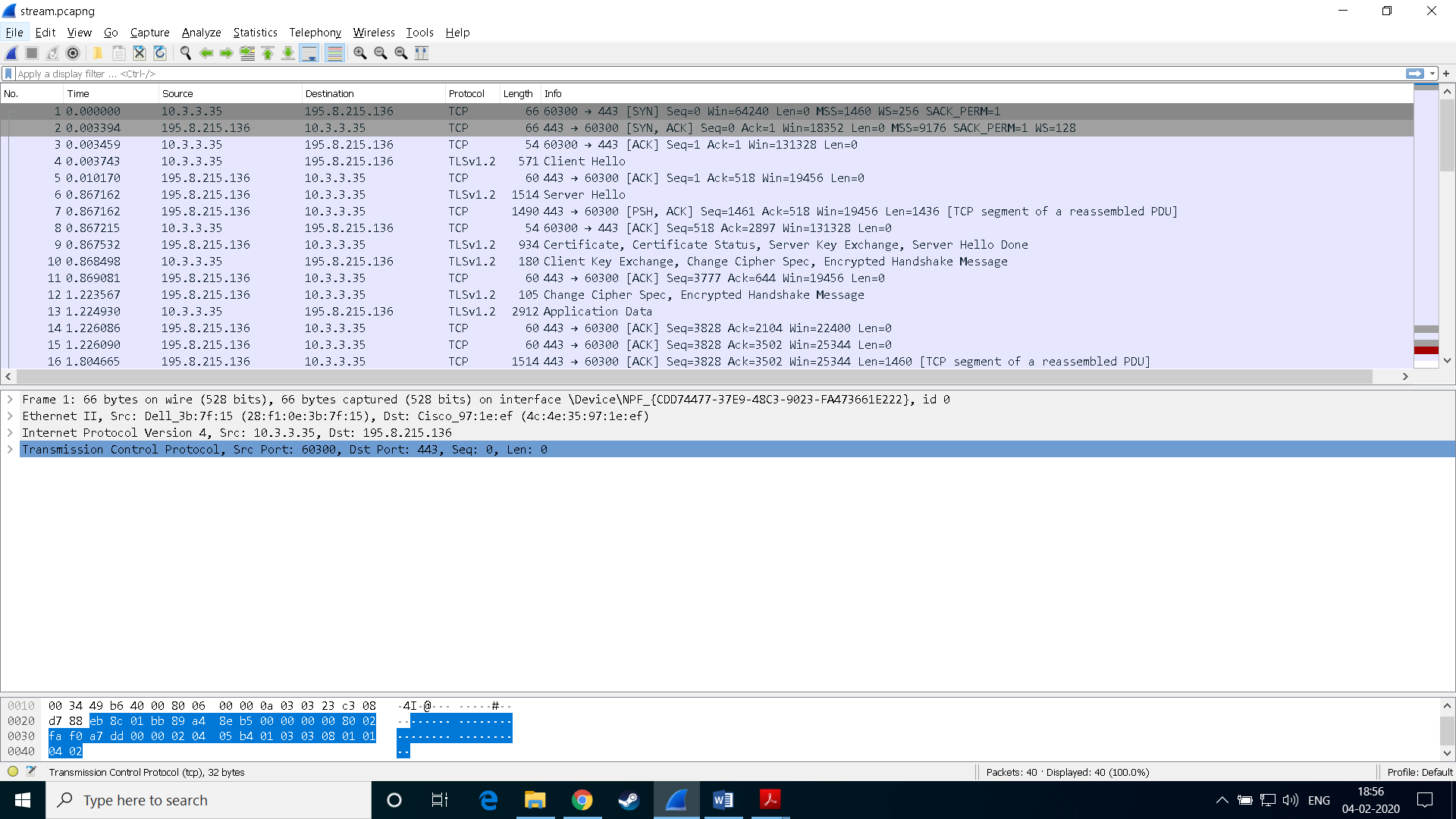
**ASSIGNMENT-2  
Submitted by: Aranya Aryaman  
Roll Number: 170101011**

**Question 1:**

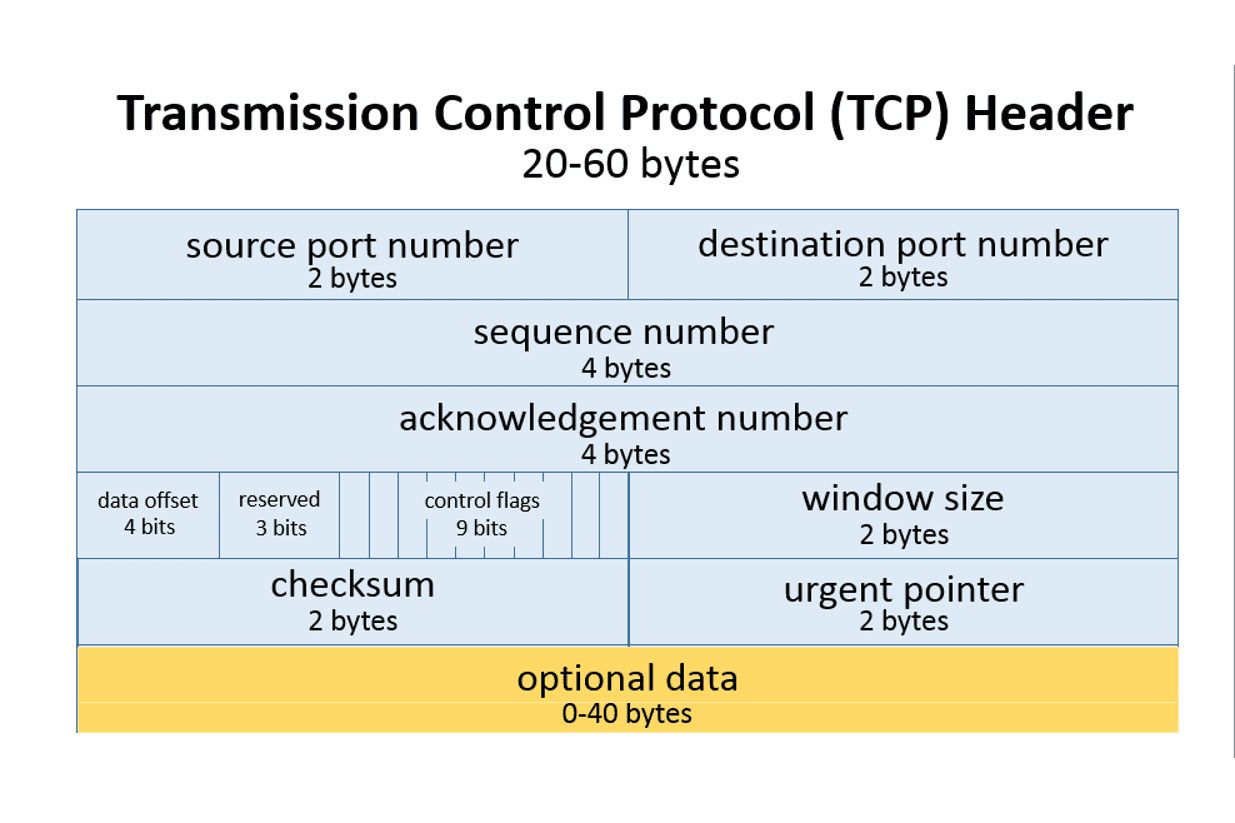
****

The different protocols used at different layers by the application [*www.dailymotion.com*](http://www.dailymotion.com)are as follows:

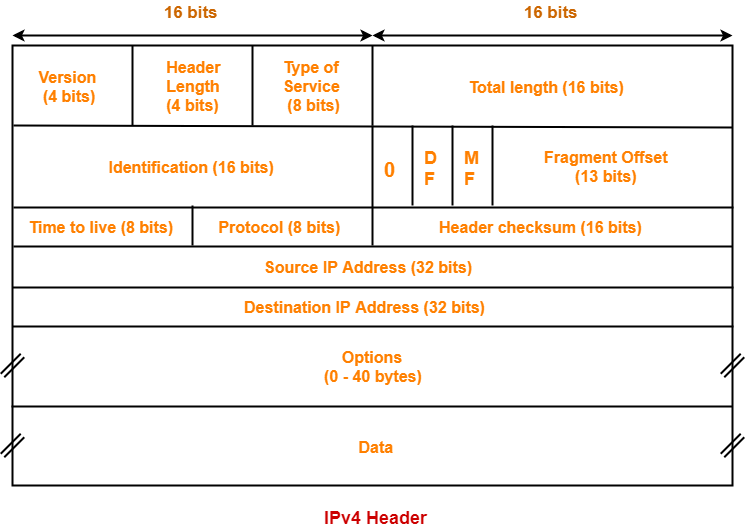
1. Transport Layer: **TCP**
2. Network Layer: **IPv4**
3. Application Layer: **HTTP**
4. Physical Layer: **Ethernet II**
5. Secure Socket Layer: **TLSv1.2 Protocol**

Note: Some of these protocols are found to be used in the application from traces but are not shown in the above figure by wireshark. Hence, they have been mentioned in the answer.  
**Packet Formats for various protocols are as follows:**

**Transport Layer: TCP** (*Details about TCP Frame Format has been mentioned in Question 2*)



**Network Layer: IPv4** (*Details about IPv4 Frame Format has been mentioned in Question 2*)



**Application Layer: HTTP**

HTTP Messages consist of requests being sent from a client to the server and responses from the server to the client.

HTTP-message = **<*Request*>|<*Response*>; HTTP/1.1 messages**

Request and Response messages use the generic message format of **RFC 822** for transferring entities (the payload of the message). This generic message format consists of the following four items:

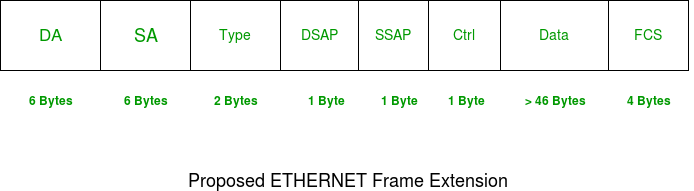
1. **Start-line**: Request Line or Status Line.
2. **Zero or more header fields followed by CRLF**: The headers are as follows: General Header, Response Header, Request Header and Entity Header.
3. **An empty line**: indicating the end of the header fields
4. **message-body**: It is used to carry the entity-body associated with the request or response.

**SSL (Secure Socket Layer): TLSv1.2**

This acts as an intermediate b/w transport layer and application layer. It deals with session and connection coordination. **TLSv1.2** was defined in RFC 5246. The packet format is as follows:

1. **Content type**: The type field is identical to TLSCompressed.type
2. **Version**: The version field is identical to TLSCompressed.version
3. **Length**: The length (in bytes) of the following TLSCiphertext.fragment. The length MUST NOT exceed 2^14 + 2048.
4. **Fragment**: The encrypted form of TLSCompressed.fragment with the MAC.
5. **Message Authentication Code (MAC)**: It is a one-way hash computed from a message and some secret data. It is difficult to forge without knowing the secret data. Its purpose is to detect if the message has been altered.

**Physical Layer: Ethernet II**

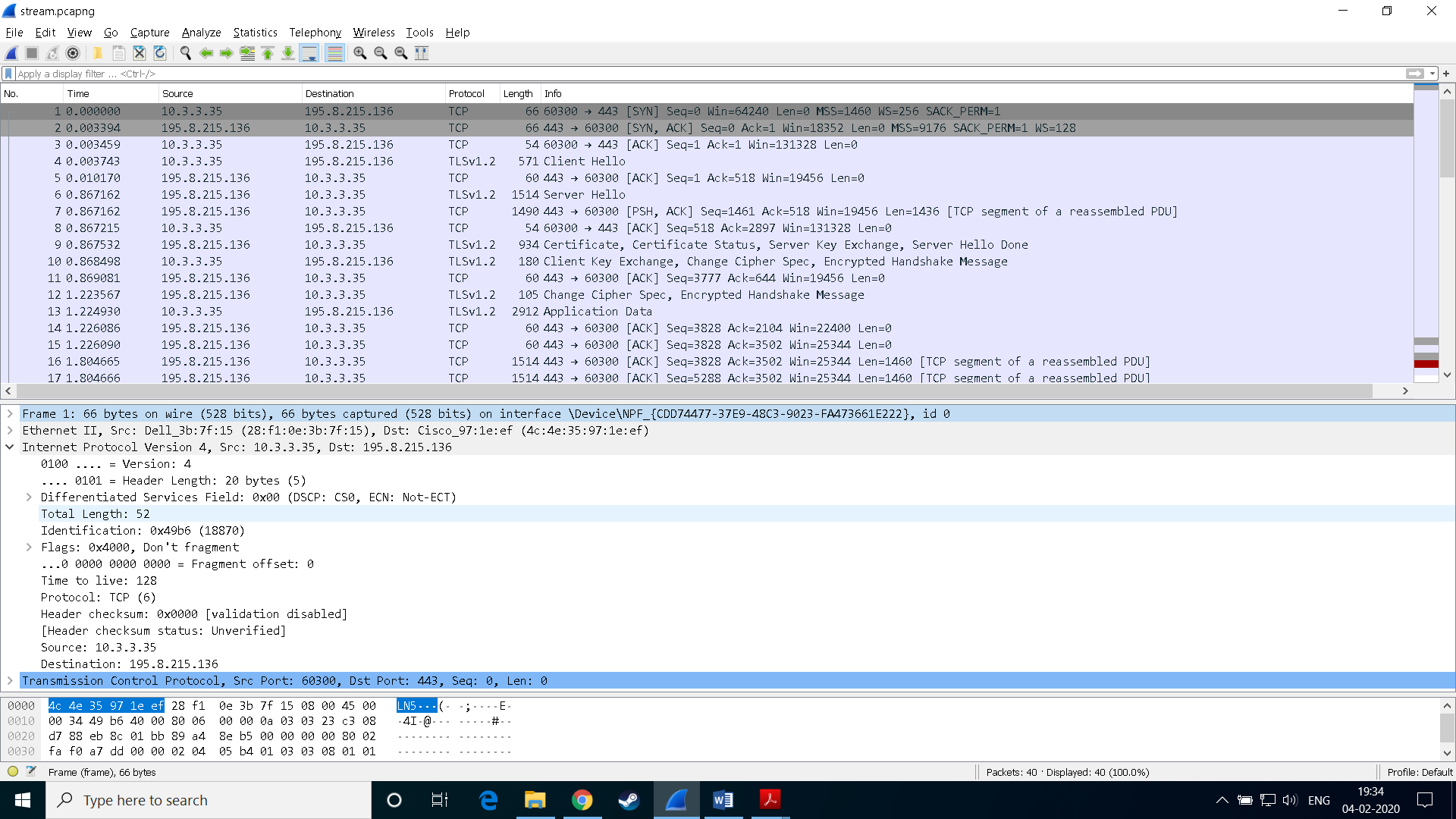


The fields in the frame are as follows

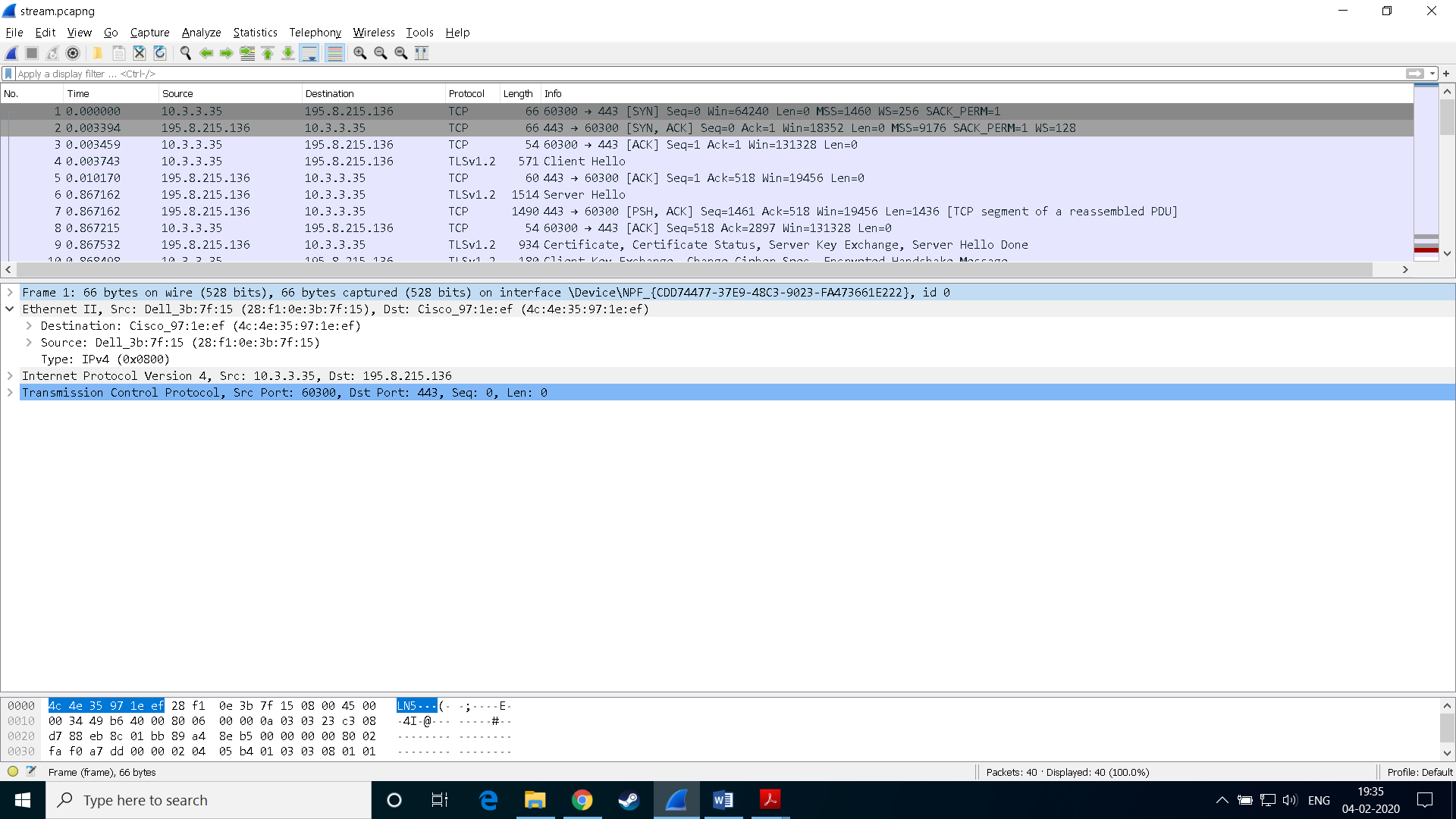
1. **DA** – Destination Address
2. **SA** – Source Address
3. **Type** – 0x8870 Ether Type
4. **DSAP** – 802.2 Destination Service Access Point
5. **SSAP** – 802.2 Source Service Access Point
6. **Ctrl** – 802.2 Control Field
7. **Data** – Protocol Data
8. **FCS** – Frame Checksum

**Question 2:**

The following figures describe the various fields of the different protocols mentioned above.

****

1. **Frame 1**: The frame number of the trace I took for the observation is 1. The frame protocol is not a real protocol itself, but used by Wireshark as a base for all protocols on top of it. 66 bytes were used by the frame.
2. **Ethernet II**:



**Src**: *Dell\_3b:7f:15 (28:f1:0e:3b:7f:15)*: MAC Address of my PC.  
 **Dst**: *Cisco\_97:1e:ef (4c:4e:35:97:1e:ef)*: MAC Address of Destination.

1. **Internet Protocol**:

**Version** **4:** Field indicates the format of the internet header.

**Source-10.3.3.35:** My PC’s IP Address

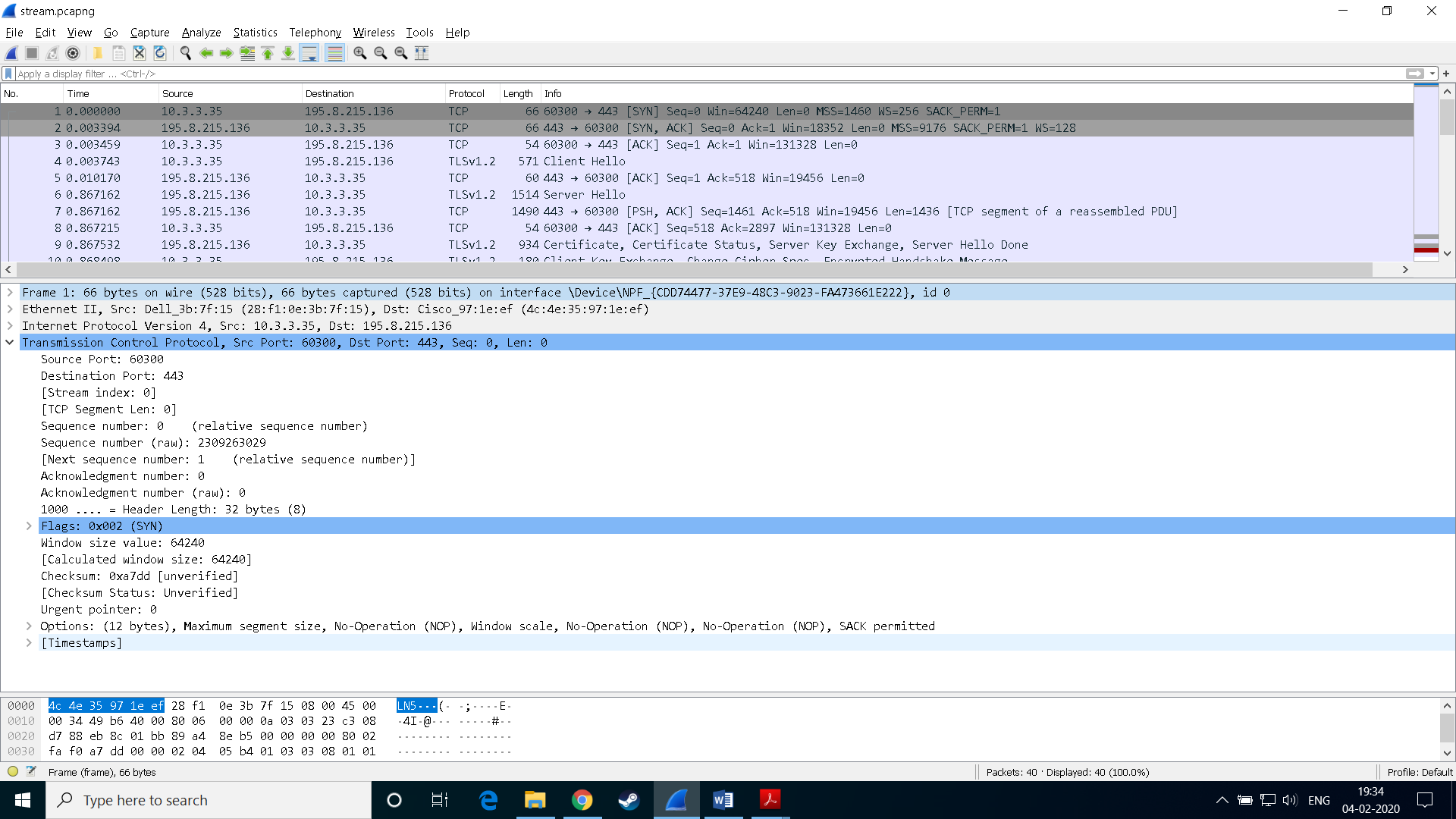
**Destination-195.8.215.136:** Destination’s IP Address

**Header Length-20 bytes:** Number of 32bit words in the TCP Header

**Differentiated Services Field 0x00:** Indicates particular quality of service needs from the network, the DSF defines the way the routers should queue packets while they are waiting to be forwarded.

**Total Length-52:** Length of the datagram, measured in octets, including internet header and data.

1. **Transmission Control Protocol:**



**Source Port- 60300:** My computer port in this case, which is sending packets.

**Destination Port-443:** Destination Port, which is receiving packets.

**TCP Segment Length-0:** This is TCP packet segment length.

**Sequence Number 1:** If the SYN flag is set to 1, this is the initial sequence number.

The sequence number of the actual first data byte and the acknowledged number in the corresponding ACK are then this sequence number plus 1.

**Acknowledgement Number 0:** The acknowledgement flag is set to 0.

**Window size value-** **64240**: This is the space for incoming data.

**Checksum- 0xa7dd:** The 16-bit Checksum field is used for error checking of the header and data.

**Urgent Pointer**-**0**: If the URG Flag is set, then this 16-bit field is an offset from the sequence number indicating the last urgent data byte.

**Question 3:**

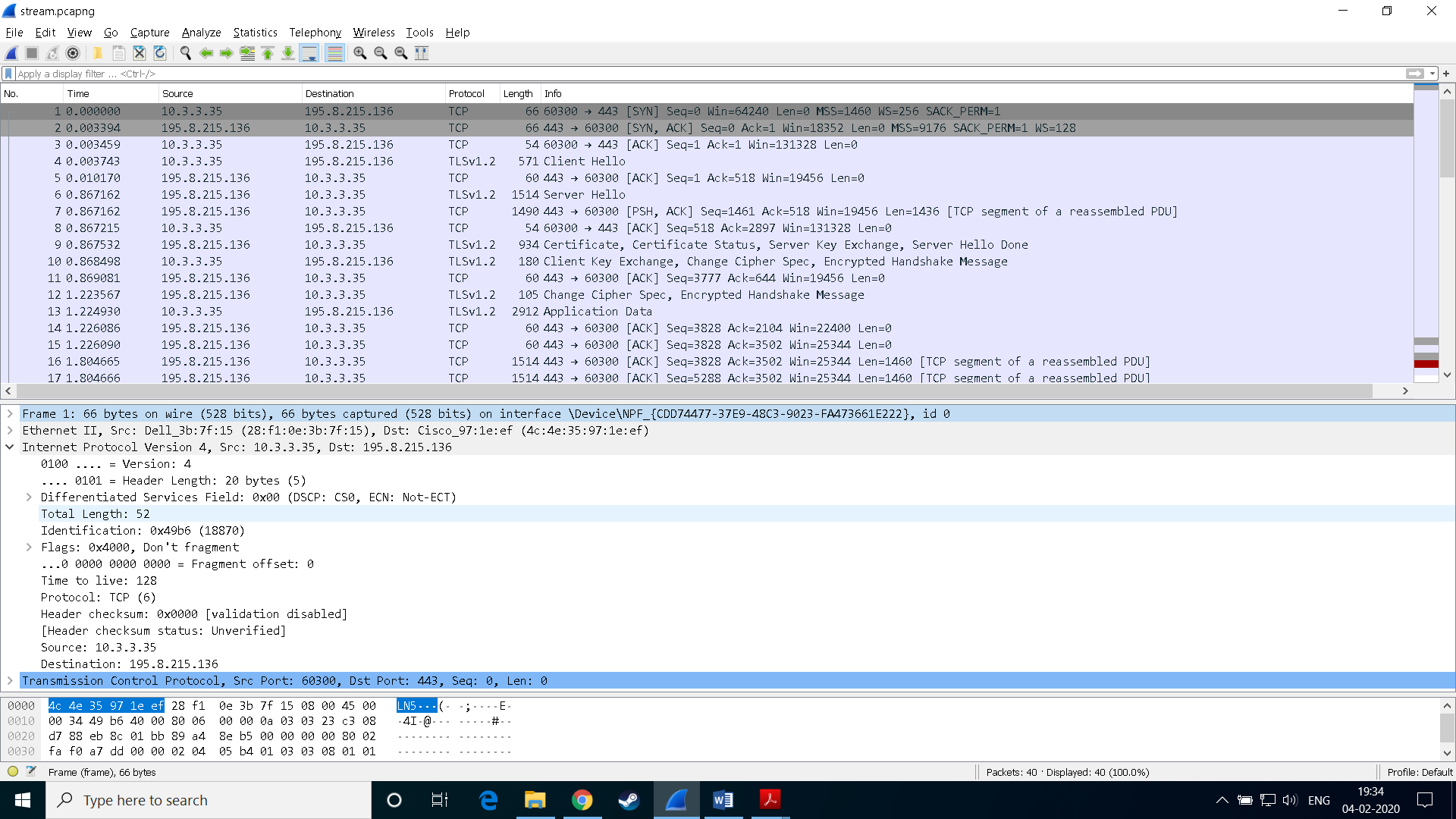
The different protocols used while streaming/downloading/uploading videos on/from Dailymotion were **TCP**, **HTTP**, **TLSv1.2**. Their functions are as follows:

1. **TCP**: Its responsibility includes end-to-end message transfer independent of the underlying network and structure of user data, along with **error control**, **segmentation**, **flow control**, and helps to minimize **traffic congestion control**. It is a connection-oriented protocol that addresses numerous **reliability** issues in providing a reliable byte stream: data arrives in-order, data has minimal error (i.e., correctness), duplicate data is discarded and lost or discarded packets are resent. TCP is optimized for **accurate delivery** rather than timely delivery, as correct sequence of buffer to be fetched. TCP's bandwidth probing and congestion control will attempt to use all of the available bandwidth between server and client.
2. **HTTP:** TCP works in the **Transport layer** while HTTP works in **Application layer** of TCP/IP model. TCP is in charge of setting up a reliable connection between two machines and HTTP uses this connection to transfer data between the web servers and the client in the communication process, we can say connection is fundamentally out of scope of HTTP as it is controlled at Transport Layer. HTTP is a **stateless protocol** though not session-less, meaning that the server does not keep any data (state) between two requests/each request message can be understood in isolation. HTTP follows a classical **client-server model**, with a client opening a connection to make a request, then waiting until it receives a response. HTTP is an **extensible protocol** that is easy to use. The client-server structure, combined with the ability to simply add headers, allows HTTP to **advance** along with the extended capabilities of the Web. Usually, HTTP responses are **buffered** rather than streamed. **HTTP 1.1** added supports for streaming through keep-alive header so data could be streamed. HTTP is Media independent i.e. any type of data can be sent by HTTP as long as both the client and the server know how to handle the data content.
3. **TLSv1.2:** TLS was designed to operate on top of a reliable transport protocol such as TCP. Web servers use cookies to identify the web user. They are small piece of data stored into the web user’s disk. TLS is used to protect **session cookies** on the rest of the sites from being intercepted to protect user accounts. The TLS protocol aims primarily to provide **privacy** and **data integrity** between two communicating computer applications. SSL and TLS are both cryptographic protocols utilizing X.509 certificates, public/private key encryption that provide **authentication** and **data encryption** between servers, machines and applications operating over a network. TLS provides verification of **identity** of server, which is as important as encryption. The goals of the TLS protocol are cryptographic **security**, **extensibility**, **and relative** **efficiency**. These goals are achieved through implementation of the TLS protocol on twolevels: the TLS Record protocol and the TLS Handshake protocol. TLS allows the peers to negotiate **a shared secret key** without having to establish any prior knowledge of each other, and to do so over an unencrypted channel, Client-server applications use the TLS protocol to communicate across a network in a way designed to prevent eavesdropping and tampering.

**Question 4:**

Some of the important functionalities of the application are:

1. **3-way TCP Handshake Message Sequence**: Before a client attempts to connect with a server, the server must first bind to and listen at a port to open it up for connections: this is called a passive open. Once the passive open is established, a client may initiate an active open. Establishing a normal TCP connection requires three separate steps:

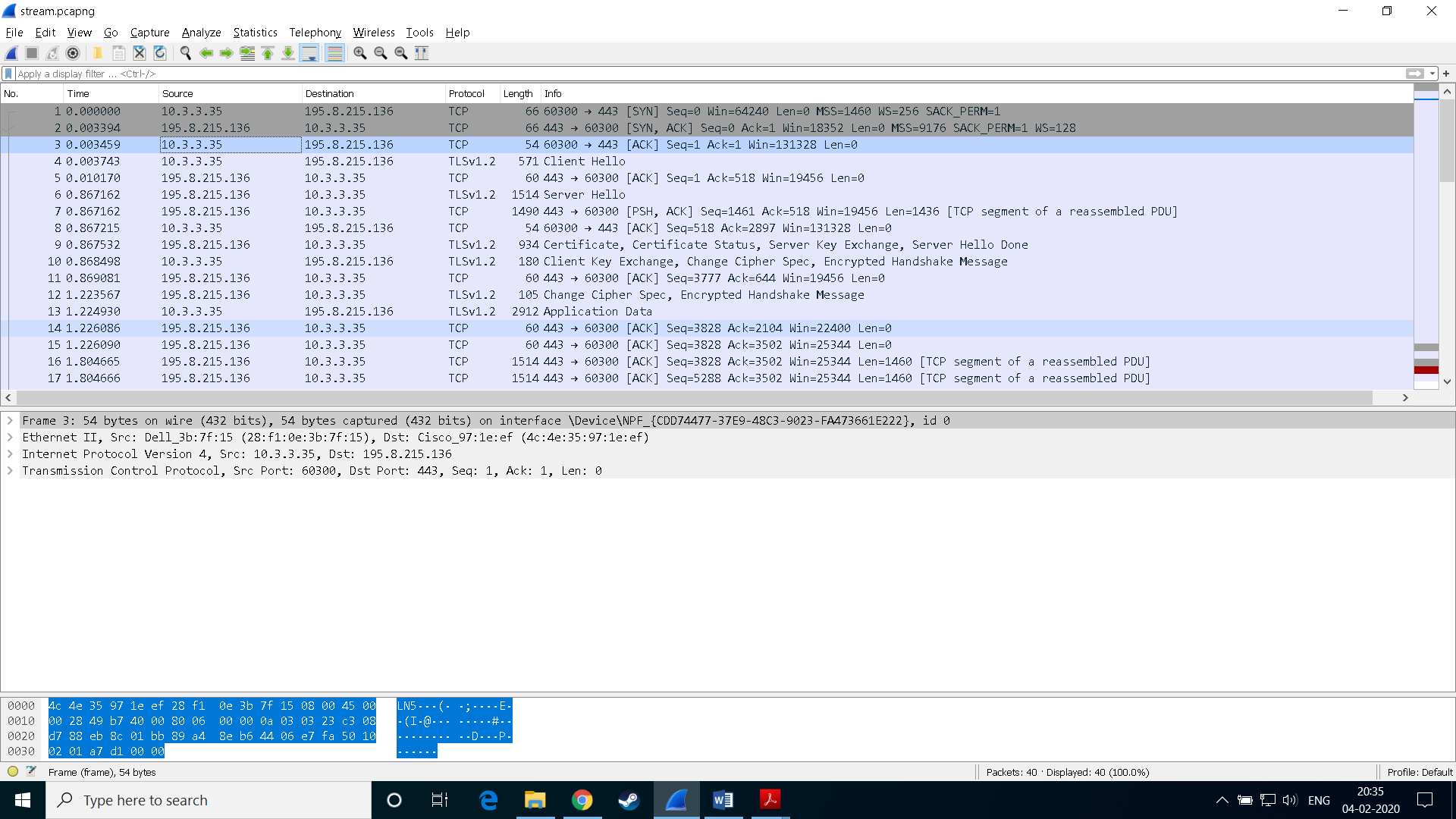


1) **SYN**: The client sending a SYN to the server performs the active open. The clientsets the segment's sequence number to a random value A.

2) **SYN,ACK**: In response, the server replies with a SYN-ACK. The acknowledgment number is set to one more than the received sequence number i.e. A+1, and the sequence number that the server chooses for the packet is another random number, B.

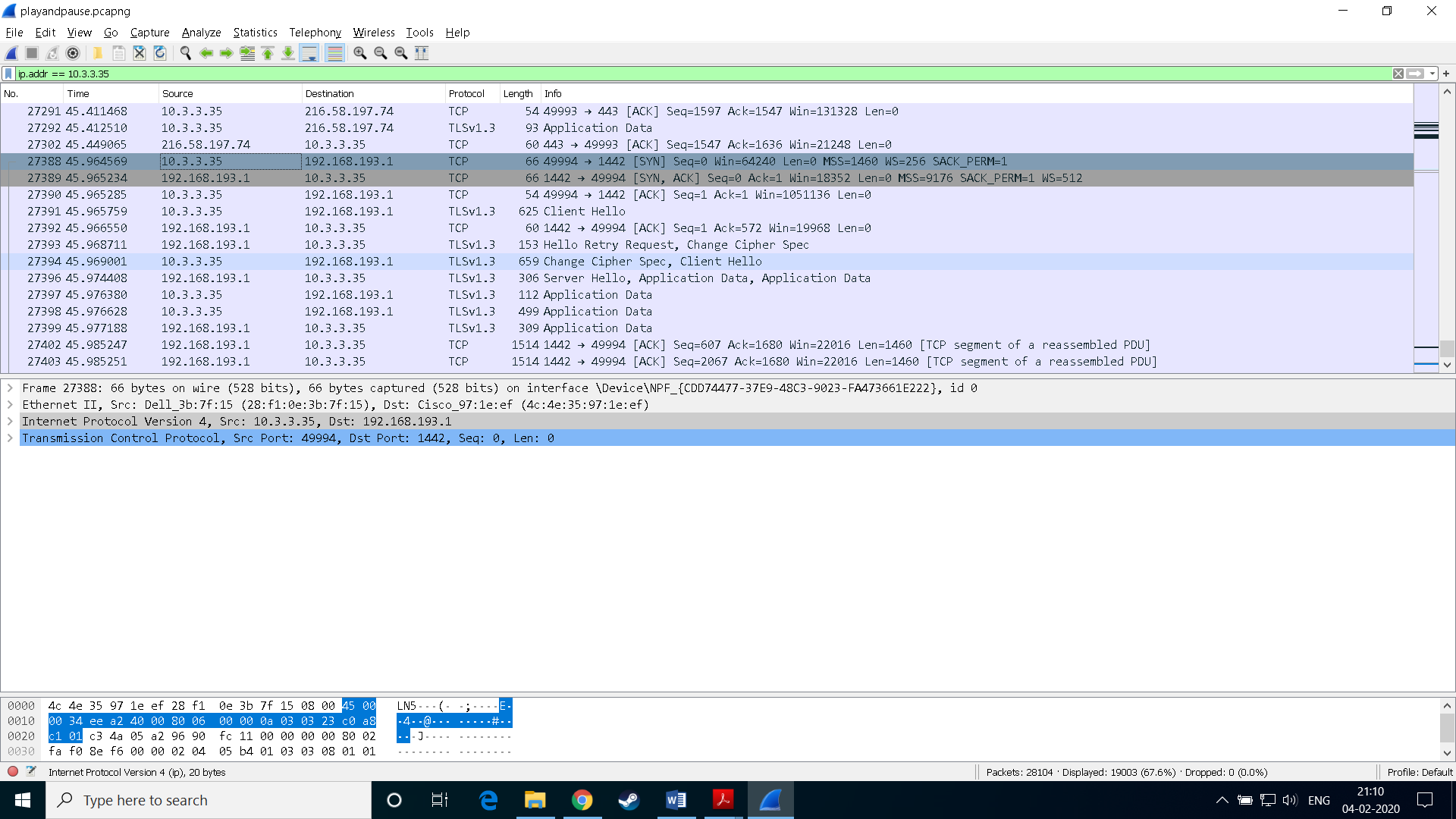
3) **ACK**: Finally, the client sends an ACK back to the server. The sequence number is set to the received acknowledgement value i.e. A+1, and the acknowledgement number is set to one more than the received sequence number i.e. B+1.

1. **TLS Handshaking Message Sequence:** The record encapsulates a "control" protocol (the handshake messaging protocol) when the TLS connection starts. This protocol is used to exchange all the information required by both sides for the exchange of the actual application data by TLS and to negotiate the **secure attributes of a session**. It defines the messages formatting or containing this information and the order of their exchange. These may vary according to the demands of the client and server. The TLS Handshake protocol allows **authenticated communication** to commence between the server and client. This protocol allows the client and server to speak the same language, allowing them to agree upon an **encryption algorithm and encryption keys** before the selected application protocol begins to send data.

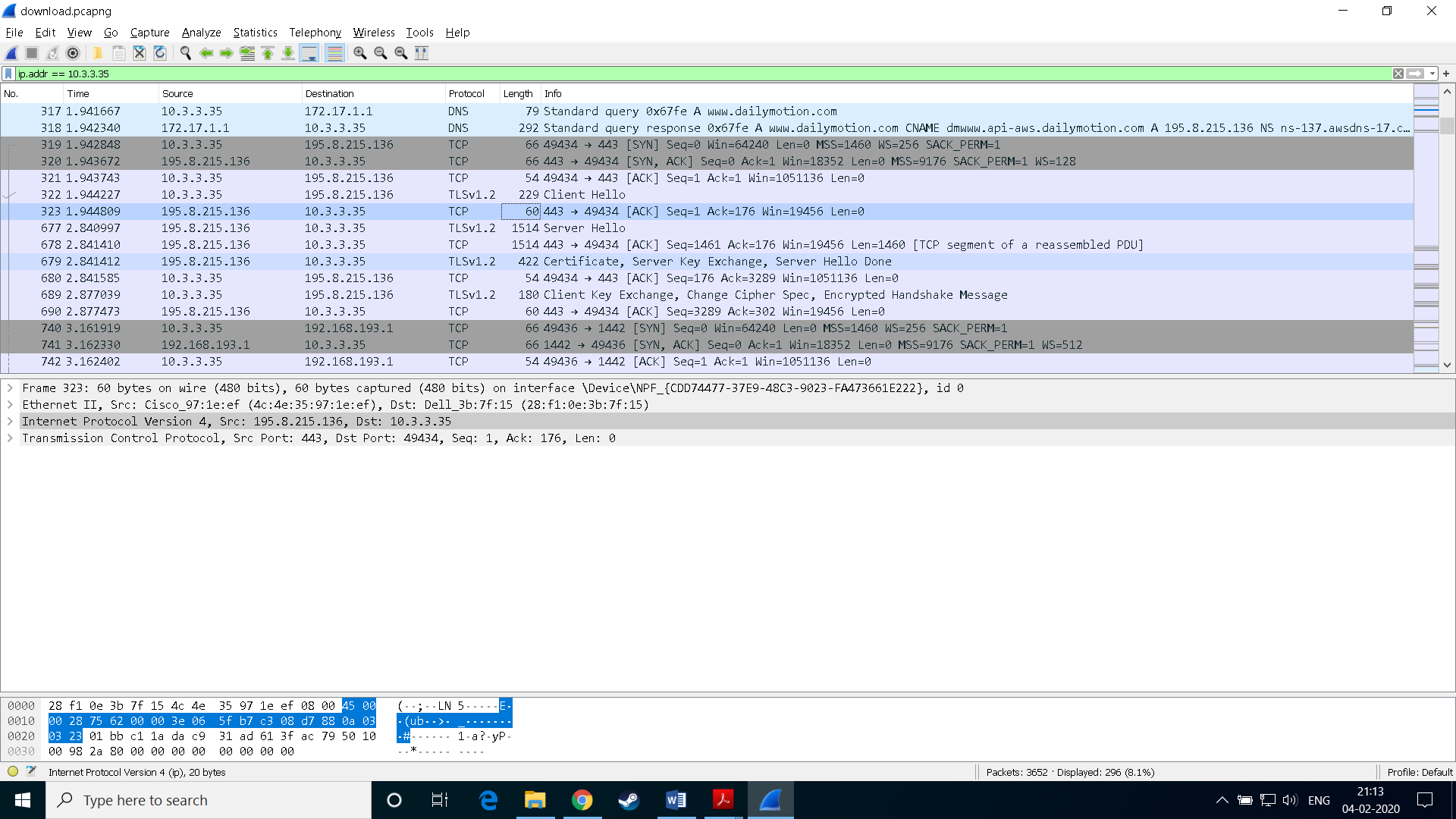
****

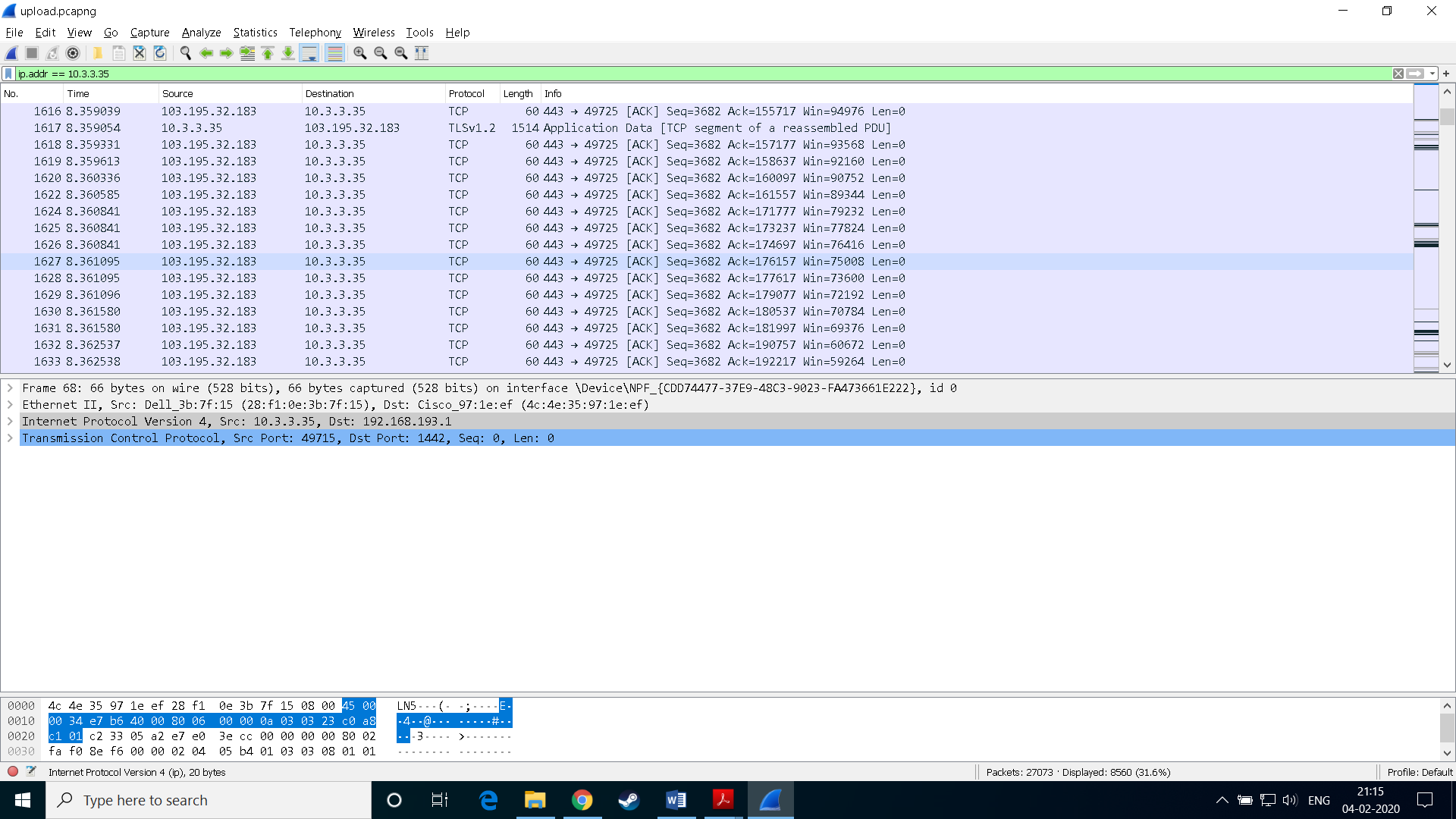
We can see various messageslike ‘**Client Hello**’, ‘**Server Hello**’, ‘**Certificate, Server Key Exchange, Server Hello Done**’, ‘**Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message**’ b/w client(10.3.3.35) and server(195.8.215.136).

1. **Streaming Function:** This function shows a sequence of TCP connection establishment through 3-way handshake, then a HTTP request to the host. This got response from TCP after a TLS handshake to ensure security.

****

1. **Downloading/Uploading Function:** After clicking on the download button, initially connection is established through 3-way handshake, then a HTTP request to the host. Then followed by TLS handshake. Finally, TCP segmentsstarted being transferred from server to our computer for download function while for upload, TCP Segments are being transferred from our computer to the server.

****



A TLS Handshake was used before transferring of TCP Segments to ensure security.

**Question 5:**

The following statistics were obtained using wireshark on 03-02-2020 for [*www.dailymotion.com*](http://www.dailymotion.com) at different times of the day. The activities for which the data has been taken has also been mentioned.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Activity** | **Time** | **Throughput (Packets/sec)** | **RTT(ms)** | **Avg. Packet Size(bytes)** | **Packets Lost** | **UDP Packets** | **TCP Packets** | **Response per request** |
| Download | 10 AM | 221.1 | 0.75 | 913 | 0 | 2563 | 313 | 1.47 |
| Upload | 2 PM | 268.3 | 1.15 | 1391 | 0 | 15141 | 8534 | 1.91 |
| Streaming | 8 PM | 587.9 | 0.78 | 987 | 0 | 7099 | 19021 | 2.02 |

**Question 6:**

The video content is being sent by the application from different sources/servers during the three different times of the day. There can be multiple reasons behind the same. The most probable reason is due to load balancing issues or high traffic in one area. Since Dailymotion keeps multiple servers across different geographical locations to reduce latency & network congestion for various clients, this can also be a reason for different servers responding at different times of the day. This also provides redundancy for the video content in case one of these servers go down for maintenance.

|  |  |
| --- | --- |
| **Time of the day** | **IP Addresses of different hosts** |
| 10 AM | 192.168.193.1,195.8.215.136 |
| 2 PM | 172.217.31.196, 192.168.193.1,195.8.215.129, 103.195.32.183 |
| 8 PM | 172.217.163.131,192.168.193.1, 172.217.31.196,188.65.124.58 |

\* There were many more IPs except the above-mentioned IP Addresses. But the number of requests being sent/received to these hosts were less than 10. These IPs can be of some other websites which might be running parallelly while doing the experiment of Wireshark or might be due to some local update of the OS.

**LINK FOR THE TRACES**

All the traces can be found from the following google drive link:  
<https://drive.google.com/open?id=1-94RkljVV5lxLExSilURi3e5-u5cyN0l>