

Assignment 1: Getting to Know Network Traffic

Arpit Prasad
COL334: Computer Network

August 22, 2025

1 Measurement Tools

1.1 ping

```
● arpit@arpit-linux:~/Desktop/iitd/sem_7/COL334/projects/Getting-To-Know-Network-Traffic$ ping -c 10 www.google.com
PING www.google.com (2404:6800:4002:830::2004) 56 data bytes
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=1 ttl=115 time=5.30 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=2 ttl=115 time=7.01 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=3 ttl=115 time=7.64 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=4 ttl=115 time=5.53 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=5 ttl=115 time=6.76 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=6 ttl=115 time=6.03 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=7 ttl=115 time=7.90 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=8 ttl=115 time=10.6 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=9 ttl=115 time=5.15 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=10 ttl=115 time=6.42 ms

--- www.google.com ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9014ms
rtt min/avg/max/mdev = 5.145/6.832/10.578/1.535 ms
```

Figure 1: Ten Pings to www.google.com

```
● arpit@arpit-linux:~/Desktop/iitd/sem_7/COL334/projects/Getting-To-Know-Network-Traffic$ ping -c 10 craigslist.com
PING craigslist.com (208.82.238.135) 56(84) bytes of data.
64 bytes from nonorg.craigslist.org (208.82.238.135): icmp_seq=1 ttl=48 time=264 ms
64 bytes from nonorg.craigslist.org (208.82.238.135): icmp_seq=2 ttl=48 time=262 ms
64 bytes from nonorg.craigslist.org (208.82.238.135): icmp_seq=3 ttl=48 time=265 ms
64 bytes from nonorg.craigslist.org (208.82.238.135): icmp_seq=4 ttl=48 time=264 ms
64 bytes from nonorg.craigslist.org (208.82.238.135): icmp_seq=5 ttl=48 time=262 ms
64 bytes from nonorg.craigslist.org (208.82.238.135): icmp_seq=6 ttl=48 time=262 ms
64 bytes from nonorg.craigslist.org (208.82.238.135): icmp_seq=7 ttl=48 time=264 ms
64 bytes from nonorg.craigslist.org (208.82.238.135): icmp_seq=8 ttl=48 time=266 ms
64 bytes from nonorg.craigslist.org (208.82.238.135): icmp_seq=9 ttl=48 time=263 ms
64 bytes from nonorg.craigslist.org (208.82.238.135): icmp_seq=10 ttl=48 time=261 ms

--- craigslist.com ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9013ms
rtt min/avg/max/mdev = 261.003/263.328/266.148/1.586 ms
```

Figure 2: Ten Pings to craigslist.com

1. **Protocols Used:** Ping uses ICMP protocol. It sends the Echo Request packet to the destination host and receives Echo Reply packet from the same. It sits on the third layer of the protocol stack, which is the network layer.
2. **Latency:**

- (a) **Avg Latency of Craigslist:** 263.328 ms
- (b) **Avg Latency of Google:** 6.832 ms
- (c) Google's host has **smaller RTT** than Craigslist
- (d) **Reason for different latencies of websites:**
 - i. Google has more number of hosts than Craigslist, which splits network traffic
 - ii. Google's traffic might directly peer with most of the ISPs in the same tier of internet hierarchy.
- (e) **Reason for different latencies across pings for same website:**
 - i. Length of Queue for service at destination host is not constant in time and varies according to the number of user who requested for service before we place any request (queueing of packets, at the host)
 - ii. Network congestion is not constant, hence each switch in the network may not have same number of packets it has to route across different time
 - iii. Different routes may be taken for different pings leading to different paths and hence latencies

3. using IPv6 for both websites

```
arpit@arpit-linux:~/Desktop/iitd/sem_7/COL334/projects/Getting-To-Know-Network-Traffic$ ping -c 10 -6 www.google.com
PING www.google.com (2404:6800:4002:830::2004) 56 data bytes
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=1 ttl=115 time=11.4 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=2 ttl=115 time=6.65 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=3 ttl=115 time=6.48 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=4 ttl=115 time=9.37 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=5 ttl=115 time=7.40 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=6 ttl=115 time=8.87 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=7 ttl=115 time=6.61 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=8 ttl=115 time=6.31 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=9 ttl=115 time=6.63 ms
64 bytes from tzelb-ap-in-x04.1e100.net (2404:6800:4002:830::2004): icmp_seq=10 ttl=115 time=7.39 ms

--- www.google.com ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9012ms
rtt min/avg/max/mdev = 6.309/7.711/11.411/1.580 ms
```

Figure 3: Ten Pings to www.google.com using IPv6

```
arpit@arpit-linux:~/Desktop/iitd/sem_7/COL334/projects/Getting-To-Know-Network-Traffic$ ping -c 10 -6 craigslist.com
ping: craigslist.com: Address family for hostname not supported
```

Figure 4: Error when pinging craigslist.com

- (a) How to force IPv6: pass a flag -6 to force ping to follow IPv6
- (b) Result: IPv6 was supported by Google's host but not by Craigslist's host
- (c) Why *ping -6* Failed for Craigslist's host:
 - i. Since IPv6 worked for Google's host, implies that my computer and Google's host both support IPv6. If a failure of Address Family support has occurred it must have occurred on Craigslist's server. This implies Craigslist's server does not support IPv6 addresses.

- ii. When checking the IPv6 address for craigslist.com using dig AAAA craigslist.com, my computer does not find any IPv6 address as can be seen from Fig. 5, hence does not know which address to resolve to. Hence, forcing ping to follow IPv6 cannot be executed.

4. Max size of the packets

Table 1: Max Size of Data Transmitted from Ping, practically obtained by experiment

Host	Max Message Size	ICMP Header Size	Max Total Size
www.google.com	1462 bytes	8 bytes	1470 bytes
craigslist.com	283 bytes	28 bytes	311 bytes

- (a) Reason of Limit to Size: The network interfaces have a Maximum Transmission Unit for precautions (not to overload a server or a network device, which would disallow the timely servicing of all packets), that limit the packet size. Hence the bottleneck in the path is the device with the smallest Maximum Transmission Unit
- (b) The packet size may also be limited by the OS
- (c) For IPv6 over ethernet networks is limited ideally to 1500 bytes, but Table 1 shows what was practically achieved
- (d) The difference between practical and ideal may be caused by a network device through which the packet was routed has smaller Maximum Transmission Unit.
- (e) To send a larger packet size, the packet must be fragmented

1.2 traceroute

- IPv4 Address for www.google.com : 172.217.26.36
- IPv4 Address for craigslist.com : 208.82.238.135

```

arpit@arpit-linux:~/Desktop/iitd/sem_7/COL334/projects/Getting-To-Know-Network-Traffic$ traceroute 172.217.26.36
traceroute to 172.217.26.36 (172.217.26.36), 30 hops max, 60 byte packets
 1 10.184.0.13 (10.184.0.13) 2.912 ms 2.897 ms 2.860 ms
 2 10.255.109.100 (10.255.109.100) 2.830 ms 2.821 ms 2.795 ms
 3 10.255.107.3 (10.255.107.3) 6.049 ms 6.010 ms 6.000 ms
 4 10.119.233.65 (10.119.233.65) 6.038 ms 6.020 ms 6.002 ms
 5 10.1.207.69 (10.1.207.69) 38.404 ms 38.420 ms 38.410 ms
 6 10.1.200.137 (10.1.200.137) 43.904 ms 45.385 ms 45.331 ms
 7 10.255.238.122 (10.255.238.122) 35.390 ms 10.255.238.254 (10.255.238.254) 36.660 ms 10.255.238.122 (10.255.238.122) 36.633 ms
 8 10.152.7.214 (10.152.7.214) 35.328 ms 35.879 ms 43.299 ms
 9 72.14.204.62 (72.14.204.62) 37.983 ms 37.966 ms 44.192 ms
10 * * *
11 142.250.61.202 (142.250.61.202) 38.307 ms 74.125.253.166 (74.125.253.166) 37.794 ms 44.505 ms
12 142.250.226.66 (142.250.226.66) 37.868 ms 142.250.226.134 (142.250.226.134) 60.989 ms 192.178.110.104 (192.178.110.104) 37.131 ms
13 142.251.198.1 (142.251.198.1) 187.928 ms * 29.302 ms
14 192.178.252.104 (192.178.252.104) 34.541 ms 192.178.252.110 (192.178.252.110) 27.211 ms 142.251.247.50 (142.251.247.50) 27.056 ms
15 192.178.83.221 (192.178.83.221) 27.184 ms 26.093 ms 216.239.54.93 (216.239.54.93) 30.241 ms
16 142.251.49.115 (142.251.49.115) 27.202 ms 31.590 ms 142.251.49.121 (142.251.49.121) 24.873 ms
17 nrt12s17-in-f4.1e100.net (172.217.26.36) 32.317 ms 25.898 ms 27.736 ms

```

Figure 5: Trace Route of sending packet to www.google.com

A Number of Hops:

```

arpit@arpit-linux:~/Desktop/iitd/sem 7/COL334/projects/Getting-To-Know-Network-Traffic$ traceroute 208.82.238.135
traceroute to 208.82.238.135 (208.82.238.135), 30 hops max, 60 byte packets
 1  10.184.0.13 (10.184.0.13)  3.480 ms  3.487 ms  3.460 ms
 2  10.255.109.100 (10.255.109.100)  2.358 ms  2.343 ms  2.304 ms
 3  10.255.107.3 (10.255.107.3)  3.406 ms  3.383 ms  3.352 ms
 4  10.119.233.65 (10.119.233.65)  3.358 ms  3.319 ms  3.331 ms
 5  * * *
 6  10.119.234.162 (10.119.234.162)  4.165 ms  8.388 ms  8.327 ms
 7  59.144.234.93 (59.144.234.93)  8.292 ms  6.017 ms  5.993 ms
 8  116.119.57.82 (116.119.57.82)  135.534 ms  116.119.57.86 (116.119.57.86)  134.643 ms  116.119.112.88 (116.119.112.88)  135.513 ms
 9  mei-b5-link.ip.twelve99.net (62.115.42.118)  192.451 ms  191.107 ms  194.825 ms
10  * * *
11  ael.6.bar2.sanfrancisco1.net.lumen.tech (4.69.140.153)  266.800 ms  266.769 ms  267.847 ms
12  craigslist.bar2.sanfrancisco1.level3.net (4.53.134.6)  284.747 ms  282.738 ms  283.102 ms
13  nonorg.craigslist.org (208.82.238.135)  266.215 ms  266.202 ms  266.177 ms

```

Figure 6: Trace Route of sending packet to craigslist.com

Table 2: Number of Hops for Websites

Host	Number of Hops
www.google.com	17
craigslist.com	13

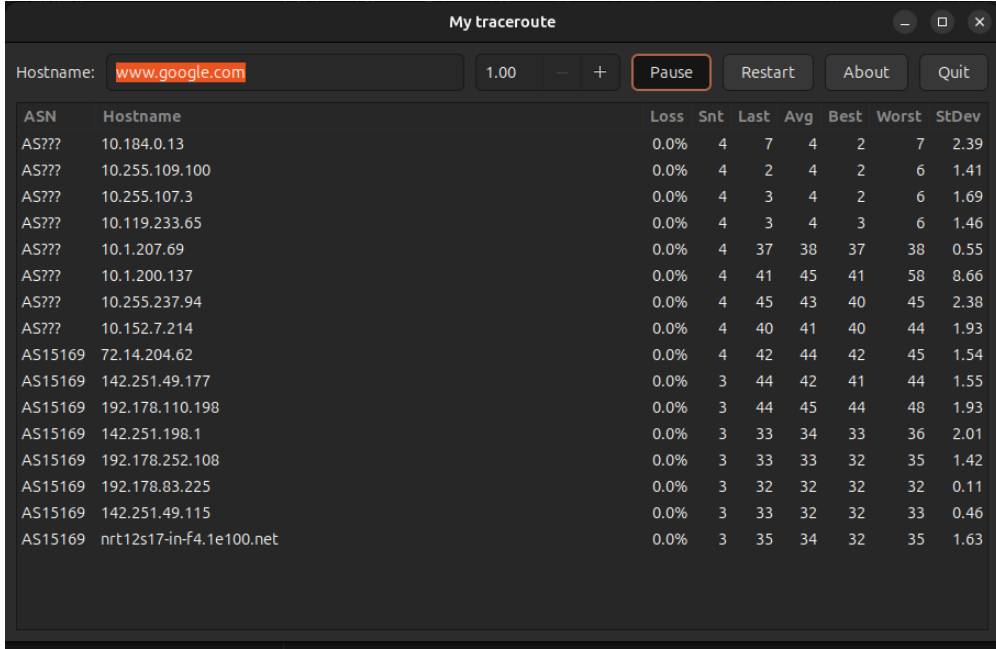


Figure 7: Autonomous System Number for each IP Address in the trace route of www.google.com (produces using the tool: mtr)

- B Explanation for ”*”: Some servers do not cater to traceroute packets, for security reasons and traffic control, hence do not send the Time Exceeded packet back to the source and hence, we do not have information about this node. This is represented in the traceroute by ”*”
- C Multiple IP Addresses for the same Hop Count: *traceroute* sends three packets for each hop. The three packets may opt for independent routes, depending on the congestion of the network. *traceroute* lists all the unique ip addresses of the nodes encountered by the three packets

ASN	Hostname	Loss	Snt	Last	Avg	Best	Worst	StDev
AS???	10.184.0.13	0.0%	9	1	3	1	10	2.96
AS???	10.255.109.100	0.0%	9	1	4	1	11	3.17
AS???	10.255.107.3	0.0%	9	5	3	1	7	2.14
AS???	10.119.233.65	0.0%	9	2	3	2	9	2.38
AS???	???	100.0%	9	0	0	0	0	0.00
AS???	10.119.234.162	0.0%	9	7	6	4	8	1.49
AS9498	59.144.234.93	0.0%	9	4	8	4	20	5.24
AS9498	116.119.112.88	0.0%	9	149	136	132	149	5.35
AS1299	62.115.42.118	0.0%	9	198	206	196	267	24.84
AS???	???	100.0%	9	0	0	0	0	0.00
AS3356	4.69.140.153	0.0%	8	267	268	262	302	13.53
AS3356	4.53.134.6	0.0%	8	281	283	280	297	5.60
AS22414	208.82.238.135	0.0%	8	260	264	260	285	8.56

Figure 8: Autonomous System Number for each IP Address in the trace route of craigslist.com (produces using the tool: mtr)

D The following table lists the IP Addresses and their corresponding RTTs and Geolocations

Table 3: IP Addresses and their GeoLocations for www.google.com

Sl No	IP Address	DNS	DNS Geolocation	Maxmind Geolocation	RTT (ms)
1	10.194.0.1	NA	NA	NA 6.100	
2	10.254.238.1	NA	NA	NA	6.087
3	10.255.107.3	NA	NA	NA	21.154
4	10.119.233.65	NA	NA	NA	21.147
5	10.1.207.69	NA	NA	NA	37.188
6	10.1.200.137	NA	NA	NA	50.264
7	10.255.237.94	NA	NA	NA	115.352
8	10.152.7.214	NA	NA	NA	215.550
9	72.14.204.62	NA	NA	United States (US), North America	215.512
10	NA	NA	NA	NA	
11	142.250.214.102	NA	NA	United States (US), North America	186.456
12	142.251.77.68	NA	NA	United States (US), North America	151.676
13	142.251.197.253	NA	NA	United States (US), North America	33.512
14	142.251.247.50	NA	NA	United States (US), North America	36.570
15	192.178.83.215	NA	NA	United States (US), North America	59.573
16	142.251.49.115	NA	NA	United States (US), North America	45.122
17	172.217.26.36	nrt12s17-in-f4.1e100.net or nrt12s17-in-f36.1e100.net or tزدelb-ap-in-f4.1e100.net	Narita, Japan, Tanzania	United States (US), North America	48.186

- For the case of www.google.com, most of the network devices that relay the packet are private and hence no information is obtained about them. However we observe a delta in 9th hop, therefore we can assume that the packet has crossed the country.
- For craigslist.com : The bigger deltas are observed when there is a change in country. For eg., from Table 4, we observe that upto Bangalore the RTT was 5.77 ms but as the relay proceeded to the US, the RTT becomes significantly higher, 197.05 ms.

Table 4: IP Addresses and their GeoLocations for craigslist.com

Sl No	IP Address	DNS	DNS Geolocation	Maxmind Geolocation	RTT (ms)
1	10.184.0.13	NA	NA	NA	685.733
2	10.255.109.100	NA	NA	NA	685.597
3	10.255.107.3	NA	NA	NA	685.551
4	10.119.233.65	NA	NA	NA	685.507
5	*	NA	NA	NA	*
6	10.119.234.162	NA	NA	NA	685.376
7	59.144.234.93	NA	NA	Bengaluru, Karnataka, India	5.715
8	116.119.112.88	NA	NA	India	138.551
9	62.115.42.118	mei-b5-link.ip.twelve99.net	Meridian, Mississippi, USA	France	197.050
10	*	NA	NA	NA	*
11	4.69.140.153	ae1.6.bar2.SanFrancisco1.net.lumen.tech	San Francisco, California, United States (US), North America	United States, North America	268.401
12	4.53.134.6	CRAIGSLIST.bar2.SanFrancisco1.Level3.net	San Francisco, California, United States (US), North America	San Francisco, California, United States (US), North America	270.387
13	208.82.238.135	nonorg.craigslist.org	San Francisco, California, United States (US), North America	San Francisco, California, United States (US), North America	259.620

Hence the data intuitively makes sense.

E Three Tier Architecture:

- (a) craigslist.com : Here we observe the three tier architecture clearly. Since first the packet travels from Delhi to Bangalore (which is a 2nd tier ISP transfer), then the exchange is observed from Bangalore to France and France to San Fransico which are a 1st Tier ISP Transfer. Finally 2nd and 3rd tier transfers occur for the packet to reach craigslist.com server
- (b) www.google.com : Here, we do not observe the three tier architecture clearly. Most of the transfers are through private ip addresses. Google peers its packets in the same level of hierarchy. This is the reason why we observe so many regional ISPs (Unites States (US), North America).

2 Network Traffic Collection and Analysis

2.1 Traffic Capture

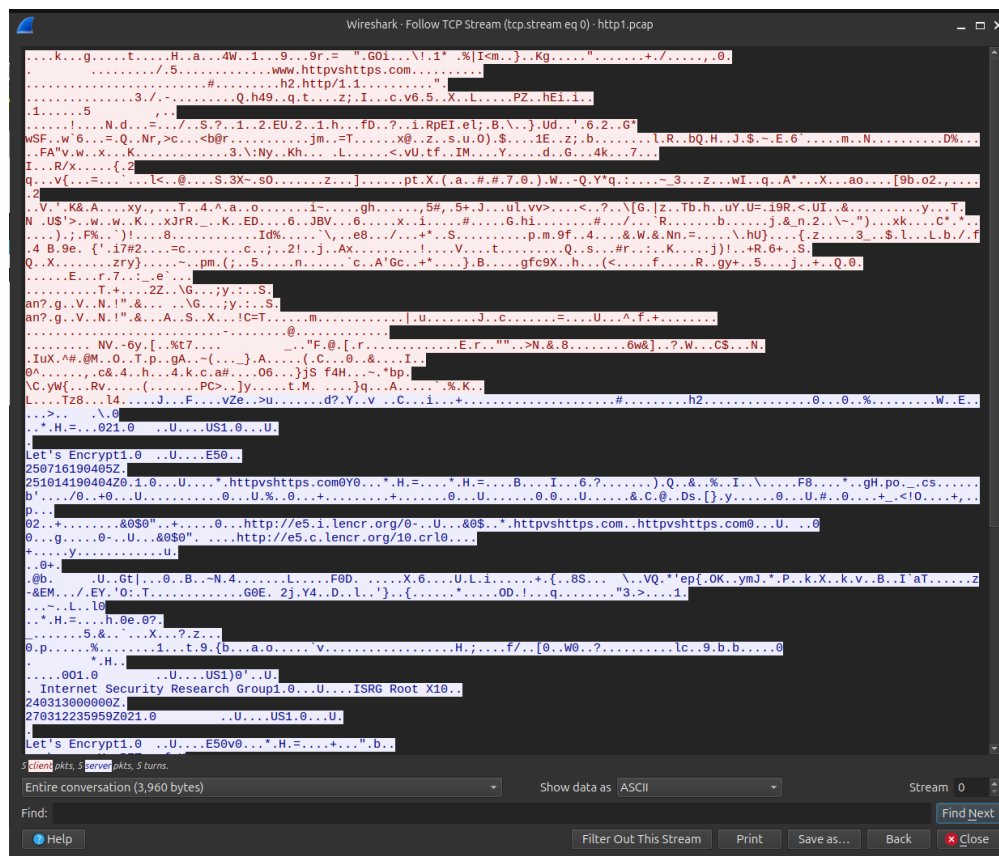
A Median Time taken for the DNS request-response to comlete: **42.43803024291992**. Note Median was taken on the pcap file when applied with the filter of DNS

B HTTP

- (a) Number of HTTP Requests = 363 (fitler used: *http.request*)
- (b) How webpages are structured: The webpage is structured like a tree. The root of the tree is the webpage itself. Root's children are the first level abstraction in the webpage, such as the title, the box that contains green ticks etc.. The next level with each abstraction, contains the green ticks itself in the case of the abstraction mentioned previously.
- (c) How browsers render complex pages with multiple images and files: The browser recursively calls on the nodes of the tree (mentioned above) and fetches the required informaiton, and displays them according to their HTML Code. This way it is able to render complex images and texts.

C TCP Connection

- (a) Number of TCP Connections = 31 (filter used: `tcp.flags.syn == 1 and tcp.flags.ack == 0`)
- (b) Number of HTTP Connections == Number of TCP Connections: No, they are generally not equal, however they are related. In one TCP Connection multiple HTTP requests can be made.
- (c) Content Object Fetch over same TCP Connection: Yes, some contents get fetched over the same TCP Connection. This can be supported from the fact that HTTP transfer are made with HTTP/1.1, which implies sustained TCP Connection for multiple HTTP transfers. This was verified using the filter `tcp.stream eq 0` where we checked the upstream flow. This showed multiple to and fro packet flows.



D HTTPs packet trace

- (a) HTTP traffic there? No, there is no HTTP Traffic (filter used: `HTTP`)
- (b) content transfer of HTTP and JavaScript files? and why?: There is no content transfer of HTTP and JavaScript. HTTPs is secured data transfer, therefore the file content is encrypted and hence not visible without the key to the file. This is the reason why wireshark does not show this content.

- (c) dns traffic: Yes, DNS Traffic is present. DNS is required for look ups to convert web addresses to their corresponding IP Addresses. Once the ip addresses for the websites are resolved no more DNS Traffic is present. (filter used: *dns*)
- (d) number of tcp connections logged = 7
- (e) Number of HTTP Connections==Number of TCP Connections?: They are not equal in this case. However the HTTPS case has smaller number of TCP Connections. A potential reason is multiplexing, allowing many HTTPS request to be sent simultaneously.

2.2 Performance Analysis

E HTTP vs HTTPS

- (a) comparison of time taken for downloads: HTTP::17.132s and HTTPS::1.614 (93% faster than HTTP)
- (b) Observation from the plots:
 - i. The download plot for https.pcap has significant throughput for download in comparison to that for http.pcap. Since the amount of data downloaded is the same, hence if the time of download reduces (as inferred from the time on the website and also the x axis of the plots), this implies throughput increases. The download throughput are upto 100 times more in https than http
 - ii. Time of downloads can also be justified from the RTT plot. RTT is much smaller in https.pcap case than in comparison to http.pcap. This implies smaller time for data transfers from https.pcap than http.pcap
- (c) Assumption for RTT Plots: We assume no retransmission of signal. Hence, we confirm ACK from the receiver by matching it to the estimated Acknowledgment Number, that is, we find if $A == S + L$. Here A is the Acknowledgment number received from the receiver, S is the sequence number when the message in consideration was sent to the receiver from the sender and L is the number of bytes of the data.
- (d) Note: RTT values are recorded at the time of ACK of the sent message
- (e) Note: The images for throughputs have been found for each second on the wall clock

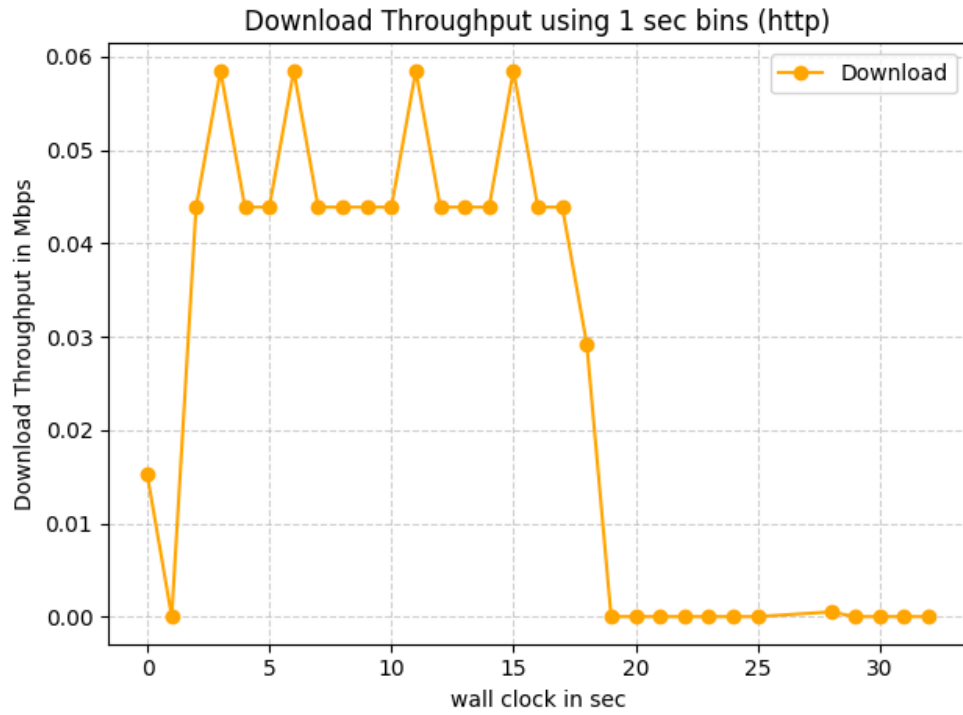


Figure 9: Download Throughput using http.pcap

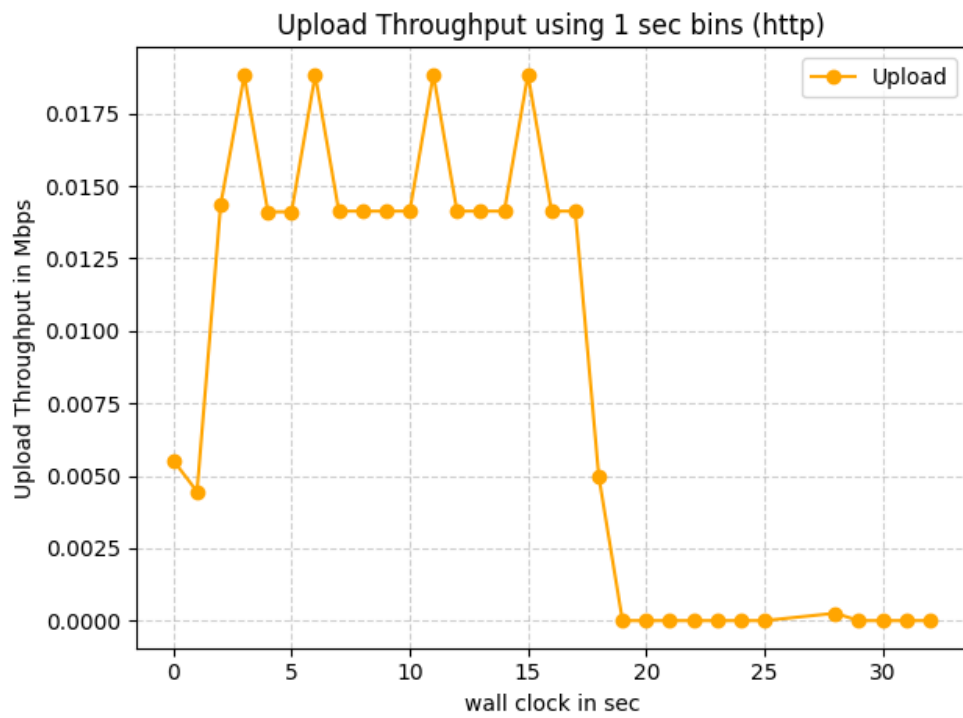


Figure 10: Upload Throughput using http.pcap

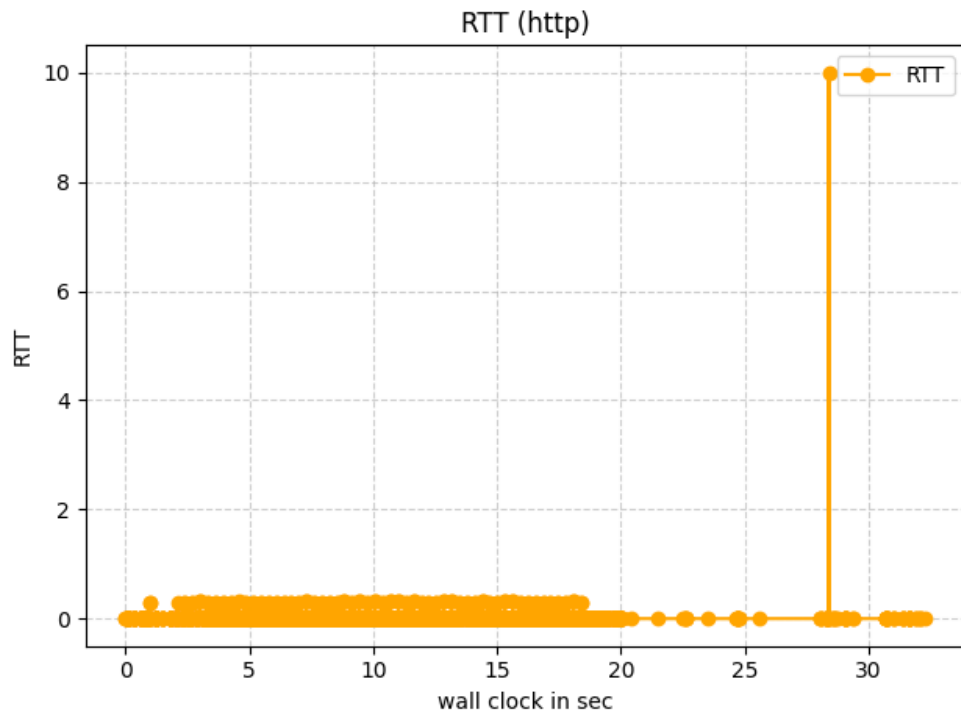


Figure 11: RTT using http.pcap

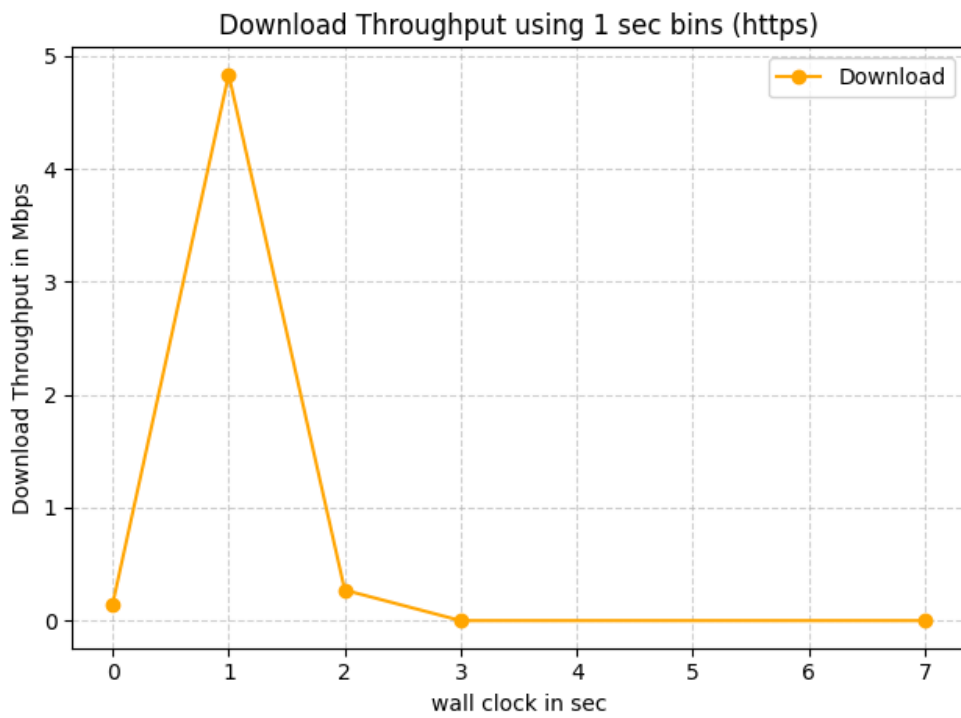


Figure 12: Download Throughput using https.pcap

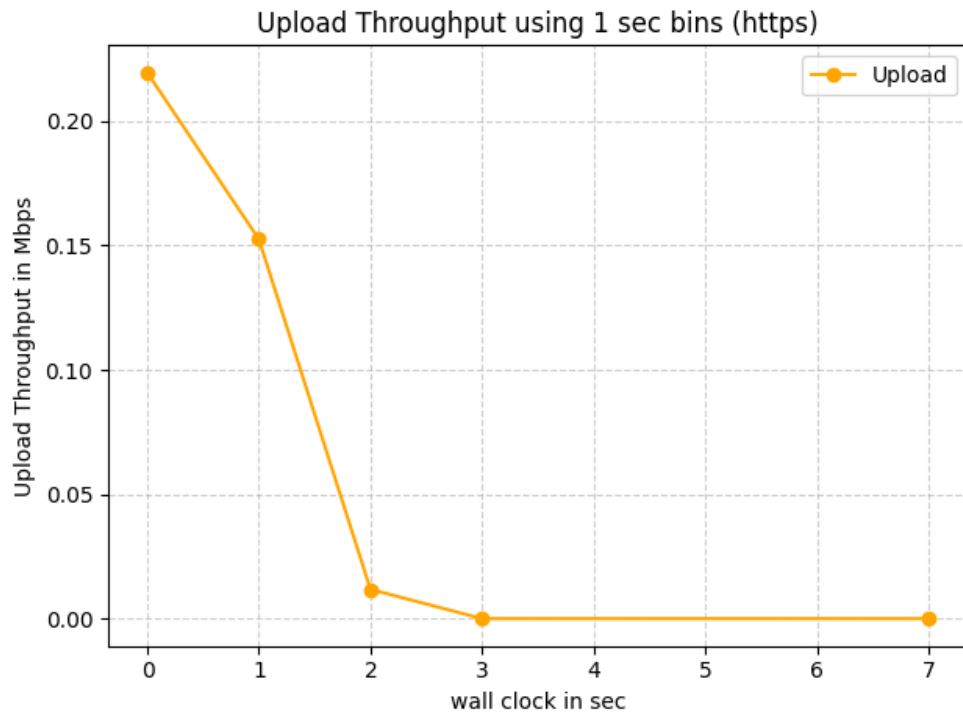


Figure 13: Upload Throughput using https.pcap

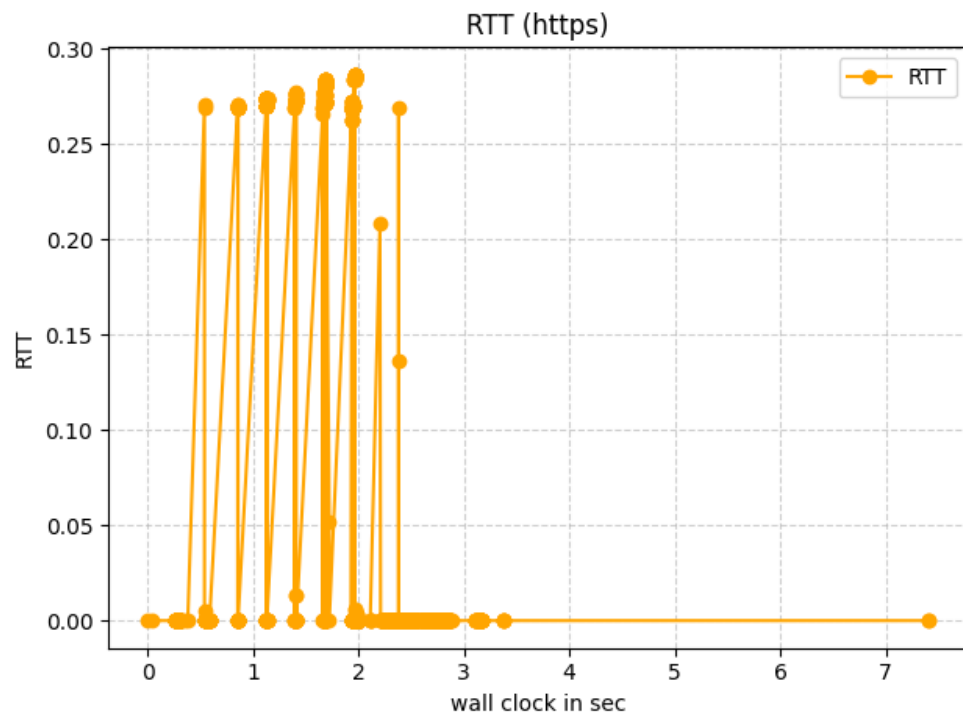


Figure 14: RTT using https.pcap