CS 342 : Assignment - 2
Group Number M20

Team Members:
1) Rasesh Srivastava - 210123072
2) Vidya Sagar G - 210123073
3) Kishlay Soni - 210123074

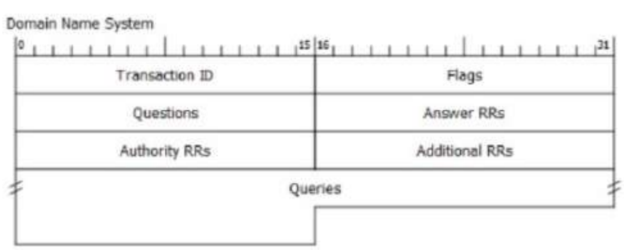
Question 4: Wireshark Analysis of Network Traffic for YouTube

We have studied network traffic for youtube.com, an American online video sharing and social media platform and we present our report while examining its network protocols, user experience, underlying technologies and network statistics.

Task 1: Protocol overview and brief explanation

1) Application Layer Protocols:

a) DNS(Domain Name System) protocol -

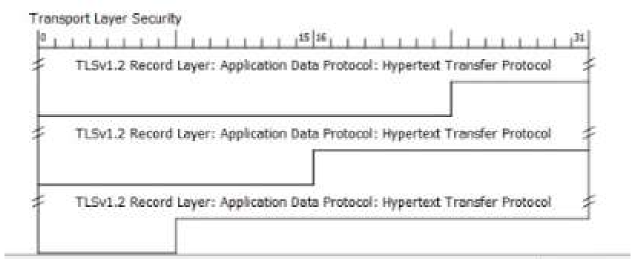


DNS is a query/response protocol. The client queries information in a single UDP request. This request is followed by a single UDP reply from the DNS server. A DNS query and a DNS reply share the same structure. The various fields include:

1) Transaction ID which is a 16-bit identification field generated by the device that creates the query.
2) Flags contain a number of sub fields like:
    a) Query/Response which contains 0/1 depending on whether it is a query or a response.
    b) Opcode specifies the type of query the message is carrying.
    c) Truncated is set to 1 when the message is truncated due to its length longer than the limit of transport medium used. (used in case of UDP). It contains further info regarding status and whether the query was recursive or not.
3) Questions contain values that provide the number of requests that are sent in the query.
4) Answer RRs/ Authority RRs/ Additional RRs: RR stands for Resources Records. They are used to store hostnames, IP addresses and other information in DNS name servers. Queries contain the domain name and type of record (A, AAAA, MX, TXT, etc.) being resolved.

```
Domain Name System (query)
    Transaction ID: 0x98b3
  ∨ Flags: 0x0100 Standard query
        0... .... .... .... = Response: Message is a query
        .000 0... .... .... = Opcode: Standard query (0)
        .... ..0. .... .... = Truncated: Message is not truncated
        .... ...1 .... .... = Recursion desired: Do query recursively
        .... .... .0.. .... = Z: reserved (0)
        .... .... ...0 .... = Non-authenticated data: Unacceptable
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  ∨ Queries
      > youtube.com: type HTTPS, class IN
```

b) TLSv1.3 (Transport Layer Security version 1.3) Protocol -

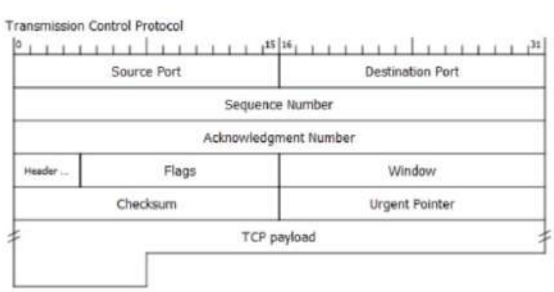


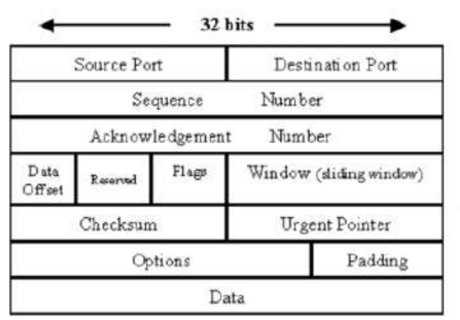
TLSv1.3 is a security protocol designed to facilitate privacy and data security for communication over the network. It is used for encrypting the communication between web applications and servers. The packet structure is as follows:

1) Content Type can be of 4 types - Handshake, Change Cipher Spec, Alert and Application Data
2) Version is the TLS protocol version that the client wants to communicate with the server.
3) Length is the length of the application data being transferred.
4) The transport protocol verifies the integrity of the data by adding a Message Authentication Code (MAC) to the packet. It is based on the shared secret (established by the key exchange), the packet sequence number, and the packet contents. It is calculated before encryption takes place.

2) <u>Transport Layer Protocols:</u>

a) TCP (Transmission Control Protocol) -





TCP is a connection-oriented protocol that defines how to establish and maintain a network connection. The various fields of a TCP packet are as follows:

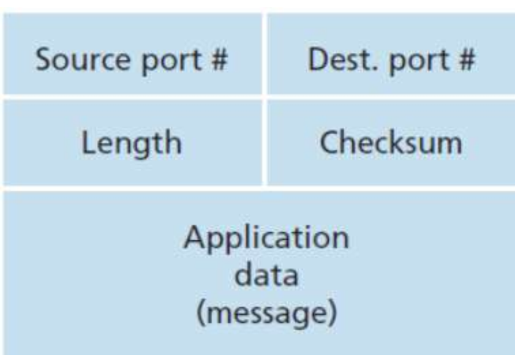
1) Source port is the port which is sending the packets.
2) Destination port is the receiving port.
3) Sequence Number: If SYN bit is set 1, then it is the initial sequence number, else this is the accumulated sequence number of the first data byte of this segment
4) Acknowledgement Number: If it is set to 1, it means that then the value of this field is the next sequence number that the sender of the ACK is expecting.
5) Header Length is the size of the header specified in multiples of 4 bytes.
6) Window Size is the size of the receiver window in bytes used to regulate the amount of data sent.
7) Checksum: This field of 16 bit size is used for the error checking of the header and the data.
8) Options data can be used to include support for special acknowledgment and window scaling algorithms.
9) The urgent pointer points to the sequence number of the octet following the urgent data.

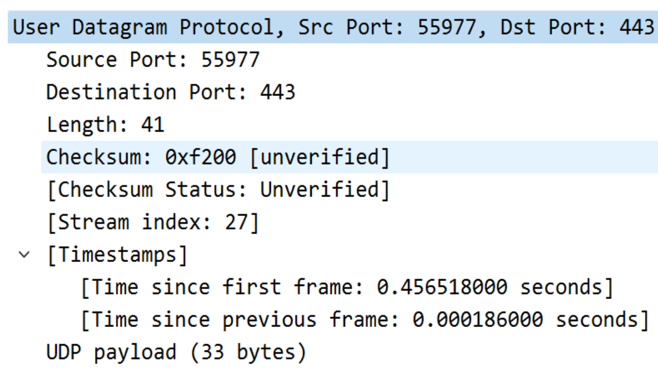
b) QUIC - Quick UDP Internet Protocols

QUIC is built on top of UDP in order to reduce latency compared to that of TCP. The various fields are:

UDP is a connectionless protocol and faster than TCP. Various fields of a UDP packet (header) are:

1) Source Port is the source port number of the packet.
2) Destination Port is the destination port number of the packet.
3) Length is the length of the UDP data and UDP header combined in bytes.
4) Checksum is used to detect errors in header and data if occurred during transmission.

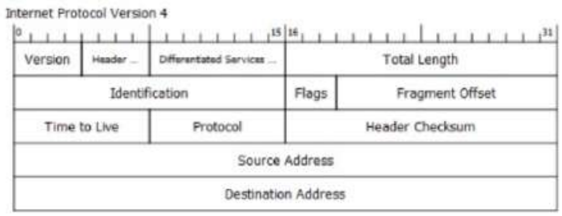


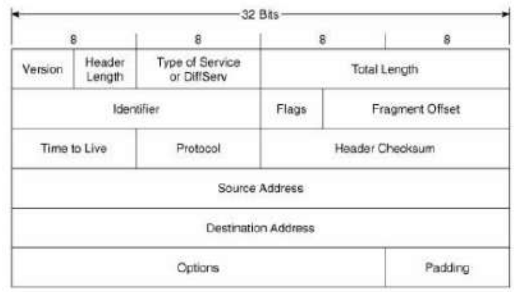


3) <u>Networks Layer:</u>

a) IPv4 (Internet Protocol version 4) -

IPv4 is the fourth version of IP. It is one of the core protocols of standards-based internetworking methods in the internet and other packet-switched networks. It is a connectionless protocol. Various fields in IPv4 datagram header are:

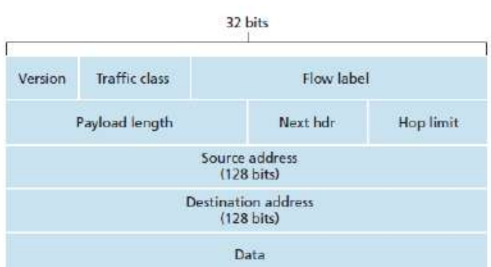




1) Version tells which version we are using. Only IPv4 uses this header.
2) Header length tells the length of the IP header in 32 bits increment.
3) Differentiated Services Field is used for particular quality of service (QoS) that is needed
4) Total Length shows the entire size of the IP packet in bytes.
5) Identification: If the IP packet is fragmented then each fragmented packet will use the same 16 bit identification number to identify to which IP packet they belong to.
6) Flags: These 3 bits are used to identify or control the fragmentation.
7) Time to Live is used to prevent packets from looping around forever.
8) The Protocol field identifies the transport-layer protocol which will interpret the Data section.
9) The Header Checksum field is used to keep the checksum value of the entire header which is then used to check if the packet is received error-free.
10) Options is an optional field, which is used if the value of IHL is greater than 5 and can be used for security or debugging.
11) The source and destination address are 32-bit addresses of the sender (or source) and the receiver (or destination) of the packet.

b) IPv6 (Internet Protocol version 6) -

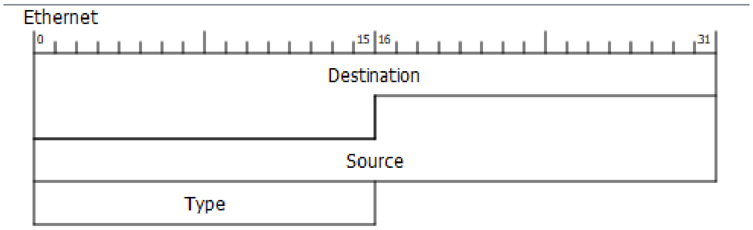
IPv6 is the sixth version of IP. It is one of the core protocols of standards-based internetworking methods in the internet and other packet-switched networks. It is a connectionless protocol but can provide connection oriented service. Various fields in IPv6 datagram header are



1) Version tells us which version we are using, IPv6 has this in order to support rollback to IPv4.
2) Traffic class indicates class/priority of the packet
3) Flow label is used by a source to label belonging to the same flow for better QoS.
4) Payload length indicates the length of the packet payload
5) Next header indicates the type of extension header (if present)
6) Hop limit controls the number of hops over which a datagram can be sent before being discarded.

4) <u>Link Layer: Ethernet II -</u>

Ethernet is used to connect devices in a network and is a popular form of network connection. There is no preamble in the fields shown in Wireshark. The preamble is a physical layer mechanism to help the NIC identify the start of a frame. It carries no useful data and is not received like other fields. The Ethernet frame starts with a 7-Bytes Preamble. This is a pattern of alternative 0's and 1's which indicates starting of the frame and allows sender and receiver to establish bit synchronization. SFD is a 1-Byte field which is always set to 10101011 and signifies the start of the frame. The destination and source address are 6-byte fields which contain the MAC address of the source machine and the machine for which the message is destined. Length gives the length of the frame. Data field contains the actual data which is also known as Payload. The Frame Check Sequence field contains 32-bits hash code of data generated over the remaining fields. If the checksum computed by destination is not the same as sent checksum value, data received is corrupted.



Task 2: Values

Explaining the values that we have obtained

1) **Application layer DNS protocol:**

```
Domain Name System (query)
    Transaction ID: 0x613d
  ˅ Flags: 0x0100 Standard query
        0... .... .... .... = Response: Message is a query
        .000 0... .... .... = Opcode: Standard query (0)
        .... ..0. .... .... = Truncated: Message is not truncated
        .... ...1 .... .... = Recursion desired: Do query recursively
        .... .... .0.. .... = Z: reserved (0)
        .... .... ...0 .... = Non-authenticated data: Unacceptable
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  ˅ Queries
      ˅ youtube.com: type AAAA, class IN
            Name: youtube.com
            [Name Length: 11]
            [Label Count: 2]
            Type: AAAA (IPv6 Address) (28)
            Class: IN (0x0001)
```

The transaction ID is 0x613d - 16-bit identification field. The first flag has a 0 indicating a response, Opcode is 0 which means it's a standard query with no recursion, Truncated is 0 which means the message is not truncated. Z is 0 which means it is reserved. 1 Question is sent in the DNS query segment. There are no RRs currently being stored. Name: youtube.com is the name of the server. Type: AAAA means Query is for the IPv6 address of the server.

2) **Application layer TLSv1.3 protocol:**

```
TLSv1.3 Record Layer: Handshake Protocol: Server Hello
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 122
    Handshake Protocol: Server Hello
        Handshake Type: Server Hello (2)
        Length: 118
        Version: TLS 1.2 (0x0303)
        Random: ffb940df403e1e5e7cb15354dc262b929d4fc7a195ae4ba11bed77e086cf6f64
        Session ID Length: 32
        Session ID: 062acff9b761785d64d5fbc19c0e5c7898014964dea75305ef58544d6e9e0ed9
        Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
        Compression Method: null (0)
        Extensions Length: 46
        Extension: key_share (len=36)
        Extension: supported_versions (len=2)
        [JA3S Fullstring: 771,4865,51-43]
        [JA3S: eb1d94daa7e0344597e756a1fb6e7054]
TLSv1.3 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
    Content Type: Change Cipher Spec (20)
    Version: TLS 1.2 (0x0303)
    Length: 1
    Change Cipher Spec Message
```

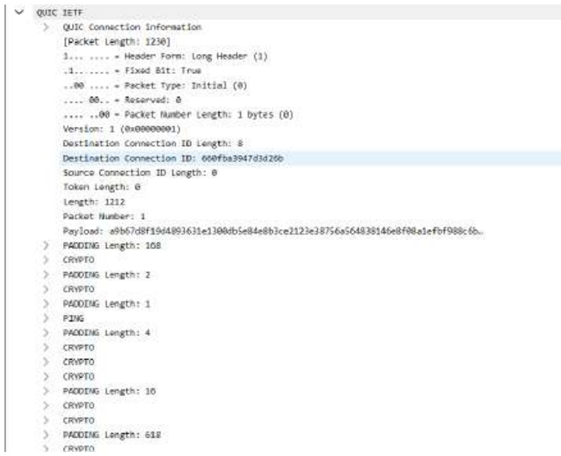
Content type for the first TLSv1.3 is Handshake. Version is TLS 1.2, which is the version the client wants to communicate with. Length is 122, depicting the application data length. There are extensions as well. Session ID is used by the client to identify the session. Random is a 32-byte pseudorandom number that is used in encryption key. Cipher Suite is a list of cipher suites supported by the client.

3) Transport layer TCP protocol:

```
Transmission Control Protocol, Src Port: 443, Dst Port: 54713, Seq: 2441, Ack: 518, Len: 1220
    Source Port: 443
    Destination Port: 54713
    [Stream index: 109]
    [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 1220]
    Sequence Number: 2441    (relative sequence number)
    Sequence Number (raw): 3653434741
    [Next Sequence Number: 3661    (relative sequence number)]
    Acknowledgment Number: 518    (relative ack number)
    Acknowledgment number (raw): 205193181
    0101 .... = Header Length: 20 bytes (5)
    Flags: 0x010 (ACK)
    Window: 261
    [Calculated window size: 66816]
    [Window size scaling factor: 256]
    Checksum: 0xaf1f [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
    [Timestamps]
    [SEQ/ACK analysis]
    TCP payload (1220 bytes)
    [Reassembled PDU in frame: 1992]
    TCP segment data (1220 bytes)
```

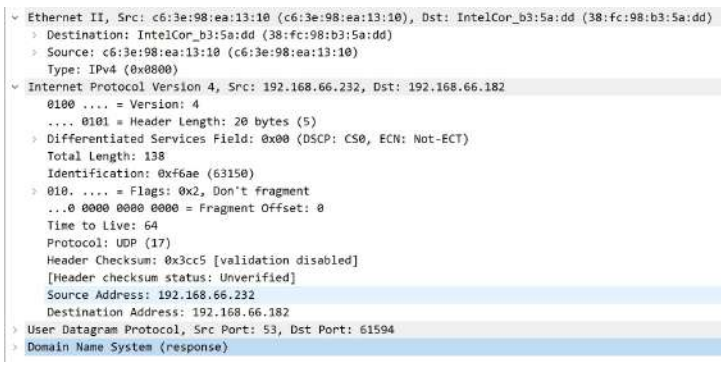
The TCP packet contains the values of the source - 443 and destination - 54713 ports, the sequence and acknowledgement and header length (which is 32 bytes in this case). The PSH and ACK flags are set. The sequence number is 2441 which is a relative sequence number. The ACKflag, which stands for “Acknowledgement”, is used to acknowledge the successful receipt of a packet. Checksum value is 0xaf1f. The window size scaling factor shows the number of leftward bit shifts that should be used for an advertised window size. The urgent pointer is not set. Header length is 20 bytes but stored as 5 (multiple of 4 bytes).

4) Transport layer QUIC protocol:



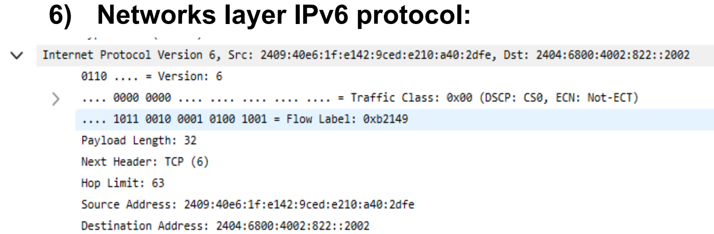
We see that the header is 1 i.e. long header, packet type is initial, version is 1. Destination connection ID length is 8 and ID is seen. Length denotes the length of the message.

5) Networks layer IPv4 protocol:



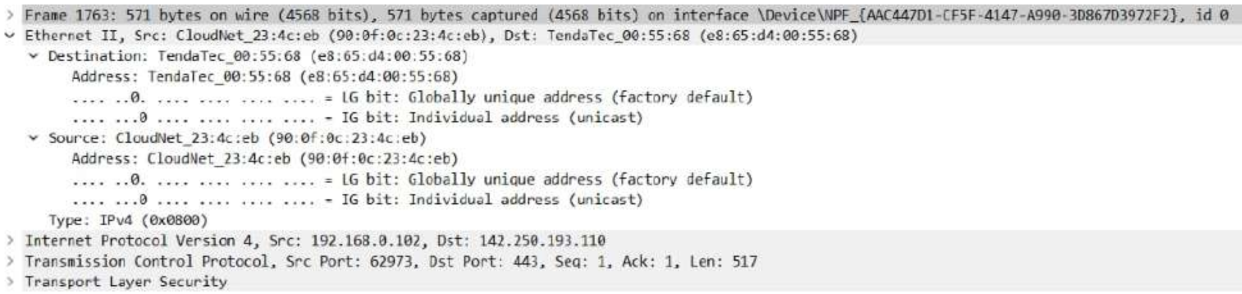
Version is IPv4, thus leading 4 bits are 0100. Header length is 20 bytes but stored as 5 (multiple of 4 bytes) while total datagram size is 864 bytes. Differentiated Services Code Point (DSCP) is a packet header value that can be used to indicate the level of service requested for traffic, such as high priority or best effort delivery. ECN is Explicit Congestion Notification. TTL is 64 means after 64 hops, this packet will be dropped. Don’t fragment flag is set due to which a router which normally would fragment a packet larger than MTU (and potentially deliver it out of order), instead will drop the packet. Upper layer protocol is TCP. Checksum is value 0x3c55. Source and destination IP addresses are also specified.

6) Networks layer IPv6 protocol:



The version is IPv6 i.e. header is 0110. Payload length is 32, Traffic class is 0x00, Flow label is also specified. Hop limit is 63. Source and destination addresses are also mentioned.

7) Link layer Ethernet protocol:



Both the destination and source are mentioned. LG and IG bits are both 0 and the individual address is unicast. 'Type' field is used to indicate which protocol is encapsulated in the payload of the frame. Type of network layer protocol used above is IPv4.

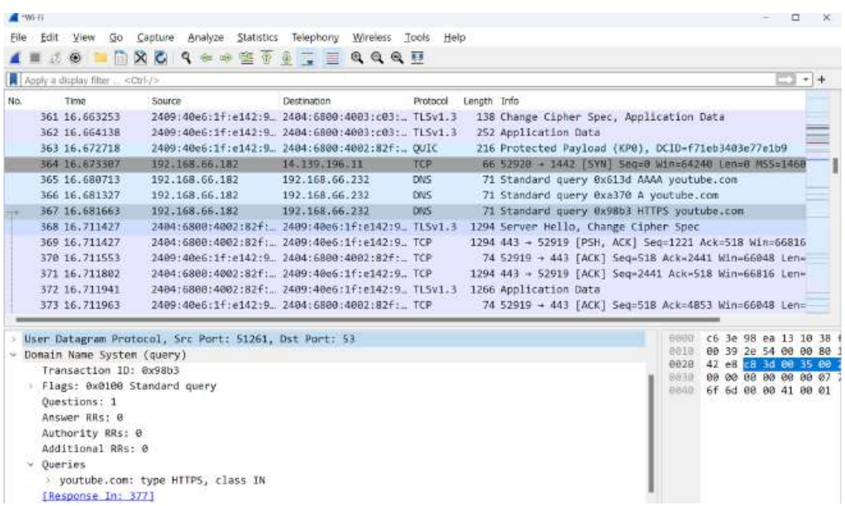
Task 3: Sequence of messages for the functionalities

The following exchange of messages (as can be seen in the picture) was observed on visiting the website.

1) A DNS query was sent to get the IP Address corresponding to the domain www.youtube.com.
2) A TCP connection is initiated by performing a 3-way handshake with the destination server.
3) An HTTP GET request is sent to get the web page and other attachments.
4) The requested data is sent by the server over the established TCP connection.
5) Connection termination on closing browser by sending FIN TCP packets.

DNS Query -

The first packets that are exchanged are DNS packets which are used to identify the IP address of the destination.



3-way handshake (TCP) -

After extracting the IP address using DNS, my pc sets up a TCP connection with the destination. First of all, TCP Handshaking is done using 3-way handshake (SYN, SYN-ACK, ACK). To begin any further communication, both the source and destination must synchronize the Sequence Numbers they both have. So, my PC sends a SYN message with its own sequence number x. Destination replies with a synchronize-acknowledgement (SYN-ACK) message with its own sequence number y, and acknowledgement number x + 1. Then my PC replies with an acknowledgement message (ACK) with acknowledgement number y + 1. So, the handshaking gets succeeded and further transfer of messages starts.

11	8.024917	192.168.43.147	14.139.196.11	TCP	66	49207 → 10443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
12	8.140535	14.139.196.11	192.168.43.147	TCP	66	10443 → 49207 [SYN, ACK] Seq=0 Ack=1 Win=18352 Len=0 MSS=1370 SACK_PERM=1 WS=1024
13	8.140633	192.168.43.147	14.139.196.11	TCP	54	49207 → 10443 [ACK] Seq=1 Ack=1 Win=16384 Len=0

Step 1 (SYN): In the first step, the client establishes a connection with the server, so it sends a segment with SYN (Synchronize Sequence Number) which informs the server that the client is likely to start communication and with what sequence number it starts its segments.

Step 2 (SYN + ACK): Server responds to the client request with SYN-ACK signal bits set. Acknowledgement (ACK) signifies the response of the segment it received and SYN signifies with what sequence number it is likely to start its segments.

Step 3 (ACK): In the final part the client acknowledges the response of the server and they both establish a reliable connection with which they will start the actual data transfer.

TLS Handshake -

After the TCP handshake, TLS handshake is done. After TLS handshake only, the session will start to use TLS encryption. During handshake, first both parties agree on the TLS version they will use and cipher suites they will be using. The authentication of the identity of the server is being done using the server's public key and the SSL certificate authority's digital signature. Then the session keys are generated in order to use symmetric encryption. We can see various messages exchanged like Client Hello, Server Hello, Change Cipher Spec, etc.

Data Transfer - The client using a GET request requests the HTML content After that data exchange happens using TCP protocol with the use of various ACKs and [PSH, ACK]s. TCP's push capability accomplishes two things: The sending application informs TCP that data should be sent immediately. The PSH flag in the TCP header informs the receiving host that the data should be pushed up to the receiving application immediately. So, the server tells the client at various intervals that it has no more data to send and requests an acknowledgement immediately to which the client also replies with an ACK.

Establishing Connection and Message Handshaking -

350	16.574032	2409:40e6:1f:e142:9…	2404:6800:4002:82f:…	TLSv1.3	591	Client Hello
353	16.659968	2404:6800:4003:c03:…	2409:40e6:1f:e142:9…	TLSv1.3	1294	Server Hello, Change Cipher Spec
359	16.660507	2404:6800:4003:c03:…	2409:40e6:1f:e142:9…	TLSv1.3	736	Application Data
361	16.663253	2409:40e6:1f:e142:9…	2404:6800:4003:c03:…	TLSv1.3	138	Change Cipher Spec, Application Data
362	16.664138	2409:40e6:1f:e142:9…	2404:6800:4003:c03:…	TLSv1.3	252	Application Data

Initial YouTube feed Application Data -

17.762186	2404:6800:4002:825:…	2409:40e6:1f:e142:9…	TCP	74	443 → 52925 [ACK] Seq=4912 Ack=592 Win=66816 Len=0
17.763640	2404:6800:4002:825:…	2409:40e6:1f:e142:9…	TLSv1.3	1024	Application Data, Application Data
17.792874	2404:6800:4002:82b:…	2409:40e6:1f:e142:9…	TCP	74	443 → 52924 [ACK] Seq=5872 Ack=1488 Win=68352 Len=0
17.810430	2409:40e6:1f:e142:9…	2404:6800:4002:825:…	TCP	74	52925 → 443 [ACK] Seq=592 Ack=5862 Win=65280 Len=0
17.812743	2404:6800:4002:82b:…	2409:40e6:1f:e142:9…	TLSv1.3	294	Application Data
17.812743	2404:6800:4002:82b:…	2409:40e6:1f:e142:9…	TLSv1.3	212	Application Data
17.812843	2409:40e6:1f:e142:9…	2404:6800:4002:82b:…	TCP	74	52924 → 443 [ACK] Seq=1488 Ack=6230 Win=65792 Len=0
17.815123	2404:6800:4002:82b:…	2409:40e6:1f:e142:9…	TLSv1.3	105	Application Data
17.815123	2404:6800:4002:82b:…	2409:40e6:1f:e142:9…	TLSv1.3	113	Application Data
17.815202	2409:40e6:1f:e142:9…	2404:6800:4002:82b:…	TCP	74	52924 → 443 [ACK] Seq=1488 Ack=6300 Win=65792 Len=0
17.815879	2409:40e6:1f:e142:9…	2404:6800:4002:82b:…	TLSv1.3	113	Application Data

After TLS handshaking is done, the data is transferred using TLS protocol, and is termed as Application Data. The client acknowledges with an ACK message through TCP for each packet it successfully receives. Further these messages are actually covered up in TLS packets so as to avoid any third party to intercept them and analyze them.

```
Transport Layer Security
  TLSv1.3 Record Layer: Application Data Protocol: Hypertext Transfer Protocol
    Opaque Type: Application Data (23)
    Version: TLS 1.2 (0x0303)
    Length: 4714
    Encrypted Application Data: 835d57fe838ee6cda76844c5e8e0f5f69d7340c86c274f7d0bf2508049269ab607ed
    [Application Data Protocol: Hypertext Transfer Protocol]
```

Data Transmission during video playback -

2022	13.915782	2606:4700:e2::ac40:…	2409:40e6:1f:e142:9…	QUIC	793	Protected Payload (KP0)
2023	13.915782	2606:4700:e2::ac40:…	2409:40e6:1f:e142:9…	QUIC	1262	Protected Payload (KP0)
2024	13.915782	2606:4700:e2::ac40:…	2409:40e6:1f:e142:9…	QUIC	122	Protected Payload (KP0)
2025	13.915782	2606:4700:e2::ac40:…	2409:40e6:1f:e142:9…	QUIC	1262	Protected Payload (KP0)
2026	13.915782	2606:4700:e2::ac40:…	2409:40e6:1f:e142:9…	QUIC	980	Protected Payload (KP0)
2027	13.915782	2606:4700:e2::ac40:…	2409:40e6:1f:e142:9…	QUIC	1262	Protected Payload (KP0)
2028	13.915983	2409:40e6:1f:e142:9…	2606:4700:e2::ac40:…	QUIC	105	Protected Payload (KP0),
2029	13.916016	2606:4700:e2::ac40:…	2409:40e6:1f:e142:9…	QUIC	652	Protected Payload (KP0)
2030	13.916016	2404:6800:4009:831:…	2409:40e6:1f:e142:9…	QUIC	1292	Protected Payload (KP0)
2031	13.916016	2606:4700:e2::ac40:…	2409:40e6:1f:e142:9…	QUIC	1262	Protected Payload (KP0)
2032	13.916078	2409:40e6:1f:e142:9…	2606:4700:e2::ac40:…	QUIC	105	Protected Payload (KP0),
2033	13.916142	2409:40e6:1f:e142:9…	2606:4700:e2::ac40:…	QUIC	105	Protected Payload (KP0),
2034	13.916201	2409:40e6:1f:e142:9…	2606:4700:e2::ac40:…	QUIC	105	Protected Payload (KP0),
2035	13.916316	2606:4700:e2::ac40:…	2409:40e6:1f:e142:9…	QUIC	440	Protected Payload (KP0)
2036	13.916316	2606:4700:e2::ac40:…	2409:40e6:1f:e142:9…	QUIC	1262	Protected Payload (KP0)
2037	13.916366	2409:40e6:1f:e142:9…	2404:6800:4009:831:…	QUIC	93	Protected Payload (KP0),
2038	13.916532	2409:40e6:1f:e142:9…	2606:4700:e2::ac40:…	QUIC	105	Protected Payload (KP0),
2039	13.916564	2606:4700:e2::ac40:…	2409:40e6:1f:e142:9…	QUIC	463	Protected Payload (KP0)
2040	13.916609	2409:40e6:1f:e142:9…	2606:4700:e2::ac40:…	QUIC	105	Protected Payload (KP0),
2041	13.916870	2404:6800:4009:831:…	2409:40e6:1f:e142:9…	QUIC	1292	Protected Payload (KP0)
2042	13.917180	2404:6800:4009:831:…	2409:40e6:1f:e142:9…	QUIC	1292	Protected Payload (KP0)

The TCP protocol is better for streaming videos as it continuously communicates with each other to ensure all data packets are received in order. Also, the error is checked and recovery is done on time if required. For transmission in an unreliable network, the TCP streaming protocol is a better choice than the UDP streaming protocol. In a TCP connection, HTTP communication occurs. Here, since the time delay is not an issue in the handling of packets, the TCP easily transfers to HTTP and web browsers to reduce buffer time.

Pause and Resume Video -

7717	167.028418	192.168.0.102	142.250.182.46	TCP	54	65364 → 443 [ACK] Seq=583 Ack=7386 Win=65536 Len=0
9075	201.378798	192.168.0.102	142.250.182.46	QUIC	1292	Initial, DCID=cb48e315d5eaaa95, PKN: 1, CRYPTO, PADDING, PING, CRYPTO, PADDING, CRYPTO, PADDING, CRYPTO, PADDING, CRYPTO, CRYPTO,
9076	201.379695	192.168.0.102	142.250.182.46	QUIC	122	0-RTT, DCID=cb48e315d5eaaa95
9361	201.472354	142.250.182.46	192.168.0.102	QUIC	1292	Initial, SCID=eb48e315d5eaaa95, PKN: 1, ACK, PADDING
9405	201.480267	142.250.182.46	192.168.0.102	QUIC	1292	Protected Payload (KP0)
9406	201.480267	142.250.182.46	192.168.0.102	QUIC	857	Protected Payload (KP0)
9407	201.480402	142.250.182.46	192.168.0.102	QUIC	193	Protected Payload (KP0)
9408	201.480402	142.250.182.46	192.168.0.102	QUIC	66	Protected Payload (KP0)
9409	201.481944	192.168.0.102	142.250.182.46	QUIC	1292	Handshake, DCID=eb48e315d5eaaa95
9410	201.482093	192.168.0.102	142.250.182.46	QUIC	115	Handshake, DCID=eb48e315d5eaaa95
9411	201.482241	192.168.0.102	142.250.182.46	QUIC	73	Protected Payload (KP0), DCID=eb48e315d5eaaa95
9412	201.482822	192.168.0.102	142.250.182.46	QUIC	1288	Protected Payload (KP0), DCID=eb48e315d5eaaa95
9413	201.482904	192.168.0.102	142.250.182.46	QUIC	1288	Protected Payload (KP0), DCID=eb48e315d5eaaa95
9414	201.482956	192.168.0.102	142.250.182.46	QUIC	1292	Protected Payload (KP0), DCID=eb48e315d5eaaa95
9415	201.483000	192.168.0.102	142.250.182.46	QUIC	1292	Protected Payload (KP0), DCID=eb48e315d5eaaa95
9416	201.483040	192.168.0.102	142.250.182.46	QUIC	342	Protected Payload (KP0), DCID=eb48e315d5eaaa95
9583	201.549421	142.250.182.46	192.168.0.102	QUIC	162	Protected Payload (KP0)
9584	201.549421	142.250.182.46	192.168.0.102	QUIC	67	Protected Payload (KP0)
9586	201.550330	142.250.182.46	192.168.0.102	QUIC	71	Protected Payload (KP0)
9587	201.550657	192.168.0.102	142.250.182.46	QUIC	73	Protected Payload (KP0), DCID=eb48e315d5eaaa95
9588	201.553605	142.250.182.46	192.168.0.102	QUIC	67	Protected Payload (KP0)
9799	201.639179	142.250.182.46	192.168.0.102	QUIC	1288	Protected Payload (KP0)
9800	201.639179	142.250.182.46	192.168.0.102	QUIC	1292	Protected Payload (KP0)
9801	201.639713	192.168.0.102	142.250.182.46	QUIC	77	Protected Payload (KP0), DCID=eb48e315d5eaaa95
9802	201.640629	142.250.182.46	192.168.0.102	QUIC	1292	Protected Payload (KP0)
9803	201.640629	142.250.182.46	192.168.0.102	QUIC	1292	Protected Payload (KP0)
9804	201.640629	142.250.182.46	192.168.0.102	QUIC	1292	Protected Payload (KP0)

When the video is paused, the receiving of data packets is stopped. When the video is resumed, handshaking happens and receiving of data packets continues again.

Connection Termination -

1078	12.449916	192.168.0.100	142.250.71.46	TCP	54	53876 → 443 [FIN, ACK] Seq=1 Ack=74 Win=507 Len=0
1079	12.507752	142.250.71.46	192.168.0.100	TCP	54	443 → 53876 [FIN, ACK] Seq=74 Ack=2 Win=346 Len=0
1080	12.507866	192.168.0.100	142.250.71.46	TCP	54	53876 → 443 [ACK] Seq=2 Ack=75 Win=507 Len=0

When the session is closed, TCP termination handshake is performed to close the connection. First Client sends a FIN packet, along with an ACK packet (which was an acknowledgement to an earlier packet). The server then sends an ACK packet as an acknowledgement to the earlier FIN packet. Following this, the server sends its own FIN packet to close the connection. The client receives this packet, and closes the connection after sending an ACK packet. The connection formally closes and all resources on the client side (including port numbers and buffer data) are released. So, the handshake is different from a 3-way handshake. It is actually a pair of 2-way handshakes.

Task 4: Relevance of the protocols used

Accessing any website or online service, whether it's YouTube, Gmail, or Instagram, begins with the essential process of DNS resolution. This process translates user-friendly domain names (such as www.youtube.com) into numerical IP addresses (for instance, 142.250.71.46). This translation is crucial as it enables clients, such as your computer or mobile device, to pinpoint the exact servers hosting the desired content.

Once DNS resolution is complete, the communication protocol used plays a significant role in ensuring a seamless user experience:

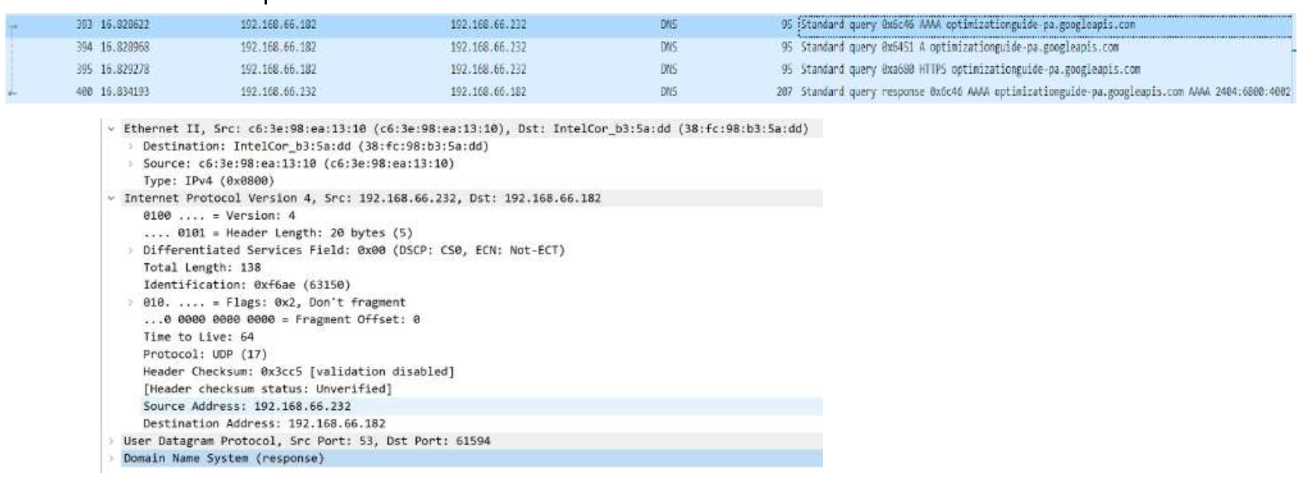
TCP (Transmission Control Protocol): TCP is the preferred choice for establishing reliable and connection-oriented communication between a client's device and servers. This protocol ensures data integrity and order, making it ideal for services like YouTube, where maintaining the sequence of video frames is critical for uninterrupted playback.

UDP (User Datagram Protocol): While TCP is commonly used, some aspects of YouTube and Instagram may utilize UDP , but for email, TCP and SMTP must be used so that there's no loss of data. UDP is a connectionless protocol that can provide faster transmission for real-time applications. It may be employed for streaming video or audio content, where minor data loss can be tolerated without impacting the overall experience.

HTTPS/TLS Encryption: Security is paramount for online communication. HTTPS (Hypertext Transfer Protocol Secure) with TLS (Transport Layer Security) is essential for establishing a secure connection between clients and servers. It encrypts data exchanged during the handshake and subsequent requests, safeguarding user privacy and thwarting potential data interception attempts.

Task 5: Caching

Yes, caching mechanisms were observed, sometimes requests are sent to YouTube's CDN server like googleapis.com which caches the videos for faster response.
DNS responses include Time to Live(TTL) values, which indicate caching. Cached DNS records can reduce the need for repeated DNS resolutions.



Also, we observed that the number of packets that were received when rewinding back to some parts of the video that was already loaded was less which also indicates that packets are cached. This local caching helps ensure smooth video streaming and minimizes re-downloads of segments that haven't changed.

Task 6: Statistics calculation

Time	11:00 PM	3:00 PM	9:00 AM
Network	WiFi through VPN (Room)	LAN (Lab)	LAN (Room)
Throughput (Mpbs)	49	17	11
Round Trip Time (RTT) (ms)	16.48	57.37	53.05
Average Packet Size in bytes	2213	2584	2197

No. of packets lost	8	4	5
No. of TCP packets	21467	19449	20764
No. of UDP packets	5	34	19
No. of responses received with respect to 1 request sent	2.107	1.083	1.273

Number of TCP packets was obtained using the tcp filter in Wireshark.
Number of UDP packets was obtained using the udp filter in Wireshark.
Number of packets lost was obtained using the tcp.analysis.lost_segment filter.
Throughput and RTT were observed from the TCP Stream Graphs option in the Statistics Tab of Wireshark.
Average Packet Size and Number of Responses per request were obtained from the Packet Capture Information and using filters.