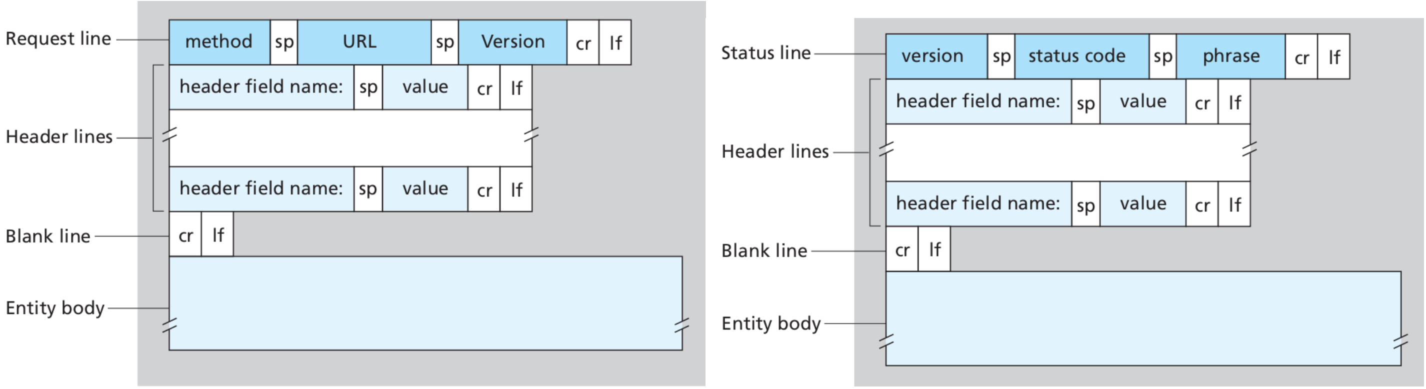
Application Assigned : Hotstar Video Streaming
Link for data : Data Collected

Question 1: Protocols Used

A) Application Layer

- **HTTP :** The protocol used in the application layer is **Hyper Text Transfer Protocol.** The HTTP messages are of two types, request and response. Request messages consist of a **request line**, followed by the **Header** and the **body** of the message. Response message format is similar but it has a **status line** instead of a request line. Request line holds the info regarding the **method** of the request, the server **URL** and the **version** of HTTP being used. Status line holds the **version**, the **status code** and the status **phrase**. Header consists of **field name** value pairs holding other meta information and finally the body consists of data that is sent or received.

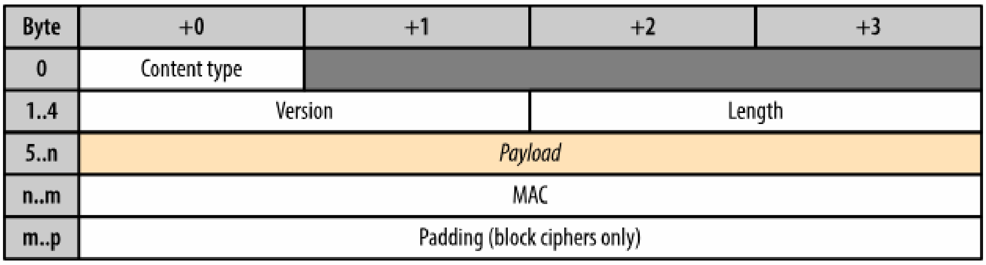


(a) HTTP Request Message

(b) HTTP Response Message

Figure 1: HTTP Message Format

- **TLSv1.2 :** TLSv1.2 is the successor of **SSL** and it provides communications security over a computer network. Symmetric cryptography is used to encrypt the data transmitted. The packet contains the type of message (handshake, alert, or data) in the '**Content Type**' field. It also contains the **version**, **length** of data and **MAC (Message Authentication Code)**.



Byte	+0	+1	+2	+3
0	Content type			
1..4	Version		Length	
5..n	*Payload*			
n..m	MAC			
m..p	Padding (block ciphers only)			

Figure 2: TLSv1.2 Message Format

B) Transport Layer

Transmission Control Protocol is a standard that defines how to establish and maintain a network conversation via which application programs can exchange data. **Source Port** and **Destionation Port** identify the hosts of the connection, source being the end point from where the segment is sent. **Sequence Number** specifies the number assigned to the first byte of data in the current message. If the ACK control bit is set, then **Acknowledgment number** refers to the next sequence number that the sender is expecting to receive. **Data offset** specifies the size of the variable sized TCP header. **Flags** are 1 bit values that specify the state of the connection and are used for control. **Window size** is the size of the buffer of the receiver. **Checksum** is used for error correction. Data field contains the payload of the segment.

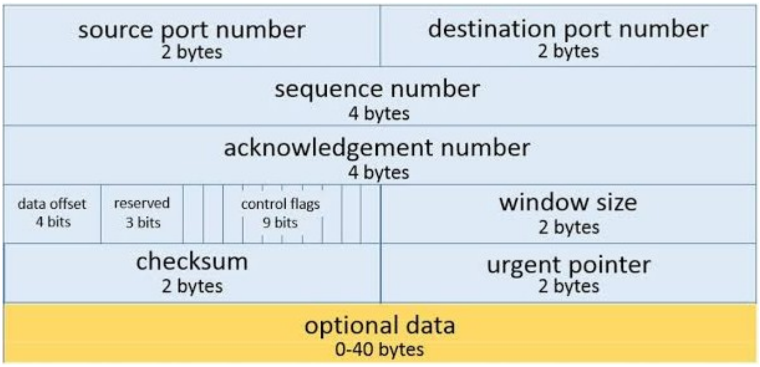


Figure 3: TCP Segment Format

C) Network Layer

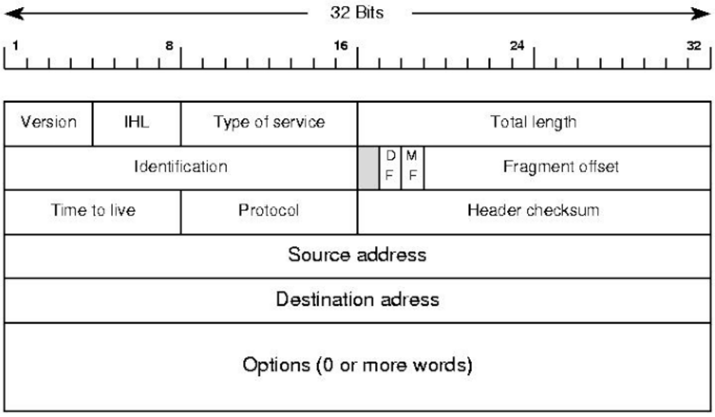


Figure 4: IP Datagram Header Format

IPv4 (Internet Protocol Version 4) is one of the core protocols of standards-based internetworking methods in the Internet. It is used in packet-switched networks. Each IP datagram consists of a **header** and a **data** part. The header has a 20 byte fixed part followed by a variable-sized optional part. **Version** refers to the version of the datagram. In this case it would be 4. **IHL (Internet Header Length)** is the size of the header. **Types of Service** contains a 3-bit precedence field (that is ignored today), 4 service bits, and 1 unused bit. Service bits specify what characteristics the physical layer shhould use. **Total Length** is the total length of the datagram in bytes. **Identification** uniquely identifies the datagram. All fragments of a datagram contain the same identification value. **TTL (Time to Live)** is the maximum routers through which the segment can be switched. **Protocol** indicates the next higher level protocol that is contained within the data portion of the packet. **Header checksum** is used for error detection. **Source** and **destination addresses** are the addresses of the source and destination of the packet respectively.

D) Link Layer

Ethernet II is used in the link layer. **Preamble** is a 7 byte pattern of alternating 0's and 1's which indicates starting of the frame and allow sender and receiver to establish bit synchronization. **SFD** is the start frame delimiter and marks the start of frame. **Destination** and **source addresses** are the MAC addresses of the sending and receiving machines of the frame respectively. **Type** field is used to specify the protocol that is being used. **FCS (Frame Check Sequence)** is the error detecting code that is added.

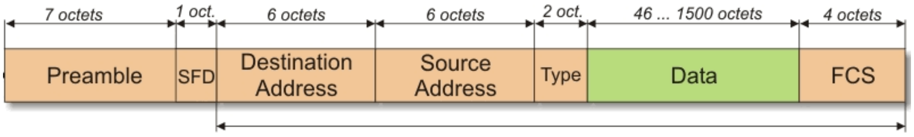
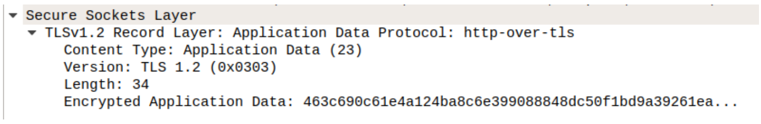


Figure 5: Ethernet Frame Format

Question 2: Observed Values in Different Protocols

A) Application Layer



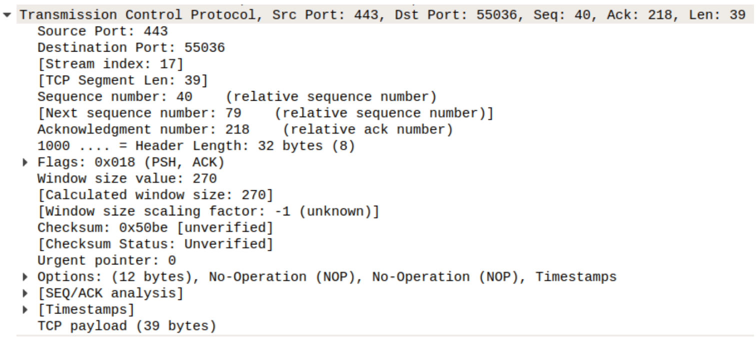
```
▾ Secure Sockets Layer
  ▾ TLSv1.2 Record Layer: Application Data Protocol: http-over-tls
      Content Type: Application Data (23)
      Version: TLS 1.2 (0x0303)
      Length: 34
      Encrypted Application Data: 463c690c61e4a124ba8c6e399088848dc50f1bd9a39261ea...
```

Figure 6: TLSv1.2 Record Example

It is visible from the example that the appication data protocol is **Http-over-tls** (aka HTTPS). The **content type** in this message is Application Data. **Version** of TLS is 1.2. **Length** of the data is 34 bytes. **Encrypted Application Data** can also be seen.
HTTPS encrypts all message contents, including the HTTP headers and the request/response data, therefore no HTTP header or request/response can be seen explicitly.

B) Transport Layer

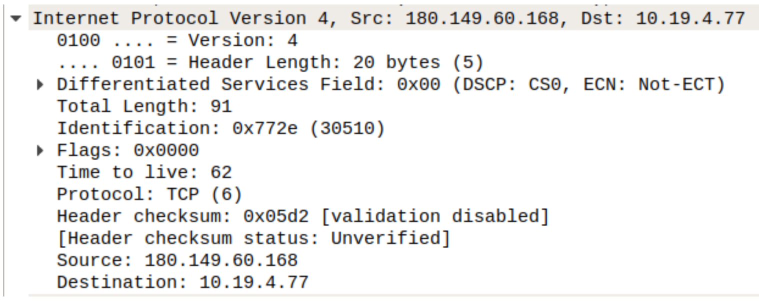
It can be seen that the **source port** is 443 (This is to be expected because the default port for HTTPS connection is 443). The **destination port** is 55036. The **TCP Segment Length** is 39 bytes (payload). The **sequence number** is 40. **Acknowledgement number** is 218 which means that the sender of this segment is expecting a segment with sequence number 218 from the reciever. **Flags** field tells us that PSH and ACK flag is enabled. PSH flag is an option provided by TCP that allows the sending application to start sending the data even when the buffer is not full. **Window size** value is 270 (Number of packets sent before acknowledgment). The **checksum** value can also be seen that is used for error detection.



```
▾ Transmission Control Protocol, Src Port: 443, Dst Port: 55036, Seq: 40, Ack: 218, Len: 39
    Source Port: 443
    Destination Port: 55036
    [Stream index: 17]
    [TCP Segment Len: 39]
    Sequence number: 40    (relative sequence number)
    [Next sequence number: 79    (relative sequence number)]
    Acknowledgment number: 218    (relative ack number)
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x018 (PSH, ACK)
    Window size value: 270
    [Calculated window size: 270]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0x50be [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▸ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  ▸ [SEQ/ACK analysis]
  ▸ [Timestamps]
    TCP payload (39 bytes)
```

Figure 7: TCP Segment Example

C) Network Layer



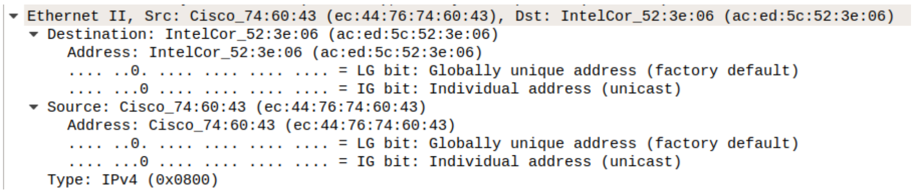
```
▾ Internet Protocol Version 4, Src: 180.149.60.168, Dst: 10.19.4.77
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 91
    Identification: 0x772e (30510)
  ▸ Flags: 0x0000
    Time to live: 62
    Protocol: TCP (6)
    Header checksum: 0x05d2 [validation disabled]
    [Header checksum status: Unverified]
    Source: 180.149.60.168
    Destination: 10.19.4.77
```

Figure 8: IP Datagram Example

Version as stated earlier is 4 because IPv4 is being used. When IPv6 will be used, then the version will become 6. **Header Length** has the value 5 which implies that the header size is 20 bytes. **Total length** of the packet is 91 bytes and the **Identification number** is 0x772e. **Flag** value of 0000 implies that the datagram is not fragmented. **TTL** is 62 meaning that it can hop 62 times before dying. **Checksum** Value (0x05d2) can also be seen which is used for error detection. **Source address** (IP address of server) is 180.149.60.168. **Destination address** (IP address of my laptop) is 10.19.4.77

D) Link Layer

The information about the **Destination** and **Source MAC addresses** can be seen. They are unique addresses assigned to the **Network Interface controllers** of the machines. The source of this frame is a Cisco device and the destination is my laptop. The **Type** of connection can also be seen.



```
▾ Ethernet II, Src: Cisco_74:60:43 (ec:44:76:74:60:43), Dst: IntelCor_52:3e:06 (ac:ed:5c:52:3e:06)
  ▾ Destination: IntelCor_52:3e:06 (ac:ed:5c:52:3e:06)
      Address: IntelCor_52:3e:06 (ac:ed:5c:52:3e:06)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
  ▾ Source: Cisco_74:60:43 (ec:44:76:74:60:43)
      Address: Cisco_74:60:43 (ec:44:76:74:60:43)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: IPv4 (0x0800)
```

Figure 9: Ethernet Frame Example

Question 3: Observed Values in Different Protocols