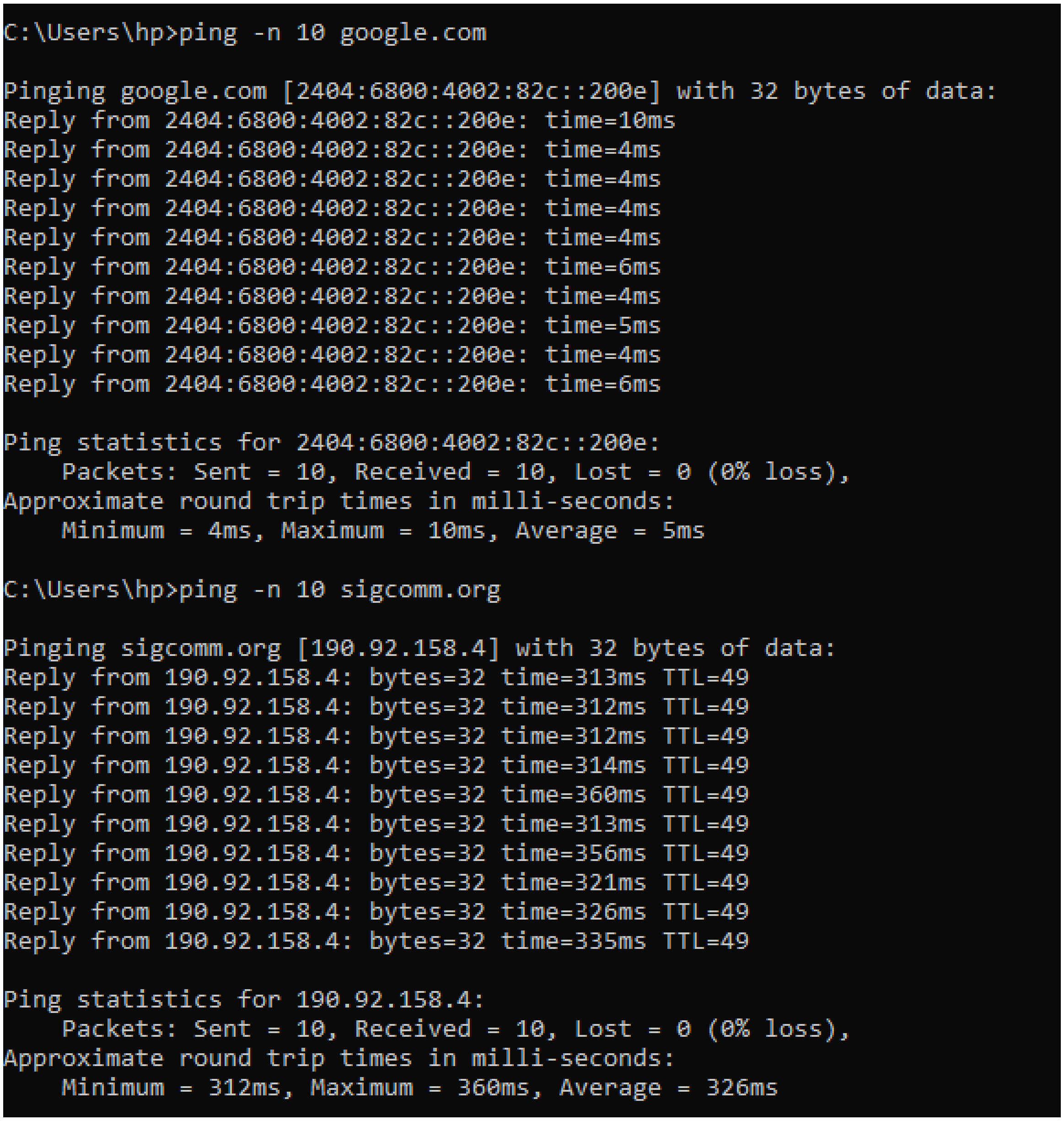
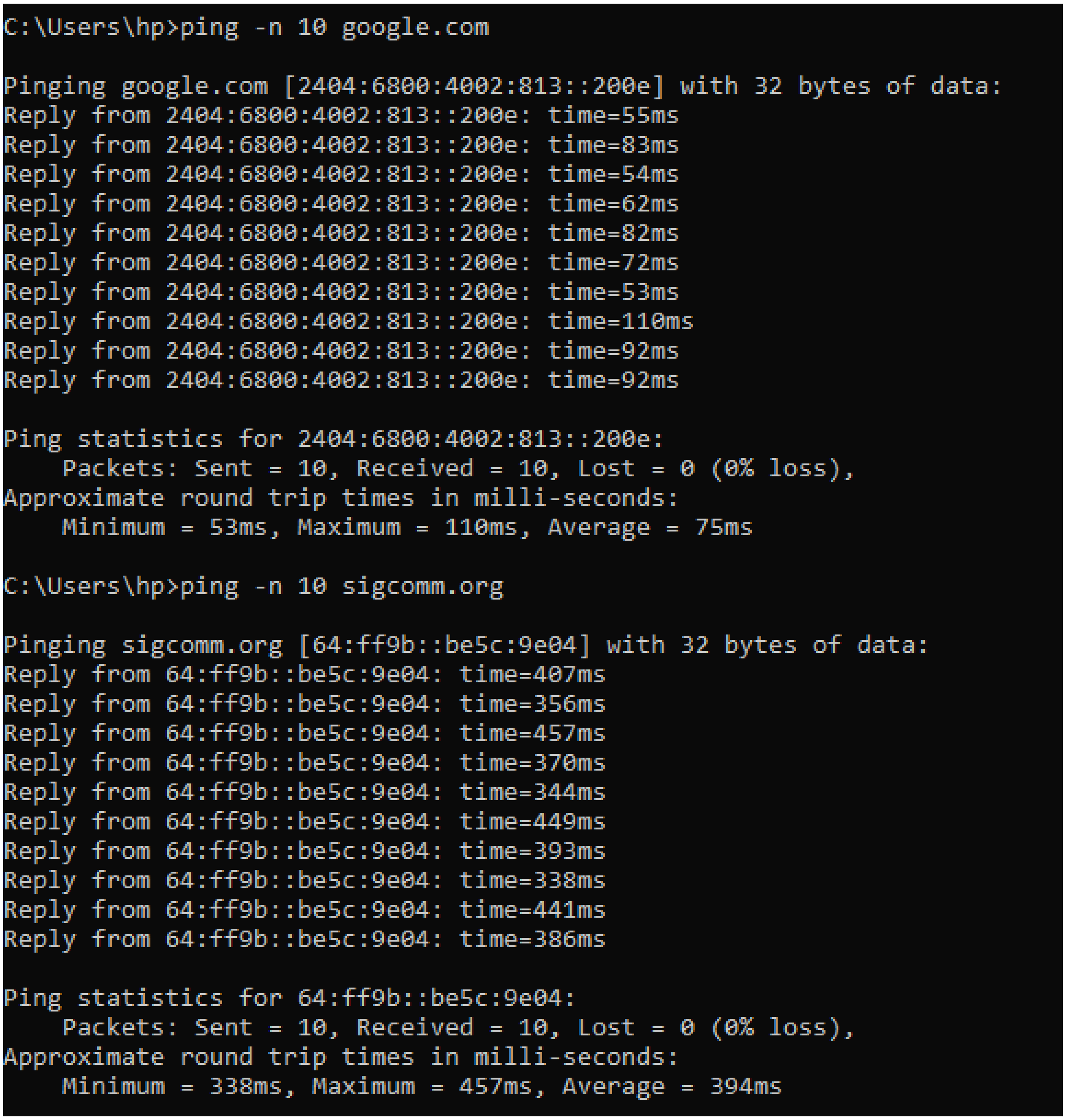
Question-1.1:

Google.com and sigcomm.org using IITD network:



```
C:\Users\hp>ping -n 10 google.com

Pinging google.com [2404:6800:4002:82c::200e] with 32 bytes of data:
Reply from 2404:6800:4002:82c::200e: time=10ms
Reply from 2404:6800:4002:82c::200e: time=4ms
Reply from 2404:6800:4002:82c::200e: time=4ms
Reply from 2404:6800:4002:82c::200e: time=4ms
Reply from 2404:6800:4002:82c::200e: time=4ms
Reply from 2404:6800:4002:82c::200e: time=6ms
Reply from 2404:6800:4002:82c::200e: time=4ms
Reply from 2404:6800:4002:82c::200e: time=5ms
Reply from 2404:6800:4002:82c::200e: time=4ms
Reply from 2404:6800:4002:82c::200e: time=6ms

Ping statistics for 2404:6800:4002:82c::200e:
    Packets: Sent = 10, Received = 10, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 10ms, Average = 5ms

C:\Users\hp>ping -n 10 sigcomm.org

Pinging sigcomm.org [190.92.158.4] with 32 bytes of data:
Reply from 190.92.158.4: bytes=32 time=313ms TTL=49
Reply from 190.92.158.4: bytes=32 time=312ms TTL=49
Reply from 190.92.158.4: bytes=32 time=312ms TTL=49
Reply from 190.92.158.4: bytes=32 time=314ms TTL=49
Reply from 190.92.158.4: bytes=32 time=360ms TTL=49
Reply from 190.92.158.4: bytes=32 time=313ms TTL=49
Reply from 190.92.158.4: bytes=32 time=356ms TTL=49
Reply from 190.92.158.4: bytes=32 time=321ms TTL=49
Reply from 190.92.158.4: bytes=32 time=326ms TTL=49
Reply from 190.92.158.4: bytes=32 time=335ms TTL=49

Ping statistics for 190.92.158.4:
    Packets: Sent = 10, Received = 10, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 312ms, Maximum = 360ms, Average = 326ms
```

Google.com and Sigcomm.org using mobile network:



```
C:\Users\hp>ping -n 10 google.com

Pinging google.com [2404:6800:4002:813::200e] with 32 bytes of data:
Reply from 2404:6800:4002:813::200e: time=55ms
Reply from 2404:6800:4002:813::200e: time=83ms
Reply from 2404:6800:4002:813::200e: time=54ms
Reply from 2404:6800:4002:813::200e: time=62ms
Reply from 2404:6800:4002:813::200e: time=82ms
Reply from 2404:6800:4002:813::200e: time=72ms
Reply from 2404:6800:4002:813::200e: time=53ms
Reply from 2404:6800:4002:813::200e: time=110ms
Reply from 2404:6800:4002:813::200e: time=92ms
Reply from 2404:6800:4002:813::200e: time=92ms

Ping statistics for 2404:6800:4002:813::200e:
    Packets: Sent = 10, Received = 10, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 53ms, Maximum = 110ms, Average = 75ms

C:\Users\hp>ping -n 10 sigcomm.org

Pinging sigcomm.org [64:ff9b::be5c:9e04] with 32 bytes of data:
Reply from 64:ff9b::be5c:9e04: time=407ms
Reply from 64:ff9b::be5c:9e04: time=356ms
Reply from 64:ff9b::be5c:9e04: time=457ms
Reply from 64:ff9b::be5c:9e04: time=370ms
Reply from 64:ff9b::be5c:9e04: time=344ms
Reply from 64:ff9b::be5c:9e04: time=449ms
Reply from 64:ff9b::be5c:9e04: time=393ms
Reply from 64:ff9b::be5c:9e04: time=338ms
Reply from 64:ff9b::be5c:9e04: time=441ms
Reply from 64:ff9b::be5c:9e04: time=386ms

Ping statistics for 64:ff9b::be5c:9e04:
    Packets: Sent = 10, Received = 10, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 338ms, Maximum = 457ms, Average = 394ms
```

1.1(A)

average ping latencies for the two websites in the IITD network(IPv4):

google.com - 5 ms

Sigcomm.org - 326 ms

average ping latencies for the two websites in the Mobile network(IPv4):

google.com - 75 ms

sigcomm.org - 394 ms

We observe that the average ping latency for google.com in both the networks is lesser than that of sigcomm.org. **The following maybe the possible reasons:**

1)Server Location or distance from the device:google.com might be closer to this location than sigcomm.org.or the router path to google.com may be simpler and easier than that of sigcomm.org

2)Server Load: High traffic on the servers of sigcomm.org could increase latency.

3) Processing delays: Maybe the processing delays at the routers on the path to sigcomm.org took more time to process the packets or had more queuing delay than that of google.com

average ping latencies for google.com in

- **IITD network:** 5 ms
- **Mobile network:** 75 ms

average ping latencies for sigcomm.org in

- **IITD network:** 326 ms
- **Mobile network:** 394 ms

We observe that ping latencies for both websites is lesser in the IITD network than that of mobile network.The following maybe the possible reasons:

1)Congestion in the network: There maybe higher congestion in the mobile network than that of IITD network
2)Routing in the network: Routing paths in IITD network maybe more efficient and less time taking than that of mobile network
3)Quality of connection and signal strength: Quality of connection and the signal strength of IITD network is much better than that of mobile network

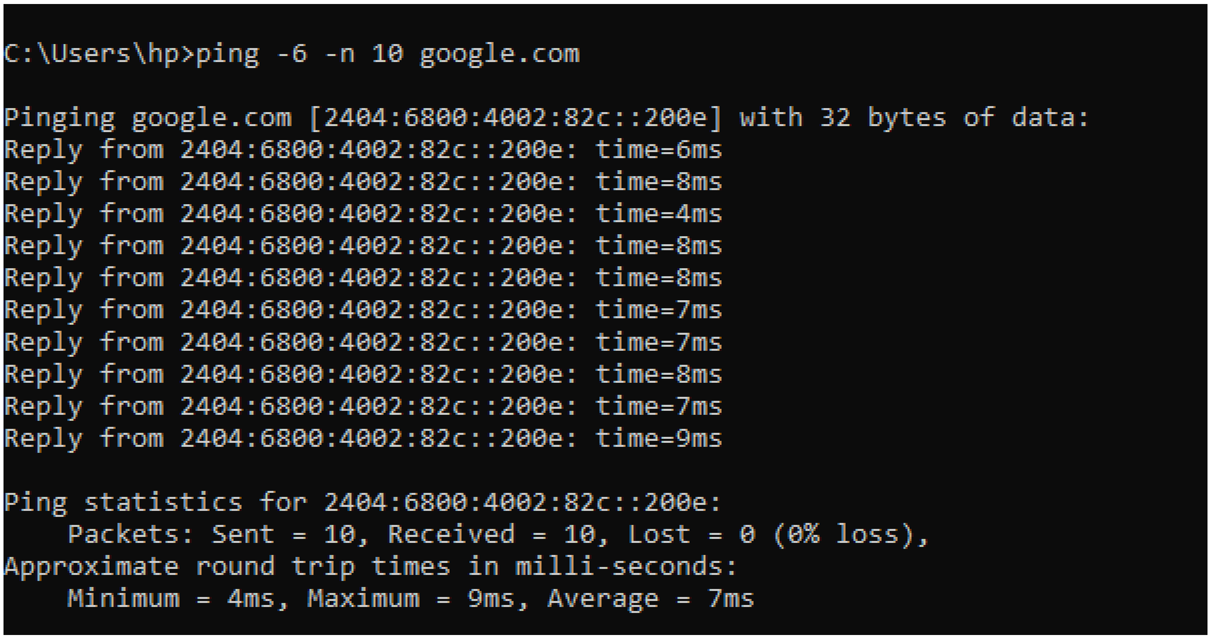
1.1(B)

Ping is a function that is available on any system with network connectivity,Ping is used to check whether a specific network device is reachable from it or not.The Ping utility uses the echo request, and echo reply messages within the **Internet Control Message Protocol (ICMP)**.It is the important part of any IP network, which is used to exchange information and error messages within IPv4 networks. When we ping a specific address, four(by default) echo request packets are sent to the specified address and when the remote host receives each one, It sends back an echo reply packet. There are also other options which can help to adjust default value. Theoretical upper limit of IPv4 packet size for ping is as large as 65,535 bytes including pings. A correctly formed ping packet will be of 56B in size which is 64 B when ICMP header is considered and 84 Bytes when IPv4 header is included.

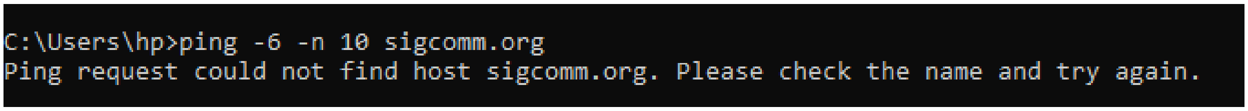
1.1(C)

<u>Forcing both networks to ping using IPv6:</u>

Google.com and Sigcomm.org using IITD network:

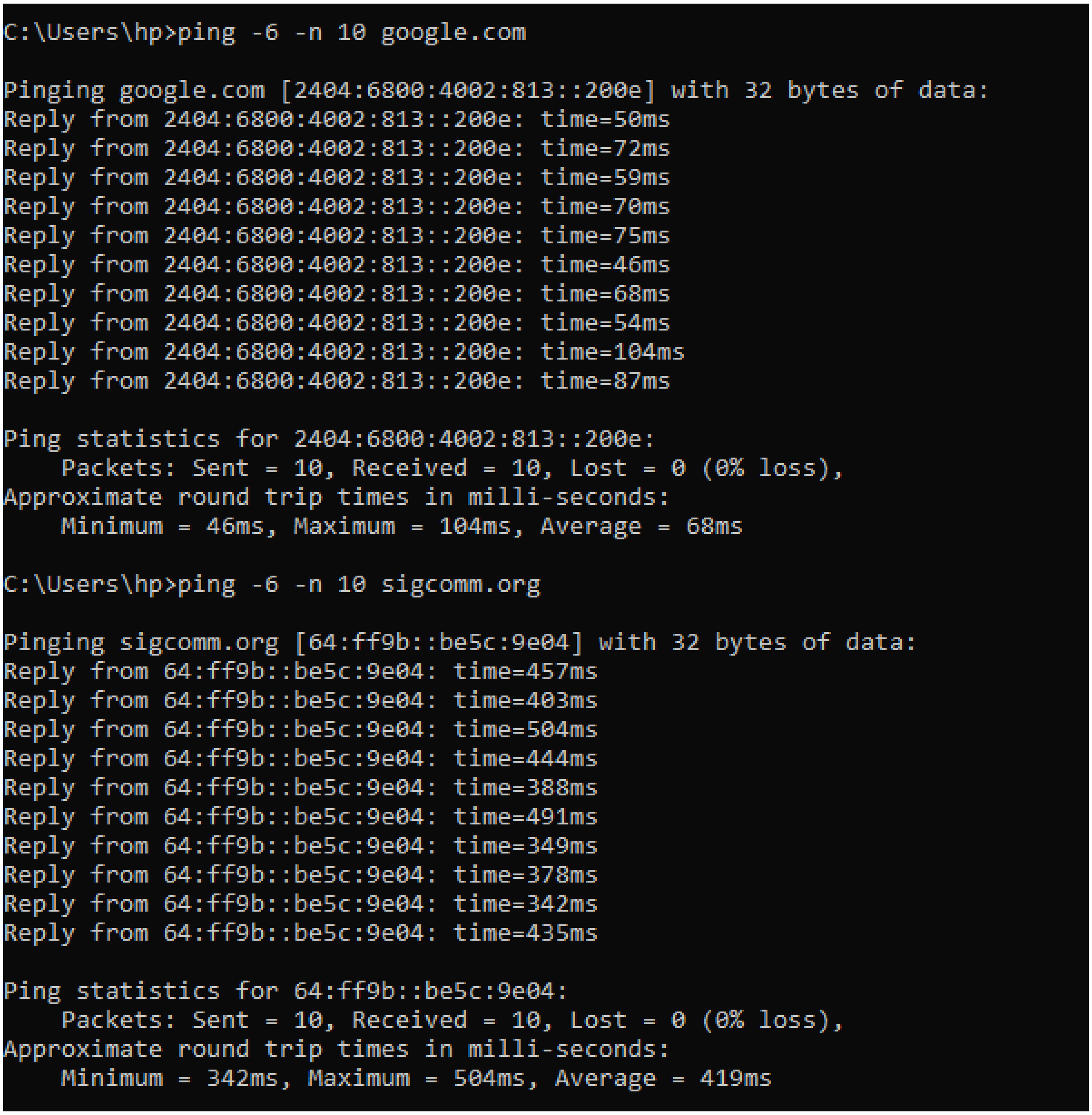


```
C:\Users\hp>ping -6 -n 10 google.com

Pinging google.com [2404:6800:4002:82c::200e] with 32 bytes of data:
Reply from 2404:6800:4002:82c::200e: time=6ms
Reply from 2404:6800:4002:82c::200e: time=8ms
Reply from 2404:6800:4002:82c::200e: time=4ms
Reply from 2404:6800:4002:82c::200e: time=8ms
Reply from 2404:6800:4002:82c::200e: time=8ms
Reply from 2404:6800:4002:82c::200e: time=7ms
Reply from 2404:6800:4002:82c::200e: time=7ms
Reply from 2404:6800:4002:82c::200e: time=8ms
Reply from 2404:6800:4002:82c::200e: time=7ms
Reply from 2404:6800:4002:82c::200e: time=9ms

Ping statistics for 2404:6800:4002:82c::200e:
    Packets: Sent = 10, Received = 10, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 9ms, Average = 7ms
```



```
C:\Users\hp>ping -6 -n 10 sigcomm.org
Ping request could not find host sigcomm.org. Please check the name and try again.
```

Google.com and Sigcomm.org using mobile network:



```
C:\Users\hp>ping -6 -n 10 google.com

Pinging google.com [2404:6800:4002:813::200e] with 32 bytes of data:
Reply from 2404:6800:4002:813::200e: time=50ms
Reply from 2404:6800:4002:813::200e: time=72ms
Reply from 2404:6800:4002:813::200e: time=59ms
Reply from 2404:6800:4002:813::200e: time=70ms
Reply from 2404:6800:4002:813::200e: time=75ms
Reply from 2404:6800:4002:813::200e: time=46ms
Reply from 2404:6800:4002:813::200e: time=68ms
Reply from 2404:6800:4002:813::200e: time=54ms
Reply from 2404:6800:4002:813::200e: time=104ms
Reply from 2404:6800:4002:813::200e: time=87ms

Ping statistics for 2404:6800:4002:813::200e:
    Packets: Sent = 10, Received = 10, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 46ms, Maximum = 104ms, Average = 68ms

C:\Users\hp>ping -6 -n 10 sigcomm.org

Pinging sigcomm.org [64:ff9b::be5c:9e04] with 32 bytes of data:
Reply from 64:ff9b::be5c:9e04: time=457ms
Reply from 64:ff9b::be5c:9e04: time=403ms
Reply from 64:ff9b::be5c:9e04: time=504ms
Reply from 64:ff9b::be5c:9e04: time=444ms
Reply from 64:ff9b::be5c:9e04: time=388ms
Reply from 64:ff9b::be5c:9e04: time=491ms
Reply from 64:ff9b::be5c:9e04: time=349ms
Reply from 64:ff9b::be5c:9e04: time=378ms
Reply from 64:ff9b::be5c:9e04: time=342ms
Reply from 64:ff9b::be5c:9e04: time=435ms

Ping statistics for 64:ff9b::be5c:9e04:
    Packets: Sent = 10, Received = 10, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 342ms, Maximum = 504ms, Average = 419ms
```

How to force IPv6 on networks:

- First, enable IPv6 in control panel, adapter settings
- Use the ‘-6’ option with ping to force IPv6 on network
- Eg: ping -6 google.com or ping -6 sigcomm.org

Analysis of forcing IPv6 on networks:

- Success for ‘google.com’ and ‘sigcomm.org’ on mobile network
- Success for ‘google.com’ on IITD network
- Failure for ‘sigcomm.org’ on mobile network

Possible Reasons for failure for sigcomm.org on IITD network:

- IITD network may have firewalls blocking or security settings blocking IPv6 traffic to certain domains

- Maybe routing issues or misconfigurations of DNS for sigcomm.org affecting IPv6 resolution
- IITD network may not have reliance on modern networking which have better support for IPv6 infrastructure like in the mobile networks

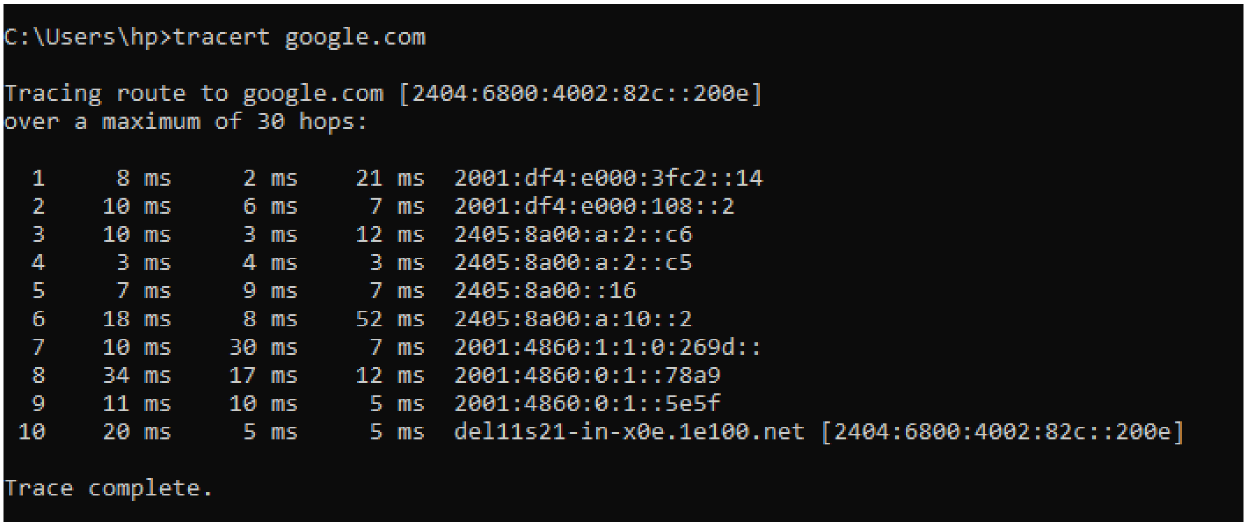
Question-1.2:

Logging IP address using IITD network in 4 cases:

<website, network>	**IP address for each case**
<google.com, IITD network>	2404:6800:4002:82c::200e
<google.com, mobile network>	2404:6800:4002:813::200e
<sigcomm.org, IITD network>	190.92.158.4
<sigcomm.org, mobile network>	64:ff9b::be5c:9e04

1.2(A)

Traceroute for google.com using IITD network:



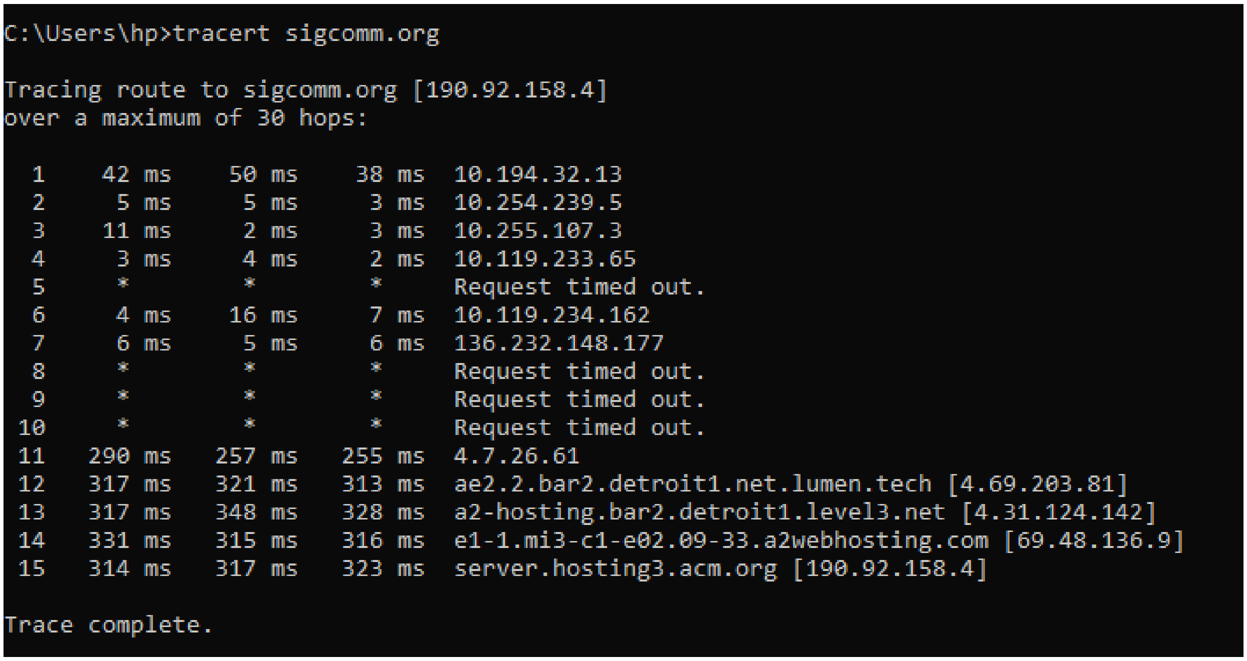
```
C:\Users\hp>tracert google.com

Tracing route to google.com [2404:6800:4002:82c::200e]
over a maximum of 30 hops:

  1     8 ms     2 ms    21 ms  2001:df4:e000:3fc2::14
  2    10 ms     6 ms     7 ms  2001:df4:e000:108::2
  3    10 ms     3 ms    12 ms  2405:8a00:a:2::c6
  4     3 ms     4 ms     3 ms  2405:8a00:a:2::c5
  5     7 ms     9 ms     7 ms  2405:8a00::16
  6    18 ms     8 ms    52 ms  2405:8a00:a:10::2
  7    10 ms    30 ms     7 ms  2001:4860:1:1:0:269d::
  8    34 ms    17 ms    12 ms  2001:4860:0:1::78a9
  9    11 ms    10 ms     5 ms  2001:4860:0:1::5e5f
 10    20 ms     5 ms     5 ms  del11s21-in-x0e.1e100.net [2404:6800:4002:82c::200e]

Trace complete.
```

- Number of IP hops=10

IP address	**List of Autonomous systems**
2001:df4:e000:3fc2::14	Indian Institute of Technology Delhi, AS132780
2001:df4:e000:108::2	

2405:8a00:a:2::c6	NKN Core Network, AS55824
2405:8a00:a:2::c5	
2405:8a00::16	
2405:8a00:a:10::2	
2001:4860:1:1:0:269d::	Google LLC, AS15169
2001:4860:0:1::78a9	
2001:4860:0:1::5e5f	
del11s21-in-x0e.1e100.net 2404:6800:4002:82c::200e	

Traceroute for sigcomm.org using IITD network:



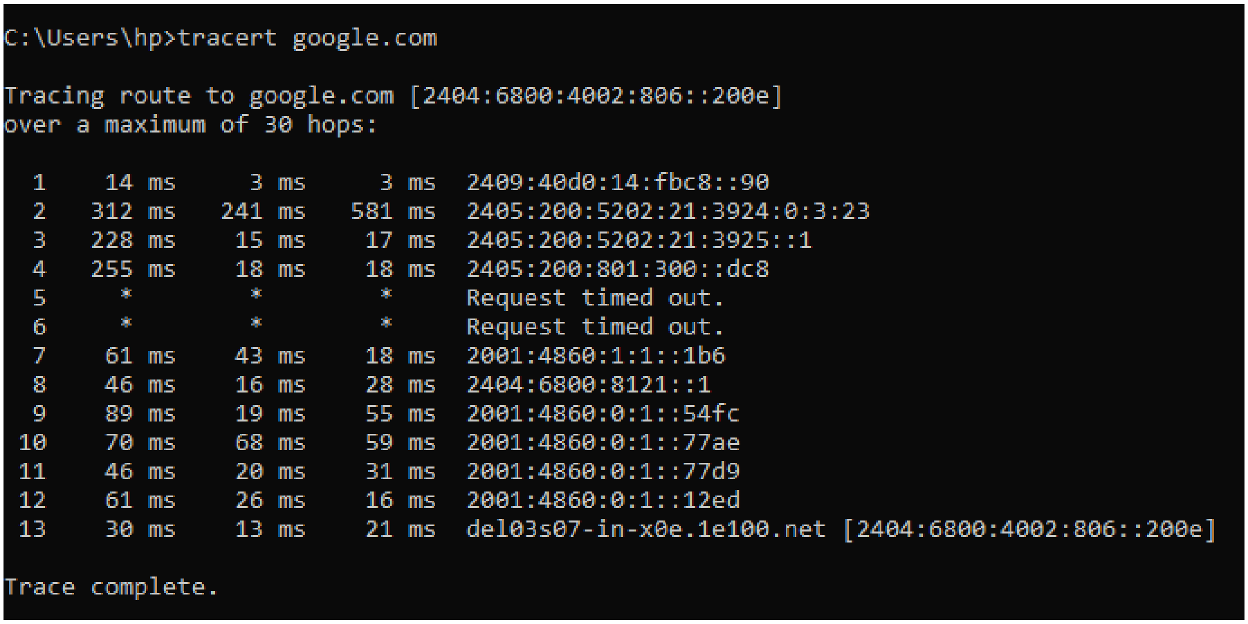
```
C:\Users\hp>tracert sigcomm.org

Tracing route to sigcomm.org [190.92.158.4]
over a maximum of 30 hops:

  1    42 ms    50 ms    38 ms  10.194.32.13
  2     5 ms     5 ms     3 ms  10.254.239.5
  3    11 ms     2 ms     3 ms  10.255.107.3
  4     3 ms     4 ms     2 ms  10.119.233.65
  5     *        *        *     Request timed out.
  6     4 ms    16 ms     7 ms  10.119.234.162
  7     6 ms     5 ms     6 ms  136.232.148.177
  8     *        *        *     Request timed out.
  9     *        *        *     Request timed out.
 10     *        *        *     Request timed out.
 11   290 ms   257 ms   255 ms  4.7.26.61
 12   317 ms   321 ms   313 ms  ae2.2.bar2.detroit1.net.lumen.tech [4.69.203.81]
 13   317 ms   348 ms   328 ms  a2-hosting.bar2.detroit1.level3.net [4.31.124.142]
 14   331 ms   315 ms   316 ms  e1-1.mi3-c1-e02.09-33.a2webhosting.com [69.48.136.9]
 15   314 ms   317 ms   323 ms  server.hosting3.acm.org [190.92.158.4]

Trace complete.
```

- Number of hops with responses =11
- Number of hops including time outs =15

IP address	List of autonomous systems
10.194.32.13	These IP addresses don’t have ASN data maybe because they are owned by IIT D and they have not delegated their IP addresses to an ASN
10.254.239.5	
10.255.107.3	

10.119.233.65	
10.119.234.162	
136.232.148.177	Reliance Jio Infocomm Limited, AS55836
4.7.26.61	Level 3 Parent, LLC, AS3356
ae2.2.bar2.detroit1.net.lumen.tech 4.69.203.81	
a2-hosting.bar2.detroit1.level3.net 4.31.124.142	
e1-1.mi3-c1-e02.09-33.a2webhosting.com 69.48.136.9	A2 Hosting, Inc., AS55293
server.hosting3.acm.org 190.92.158.4	

Traceroute for google.com using mobile network:



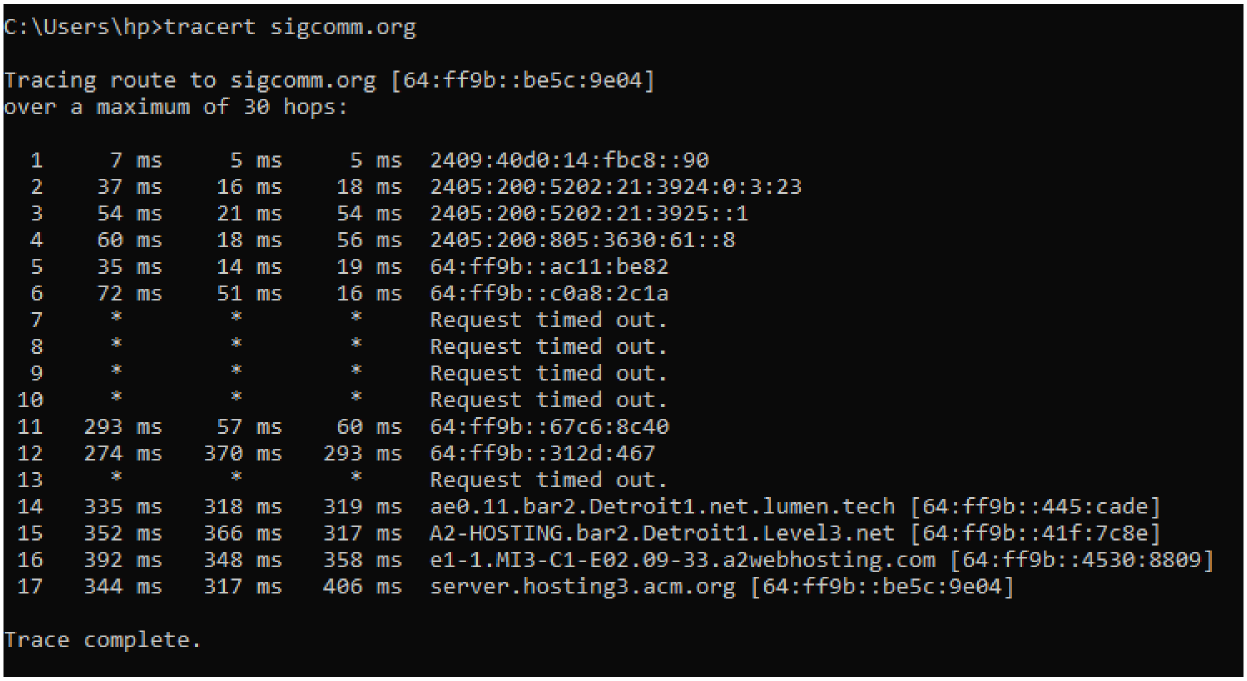
```
C:\Users\hp>tracert google.com

Tracing route to google.com [2404:6800:4002:806::200e]
over a maximum of 30 hops:

  1    14 ms     3 ms     3 ms  2409:40d0:14:fbc8::90
  2   312 ms   241 ms   581 ms  2405:200:5202:21:3924:0:3:23
  3   228 ms    15 ms    17 ms  2405:200:5202:21:3925::1
  4   255 ms    18 ms    18 ms  2405:200:801:300::dc8
  5     *        *        *     Request timed out.
  6     *        *        *     Request timed out.
  7    61 ms    43 ms    18 ms  2001:4860:1:1::1b6
  8    46 ms    16 ms    28 ms  2404:6800:8121::1
  9    89 ms    19 ms    55 ms  2001:4860:0:1::54fc
 10    70 ms    68 ms    59 ms  2001:4860:0:1::77ae
 11    46 ms    20 ms    31 ms  2001:4860:0:1::77d9
 12    61 ms    26 ms    16 ms  2001:4860:0:1::12ed
 13    30 ms    13 ms    21 ms  del03s07-in-x0e.1e100.net [2404:6800:4002:806::200e]

Trace complete.
```

- Number of hops with responses =11
- Number of hops including time outs =13

IP Address	**List of autonomous systems**
2409:40d0:14:fbc8::90	Reliance Jio Infocomm Limited, AS55836
2405:200:5202:21:3924:0:3:23	

2405:200:5202:21:3925::1	
2405:200:801:300::dc8	
2001:4860:1:1::1b6	Google LLC, AS15169
2404:6800:8121::1	Google IPv6 address block in AP, AS15169
2001:4860:0:1::54fc	Google LLC, AS15169
2001:4860:0:1::77ae	
2001:4860:0:1::77d9	
2001:4860:0:1::12ed	
del03s07-in-x0e.1e100.net 2404:6800:4002:806::200e	Google IPv6 address block in AP, AS15169

Traceroute for sigcomm.org using mobile network:



```
C:\Users\hp>tracert sigcomm.org

Tracing route to sigcomm.org [64:ff9b::be5c:9e04]
over a maximum of 30 hops:

  1     7 ms     5 ms     5 ms  2409:40d0:14:fbc8::90
  2    37 ms    16 ms    18 ms  2405:200:5202:21:3924:0:3:23
  3    54 ms    21 ms    54 ms  2405:200:5202:21:3925::1
  4    60 ms    18 ms    56 ms  2405:200:805:3630:61::8
  5    35 ms    14 ms    19 ms  64:ff9b::ac11:be82
  6    72 ms    51 ms    16 ms  64:ff9b::c0a8:2c1a
  7     *        *        *     Request timed out.
  8     *        *        *     Request timed out.
  9     *        *        *     Request timed out.
 10     *        *        *     Request timed out.
 11   293 ms    57 ms    60 ms  64:ff9b::67c6:8c40
 12   274 ms   370 ms   293 ms  64:ff9b::312d:467
 13     *        *        *     Request timed out.
 14   335 ms   318 ms   319 ms  ae0.11.bar2.Detroit1.net.lumen.tech [64:ff9b::445:cade]
 15   352 ms   366 ms   317 ms  A2-HOSTING.bar2.Detroit1.Level3.net [64:ff9b::41f:7c8e]
 16   392 ms   348 ms   358 ms  e1-1.MI3-C1-E02.09-33.a2webhosting.com [64:ff9b::4530:8809]
 17   344 ms   317 ms   406 ms  server.hosting3.acm.org [64:ff9b::be5c:9e04]

Trace complete.
```

- Number of hops with responses =12
- Number of hops including time outs =17

IP address	List of Autonomous systems
2409:40d0:14:fbc8::90	Reliance Jio Infocomm Limited, AS55836

2405:200:5202:21:3924:0:3:23	
2405:200:5202:21:3925::1	
2405:200:805:3630:61::8	
64:ff9b::ac11:be82	Reliance Jio Infocomm Pte. Ltd, AS64049
64:ff9b::c0a8:2c1a	
64:ff9b::67c6:8c40	
64:ff9b::312d:467	
ae0.11.bar2.Detroit1.net.lumen.tech 64:ff9b::445:cade	Level 3 Parent, LLC, AS3356
A2-HOSTING.bar2.Detroit1.Level3.net 64:ff9b::41f:7c8e	
e1-1.MI3-C1-E02.09-33.a2webhosting.com 64:ff9b::4530:8809	A2 Hosting, Inc. , AS55293
server.hosting3.acm.org 64:ff9b::be5c:9e04	

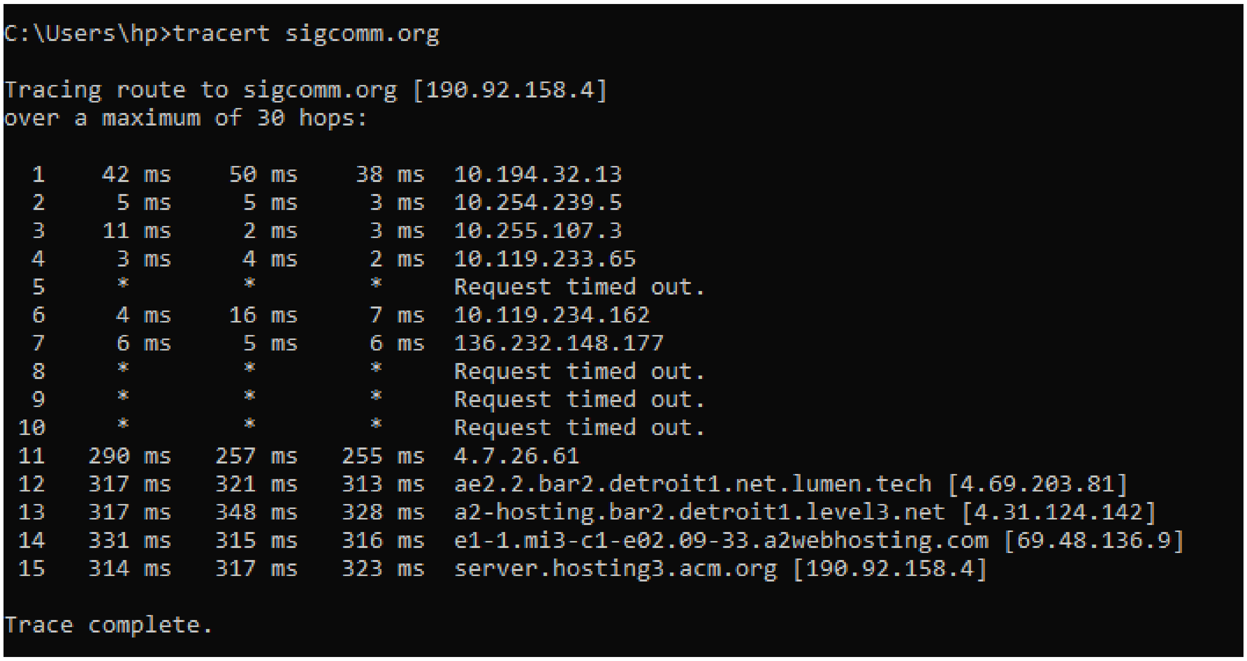
1.2(B)

Yes, I observed '*' in the output in 3 traceroutes out of 4 that I performed. Possible reasons are the following:

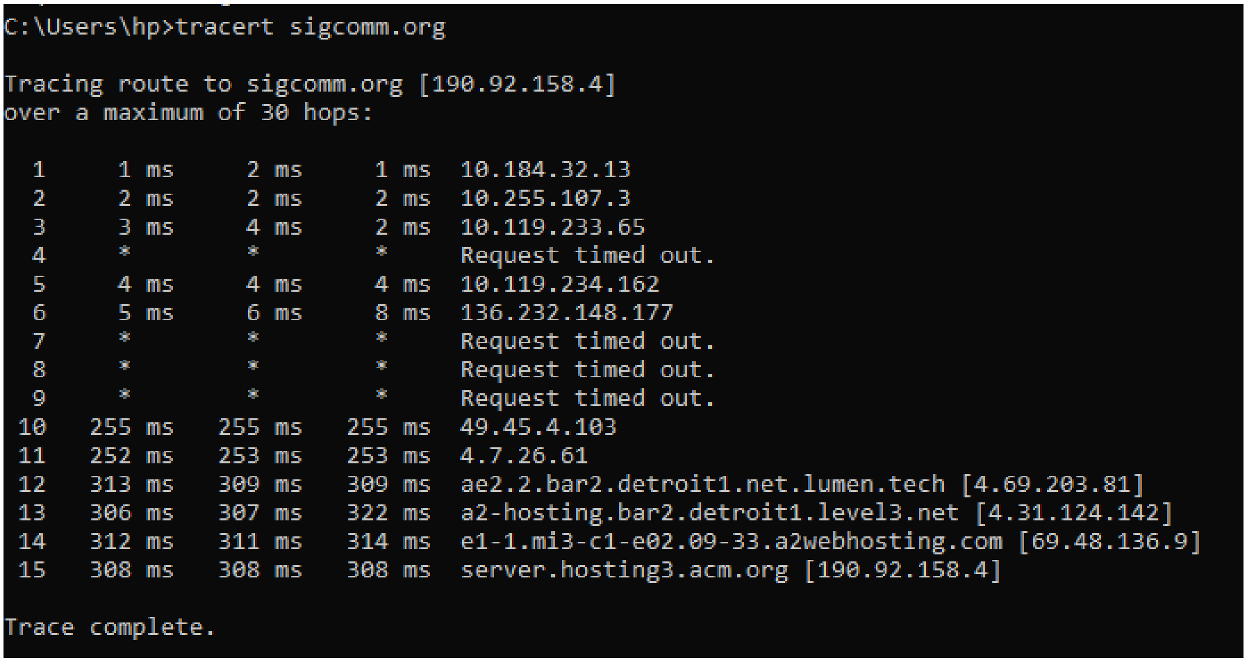
- Maybe some routers ignored the traceroute packets and so as time out occurred it displayed *.
- Firewalls might have blocked the traffic
- Packets may have been dropped due to network congestion. As ICMP packets have low priority and if the routers are busy processing other types of traffic, they choose to drop ICMP packets

1.2(C)

Yes, I have observed multiple IP addresses for same hop count when I did traceroute for sigcomm.org twice using IITD network.Here are the screenshots:



```
C:\Users\hp>tracert sigcomm.org

Tracing route to sigcomm.org [190.92.158.4]
over a maximum of 30 hops:

  1    42 ms    50 ms    38 ms  10.194.32.13
  2     5 ms     5 ms     3 ms  10.254.239.5
  3    11 ms     2 ms     3 ms  10.255.107.3
  4     3 ms     4 ms     2 ms  10.119.233.65
  5     *        *        *     Request timed out.
  6     4 ms    16 ms     7 ms  10.119.234.162
  7     6 ms     5 ms     6 ms  136.232.148.177
  8     *        *        *     Request timed out.
  9     *        *        *     Request timed out.
 10     *        *        *     Request timed out.
 11   290 ms   257 ms   255 ms  4.7.26.61
 12   317 ms   321 ms   313 ms  ae2.2.bar2.detroit1.net.lumen.tech [4.69.203.81]
 13   317 ms   348 ms   328 ms  a2-hosting.bar2.detroit1.level3.net [4.31.124.142]
 14   331 ms   315 ms   316 ms  e1-1.mi3-c1-e02.09-33.a2webhosting.com [69.48.136.9]
 15   314 ms   317 ms   323 ms  server.hosting3.acm.org [190.92.158.4]

Trace complete.
```



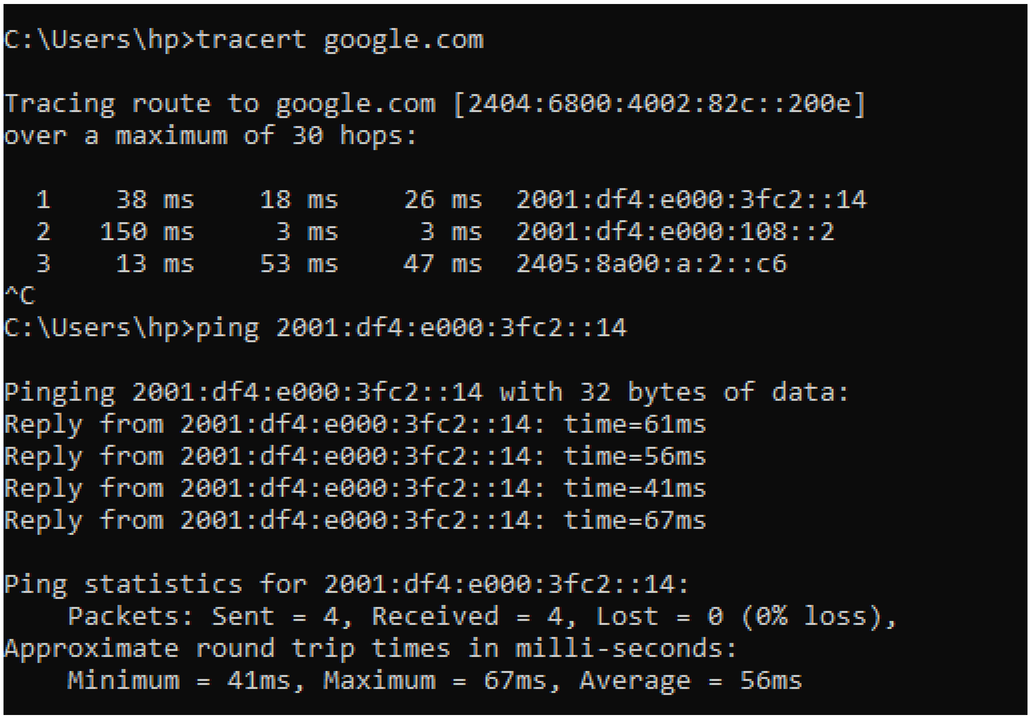
```
C:\Users\hp>tracert sigcomm.org

Tracing route to sigcomm.org [190.92.158.4]
over a maximum of 30 hops:

  1     1 ms     2 ms     1 ms  10.184.32.13
  2     2 ms     2 ms     2 ms  10.255.107.3
  3     3 ms     4 ms     2 ms  10.119.233.65
  4     *        *        *     Request timed out.
  5     4 ms     4 ms     4 ms  10.119.234.162
  6     5 ms     6 ms     8 ms  136.232.148.177
  7     *        *        *     Request timed out.
  8     *        *        *     Request timed out.
  9     *        *        *     Request timed out.
 10   255 ms   255 ms   255 ms  49.45.4.103
 11   252 ms   253 ms   253 ms  4.7.26.61
 12   313 ms   309 ms   309 ms  ae2.2.bar2.detroit1.net.lumen.tech [4.69.203.81]
 13   306 ms   307 ms   322 ms  a2-hosting.bar2.detroit1.level3.net [4.31.124.142]
 14   312 ms   311 ms   314 ms  e1-1.mi3-c1-e02.09-33.a2webhosting.com [69.48.136.9]
 15   308 ms   308 ms   308 ms  server.hosting3.acm.org [190.92.158.4]

Trace complete.
```

Reason:

- A network might dynamically change the IP addresses of its routers to increase security or for maintenance purposes.
- Some routers might have multiple interfaces or IP addresses assigned to them, and the IP address used in traceroute might vary based on routing policies or configurations.
- Maybe the network uses load balancing that distributes traffic across multiple paths.

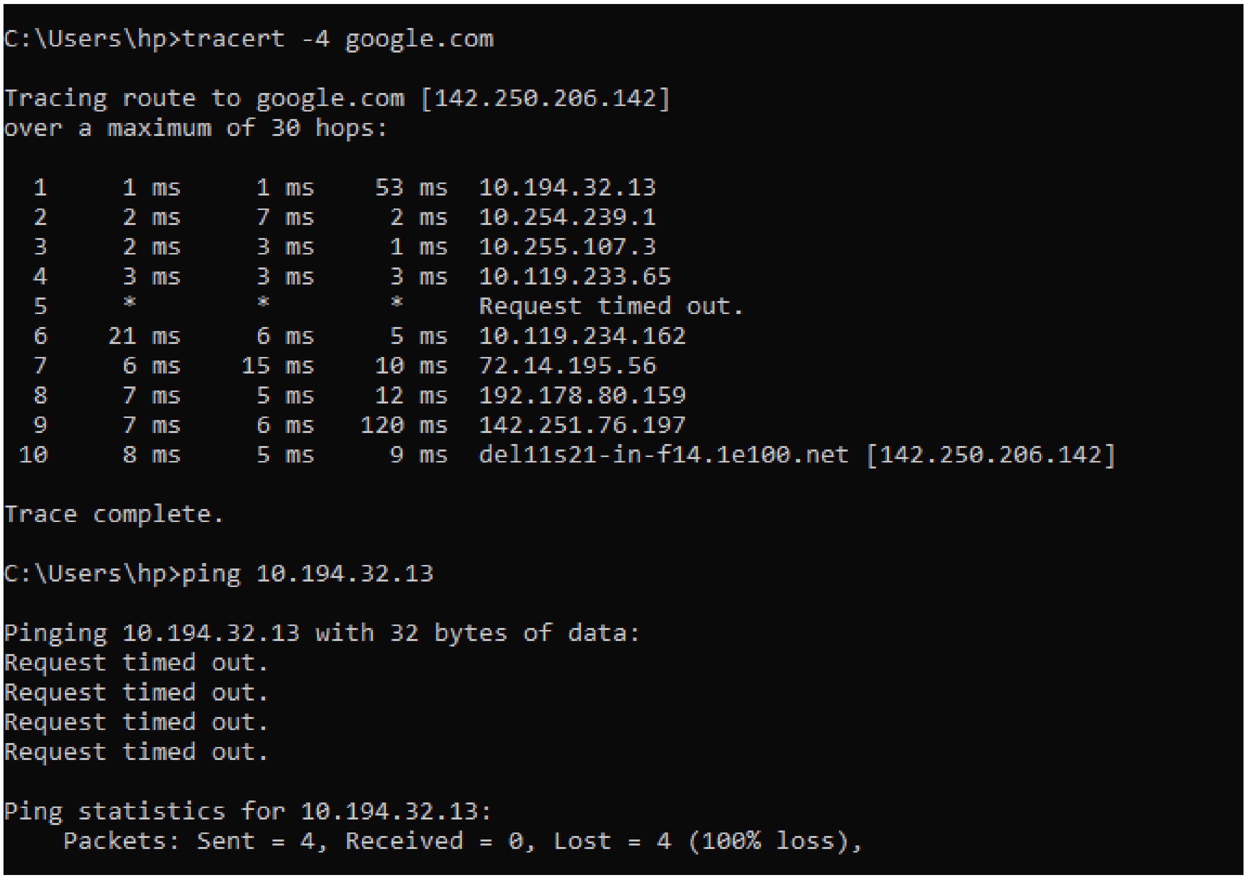
1.2(D)

Using IPv6 address:



```
C:\Users\hp>tracert google.com

Tracing route to google.com [2404:6800:4002:82c::200e]
over a maximum of 30 hops:

  1    38 ms    18 ms    26 ms  2001:df4:e000:3fc2::14
  2   150 ms     3 ms     3 ms  2001:df4:e000:108::2
  3    13 ms    53 ms    47 ms  2405:8a00:a:2::c6
^C
C:\Users\hp>ping 2001:df4:e000:3fc2::14

Pinging 2001:df4:e000:3fc2::14 with 32 bytes of data:
Reply from 2001:df4:e000:3fc2::14: time=61ms
Reply from 2001:df4:e000:3fc2::14: time=56ms
Reply from 2001:df4:e000:3fc2::14: time=41ms
Reply from 2001:df4:e000:3fc2::14: time=67ms

Ping statistics for 2001:df4:e000:3fc2::14:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 41ms, Maximum = 67ms, Average = 56ms
```

Using IPv4 address:



```
C:\Users\hp>tracert -4 google.com

Tracing route to google.com [142.250.206.142]
over a maximum of 30 hops:

  1     1 ms     1 ms    53 ms  10.194.32.13
  2     2 ms     7 ms     2 ms  10.254.239.1
  3     2 ms     3 ms     1 ms  10.255.107.3
  4     3 ms     3 ms     3 ms  10.119.233.65
  5     *        *        *     Request timed out.
  6    21 ms     6 ms     5 ms  10.119.234.162
  7     6 ms    15 ms    10 ms  72.14.195.56
  8     7 ms     5 ms    12 ms  192.178.80.159
  9     7 ms     6 ms   120 ms  142.251.76.197
 10     8 ms     5 ms     9 ms  del11s21-in-f14.1e100.net [142.250.206.142]

Trace complete.

C:\Users\hp>ping 10.194.32.13

Pinging 10.194.32.13 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 10.194.32.13:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

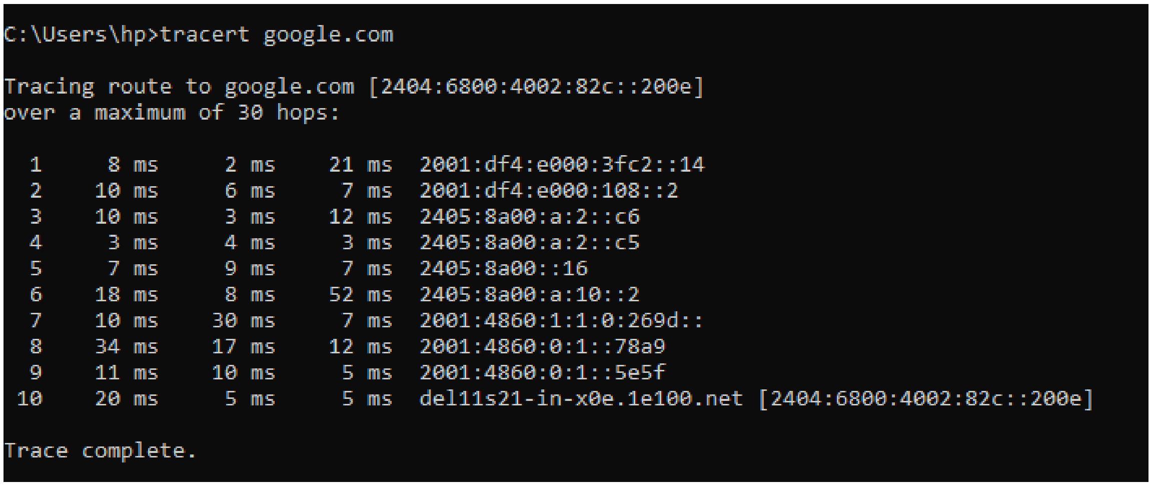
Reasons I'm able to ping the address of first hop router on IITD network from my mobile network in IPv6 and not in IPv4:

1. **Network configuration:**
    - ○ **IPv4:** IITD network may be using private IP addresses for internal routers which cannot be reachable from other networks like mobile networks. So, when I attempt to ping an internal router's IPv4 address from mobile network(public network) it's not reachable
    - ○ **IPv6:** IPv6 addresses are more globally routable than IPv4 addresses. May be the network configuration of IITD allows certain internal addresses to be reachable and hence receiving a response when I'm pinging an internal IPv6 address from mobile network
2. **Firewalls:**
    - ○ **IPv4:** To avoid external probing of internal network infrastructure, Firewalls and security settings might have blocked the pings to private IPv4 address
    - ○ **IPv6:** Maybe IPv6 addresses are less restrictive or have different security settings compared to IPv4
3. **Network address translation:**
    - ○ **IPv4:** Maybe the IITD private IPv4 addresses use NAT to interact with the internet and so are not reachable from outside the NATed network
    - ○ **IPv6:** IPv6 is designed to avoid the need for NAT. Hence the internal IPv6 addresses may be more accessible

1.2(E)

Yes, I **observed a 2-tier architecture** in the traceroute of google.com from the IITD network. Here’s the explanation:

- From the traceroute information(screenshot), The first 2 IP addresses comes under IITD network and next 4 IP addresses comes under NKN network and the last 4 IP addresses comes under Google network

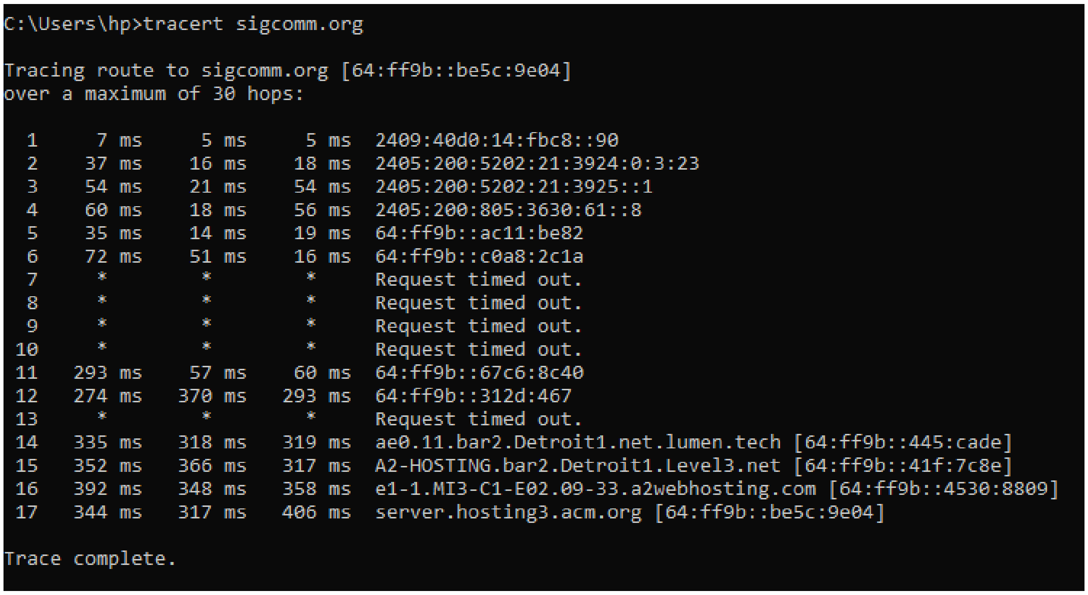


```
C:\Users\hp>tracert google.com

Tracing route to google.com [2404:6800:4002:82c::200e]
over a maximum of 30 hops:

  1     8 ms     2 ms    21 ms  2001:df4:e000:3fc2::14
  2    10 ms     6 ms     7 ms  2001:df4:e000:108::2
  3    10 ms     3 ms    12 ms  2405:8a00:a:2::c6
  4     3 ms     4 ms     3 ms  2405:8a00:a:2::c5
  5     7 ms     9 ms     7 ms  2405:8a00::16
  6    18 ms     8 ms    52 ms  2405:8a00:a:10::2
  7    10 ms    30 ms     7 ms  2001:4860:1:1:0:269d::
  8    34 ms    17 ms    12 ms  2001:4860:0:1::78a9
  9    11 ms    10 ms     5 ms  2001:4860:0:1::5e5f
 10    20 ms     5 ms     5 ms  del11s21-in-x0e.1e100.net [2404:6800:4002:82c::200e]

Trace complete.
```

- **IIT Delhi-** AS132780- Represents educational institution network which connects to **NKN Core Network** (AS55824) which connects various educational and research Institutions across India.to Global networks like Google. In the traceroute Tier 3(IIT Delhi) connects to Tier 2(NKN) which inturn connects to **Tier 1(Google)**. Direct transition from NKN to Google indicates efficient peering, where NKN can directly route traffic to Google’s global network.This avoids the need for additional intermediary Tier 2 or Tier 1 network , which simplifies the path
- **No 3-Tier architecture because:** NKN has direct peering with Google which implies that there is no need of an additional Tier-1 network between them and In NKN routing is optimized for efficiency, leading to fewer tiers in the path from source to destination.

I have also observed 2-tier architecture in the traceroute for sigcomm.org using mobile network. Here’s the explanation:

- For the trace route of sigcomm.org,From the traceroute(screenshot) and the ASN numbers, the traceroute moves from Tier-3 network which is Reliance Jio Ltd. to a Tier 2 network (Reliance Jio Infocomm Pte. Ltd) and then to Level 3 Parent, LLC which is a tier 1 network and finally reaching the host of provider’s network, which could be classified as either Tier 2 or Tier 3 network



```
C:\Users\hp>tracert sigcomm.org

Tracing route to sigcomm.org [64:ff9b::be5c:9e04]
over a maximum of 30 hops:

  1     7 ms     5 ms     5 ms  2409:40d0:14:fbc8::90
  2    37 ms    16 ms    18 ms  2405:200:5202:21:3924:0:3:23
  3    54 ms    21 ms    54 ms  2405:200:5202:21:3925::1
  4    60 ms    18 ms    56 ms  2405:200:805:3630:61::8
  5    35 ms    14 ms    19 ms  64:ff9b::ac11:be82
  6    72 ms    51 ms    16 ms  64:ff9b::c0a8:2c1a
  7     *        *        *     Request timed out.
  8     *        *        *     Request timed out.
  9     *        *        *     Request timed out.
 10     *        *        *     Request timed out.
 11   293 ms    57 ms    60 ms  64:ff9b::67c6:8c40
 12   274 ms   370 ms   293 ms  64:ff9b::312d:467
 13     *        *        *     Request timed out.
 14   335 ms   318 ms   319 ms  ae0.11.bar2.Detroit1.net.lumen.tech [64:ff9b::445:cade]
 15   352 ms   366 ms   317 ms  A2-HOSTING.bar2.Detroit1.Level3.net [64:ff9b::41f:7c8e]
 16   392 ms   348 ms   358 ms  e1-1.MI3-C1-E02.09-33.a2webhosting.com [64:ff9b::4530:8809]
 17   344 ms   317 ms   406 ms  server.hosting3.acm.org [64:ff9b::be5c:9e04]

Trace complete.
```

- So there is a direct transition from Tier 2 network (Reliance Jio Infocomm) to a Tier 1 network (Level 3) with no intermediary in between. This simplifies the architecture to 2 tiers.

1.2(F)

Analysis of Traceroute for Google.com

Hop	IP address	Geo-Locations	RTT 1	RTT 2	RTT 3
1	2001:df4:e000:3fc2::14	Delhi	8ms	2 ms	21 ms
2	2001:df4:e000:108::2	Delhi	10 ms	6 ms	7 ms
3	2405:8a00:a:2::c6	Chennai, Tamil Nadu	10 ms	3 ms	12 ms
4	2405:8a00:a:2::c5	Chennai, Tamil Nadu	3 ms	4 ms	3 ms
5	2405:8a00::16	Delhi	7 ms	9 ms	7 ms
6	2405:8a00:a:10::2	Delhi	18 ms	8 ms	52 ms
7	2001:4860:1:1:0:269d::	Hesse, DE, Germany	10 ms	30 ms	7 ms
8	2001:4860:0:1::78a9	Hesse, DE, Germany	34 ms	17 ms	12 ms
9	2001:4860:0:1::5e5f	Hesse, DE, Germany	11 ms	10 ms	5 ms
10	del11s21-in-x0e.1e100.net	Delhi	20 ms	5 ms	5 ms

	2404:6800:4002:82c::200e				

Geographical Path and RTT Analysis:

- **Delhi hops:** Hops 1 and 2 are in Delhi, with RTT ranging from 2 ms to 21 ms. Low RTTs consistent with local traffic within the city. 21 ms is slightly higher maybe due to network conditions or temporary conditions.
- **Transition to Chennai:** Hop 3 is transition from Delhi to Chennai with RTTs from 3 ms to 12 ms. 3ms is slightly low which shows efficient routing between Delhi and Chennai but the next RTTs are 10 ms and 12 ms which maybe due to the network traffic between the cities
- **Chennai Hops:** Hop 4 is within Chennai and the RTTs are very low as it is within the same city and the network traffic is also very less. Hence lower RTTs 3 ms, 4 ms
- **Return to Delhi:** Hop 5 is return to Delhi with RTTs 7ms, 9 ms and 7 ms. This low RTT maybe because of efficient routing or direct connection between Delhi and Chennai
- **Hop within Delhi:** Hop 6 within Delhi where RTTs are 18 ms , 8 ms, 52 ms.There is so much variation in RTT here this maybe because of queuing or processing delays at intermediate routers for some packets. Maybe these packets took different paths and maybe congestion was high in one path and so higher value of RTT in that path**.**
- **International Hop:** Hop 7 is an international hop from Delhi to Germany and RTTs are 10 ms, 30 ms and 7 ms.But the expected RTT between Delhi to Germany(5,800 Km to 600 Km) should be in the range of 60 ms to 80 ms and due to additional network processing, routing and congestion, Practical RTTs would be in the range of 100 ms to 200 ms depending on the network connection quality but here the RTTs are far below than expected which shows that something unusual is happening at this hop like incorrect time stamping, network load balancing or maybe a router misreported the destination of packet or looped the packets back to a nearby location.
- **Within Germany:** Hop 8 and 9 are within Germany with RTTs in the range 5 ms to 34 ms. Hop 9 has relatively lower RTT because this hop maybe very close to the previous one, maybe within same data center. Hop 8 has slightly larger RTTs maybe because they are farther apart within Germany but the variation in RTT upto 34 ms could be due to temporary network congestion or varying processing delays in different paths.
- **Final hop Germany to Delhi:** The final hop from Germany to Delhi with RTTs varying from 5 ms to 20 ms. This variation could be due to final routing within Delhi or slight network congestion which increased RTT to 20 ms

Conclusion: The initial RTTs make sense within Delhi with low latency consistent with local network traffic. Transition to Chennai shows stable and efficient national routing. The jump to Germany which doesn't match with the expected RTT, within Germany shows moderate increase maybe due to network traffic and return to Delhi sees a reduction in RTT but slight increase in network traffic or processing delays.

Analysis of Trace route for Sigcomm.org

IP address	Geo locations	RTT 1	RTT 2	RTT 3

10.194.32.13	Private address	42 ms	50 ms	38 ms
10.254.239.5	Private address	5 ms	5 ms	3 ms
10.255.107.3	Private address	11 ms	2 ms	3 ms
10.119.233.65	Private address	3 ms	4 ms	2 ms
10.119.234.162	Private address	4 ms	16 ms	7 ms
136.232.148.177	Reliance Jio Infocomm Limited, Delhi	6 ms	5 ms	6 ms
4.7.26.61	Level 3 Parent, LLC, Los Angeles, California, US.	290 ms	257 ms	255 ms
ae2.2.bar2.detroit1.net.lumen.tech 4.69.203.81	Level 3 Parent, LLC,Detroit, Michigan, US.	317 ms	321 ms	313 ms
a2-hosting.bar2.detroit1.level3.net 4.31.124.142	Level 3 Parent, LLC,Detroit, Michigan, US.	317 ms	348 ms	328 ms
e1-1.mi3-c1-e02.09-33.a2webhosting.com 69.48.136.9	A2 Hosting, Inc., AS55293, Detroit, Michigan, US	331 ms	315 ms	316 ms
server.hosting3.acm.org 190.92.158.4	A2 Hosting, Inc., AS55293, Detroit, Michigan, US	314 ms	317 ms	323 ms

Geographical Path and RTT Analysis:

- **Internal IP hops:** Hops 1,2,3,4,5 are hops within private network where RTTs are generally low, but the first entry(10.194.32.13) is slightly higher RTTs maybe due to network conditions or congestion or maybe the distance between network nodes within the internal network.
- **Delhi hop:** Hop 6 is transition from private IPs to Delhi(Reliance)with RTTs 5ms-6ms. This is because the IP is in Delhi itself and so it is geographically close hence very fast.
- **International Hop:** Hop 7 is transition from Delhi to US. RTTs are higher(255 ms -290 ms) which is reasonable given distance between them, which increases latency
- **Los Angeles to Detroit, Michigan:** Hop 8 has RTTs ranging 313 ms - 348 ms which are slightly higher than the international hop, possibly due to routing path taken to reach Detroit or network congestion.Hop 9 within Detriot also show higher values of RTTs(314 ms-348 ms)maybe due to the same above reasons
- **Hop to server, Detroit, US:** Last 2 hops to reach server also show higher RTTs(314 ms-331 ms) maybe reflecting the physical distance and likely network routing from source to the servers in Detroit

Conclusion: The initial RTTs make sense within private IPs have low latency consistent with minor variations likely due to network conditions within the private network.The RTTs for US based IPs are higher which makes sense given the international distances involved.The RTTs increase as expected when moving from Los Angeles to Detroit.

Comparison between Google.com and Sigcomm.org:

- **Google.com:** Local RTTs are very low with minimal variation; international RTTs are lower than expected, possibly shows measurement anomalies
- **Sigcomm.org:** Local RTTs are low but the international RTTs are much higher, showing the true latency over long distances
- **Routing complexity: Google.com** shows efficient national and international routing but few anomalies in international paths. **Sigcomm.org** shows standard delays for international and domestic US routing indicating expected network performance given the geographical distances
- **Consistency: Google.com** consistent RTTs with some international anomalies and Sigcomm.org consistent with high RTTs for long international routes and also intra-country routing.

Question-2:

2(B)pcap file link:https://drive.google.com/file/d/1sHi9cu91IG2b5pzDk585H7HnQQux46sQ/view?usp=sharing

Network-Layer protocols: predominantly IPv4

- **IPv4:** Filter: ‘ip’,percentage=99.9%, Packets=39,741
- **IPv6:** Filter: ‘ipv6’, percentage=0.1%, Packets=39

Transport-Layer protocols: predominantly UDP

- **User Datagram protocol(UDP):** Filter: ‘udp’, percentage=99.4%, Packets=39,552
- **Transmission Control protocol(TCP):** Filter: ‘tcp’, percentage=0.57%, Packets=228

Application- Layer protocols: STUN and RTP are primary protocols with a significant percentage of the traffic being unclassified “Data”.

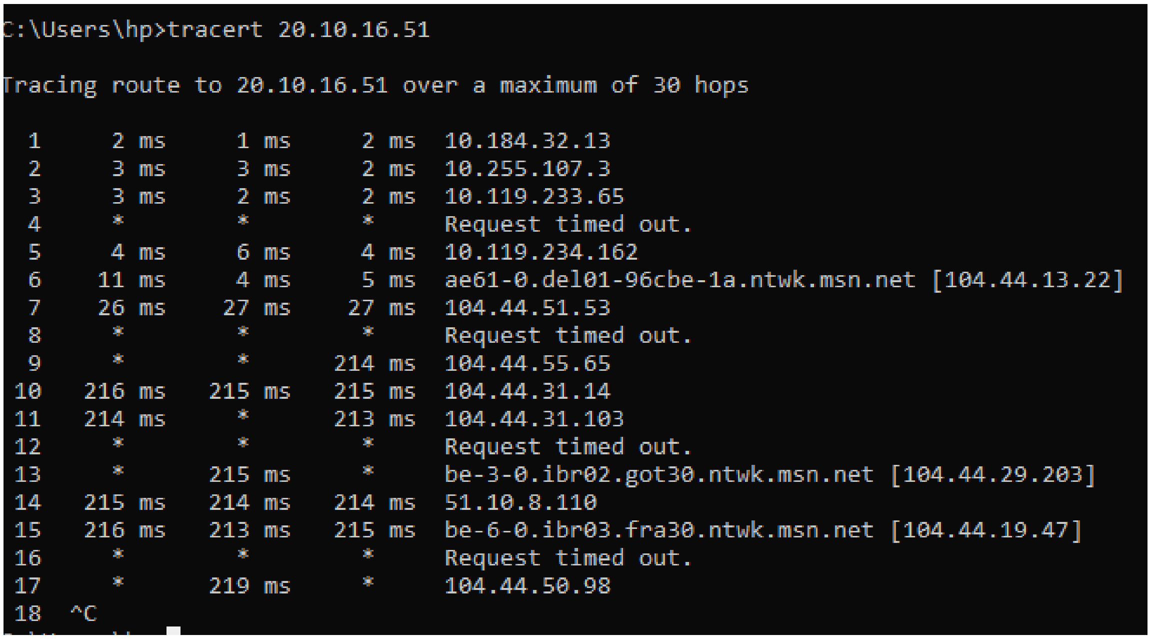
- **Session Traversal Utilities for NAT(STUN):** Filter: ‘stun’, percentage=57.1%, Packets=22,731
- **Real-time Transport Protocol (RTP):** Filter: ‘rtp’, percentage=0.7%, Packets=298
- **Simple Service Discovery Protocol (SSDP):** Filter: ‘ssdp’,percentage=0.0%, Packets=8
- **Domain Name System (DNS):** Filter: ‘dns’, percentage=0.1%, Packets=24
- **Transport Layer Security (TLS):** Filter: ‘tls’, percentage=0.18%, Packets=71

- **Malformed packets:** percentage=0.0%, Packets=16
- **Data(Unclassified):** percentage=98.1%, Packets=39,022

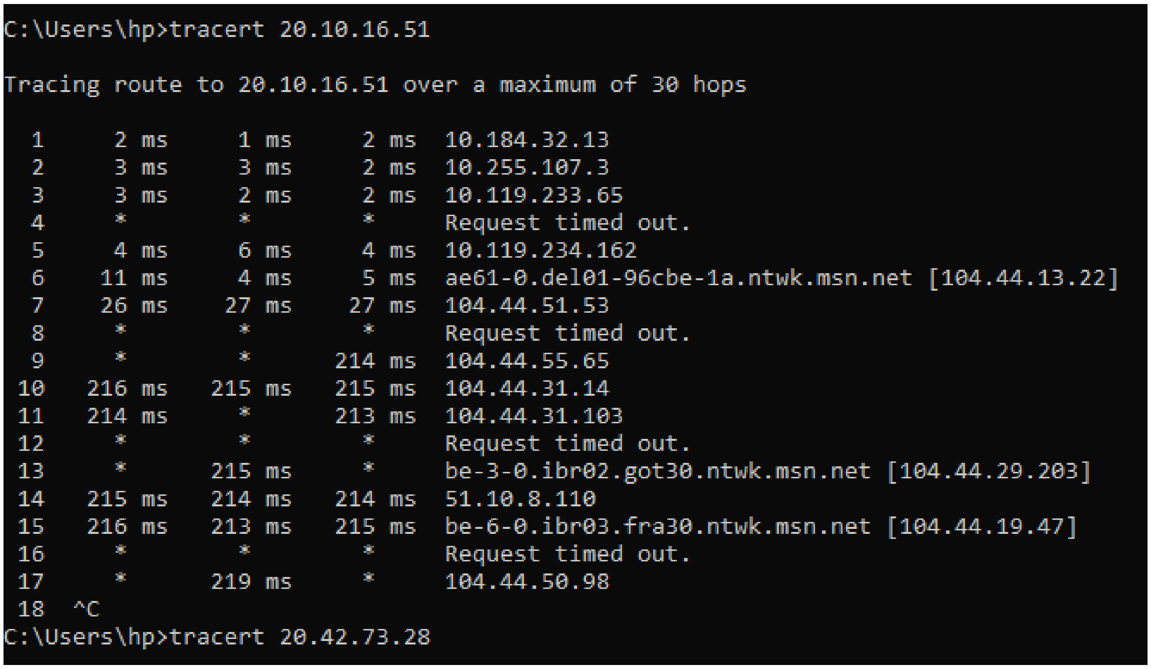
2(C)

No, I don’t observe a direct connection between the two hosts.Endpoint for host A could be ‘2001:df4:e000:3fd2:8cb8:fa74:d9d5:819e’ communicating with various IPv6 addresses, indicating indirect communication. The connections sem to involve various external IPs, which implies different network paths or intermediaries. Therefore, **it is not the same endpoint;**the traffic is routed through different IP addresses or networks.
What could be happening if it is not a direct connection:
Because I do not observe packets going back and forth between two IP addresses and the percentage of packets transferred using STUN protocol is 57.1% which shows that the hosts maybe behind an NAT. NAT traversal includes intermediate servers. Maybe there are security conditions or firewall which makes it impossible for a direct communication even when it is possible and inturn connects both through a central server.
NAT is used to map private IP addresses to a public IP address.In the wireshark communications, I have observed that there is connection between private IP address and public IP address which shows that it might involve NAT. Few examples where I found that private IP address is directly communicating with public IP address are:

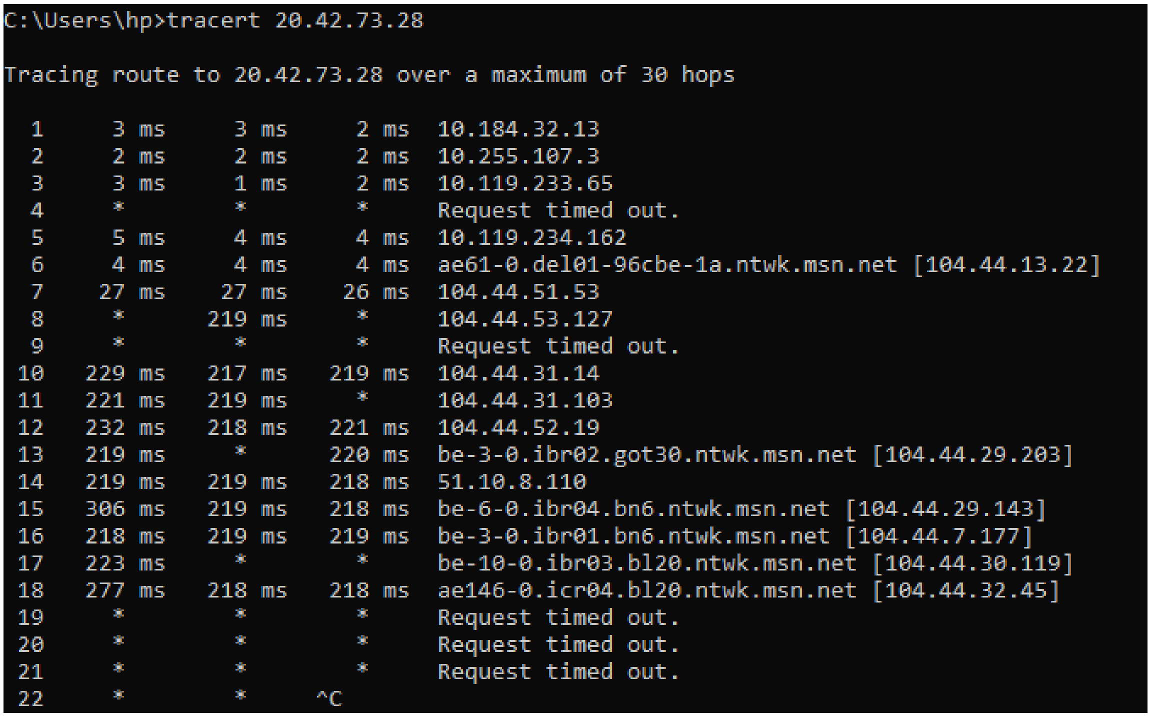
- 10.184.50.20 which is a private IP address (in the range 10.0.0.0 to 10.255.255.255) communicates to IP addresses like 20.10.16.51, 20.42.73.28, 20.189.173.16, 34.193.227.236 which are identified as public IP addresses
- I have also performed a trace route for addresses: 20.10.16.51, 20.42.73.28 and 20.189.173.16 which showed that initial hops are private IP addresses and the later hops are Public IP addresses which show that NAT is used.



```
:\Users\hp>tracert 20.10.16.51

racing route to 20.10.16.51 over a maximum of 30 hops

  1     2 ms     1 ms     2 ms  10.184.32.13
  2     3 ms     3 ms     2 ms  10.255.107.3
  3     3 ms     2 ms     2 ms  10.119.233.65
  4     *        *        *     Request timed out.
  5     4 ms     6 ms     4 ms  10.119.234.162
  6    11 ms     4 ms     5 ms  ae61-0.del01-96cbe-1a.ntwk.msn.net [104.44.13.22]
  7    26 ms    27 ms    27 ms  104.44.51.53
  8     *        *        *     Request timed out.
  9     *        *      214 ms  104.44.55.65
 10   216 ms   215 ms   215 ms  104.44.31.14
 11   214 ms     *      213 ms  104.44.31.103
 12     *        *        *     Request timed out.
 13     *      215 ms     *     be-3-0.ibr02.got30.ntwk.msn.net [104.44.29.203]
 14   215 ms   214 ms   214 ms  51.10.8.110
 15   216 ms   213 ms   215 ms  be-6-0.ibr03.fra30.ntwk.msn.net [104.44.19.47]
 16     *        *        *     Request timed out.
 17     *      219 ms     *     104.44.50.98
 18  ^C
```



```
C:\Users\hp>tracert 20.10.16.51

Tracing route to 20.10.16.51 over a maximum of 30 hops

  1     2 ms     1 ms     2 ms  10.184.32.13
  2     3 ms     3 ms     2 ms  10.255.107.3
  3     3 ms     2 ms     2 ms  10.119.233.65
  4     *        *        *     Request timed out.
  5     4 ms     6 ms     4 ms  10.119.234.162
  6    11 ms     4 ms     5 ms  ae61-0.del01-96cbe-1a.ntwk.msn.net [104.44.13.22]
  7    26 ms    27 ms    27 ms  104.44.51.53
  8     *        *        *     Request timed out.
  9     *        *      214 ms  104.44.55.65
 10   216 ms   215 ms   215 ms  104.44.31.14
 11   214 ms     *      213 ms  104.44.31.103
 12     *        *        *     Request timed out.
 13     *      215 ms     *     be-3-0.ibr02.got30.ntwk.msn.net [104.44.29.203]
 14   215 ms   214 ms   214 ms  51.10.8.110
 15   216 ms   213 ms   215 ms  be-6-0.ibr03.fra30.ntwk.msn.net [104.44.19.47]
 16     *        *        *     Request timed out.
 17     *      219 ms     *     104.44.50.98
 18  ^C
C:\Users\hp>tracert 20.42.73.28
```



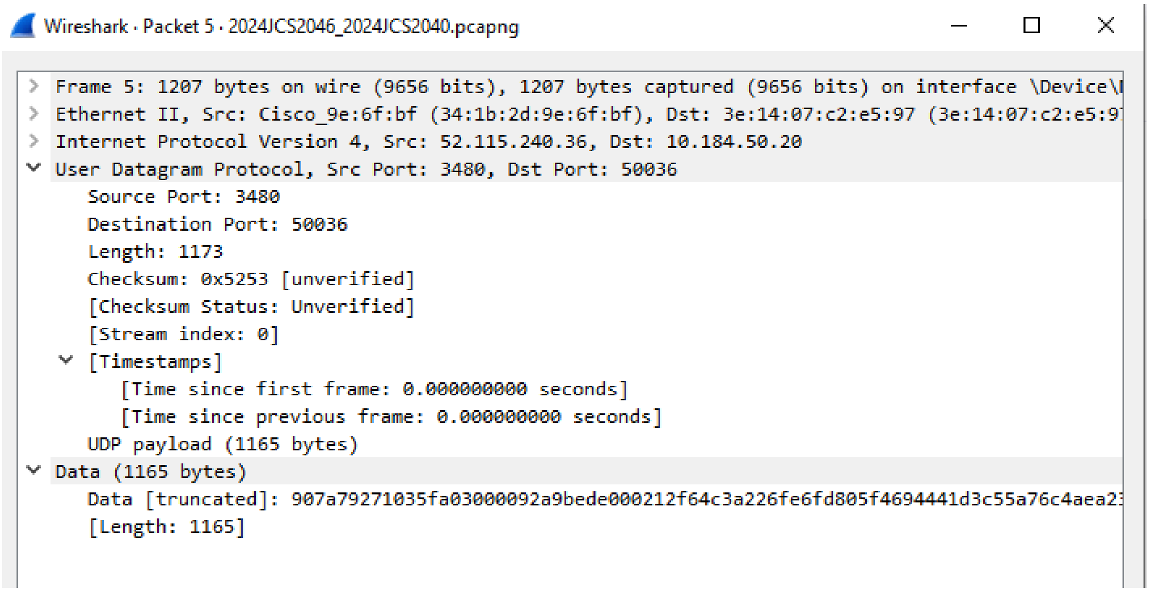
```
C:\Users\hp>tracert 20.42.73.28

Tracing route to 20.42.73.28 over a maximum of 30 hops

  1     3 ms     3 ms     2 ms  10.184.32.13
  2     2 ms     2 ms     2 ms  10.255.107.3
  3     3 ms     1 ms     2 ms  10.119.233.65
  4     *        *        *     Request timed out.
  5     5 ms     4 ms     4 ms  10.119.234.162
  6     4 ms     4 ms     4 ms  ae61-0.del01-96cbe-1a.ntwk.msn.net [104.44.13.22]
  7    27 ms    27 ms    26 ms  104.44.51.53
  8     *      219 ms     *     104.44.53.127
  9     *        *        *     Request timed out.
 10   229 ms   217 ms   219 ms  104.44.31.14
 11   221 ms   219 ms     *     104.44.31.103
 12   232 ms   218 ms   221 ms  104.44.52.19
 13   219 ms     *      220 ms  be-3-0.ibr02.got30.ntwk.msn.net [104.44.29.203]
 14   219 ms   219 ms   218 ms  51.10.8.110
 15   306 ms   219 ms   218 ms  be-6-0.ibr04.bn6.ntwk.msn.net [104.44.29.143]
 16   218 ms   219 ms   219 ms  be-3-0.ibr01.bn6.ntwk.msn.net [104.44.7.177]
 17   223 ms     *        *     be-10-0.ibr03.bl20.ntwk.msn.net [104.44.30.119]
 18   277 ms   218 ms   218 ms  ae146-0.icr04.bl20.ntwk.msn.net [104.44.32.45]
 19     *        *        *     Request timed out.
 20     *        *        *     Request timed out.
 21     *        *        *     Request timed out.
 22     *        *     ^C
```

2(D)

The total number of audio and video packets = 16612. These are the packets using RTP(Real time transport Protocol) which includes audio and video packets. (Below is the explanation on how I found it.)

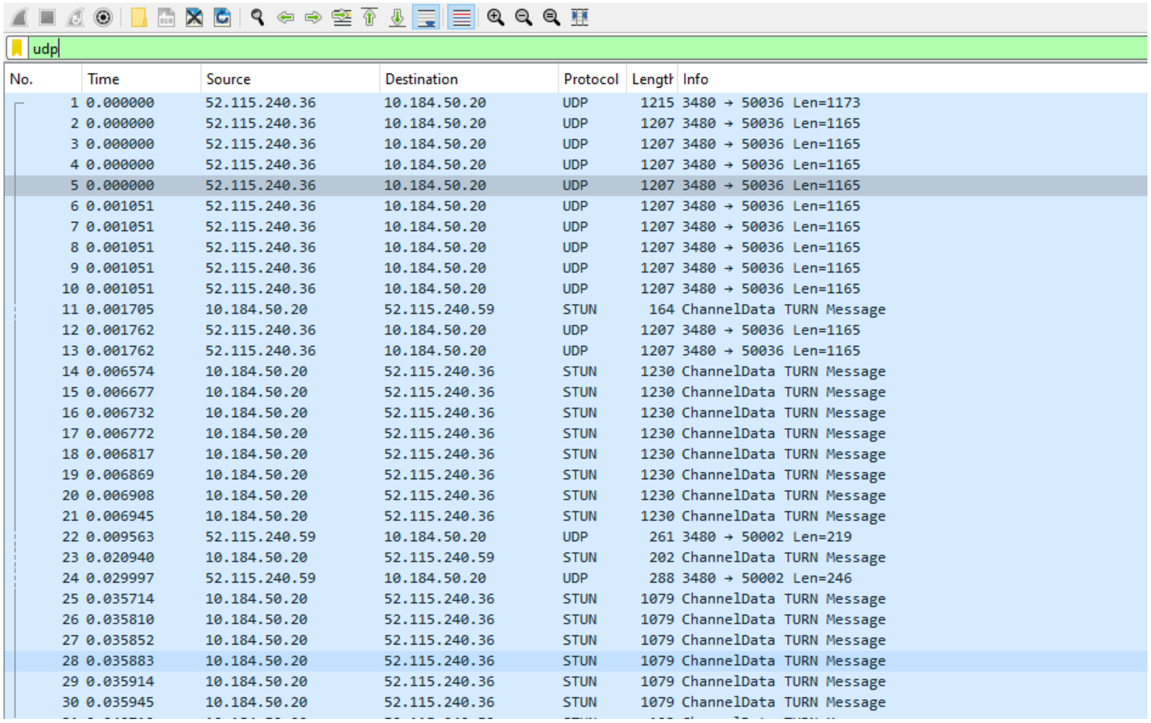
How I filtered RTP packets manually:(Manual decoding was necessary because wireshark didn't automatically recognize the RTP traffic)

- To identify the media streams in the Teams call, I started by filtering the traffic for UDP packets, which are typically used for RTP.
- After filtering, I applied an RTP filter to check for any RTP packets. Initially, no RTP packets were automatically recognized, so I manually inspected the UDP streams and was checking packet details of a UDP packet(screenshot below)



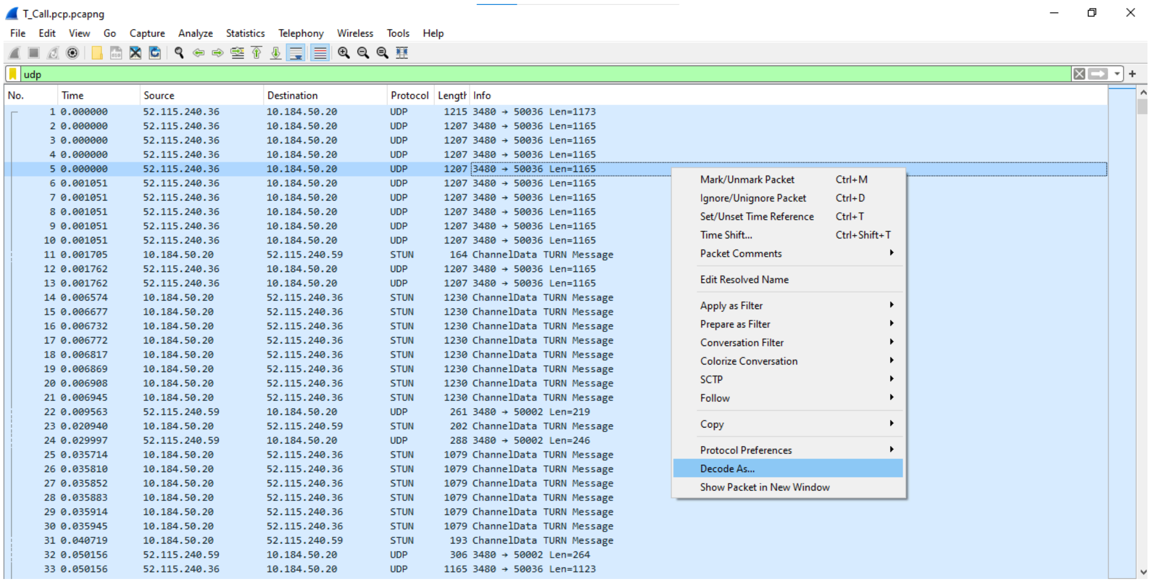
```
> Frame 5: 1207 bytes on wire (9656 bits), 1207 bytes captured (9656 bits) on interface \Device\
> Ethernet II, Src: Cisco_9e:6f:bf (34:1b:2d:9e:6f:bf), Dst: 3e:14:07:c2:e5:97 (3e:14:07:c2:e5:9
> Internet Protocol Version 4, Src: 52.115.240.36, Dst: 10.184.50.20
v User Datagram Protocol, Src Port: 3480, Dst Port: 50036
     Source Port: 3480
     Destination Port: 50036
     Length: 1173
     Checksum: 0x5253 [unverified]
     [Checksum Status: Unverified]
     [Stream index: 0]
   v [Timestamps]
        [Time since first frame: 0.000000000 seconds]
        [Time since previous frame: 0.000000000 seconds]
     UDP payload (1165 bytes)
v Data (1165 bytes)
     Data [truncated]: 907a79271035fa03000092a9bede000212f64c3a226fe6fd805f4694441d3c55a76c4aea2
     [Length: 1165]
```

- Found the following details for a packet which I thought can be a RTP packet because port numbers, src port 3480 falls in the range 1024-65535 which are used by RTP as it dynamically assigns ports. Next the UDP payload=1165 Bytes which can be typical size of RTP packets which carry audio or video packets(which are usually large). The packet also includes valid checksum and a stream index which can be part of a media stream (as media packets/RTP packets contain these factors).
- Based on this I manually decoded the packet as RTP which inturn showed typical RTP headers confirming that the stream was actually RTP. Steps I used to manually decode the udp stream as RTP is as follows:
- In the wireshark window, right click on the packet I identified as RTP and chose the 'Decode As...' option later selected RTP option from all the options that were displayed and applied it. (Attached all the screenshots below)



udp

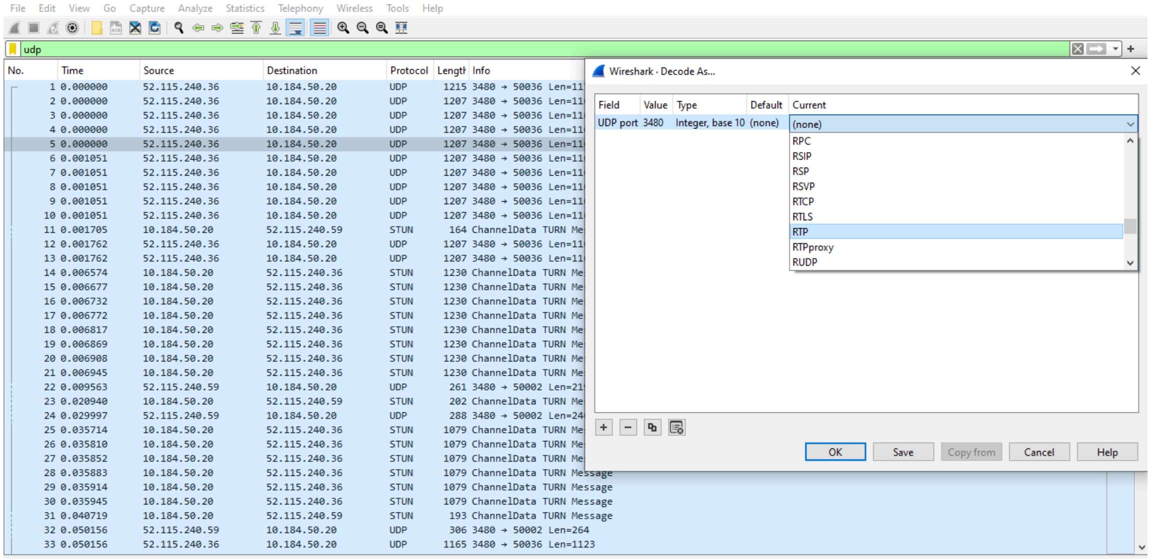
```
No.   Time      Source          Destination     Protocol Length Info
    1 0.000000  52.115.240.36   10.184.50.20    UDP      1215 3480 → 50036 Len=1173
    2 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    3 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    4 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    5 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    6 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    7 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    8 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    9 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
   10 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
   11 0.001705  10.184.50.20    52.115.240.59   STUN      164 ChannelData TURN Message
   12 0.001762  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
   13 0.001762  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
   14 0.006574  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   15 0.006677  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   16 0.006732  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   17 0.006772  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   18 0.006817  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   19 0.006869  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   20 0.006908  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   21 0.006945  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   22 0.009563  52.115.240.59   10.184.50.20    UDP       261 3480 → 50002 Len=219
   23 0.020940  10.184.50.20    52.115.240.59   STUN      202 ChannelData TURN Message
   24 0.029997  52.115.240.59   10.184.50.20    UDP       288 3480 → 50002 Len=246
   25 0.035714  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   26 0.035810  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   27 0.035852  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   28 0.035883  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   29 0.035914  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   30 0.035945  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
```



T_Call.pcp.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

udp

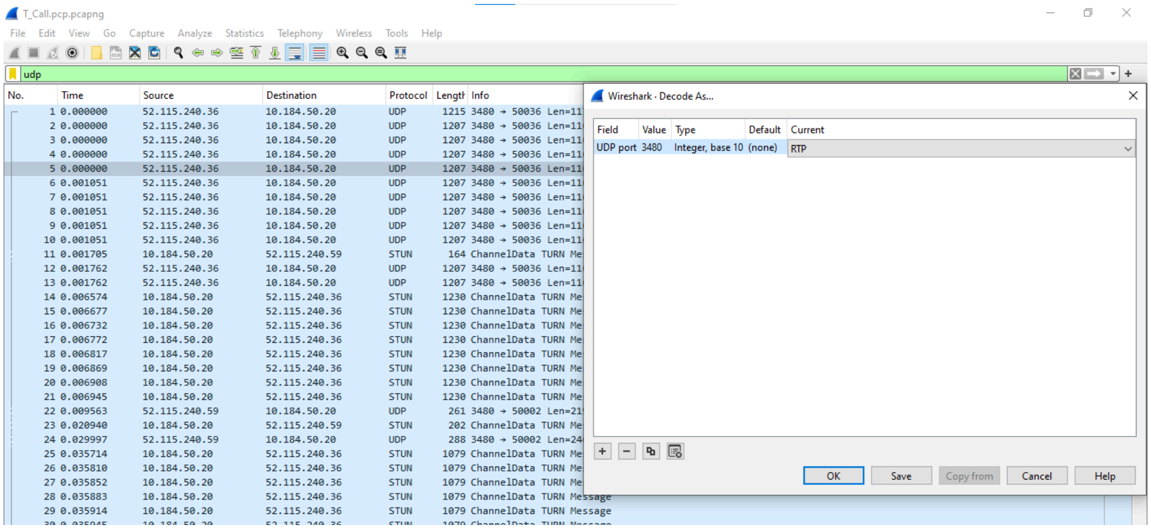
```
No.   Time      Source          Destination     Protocol Length Info
    1 0.000000  52.115.240.36   10.184.50.20    UDP      1215 3480 → 50036 Len=1173
    2 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    3 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    4 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    5 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    6 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    7 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    8 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
    9 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
   10 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
   11 0.001705  10.184.50.20    52.115.240.59   STUN      164 ChannelData TURN Message
   12 0.001762  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
   13 0.001762  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=1165
   14 0.006574  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   15 0.006677  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   16 0.006732  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   17 0.006772  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   18 0.006817  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   19 0.006869  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   20 0.006908  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   21 0.006945  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Message
   22 0.009563  52.115.240.59   10.184.50.20    UDP       261 3480 → 50002 Len=219
   23 0.020940  10.184.50.20    52.115.240.59   STUN      202 ChannelData TURN Message
   24 0.029997  52.115.240.59   10.184.50.20    UDP       288 3480 → 50002 Len=246
   25 0.035714  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   26 0.035810  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   27 0.035852  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   28 0.035883  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   29 0.035914  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   30 0.035945  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   31 0.040719  10.184.50.20    52.115.240.59   STUN      193 ChannelData TURN Message
   32 0.050156  52.115.240.59   10.184.50.20    UDP       306 3480 → 50002 Len=264
   33 0.050156  52.115.240.36   10.184.50.20    UDP      1165 3480 → 50036 Len=1123
```


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Wireshark · Decode As...
RPC
RSIP
RSP
RSVP
RTCP
RTLS
RTP
RTPproxy
RUDP
+ − OK Save Copy from Cancel Help

udp

```
No.   Time      Source          Destination     Protocol Length Info
    1 0.000000  52.115.240.36   10.184.50.20    UDP      1215 3480 → 50036 Len=11
    2 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    3 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    4 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    5 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    6 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    7 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    8 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    9 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
   10 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
   11 0.001705  10.184.50.20    52.115.240.59   STUN      164 ChannelData TURN Me
   12 0.001762  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
   13 0.001762  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
   14 0.006574  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   15 0.006677  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   16 0.006732  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   17 0.006772  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   18 0.006817  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   19 0.006869  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   20 0.006908  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   21 0.006945  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   22 0.009563  52.115.240.59   10.184.50.20    UDP       261 3480 → 50002 Len=21
   23 0.020940  10.184.50.20    52.115.240.59   STUN      202 ChannelData TURN Me
   24 0.029997  52.115.240.59   10.184.50.20    UDP       288 3480 → 50002 Len=24
   25 0.035714  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Me
   26 0.035810  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Me
   27 0.035852  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Me
   28 0.035883  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   29 0.035914  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   30 0.035945  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   31 0.040719  10.184.50.20    52.115.240.59   STUN      193 ChannelData TURN Message
   32 0.050156  52.115.240.59   10.184.50.20    UDP       306 3480 → 50002 Len=264
   33 0.050156  52.115.240.36   10.184.50.20    UDP      1165 3480 → 50036 Len=1123
```

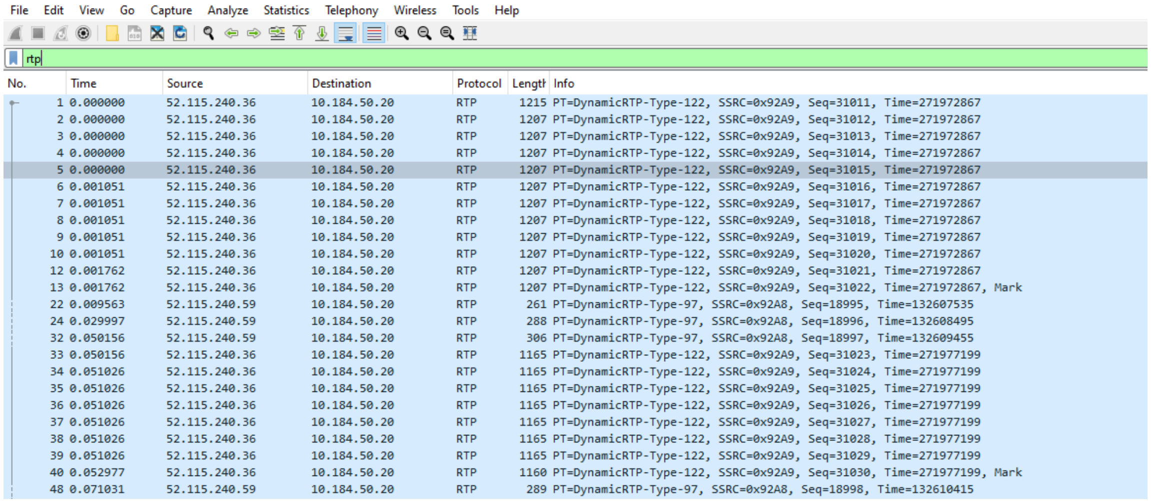
Field	Value	Type	Default	Current
UDP port	3480	Integer, base 10	(none)	(none)


T_Call.pcp.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Wireshark · Decode As...
+ − OK Save Copy from Cancel Help

udp

```
No.   Time      Source          Destination     Protocol Length Info
    1 0.000000  52.115.240.36   10.184.50.20    UDP      1215 3480 → 50036 Len=11
    2 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    3 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    4 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    5 0.000000  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    6 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    7 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    8 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
    9 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
   10 0.001051  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
   11 0.001705  10.184.50.20    52.115.240.59   STUN      164 ChannelData TURN Me
   12 0.001762  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
   13 0.001762  52.115.240.36   10.184.50.20    UDP      1207 3480 → 50036 Len=11
   14 0.006574  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   15 0.006677  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   16 0.006732  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   17 0.006772  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   18 0.006817  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   19 0.006869  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   20 0.006908  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   21 0.006945  10.184.50.20    52.115.240.36   STUN     1230 ChannelData TURN Me
   22 0.009563  52.115.240.59   10.184.50.20    UDP       261 3480 → 50002 Len=21
   23 0.020940  10.184.50.20    52.115.240.59   STUN      202 ChannelData TURN Me
   24 0.029997  52.115.240.59   10.184.50.20    UDP       288 3480 → 50002 Len=24
   25 0.035714  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Me
   26 0.035810  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Me
   27 0.035852  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Me
   28 0.035883  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   29 0.035914  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
   30 0.035945  10.184.50.20    52.115.240.36   STUN     1079 ChannelData TURN Message
```

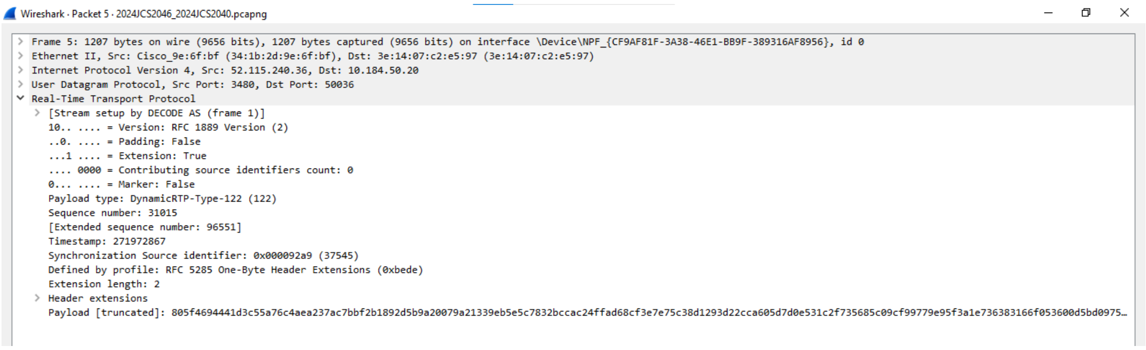
Field	Value	Type	Default	Current
UDP port	3480	Integer, base 10	(none)	RTP


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

rtp

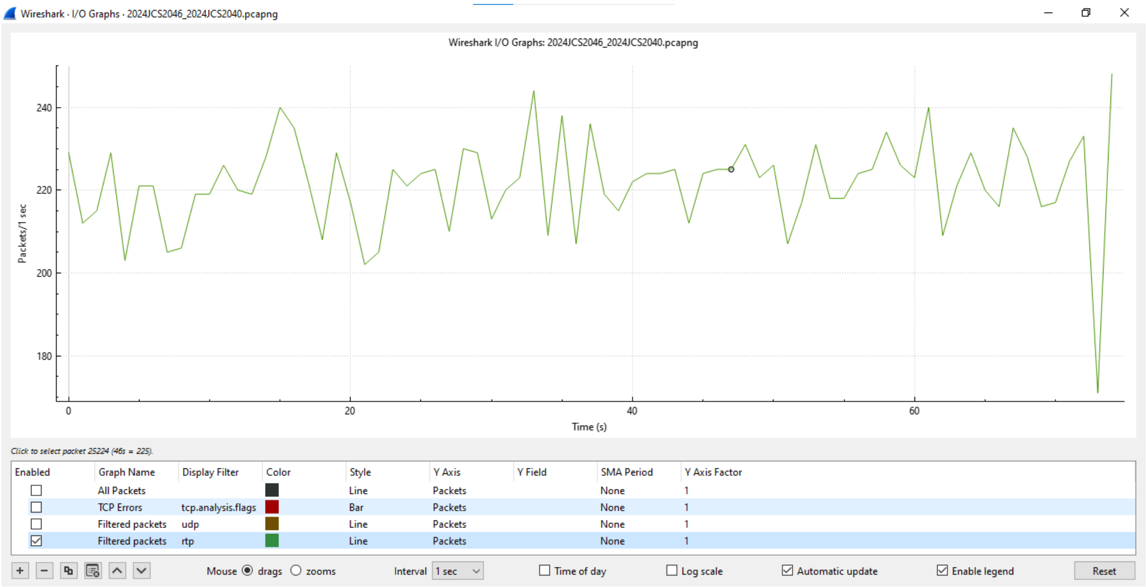
```
No.   Time      Source          Destination     Protocol Length Info
    1 0.000000  52.115.240.36   10.184.50.20    RTP      1215 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31011, Time=271972867
    2 0.000000  52.115.240.36   10.184.50.20    RTP      1207 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31012, Time=271972867
    3 0.000000  52.115.240.36   10.184.50.20    RTP      1207 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31013, Time=271972867
    4 0.000000  52.115.240.36   10.184.50.20    RTP      1207 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31014, Time=271972867
    5 0.000000  52.115.240.36   10.184.50.20    RTP      1207 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31015, Time=271972867
    6 0.001051  52.115.240.36   10.184.50.20    RTP      1207 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31016, Time=271972867
    7 0.001051  52.115.240.36   10.184.50.20    RTP      1207 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31017, Time=271972867
    8 0.001051  52.115.240.36   10.184.50.20    RTP      1207 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31018, Time=271972867
    9 0.001051  52.115.240.36   10.184.50.20    RTP      1207 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31019, Time=271972867
   10 0.001051  52.115.240.36   10.184.50.20    RTP      1207 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31020, Time=271972867
   12 0.001762  52.115.240.36   10.184.50.20    RTP      1207 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31021, Time=271972867
   13 0.001762  52.115.240.36   10.184.50.20    RTP      1207 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31022, Time=271972867, Mark
   22 0.009563  52.115.240.59   10.184.50.20    RTP       261 PT=DynamicRTP-Type-97, SSRC=0x92A8, Seq=18995, Time=132607535
   24 0.029997  52.115.240.59   10.184.50.20    RTP       288 PT=DynamicRTP-Type-97, SSRC=0x92A8, Seq=18996, Time=132608495
   32 0.050156  52.115.240.59   10.184.50.20    RTP       306 PT=DynamicRTP-Type-97, SSRC=0x92A8, Seq=18997, Time=132609455
   33 0.050156  52.115.240.36   10.184.50.20    RTP      1165 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31023, Time=271977199
   34 0.051026  52.115.240.36   10.184.50.20    RTP      1165 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31024, Time=271977199
   35 0.051026  52.115.240.36   10.184.50.20    RTP      1165 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31025, Time=271977199
   36 0.051026  52.115.240.36   10.184.50.20    RTP      1165 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31026, Time=271977199
   37 0.051026  52.115.240.36   10.184.50.20    RTP      1165 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31027, Time=271977199
   38 0.051026  52.115.240.36   10.184.50.20    RTP      1165 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31028, Time=271977199
   39 0.051026  52.115.240.36   10.184.50.20    RTP      1165 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31029, Time=271977199
   40 0.052977  52.115.240.36   10.184.50.20    RTP      1160 PT=DynamicRTP-Type-122, SSRC=0x92A9, Seq=31030, Time=271977199, Mark
   48 0.071031  52.115.240.59   10.184.50.20    RTP       289 PT=DynamicRTP-Type-97, SSRC=0x92A8, Seq=18998, Time=132610415
```

- If the packet is successfully decoded as RTP, you will see RTP headers in the packet details.Version: Should be 2 for RTP.Payload Type: Identifies the codec used (audio or video).Sequence Number and Timestamp.
- These helped me identify the packet as RTP.This allowed me to pinpoint the media traffic during the call.I was able to decode the identified UDP streams as RTP and successfully identify the RTP headers.
- (Packet details which I decoded as RTP after decoding it as RTP)


Wireshark · Packet 5 · 2024JCS2046_2024JCS2040.pcapng

```
> Frame 5: 1207 bytes on wire (9656 bits), 1207 bytes captured (9656 bits) on interface \Device\NPF_{CF9AF81F-3A38-46E1-BB9F-389316AF8956}, id 0
> Ethernet II, Src: Cisco_9e:6f:bf (34:1b:2d:9e:6f:bf), Dst: 3e:14:07:c2:e5:97 (3e:14:07:c2:e5:97)
> Internet Protocol Version 4, Src: 52.115.240.36, Dst: 10.184.50.20
> User Datagram Protocol, Src Port: 3480, Dst Port: 50036
v Real-Time Transport Protocol
  > [Stream setup by DECODE AS (frame 1)]
    10.. .... = Version: RFC 1889 Version (2)
    ..0. .... = Padding: False
    ...1 .... = Extension: True
    .... 0000 = Contributing source identifiers count: 0
    0... .... = Marker: False
    Payload type: DynamicRTP-Type-122 (122)
    Sequence number: 31015
    [Extended sequence number: 96551]
    Timestamp: 271972867
    Synchronization Source identifier: 0x000092a9 (37545)
    Defined by profile: RFC 5285 One-Byte Header Extensions (0xbede)
    Extension length: 2
  > Header extensions
    Payload [truncated]: 805f4694441d3c55a76c4aea237ac7bbf2b1892d5b9a20079a21339eb5e5c7832bccac24ffad68cf3e7e75c38d1293d22cca605d7d0e531c2f735685c09cf99779e95f3a1e736383166f053600d5bd0975…
```

- Using Wire shark filters to plot a time-series diagram showing the bandwidth utilization by the two media types:On the main window of wireshark go to Statistics -> IO Graphs
- Set the display filter to show RTP traffic by selecting ‘rtp’ option in the Y-axis


Wireshark · I/O Graphs · 2024JCS2046_2024JCS2040.pcapng
Wireshark I/O Graphs: 2024JCS2046_2024JCS2040.pcapng
Packets/1 sec
240
220
200
180
0
20
40
60
Time (s)
+
−
Mouse
Interval
Reset

Click to select packet 25224 (46s = 225).

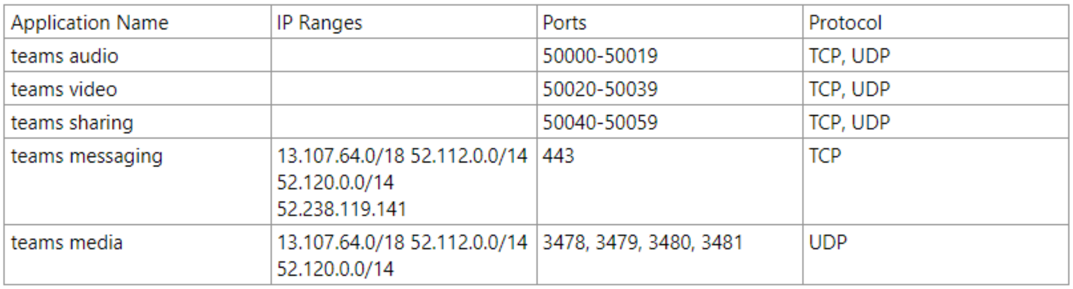
Enabled	Graph Name	Display Filter	Color	Style	Y Axis	Y Field	SMA Period	Y Axis Factor
□	All Packets		⬛	Line	Packets		None	1
□	TCP Errors	tcp.analysis.flags	🟥	Bar	Packets		None	1
□	Filtered packets	udp	🟫	Line	Packets		None	1
☑	Filtered packets	rtp	🟩	Line	Packets		None	1

Mouse ● drags ○ zooms   Interval 1 sec   □ Time of day   □ Log scale   ☑ Automatic update   ☑ Enable legend

Separating audio and video packets using port numbers of RTP packets:

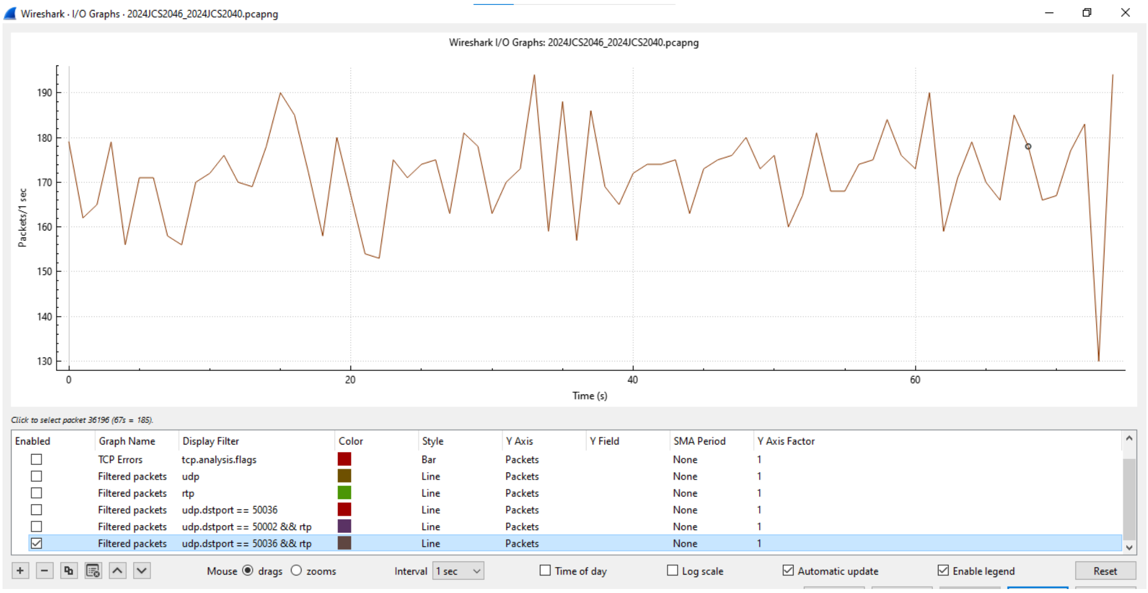
- Found that all the packets after applying the filter ‘rtp’ falling into any 2 of the following categories:1)User Datagram Protocol, Src Port: 3480, Dst Port: 50036 2)User Datagram Protocol, Src Port: 3480, Dst Port: 50002

- Using the microsoft teams forum, where I found this(screenshot below) I was able to filter out the packets using destination port numbers



Application Name	IP Ranges	Ports	Protocol
teams audio		50000-50019	TCP, UDP
teams video		50020-50039	TCP, UDP
teams sharing		50040-50059	TCP, UDP
teams messaging	13.107.64.0/18 52.112.0.0/14 52.120.0.0/14 52.238.119.141	443	TCP
teams media	13.107.64.0/18 52.112.0.0/14 52.120.0.0/14	3478, 3479, 3480, 3481	UDP

- Applied filter ‘udp.dstport == 50036 && rtp’ which displayed **12884 packets(32.4%)** which are **video packets** and then chose IO graphs option in ‘Statistics’ and applied the same filter in Y-axis to get the graph below:

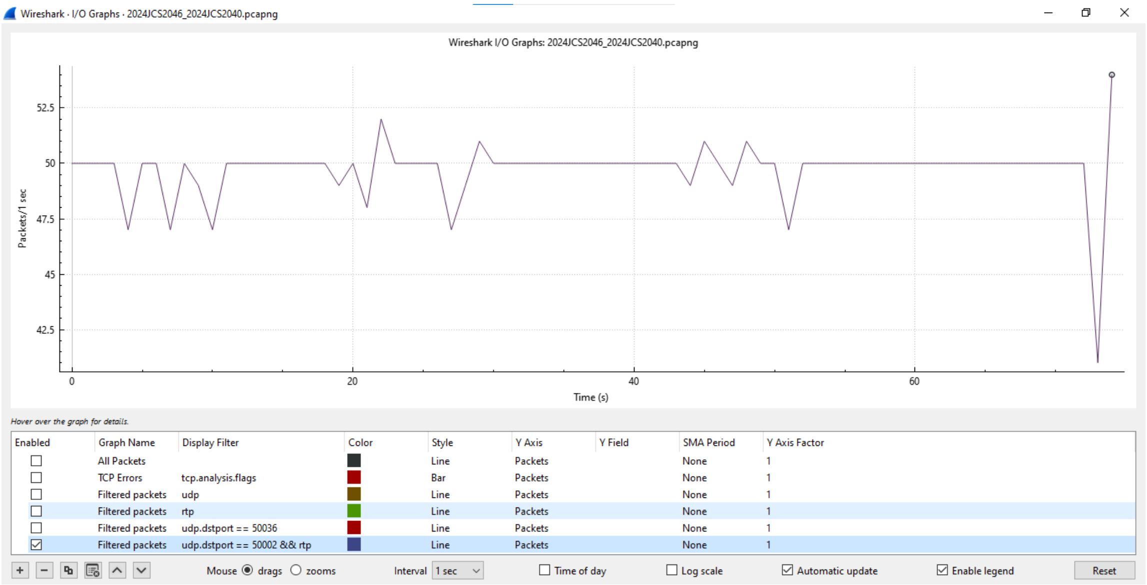


Wireshark · I/O Graphs · 2024JCS2046_2024JCS2040.pcapng
Wireshark I/O Graphs: 2024JCS2046_2024JCS2040.pcapng
Packets/1 sec
190
180
170
160
150
140
130
0
20
40
60
Time (s)

Click to select packet 36196 (67s = 185).

Enabled	Graph Name	Display Filter	Color	Style	Y Axis	Y Field	SMA Period	Y Axis Factor
☐	TCP Errors	tcp.analysis.flags	🟥	Bar	Packets		None	1
☐	Filtered packets	udp	🟫	Line	Packets		None	1
☐	Filtered packets	rtp	🟩	Line	Packets		None	1
☐	Filtered packets	udp.dstport == 50036	🟥	Line	Packets		None	1
☐	Filtered packets	udp.dstport == 50002 && rtp	🟪	Line	Packets		None	1
☑	Filtered packets	udp.dstport == 50036 && rtp	🟫	Line	Packets		None	1

- Applied filter ‘udp.dstport == 50002 && rtp’ which displayed **3728 packets(9.4%)** which are **audio packets** and then chose IO graphs option in ‘Statistics’ and applied the same filter in Y-axis to get the graph below:


Wireshark · I/O Graphs · 2024JCS2046_2024JCS2040.pcapng
Wireshark I/O Graphs: 2024JCS2046_2024JCS2040.pcapng
Packets/1 sec
52.5
50
47.5
45
42.5
0
20
40
60
Time (s)

Hover over the graph for details.

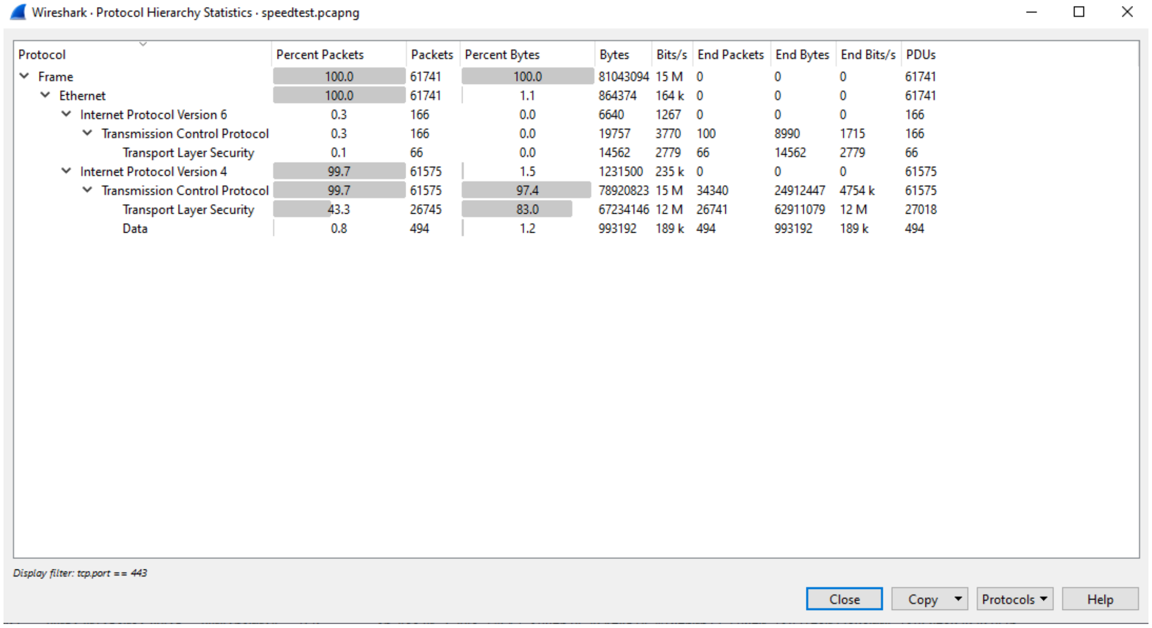
Enabled	Graph Name	Display Filter	Color	Style	Y Axis	Y Field	SMA Period	Y Axis Factor
☐	All Packets		■	Line	Packets		None	1
☐	TCP Errors	tcp.analysis.flags	■	Bar	Packets		None	1
☐	Filtered packets	udp	■	Line	Packets		None	1
☐	Filtered packets	rtp	■	Line	Packets		None	1
☐	Filtered packets	udp.dstport == 50036	■	Line	Packets		None	1
☑	Filtered packets	udp.dstport == 50002 && rtp	■	Line	Packets		None	1

Mouse ● drags ○ zooms Interval 1 sec ☐ Time of day ☐ Log scale ☑ Automatic update ☑ Enable legend

3(A)

To separate the speed test traffic from the rest:

1. Applied filter tcp.port=443 which filters the traffic over HTTPS, Protocol hierarchy after filter:

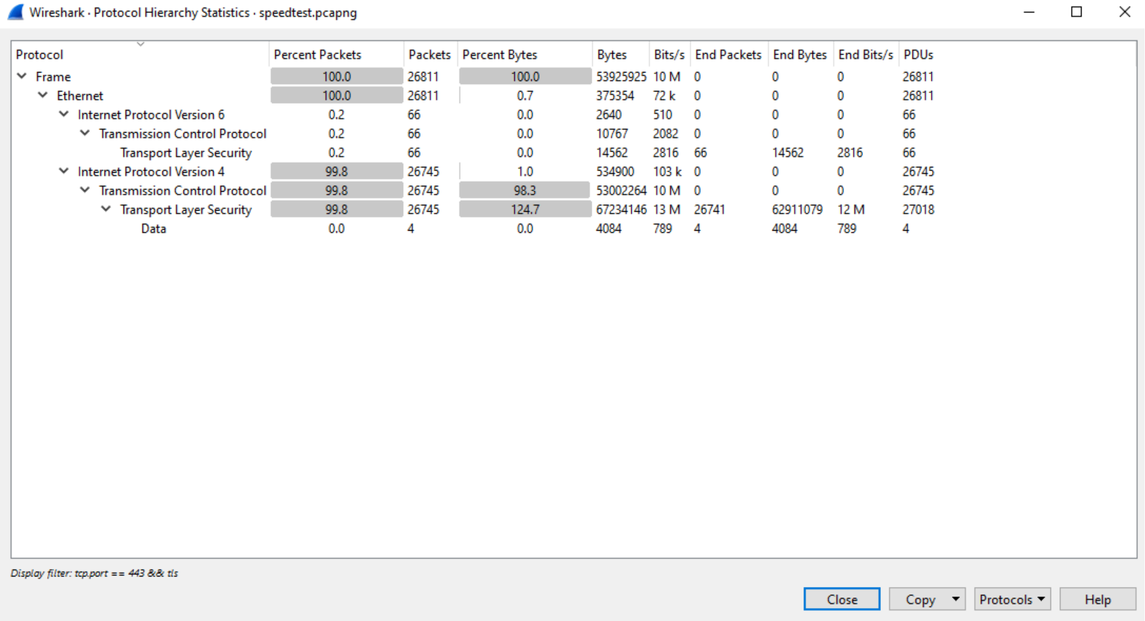

Wireshark · Protocol Hierarchy Statistics · speedtest.pcapng

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s	PDUs
Frame	100.0	61741	100.0	81043094	15 M	0	0	0	61741
Ethernet	100.0	61741	1.1	864374	164 k	0	0	0	61741
Internet Protocol Version 6	0.3	166	0.0	6640	1267	0	0	0	166
Transmission Control Protocol	0.3	166	0.0	19757	3770	100	8990	1715	166
Transport Layer Security	0.1	66	0.0	14562	2779	66	14562	2779	66
Internet Protocol Version 4	99.7	61575	1.5	1231500	235 k	0	0	0	61575
Transmission Control Protocol	99.7	61575	97.4	78920823	15 M	34340	24912447	4754 k	61575
Transport Layer Security	43.3	26745	83.0	67234146	12 M	26741	62911079	12 M	27018
Data	0.8	494	1.2	993192	189 k	494	993192	189 k	494

Display filter: tcp.port == 443

2. After filtering found that, Data under Transport Layer Security(TLS) could be the actual speed test data. So applied filter 'tcp.port=443 && tls' which gave the following protocol hierarchy and conversations as follows:The total data sent after filtering is 26811 packets with 53925925 Bytes and Total data captured is 71113 packets with total data of

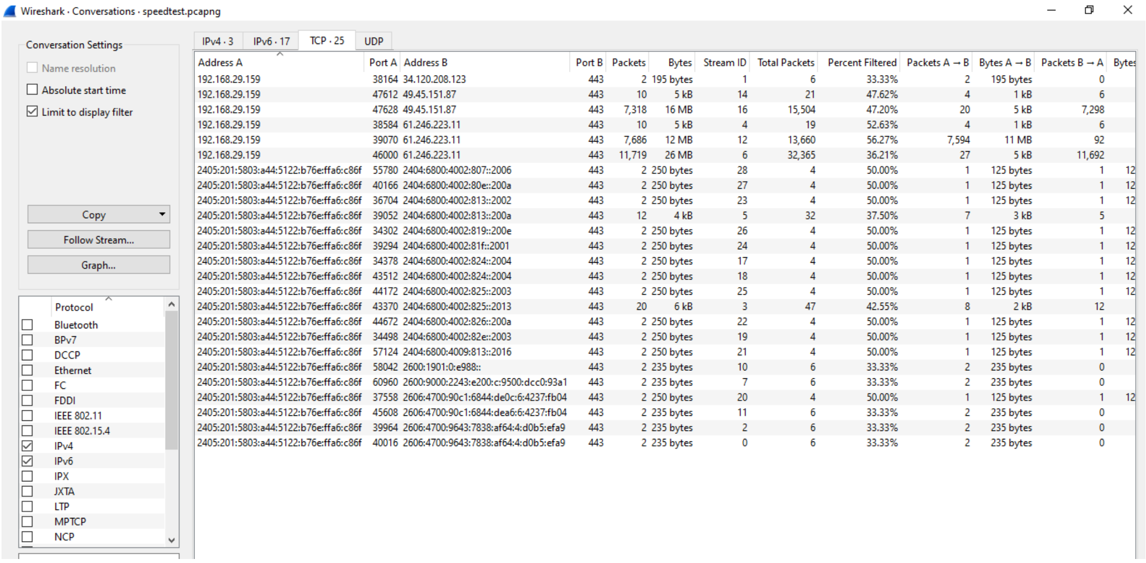
91623122 Bytes. So the **percentage of speed traffic=66.54%**



Wireshark · Protocol Hierarchy Statistics · speedtest.pcapng

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s	PDUs
Frame	100.0	26811	100.0	53925925	10 M	0	0	0	26811
Ethernet	100.0	26811	0.7	375354	72 k	0	0	0	26811
Internet Protocol Version 6	0.2	66	0.0	2640	510	0	0	0	66
Transmission Control Protocol	0.2	66	0.0	10767	2082	0	0	0	66
Transport Layer Security	0.2	66	0.0	14562	2816	66	14562	2816	66
Internet Protocol Version 4	99.8	26745	1.0	534900	103 k	0	0	0	26745
Transmission Control Protocol	99.8	26745	98.3	53002264	10 M	0	0	0	26745
Transport Layer Security	99.8	26745	124.7	67234146	13 M	26741	62911079	12 M	27018
Data	0.0	4	0.0	4084	789	4	4084	789	4

Display filter: tcp.port == 443 && tls



Wireshark · Conversations · speedtest.pcapng

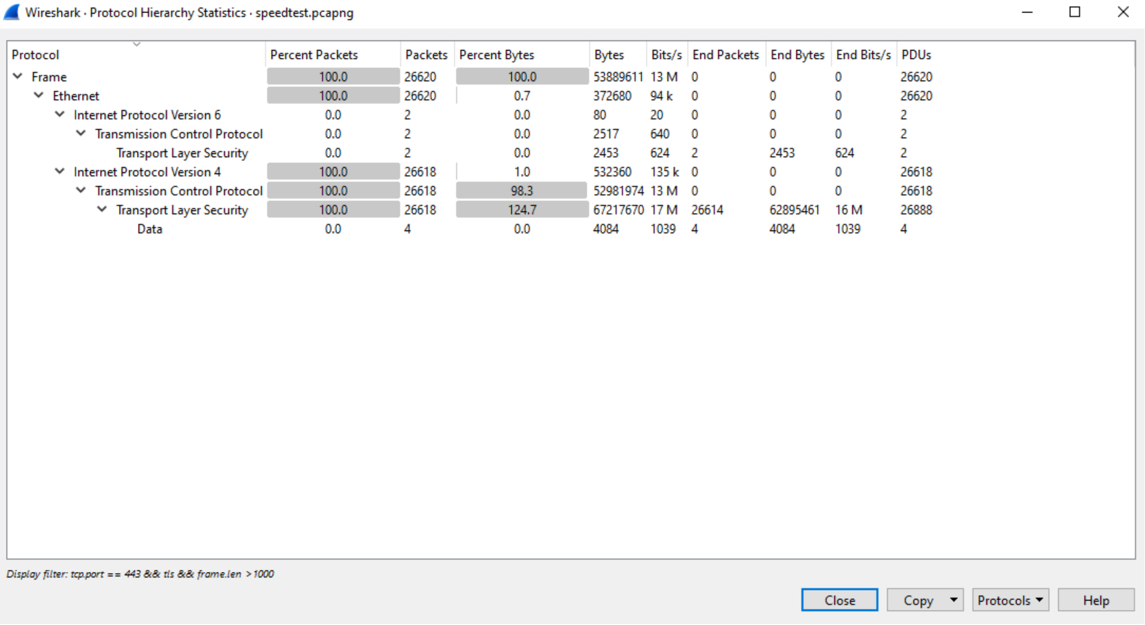
Conversation Settings

□ Name resolution
□ Absolute start time
☑ Limit to display filter

	Protocol
□	Bluetooth
□	BPv7
□	DCCP
□	Ethernet
□	FC
□	FDDI
□	IEEE 802.11
□	IEEE 802.15.4
☑	IPv4
☑	IPv6
□	IPX
□	JXTA
□	LTP
□	MPTCP
□	NCP

Address A	Port A	Address B	Port B	Packets	Bytes	Stream ID	Total Packets	Percent Filtered	Packets A → B	Bytes A → B	Packets B → A	Bytes
192.168.29.159	38164	34.120.208.123	443	2	195 bytes	1	6	33.33%	2	195 bytes	0	
192.168.29.159	47612	49.45.151.87	443	10	5 kB	14	21	47.62%	4	1 kB	6	
192.168.29.159	47628	49.45.151.87	443	7,318	16 MB	16	15,504	47.20%	20	5 kB	7,298	
192.168.29.159	38584	61.246.223.11	443	10	5 kB	4	19	52.63%	4	1 kB	6	
192.168.29.159	39070	61.246.223.11	443	7,686	12 MB	12	13,660	56.27%	7,594	11 MB	92	
192.168.29.159	46000	61.246.223.11	443	11,719	26 MB	6	32,365	36.21%	27	5 kB	11,692	
2405:201:5803:a44:5122:b76e:ffa6:c86f	55780	2404:6800:4002:807::2006	443	2	250 bytes	28	4	50.00%	1	125 bytes	1	12
2405:201:5803:a44:5122:b76e:ffa6:c86f	40166	2404:6800:4002:80e::200a	443	2	250 bytes	27	4	50.00%	1	125 bytes	1	12
2405:201:5803:a44:5122:b76e:ffa6:c86f	36704	2404:6800:4002:813::2002	443	2	250 bytes	23	4	50.00%	1	125 bytes	1	12
2405:201:5803:a44:5122:b76e:ffa6:c86f	39052	2404:6800:4002:813::200a	443	12	4 kB	5	32	37.50%	7	3 kB	5	
2405:201:5803:a44:5122:b76e:ffa6:c86f	34302	2404:6800:4002:819::200e	443	2	250 bytes	26	4	50.00%	1	125 bytes	1	12
2405:201:5803:a44:5122:b76e:ffa6:c86f	39294	2404:6800:4002:81f::2001	443	2	250 bytes	24	4	50.00%	1	125 bytes	1	12
2405:201:5803:a44:5122:b76e:ffa6:c86f	34378	2404:6800:4002:824::2004	443	2	250 bytes	17	4	50.00%	1	125 bytes	1	12
2405:201:5803:a44:5122:b76e:ffa6:c86f	43512	2404:6800:4002:824::2004	443	2	250 bytes	18	4	50.00%	1	125 bytes	1	12
2405:201:5803:a44:5122:b76e:ffa6:c86f	44172	2404:6800:4002:825::2003	443	2	250 bytes	25	4	50.00%	1	125 bytes	1	12
2405:201:5803:a44:5122:b76e:ffa6:c86f	43370	2404:6800:4002:825::2013	443	20	6 kB	3	47	42.55%	8	2 kB	12	
2405:201:5803:a44:5122:b76e:ffa6:c86f	44672	2404:6800:4002:826::200a	443	2	250 bytes	22	4	50.00%	1	125 bytes	1	12
2405:201:5803:a44:5122:b76e:ffa6:c86f	34498	2404:6800:4002:82e::2003	443	2	250 bytes	19	4	50.00%	1	125 bytes	1	12
2405:201:5803:a44:5122:b76e:ffa6:c86f	57124	2404:6800:4009:813::2016	443	2	250 bytes	21	4	50.00%	1	125 bytes	1	12
2405:201:5803:a44:5122:b76e:ffa6:c86f	58042	2600:1901:0:e988::	443	2	235 bytes	10	6	33.33%	2	235 bytes	0	
2405:201:5803:a44:5122:b76e:ffa6:c86f	60960	2600:9000:2243:e200:c:9500:dcc0:93a1	443	2	235 bytes	7	6	33.33%	2	235 bytes	0	
2405:201:5803:a44:5122:b76e:ffa6:c86f	37558	2606:4700:90c1:6844:de0c:6:4237:fb04	443	2	250 bytes	20	4	50.00%	1	125 bytes	1	12
2405:201:5803:a44:5122:b76e:ffa6:c86f	45608	2606:4700:90c1:6844:dea6:6:4237:fb04	443	2	235 bytes	11	6	33.33%	2	235 bytes	0	
2405:201:5803:a44:5122:b76e:ffa6:c86f	39964	2606:4700:9643:7838:af64:4:d0b5:efa9	443	2	235 bytes	2	6	33.33%	2	235 bytes	0	
2405:201:5803:a44:5122:b76e:ffa6:c86f	40016	2606:4700:9643:7838:af64:4:d0b5:efa9	443	2	235 bytes	0	6	33.33%	2	235 bytes	0	

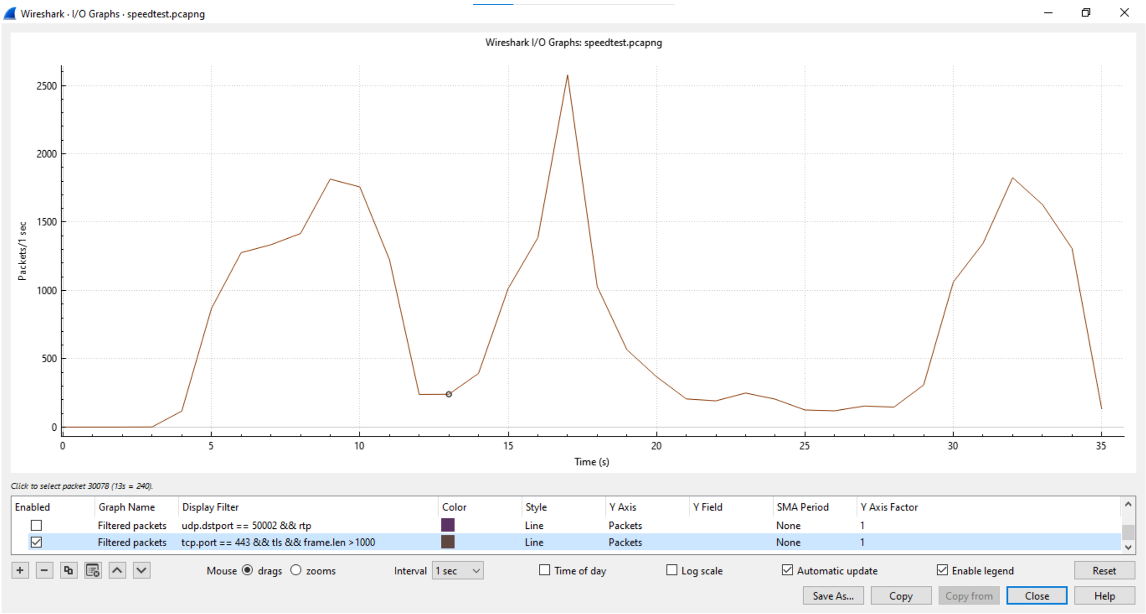
3. Also I found that there are variations in the data that is transferred. Since speed tests involve transferring large amount of data, I also applied filter to get packets larger than 1000 Bytes.So i applied filter ‘‘tcp.port=443 && tls && frame.len>1000’ which gave the following protocol hierarchy:so the data sent now is 26620 packets with 53889611 Bytes.So after the applying this filter percentage of speed test data=58.82%The IO graph for this filter is below:



Wireshark · Protocol Hierarchy Statistics · speedtest.pcapng

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s	PDUs
Frame	100.0	26620	100.0	53889611	13 M	0	0	0	26620
Ethernet	100.0	26620	0.7	372680	94 k	0	0	0	26620
Internet Protocol Version 6	0.0	2	0.0	80	20	0	0	0	2
Transmission Control Protocol	0.0	2	0.0	2517	640	0	0	0	2
Transport Layer Security	0.0	2	0.0	2453	624	2	2453	624	2
Internet Protocol Version 4	100.0	26618	1.0	532360	135 k	0	0	0	26618
Transmission Control Protocol	100.0	26618	98.3	52981974	13 M	0	0	0	26618
Transport Layer Security	100.0	26618	124.7	67217670	17 M	26614	62895461	16 M	26888
Data	0.0	4	0.0	4084	1039	4	4084	1039	4

Display filter: tcp.port == 443 && tls && frame.len > 1000



Wireshark · I/O Graphs · speedtest.pcapng

Click to select packet 30078 (13s = 240).

Enabled	Graph Name	Display Filter	Color	Style	Y Axis	Y Field	SMA Period	Y Axis Factor
☐	Filtered packets	udp.dstport == 50002 && rtp	■	Line	Packets		None	1
☑	Filtered packets	tcp.port == 443 && tls && frame.len > 1000	■	Line	Packets		None	1

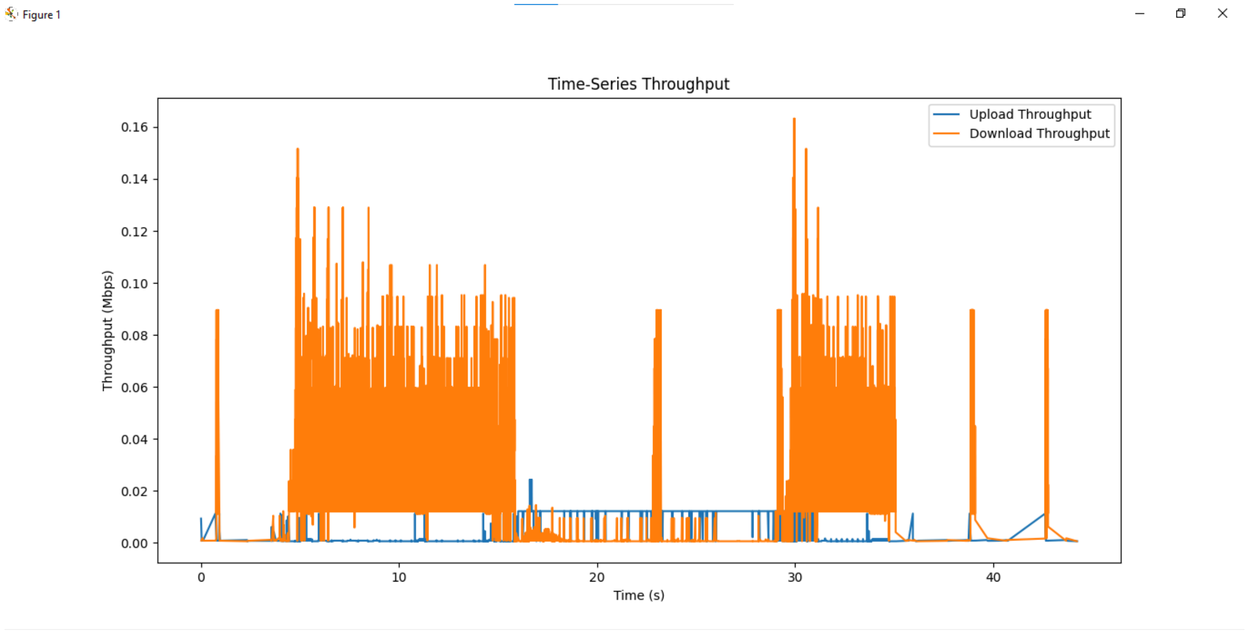
Mouse ● drags ○ zooms Interval 1 sec ☐ Time of day ☐ Log scale ☑ Automatic update ☑ Enable legend

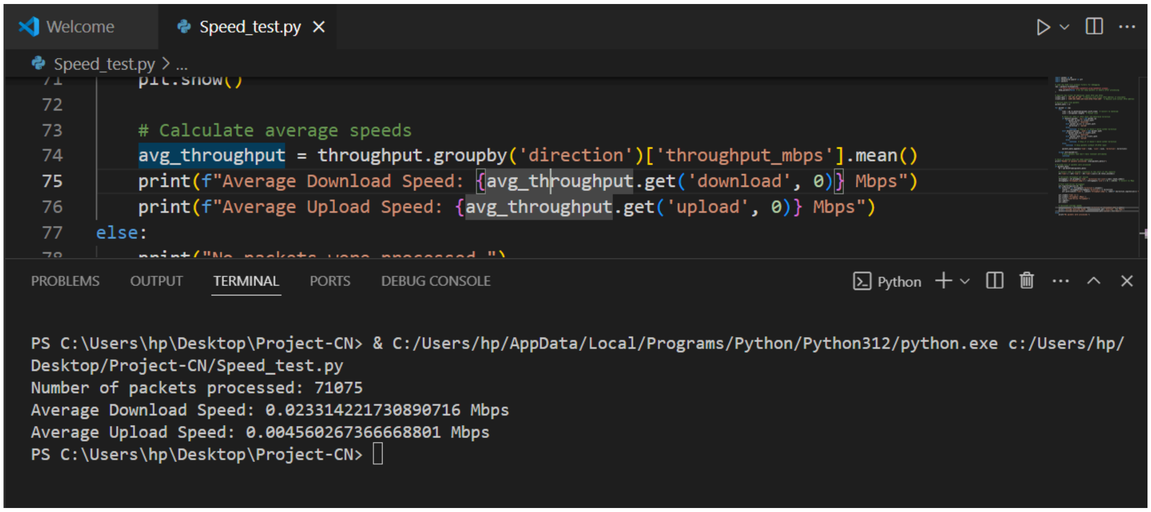
3(B),3(C)

Logic for obtaining throughput and average download and upload speeds:

1. Loading pcap file: Used ‘pyshark’ library to inspect each packet in the file and also used ‘keep_packets=False’ to manage memory usage by not putting packets in memory after processing.
2. From the speed_test.pcap conversations, I found that the following ipv4 and ipv6 are corresponding to client ‘client_ipv4 =’192.168.29.159’ client_ipv6=’2405:201:5803:a44:5122:b76e:ffa6:c86f’.This is done to distinguish between uplink (upload) and downlink(download) traffic.

3. Wrote a loop to iterate through each packet in PCAP file and checks if packet is using IPv4 or IPv6 based on source and destination IP addresses. Download traffic implies destination IP matches client IP and Upload traffic implies source IP matches with client IP.
4. Then converted the collected packets into a Pandas dataframe for easier analysis.
5. Calculating throughput per second. Here the script groups the data by time(in seconds) and direction. For each group, sums the packets sizes to calculate the total data transferred in that second. Then converted data size into Mbps(multiply by 8) divide by 10^6 to get in Mbps.
6. Next step is to plot the time-series throughput, Each upload and download throughput over time is plotted
7. Finally, calculating the average throughput for both download and upload direction.
8. Time series plot that I got and output of the throughputs are attached below:





```python
        plt.show()

        # Calculate average speeds
        avg_throughput = throughput.groupby('direction')['throughput_mbps'].mean()
        print(f"Average Download Speed: {avg_throughput.get('download', 0)} Mbps")
        print(f"Average Upload Speed: {avg_throughput.get('upload', 0)} Mbps")
    else:
        print("No packets were processed.")
```

```
PS C:\Users\hp\Desktop\Project-CN> & C:/Users/hp/AppData/Local/Programs/Python/Python312/python.exe c:/Users/hp/
Desktop/Project-CN/Speed_test.py
Number of packets processed: 71075
Average Download Speed: 0.023314221730890716 Mbps
Average Upload Speed: 0.004560267366668801 Mbps
PS C:\Users\hp\Desktop\Project-CN> 
```