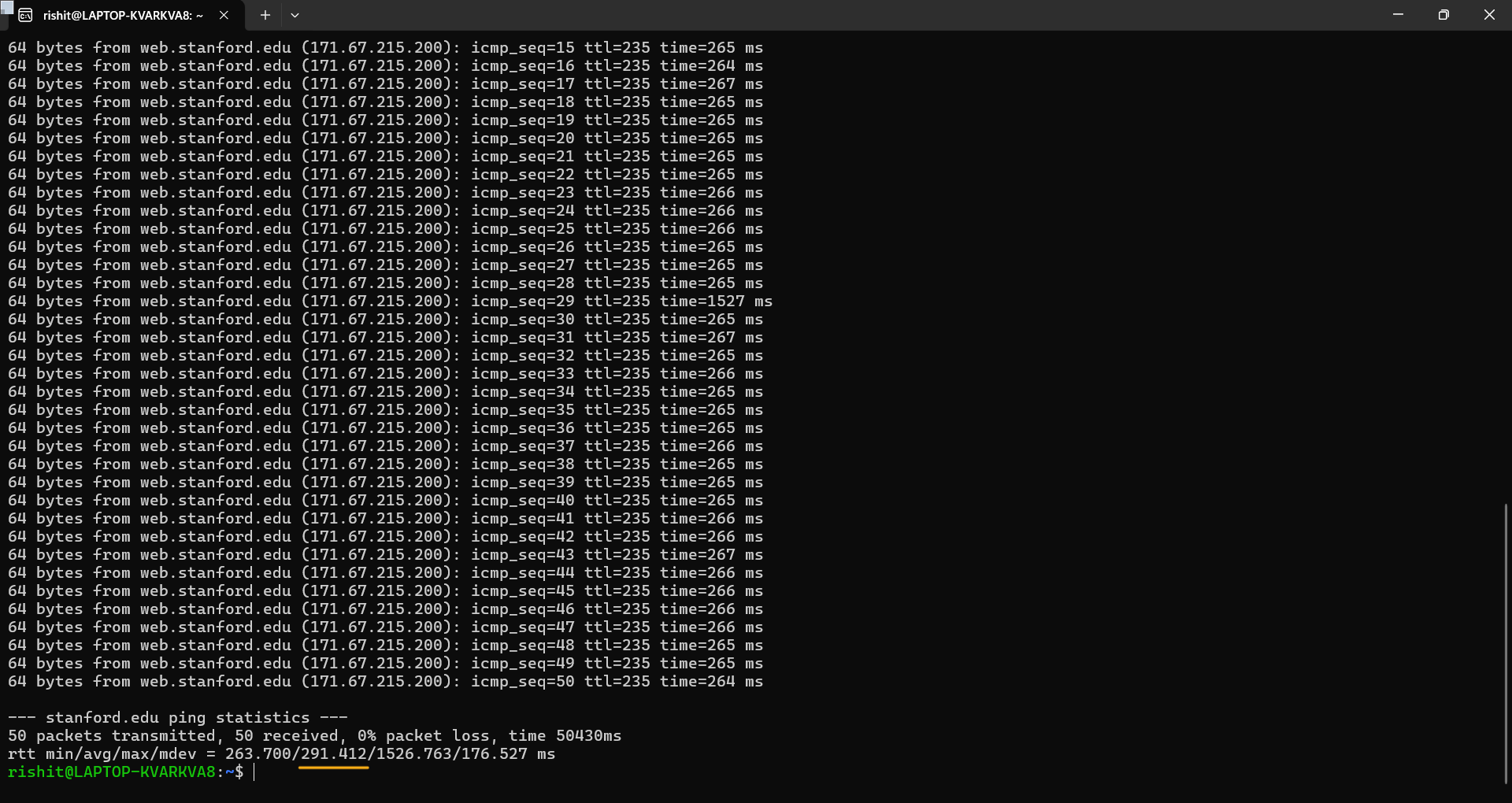
**e)** Using **traceroute**, there are multiple entries on a single line (hops) like the following:



This is because there may be more than 2 or 3 paths/routes to reach your destination starting from that router.

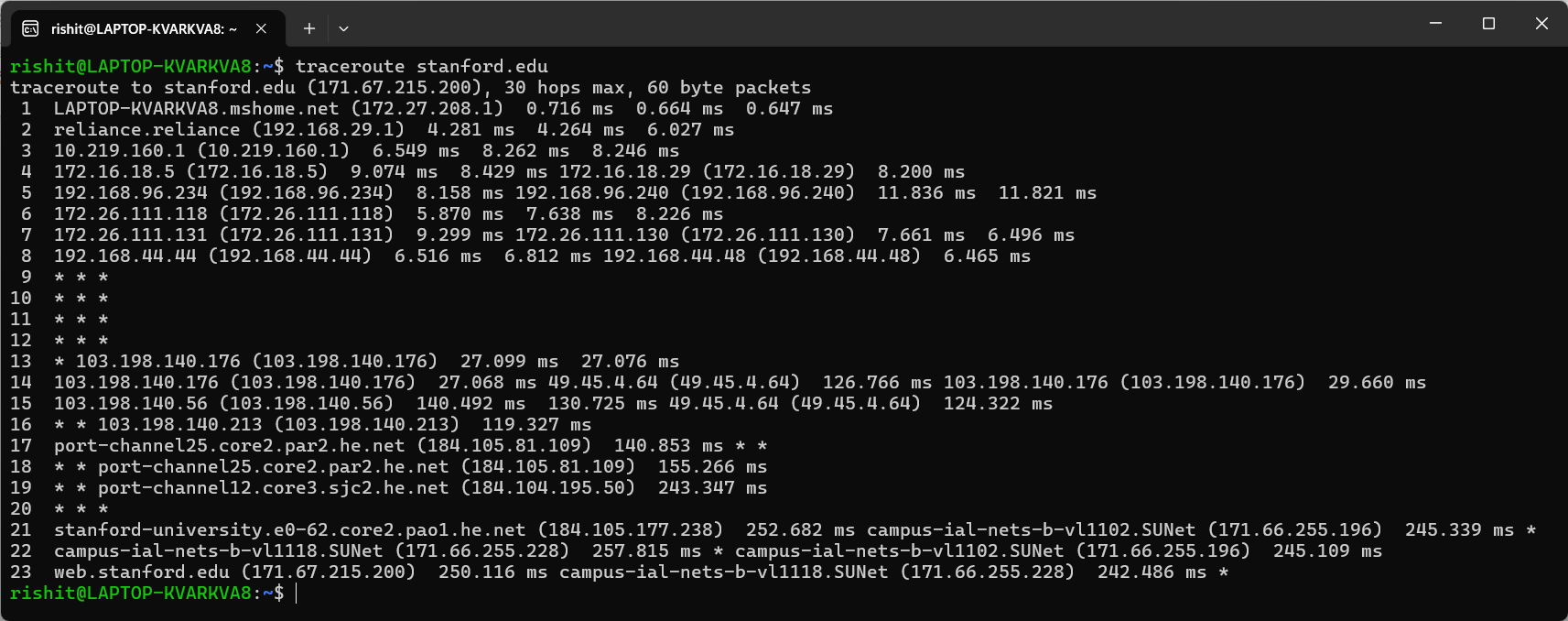
**traceroute** may send packets along different paths, and each path may encounter different network devices. As a result, you see multiple IP addresses listed on the same line, representing the various routes that the packets took.

**f)** 



Average latency using **ping** by sending 50 packets to **stanford.edu**: **291.412 ms**

**g)** Running **traceroute stanford.edu:**

****

**NOTE:** I’m not considering those hops that contain even a single star (\*).

Total no. of hops: **10**

Using **traceroute google.in**, no. of hops: **14**

**h)** Average latency using **ping -c 50 google.in: 12.674 ms**

Average latency **using ping -c 50 stanford.edu: 291.412 ms**

This latency difference is because of several factors like:

1. **Distance: google.in** routes you to a server location in India itself, resulting in a lower latency.

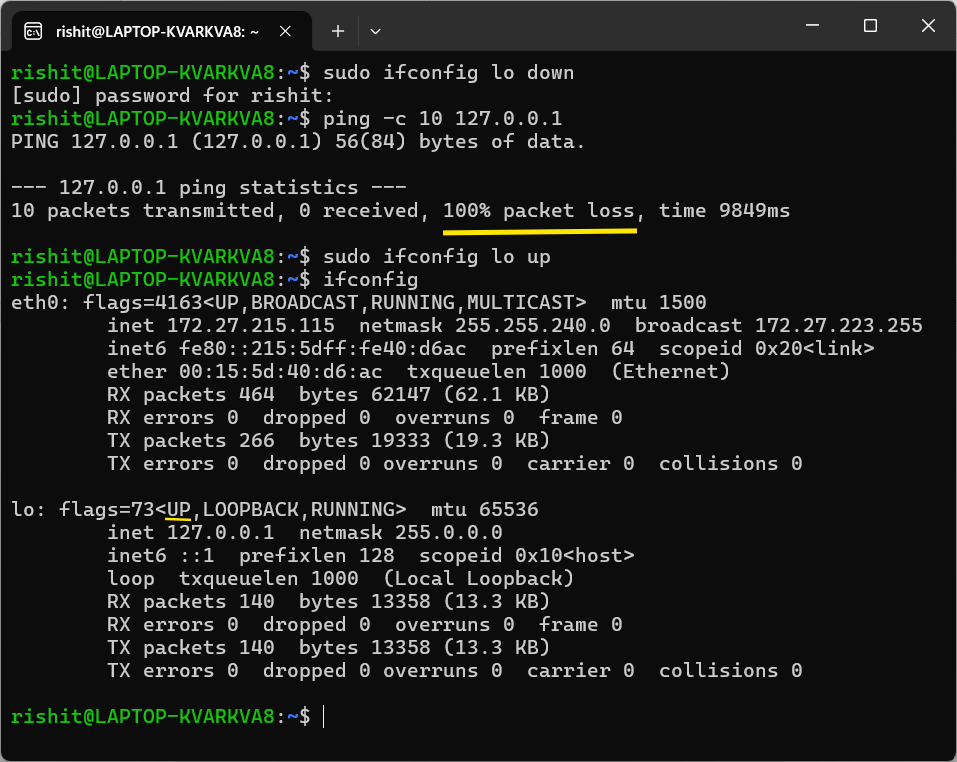
**stanford.edu** is hosted in US, which is much farther away. The physical distance leads to higher latency due to longer time it takes for packets to travel across international boundaries and through multiple network hops.

1. **Network Routing:** Google has an extensive global infrastructure with data centers in many regions worldwide. This allows traffic to be routed efficiently, often reducing latency.

**stanford.edu** is hosted on an academic network, which might have fewer global distribution points, and the routing might not be as optimized for global reach as Google's infrastructure, leading to longer paths and higher latency.

1. **Network Backbone:** The network backbone your location and Google's servers is likely more efficient due to Google's investments in global infrastructure.

The backbone to **stanford.edu** might involve more public internet routes, which can introduce additional delays.

**Ans 6.** 

First, we disable the network interface temporarily using:

**sudo ifconfig lo down**

This will disable the loopback network interface on your computer.

The loopback network interface is a virtual network interface that your computer uses to communicate with itself. It is assigned the IP address **127.0.0.1**, commonly known as **localhost**.

The loopback interface is always active by default on most OSs and is essential for the proper functioning of network-related services.

When we disable the loopback interface, we effectively stop the computer from being able to communicate with itself via **127.0.0.1.**

Then, we send 10 packets of data to the localhost using:

**ping -c 10 127.0.0.1**

After almost 10 seconds, we receive the message that there has been **100% packet loss**.

To enable the loopback interface again, use:

**sudo ifconfig lo up**

**GitHub link:** <https://github.com/RishitIIITD/CN_assignments/tree/main/assignment-1>