CS349- Assignment 2

Tushar Bhutada- 170101073

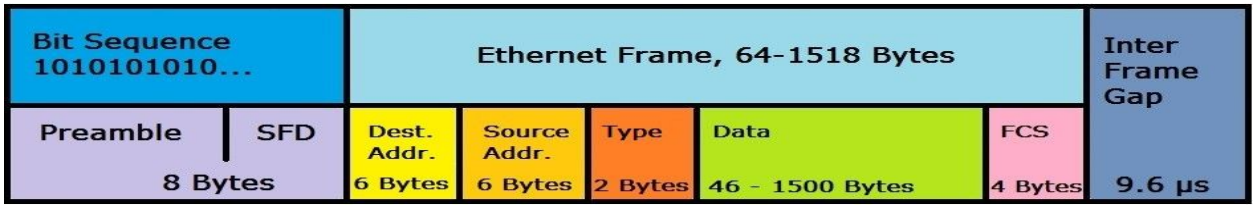
Captures traces-

https://drive.google.com/drive/folders/1uxElzqrKW0Rh_dRNjtDnfR0eJl1Rmutq?usp=sharing

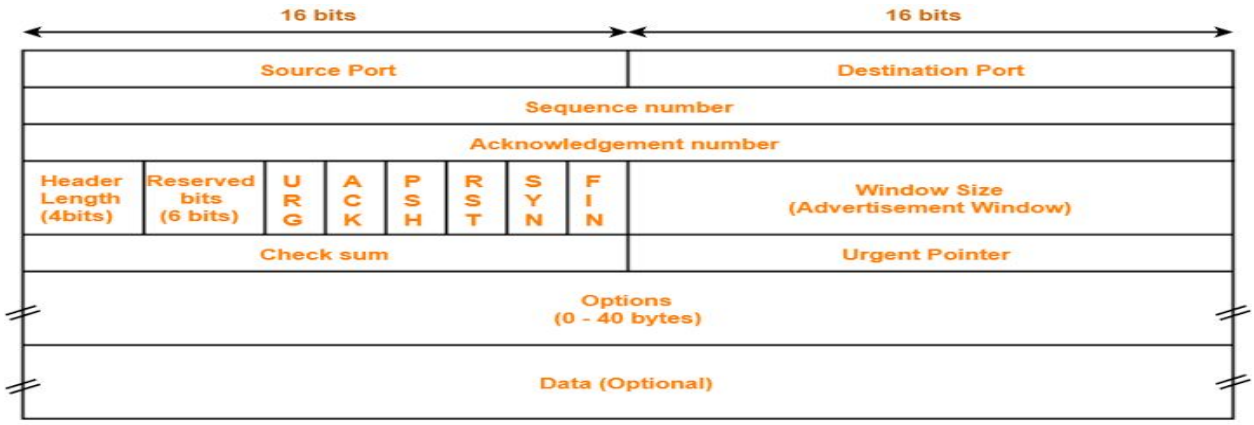
Answer 1

The following protocols have been used at various layers by the application NPTEL as observed using Wireshark. Also, all fields have not been explained in this part because the rest of them have been explained in Answer 2.

Link Layer- **Ethernet II** is the used protocol. **Preamble** allows synchronization of network clocks across devices. **Start Frame Delimiter**(SFD) marks the beginning of a new frame. **Type** denotes the type of connection. The **frame check sequence** (FCS) is a cyclic redundancy check (CRC) that allows the detection of corrupted data within the entire frame as received on the receiver side.

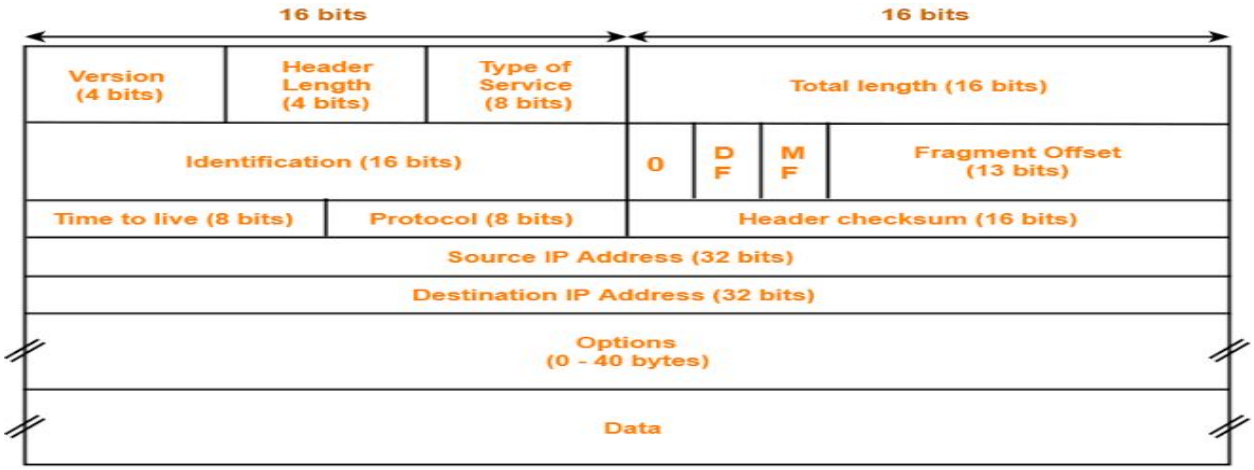


Transport layer- TCP is the used protocol. **Sequence Number field** (32 bits) specifies the number assigned to the first byte of data in the current message. **Acknowledgment Number field** (32 bits) contains the value of the next sequence number that the sender of the segment is expecting to receive if the ACK control bit is set. The Flags is a set of six values after the reserved bits. **Window field** (16 bits) specifies the size of the sender's receive window (that is, buffer space available for incoming data). **Checksum field** (16 bits) indicates whether the header was damaged in transit. **Urgent pointer field** (16 bits) points to the first urgent data byte in the packet. **Options field** (variable length) specifies various TCP options. The **data field** (variable length) contains upper-layer information.



TCP Header

Network layer- IPv4 is the used protocol. **Total length** field defines the entire packet size in bytes, including header and data. The **identification** field is primarily used for uniquely identifying the group of fragments of a single IP datagram. **Fragment** specifies the offset of a particular fragment relative to the beginning of the original unfragmented IP datagram.



IPv4 Header

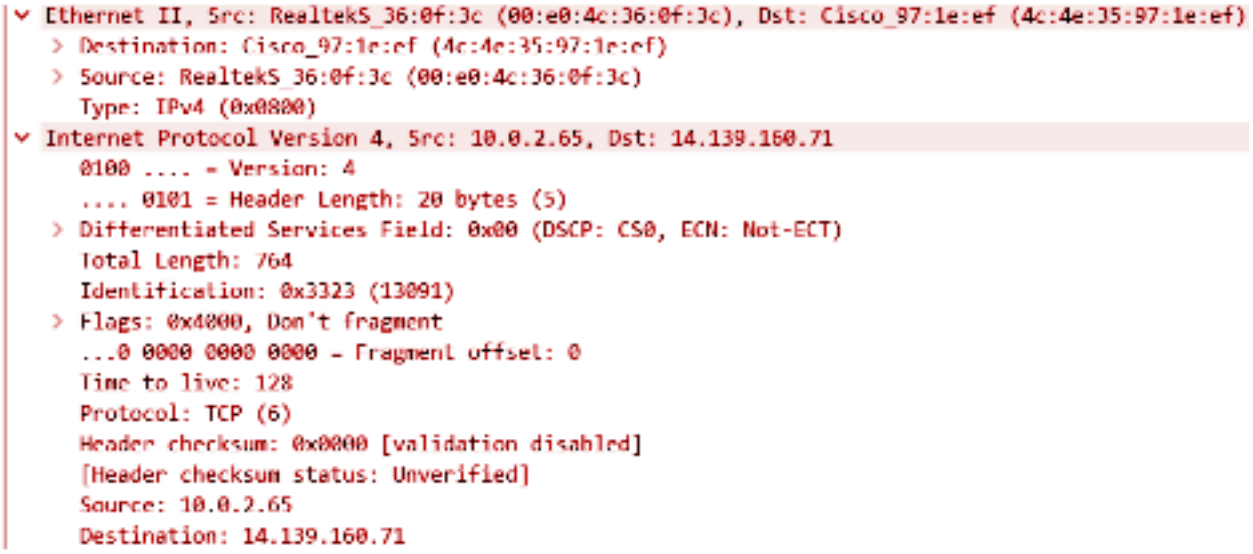
Application layer- Transport Layer Security(TLS) is the used protocol for security. The basic unit of data in SSL is a **record**. Each record consists of a five-byte record **header**, followed by **data**. The header contains Record Type which can be of four types(Handshake, Change Cipher Spec, Alert, Application Data). It is a 16-byte value formatted in network order.

Answer 2

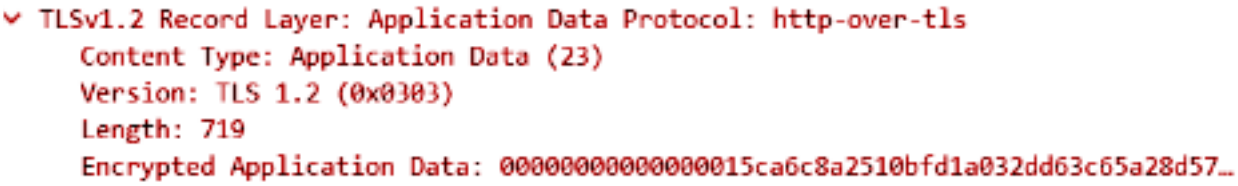
Ethernet II- **Destination** contains MAC Address of the destination (Hostel switch). **Source** contains the MAC address of the sending device (My laptop). **Type** refers to the protocol used for the payload in the frame.

IPv4- IPv4 address is in use (10.11.12.9). **Header length** is 20 bytes and shown as 5 because the jump is 4 bytes each. **DCP: CS0** indicates best-effort delivery service and **ECN: Not-ECT** implies Non ECN-Capable Transport. **Packet size** is 764 bytes. **Flag** value equals “Don’t Fragment” to not break the packet into smaller fragments. **TTL(**Time To Live**)** is 128 hops.TCP

is used as **protocol** for the layer above. **Header Checksum** is the technique used for error detection of packet headers. **Source and Destination** contain the IP address of my laptop and the application respectively.



SSL-The **Application Data Protocol** indicates that the protocol used at the application layer is HTTP (secured with tls. **Content-Type** refers to the payload type(Application data here). **Version** tells the protocol version for communication security(TLSv1.2 here). **Length** is the message length.**Encrypted Application Data** is unreadable by anyone who does not have the required key.



TCP- The **Source, Destination, Header length** are the same as explained before.**Flags** PSH and ACK are set. ACK refers to the acknowledgment from another device upon receiving. PSH is an indication by the sender that, if the receiving machine's TCP implementation has not yet provided the data it's received to the code that's reading the data, it should do so at that point. The **Window Size Value** indicates buffer space at one end for receiving. **Scaling Factor** refers to the multiplier for window size displayed.

```
Transmission Control Protocol, Src Port: 62020, Dst Port: 443, Seq: 611, Ack: 5558, Len: 724
    Source Port: 62020
    Destination Port: 443
    [Stream index: 64]
    [TCP Segment Len: 724]
    Sequence number: 611    (relative sequence number)
    Sequence number (raw): 3737707613
    [Next sequence number: 1335    (relative sequence number)]
    Acknowledgment number: 5558    (relative ack number)
    Acknowledgment number (raw): 3137455813
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x018 (PSH, ACK)
    Window size value: 256
    [Calculated window size: 65536]
    [Window size scaling factor: 256]
    Checksum: 0xbc01 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  > [SEQ/ACK analysis]
      [iRTT: 0.000767000 seconds]
      [Bytes in flight: 724]
      [Bytes sent since last PSH flag: 724]
  > [Timestamps]
      [Time since first frame in this TCP stream: 2.981207000 seconds]
      [Time since previous frame in this TCP stream: 0.000225000 seconds]
    TCP payload (724 bytes)
```

Answer 3

- Coursera/NPTEL has one of the most important features of their application as streaming videos. Since these are educational applications, every piece of data has to be received perfectly and as a result, TCP is used for reliable data transfer, making its use inevitable. Also, data arrives in order which is a necessity for a video stream.
- TLSv1.2 serves as the Secure Sockets Layer which provides the most required security to the application. This allows the application to send packets between two end devices with protection from hackers/malicious bugs by encrypting the data.
- Ethernet II is used for its proper error handling and flow control mechanisms to ensure a smooth run of the application.
- IPv4 is used to assist TCP as TCP is not compatible with other network layer protocols(except IP).

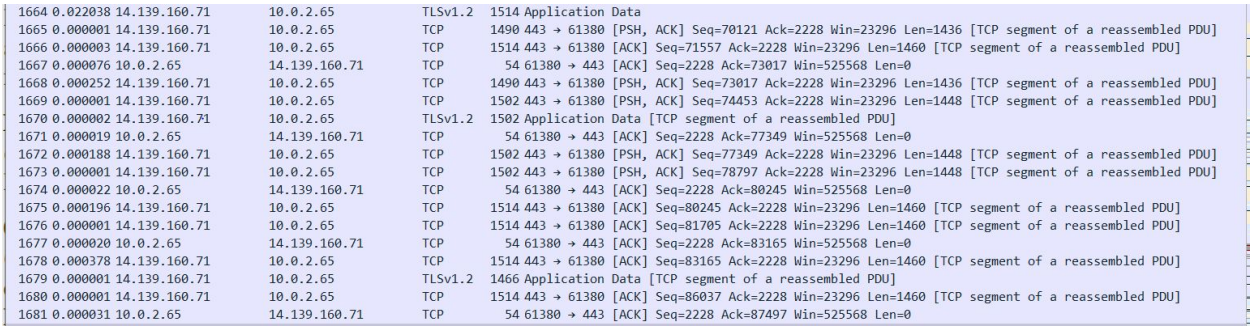
Answer 4

DNS querying: On starting the application, DNS querying is done by the browser. Series of messages are exchanged to allow the browser to learn the IP address of nptel.ac.in. the querying is for ‘A’ record type, which is used to relate IP addresses with domain names.

```
1819 29.409774 10.0.2.65    172.17.1.1    DNS    71 Standard query 0xc247 A nptel.ac.in
1820 29.409775 10.0.2.65    172.17.1.1    DNS    80 Standard query 0x6b99 A fonts.googleapis.com
1822 29.410632 172.17.1.1   10.0.2.65     DNS   162 Standard query response 0xc247 A nptel.ac.in A 14.139.160.71 NS dns2.iitm.ac.in NS dns1.iitm.a
```

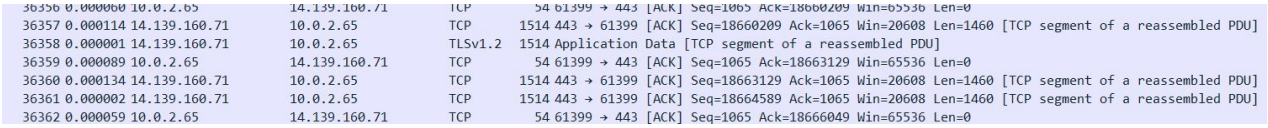
Streaming videos- Playing a video on the application causes the server to send data to my laptop, each of which is ACKnowledged by a message. My laptop sent a cumulative ACK back for multiple TCP segments of a reassembled PDU (upper-layer protocol data unit is broken down into smaller TCP segments and then sent through the TCP connection) indicating the

first-byte number that it wants to be sent next. Once all these segments arrive, they are re-assembled and the application layer receives data in the same order. Functionalities, like playing, pausing, etc, are enabled through different port on my laptop but same port 443 on the NPTEL server. The communication is achieved by sending acknowledgments through piggybacking mechanism which is evident from the fact that ACK packet has non zero length implying some data. KEEP-ALIVE messages are periodically exchanged to keep the connection alive while a video is paused.



1664	0.022038	14.139.160.71	10.0.2.65	TLSv1.2	1514	Application Data
1665	0.000001	14.139.160.71	10.0.2.65	TCP	1490	443 → 61380 [PSH, ACK] Seq=70121 Ack=2228 Win=23296 Len=1436 [TCP segment of a reassembled PDU]
1666	0.000003	14.139.160.71	10.0.2.65	TCP	1514	443 → 61380 [ACK] Seq=71557 Ack=2228 Win=23296 Len=1460 [TCP segment of a reassembled PDU]
1667	0.000076	10.0.2.65	14.139.160.71	TCP	54	61380 → 443 [ACK] Seq=2228 Ack=73017 Win=525568 Len=0
1668	0.000252	14.139.160.71	10.0.2.65	TCP	1490	443 → 61380 [PSH, ACK] Seq=73017 Ack=2228 Win=23296 Len=1436 [TCP segment of a reassembled PDU]
1669	0.000001	14.139.160.71	10.0.2.65	TCP	1502	443 → 61380 [PSH, ACK] Seq=74453 Ack=2228 Win=23296 Len=1448 [TCP segment of a reassembled PDU]
1670	0.000002	14.139.160.71	10.0.2.65	TLSv1.2	1502	Application Data [TCP segment of a reassembled PDU]
1671	0.000019	10.0.2.65	14.139.160.71	TCP	54	61380 → 443 [ACK] Seq=2228 Ack=77349 Win=525568 Len=0
1672	0.000188	14.139.160.71	10.0.2.65	TCP	1502	443 → 61380 [PSH, ACK] Seq=77349 Ack=2228 Win=23296 Len=1448 [TCP segment of a reassembled PDU]
1673	0.000001	14.139.160.71	10.0.2.65	TCP	1502	443 → 61380 [PSH, ACK] Seq=78797 Ack=2228 Win=23296 Len=1448 [TCP segment of a reassembled PDU]
1674	0.000022	10.0.2.65	14.139.160.71	TCP	54	61380 → 443 [ACK] Seq=2228 Ack=80245 Win=525568 Len=0
1675	0.000196	14.139.160.71	10.0.2.65	TCP	1514	443 → 61380 [ACK] Seq=80245 Ack=2228 Win=23296 Len=1460 [TCP segment of a reassembled PDU]
1676	0.000001	14.139.160.71	10.0.2.65	TCP	1514	443 → 61380 [ACK] Seq=81705 Ack=2228 Win=23296 Len=1460 [TCP segment of a reassembled PDU]
1677	0.000020	10.0.2.65	14.139.160.71	TCP	54	61380 → 443 [ACK] Seq=2228 Ack=83165 Win=525568 Len=0
1678	0.000378	14.139.160.71	10.0.2.65	TCP	1514	443 → 61380 [ACK] Seq=83165 Ack=2228 Win=23296 Len=1460 [TCP segment of a reassembled PDU]
1679	0.000001	14.139.160.71	10.0.2.65	TLSv1.2	1466	Application Data [TCP segment of a reassembled PDU]
1680	0.000001	14.139.160.71	10.0.2.65	TCP	1514	443 → 61380 [ACK] Seq=86037 Ack=2228 Win=23296 Len=1460 [TCP segment of a reassembled PDU]
1681	0.000031	10.0.2.65	14.139.160.71	TCP	54	61380 → 443 [ACK] Seq=2228 Ack=87497 Win=525568 Len=0

Downloading video- Fundamentally it is the same as streaming, the only difference being that the laptop ow sends messages to the server through the same port, no different ports this time. The server also sends PSH packets when the video is about to finish being downloaded so that the buffer is emptied by giving it to the relevant application.



36356	0.000060	10.0.2.65	14.139.160.71	TCP	54	61399 → 443 [ACK] Seq=1065 Ack=18660209 Win=65536 Len=0
36357	0.000114	14.139.160.71	10.0.2.65	TCP	1514	443 → 61399 [ACK] Seq=18660209 Ack=1065 Win=20608 Len=1460 [TCP segment of a reassembled PDU]
36358	0.000001	14.139.160.71	10.0.2.65	TLSv1.2	1514	Application Data [TCP segment of a reassembled PDU]
36359	0.000089	10.0.2.65	14.139.160.71	TCP	54	61399 → 443 [ACK] Seq=1065 Ack=18663129 Win=65536 Len=0
36360	0.000134	14.139.160.71	10.0.2.65	TCP	1514	443 → 61399 [ACK] Seq=18663129 Ack=1065 Win=20608 Len=1460 [TCP segment of a reassembled PDU]
36361	0.000002	14.139.160.71	10.0.2.65	TCP	1514	443 → 61399 [ACK] Seq=18664589 Ack=1065 Win=20608 Len=1460 [TCP segment of a reassembled PDU]
36362	0.000059	10.0.2.65	14.139.160.71	TCP	54	61399 → 443 [ACK] Seq=1065 Ack=18666049 Win=65536 Len=0

TCP handshakes can be seen from the packets sequencing. The IP 14.139.160.71 corresponds to the NPTEL home page.

- The first handshake is the **three-way connection handshake** (SYN SYN-ACK ACK) that is used by TCP to set up a connection. It is initiated by a client hello message which is responded with a server hello. The connection then acts as a pipeline where the two ends can directly transfer data. To establish a connection, each device must send a SYN and receive an ACK for it from the other device. Thus, conceptually, we need to have four control messages pass between the devices. However, it's inefficient to send a SYN and an ACK in separate messages when one could communicate both simultaneously.

01	0.000700	10.0.2.65	14.139.160.71	TCP	66	60954 → 443 [SYN] Seq=0 Win=64240
02	0.000302	14.139.160.71	10.0.2.65	TCP	66	443 → 60953 [SYN, ACK] Seq=0 Ack=
03	0.000077	10.0.2.65	14.139.160.71	TCP	54	60953 → 443 [ACK] Seq=1 Ack=1 Win

- The second handshake is the **termination sequence** (FIN ACK FIN-ACK ACK). This sequence is observed when the application is closed. Termination is done on each side by sending a message with FIN bit set which serves as the termination request. The receiving device replies with an acknowledgment. The connection is terminated if this pair of FIN-ACK messages are received from both sides, hence the sequence isn't three-way handshake.

Answer 5

a) **For streaming**

Time	Throughput	RTT	Packet size	Lost packets	UDP packets	TCP packets	Received Responses
12 AM	16 KB/s	0.0259 sec	429 B	0	489	1512	1.175
11 AM	74 KB/s	0.0157 sec	684 B	0	888	7452	1.346
6 PM	111 KB/s	0.0098 sec	891 B	0	1131	11313	1.227

a) **For downloading**

Time	Throughput	RTT	Packet size	Lost packets	UDP packets	TCP packets	Received Responses
12 AM	38 KB/s	0.0471 sec	356 B	0	832	5386	1.392
11 AM	93 KB/s	0.0113 sec	532 B	0	599	4005	1.243
6 PM	155 KB/s	0.0243 sec	671 B	0	1036	7569	1.107

Answer 6

I had NPTEL allotted to me as the application. In all my experiments I found only one IP, which is 14.139.160.71. The reason being NPTEL mostly has its video lectures on youtube as a result of which the traffic to the application nptel.ac.in is not high and hence having one server suffices the daily traffic the site encounters. But on the other hand, cousera.org had multiple IP's because of the fact that Coursera has its audience over the entire globe and hence to maintain speed and efficiency to handle heavy traffic for educational learning it has various IP's. Multiple servers help in reducing network congestion and increase reliability, as there is no single point of error.