CS349

Networks Lab
Assignment 2

Hotstar Streaming

Arkadeep DAS
150123053

1. List of all **protocols** and their **packet formats**:

a) **Internet Protocol Version 4**
Src [IP address of source] : 172.16.114.191   Dest [IP address of destination]: 202.141.80.24
Version:(4 bits): Indicates the version number, to allow evolution of the protocol.
Header Length: Total Length [total datagram length ,in octets]: 557
Header checksum: to protect the header of IPv4 data packets against data corruption
 Fragment offset [The fragment offset field identifies the order in which to place the packet fragment in the reconstruction] : 0
Flags(0x02) : The Don't Fragment (DF) flag is a single bit in the Flag field that indicates that fragmentation of the packet is not allowed.
Time to live : [8-bit binary value that indicates the remaining "life" of the packet]
Protocol [This 8-bit binary value indicates the data payload type that the packet is carrying]: TCP (6)
.Destination GeoIP: India, India [Geographical location of IP is India]

b) **Transmission Control Protocol** , [The Transport Layer]
Src Port ,Dst Port [identify the end points of the connection connection is from source,16 bits each]
Window field (16 bits) specifies the size of the sender's receive window (that is, buffer space available for incoming data)
Flags field (6 bits) contains the various flags: URG—Indicates that some urgent data has been placed. ACK—Indicates that acknowledgement number is valid. PSH—Indicates that data should be passed to the application as soon as possible. RST—Resets the connection. SYN—Synchronizes sequence numbers to initiate a connection. FIN—Means that the sender of the flag has finished sending data.
Checksum field (16 bits) indicates whether the header was damaged in transit.
Data Offset (a.k.a. Header Length) field (variable length) tells how many 32-bit words are contained in the TCP header
Window field (16 bits) specifies the size of the sender's receive window (that is, buffer space available for incoming data).
Acknowledgement 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.
Sequence Number field (32 bits) specifies the number assigned to the first byte of data in the current message.

c) **Hypertext Transfer Protocol** [The application layer]
Request methods : GET [requests a representation of the specified resource], CONNECT[converts the request connection to a transparent TCP/IP tunnel],
request uri :Uniform Resource Identifier and identifies the resource upon which to apply the request
request header fields : allows the client to pass additional information about the request, and about the client itself, to the server.
Host : Address of the Host
status code : 3-digit integer where first digit of the Status-Code defines the class of response and the last two digits do not have any categorization role

server ,
Content-type: indicates the media type of the resource ,
Content-length : length of the response

d) **Secure Sockets Layer**
Content Type: Handshake (22) [TLS Record protocol which 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.
Version[The version of the protocol being employed]: TLS 1.2.
Length[The length (in bytes) of the following TLSPlaintext.fragment]
Handshake Protocol: Client Hello [responsible for negotiating a session]

2. For Packet no. 11096

a)**Internet Protocol Version 4** :
Source address : 172.16.114.191(IP address of source from where request for packets is sent)
Destination address : 202.141.80.24 (IP address of destination from where response is sent back)
Protocol : TCP (a standard that defines how to establish and maintain a network conversation via which application programs can exchange data)
Protocol number : 6 [In an IP header, the Protocol field identifies the service in the next higher level in the protocol stack to which data is passed. Protocol numbers are used to configure firewalls, routers and proxies]
Header checksum : 0x6fb4 to protect the header of IPv4 data packets against data corruption.
Flags: 0x02 (Don't Fragment) [The Don't Fragment (DF) flag is a single bit in the Flag field that indicates that fragmentation of the packet is not allowed].
Differentiated Services Field 0000 00.. = Differentiated Services Codepoint: Default (0)
0000 00.. = Differentiated Services Codepoint: Default (0)
that specifies a simple and scalable mechanism for classifying and managing network traffic and providing quality of service (QoS) on modern IP networks
, GeoIP (Source and Destination) - India
Source Ethernet Address: Hangzhou_0c:ef:99 (38:22:d6:0c:ef:99)
[MAC address of the source computer]
Destination Ethernet Address : Address: Dell_25:6a:c5 (74:86:7a:25:6a:c5)
[MAC address of the destination computer] SInce it is a response packet , the destination is the client computer.

b) **TCP** : Source Port : 3128
Dest Port : 60527 [identify the end points of the connection connection is from source,16 bits each]
Stream Index : 89 , tcp segment length : 8 , sequence no : 1269174 [specifies the number assigned to the first byte of data in the current message].
, acknowledgement no. : 1500,[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].
Flags : 0x018 (PSH,ACK) , window size : 138,specifies the size of the sender's receive window (that is, buffer space available for incoming data).
checksum :0xfde8 checksum status :unverified , TCP payload : 8 bytes , TCP segment data (8 bytes)

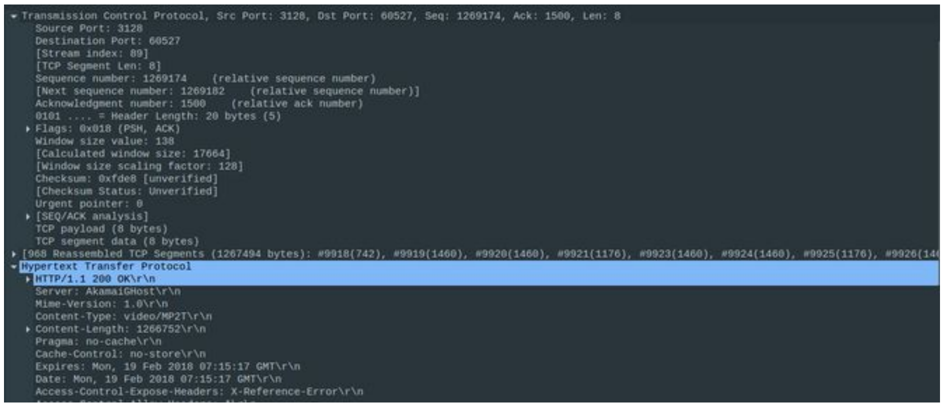
c) **HTTP** -
Server :AkamaiGHost [a computer system that processes requests via HTTP] Mime-version : 1.0 ,
Content-type : video/mp2t [the type of the media file obtained in the response], Content Length : 1266752 [the total length of the request body in octets]
FIle Data: 1266752 bytes [the size of the response body in bytes].
Status code :200 [Ok,Success] Pragma: no-cache Cache-Control: no-store

X-Cache: MISS from barapani.iitg.ernet.in [Prefix 'X' in X-Cache indicates that the header is not a standard HTTP Header Field.]
X-Cache-Lookup: MISS from barapani.iitg.ernet.in:3128

d) **Secure Sockets Layer :** [Packet 109]

Content Type: Handshake (22)TLS Record protocol which 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.

Version: TLS 1.2 (0x0301) Length: 512 Handshake Protocol: Client Hello [responsible for negotiating a session]

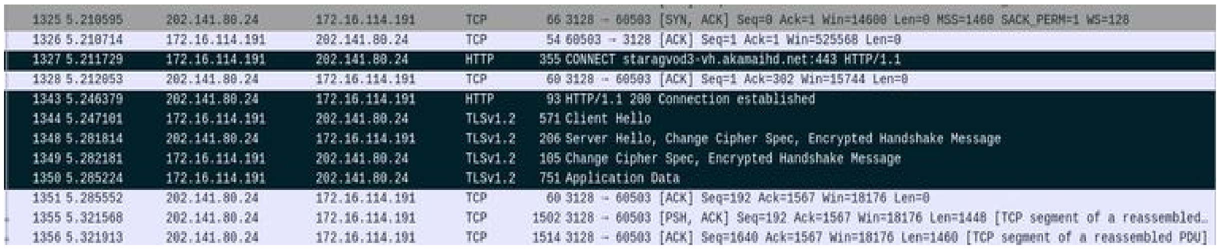


```
▾ Transmission Control Protocol, Src Port: 3128, Dst Port: 60527, Seq: 1269174, Ack: 1500, Len: 8
    Source Port: 3128
    Destination Port: 60527
    [Stream index: 89]
    [TCP Segment Len: 8]
    Sequence number: 1269174    (relative sequence number)
    [Next sequence number: 1269182    (relative sequence number)]
    Acknowledgment number: 1500    (relative ack number)
    0101 .... = Header Length: 20 bytes (5)
  ▸ Flags: 0x018 (PSH, ACK)
    Window size value: 138
    [Calculated window size: 17664]
    [Window size scaling factor: 128]
    Checksum: 0xfde8 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▸ [SEQ/ACK analysis]
    TCP payload (8 bytes)
    TCP segment data (8 bytes)
▸ [968 Reassembled TCP Segments (1267494 bytes): #9918(742), #9919(1460), #9920(1460), #9921(1176), #9923(1460), #9924(1460), #9925(1176), #9926(14
▾ Hypertext Transfer Protocol
  ▸ HTTP/1.1 200 OK\r\n
    Server: AkamaiGHost\r\n
    Mime-Version: 1.0\r\n
    Content-Type: video/MP2T\r\n
  ▸ Content-Length: 1266752\r\n
    Pragma: no-cache\r\n
    Cache-Control: no-store\r\n
    Expires: Mon, 19 Feb 2018 07:15:17 GMT\r\n
    Date: Mon, 19 Feb 2018 07:15:17 GMT\r\n
    Access-Control-Expose-Headers: X-Reference-Error\r\n
```

3. During loading of video in hotstar :

Yes there are handshaking message sequences [TLS HandShake Protocol], which is explained in detail below. The sequence of messages exchanged by the application :
1. Connecting from source ip to host : staragvod3-vh.akamai.net:443 through http/1.1 protocol .
2. HTTP connection established (http.response.code = 200)
3 . TLSv1.2 - Client Hello : When a client first connects to a server, it is required to send the ClientHello as its first message. The client can also send a ClientHello in response to a HelloRequest or on its own initiative in order to renegotiate the security parameters in an existing connection.
4 . TLSV1.2 - Server Hello : Information that the server needs to communicate with the client using SSL. This includes the SSL version number, cipher settings, session-specific data.
5. Encrypted handshake message : 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. Application Data : Client and server can now exchange application data over the secured channel

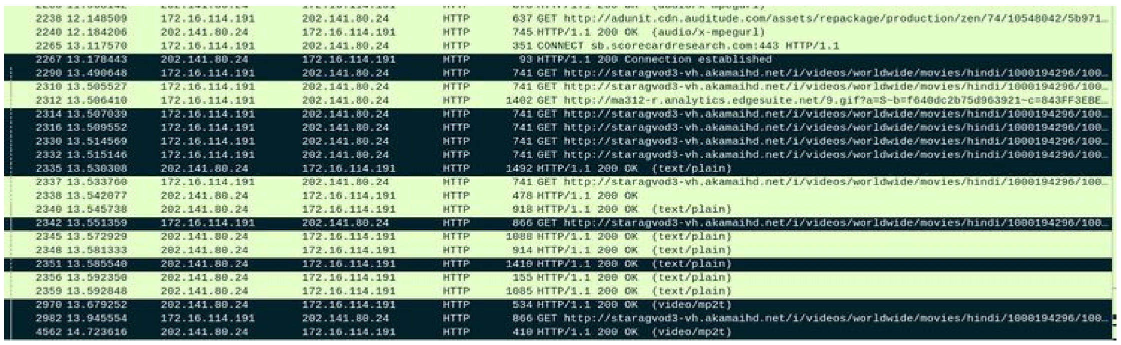
they have established. All messages sent from client to server and from server to client are encrypted using session key.



```
1325 5.210595   202.141.80.24    172.16.114.191   TCP       66 3128 → 60503 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1460 SACK_PERM=1 WS=128
1326 5.210714   172.16.114.191   202.141.80.24    TCP       54 60503 → 3128 [ACK] Seq=1 Ack=1 Win=525568 Len=0
1327 5.211729   172.16.114.191   202.141.80.24    HTTP     355 CONNECT staragvod3-vh.akamaihd.net:443 HTTP/1.1
1328 5.212053   202.141.80.24    172.16.114.191   TCP       60 3128 → 60503 [ACK] Seq=1 Ack=302 Win=15744 Len=0
1343 5.246379   202.141.80.24    172.16.114.191   HTTP      93 HTTP/1.1 200 Connection established
1344 5.247101   172.16.114.191   202.141.80.24    TLSv1.2  571 Client Hello
1348 5.281814   202.141.80.24    172.16.114.191   TLSv1.2  206 Server Hello, Change Cipher Spec, Encrypted Handshake Message
1349 5.282181   172.16.114.191   202.141.80.24    TLSv1.2  105 Change Cipher Spec, Encrypted Handshake Message
1350 5.285224   172.16.114.191   202.141.80.24    TLSv1.2  751 Application Data
1351 5.285552   202.141.80.24    172.16.114.191   TCP       60 3128 → 60503 [ACK] Seq=192 Ack=1567 Win=18176 Len=0
1355 5.321568   202.141.80.24    172.16.114.191   TCP     1502 3128 → 60503 [PSH, ACK] Seq=192 Ack=1567 Win=18176 Len=1448 [TCP segment of a reassembled…
1356 5.321913   202.141.80.24    172.16.114.191   TCP     1514 3128 → 60503 [ACK] Seq=1640 Ack=1567 Win=18176 Len=1460 [TCP segment of a reassembled PDU]
```

During playing of video :

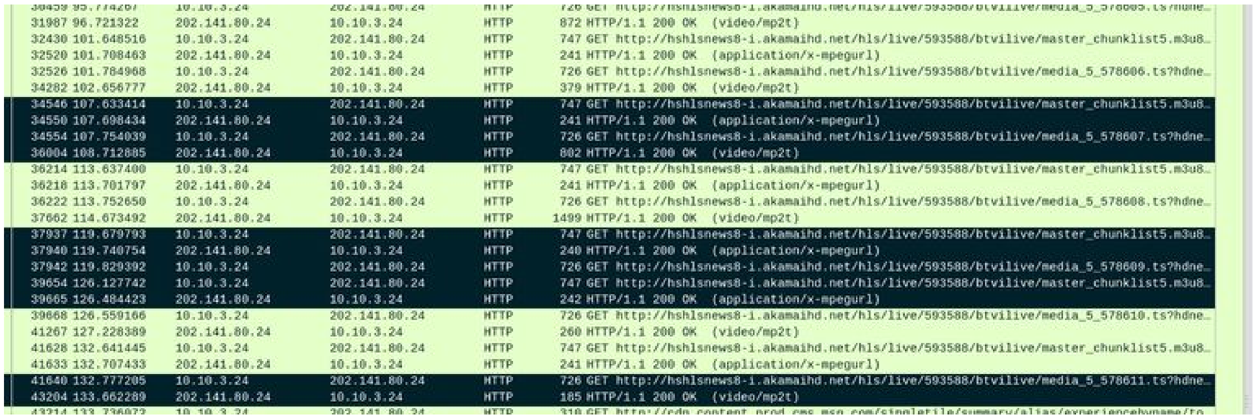
1. Before playing of videos, there are certain advertisements from host: adunit.cdn.auditude.com (file extension : m3u8) and the response from host is of the form audio/x-mpegurl (MIME type).
2. After connection is established by 202.141.80.24 with client ip (172.16.114.191) , the client ip issues multiple
Http GET requests to server AkamaiGHost , the request uri being a .ts file .
3. The response is of the form text [for subtitles if subtitle feature is kept on] or video/mp2t [the file type being typescript (.ts extension).



```
2238 12.148509   172.16.114.191   202.141.80.24    HTTP    637 GET http://adunit.cdn.auditude.com/assets/repackage/production/zen/74/10548042/5b971…
2240 12.184206   202.141.80.24    172.16.114.191   HTTP    745 HTTP/1.1 200 OK  (audio/x-mpegurl)
2265 13.117570   172.16.114.191   202.141.80.24    HTTP    351 CONNECT sb.scorecardresearch.com:443 HTTP/1.1
2267 13.178443   202.141.80.24    172.16.114.191   HTTP     93 HTTP/1.1 200 Connection established
2290 13.490648   172.16.114.191   202.141.80.24    HTTP    741 GET http://staragvod3-vh.akamaihd.net/i/videos/worldwide/movies/hindi/1000194296/100…
2310 13.505527   172.16.114.191   202.141.80.24    HTTP    741 GET http://staragvod3-vh.akamaihd.net/i/videos/worldwide/movies/hindi/1000194296/100…
2312 13.506410   172.16.114.191   202.141.80.24    HTTP   1402 GET http://ma312-r.analytics.edgesuite.net/9.gif?a=S~b=f640dc2b75d963921~c=843FF3EBE…
2314 13.507039   172.16.114.191   202.141.80.24    HTTP    741 GET http://staragvod3-vh.akamaihd.net/i/videos/worldwide/movies/hindi/1000194296/100…
2316 13.509552   172.16.114.191   202.141.80.24    HTTP    741 GET http://staragvod3-vh.akamaihd.net/i/videos/worldwide/movies/hindi/1000194296/100…
2330 13.514569   172.16.114.191   202.141.80.24    HTTP    741 GET http://staragvod3-vh.akamaihd.net/i/videos/worldwide/movies/hindi/1000194296/100…
2332 13.515146   172.16.114.191   202.141.80.24    HTTP    741 GET http://staragvod3-vh.akamaihd.net/i/videos/worldwide/movies/hindi/1000194296/100…
2335 13.530308   202.141.80.24    172.16.114.191   HTTP   1492 HTTP/1.1 200 OK  (text/plain)
2337 13.533760   172.16.114.191   202.141.80.24    HTTP    741 GET http://staragvod3-vh.akamaihd.net/i/videos/worldwide/movies/hindi/1000194296/100…
2338 13.542077   202.141.80.24    172.16.114.191   HTTP    478 HTTP/1.1 200 OK
2340 13.545738   202.141.80.24    172.16.114.191   HTTP    918 HTTP/1.1 200 OK  (text/plain)
2342 13.551359   172.16.114.191   202.141.80.24    HTTP    866 GET http://staragvod3-vh.akamaihd.net/i/videos/worldwide/movies/hindi/1000194296/100…
2345 13.572929   202.141.80.24    172.16.114.191   HTTP   1088 HTTP/1.1 200 OK  (text/plain)
2348 13.581333   202.141.80.24    172.16.114.191   HTTP    914 HTTP/1.1 200 OK  (text/plain)
2351 13.585540   202.141.80.24    172.16.114.191   HTTP   1410 HTTP/1.1 200 OK  (text/plain)
2356 13.592350   202.141.80.24    172.16.114.191   HTTP    155 HTTP/1.1 200 OK  (text/plain)
2359 13.592848   202.141.80.24    172.16.114.191   HTTP   1085 HTTP/1.1 200 OK  (text/plain)
2970 13.679252   202.141.80.24    172.16.114.191   HTTP    534 HTTP/1.1 200 OK  (video/mp2t)
2982 13.945554   172.16.114.191   202.141.80.24    HTTP    866 GET http://staragvod3-vh.akamaihd.net/i/videos/worldwide/movies/hindi/1000194296/100…
4562 14.723616   202.141.80.24    172.16.114.191   HTTP    410 HTTP/1.1 200 OK  (video/mp2t)
```

During live video stream :

1. After connection is established by 202.141.80.24 with client ip (172.16.114.191) , the client ip issues multiple Http GET requests, the request uris being .mu38 files.
2. The response is of the MIME Type application/x-mpegurl.
3. After a series of http GET requests with request uri being m3u8 files, and response status code being 200(OK), a http get request is made to server X-Akamai-Server: Akamai-SMT, the request uri being a .ts file.
4. After successful get request, a response is sent by host server to client as video/mp2t MIME Type (content_type of http response).



```
30459 95.774267   10.10.3.24     202.141.80.24  HTTP   726 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/media_5_578605.ts?hdne…
31987 96.721322   202.141.80.24  10.10.3.24     HTTP   872 HTTP/1.1 200 OK  (video/mp2t)
32430 101.648516  10.10.3.24     202.141.80.24  HTTP   747 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/master_chunklist5.m3u8…
32520 101.708463  202.141.80.24  10.10.3.24     HTTP   241 HTTP/1.1 200 OK  (application/x-mpegurl)
32526 101.784968  10.10.3.24     202.141.80.24  HTTP   726 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/media_5_578606.ts?hdne…
34282 102.656777  202.141.80.24  10.10.3.24     HTTP   379 HTTP/1.1 200 OK  (video/mp2t)
34546 107.633414  10.10.3.24     202.141.80.24  HTTP   747 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/master_chunklist5.m3u8…
34550 107.698434  202.141.80.24  10.10.3.24     HTTP   241 HTTP/1.1 200 OK  (application/x-mpegurl)
34554 107.754039  10.10.3.24     202.141.80.24  HTTP   726 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/media_5_578607.ts?hdne…
36004 108.712885  202.141.80.24  10.10.3.24     HTTP   802 HTTP/1.1 200 OK  (video/mp2t)
36214 113.637400  10.10.3.24     202.141.80.24  HTTP   747 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/master_chunklist5.m3u8…
36218 113.701797  202.141.80.24  10.10.3.24     HTTP   241 HTTP/1.1 200 OK  (application/x-mpegurl)
36222 113.752650  10.10.3.24     202.141.80.24  HTTP   726 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/media_5_578608.ts?hdne…
37662 114.673492  202.141.80.24  10.10.3.24     HTTP  1499 HTTP/1.1 200 OK  (video/mp2t)
37937 119.679793  10.10.3.24     202.141.80.24  HTTP   747 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/master_chunklist5.m3u8…
37940 119.740754  202.141.80.24  10.10.3.24     HTTP   240 HTTP/1.1 200 OK  (application/x-mpegurl)
37942 119.829392  10.10.3.24     202.141.80.24  HTTP   726 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/media_5_578609.ts?hdne…
39654 126.127742  10.10.3.24     202.141.80.24  HTTP   747 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/master_chunklist5.m3u8…
39665 126.484423  202.141.80.24  10.10.3.24     HTTP   242 HTTP/1.1 200 OK  (application/x-mpegurl)
39668 126.559166  10.10.3.24     202.141.80.24  HTTP   726 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/media_5_578610.ts?hdne…
41267 127.228389  202.141.80.24  10.10.3.24     HTTP   260 HTTP/1.1 200 OK  (video/mp2t)
41628 132.641445  10.10.3.24     202.141.80.24  HTTP   747 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/master_chunklist5.m3u8…
41633 132.707433  202.141.80.24  10.10.3.24     HTTP   241 HTTP/1.1 200 OK  (application/x-mpegurl)
41640 132.777205  10.10.3.24     202.141.80.24  HTTP   726 GET http://hshlsnews8-i.akamaihd.net/hls/live/593588/btvilive/media_5_578611.ts?hdne…
43204 133.662289  202.141.80.24  10.10.3.24     HTTP   185 HTTP/1.1 200 OK  (video/mp2t)
43214 133.736072  10.10.3.24     202.141.80.24  HTTP   310 GET http://cdn.content.prod.cms.msn.com/singletile/summary/alias/experiencebyname/to
```

The communication of the client and the host remains unchanged in the case of pausing video / live stream that is is similar to the behaviour shown during playing video.

The difference in normal media playback and live streaming may stem from the fact that in live streaming the content has to be continuously fetched and it is fetched in the form of playlists (.mu38 files) and then segmented files (.ts) are sent from it . But in case of media playback, it is different since the whole content is prefetched and so progressive download of video occurs in the form of .ts files through HTTP protocol.

4 . IPv4 is a connectionless protocol used in packet-switched layer networks, such as Ethernet. It provides the logical connection between network devices by providing identification for each device.

HTTP is by far the most commonly supported protocol used to transfer media on demand or live. So, hotstar uses http protocol for progressively downloading normal videos.
For live streaming, hotstar uses HTTP Live Streaming (HLS).HLS is supported on desktop browsers, smart TVs, and both Android and iOS mobile devices. HTML5 video players also natively support HDS, in comparison with HDS and RTMP. This allows a stream to reach as many viewers as possible, making HLS the safest protocol today for scaling a live stream to large audiences. As far as features, the HLS standard also supports adaptive bitrate streaming, dynamically delivering the best possible video quality at any moment. It was also recently updated to support the latest-and-greatest H.265 codec, which delivers twice the video quality at the same file size as H.264. Since HLS is such a robust protocol, Hotstar uses it.

Video streaming meets with TCP in their nature. First, video streaming adopts prefetching and buffering to achieve smooth play-out. TCP provides such (network) buffer, as well as the reliable transmission guarantee for no loss of frame (a frame could still miss the play-out deadline and gets discarded, however).
Second, TCP's bandwidth probing and congestion control will attempt to use all of the available bandwidth between the server and client, fetching content as quickly as possible while being friendly to other (TCP) traffic on the same links. With TCP, it's much harder to snoop a stream by tapping a copy because the client has to send the right acknowledgments for each segment received. Since it's connection oriented, Hotstar can do authentication and validation of clients better. Moreover, TCP provides error recovery by requesting retransmission of missing data. That is why Hotstar uses TCP protocol.

Hotstar uses TLSv1.2 containing message authentication, key material generation and the supported cipher suites . TLS is composed of two layers: the TLS Record Protocol and the TLS Handshake Protocol. The Record Protocol provides connection security, while the Handshake

Protocol allows the server and client to authenticate each other and to negotiate encryption algorithms and cryptographic keys before any data is exchanged, which has been explained above regarding how Hotstar uses it describing in detail the communication messages.

5.

Data taken from CSE Lab:

Throughput (bytes/s)	RTT (s)	Packet Size (bytes)	Packets lost	UDP packets	TCP packets	Number of responses
279k	0.004359	1216.5	419	0	43756	4.986
274k	0.004636	1256.5	171	0	181214	5.689
242k	0.005230	1269.5	320	0	47230	6.2576

Calculations are done as follows:

RTT = 1/avg(packets/s) throughput = avg bytes/s

Data taken from hostel :

Throughput (bytes/s)	RTT (s)	Packet Size (bytes)	Packets lost	UDP packets	TCP packets	Number of responses
263k	0.004468	1175.5	871	0	55675	4.518
317k	0.003602	1144.5	2789	0	89195	3.304
321k	0.003646	1171.5	955	0	41118	3.609

6.
The content is being sent from multiple sources.
Here are some of the IPs of the content providers :
49.44.121.218 (Approx. location - Maharashtra)
49.44.121.209 (Approx. location - Maharashtra)
49.44.121.232 (Approx. location - Maharashtra)
49.44.121.224 (Approx. location - Maharashtra)

Multiple sources must exist to provide content because if one source malfunctions or fails to perform due to various reasons, then hotstar will unable to provide media playback or live video stream. More multiple sources also exist for the purpose of load balancing.
Load balancing is dividing the amount of work that a computer has to do between two or more computers so that more work gets done in the same amount of time and, in general, all users get served faster.

Companies whose websites get a great deal of traffic usually use load balancing. Usually, if two servers are used to balance a work load, a third server is needed to determine which server to assign the work to. Since load balancing requires multiple servers, it is usually combined with failover and backup services. In some approaches, the servers are distributed over different geographic locations. That is why hotstar uses multiple sources to provide content since it experiences huge network traffic.

Traces zip file link (>720 mb):

https://drive.google.com/open?id=1AGcyEzb4L0Ruv3xnRDGclN0g-sSoBsbq