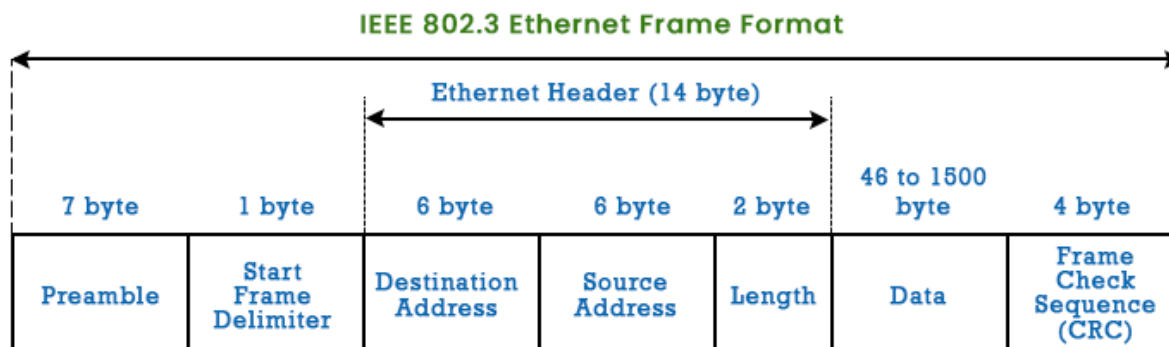


## Question 1 and 2:

### 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

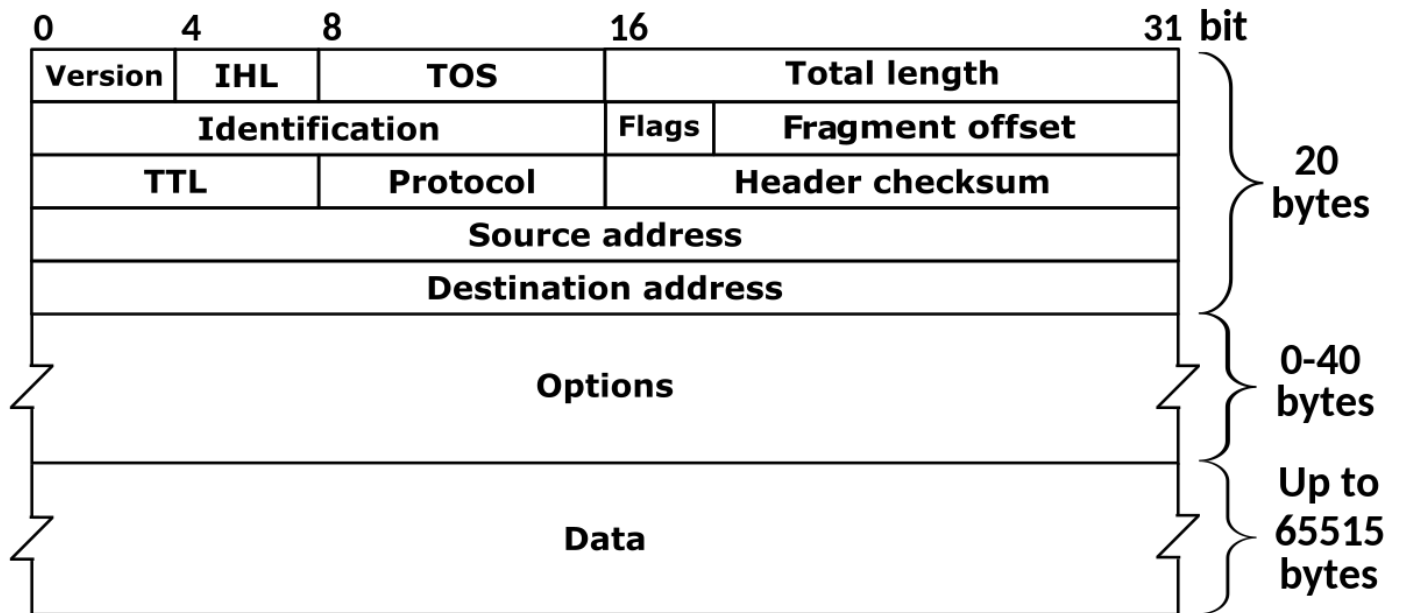
Source Address -> a8:64:f1:b1:3b:63(pc)

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:

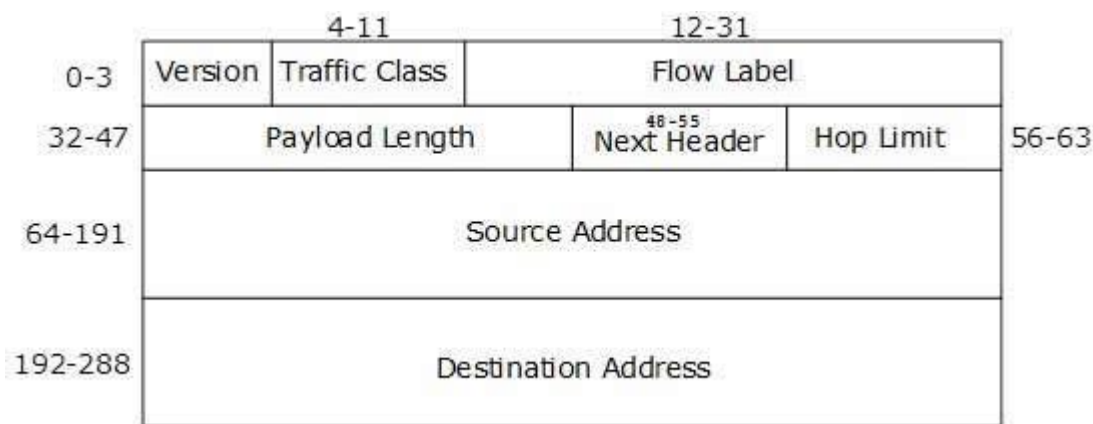
- Version: Denotes the IP version, often "4" for IPv4.
- Header Length: Specifies the header size.
- Total Length: Indicates overall datagram size in bytes (header + payload).
- DF Bit: Directs routers not to fragment the datagram.
- MF Bit: Signals more fragments if set.
- TTL: Limits the maximum hops.
- 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.

```
Internet Protocol Version 4, Src: 192.168.59.76, Dst: 34.149.100.209
0100 .... = Version: 4
.... 0101 = Header Length: 28 bytes (5)
+ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  0000 00.. = Differentiated Services Codepoint: Default (0)
  .... 00.. = Explicit Congestion Notification: Not ECN-Capable Transport (0)
Total Length: 52
Identification: 0x0ba5 (2981)
+ 010. .... = Flags: 0x2, Don't fragment
  0... .... = Reserved bit: Not set
  .1... .... = Don't fragment: Set
  ..0. .... = More fragments: Not set
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 64
Protocol: TCP (6)
Header Checksum: 0xab04 [validation disabled]
[Header checksum status: Unverified]
Source Address: 192.168.59.76
Destination Address: 34.149.100.209
```

ii v u.



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.

```

Internet Protocol Version 6, Src: 2401:4900:3def:ed68:d17f:28e7:c98b:1a20, Dst: 2404:6800:4009:831::2002
  0110 .... = Version: 6
  .... 0000 0000 .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
  .... 0000 00.. .... = Differentiated Services Codepoint: Default (0)
  .... ..00 .... = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  .... 0101 0001 1100 1101 0010 = Flow Label: 0x51cd2
  Payload Length: 32
  Next Header: TCP (6)
  Hop Limit: 64
  Source Address: 2401:4900:3def:ed68:d17f:28e7:c98b:1a20
  Destination Address: 2404:6800:4009:831::2002

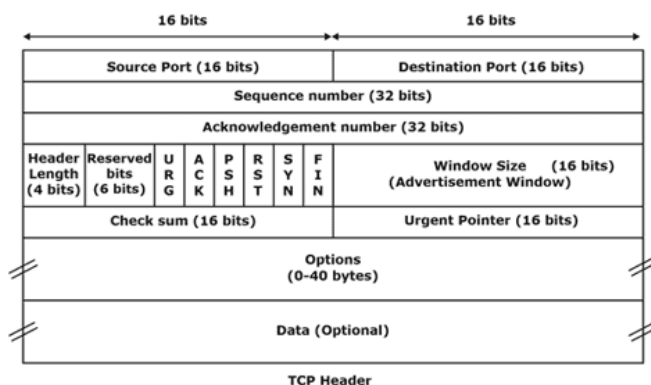
```

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.

```

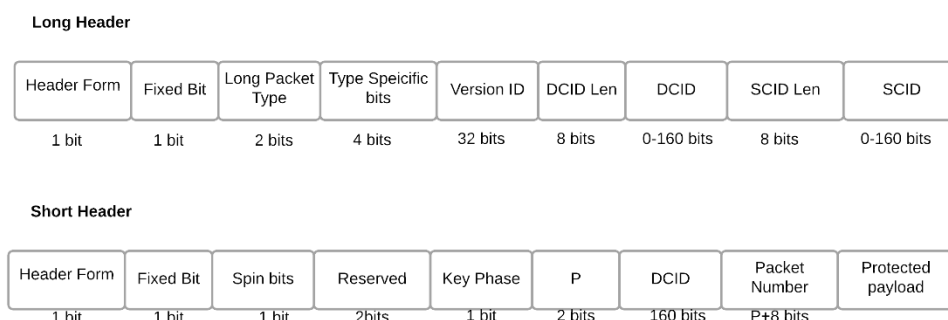
Transmission Control Protocol, Src Port: 49942, Dst Port: 443, Seq: 0, Len: 0
Source Port: 49942
Destination Port: 443
[Stream index: 0]
[Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 0]
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 2964159223
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 0 (relative sequence number)
Acknowledgment number (raw): 0
1010 .... = Header Length: 40 bytes (10)
  Flags: 0x002 (SYN)
    Window: 64240
      [Calculated window size: 64240]
      Checksum: 0x8389 [unverified]
      [Checksum Status: Unverified]
      Urgent Pointer: 0
    Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale
      TCP Option - Maximum segment size: 1460 bytes
      TCP Option - SACK permitted
      TCP Option - Timestamps
      TCP Option - No-Operation (NOP)
        Kind: No-Operation (1)
      TCP Option - Window scale: 7 (multiply by 128)
    [Timestamps]
      [Time since first frame in this TCP stream: 0.000000000 seconds]
      [Time since previous frame in this TCP stream: 0.000000000 seconds]

```

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.

```

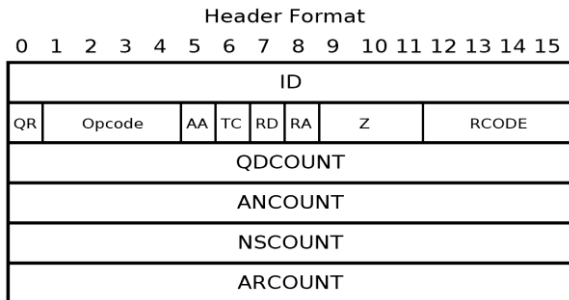
User Datagram Protocol, Src Port: 41090, Dst Port: 443
Source Port: 41090
Destination Port: 443
Length: 39
Checksum: 0x6ae0 [unverified]
[Checksum Status: Unverified]
[Stream index: 7]
[Timestamps]
[Time since first frame: 0.773388732 seconds]
  
```

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:



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.

```

Domain Name System (response)
Transaction ID: 0xbce0
Flags: 0x8180 Standard query response, No error
 1... .. = Response: Message is a response
.000 0... .. = Opcode: Standard query (0)
... .. = Authoritative: Server is not an authority for domain
... .. = Truncated: Message is not truncated
... ..1... .. = Recursion desired: Do query recursively
... ..1... .. = Recursion available: Server can do recursive queries
... ..0... .. = Z: reserved (0)
... ..0... .. = Answer authenticated: Answer/authority portion was not authenticated by the server
... ..0... .. = Non-authenticated data: Unacceptable
... ..0000 = Reply code: No error (0)

Questions: 1
Answer RRs: 1
Authority RRs: 0
Additional RRs: 0
Queries
  fonts.googleapis.com: type AAAA, class IN
    Name: fonts.googleapis.com
    [Name Length: 20]
    [Label Count: 3]
    Type: AAAA (IPv6 Address) (28)
    Class: IN (0x0001)
Answers
  fonts.googleapis.com: type AAAA, class IN, addr 2404:6800:4009:82d::200a
    Name: fonts.googleapis.com
    Type: AAAA (IPv6 Address) (28)
    Class: IN (0x0001)
    Time to live: 228 (3 minutes, 48 seconds)
    Data length: 16
    AAAA Address: 2404:6800:4009:82d::200a
[Request In: 304]
[Time: 0.198963547 seconds]
  
```

### Observations:

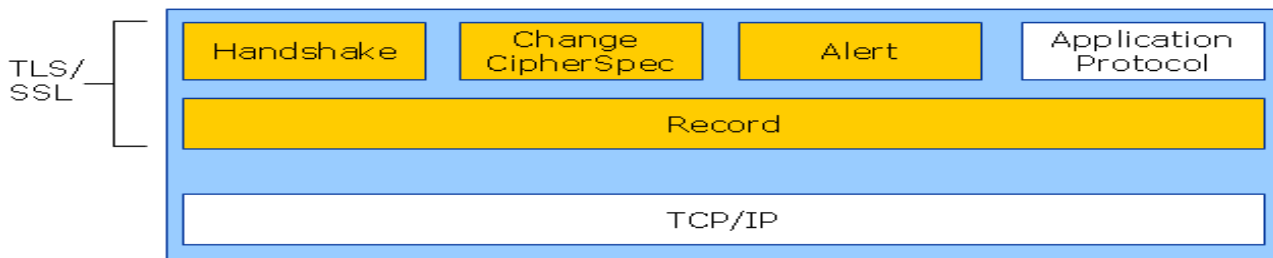
Transaction ID of this DNS request is 0xbce0 (16-bit). There are many flag values specified of which Opcode specifies that the query type is standard and the first bit (QR) indicates that the type is response. QDCOUNT or Questions in the pictures gives the number of queries which is 1. Answer RR(ANCOUNT) 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
  - NS: Name Server record
  - AAAA: IPv6 address record
  - TXT: Text record
  - CNAME: Canonical Name record
  - SOA: Start of Authority record
  - MX: Mail Exchanger 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: 579ccefc312d18482fc42e2b822ca2430]

```



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
2 0.000016270 192.168.59.76 192.168.59.83 DNS
3 0.043340492 192.168.59.76 192.168.59.83 DNS
4 0.043358073 192.168.59.76 192.168.59.83 DNS
5 0.108233358 192.168.59.76 192.168.59.83 DNS
6 0.108253936 192.168.59.76 192.168.59.83 DNS
7 0.355099789 192.168.59.83 192.168.59.76 DNS
8 0.355099791 192.168.59.83 192.168.59.76 DNS
9 0.355100096 192.168.59.83 192.168.59.76 DNS
10 0.355100097 192.168.59.83 192.168.59.76 DNS
11 0.355100125 192.168.59.83 192.168.59.76 DNS
95 Standard query 0xc8a0 A content-signature-2.cdn.mozilla.net
95 Standard query 0xc9ae AAAA content-signature-2.cdn.mozilla.net
85 Standard query 0xc733 A push.services.mozilla.com
85 Standard query 0x354c AAAA push.services.mozilla.com
97 Standard query 0x34a3 A firefox.settings.services.mozilla.com
97 Standard query 0x28a6 AAAA firefox.settings.services.mozilla.com
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-1086.awsdns-56.co.uk NS ns-377.awsdns-47.com NS ns-614.
175 Standard query response 0x34a3 A firefox.settings.services.mozilla.com CNAME prod.remote-settings.prod.webservices.mozgcp.net A 34.149.109.209
295 Standard query response 0x354c AAAA push.services.mozilla.com CNAME autopush.prod.mozaws.net SOA ns-1260.awsdns-29.org
249 Standard query response 0x28a6 AAAA firefox.settings.services.mozilla.com CNAME prod.remote-settings.prod.webservices.mozgcp.net SOA ns-cloud-a1.googledomains.com
249 Standard query response 0xc8a0 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 0.505899295 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.

```
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 0.505899295 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
30 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
31 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
32 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
33 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
34 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
35 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
36 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
37 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
38 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
39 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
40 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
41 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
42 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
43 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
44 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
45 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
46 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
47 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
48 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
49 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
50 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
51 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
52 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
53 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
54 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
55 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
56 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
57 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
58 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
59 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
60 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
61 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
62 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
63 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
64 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
65 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
66 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
67 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
68 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
69 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
70 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
71 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
72 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
73 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
74 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
75 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
76 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
77 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
78 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
79 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
80 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
81 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
82 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
83 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
84 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
85 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
86 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
87 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
88 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
89 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
90 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
91 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
92 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
93 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
94 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
95 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
96 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
97 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
98 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
99 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
100 0.517821395 34.149.100.209 192.168.59.76 TLSv1.3 1845 Application Data
```



## Launching YouTube:

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

```
178 22.716790872 192.168.59.76 192.168.59.83 DNS 75 Standard query 0xa79e A www.youtube.com
179 22.716806702 192.168.59.76 192.168.59.83 DNS 75 Standard query 0xa79e AAAA www.youtube.com
180 22.723365529 192.168.59.76 192.168.59.76 DNS 368 Standard query response 0xa79e A www.youtube.com CNAME youtube-us.l.google.com A 216.58.203.46 A 142.250.67.174 A 172.217.160.174 A 172.217.17
181 22.723365552 192.168.59.83 192.168.59.76 DNS 224 Standard query response 0xa79e AAAA www.youtube.com CNAME youtube-us.l.google.com AAAA 2404:6800:4009:820::200e AAAA 2404:6800:4009:81f::200e
```

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.

```
2319 192.9213155... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e TLSv1.3 125 Application Data
2320 192.9239567... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TLSv1.3 1294 Application Data
2321 192.9249981... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TLSv1.3 1294 Application Data
2322 192.9250060... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e TCP 86 46724 ... 443 [ACK] Seq=12341 Ack=38531 Win=68992 Len=0 TSval=3723590056 TSecr=1974055375
2323 192.9316208... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TLSv1.3 1294 Application Data
2324 192.9326428... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TLSv1.3 1294 Application Data
2325 192.9326497... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e TCP 86 46724 ... 443 [ACK] Seq=12341 Ack=40947 Win=73856 Len=0 TSval=3723590063 TSecr=1974055380
2326 192.9361220... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TLSv1.3 2502 Application Data, Application Data
2327 192.9361333... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e TCP 86 46724 ... 443 [ACK] Seq=12341 Ack=43363 Win=78720 Len=0 TSval=3723590067 TSecr=1974055386
2328 192.9401802... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TLSv1.3 1294 Application Data
2329 192.9418021... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TLSv1.3 1294 Application Data
2330 192.9418095... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e TCP 86 46724 ... 443 [ACK] Seq=12341 Ack=45779 Win=83456 Len=0 TSval=3723590073 TSecr=1974055392
2331 192.9460525... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TLSv1.3 1294 Application Data
2332 192.9476095... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TLSv1.3 1294 Application Data
2333 192.9476387... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e TCP 86 46724 ... 443 [ACK] Seq=12341 Ack=48195 Win=88320 Len=0 TSval=3723590078 TSecr=1974055397
2334 192.9543412... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TLSv1.3 2502 Application Data, Application Data
2335 192.9543532... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e TCP 86 46724 ... 443 [ACK] Seq=12341 Ack=50611 Win=93184 Len=0 TSval=3723590085 TSecr=1974055403
2336 192.9588235... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 TLSv1.3 2502 Application Data, Application Data, Application Data
```

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.

```
3293 461.7701234... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e QUIC 93 Protected Payload (KP0), DCID=c7a9911e27ea8a31
3294 461.7711499... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3295 461.7711500... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3296 461.7714474... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e QUIC 93 Protected Payload (KP0), DCID=c7a9911e27ea8a31
3297 461.7732307... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 93 Protected Payload (KP0), DCID=0292a5
3298 461.7734863... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3299 461.7735810... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e QUIC 93 Protected Payload (KP0), DCID=c7a9911e27ea8a31
3300 461.7744271... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3301 461.7746459... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3302 461.7747162... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e QUIC 93 Protected Payload (KP0), DCID=c7a9911e27ea8a31
3303 461.7756719... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3304 461.7756720... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3305 461.7757832... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e QUIC 93 Protected Payload (KP0), DCID=c7a9911e27ea8a31
3306 461.7771764... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3307 461.7773117... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3308 461.7773319... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e QUIC 93 Protected Payload (KP0), DCID=c7a9911e27ea8a31
3309 461.7791896... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3310 461.7791896... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3311 461.7792600... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e QUIC 93 Protected Payload (KP0), DCID=c7a9911e27ea8a31
3312 461.7948727... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3313 461.7956715... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3314 461.7956716... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3315 461.7958400... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e QUIC 93 Protected Payload (KP0), DCID=c7a9911e27ea8a31
3316 461.7979535... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3317 461.7979538... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3318 461.7982968... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e QUIC 93 Protected Payload (KP0), DCID=c7a9911e27ea8a31
3319 461.7996645... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3320 461.7996647... 2404:6800:4009:820::200e 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 QUIC 1399 Protected Payload (KP0), DCID=0292a5
3321 461.8000382... 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 2404:6800:4009:820::200e QUIC 93 Protected Payload (KP0), DCID=c7a9911e27ea8a31
```

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).

[illegible]

### 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 throughput 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:**

No.	Time	Source	Destination	Protocol	Length	Info
57	0.00384103	2600:140f:2800::1724:b18a	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	975	Response
58	0.00000256	2600:140f:2800::1724:b18a	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	975	Response
87	0.000484717	2600:140f:2800::1724:b18a	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	975	Response
233	0.01020994	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	787	Response
240	0.022591187	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	787	Response
392	0.00194898	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	787	Response
457	0.010250147	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	787	Response
607	0.000011537	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
623	0.010117421	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
960	0.000297066	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	787	Response
1075	0.005733783	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
1184	0.000000000	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
1270	0.013717131	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
1286	0.002177651	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
1289	0.002320869	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
1310	0.000891495	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
1364	0.002291173	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
1530	0.000136656	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
1532	0.000021537	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
1562	0.000000977	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
1571	0.000166698	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
1591	0.000000958	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
1601	0.004405602	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
2585	0.002624955	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
3470	0.000543178	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
3480	0.000707188	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
3484	0.000448827	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
3487	0.000000106	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	788	Response
5831	0.010288397	2404:6800:4009:82b::2003	2401:4900:3def:ed68:d17f:28e7:c98b:1a20	OCSP	787	Response

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

<pre> Frame 1184: 788 bytes on wire (6304 bits), 788 bytes captured (6304 bits) on interface wlp0s20f3, id 0 Ethernet II, Src: 46:71:16:a6:07:7b (46:71:16:a6:07:7b), Dst: IntelCor_b1:3b:63 (a8:64:f1:b1:3b:63) Internet Protocol Version 6, Src: 2404:6800:4009:82b::2003, Dst: 2401:4900:3def:ed68:d17f:28e7:c98b:1a20 Transmission Control Protocol, Src Port: 80, Dst Port: 47950, Seq: 2806, Ack: 2123, Len: 702 Hypertext Transfer Protocol   HTTP/1.1 200 OK\r\n     Content-Type: application/ocsp-response\r\n     Date: Thu, 07 Sep 2023 17:47:28 GMT\r\n     Cache-Control: public, max-age=14400\r\n     Server: ocsp_responder\r\n     Content-Length: 472\r\n     [Content length: 472]     X-XSS-Protection: 0\r\n     X-Frame-Options: SAMEORIGIN\r\n     \r\n [HTTP response 5/6] [Time since request: 0.369142142 seconds] [Prev request in frame: 721] [Prev response in frame: 960] [Request in frame: 964] [Next request in frame: 1358] [Next response in frame: 1530] [Request URI: http://ocsp.pki.goog/gts1c3] File Data: 472 bytes Online Certificate Status Protocol </pre>	<pre> 005 006 007 008 009 00a 00b 00c 00d 00e 00f 010 011 012 013 014 015 016 017 018 019 01a 01b 01c 01d 01e 01f 020 021 </pre>
---	--

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 there are indeed headers like `cache-control` which can prove that there is indeed a caching mechanism.

Analyzing 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 http 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.