

# Wireshark Analysis

Group CS18:

Ridhiman Kaur Dhindsa, 210101088

Harsh Katara, 210101045

Dhruv Patel, 210101075

**CS342**

**Assignment 2,**

**Question 4**

*Wireshark*

*Analysis of HTTP,*

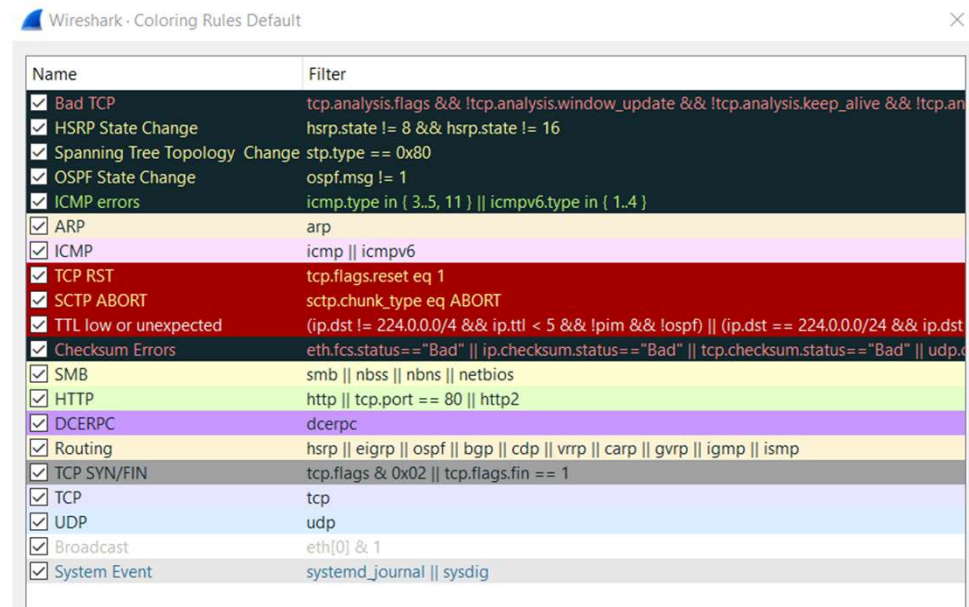
*Web Cache, DNS,*

*and Transport*

*Layer Protocols*

# Wireshark colour coding

Color of row	Packet Type
Light purple	TCP
Light blue	UDP
Black	Packets with errors
Light green	HTTP traffic
Light yellow	Windows-specific traffic, Server Message Blocks (SMBs) etc.
Dark yellow	Routing
Grey	TCP SYN/FIN
Red	TTL low or unexpected



## Wireshark filters used

- `ip.addr == 172.17.1.1 or ip.src/ip.dst == 172.17.1.1`
- `tcp, udp, dns, http`
- `tcp.port == 80 || udp.port == 80, TCP or UDP port is 80 (HTTP)`
- `tcp contains "youtube", udp contains "youtube", http contains "youtube"`
- `frame contains "mail"`

# Wireshark analysis of Network packets

## Activity: Watching Youtube videos

Time: 12pm

Location: Hostel

Internet connection: LAN

Total packets= 241266

TCP packets= 18013 = 7.5% (light purple colour)

UDP packets= 139796 = 57.9% (light blue colour)

DNS packets= 606 = 0.3% (light blue colour)

HTTP GET, POST, OK 200 packets= 48 = 0.0002% (light green colour)

ARP packets (Broadcast) = 14308 = 5.9% (light yellow colour)

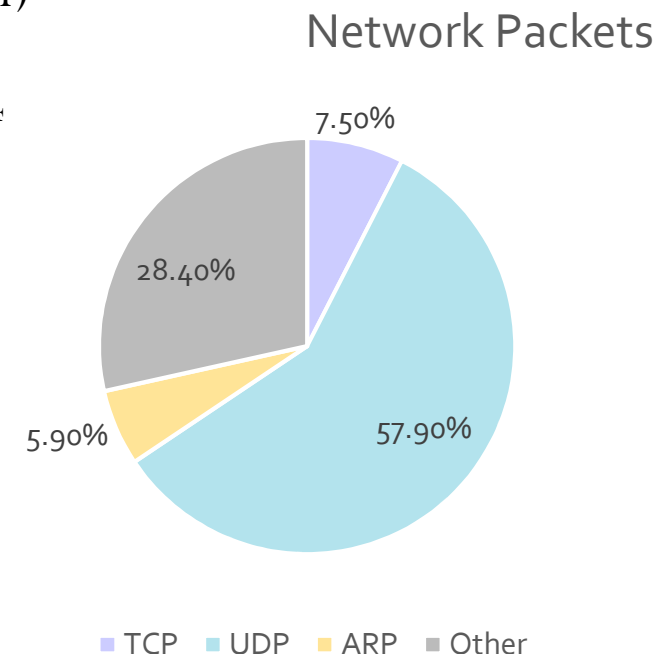
Total packets related to youtube IP address= 189+345 = 534

Packets going to youtube server= 189 = 35.4%

Packets coming from youtube server= 345 = 64.6%

UDP packets containing "youtube" = 25

TCP packets containing "youtube" = 0



5.9sec after starting packet capture, found 1 packet from server (172.17.1.1) to PC (10.20.1.57, port 53) which indicated a standard query response from server, i.e., the client-server handshake. The length was 548 bytes, UDP protocol, source port 53, destination port 55618. DNS response was standard- 1 question, 1 answer, no errors.

youtube study.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

udp

No.	Time	Source	Destination	Protocol	Length	Info
67810	139.054682	fe80::be22:28ff:fec...	ff02::1:2	DHCPv6	108	Solicit XID: 0xa6f595 CID: 00030
67843	139.066586	10.20.3.82	224.0.0.251	MDNS	606	Standard query response 0x0000 T
68156	139.145188	fe80::be22:28ff:fec...	ff02::1:2	DHCPv6	108	Solicit XID: 0x505002 CID: 00030
68180	139.153642	10.20.3.85	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
68263	139.189558	10.20.6.81	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
68399	139.242807	10.20.6.81	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1
68505	139.288099	10.20.3.71	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
68569	139.312540	10.20.0.224	239.255.255.250	SSDP	167	M-SEARCH * HTTP/1.1
68575	139.315677	fe80::6ef6:1142:635...	ff02::1:2	DHCPv6	151	Solicit XID: 0x651545 CID: 00010
68630	139.347114	10.20.5.140	224.0.0.251	MDNS	88	Standard query 0x0000 PTR _servi
68633	139.347457	10.20.1.57	10.20.5.140	MDNS	121	Standard query response 0x0000 P
68634	139.347831	10.20.0.235	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
68705	139.383304	10.20.3.226	239.255.255.250	SSDP	179	M-SEARCH * HTTP/1.1
68727	139.396879	fe80::be22:28ff:fec...	ff02::1:2	DHCPv6	108	Solicit XID: 0x860599 CID: 00030
68839	139.451696	10.20.3.250	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
68955	139.556203	fe80::6ef6:1142:635...	ff02::1:2	DHCPv6	151	Solicit XID: 0x651545 CID: 00010

< >

> Source: HP\_58:89:d6 (e8:d8:d1:58:89:d6)  
Type: IPv6 (0x86dd)

> Internet Protocol Version 6, Src: fe80::fa8:c813:e3a2  
User Datagram Protocol, Src Port: 546, Dst Port: 547

Source Port: 546  
Destination Port: 547  
Length: 103  
Checksum: 0xdebe0 [unverified]  
[Checksum Status: Unverified]  
[Stream index: 415]  
[Timestamp]

0000 33 33 00 01 00 02 e8 d8 d1 58 89 d6 86 dd 60 07  
0010 91 69 00 67 11 01 fe 80 00 00 00 00 00 0f a8  
0020 c8 13 e3 a2 57 c2 ff 02 00 00 00 00 00 00 00  
0030 00 00 00 01 00 02 02 22 02 23 00 67 de b0 01 84  
0040 8e 49 00 08 00 02 18 a5 00 01 00 0e 00 01 00 01  
0050 25 b7 f7 53 e8 d8 d1 58 89 d6 00 03 00 0c 0b e8  
0060 d8 d1 00 00 00 00 00 00 00 00 00 27 00 11 00 0f  
0070 4c 41 50 54 4f 50 2d 55 39 36 4e 49 50 33 4d 00  
0080 10 00 0e 00 00 01 37 00 08 4d 53 46 54 20 35 2e  
0090 30 00 06 00 08 00 11 00 17 00 18 00 27

User Datagram Protocol: Protocol

Packets: 241266 · Displayed: 139796 (57.9%) · Dropped: 0 (0.0%) Profile: Default

There were packets for client hello, server hello, neighbor solicitation (light pink), ARP broadcast (light yellow), router advertisement, multicast listener reports etc.

# Activity: Watching Instagram reels

Time: 1.30pm      Location: Lab      Internet connection: LAN

Total packets= 110468

TCP packets= 50258 = 45.5% (light purple/grey colour)

UDP packets= 50542 = 45.8% (light blue colour)

DNS packets= 163 = 0.1% (light blue colour)

HTTP GET, POST, OK 200 packets= 6 = 0.00% (light green colour)

ARP packets (Broadcast) = 2098 = 1.9% (light yellow colour)

Total packets related to instagram IP address= 419+681 = 1100

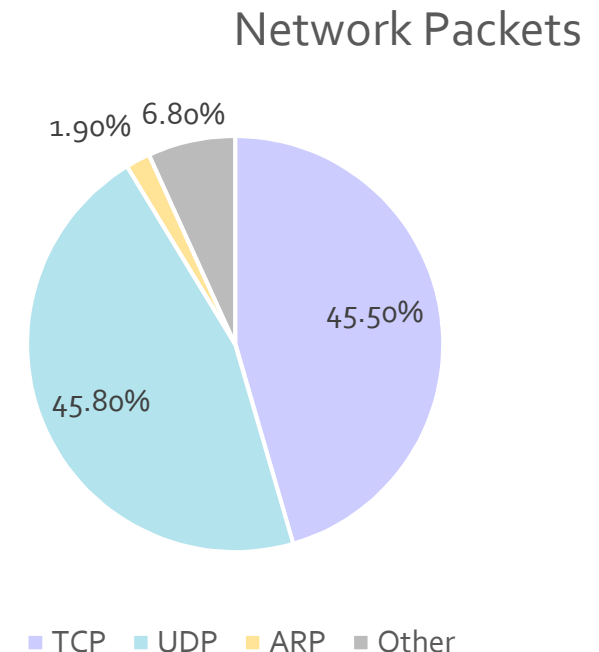
Packets going to instagram server= 419 = 38.09%

Packets coming from instagram server= 681 = 61.90%

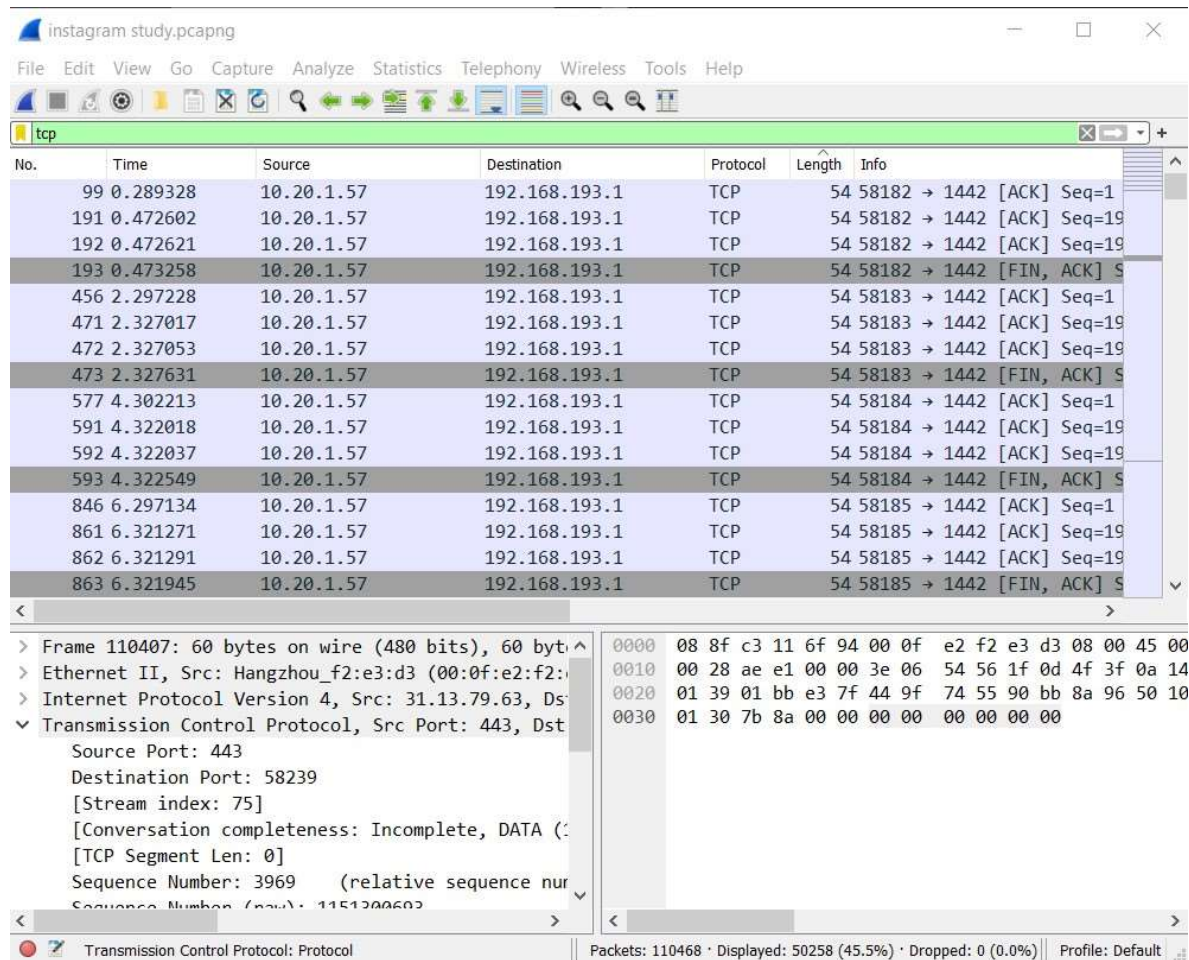
UDP packets containing “instagram” = 40

TCP packets containing “instagram” = 21

Most packets being sent to and from instagram servers ranged from 200-1000 bytes.



## Observations:



The image shows a Wireshark packet capture of an Instagram session. The top pane displays a list of TCP packets, all originating from 10.20.1.57 and destined for 192.168.193.1. The middle pane shows the details of a selected packet (No. 110407), identifying it as a Transmission Control Protocol (TCP) segment. The bottom pane shows the raw packet data in hexadecimal and ASCII format.

No.	Time	Source	Destination	Protocol	Length	Info
99	0.289328	10.20.1.57	192.168.193.1	TCP	54	58182 → 1442 [ACK] Seq=1
191	0.472602	10.20.1.57	192.168.193.1	TCP	54	58182 → 1442 [ACK] Seq=19
192	0.472621	10.20.1.57	192.168.193.1	TCP	54	58182 → 1442 [ACK] Seq=19
193	0.473258	10.20.1.57	192.168.193.1	TCP	54	58182 → 1442 [FIN, ACK] S
456	2.297228	10.20.1.57	192.168.193.1	TCP	54	58183 → 1442 [ACK] Seq=1
471	2.327017	10.20.1.57	192.168.193.1	TCP	54	58183 → 1442 [ACK] Seq=19
472	2.327053	10.20.1.57	192.168.193.1	TCP	54	58183 → 1442 [ACK] Seq=19
473	2.327631	10.20.1.57	192.168.193.1	TCP	54	58183 → 1442 [FIN, ACK] S
577	4.302213	10.20.1.57	192.168.193.1	TCP	54	58184 → 1442 [ACK] Seq=1
591	4.322018	10.20.1.57	192.168.193.1	TCP	54	58184 → 1442 [ACK] Seq=19
592	4.322037	10.20.1.57	192.168.193.1	TCP	54	58184 → 1442 [ACK] Seq=19
593	4.322549	10.20.1.57	192.168.193.1	TCP	54	58184 → 1442 [FIN, ACK] S
846	6.297134	10.20.1.57	192.168.193.1	TCP	54	58185 → 1442 [ACK] Seq=1
861	6.321271	10.20.1.57	192.168.193.1	TCP	54	58185 → 1442 [ACK] Seq=19
862	6.321291	10.20.1.57	192.168.193.1	TCP	54	58185 → 1442 [ACK] Seq=19
863	6.321945	10.20.1.57	192.168.193.1	TCP	54	58185 → 1442 [FIN, ACK] S

Frame 110407: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0  
Ethernet II, Src: Hangzhou\_f2:e3:d3 (00:0f:e2:f2:d3), Dst: 31:13:79:63:00:00 (08:00:27:00:00:00)  
Internet Protocol Version 4, Src: 31.13.79.63, Destination: 192.168.193.1  
Transmission Control Protocol, Src Port: 443, Destination Port: 58239  
[Stream index: 75]  
[Conversation completeness: Incomplete, DATA (0)]  
[TCP Segment Len: 0]  
Sequence Number: 3969 (relative sequence number)  
Sequence Number (raw): 1151200602

0000 08 8f c3 11 6f 94 00 0f e2 f2 e3 d3 08 00 45 00  
0010 00 28 ae e1 00 00 3e 06 54 56 1f 0d 4f 3f 0a 14  
0020 01 39 01 bb e3 7f 44 9f 74 55 90 bb 8a 96 50 10  
0030 01 30 7b 8a 00 00 00 00 00 00 00 00 00 00 00

Transmission Control Protocol: Protocol | Packets: 110468 · Displayed: 50258 (45.5%) · Dropped: 0 (0.0%) | Profile: Default

All TCP packets related to Instagram had source IP address 10.20.1.57 and destination IP address 31.13.79.174 or 180.149.62.32. This indicates that while watching reels, most packets contain request queries from PC to Instagram servers.

It is observed that DNS packets constitute a very small percentage (0.3-0.6%) of total packets picked up by Wireshark.



# Activity: Using Gmail

*Sending and receiving emails via Gmail*

Time: 2.30pm      Location: Lab      Internet connection: LAN

Total packets= 128100

TCP packets= 17740 = 13.8% (light purple/grey colour)

UDP packets= 61374 = 47.9% (light blue colour)

DNS packets= 772 = 0.6% (light blue colour)

HTTP GET, POST, OK 200 packets= 12 = 0.00% (light green colour)

ARP packets (Broadcast) = 6764 = 5.3% (light yellow colour)

Total packets related to Gmail IP address= 17+17 = 34

Packets going to Gmail server= 17 = 50%

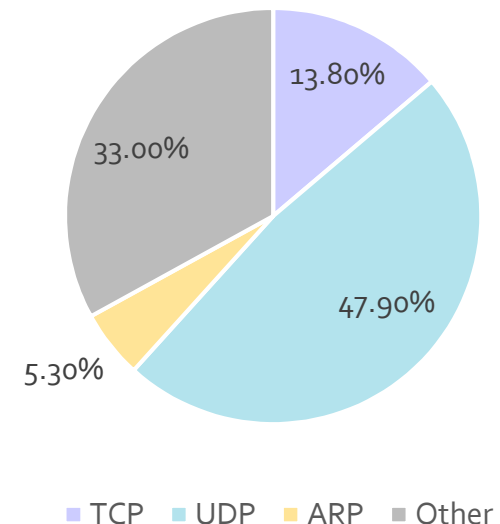
Packets coming from Gmail server= 17 = 50%

UDP packets containing “mail” = 34

TCP packets containing “mail” = 10

The Gmail packets have length approximately 380 bytes.  
Protocol= UDP, IPv4, Time to live (TTL)= 63 sec. Since it is Google (Gmail) so port 53 was used. There were 10 TCP packets with TLS Handshake protocol: Client Hello. No explicit email body could be found as Gmail uses SSL encryption for privacy and secure emailing.

Network Packets





# Activity: Online gaming

*Playing online game paper.io*

**Time:** 3pm

**Location:** Hostel

**Internet connection:** Airtel

Total packets= 1331

TCP packets= 772 = 58% (light purple/grey colour)

UDP packets= 543 = 40.8% (light blue colour)

DNS packets= 84 = 6.3% (light blue colour)

HTTP GET, POST, OK 200 packets= 0 = 0.00% (light green colour)

ARP packets (Broadcast) = 16 = 1.2% (light yellow colour)

Total packets related to gaming IP address= 84

Packets going to gaming server= 42 = 50%

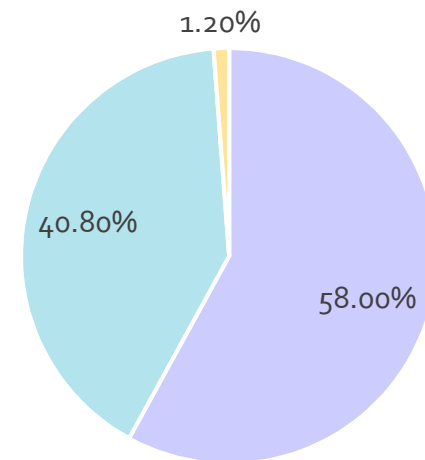
Packets coming from gaming server= 42 = 50%

UDP packets containing “paper.io” = 12

TCP packets containing “paper.io” = 0

The length of gaming packets sent over the network, range from 70-120 bytes.

Network Packets



■ TCP ■ UDP ■ ARP

# On visiting sites under HTTP protocol

Websites under HTTP protocol are not considered secure anymore, unlike those which open under HTTPS. HTTPS is essentially a more secure version of HTTP. It uses SSL/TLS to encrypt connections between web browsers and servers. This is why most packets in our previous activities contained TLS (Transport Layer Security) section.



No.	Time	Source	Destination	Protocol	Length	Info
15	0.247941	2401:4900:3a70:8de6...	2600:140f:a00::17df...	TCP	74	64523 → 443 [FIN]
16	0.252387	2600:140f:a00::17df...	2401:4900:3a70:8de6...	TLSv1.2	112	Application Data
17	0.252642	2401:4900:3a70:8de6...	2600:140f:a00::17df...	TCP	74	64523 → 443 [RST]
18	0.459268	2600:140f:a00::17df...	2401:4900:3a70:8de6...	TCP	86	[TCP Dup ACK 9#1]
19	0.459268	2600:140f:a00::17df...	2401:4900:3a70:8de6...	TCP	74	443 → 64523 [ACK]
20	0.687236	192.168.53.1	14.139.196.11	TCP	66	64524 → 1442 [SYN]
82	1.699100	192.168.53.1	14.139.196.11	TCP	66	[TCP Retransmissi]
83	1.808094	192.168.53.1	14.139.196.11	TCP	66	64520 → 1442 [SYN]
84	2.748604	192.168.53.1	14.139.196.11	TCP	66	64525 → 1442 [SYN]
88	2.825166	192.168.53.1	18.236.36.28	TCP	55	64507 → 80 [ACK]
94	2.904738	192.168.53.1	18.236.36.28	TCP	55	64509 → 80 [ACK]
109	3.127661	18.236.36.28	192.168.53.1	TCP	66	80 → 64507 [ACK]
110	3.208840	18.236.36.28	192.168.53.1	TCP	66	80 → 64509 [ACK]
111	3.705397	192.168.53.1	14.139.196.11	TCP	66	[TCP Retransmissi]
112	3.752400	192.168.53.1	14.139.196.11	TCP	66	[TCP Retransmissi]
113	3.768309	192.168.53.1	14.139.196.11	TCP	66	64521 → 1442 [SYN]
116	4.143959	192.168.53.1	14.139.196.11	TCP	66	[TCP Retransmissi]
130	5.756567	192.168.53.1	14.139.196.11	TCP	66	[TCP Retransmissi]
140	7.715078	192.168.53.1	14.139.196.11	TCP	66	[TCP Retransmissi]

In the above picture, most rows are red or black, indicating a non-secure connection.

13	0.247155	2401:4900:3a70:8de6...	2600:140f:a00::17df...	TCP	74 64523 → 443 [ACK] Seq
14	0.247483	2401:4900:3a70:8de6...	2600:140f:a00::17df...	TLSv1.2	112 Application Data
15	0.247941	2401:4900:3a70:8de6...	2600:140f:a00::17df...	TCP	74 64523 → 443 [FIN, ACK]
16	0.252387	2600:140f:a00::17df...	2401:4900:3a70:8de6...	TLSv1.2	112 Application Data
17	0.252642	2401:4900:3a70:8de6...	2600:140f:a00::17df...	TCP	74 64523 → 443 [RST, ACK]
18	0.459268	2600:140f:a00::17df...	2401:4900:3a70:8de6...	TCP	86 [TCP Dup ACK 9#1] 443
19	0.459268	2600:140f:a00::17df...	2401:4900:3a70:8de6...	TCP	74 443 → 64523 [ACK] Seq
20	0.687236	192.168.53.1	14.139.196.11	TCP	66 64524 → 1442 [SYN] Seq
21	0.689136	192.168.53.1	192.168.53.226	DNS	80 Standard query 0xbdea
22	0.689625	192.168.53.1	192.168.53.226	DNS	80 Standard query 0xe0fa
23	0.690050	192.168.53.1	192.168.53.226	DNS	80 Standard query 0xf410

> Frame 17: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface		0000	4e fe 4f 92 0f f0
> Ethernet II, Src: IntelCor_aa:54:a6 (54:14:f3:aa:54:a6), Dst: 4e:fe:4f:92		0010	71 3f 00 14 06 3f
> Internet Protocol Version 6, Src: 2401:4900:3a70:8de6:6109:94c2:5ac3:ebb1		0020	94 c2 5a c3 eb bf
> Transmission Control Protocol, Src Port: 64523, Dst Port: 443, Seq: 249,		0030	00 00 17 df f3 5a
		0040	59 00 50 14 00 00

HTTP sites don't have  
TLS section



72413	71.062156	31.13.79.63	10.20.1.57	TLSv1.3	1354 Application Data
72414	71.062199	10.20.1.57	31.13.79.63	TCP	54 58198 → 443 [ACK] Seq
72415	71.062789	31.13.79.63	10.20.1.57	TLSv1.3	1434 Application Data
72416	71.062789	31.13.79.63	10.20.1.57	TLSv1.3	1434 Application Data
72417	71.062841	10.20.1.57	31.13.79.63	TCP	54 58198 → 443 [ACK] Seq
72418	71.063804	31.13.79.63	10.20.1.57	TLSv1.3	1514 Application Data
72419	71.063804	31.13.79.63	10.20.1.57	TLSv1.3	1354 Application Data
72420	71.063804	31.13.79.63	10.20.1.57	TLSv1.3	1514 Application Data
72421	71.063876	10.20.1.57	31.13.79.63	TCP	54 58198 → 443 [ACK] Seq
72422	71.063892	10.20.1.57	31.13.79.63	TCP	54 58198 → 443 [ACK] Seq
72423	71.063894	31.13.79.63	10.20.1.57	TLSv1.3	1354 Application Data

> Frame 72415: 1434 bytes on wire (11472 bits), 1434 bytes captured (11472 b		0000	08 8f c3 11 6f 9
> Ethernet II, Src: Hangzhou_f2:e3:d3 (00:0f:e2:f2:e3:d3), Dst: CompalIn_11:		0010	05 8c 4f 0f 00 0
> Internet Protocol Version 4, Src: 31.13.79.63, Dst: 10.20.1.57		0020	01 39 01 bb e3 5
> Transmission Control Protocol, Src Port: 443, Dst Port: 58198, Seq: 608053		0030	04 cf 3a f9 00 0
> [2 Reassembled TCP Segments (1094 bytes): #72413(486), #72415(608)]		0040	5d 61 28 da 46 f
> Transport Layer Security		0050	cb 20 df c1 52 a
▼ TLSv1.3 Record Layer: Application Data Protocol: Hypertext Transfer Pro		0060	6b d8 03 37 f2 e
Opaque Type: Application Data (23)		0070	39 39 21 10 1d 0
Version: TLS 1.2 (0x0303)		0080	a1 50 5e dc 63 f
Length: 1089		0090	22 a7 55 55 3c 8
Encrypted Application Data: 8c3d98ea35afb501729123b6d882a3d386c72a2f		00a0	2c f3 03 65 a9 a
[Application Data Protocol: Hypertext Transfer Protocol]		00b0	76 07 32 3b 97 2
		00c0	23 9e 47 8f 45 d
		00d0	bb 58 15 c3 c8 5
		00e0	ff 43 bd 6e 56 a

HTTPS sites have TLS section  
indicating the version of TLS  
used (TLS 1.2), length,  
encrypted application data and  
protocol (Hypertext Transfer  
Protocol).

## Answers to assignment questions

Part 1)

Each packet typically contains the following fields:

- Frame- number, length, arrival time, epoch time, coloring rules etc.
- Ethernet- Source, Destination, Type (IPv4)
- Internet Protocol Version 4- differentiated services, TTL, protocol(TCP/UDP), src/dest address
- TCP/UDP field- src/dest port, segment length, flags, checksum etc.
- HTTP field- status code (e.g. 200 OK), response version (HTTP/1.1), date, content type, cache control
- Line based text data

The protocols involved include:

- Application Layer: HTTP, HTTPS, DNS.
- Transport Layer: This layer comprises of TCP, UDP, TLS
- Network Layer: IPv4
- Link Layer: ARP (Address Resolution Protocol), broadcasting over LAN

Contd...



Part 1) contd.

HTTP Packet format:

```
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
    [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
    Response Version: HTTP/1.1
    Status Code: 200
    [Status Code Description: OK]
    Response Phrase: OK
  Content-Length: 22\r\n
  Date: Sun, 10 Sep 2023 06:52:36 GMT\r\n
  Connection: close\r\n
  Content-Type: text/plain\r\n
  Cache-Control: max-age=30, must-revalidate\r\n
  \r\n
  [HTTP response 1/1]
  [Time since request: 0.376595000 seconds]
  [Request in frame: 83547]
  [Request URI: http://www.msftconnecttest.com/connecttest.txt]
  File Data: 22 bytes
```

**Status Line:** Contains the HTTP version (HTTP/1.1), a numeric status code (200 OK, 404 Not Found), a reason phrase (OK, Not Found) indicating the outcome of the request.

**Headers:** Key-value pairs providing additional information about the response, such as the Content-Type, Content-Length etc.

**Blank Line:** A blank line separating the headers from the message body (if any).

**Message Body:** Contains the actual content of the response, such as HTML, JSON, or binary data.

DNS packet format:

```
Domain Name System (response)
  Transaction ID: 0xa253
  > Flags: 0x8180 Standard query response, No error
  Questions: 1
  Answer RRs: 0
  Authority RRs: 1
  Additional RRs: 0
  < Queries
    > optimizationguide-pa.googleapis.com: type HTTPS, class IN
  < Authoritative nameservers
    > googleapis.com: type SOA, class IN, mname ns1.google.com
    [Request In: 65469]
  [Time: 0.001794000 seconds]
```

**DNS Header (12 bytes):** Identification (ID), Flags, Question Count, Answer Count, Authority Count, Additional Count- all 2 bytes each.

**Question Section (Variable Length):** Contains one or more DNS question records (QNAME, QTYPE, QCLASS).

**Answer Section (Variable Length):** Contains resource records (RRs) with answers to the DNS query.

**Authority Section (Variable Length):** Contains RRs that point to authoritative DNS servers for the queried domain.

**Additional Information Section (Variable Length):** Contains additional RRs that provide extra information about the domain in question.

**Part 2)** Source IP fields contain the IP address of the machine sending the request (most cases our PC) and destination IP field contains the IP address of server to which the request is directed (such as Instagram, YouTube, Google). Port number indicates the port from which request originates, e.g. Port 53 for Google related queries, port 587 for secure emailing etc. Destination port describes the port number on server where request must be entertained. Ethernet address (MAC) contains unique ID, src/dest addresses, routing information etc. Protocol number indicates TCP (6), UDP (17) etc.

---

### Part 3)

Sequence of messages while watching Youtube Videos:

1. **DNS Resolution (Client to DNS Server):** Protocol: Typically uses DNS (Domain Name System).
2. **TCP Handshake (Client to YouTube Server):** Protocol: TCP (Transmission Control Protocol). The three-way handshake (SYN, SYN-ACK, ACK) is used to establish the connection.
3. **HTTPS/TLS Handshake (Client to YouTube Server):** Protocol: HTTPS (HTTP over TLS/SSL). The client validates the server's certificate.
4. **HTTP Request for Video (Client to YouTube Server):** Protocol: HTTP (Hypertext Transfer Protocol) or HTTPS. The client sends an HTTP GET request to request a specific video resource.
5. **HTTP Response with HTML Page (YouTube Server to Client):** Protocol: HTTP or HTTPS.
6. **Client Parses HTML and Loads Video Player (Client-Side):** Protocol: HTML and JavaScript.
7. **HTTP Request for Video Manifest (Client to YouTube Server):** Protocol: HTTP or HTTPS. The client sends an HTTP request for the video manifest, which can be in DASH or HLS format.
8. **HTTP Response with Video Manifest (YouTube Server to Client):** Protocol: HTTP or HTTPS. The YouTube server responds with the video manifest, providing information about video quality options and URLs for video segments.
9. **Client Selects Playback Quality (Client-Side):** Protocol: JavaScript.
10. **HTTP Request for Video Segments (Client to YouTube Server):** Protocol: HTTP or HTTPS.
11. **HTTP Responses with Video Segments (YouTube Server to Client):** Protocol: HTTP or HTTPS.
12. **Client Continues to Fetch Segments (Streaming) (Client-Side):** Protocol: JavaScript.

Contd.....

### Part 3) Contd.

Handshaking mechanism:

- **TCP Handshake (Client to YouTube Server):** This involves a three-way handshake:

SYN (Synchronize): The client sends a TCP packet with the SYN flag set to request a connection.

SYN-ACK (Synchronize-Acknowledge): The YouTube server responds with a packet that has the SYN and ACK (Acknowledge) flags set, indicating its willingness to establish a connection.

ACK: The client acknowledges the server's response by sending a final ACK packet. At this point, the TCP connection is established.

- **HTTPS/TLS Handshake (Client to YouTube Server):** The handshake includes the following steps:

ClientHello: The client sends a message specifying its supported encryption algorithms and other parameters.

ServerHello: The YouTube server responds with its selected encryption settings.

Certificate Exchange: The server sends its digital certificate to prove its identity.

Key Exchange: The client and server exchange cryptographic keys.

Finished: Both parties confirm the handshake completion.

---

### Part 4)

DNS resolution is the first step in accessing any website or service, including YouTube, Gmail, Instagram. It translates human-readable domain names (e.g., [www.youtube.com](https://www.youtube.com)) into IP addresses (e.g., 216.58.192.14), allowing clients to locate appropriate servers. TCP is used for the establishment of reliable, connection-oriented communication between the client's device and YouTube's servers. UDP may be used for YouTube/Instagram, but for email, TCP and SMTP must be used so that there's no loss of data. HTTPS/TLS is essential for secure communication between the client and server. It encrypts data exchanged during the handshake and subsequent requests, protecting user privacy and preventing data interception.



### Part 5)

The Youtube packet data contained 139 packets with HTTP status code: 304 Not Modified. This indicates that these packets were cached on the system and weren't required to be fetched from server. This is part of **Video Segment Caching**:

Video segments, which make up the video stream, can be cached temporarily by the client-side video player. As a user watches a video, previously downloaded segments may be stored in memory or on disk for near-future playback. This local caching helps ensure smooth video streaming and minimizes re-downloads of segments that haven't changed.

---

### Part 6) Statistics for Youtube experiment:

Time: 12pm

Location: Lab

Connection: IITG LAN

Quantity	Value
Throughput	26-27 packets/sec
RTT	17.43 ms
Packet Size	470-500 bytes
Number of packets lost	0%
TCP packets	18013 = 7.5%
UDP packets	139796 = 57.9%
Number of responses per request	1-4

Time: 5pm

Location: Hostel

Connection: Airtel

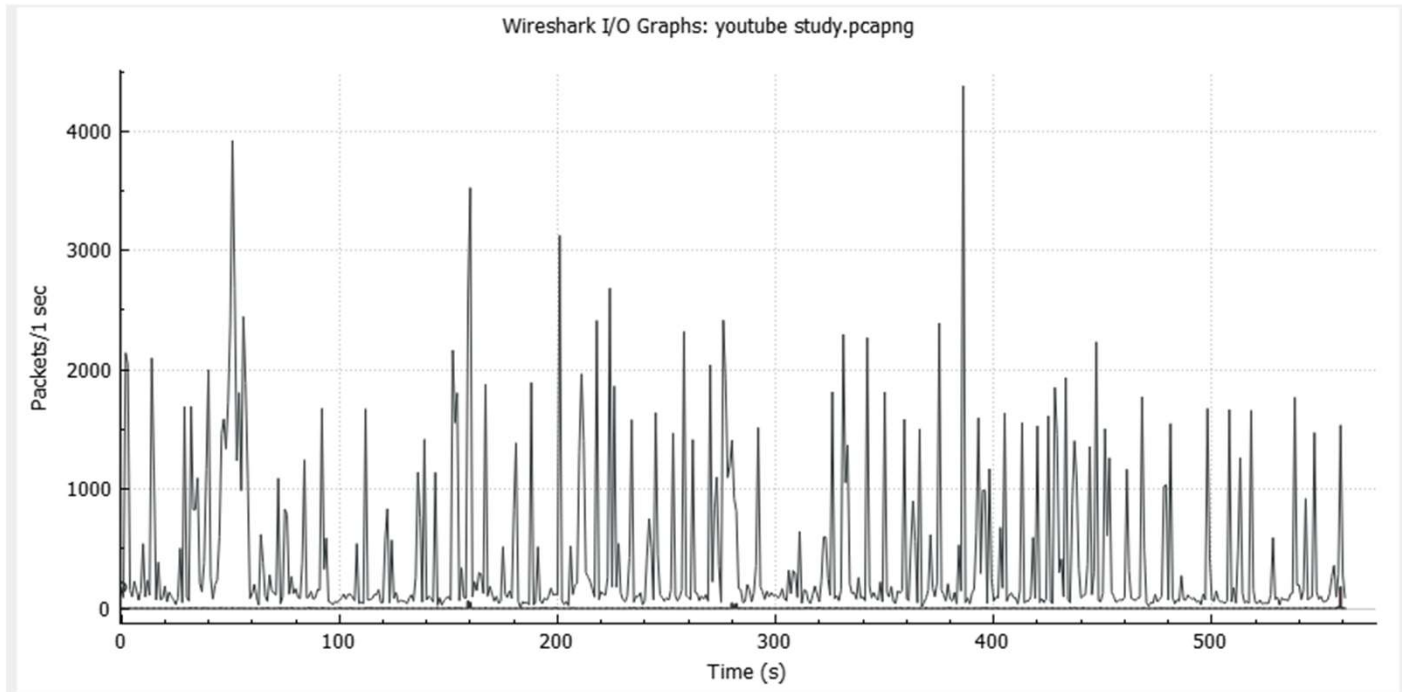
Quantity	Value
Throughput	22 packets/sec
RTT	19 ms
Packet Size	400-450 bytes
Number of packets lost	2%
TCP packets	6.3%
UDP packets	45.6%
Number of responses per request	1-4

Time: 10am

Location: Hostel

Connection: Airtel

Quantity	Value
Throughput	25 packets/sec
RTT	17.5 ms
Packet Size	400-450 bytes
Number of packets lost	0%
TCP packets	7.1%
UDP packets	47.3%
Number of responses per request	1-4



Throughput