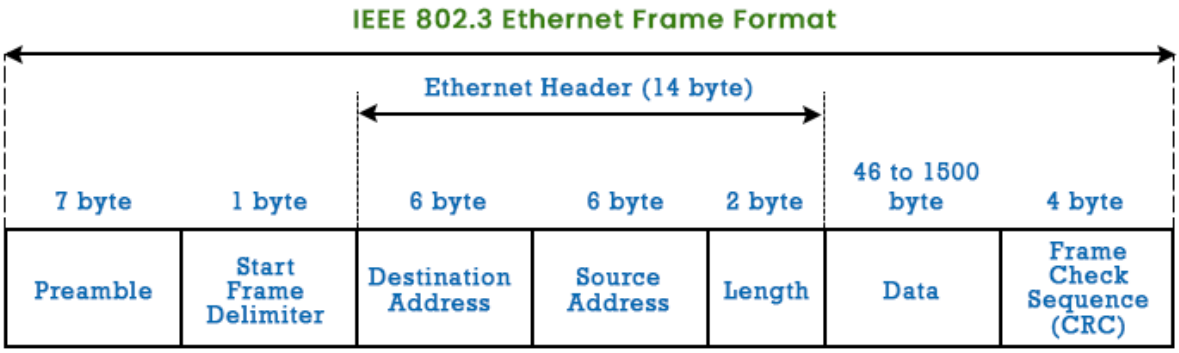
Question 1 and 2:

Ethernet:



Ethernet, a widely used LAN technology, structures its data packets as follows:

- Preamble and a 1-byte Start Frame Delimiter (SFD) synchronize and mark the frame's start.
- Destination Address specifies the recipient's MAC address.
- Source Address identifies the sender with a 6-byte MAC address.
- The Type Field distinguishes Ethernet standards.
- The User Data block carries a payload of up to 1500 bytes.
- The FCS contains a CRC for error detection.

Ethernet's packet format ensures efficient LAN communication.

```
▾ Ethernet II, Src: IntelCor_b1:3b:63 (a8:64:f1:b1:3b:63), Dst: 46:71:16:a6:07:7b (46:71:16:a6:07:7b)
  ▾ Destination: 46:71:16:a6:07:7b (46:71:16:a6:07:7b)
      Address: 46:71:16:a6:07:7b (46:71:16:a6:07:7b)
      .... ..1. .... .... .... .... = LG bit: Locally administered address (this is NOT the factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
  ▾ Source: IntelCor_b1:3b:63 (a8:64:f1:b1:3b:63)
      Address: IntelCor_b1:3b:63 (a8:64:f1:b1:3b:63)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: IPv4 (0x0800)
```

Observations:

Destination Address -> 46:71:16:a6:07:7b

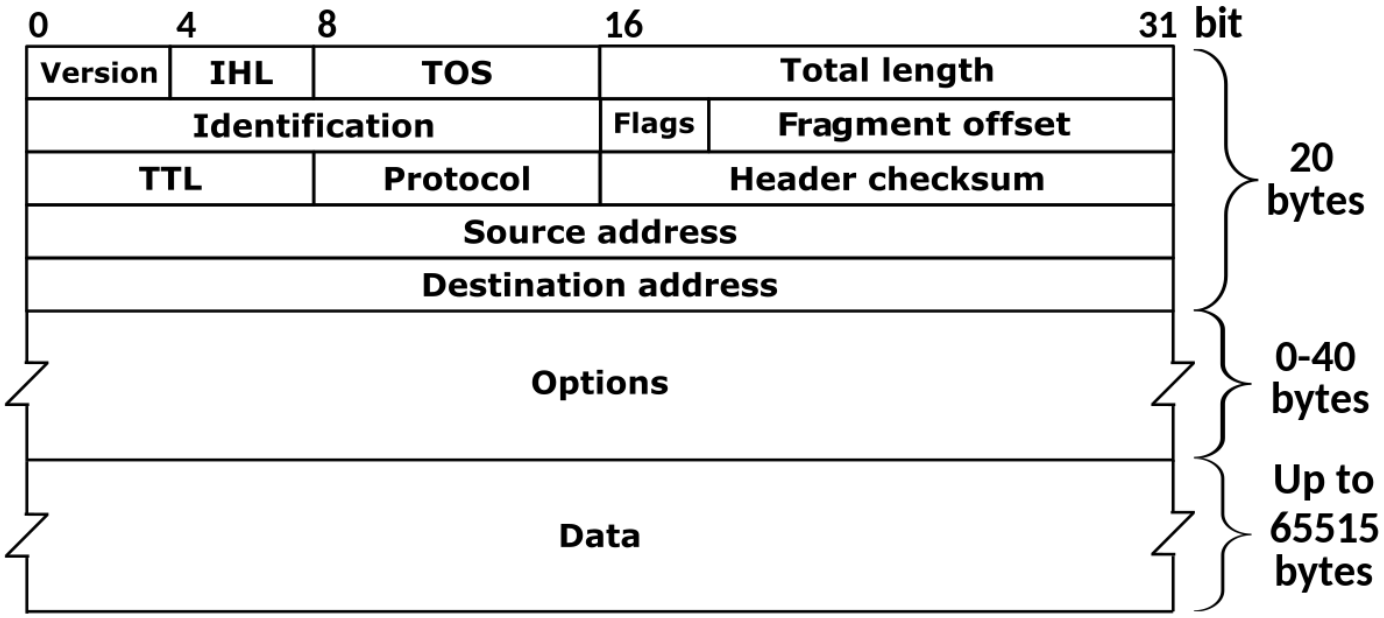
Source Address –> a8:64:f1:b1:3b:63(pc)

In MAC addresses, both the source and destination fields feature two significant bits: the LG (Locally/Global Assigned) bit and the IG (Individual/Group) bit. The LG bit distinguishes between addresses that are vendor-assigned (0) and administratively assigned (1). Meanwhile, the IG bit specifies whether the MAC address is intended for unicast (0) or multicast (1) communication. In the scenario described, both the source and destination MAC addresses have LG and IG bits set to 0, signifying that they are globally unique addresses and designated for unicast communication.

Type indicates the upper layer protocol to be used which is IPv4 in this case.

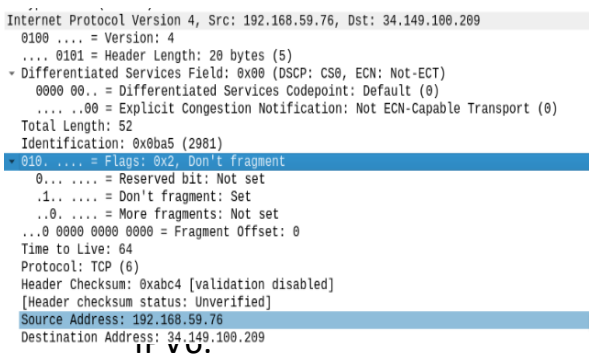
Network Layer:

Internet protocol version 4



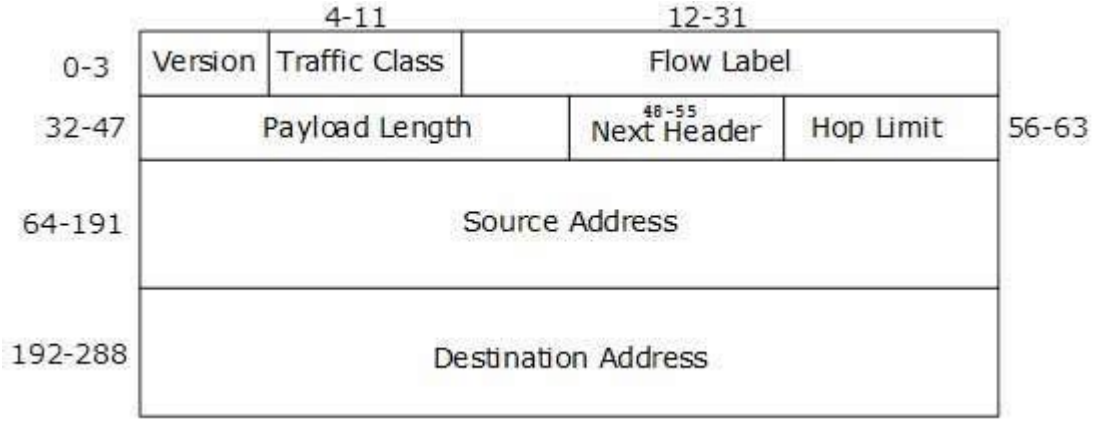
IP (Internet Protocol) facilitates data packet routing from source to destination in networks. The header includes key details:

- Version: Denotes the IP version, often "4" for IPv4.
- Header Length: Specifies the header size.
- Total Length: Indicates overall datagram size in bytes (header + payload).
- DF Bit: Directs routers not to fragment the datagram.
- MF Bit: Signals more fragments if set.
- TTL: Limits the maximum hops.
- Protocol Field: Identifies the destination's higher-layer protocol.
- Source/Destination Address: Holds sender and receiver addresses.
- Options: Used for various purposes like recording routes, source routing, or padding.



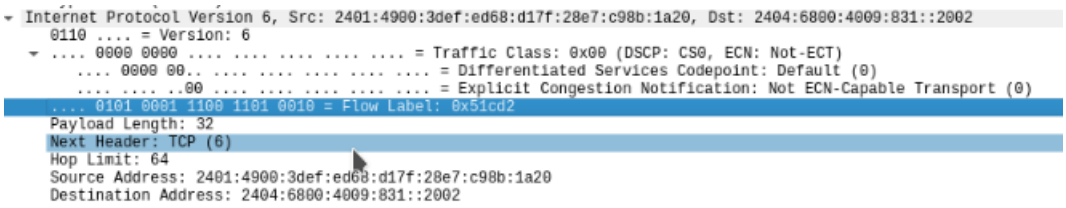
Observations:

Header length is 28 bytes and total length of packet is 52 bytes and DF bit is 1 which means packet should not be fragmented . Time to live is 64 which means packet can make at most 64 hops. TCP is used as the next level protocol for that the source address(pc) is 192.168.59.76 and destination address is 34.149.100.209.



The IPv6 (Internet Protocol version 6) header is a fundamental part of the IPv6 packet structure, providing essential information for the routing and delivery of data over an IPv6 network. Here's a description of the IPv6 header:

- Version (4 bits): The Version field indicates the IP protocol version, which is set to 6 for IPv6.
- Traffic Class (8 bits): Is used to prioritize and classify packets for quality of service (QoS) and traffic management purposes.
- Flow Label (20 bits): The Flow Label field is used to label packets belonging to the same flow or traffic stream, allowing routers to handle them with specific treatment, such as low-latency or high-priority forwarding.
- Payload Length (16 bits): The Payload Length field specifies the length of the IPv6 packet's payload in octets (8-bit bytes).
- Next Header (8 bits): The Next Header field indicates the type of the next protocol or header following the IPv6 header. Common values include ICMPv6 (for control messages), TCP, UDP, and more.
- Hop Limit (8 bits): The Hop Limit field, similar to the Time-to-Live (TTL) field in IPv4, specifies the maximum number of hops (routers) a packet can traverse before being discarded.
- Source Address (128 bits): Source Address field contains the 128-bit IPv6 address.
- Destination Address (128 bits): Destination Address field contains the 128-bit IPv6 address.

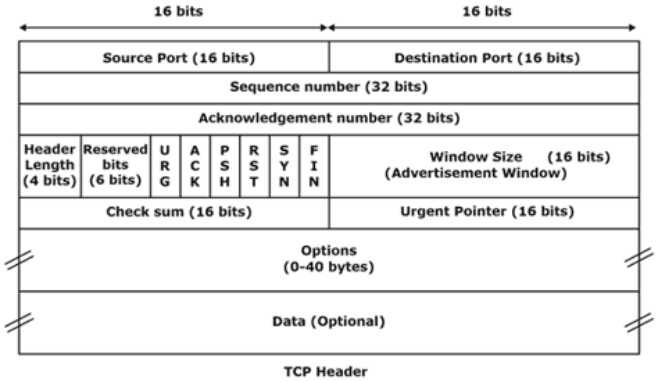


Observations:

Source and destination address are 2401:4900:3def:ed68:d17f:c98b:1a20 and 2404:6800:4009:831:2002 respectively. Payload Length is 32, Hop Limit is 64.Next Header is TCP(4).

Transport Layer:

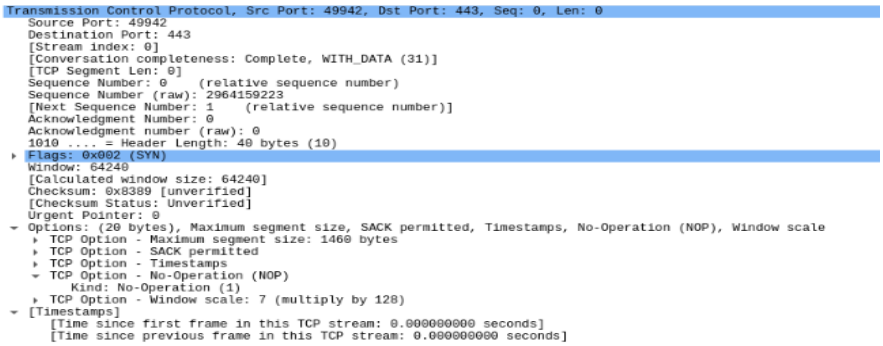
TCP (Transmission Control Protocol)



TCP (Transmission Control Protocol) is a reliable, connection-oriented protocol in the Internet suite. It ensures data integrity, uses port numbers to identify services, manages flow and congestion control, supports full-duplex communication, and handles connection setup and termination with checksum for error detection. It is crucial for reliable data transmission in various applications and the Internet.

The header includes key details:

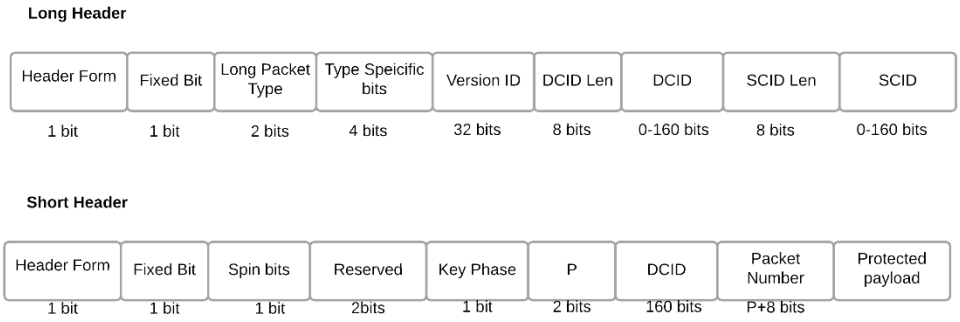
- Source Port and Destination Port: Identify sender and receiver applications.
- Sequence Number: Tracks the first data byte's sequence number.
- Acknowledgement Number: Specifies the expected next data byte.
- Header Length: Indicates TCP header length.
- Flags (e.g., ACK, PSH, SYN): Control packet behaviour.
- Checksum: Ensures payload data integrity.
- Window Size: Shows sender's window for unacknowledged data.
- Urgent Pointer: Identifies urgent data in the current segment.
- Options: Used for various purposes like timestamps, window size extension, and parameter negotiation.



Observations:

Source and destination port numbers are 49942 and 443 respectively. Only SYN flag is set which indicates the initiation of new TCP connection. Header length is 40 bytes. Window size is 64240 and urgent pointer is 0 which means no further bytes is urgent.

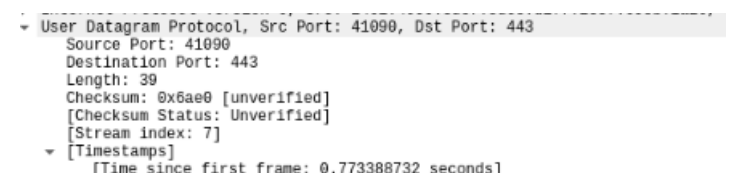
QUIC(Quick UDP Internet Connections):



QUIC is a modern transport protocol over UDP, improving speed, security, and reliability. It reduces latency, supports multiplexing, has built-in error correction, enhances security with encryption, adapts to congestion, allows seamless network changes, compresses headers, and uses UDP for better firewall and NAT compatibility. It is widely used for HTTP/3, offering faster web experiences. It has the header of UDP also.

The header includes key details:

- Header Form (1 bit): It says the header is long (0) or short (1).
- Fixed Bit (1 bit): It indicating the presence of a long header by setting it to 1.
- Spin Bit (1 bit): This bit is reserved for possible future use and is not yet defined in the QUIC specification.
- Reserved Bits (2 bits): Reserved for future use and should be set to zero.
- Packet Number Length (2 bits): This field specifies the length of the packet number that follows.
- Version (32 bits): In the long header, this field represents the QUIC version being used. It is used for version negotiation during connection establishment.
- Destination Connection ID (0 to 160 bits): This field identifies the recipient's connection. It may not be present in short headers.
- Packet Number (Variable Length): The packet number is included to aid in packet sequencing and reordering
- Payload (Variable Length): The payload carries application data or control information, depending on the packet's purpose and type.



Observations:

Source and destination port numbers are 49942 and 443 respectively. Header form is 0 which indicates it is short header. DCID is dd456f2d79573f73 and Fixed bit, Spin bit is set to 1. Payload is 31 bytes and Packet length is 31. Total Header length is 39.

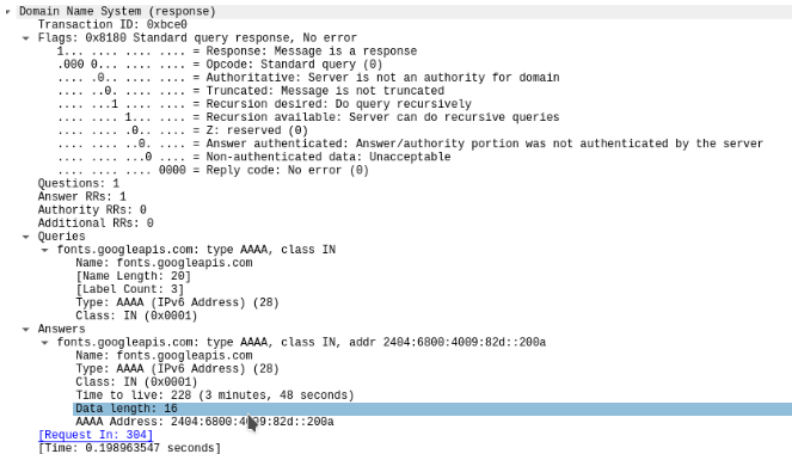
Application Layer:

Domain Name System:

Header Format

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15							
ID							
QR	Opcode	AA	TC	RD	RA	Z	RCODE
QDCOUNT							
ANCOUNT							
NSCOUNT							
ARCOUNT							

The DNS protocol works when your computer sends out a DNS query to a name server to resolve a domain. The first 12 bytes is the header section. ID is a 16-bit identifier assigned by the program that generates any kind of query. QR is one-bit field that specifies whether this message is a query (0), or response (1). OPCODE specifies the kind of query in the message. RCODE or response code for messages specifies error condition. QDCOUNT, ANCOUNT, NSCOUNT, ARCOUNT, specify number of, entries in the question section, resource records in the answer section, name server resource records in the authority records section resource records in the additional records section respectively. QUESTION section contains information about the query that is being made. ANSWER section contains the resource records for the name that was originally queried. AUTHORITY section contains records of other authoritative servers.



Observations:

Transaction ID of this DNS request is 0xbce0 (16-bit). There are many flag values specified of which Opcode specifies that the query type is standard and the first bit (QR) indicates that the type is response. QDCOUNT or Questions in the pictures gives the number of queries which is 1. Answer RR(ANCOUNT) is 1, Authority RR(NSCOUNT) is 0 and Addition RR(ARCOUNT) is also 0. The next two sections list out the queries and answers.

Resource Records contains 4 fields which are:

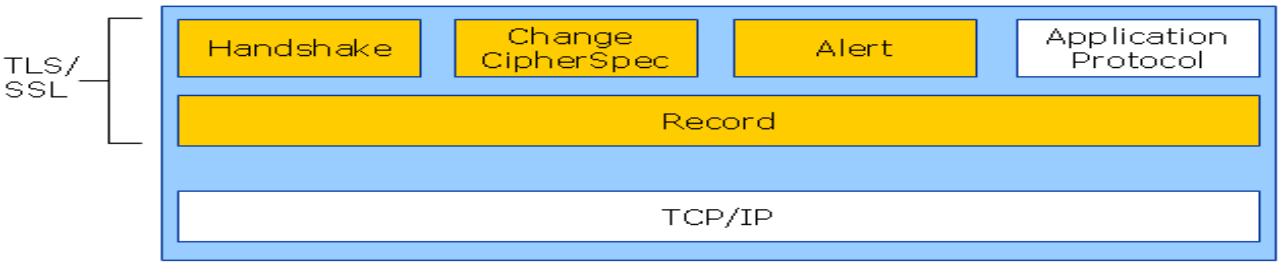
1. Name (Domain Name)
2. Type (16 bits): The Type field specifies the type of resource record and the kind of data it

- A: IPv4 address record
- AAAA: IPv6 address record
- CNAME: Canonical Name record
- MX: Mail Exchanger record
- NS: Name Server record
- TXT: Text record
- SOA: Start of Authority record
- PTR: Pointer record

3. Class (16 bits): The Class field specifies the class of data. Typically, this is set to IN (Internet), which is the most common class for DNS records.

4. TTL (32 bits): The Time-to-Live field indicates how long the resource record can be cached by DNS resolvers or other devices.

Transport Security Layer



The TLS (Transport Layer Security) header is a crucial component of the TLS protocol, which is used to secure data transmitted over a network connection, typically the internet. The TLS header provides essential information for secure communication.

The Header includes key details:

- Version (2 bytes): Specifies the version in use, such as TLS 1.0, 1.1, 1.2, or 1.3.
- Length (2 bytes): Indicates the total record length, including header and payload.
- TLS Record Payload (Variable Length): Contains actual data for transmission, the content of which depends on the TLS record's type.
- Padding: Added for cryptographic purposes, increasing security by obscuring plaintext length.
- Content Type: The Content Type field specifies the type of data contained in the TLS record. Common types include:
    - Handshake: Used for the initial handshake process.
    - Application Data: Contains the application-layer data being transmitted.
    - Alert: Communicates error messages or warnings.
    - Change Cipher Spec: Signals a change in the encryption parameters.

```
▾ Transport Layer Security
  ▾ TLSv1.3 Record Layer: Handshake Protocol: Client Hello
      Content Type: Handshake (22)
      Version: TLS 1.0 (0x0301)
      Length: 512
    ▾ Handshake Protocol: Client Hello
        Handshake Type: Client Hello (1)
        Length: 508
        Version: TLS 1.2 (0x0303)
        Random: bf423ce2c4237ad61de2511a12d488e2e112afd7b70191fa5a1514720a119971
        Session ID Length: 32
        Session ID: 195b97793d60dab0cdaf001a2c00886a4b5d0ce39a9b2c268095a91b2536cf40
        Cipher Suites Length: 34
      ▸ Cipher Suites (17 suites)
        Compression Methods Length: 1
      ▸ Compression Methods (1 method)
        Extensions Length: 401
      ▸ Extension: server_name (len=30)
      ▸ Extension: extended_master_secret (len=0)
      ▸ Extension: renegotiation_info (len=1)
      ▸ Extension: supported_groups (len=14)
      ▸ Extension: ec_point_formats (len=2)
      ▸ Extension: session_ticket (len=0)
      ▸ Extension: application_layer_protocol_negotiation (len=11)
      ▸ Extension: status_request (len=5)
      ▸ Extension: delegated_credentials (len=10)
      ▸ Extension: key_share (len=107)
      ▸ Extension: supported_versions (len=5)
      ▸ Extension: signature_algorithms (len=24)
      ▸ Extension: psk_key_exchange_modes (len=2)
      ▸ Extension: record_size_limit (len=2)
      ▸ Extension: padding (len=128)
        [JA3 Fullstring: 771,4865-4867-4866-49195-49199-52393-52392-49196-49200-49162-49161-49171-49172-156-157-47-53,0-23-65281-10-11-35-16-5-34…
        [JA3: 579ccef312d18482fc42e2b822ca2430]
```

Observations:

Content Type is handshake (Client Hello) and version used is TLS 1.2. Total length is 512 and session id length is 508.

It is used instead of http protocol.

Question 3:

Launching Web Browser:

After opening the browser, it will send some DNS requests and responses will come for that query.

```
 1 0.000000000   192.168.59.76 192.168.59.83 DNS      95 Standard query 0xc6a0 A content-signature-2.cdn.mozilla.net
 2 0.000016270   192.168.59.76 192.168.59.83 DNS      95 Standard query 0xc9ae AAAA content-signature-2.cdn.mozilla.net
 3 0.043348492   192.168.59.76 192.168.59.83 DNS      85 Standard query 0xc733 A push.services.mozilla.com
 4 0.043365873   192.168.59.76 192.168.59.83 DNS      85 Standard query 0x354c AAAA push.services.mozilla.com
 5 0.108233358   192.168.59.76 192.168.59.83 DNS      97 Standard query 0x34a3 A firefox.settings.services.mozilla.com
 6 0.108253936   192.168.59.76 192.168.59.83 DNS      97 Standard query 0x20a6 AAAA firefox.settings.services.mozilla.com
 7 0.355099789   192.168.59.83 192.168.59.76 DNS     273 Standard query response 0xc733 A push.services.mozilla.com CNAME autopush.prod.mozaws.net A 34.117.65.55 NS ns-1260.awsdns-29.org NS ns-1986.awsdns-56.co.uk NS ns-377.awsdns-47.com NS ns-614.
 8 0.355099791   192.168.59.83 192.168.59.76 DNS     175 Standard query response 0x34a3 A firefox.settings.services.mozilla.com CNAME prod.remote-settings.prod.webservices.mozgcp.net A 34.149.100.209
 9 0.355100096   192.168.59.83 192.168.59.76 DNS     205 Standard query response 0x354c AAAA push.services.mozilla.com CNAME autopush.prod.mozaws.net SOA ns-1260.awsdns-29.org
10 0.355100097   192.168.59.83 192.168.59.76 DNS     249 Standard query response 0x20a6 AAAA firefox.settings.services.mozilla.com CNAME prod.remote-settings.prod.webservices.mozgcp.net SOA ns-cloud-a1.googledomains.com
11 0.355100125   192.168.59.83 192.168.59.76 DNS     249 Standard query response 0xc6a0 A content-signature-2.cdn.mozilla.net CNAME content-signature-chains.prod.autograph.services.mozaws.net CNAME prod.content-signature-chains.prod.webservices.moz
```

Three way hand shaking:

When the application is launched two handshaking procedure takes place, first 3-way TCP handshaking (first 3 messages) and then TLS Handshaking (last 3 messages). TCP 3-way Handshake: This process is used to make a connection between the client (PC) and the server (outlook.office.com) in a TCP/IP network. It is a 3-step process between port 49942 of client and port 443 of the server.

Step 1: SYN: The client sends a segment to the server with SYN (Synchronize Sequence Number) which is the initial sequence number it plans to use and informs the server that client is likely to start communication.

Step 2: SYN, ACK : The server sends a response with SYN-ACK bit set. ACK (Acknowledgement) indicates that the server has acknowledged the client's sequence number and SYN signifies server's sequence number with which it is likely to start with.

Step 3: ACK : The client then sends a message with ACK bit set, and acknowledges the server's response. The client and server establish a reliable connection for actual data transfer.

TLS Handshake: TLS handshake is used to make the connection secure. First the TLS protocol sends 'Client Hello' message to initiate a session with the server. The server responds with a 'Server Hello' message containing the server certificate which is used primarily for authentication, cipher suite requirements, and randomly generated data for creating session keys. The client responds with a client key and a secure connection is established between the client and the server.

```
25 0.504471469  34.149.100.209   192.168.59.76     TLSv1.3   1454 Server Hello, Change Cipher Spec
26 0.504483910  192.168.59.76    34.149.100.209    TCP         66 49942 → 443 [ACK] Seq=518 Ack=1389 Win=63104 Len=0 TSval=1164564363 TSecr=1536175219
27 0.505875348  34.149.100.209   192.168.59.76     TCP       1454 443 → 49942 [PSH, ACK] Seq=1389 Ack=518 Win=66816 Len=1388 TSval=1536175219 TSecr=1164564283 [TCP segment of a reassembled PDU]
28 0.505890295  192.168.59.76    34.149.100.209    TCP         66 49942 → 443 [ACK] Seq=518 Ack=2777 Win=63104 Len=0 TSval=1164564365 TSecr=1536175219
29 0.517821395  34.149.100.209   192.168.59.76     TLSv1.3   1845 Application Data
```

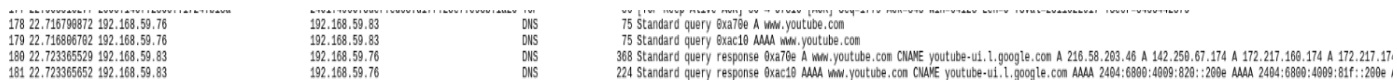
Keep-Alive:

If nothing is performed then there will be exchange of TCP(keep-alive) packets to keep the TCP connections alive.

```
166 10.644680726 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2600:140f:2800::1724:b18a TCP  86 [TCP Keep-Alive] 57812 → 80 [ACK] Seq=421 Ack=890 Win=64128 Len=0 TSval=3453452775 TSecr=2311001488
167 10.673027313 2600:140f:2800::1724:b18a 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TCP  86 [TCP Keep-Alive ACK] 80 → 57812 [ACK] Seq=890 Ack=422 Win=64128 Len=0 TSval=2311011534 TSecr=3453442755
168 10.744665844 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2600:140f:2800::1724:b18a TCP  86 [TCP Keep-Alive] 57816 → 80 [ACK] Seq=842 Ack=1779 Win=64128 Len=0 TSval=3453452875 TSecr=2311001610
169 10.767216304 2600:140f:2800::1724:b18a 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TCP  86 [TCP Keep-Alive ACK] 80 → 57816 [ACK] Seq=1779 Ack=843 Win=64128 Len=0 TSval=2311011634 TSecr=3453442875
170 20.884543673 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2600:140f:2800::1724:b18a TCP  86 [TCP Keep-Alive] 57816 → 80 [ACK] Seq=842 Ack=1779 Win=64128 Len=0 TSval=3453463015 TSecr=2311011634
171 20.884577092 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2600:140f:2800::1724:b18a TCP  86 [TCP Keep-Alive] 57812 → 80 [ACK] Seq=421 Ack=890 Win=64128 Len=0 TSval=3453463015 TSecr=2311011534
172 21.898006220 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2600:140f:2800::1724:b18a TCP  86 [TCP Keep-Alive] 57812 → 80 [ACK] Seq=421 Ack=890 Win=64128 Len=0 TSval=3453464029 TSecr=2311011534
173 21.898040789 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2600:140f:2800::1724:b18a TCP  86 [TCP Keep-Alive] 57816 → 80 [ACK] Seq=842 Ack=1779 Win=64128 Len=0 TSval=3453464029 TSecr=2311011634
174 22.063509964 2600:140f:2800::1724:b18a 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TCP  86 [TCP Keep-Alive ACK] 80 → 57812 [ACK] Seq=890 Ack=422 Win=64128 Len=0 TSval=2311021799 TSecr=3453442755
175 22.063509961 2600:140f:2800::1724:b18a 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TCP  86 [TCP Keep-Alive ACK] 80 → 57816 [ACK] Seq=1779 Ack=843 Win=64128 Len=0 TSval=2311021794 TSecr=3453442875
176 22.063510299 2600:140f:2800::1724:b18a 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TCP  86 [TCP Keep-Alive ACK] 80 → 57812 [ACK] Seq=890 Ack=422 Win=64128 Len=0 TSval=2311022912 TSecr=3453442755
177 22.063510277 2600:140f:2800::1724:b18a 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TCP  86 [TCP Keep-Alive ACK] 80 → 57816 [ACK] Seq=1779 Ack=843 Win=64128 Len=0 TSval=2311022917 TSecr=3453442875
```

Launching YouTube:

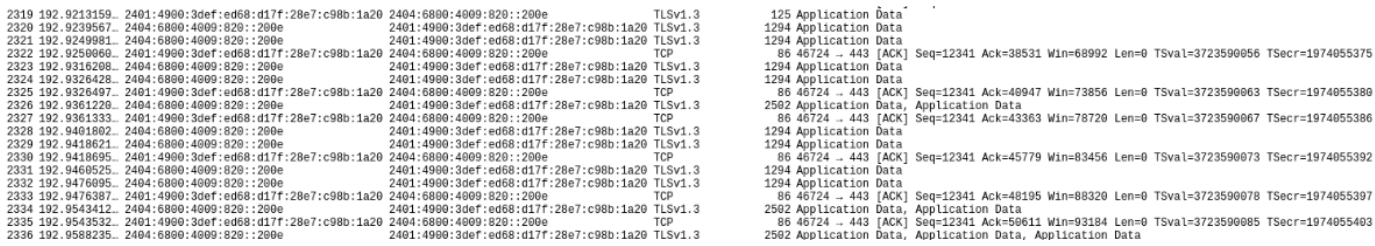
The browser will send the DNS queries(one A and one AAA type) to find the IP address of the you tube.



Afterwards handshaking mechanism will takes place in QUIC and TCP protocols.

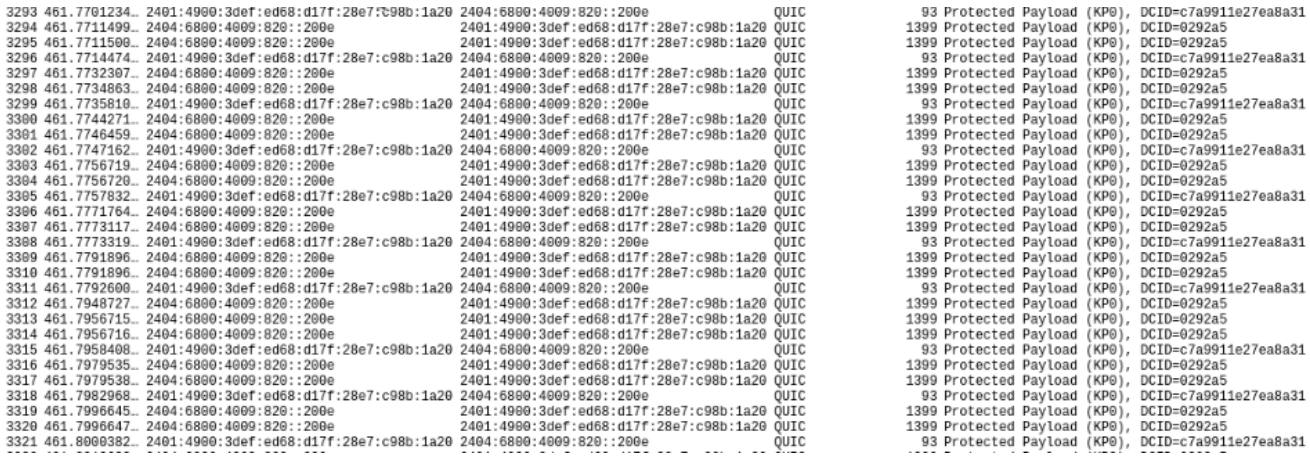
Search:

If we search for a video the data, then the webpage is requested by TLSv 1.3. The TLS (Transport Layer Security) header is a crucial component of the TLS protocol, which is used to secure data transmitted over a network connection, typically the internet. The TLS header provides essential information for secure communication.



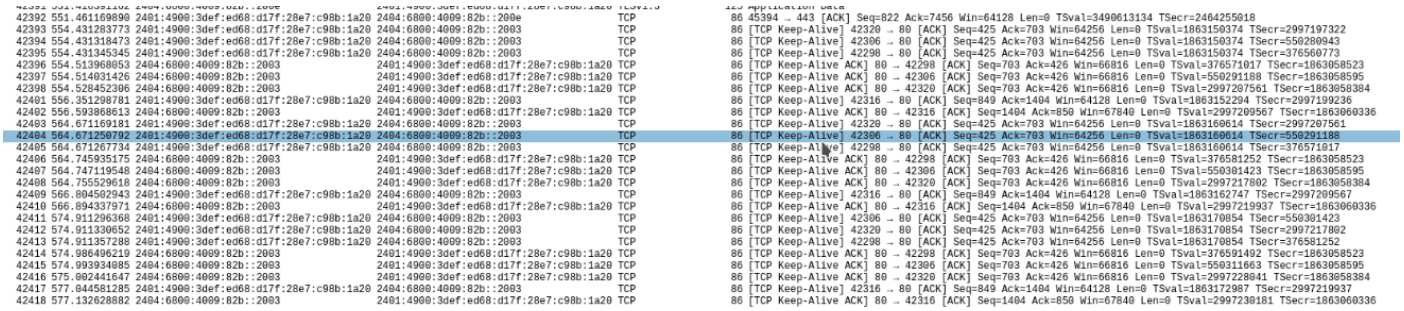
Play:

The content is transported using QUIC protocol and it is encrypted to prevent from sniffing. QUIC is preferred than TCP due to its less delay. The quality of the content depends on the available bandwidth. Average packet length is above 1300 as per observations. It can be changed dynamically by DASH, or Dynamic Adaptive Streaming over HTTP, is a streaming protocol used for delivering multimedia content, such as video and audio, over the internet. DASH is designed to provide a high-quality viewing experience by adapting the streaming bitrate and quality to match the viewer's network conditions.



Pause:

Since when we pause no requests are performed by the browser. To maintain the persistent connection it will exchange the TCP (keep alive packets).



Question 4:

TCP:

Majorly TCP protocol is used for data communications between the client and the outlook server because:
o Transmission control protocol is a reliable protocol i.e. it ensures data which is sent reaches the destination successfully. When a sender doesn't get an acknowledgement after a certain period of time, it will assume that the packet got lost on its way. So, it will send it again. TCP ensures the ordered delivery of packets. Although packets may come out of order, TCP rearranges them before sending them to application. TCP also ensures proper error handling and flow control mechanisms to minimize the error loss rate as we cannot afford any data loss for communication over email.

QUIC:

YouTube should have less latency and high throughout as its primary concerns and secondary concerns are reliability and data security.

YouTube uses QUIC (Quick UDP Internet Connections) for several reasons (**PLAY**):

- Faster Streaming: QUIC reduces latency and connection setup time, enabling faster video streaming by combining the handshake and encryption setup.
- Improved Reliability: QUIC's built-in error correction helps recover lost packets without retransmissions, reducing buffering and interruptions.
- Adaptive Streaming: QUIC's adaptability adjusts video quality to match viewers' network conditions, ensuring a smooth viewing experience.
- Security: QUIC provides encryption by default, enhancing data privacy and security for YouTube viewers.
- Compatibility: QUIC works well with YouTube's large-scale video distribution, optimizing performance across various devices and network types.

TLS:

TLSv (Transport Layer Security version) plays a crucial role in YouTube's security and privacy. When you visit YouTube's website or use its mobile app, TLSv is responsible for encrypting the communication between

your device and YouTube's servers. This encryption ensures that your interactions with YouTube, including video streaming, searches, and account information, are protected from eavesdropping and tampering.

Here's how TLSv benefits YouTube:

- Data Privacy: TLSv encrypts data in transit, safeguarding your sensitive information like login credentials, video preferences, and comments from unauthorized access.
- Content Security: It prevents attackers from intercepting and altering the content you receive from YouTube, ensuring that the videos and ads you see are genuine and unaltered.
- Authentication: TLSv helps verify that you are connecting to the genuine YouTube servers, protecting you from phishing and man-in-the-middle attacks.
- Integrity: It ensures that the data transmitted between your device and YouTube's servers remains intact and unmodified during transit.

DNS:

DNS protocol is used to map domain names to IP addresses, which indicate the server address to which the client has to connect. It allows users to have human-readable domains while ensuring a mapping to the IP addresses corresponding to the domain names.

IPv4:

Internet Protocol version 4 is a connection less protocol which enables data communication over packet switched networks like the internet. It is generally used with the TCP protocol as TCP is only compatible with IP at network Layer. IP is neither reliable nor guarantees ordered data transfer therefore TCP is needed to supplement these shortcomings of IP protocol.

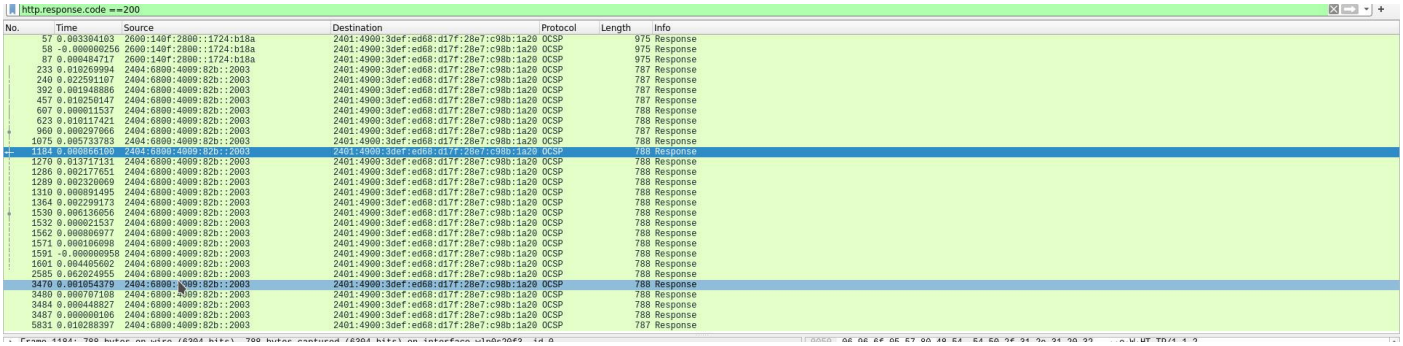
Ethernet II:

Ethernet is the most widely used data link layer protocol. It is preferred over other protocols because of its reliable data transfer, high speed and security. It involves proper error handling and flow control mechanisms for error handling along with CRC for error detection and preamble for synchronization.

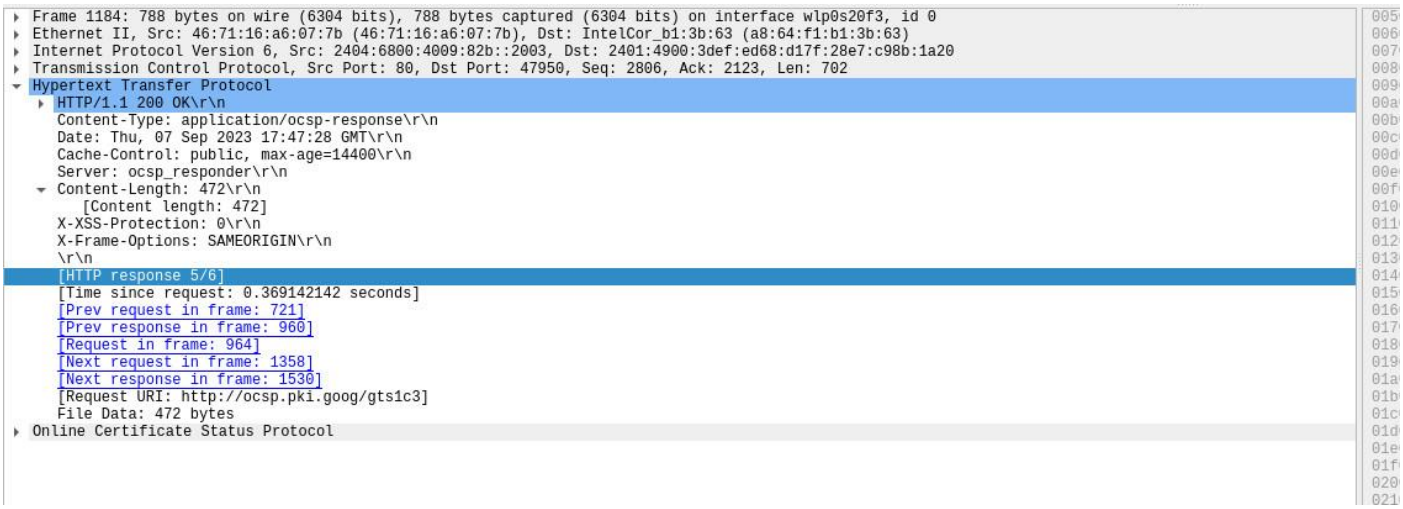
IPv6:

IPv6 is instrumental in YouTube's efforts to maintain a global and efficient content delivery platform. It allows YouTube to address the challenges of scalability and network efficiency while ensuring that users can access their vast library of videos on a variety of devices, including those that rely on IPv6 connectivity.

Question 5:



To narrow down the captured traffic and focus on caching-related packets, we can use display



filters in Wireshark. We can create custom display filters using Wireshark's filtering language. For caching-related traffic, you might use filters like http.cache or http.response.code == 304 or http.response.code == 200 to focus on cached responses.

Once we have applied the appropriate filters, we can select individual packets from the captured traffic and inspect their details. Look for information related to caching in the packet details. This information might include HTTP headers like Cache-Control, ETag, Last-Modified, or other caching-related headers. From the above pictures we can check that ther are indeed headers like cache-control which can prove that therw is indeed a caching mechanism.

Analyzeing the behavior of caching by examining the response codes and caching headers in the captured packets we can look for indications of cache hits (304 Not Modified) and cache misses (200 OK). we can also analyze the freshness of cached content by checking the Age header.

But since YouTube is a dynamic webpage almost all cache access get htpp requests are cache misses. But in static websites caching can be seen more.

Question 6:

	1:00 AM (Lohit Hostel)	7:00 PM (Lohit Hostel)	12:00 PM (Lohit Hostel)
Throughput	114 KB/s	441 KB/s	355 KB/s
RTT	16.9 ms	15.1ms	14.7ms
Packet Size	1228 Bytes	1169 Byte	1245 Bytes

# Packets lost	0	0	0
# UDP Packets	75963(95.2%)	66040(93.9%)	61093(94.4%)
# TCP Packets	3731(4.71%)	3492(5.0%)	3612(5.6%)
# Responses per Request Sent	10.29	6.09	10.48

Throughput ,Avg Packet Size can be directly checked through wireshark

Packets lost finally are 0 but in between there are retransmissions due to packet dropping, packet duplication etc. for 1 AM there are 13 of them for 7 PM there are 19 of them and for 12 PM there are 18 of them

#UDP and TCP packets and RTT can be Directly checked through adding filters.

#Responces to Request send are calculated by filtering the soutce and destination ports.