COMPUTER NETWORKS LAB (CS 349)
LAB 01

Prepared by-
Rishabh Agrawal
Roll No.:- 150123032

QUESTION 1-

All the protocols that are used by **www.vimeo.com** at different layers (only those which I can figure out from traces) along with the description of their packet formats are :-

- **Application Layer**

  TLSv1.2- The primary goal of the TLS protocol is to provide privacy and data integrity between two communicating applications. The protocol is composed of **two** layers: the **TLS Record Protocol** and the **TLS Handshake Protocol**. At the lowest level, layered on top of some reliable transport protocol (e.g., TCP), is the TLS Record Protocol. The TLS Record Protocol provides connection security that has two basic properties:

  a) The connection is private. Symmetric cryptography is used for data encryption (e.g., AES, RC4, etc.). The keys for this symmetric encryption are generated uniquely for each connection and are based on a secret negotiated by another protocol (such as the TLS Handshake Protocol). The Record Protocol can also be used without encryption.

  b) The connection is reliable. Message transport includes a message integrity check using a keyed MAC. Secure hash functions (e.g., SHA-1, etc.) are used for MAC computations. The Record Protocol can operate without a MAC, but is generally only used in this mode while another protocol is using the Record Protocol as a transport for negotiating security parameters.

  The TLS Record Protocol is used for encapsulation of various higher- level protocols. One such encapsulated protocol, the TLS Handshake Protocol, allows the server and client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before the application protocol transmits or receives its first byte of data. The TLS Handshake Protocol provides connection security that has **three** basic properties:

  a) The peer's identity can be authenticated using asymmetric, or public key, cryptography (e.g., RSA, DSA, etc.). This authentication can be made optional, but is generally required for at least one of the peers.

  b) The negotiation of a shared secret is secure: the negotiated secret is unavailable to eavesdroppers, and for any authenticated connection the secret cannot be obtained, even by an attacker who can place himself in the middle of the connection.

  c) The negotiation is reliable: no attacker can modify the negotiation communication without being detected by the parties to the communication.

- **Transport Layer**

  TCP- Each TCP header has 10 required fields totalling 20 bytes in size. They can also optionally include an additional data section up to 40 bytes in size. TCP headers has **- Source and destination TCP ports** which are the communication endpoints for sending and receiving devices, **sequence and acknowledgement numbers** to mark the ordering in a group of messages, **data offset** stores the total size of a TCP header in multiples of four bytes, **reserved data** in TCP headers always has a value of zero, a set of six standard and three extended **control flags** (each an individual bit representing on or off) to manage data flow in specific situations, **window size** to regulate how much data sender sends to a receiver before requiring an acknowledgment in return, **checksum** for error detection, **urgent pointer** can be used as a data offset to mark a subset of a message which require priority processing. **Optional TCP data** can be used to include support for special acknowledgment and window scaling algorithms.

- **Network Layer**

  IPv4- It is one of the core protocols of standards-based inter-networking 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

- **Data Link Layer**

  Ethernet(II)- It is the most common local area networking technology. It has Preamble, Destination MAC Address, Source MAC Address, Type that identifies an upper layer protocol encapsulated by the frame data, Length of frame and Frame Checksum.

QUESTION 2-

The observed values for various fields of the protocols are as follows:- Example, Source or destination IP address and port no., Ethernet address, protocol number, etc

QUESTION 3-

The sequence of messages exchanged by the application for using the available functionalities in the application are:-

a) **TCP Connection Handshake-** For establishing a connection with the application, three segments are exchanged between the client and the server which is known as the TCP Three-Way Handshake Process. TCP provides reliable communication with something called Positive Acknowledgement with Re-transmission (PAR). If the data unit received at the receiver's end is damaged, the receiver discards the segment. So the sender has to resend the data unit for which positive acknowledgement is not received. This procedure is illustrated in Figure 4. The packets for 3-way handshake in the trace are highlighted in Figure 5. This mechanism works in the following way:-

   - **Step 1:(SYN)-**The active open is performed by the client sending a SYN (Synchronize Sequence Number) to the server. The client sets the segment's sequence number to a random value X.
   - **Step 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., X+1, and the sequence number that the server chooses for the packet is another random number, Y.
   - **Step 3: (ACK)-** Finally, the client sends an ACK (Acknowledgement) back to the server. The sequence number is set to the received acknowledgement value i.e., X+1, and the acknowledgement number is set to one more than the received sequence number i.e., Y+1.

b) **TLS handshake-** After the TCP handshake, TLS handshake takes place which is implemented by following way:-

   - **ClientHello-** Client sends a ClientHello message containing the supported TLS options. For vimeo, it was also containing the TLS session token extension.
   - **ServerHello-** Server sends a ServerHello message containing the selected TLS options.
   - **Certificate-** Server sends its certificate chain to the client.
   - **ServerHelloDone-** Server completes its part of the negotiation.
   - **ClientKeyExchange-** Client sends encrypted session key to be used for the communication.
   - **ClientCipherSpec-** Client initializes the negotiated options for all the future messages it will send.
   - **Finished -** Client notifies the server to verify the negotiated options for the session.
   - **New Session Ticket-** Server stores its session state (such as ciphersuite and master secret) to a ticket that is encrypted and integrity-protected by a key known only to the server. The ticket is distributed to the client using the NewSessionTicket. The client caches this ticket along with the master secret and other parameters associated with the current session. When the client wishes to resume the session, it includes

the ticket in the SessionTicket extension within the ClientHello message. The server then decrypts the received ticket, verifies the ticket's validity, retrieves the session state from the contents of the ticket, and uses this state to resume the session.

- **ChangeCipherSpec-** Server initializes the negotiated options for all the future messages it will send.
- **Finished-** Server notifies the client to verify the negotiated options for the session and indicating that the server part of the handshake is complete.
- The server and client can now exchange messages that are symmetrically encrypted with the shared secret key.

c) **TCP Termination Handshake-** In the normal case, each side terminates its end of the connection by sending a special message with the FIN (finish) bit set. This message serves as a connection termination request to the other device. The device receiving the FIN responds with an acknowledgment to the FIN to indicate that it was received. The connection as a whole is not considered terminated until both sides have finished the shut down procedure by sending a FIN and receiving an ACK. Thus, termination isn't a three-way handshake like establishment: it is a pair of two-way handshakes. This states that the two devices in the connection move through during a normal connection shutdown are different because the device initiating the shutdown must behave differently than the one that receives the termination request. In particular, the TCP on the device receiving the initial termination request must inform its application process and wait for a signal that the process is ready to proceed. The initiating device doesn't need to do this, since the application is what started the ball rolling in the first place.

QUESTION 4-

In following way the particular protocols listed below relevant for functioning of the application:-

a) **TLSv1.2-** HTTP is not encrypted and is vulnerable to man-in-the-middle and eavesdropping attacks, which can let attackers gain access to website accounts and sensitive information, and modify webpages to inject malware or advertisements. Using HTTPS, the computers agree on a ”code” between them, and then they scramble the messages using that ”code” so that no one in between can read them. This keeps your information safe from hackers. They use the ”code” on a Secure Sockets Layer (SSL), sometimes called Transport Layer Security (TLS) to send the information back and forth. So, this security protocol protects the integrity of the website by helping to prevent intruders tampering with communications between the site and the visitors browsing (a common tactic here is injecting malware) as well as safeguarding privacy and security. It makes sense that there are a host of benefits for making a site more secure, chief among them is the ability to safeguard sensitive data and the peace of mind that comes from knowing the domain is protected from disasters such as malicious ads or spyware being injected into the site and displayed to users when communications aren't protected. Also, the security tiebreaker approach means that if your site and another site are essentially comparable and vying for top ranking for a particular keyword, the addition of https in your domain could be enough to give you the edge and secure the top spot.

b) **TCP-**

Vimeo videos are not real-time streaming videos. They are simply videos which you fetch and watch once buffered. For real time videos, using UDP has the merit that it's faster and has lower overhead, and you don't care if you drop a few packets in between. But, not so with vimeo videos. Vimeo buffers them and you can watch them again, rewind it, pause it etc. So, we definitely need a reliable transmission protocol like TCP. So if there's a missing packet which causes a glitch in the video, TCP will let the packet to be retransmitted which is not the case with UDP. Vimeo also adjusts video quality based on network congestion, and this can be detected by TCP. So, vimeo needs everything that TCP provides (receive windows, reordering, duplicate rejection, and so on). They would either have to use TCP or try to do all those things themselves. There's no way they could do that better than each operating system's optimized TCP implementation.

QUESTION 6-

The IP address of the content provider of the application are 151.101.64.217, 151.101.128.217, 151.101.192.217, 151.101.0.217, 151.101.10.109. The main reasons for having multiple source servers are as follows:-

- Multiple servers are used for load balancing. It improves the distribution of workloads across multiple computing resources. It aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource.

- If an application runs on two or more servers, the failure of any one server will not do much damage and the application will still continue to work. This makes the application more reliable.
- A single server which is very powerful may cost as much as ten times compared to a server which is half powerful. So, there is a cost factor driving the need of multiple servers.