CS 349 Assignment 2

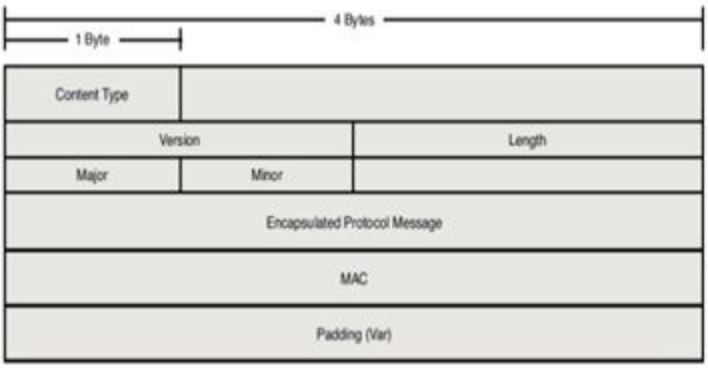
HARIT GUPTA | 170123020

Drive link : https://drive.google.com/open?id=11zM8o1CXo9h7ESz6G_DLMiZujDSclquY

1) Protocols used by the application at different layers

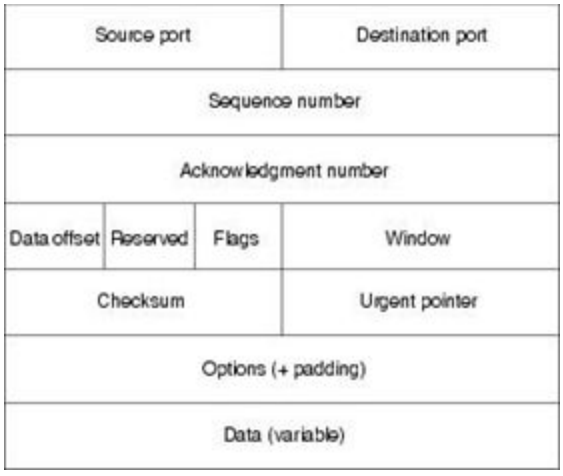
Application Layer

- · **HTTP -** HTTP header fields provide required information about the request or response, or about the object sent in the message body. There are four types of HTTP message headers: ***General-header*** have applicability for both request and response messages, ***Client Request-header*** have applicability only for request messages, ***Server Response-header*** have applicability only for response messages, ***Entity-header*** defines meta information about the entity-body. The header fields are - ***Connection general-header*** allows the sender to specify options that are desired for that particular connection and must not be communicated by proxies over further connections, ***Date***, ***Authorization request-header*** field value consists of credentials containing the authentication information of the user agent, ***Cookie request-header*** field value contains a name/value pair of information stored for that URL, ***From request-header*** contains an Internet e-mail address for the human user who controls the requesting user agent, ***Host request-header*** is used to specify the Internet host and the port number of the resource being requested, ***Proxy-Authorization request-header*** allows the client to identify itself to a proxy which requires authentication, ***User-Agent request-header*** contains information about the user agent, ***Location response-header*** is used to redirect the recipient to a location other than the Request-URI for completion, ***Server response-header*** contains information about the software used by the origin server to handle the request.



- **Secure Sockets Layer**: SSL provides security in the communication between two hosts. It provides integrity, authentication and confidentiality. It can be used with any protocol that uses TCP as the transport layer. 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** can be of four types(Handshake, Change Cipher Spec, Alert, Application Data), **Record Version** is 16-byte value formatted in network order, **Record Length** is 16-byte value.

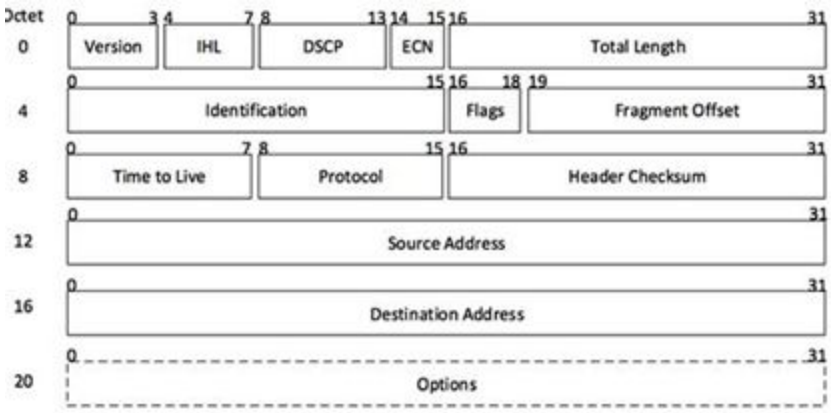
TRANSPORT LAYER



- **Transmission Control Protocol**: TCP segment consists of data bytes to be sent and a header, which can range from 20-60 bytes. The header fields are: **Source and Destination Port Address**: 16 bit fields that holds the port address of the endpoints. **Sequence and Acknowledgement Numbers**: 32 bit fields used for ordering of bytes received. **Header Length:** 4 bit field indicating length of the TCP header. **Control flags**: These are 6 1-bit control bits that control connection establishment, connection termination, connection abortion, flow control, mode of transfer etc. **Window size**: This field tells the window size of the sending TCP in bytes. **Checksum**: This field holds the checksum for error control. **Urgent pointer**: Used to point to data that is urgently required.

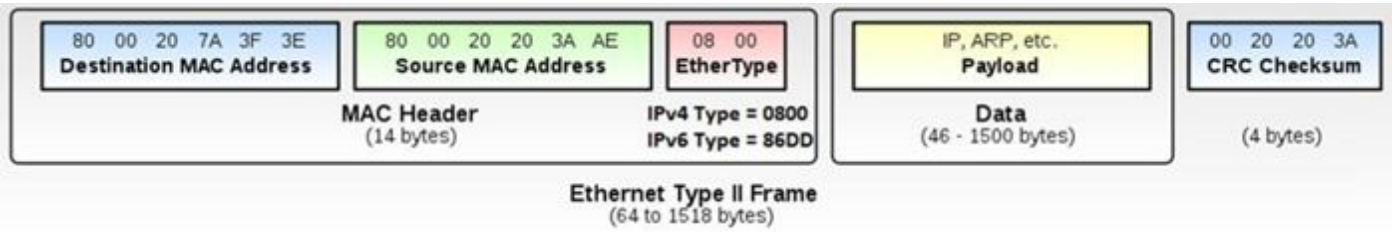
NETWORK LAYER

•**IPv4** is one of the core protocols of standards-based internetworking methods in the Internet. It is a connectionless protocol for use on packet-switched networks. The header consists of 14 fields, of which 13 are required. They are – **Version** is always equal to 4, Internet Header Length (IHL) has 4 bits which is the number of 32-bit words in header, **Differentiated Services Code Point (DSCP)** used in QoS, **Explicit Congestion Notification (ECN)** allows end-to-end notification of network congestion without dropping packets, Total Length is 16-bit field which defines the entire packet size in bytes, **identification field** is primarily used for uniquely identifying the group of fragments of a single IP datagram, flags bit is used to control or identify fragments (0th bit: Reserved and is always 0; 1st bit: **Don't Fragment (DF)**; 2nd bit: **More Fragments (MF))**, Fragment Offset specifies the offset of a particular fragment, **Time To Live (TTL)** helps prevent datagrams from persisting on network forever, Protocol defines the protocol used in the data portion of the IP datagram, **Header Checksum** is used for error-checking of the header, **Source address & Destination address** is the IPv4 address of the sender and receiver of the packet respectively



LINK LAYER

- · **Ethernet(II)** is the most common local area networking technology. It has ***Preamble*** which is 56 bits of alternating 1's and 0's, ***Destination MAC Address, Source MAC Address, Type*** that identifies an upper layer protocol encapsulated by the frame data, ***Length*** of frame and ***Frame Checksum*** for error detection.



2) Observed fields for the protocols used

- The protocols being used are Ethernet, IPv4, TCP, and UDP
  TCP - As shown in the figure, the packet contains destination port, source port, stream index, sequence number, acknowledgement number, header length and flags which have been set as 018 in hexadecimal format. Window size value has been booked at 257. Checksum's value highlights the error detection in the packet. Scaling factor shows the number of leftward bits shifts that are there for a window size.

```
Transmission Control Protocol, Src Port: 80, Dst Port: 57394, Seq: 0, Ack: 1, Len: 0
  Source Port: 80
  Destination Port: 57394
  [Stream index: 20]
  [TCP Segment Len: 0]
  Sequence number: 0    (relative sequence number)
  Sequence number (raw): 788580296
  [Next sequence number: 1    (relative sequence number)]
  Acknowledgment number: 1    (relative ack number)
  Acknowledgment number (raw): 3964457545
  1000 .... = Header Length: 32 bytes (8)
```

UDP – As we can see in the adjoining figure, the packet contains destination port, source port. Along with length of packet being 296. Checksum is there for error detection. There are no flags as shown in TCP above and this is one of an indicator about why UDP is far less complex than TCP. But still TCP ensures much better data reliability than UDP although since TeamViewer decides the protocol by itself depending upon different circumstances, we cannot do anything about it.

```
User Datagram Protocol, Src Port: 60811, Dst Port: 39481
   Source Port: 60811
   Destination Port: 39481
   Length: 1032
   Checksum: 0x2b53 [unverified]
   [Checksum Status: Unverified]
   [Stream index: 0]
 > [Timestamps]
```

IPv4 – Header length here is always 5 which means 20 bytes because it takes 4 bytes word. Packet length is 52 bytes. Don't fragment keyword in flag instructs the nodes through which this packet passes to not break it down. From the figure, it is observable that Time to Live is 128 with protocol for the same being TCP. The packet is being sent from my room IP (10.11.12.6) to proxy server (202.141.80.20).

```
Internet Protocol Version 4, Src: 10.1.1.62, Dst: 239.255.255.250
   0100 .... = Version: 4
   .... 0101 = Header Length: 20 bytes (5)
 > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
   Total Length: 202
   Identification: 0x7e24 (32292)
 > Flags: 0x0000
   ...0 0000 0000 0000 = Fragment offset: 0
   Time to live: 1
   Protocol: UDP (17)
   Header checksum: 0x3fc6 [validation disabled]
   [Header checksum status: Unverified]
   Source: 10.1.1.62
   Destination: 239.255.255.250
```

Ethernet – We can observe here the physical MAC address of devices that are communicating. As can be seen, Destination is the switch and source is my Dell device. Also, a point to note here is that source and destination are unicast and both of them have a global unique address and not a local address.

```
Ethernet II, Src: Cisco_97:1e:ef (4c:4e:35:97:1e:ef), Dst: CompalIn_77:d8:7f (08:97:98:77:d8:7f)
 > Destination: CompalIn_77:d8:7f (08:97:98:77:d8:7f)
 > Source: Cisco_97:1e:ef (4c:4e:35:97:1e:ef)
   Type: IPv4 (0x0800)
```

3) The Relevance of the protocols for the application.

According to what mentioned on TeamViewer's website, once connection is established, the data exchange may use TCP or UDP depending on requirements and conditions.

TeamViewer also uses SLL or TLSv1.2 to ensure that hackers don't snoop packets. In fact, even the TeamViewer developer's team cannot view the original data because of this level of protection.

There are three phases- connection, data transfer and termination between the pairs. During the connection and termination phase, TCP handshake occurs as explained previously. Once connection is established for data transmission TCP is used

- There are three phases- connection, data transfer and termination between the pairs. During the connection and termination phase, TCP handshake occurs as explained previously.
- Once connection is established for data transmission TCP is used ***TCP(Transmission Control Protocol)*** is a **reliable** protocol, meaning that the data that is sent is reached by the receiving party. Data packets that are lost are resent again, if the connection fails then the data is re-requested, thus making sure that data is received at the other end.
- Also as TCP is a **stream-oriented** protocol, it means that it is considered to be along the stream of data that is transmitted from one end of the connection to the other end. The TCP/IP stack is responsible for breaking the stream of data into packets and sending those packets while the stack at the other end is responsible for **reassembling the packets into a data stream** using the information in the packet headers.

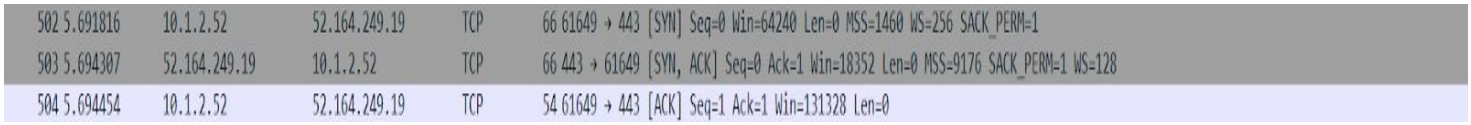
- ***Ethernet II*** is used at the link layer as it ensures **reliable data transfer** between two nodes and involves proper error handling and flow control mechanisms to minimize errors. Other features include CRC for **error detection** and preamble for **synchronization**.
- ***SSL*** is used on top of HTTP to prevent hackers to snoop on the packets being sent between the two endpoints. SSL **encrypts** the message data, which is not readable without the required key. Its basic core function is to protect server-client information.
- TeamViewer also uses SLL or TLSv1.2 to ensure that hackers don't snoop packets. In fact, even the TeamViewer developer's team cannot view the original data because of this level of protection.

4) The Sequence of Messages Exchange

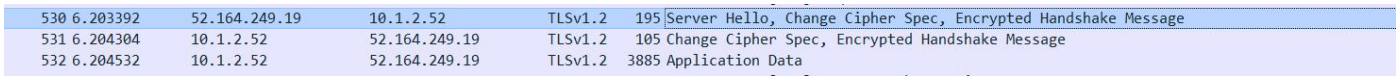
TeamViewer checks network connections to see what protocol it can efficiently use in a situation to make the P2P communication efficient. If both the systems are connected in the same WIFI or same LAN, TeamViewer sees no need to send data through TCP or HTTP/SSL ports but if this is not the case, we see TCP or HTTP/SSL as the protocol along with TLSv1.2 level security so that not even the developer team can decipher the data. The sequence of the message follows broadly the following steps:

- **DNS querying:** First, when the site is loaded, DNS querying is done by the browser. DNS querying is a demand for information sent from the user's computer(DNS client) to a DNS server to ask for the IP address associated with the domain name vimeo.com. As we can see, the querying is for 'A' record type, which is used to relate IP addresses with domain names.

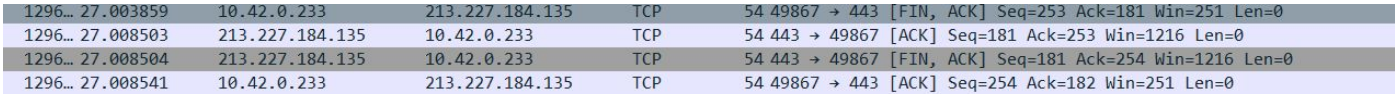
- **TCP 3-way handshake**: It is a process which is used in a TCP/IP network to make a connection between the server(Vimeo) and client(my PC). It is a three-step process where 53386 is client port and port 443 is Vimeo's server:
  - **Step 1:** The client establishes a connection with a server. It sends a segment with the SYN and informs the server about the client should start communication, and with what should be its sequence number.
  - **Step 2:** The server responds to the client request with an SYN-ACK signal set. ACK helps you to signify the response of segment that is received and SYN signifies what sequence number it should able to start with the segments.
  - **Step 3:** In this final step, the client acknowledges the response of the Server, and they both create a stable connection will begin the actual data transfer process.



- **TLS Handshaking:** To make the connection secure, TLSv1.2 is being used. The first message in the TLS Handshake is the *Client Hello* which is sent by the client to initiate a session with the server. The server responds with *Server Hello* and the *Server Certificate (for authentication)*. The server also sends a *Server Key*, which is used by the client to encrypt Client Key Exchange later in the process. *Server Hello Done* indicates that the server is done and is now awaiting the client's response. The client responds with the *Client Key* and is issued a New Session Ticket. The TLS session is now established, and application data can be exchanged between both the server and the client.



- **Connection Termination**: When the communication is over, there is an exchange of messages to terminate the TCP connection. My PC first sends a FIN-ACK message. The server acknowledges it and sends its own FIN-ACK message, which is acknowledged by my PC. This is a four-way handshake to end the connection.



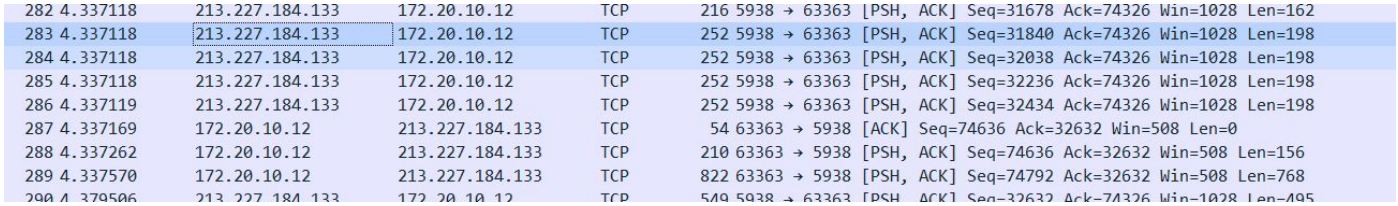
5) Trace Statistics
(This is when i received file from my friend's laptop to my laptop)

Time of the Day	Avg. Throughput (Bytes/sec)	Avg. RTT	Avg. Packet Size(Bytes)	Packets Lost	UDP Packets	TCP Packets	Responses Per Packet
2 PM	1639938	10.70	946	0.1%	26318	81	1
5 PM	642734	11.02	915	0%	7940	68	1
8 PM	1002352	11.64	1086	0%	9479	57	1

6) The Location/Source of content providers

TeamViewer is a peer to peer remote desktop application. Hence, all the data is exchanged between two peers and TeamViewer's site doesn't provide any content. As stated in earlier parts, if trying to connect through LAN, the application connects both the devices through LAN path directly and thus data is always there from one IP and not multiple IP.
In the different combination (systems connected directly to Hotspot), packets are bypassed through TeamViewer servers which route the data using standard HTTP and SSL ports. However, in one of our traces, I connected my system with Airtel No proxy Hotspot while my partner connected his system to Lan and we can see in that particular trace that the communication majorly takes place between IP1 = 172.20.10.12 and IP2 = 213.227.184.133.

We can also observe in the info column about how the handshakes are taking place in this particular case. The IP's mentioned above are the ones responsible for the same.



282	4.337118	213.227.184.133	172.20.10.12	TCP	216 5938 → 63363 [PSH, ACK] Seq=31678 Ack=74326 Win=1028 Len=162
283	4.337118	213.227.184.133	172.20.10.12	TCP	252 5938 → 63363 [PSH, ACK] Seq=31840 Ack=74326 Win=1028 Len=198
284	4.337118	213.227.184.133	172.20.10.12	TCP	252 5938 → 63363 [PSH, ACK] Seq=32038 Ack=74326 Win=1028 Len=198
285	4.337118	213.227.184.133	172.20.10.12	TCP	252 5938 → 63363 [PSH, ACK] Seq=32236 Ack=74326 Win=1028 Len=198
286	4.337119	213.227.184.133	172.20.10.12	TCP	252 5938 → 63363 [PSH, ACK] Seq=32434 Ack=74326 Win=1028 Len=198
287	4.337169	172.20.10.12	213.227.184.133	TCP	54 63363 → 5938 [ACK] Seq=74636 Ack=32632 Win=508 Len=0
288	4.337262	172.20.10.12	213.227.184.133	TCP	210 63363 → 5938 [PSH, ACK] Seq=74636 Ack=32632 Win=508 Len=156
289	4.337570	172.20.10.12	213.227.184.133	TCP	822 63363 → 5938 [PSH, ACK] Seq=74792 Ack=32632 Win=508 Len=768
290	4.379506	213.227.184.133	172.20.10.12	TCP	549 5938 → 63363 [PSH, ACK] Seq=32632 Ack=74326 Win=1028 Len=495