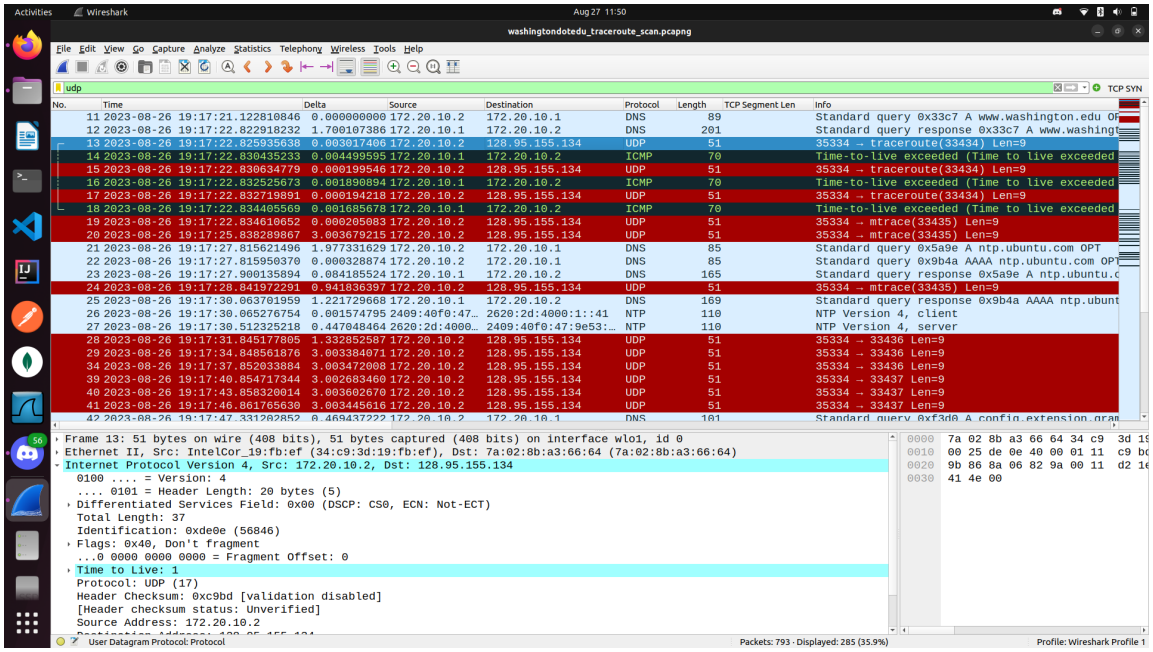
Lab-2 Assignment Submission

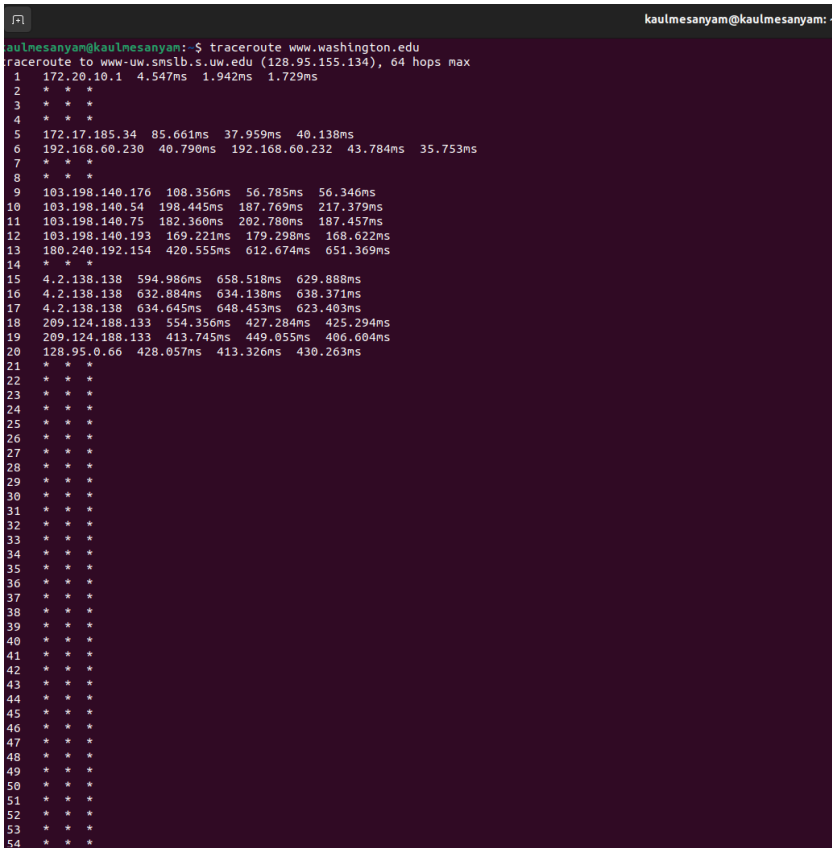
-Sanyam Kaul
-CS23MTECH14011

Q1.

1. The protocol of probe packets is UDP. The terminal snapshot below shows the traceroute output ran for washington.edu and the wireshark snapshot shows the UDP probe packets sent during the traceroute scan. Note that packet number 13 is the very first probe packet and that is the reason its TTL is 1. Packets 13, 15 and 17 each have a TTL of 1 as they are the probe packets of the first hop. Packets 19, 20 and 24 are the probe packets of 2nd hop and have the TTL of 2. The subsequent 3 probe packets have a TTL of 3 and it increases like this as we continue.

   Note that in flags the Fragment offset is set to 0 and don't fragment bit is set. This indicates the routers in the network not to fragment these packets. We can also get some other fields like source and destination IPs in this packet as well



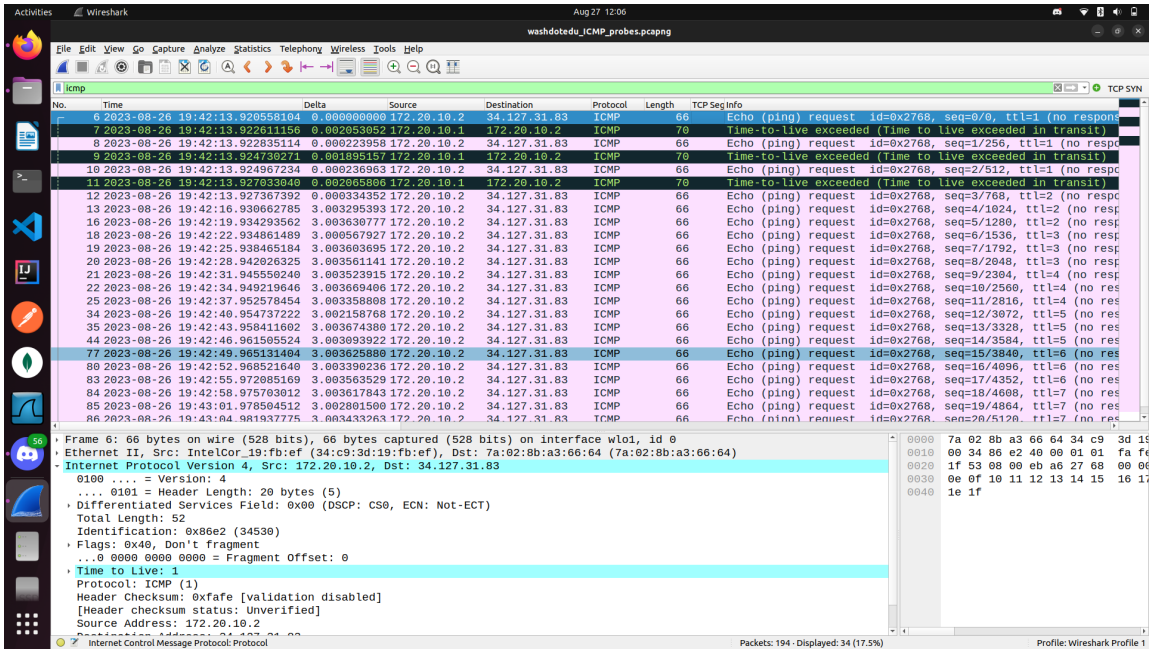


Note that we got info in the first hop and that is why we can also see the ICMP response packets (numbers 14, 16, and 18) for the first 3 probe packets (see in the snapshot above). We got * * * in 2nd hop and as we can see in the snapshot, there are no ICMP response packets for the probe packets for 2nd. Note that the info of the ICMP response packets says that TIme-To-Live has exceeded.

2. Yes, we can change the protocol of probe packets. We can use ICMP or the TCP SYN protocol instead of UDP. The snapshot below shows the wireshark scan of running the traceroute command with -I which forces the use of ICPM for probe packets. The terminal snapshot is also below: -

```
kaulmesanyam@kaulmesanyam:~$ traceroute --help
Usage: traceroute [OPTION...] HOST
Print the route packets trace to network host.

  -f, --first-hop=NUM         set initial hop distance, i.e., time-to-live
  -g, --gateways=GATES        list of gateways for loose source routing
  -I, --icmp                  use ICMP ECHO as probe
  -m, --max-hop=NUM           set maximal hop count (default: 64)
  -M, --type=METHOD           use METHOD (`icmp' or `udp') for traceroute
                              operations, defaulting to `udp'
  -p, --port=PORT             use destination PORT port (default: 33434)
  -q, --tries=NUM             send NUM probe packets per hop (default: 3)
      --resolve-hostnames     resolve hostnames
  -t, --tos=NUM               set type of service (TOS) to NUM
  -w, --wait=NUM              wait NUM seconds for response (default: 3)
  -?, --help                  give this help list
      --usage                 give a short usage message
  -V, --version               print program version

Mandatory or optional arguments to long options are also mandatory or optional
for any corresponding short options.

Report bugs to <bug-inetutils@gnu.org>.
kaulmesanyam@kaulmesanyam:~$ traceroute -I www.washington.edu
traceroute to www-uw.smslb.s.uw.edu (34.127.31.83), 64 hops max
  1   172.20.10.1  2.132ms  1.992ms  2.239ms
  2   *  *  *
  3   *  *  *
  4   *  *  *
  5   *  *  *
  6   *  *  *
  7   *  *  *
  8   *  *  *
  9   *  *  *
 10   *  *  *
^C
kaulmesanyam@kaulmesanyam:~$
```

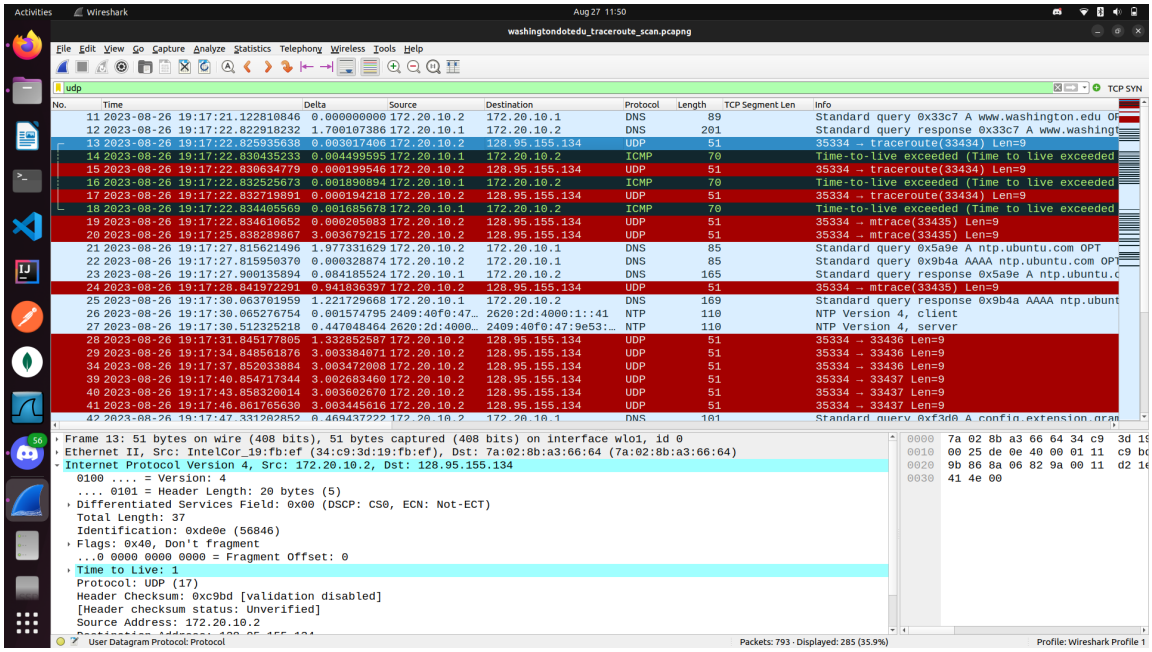
Here, I have executed - the ‘traceroute -I www.washington.edu’ command. The -I in the command forces the use of ICMP protocol for probe packets instead of UDP.



This is the Wireshark scan of the command run above. Note that packets 6, 8 and 10 are the probe packets of the very first hop and they are in ICMP protocol. The TTL is 1 which

corresponds to the TTL needed in the first hop. From the Info column, we can see that each packet is an Echo (ping) request but since the TTL is 1, it can go to 1 hop only. The TTL will expire there and a response ICMP will be sent which are packets 7, 9 and 11 in the snapshot.

3. Taking this scan: -

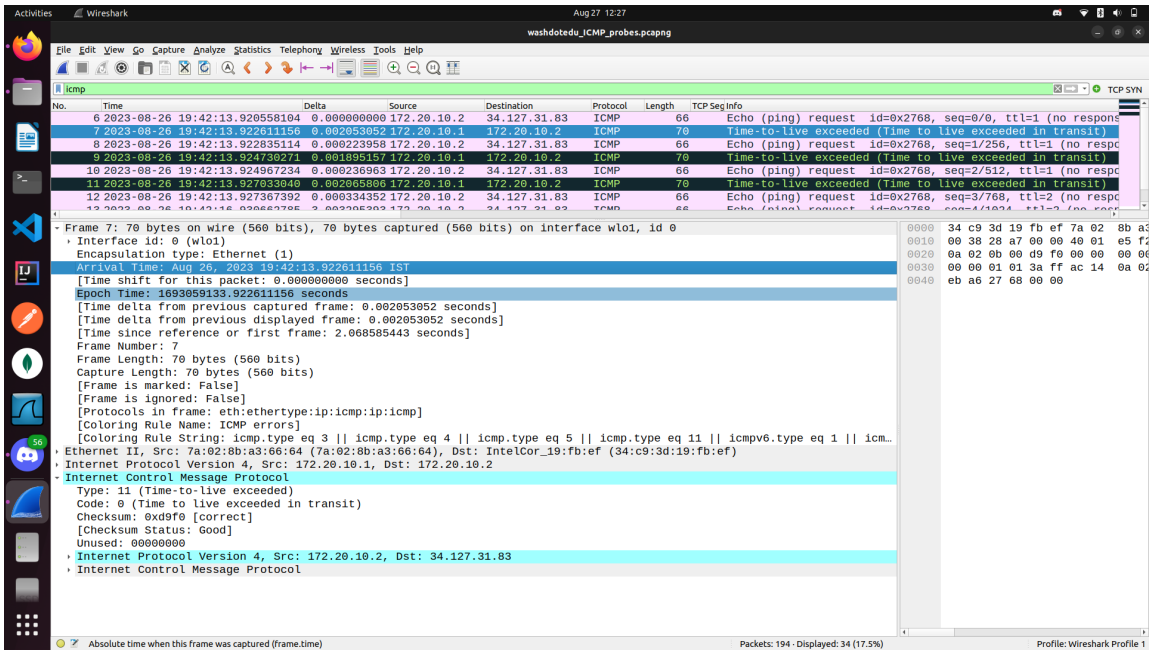


Take packets 13 and 15 which are the probe packets of 1st hop - The delay between them is about 0.005 sec and if we take packets 15 and 17 the delay is 0.002 sec. Also, note that for these packets we have received the ICMP response packets.

However, if we take packets 19 and 20 which are of 2nd hop, the delay between these is 3 sec. In fact between all the probe packets for which we have not received the ICMP response is 3 sec.
So it looks like it waits for a max of 3 sec for the response packet. If the response is received before that we see the RTT value on the terminal and the subsequent probe is sent instantly. If the response is not received in 3 sec, we see a * printed on the terminal and the subsequent probe is sent

4. We get the info regarding the arrival time of the response packet which tells us the info about the RTT. Also, we get ICMP Type - 11 and code-0 which tells us that the TTL of the probe packets expired and that is why this ICMP response is sent. The snapshot below shows the packet details of packet 7 which is an ICMP response packet for probe

packet 6

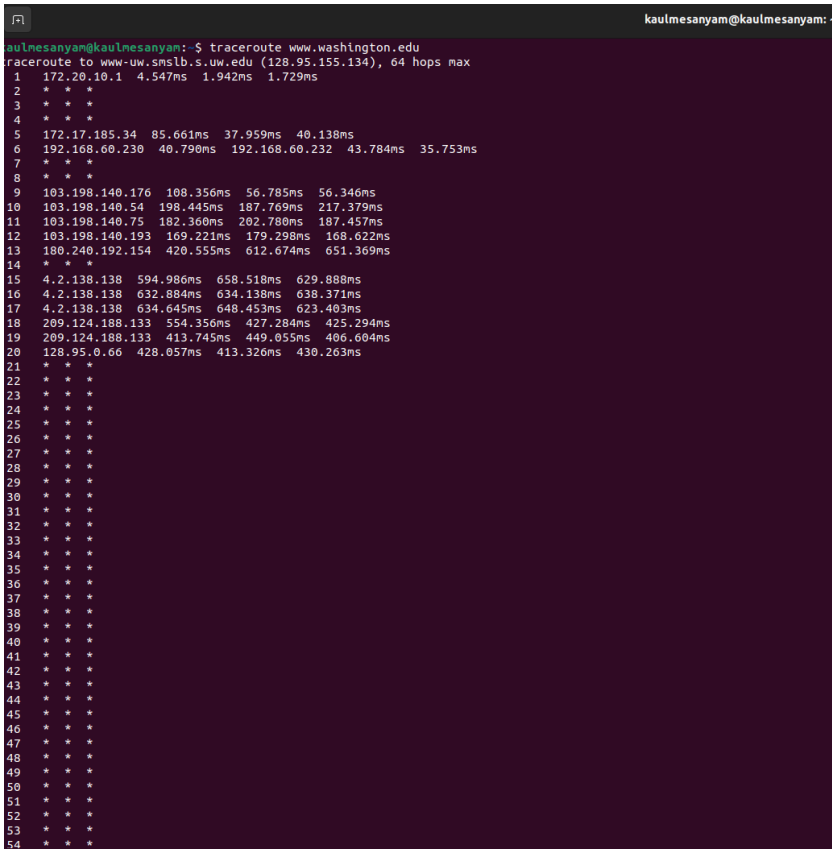


5. I saw that both the UDP and ICMP probe packets have TTL fields. But the TTL value was increasing incrementally. For the very first hop, the TTL is set to 1. Once it reaches the very first router, the TTL expires and the response ICMP packet is generated. Then the TTL is set to 2 in the probe packets for the second hop. So it completes 2 hops only after which the TTL expires and a response is generated. The response tells us about the RTT of hops in a step-by-step manner.

   In the case of response packets, I saw that the protocol was ICMP and the TTL was set to 64 in all the response packets

6. I tried multiple times but the traceroute for washington.edu did not get fully complete. Talking about the bottleneck, in the snapshot below, we can see that the packets got dropped after row 20 (router IP - 128.95.0.66). Also, looking at rows 15 to 19, the RTTs for these hops range from about 600 to 670 ms which is considerably more than a

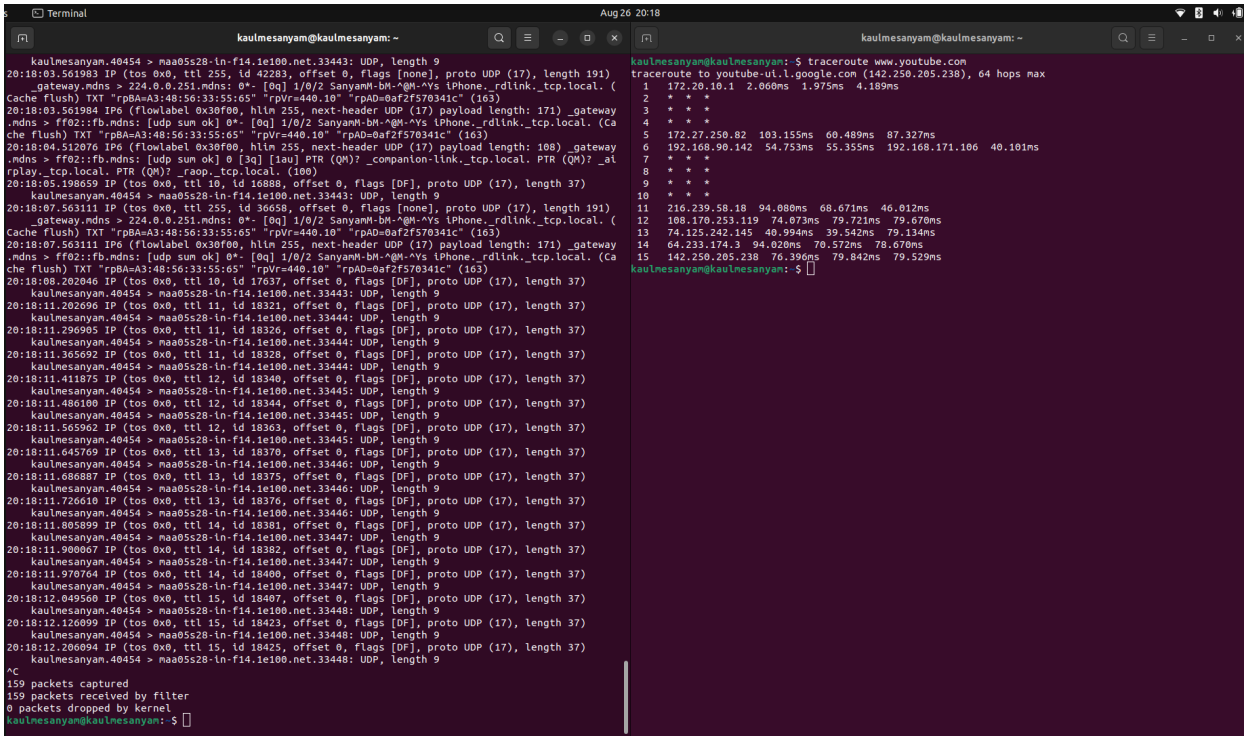
normal scenario



7. Talking about washington.edu traceroute - The consistent display of stars after row 20 could mean that these probe packets are being droppedby the next router after this point. This could be by intension or by some network issue. I tried opening the website and it got opened without any issue so it seems like the router has been programmed to drop the probe packets. One additional reason could be that the router is dropping packets due to traffic overload.

Q2.

Performed traceroute for youtube.com and captured packets using traceroute: -



1. Looking at the last probe packet with TTL 10 and the first probe packet with TTL 11 - from the terminal we can see that we have * for the last probe packet with TTL 10. Also, note that the delay between these packets is about 2.37 sec. This is the case where no response has been captured. In the case of Wireshark, the delay in case of no response was about 3 sec. Taking any 2 probe packets with TTL 11 (we have received a response for these) we can see that the delay is in the range of 0.005 sec to 0.025
So the overall idea remains the same. It is waiting for a certain amount of time for the response packet before sending the next probe packet. And the threshold of the waiting time seems to be 2.5 to 3 sec

2. The TTL values are the same as seen in Wireshark. For probe packets, the TTL values start from 1 and get incremented by 1 for each subsequent hop. In the case of response packets, the TTL is 64

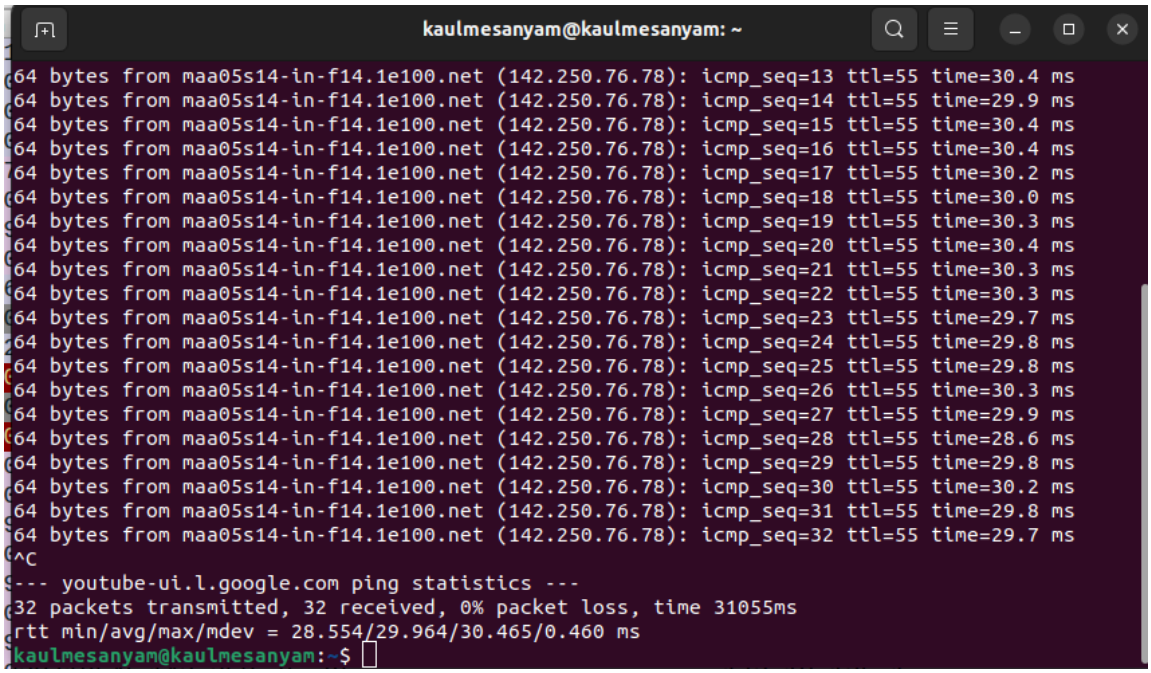
3. It took about 72 seconds for the traceroute scan to complete. Talking about the bottleneck, the router on hop 5 took about 87 ms to 104 ms. The router on hop 14 took about 94 ms for one of the packets. All the subsequent packets after the first hop took about 5ms to 70ms as compared to the first hop where the delays were from 2 ms to 4 ms. values can be seen in the snapshot above.

Q3.

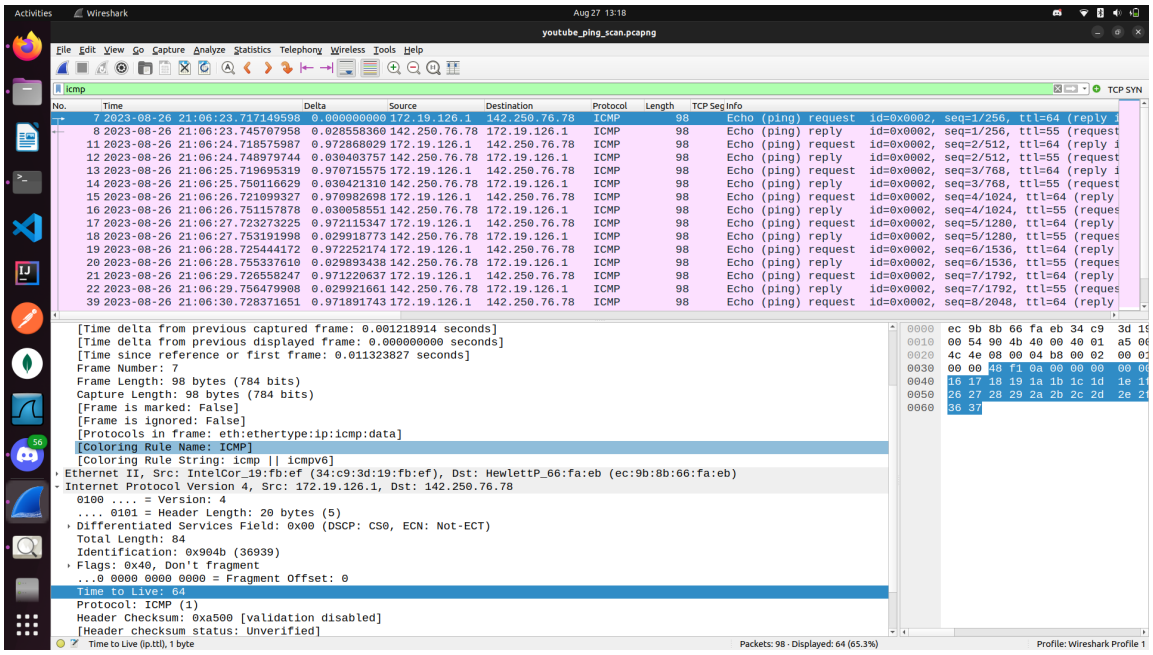
1. **Playing with Ping: -**

   Snapshots below show the ping command executed in terminal and the wireshark and tcpdump packet captures: -

   Tcpdump capture: -



   Wireshark capture: -



Command execution on terminal: -

```
kaulmesanyam@kaulmesanyam:~$ ping www.youtube.com
PING youtube-ui.l.google.com (142.250.183.238) 56(84) bytes of data.
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=1 ttl=55 time=33.3 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=2 ttl=55 time=34.8 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=3 ttl=55 time=34.8 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=4 ttl=55 time=35.3 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=5 ttl=55 time=34.6 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=6 ttl=55 time=38.6 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=7 ttl=55 time=34.1 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=8 ttl=55 time=34.5 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=9 ttl=55 time=34.4 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=10 ttl=55 time=34.5 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=11 ttl=55 time=34.7 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=12 ttl=55 time=98.2 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=13 ttl=55 time=34.6 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=14 ttl=55 time=33.6 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=15 ttl=55 time=32.9 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=16 ttl=55 time=33.8 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=17 ttl=55 time=35.1 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=18 ttl=55 time=34.8 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=19 ttl=55 time=34.2 ms
64 bytes from maa05s23-in-f14.1e100.net (142.250.183.238): icmp_seq=20 ttl=55 time=37.8 ms
^C
--- youtube-ui.l.google.com ping statistics ---
20 packets transmitted, 20 received, 0% packet loss, time 19032ms
rtt min/avg/max/mdev = 32.904/37.935/98.219/13.890 ms
kaulmesanyam@kaulmesanyam:~$
```

Observations: -

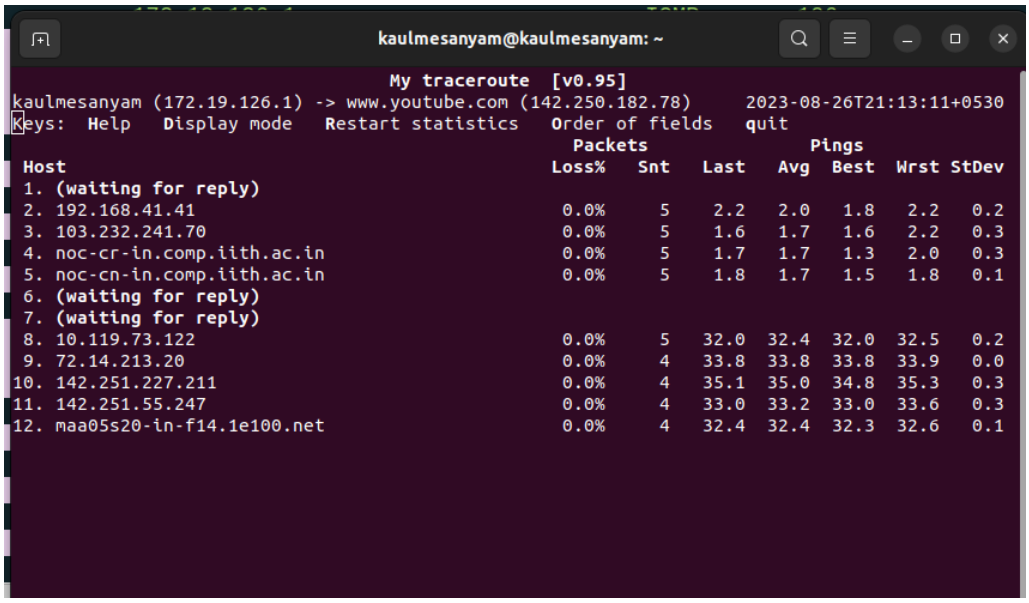
A. The ping command sends ICMP Echo probe packets but unlike traceroute the TTL for all the packets is set to 64. Hence, in ping, each ICMP probe packet can go up to 64 hops.

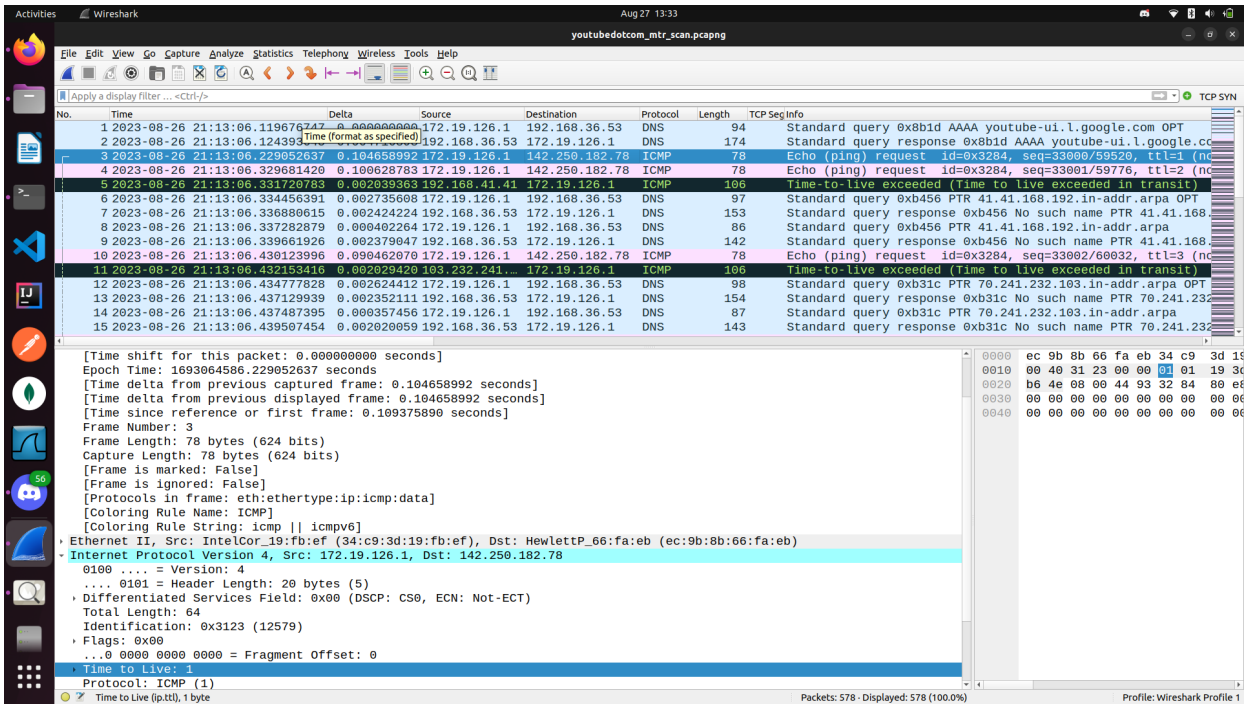
B. If the destination is within 64 hops then we will get the response ICMP packet which will tell us the RTT info and the delay is displayed on the terminal.

C. The probe packets are sent every second and the sequence number is incremented by 1 every time. Also, the TTL in the response packet is 55.

Hence, ping tells us the RTT from source to destination every second.

2. **Playing with mtr: -**

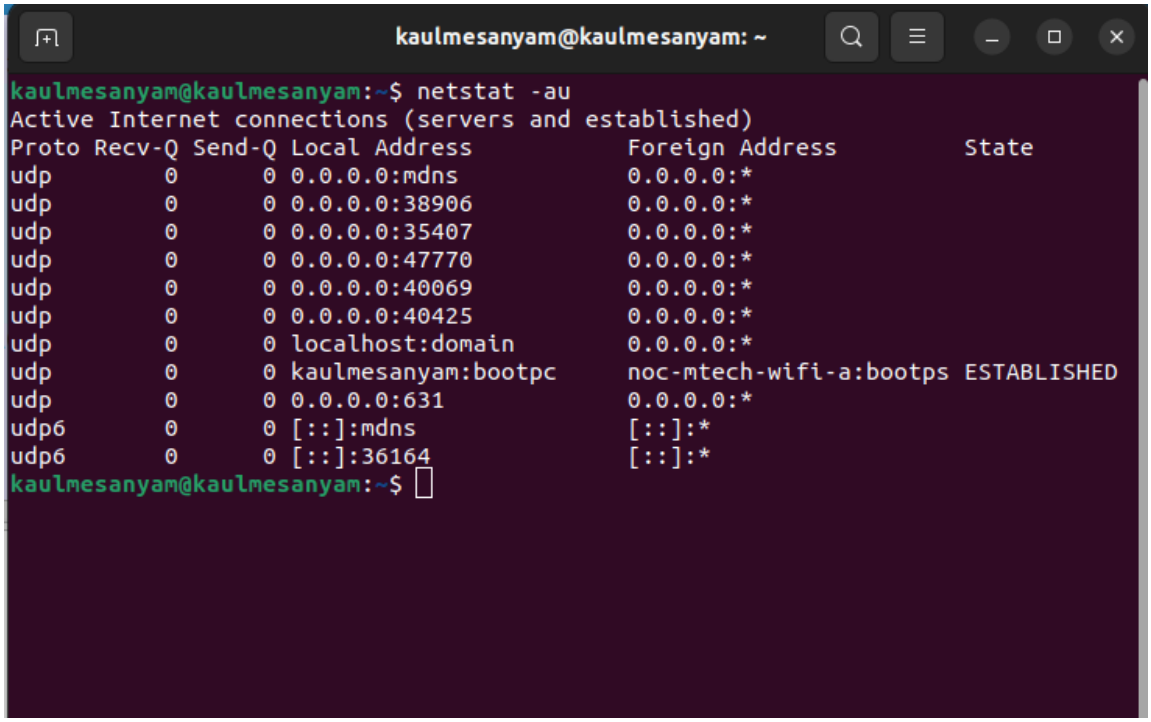




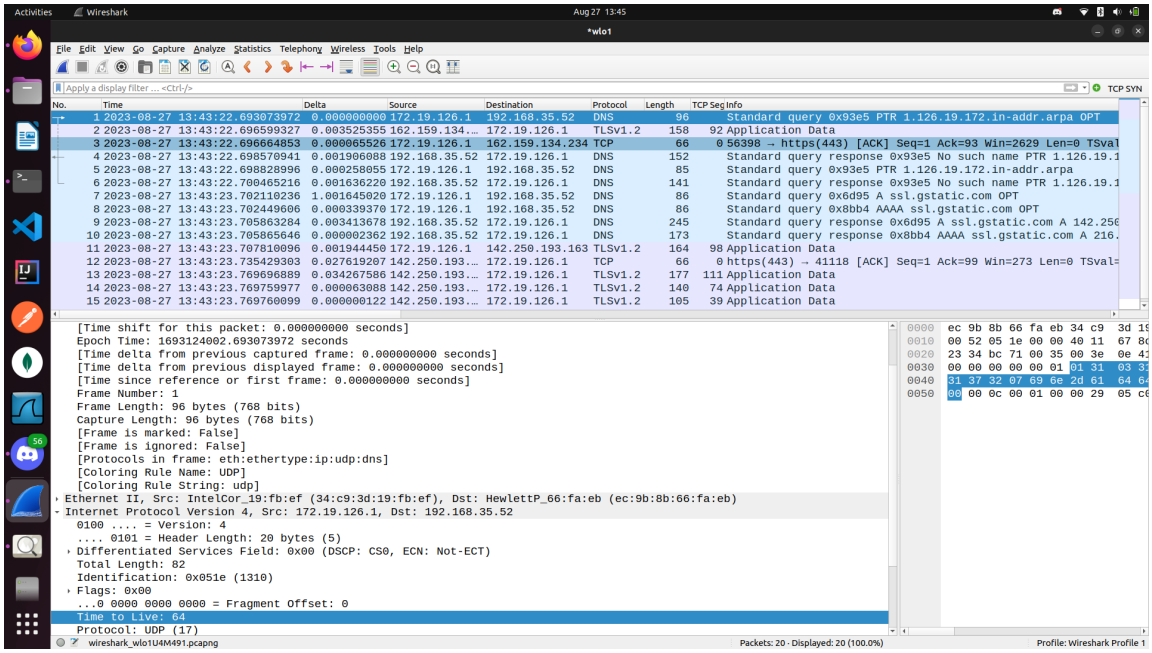
This is mostly to traceroute. It is sending ICMP probe packets and getting ICMP responses. Like traceroute the TTL starts from 1 and gets incremented by 1 for every subsequent hop. The only difference is that traceroute sends 3 probes per hop but the in case of mtr there is 1 probe packer per hop. Like in the snapshot, packet 3 is the first probe packet and has a TTL of 1. Packet 4 is the next probe packet and has a TTL of 2. In the case of traceroute, if would have been 1 only.

3. **Playing with netstat: -**

   Here, I am running the netstat command with -au which is giving me all the active UDP connections from my system: -



Wireshark scan: -

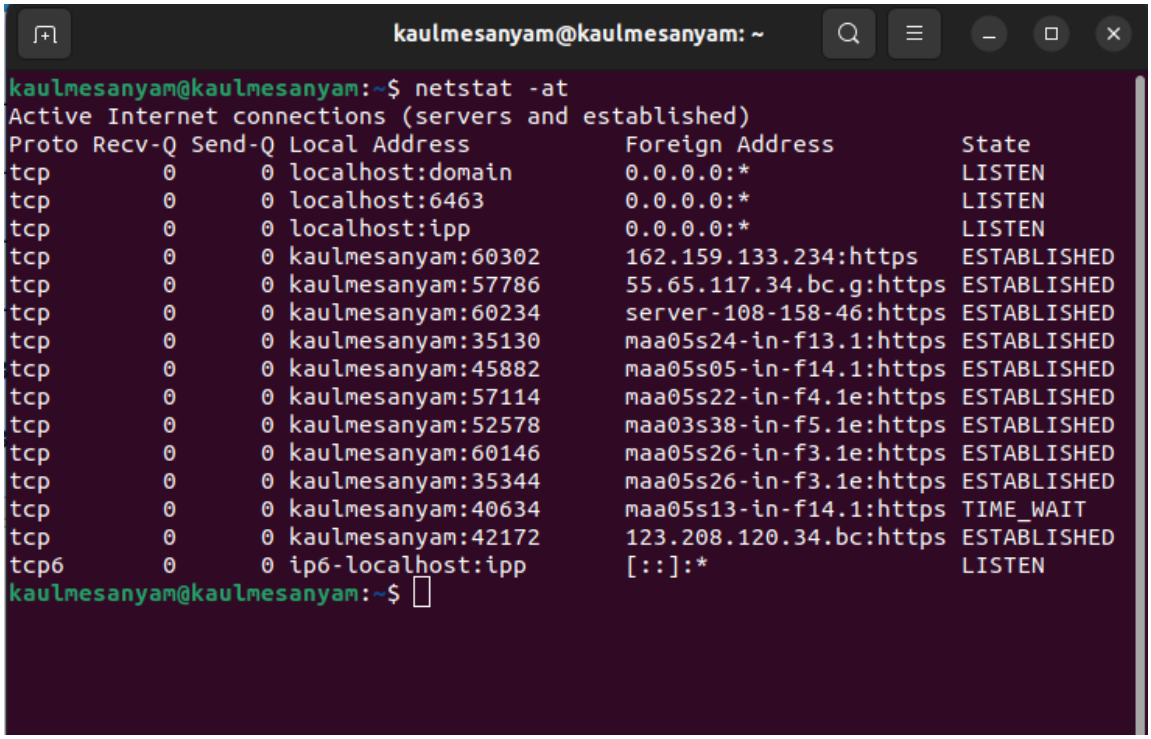


Note that when this command is executed, a DNS request is sent from my system to destination IP - 192.168.35.52
I ran nslookup on this IP and found that this IP maps to the dns1.iith.ac.in the domain which looks like a DNS server

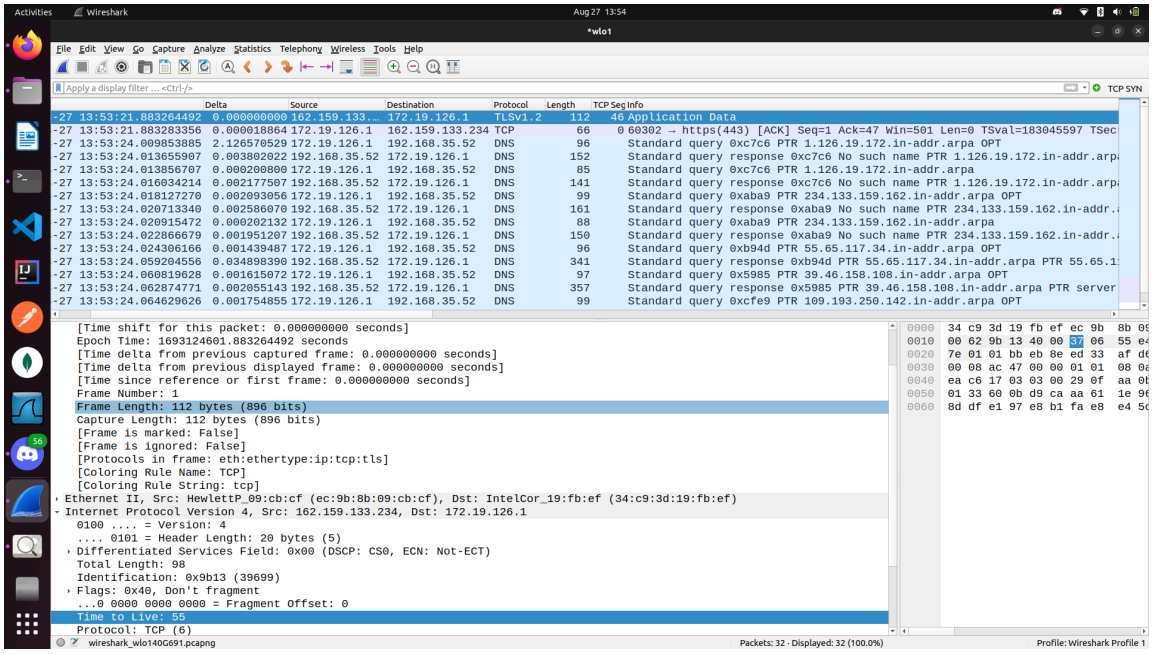
```
kaulmesanyam@kaulmesanyam:~$ nslookup 192.168.35.52
52.35.168.192.in-addr.arpa      name = monitor.iith.ac.in.
52.35.168.192.in-addr.arpa      name = dns1.iith.ac.in.
52.35.168.192.in-addr.arpa      name = home.iith.ac.in.

Authoritative answers can be found from:
168.192.in-addr.arpa    nameserver = dns2.iith.ac.in.
168.192.in-addr.arpa    nameserver = dns1.iith.ac.in.
dns2.iith.ac.in internet address = 192.168.36.53
dns1.iith.ac.in internet address = 192.168.35.52
```

The subsequent requests from my system are DNS queries to this destination IP only and I am getting DNS responses as well. Looks like when I am running the command the DNS server of the network I am connected to (IITH network) in this case is being queried for all the open UDP connections. That's how I am getting the result on my terminal.

I ran the 'netstat -at' command as well which gives me all the open TCP connections from my system: -



Wireshark capture: -



Again, there is DNS request and responses from my system and the IITH DNS server. The DNS server is being queried for all the open TCP connections this time.