

NOTE: Screenshots for particular object in each question is attached separately.

Link for Trace Files:

<https://drive.google.com/drive/folders/1f4jvXk9V9g1eP8Jy8lYa1XeL7ePls6Ux?usp=sharing>

Question 1:

Various protocols used by Twitch at different layers are :

- ✓ Transport layer: TCP, UDP, ICMP
- ✓ Network layer: IPv4, ICMPv6
- ✓ Application layer: DNS, HTTP, SSDP
- ✓ Session layer: TLSv1.2 and TLSv1.3
- ✓ Physical layer: Ethernet II Protocol
- ✓ Link layer: ARP

NOTE: Some protocols are multilayered but mentioned only once.

Question 2:

1) Frame 2081: The frame number of the trace I took for observation is 2081. The frame protocol is not a real protocol itself, but used by Wireshark as a base for all the protocols on top of it. We also have information about the bytes used by the frame (499 bytes)

2) Ethernet II:

Src: PcsCompu_02:af:5d (08:00:27:02:af:5d) : Mac address of my computer (Source).

Dst: aa:5e:a4:2d:70:a8 (aa:5e:a4:2d:70:a8) : Mac address of destination.

3) Internet Protocol Version 4:

- Src: 192.168.43.75 – This is my computer IP address (Source IP Address). Dst: 52.24.254.52 – This destination computer IP address. These IP addresses help us to uniquely identify both machines on the network.
- Version: 4: field indicates the format of the internet header.
- Header length: 20 bytes: Number of 32-bit words in the TCP header.
- Differentiated Services Field (DSF): 0x00: Indicates particular Quality of Service needs from the network, the DSF defines the way routers should queue packets while they are waiting to be forwarded.
- Total Length: 485: is the length of the datagram, measured in octets, including internet header and data. This field allows the length of a datagram to be up to 65,535 octets.

- Identification: 0x0a1c (2588) : it is a 16-bit value that is unique for every datagram for a given source address, destination address, and protocol, such that it does not repeat within the maximum datagram lifetime
- Flags: As required by the network resources, if IP Packet is too large to handle, these 'flags' tells if they can be fragmented or not. In this 3-bit flag, the MSB is always set to '0'.
- Fragment Offset: 0 bits: This field indicates where in the datagram this fragment belongs.
- Time to live: 64: This field indicates the maximum time the datagram is allowed to remain in the internet system.
- Protocol (Same as protocol number): TCP (6): Tells the Network layer at the destination host, to which Protocol this packet belongs to, i.e., the next level Protocol. For example, protocol number of ICMP is 1, TCP is 6 and UDP is 17.
- Header checksum: 0x83ce: A checksum on the header only and its status is unverified.

4) Transmission Control Protocol:

- Src Port: 58978: The sending device's port, my computer port in this case.
- Dst Port: 80: This is receiving/Destination port which is receiving packets.
- TCP Segment Length: 433: This is TCP packet segment length.
- Sequence number: 1: A device initiating a TCP connection must choose a random initial sequence number, which is then incremented according to the number of transmitted bytes.
- Acknowledgment number: 1: The ACK flag is set to 1 which means the value of this field is the next sequence number that the sender is expecting.
- Flags (9 bits): It contains 9 1-bit flags. ACK (1 bit):1: indicates that the Acknowledgment field is significant. All packets after the initial SYN packet sent by the client should have this flag set. PSH (1 bit):1: Push function. Asks to push the buffered data to the receiving application.
 Remaining seven flags i.e., NS, CWR, ECE, URG, RST, SYN, FIN have value 0.
- Window size value: 502: This is the space for the incoming data.
- Checksum (16 bits): 0x2018: The 16-bit checksum field is used for error checking of the header and data.
- Urgent pointer (16 bits): 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.

5) TLSv1.2 Protocol: Transport Layer Security (TLS): Some of the fields of the protocol can be seen. Some other messages include Client Hello, Server Hello, Certificate, Server Key Exchange, Server Hello Done between client (192.168.43.75) and server (52.24.254.52) and TLS is implemented on two levels- The TLS Record protocol and the TLS Handshake protocol to ensure security, efficiency and extensibility.

6) Domain Name System:

1. **Flags:** This tells about any error if detected (No error).
2. **Queries:** Website for which ip address is required. (twitch.com: type A, class IN)
3. **Answers:** IP address of the required website (twitch.com: type A, class IN, addr 54.24.254.52)

4. Time: This field tells the RTT time (0.400969489 seconds)

7) User Datagram Protocol:

1. Source Port: Port Number of the application in source machine (59184)
2. Destination Port: Port Number of the application in destination machine (443)
3. Length: The total length of the UDP packet. (1358)
4. Checksum: Used to check if the packet was transmitted correctly or not. (0x45cf)

8) Address Resolution Protocol:

1. Sender MAC address: MAC address of the sender (aa:5e:a4:2d:70:a8)
2. Sender IP address: IP address of the sender (192.168.43.1)
3. Target MAC address: MAC address of the receiver (00:00:00:00:00:00)
4. Target IP address: IP address of the receiver (192.168.43.75)

9) Hypertext Transfer Protocol:

- GET HTTP/1.1\r\n: This is start line, indicating it is request packet.
- Host: twitch.com\r\n: This tells hostname to which requesting the packet.
- Connection: keep-alive\r\n: This is connection status, which is active here.
- User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4386.0 Safari/537.36 Edg/89.0.767.0\r\n: This is part of requesting header. Tells about host information about the browser and operating system.

Question 3:

NOTE: Both activities 1 and 2 refer to video play and pause function. Activity 3 refers to search function.

The following exchange of messages was observed when I visited live streaming website twitch:

- A DNS query was sent to get IP address corresponding to the domain <http://twitch.com>
- A tcp connection is initiated by performing a 3-way handshake with destination server.
- A HTTP GET request is send to get the webpage and other attachments.
- We receive the requested data through tcp connection already established.
- Termination of the connection is achieved by sending FIN TCP packet.

3-way Handshake:

1. Step 1 (SYN): In the first step, client wants to establish a connection with server, so it sends a segment with SYN (Synchronize Sequence Number) which informs server that client is likely to start communication and with what sequence number it starts its' segments.
2. Step 2 (SYN + ACK): Server responds to the client request with SYN-ACK signal bits set. Acknowledgement (ACK) signifies the response of segment it received and SYN signifies with what sequence number it is likely to start its' segments.
3. Step 3 (ACK): In the final part client acknowledges the response of server and they both establish a reliable connection with which they will start the actual data transfer.

TLS Handshake:

This protocol takes care of authentication and key exchange necessary to establish or resume secure (encrypted) session. 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. Following steps occur for this:

(1) The client sends a "Client hello" message to the server, along with the client's random value and supported cipher suites. (2) The server responds by sending a "Server hello" message to the client, along with the server's random value. (3) The server sends its certificate to the client for authentication and may request a certificate from the client. The server sends the "Server hello done" message. (4) If the server has requested a certificate from the client, the client sends it. (5) The client creates a random Pre-Master Secret and encrypts it with the public key from the server's certificate, sending the encrypted Pre-Master Secret to the server. (6) The server receives the Pre-Master Secret. The server and client each generate the Master Secret and session keys based on the Pre-Master Secret. (7) The client sends "Change cipher spec" notification to server to indicate that the client will start using the new session keys for hashing and encrypting messages. Client also sends "Client finished" message. (8) Server receives "Change cipher spec" and switches its record layer security state to symmetric encryption using the session keys. Server sends "Server finished" message to the client. (9) Client and server can now exchange application data over the secured channel they have established. All messages sent from client to server and from server to client are encrypted using session key.

Data Transfer:

The client requests for webpage and its content using a GET request. Then data exchange happens using TCP protocol with the use of various ACKs and [PSH, ACK] s. TCP's push capability accomplishes two things: The sending application informs TCP that data should be sent immediately. The PSH flag in the TCP header informs the receiving host that the data should be pushed up to the receiving application immediately. So, the server tells the client at various intervals that it has no more data to send and requests an acknowledgement immediately to which the client also replies with an ACK.

Video Play/Pause:

First TCP connection is established through 3-way handshake and also a TLS handshake to ensure security. Thereafter, we receive response through continuously getting TCP packets transfer and obtaining application data until the video is running. When we pause or close the live stream video ongoing TCP connection terminates and we receive **FIN** packet from the server.

Search:

When we search something, initially connection is established through 3-way handshake, then a HTTP GET request is sent. Then followed by TLS handshake, finally, TCP segments started being transferred from server to my computer.

Termination:

The connection termination phase uses a four-way handshake, with both side of the connection terminating independently. When an endpoint wishes to end the connection, it transmits a FIN packet, which the other end acknowledges with an ACK. The other endpoint also sends a FIN and expect a ACK

from the first endpoint. After both FIN/ACK exchanges are concluded, the side which sent the first FIN before receiving one waits for a timeout before finally closing the connection, during which time the local port is unavailable for new connections

Question 4:

Protocol	Function
DNS	It is used to associate the domain with the appropriate IP address. DNS servers distributed throughout the world convert domain names into IP addresses, thereby taking control of which server a user can access via a specific domain.
HTTP	The communications protocol used to connect to Web servers on the Internet or on a local network (intranet). Its primary function is to establish a connection with the server and send HTML pages back to the user's browser.
SSDP	A standard for advertising services on a TCP/IP network and discovering them. The Universal Plug and Play (UPnP) protocol uses SSDP to announce and find devices in order, for example, to stream video from a source to a playback system.
TLS	TLS is a cryptographic protocol that provides end-to-end communications security over networks and is widely used for internet communications and online transactions. It is an IETF standard intended to prevent eavesdropping, tampering and message forgery.
TCP	TCP is used for organizing data in a way that ensures the secure transmission between the server and client. It guarantees the integrity of data sent over the network, regardless of the amount. For this reason, it is used to transmit data from other higher-level protocols that require all transmitted data to arrive
UDP	It is primarily used for establishing low-latency and loss-tolerating connections between applications on the internet. It speeds up transmissions by enabling the transfer of data before an agreement is provided by the receiving party.
IP	These addresses are the unique numbers assigned to every computer or device that is connected to the Internet. Among other important functions, they identify every device connected to the Internet, whether it is a web server, smartphone, mail server, or laptop.
ICMP	This protocol communicates information about network connectivity issues back to the source of the compromised transmission. It sends control messages such as destination network unreachable, source route failed, and source quench.
ARP	It provides two basic functions: Resolving IPv4 addresses to MAC addresses and Maintaining a table of mappings

Question 5:

Index	1	2	3	Act_1	Act_2	Act_3
Time	2021-01-31 15:57:45	2021-01-31 20:10:27	2021-02-01 16:39:35	2021-02-01 17:58:52	2021-02-01 18:50:13	2021-02-01 22:26:33
Throughput (in bytes/sec)	100k	110k	59k	364k	186k	310k
Total no. Of packets	5677	3308	4124	2413	6084	3478
RTT (msec)	341.662	385.277	330.510	28628.43	26453.64	210.673
Avg. Packet size (Bytes)	946946	993	677	1831	1019	915
No. of Packets lost	64	21	14	5	30	34
No. Of TCP packets	5462	3035	3612	2395	6009	3450
No. Of UDP packets	199	266	483	12	64	22
No. Of responses per one request sent	1.37	1.08	1.03	1.47	1.23	1.20

RTT is for HTTP packets.

Formula used for No. Of Responses per one request = (No. Of packets displayed with filter 'ip.dst == 192.168.43.75 and ip.src != 192.168.43.1') / (No. Of packets displayed with filter 'ip.src == 192.168.43.75 and ip.dst != 192.168.43.1')

Here this formula calculates : No. Of packets received by next hop but sent by my pc / No. Of packets sent by next hop but not received by my pc.

Calculations for throughput:

Throughput = bytes transferred / time span

Throughput for trace 1 = 5368567 bytes / 53.479 sec = 100386.45 ~ = 100k bytes/sec

Throughput for trace 2 = 3284776 bytes / 29.663 sec = 110736.47 ~ = 110k bytes/sec

Throughput for trace 3 = 2791953 bytes / 46.605 sec = 59906.73 ~ = 59k bytes/sec

Throughput for trace act_1 = 4417792 bytes / 12.129 sec = 364223.82 ~ = 364k bytes/sec

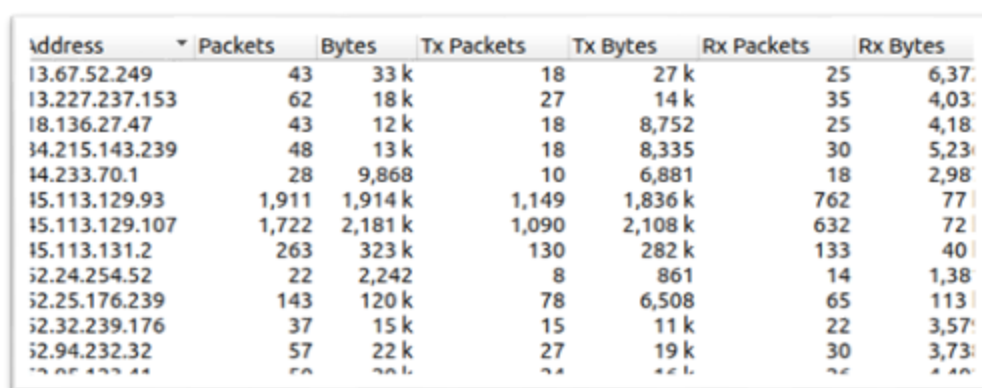
Throughput for trace act_2 = 6198907 bytes / 33.196 sec = 186736.56 ~ = 186k bytes/sec

Throughput for trace act_3 = 3182697 bytes / 10.237 sec = 310901.33 ~ = 310k bytes/sec

Question 6:

Actually, may or may not be checked if the whole content is sent from the same location/source.

- When we are behind a **proxy** authenticated network, we **cannot check** whether the whole content is being sent from same location. A proxy server is an intermediate program or computer used when navigating through different networks of the Internet. They facilitate access to content on the World Wide Web. A proxy intercepts requests and serves back responses.
- But when we don't have proxy behind, we can check this. When I used my **mobile network**, twitch.tv transmitted tcp packets, from **multiple IP's**. This is evident from below screenshot:



address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
13.67.52.249	43	33 k	18	27 k	25	6,37
13.227.237.153	62	18 k	27	14 k	35	4,03
18.136.27.47	43	12 k	18	8,752	25	4,18
14.215.143.239	48	13 k	18	8,335	30	5,23
14.233.70.1	28	9,868	10	6,881	18	2,98
15.113.129.93	1,911	1,914 k	1,149	1,836 k	762	77
15.113.129.107	1,722	2,181 k	1,090	2,108 k	632	72
15.113.131.2	263	323 k	130	282 k	133	40
12.24.254.52	22	2,242	8	861	14	1,38
12.25.176.239	143	120 k	78	6,508	65	113
12.32.239.176	37	15 k	15	11 k	22	3,57
12.94.232.32	57	22 k	27	19 k	30	3,73
12.95.133.44	58	28 k	28	22 k	30	4,48

Reason for multiple IP's:

Nowadays, websites deploy load balancing on servers, so that the data is sent to clients in most efficient manner from different IP's. Round Robin DNS mechanism for faster fetching of relevant pages by balancing the page requests across many servers may lead to multiple IP's. Also, sometimes the video is streamed from one IP and other site data is being loaded from another server with different IP. Thus, leading to multiple IP's.