CS-349

NETWORKS LAB

ASSIGNMENT 2

APPLICATION : DAILYMOTION

ANUBHAV TYAGI

170101009

April 11, 2020

Link to Traces :

`https://drive.google.com/open?id=1AHVyHUKPO811NAEVX6FZeLKXfF0TE4I_`

Q1.
Application Layer Protocol : TLSv1.2

TLSv1.2 is the latest **SSL** protocol version. The TLS protocol is designed to provide three essential services to all applications running above it: **encryption, authentication, and data integrity**.The TLS Record header comprises three fields:

- **Record Type (1 byte)**: Possible Values for this field are: CHANGE_CIPHER_SPEC, ALERT, HANDSHAKE and APPLICATION_DATA.
- **Record Version (2 bytes)**: Specifies the version of the TLS protocol being used. Possible values are: SSL 3.0, TLS 1.0, TLS 1.1 and TLS 1.2
- **Record length (2 bytes)**: Specifies the length of the data in the record (excluding the header). Maximum supported is 16KB.

Transport Layer Protocol : TCP

TCP (Transmission Control Protocol) is a standard that defines how to establish and maintain a network conversation through which application programs can exchange data. The TCP packet format consists of the following fields:

- **Source Port** and **Destination Port** fields (16 bits each) identify the end points of the connection.
- **Sequence Number** field (32 bits) specifies the number assigned to the first byte of data in the current message.
- **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.
- **Data Offset** field (variable length) tells how many 32-bit words are contained in the TCP header.
- **Reserved** field (6 bits) must be zero.
- **Flags** field (6 bits) contains the various flags: URG, ACK, PSH, RST, SYN, FIN.
- **Window** field (16 bits) specifies the size of the sender's receive window.
- **Checksum** field (16 bits) indicates whether the header was damaged in transit.
- **Urgent pointer** field (16 bits) points to the first urgent data byte in the packet.
- **Options** field (variable length) specifies various TCP options.
- **Data** field (variable length) contains upper-layer information.

Network Layer Protocol : IPv4

IPv4 (Internet Protocol) is the core protocol that routes most of the internet traffic. The IPv4 header consistes of following fields:

- **Version**(4 bits): Provides the version number of Internet Protocol used.
- **IHL**(4 bits): Refers to Internet Header Length which is the length of an entire IP header.
- **Type of Service**: The Type of Service provides an indication of the abstract parameters of the quality of service desired.
- **Total Length**(16 bits): Length of entire IP packet, which includes IP header and encapsulated data.
- **Identification**(16 bits): This field is used to uniquely identify a group of fragments in the single IP packet.

- **Flags**(3 bits): This is a three-bit field that's used to identify and control fragments.
- **Fragment Offset**(13 bits): This offset provides the location of the fragment in the original IP Packet.
- **Time To Live**(8 bits): Maximum time the datagram is allowed to remain in the internet system.
- **Protocol**(8 bits): This field provides the protocol that's used in the data part of the packet.
- **Header Checksum**(16 bits): This field is used for error-checking of the entire header.
- **Source Address**(32 bits): This field is the 32-bit address of the sender of the packet.
- **Destination Address**(32 bits): This field is the 32-bit address of the reciever of the packet.
- **Options**(32 bits): This is optional field, which is used if the value of IHL is greater than 5.

Link Layer Protocol : Ethernet II

Ethernet is the most common local area networking technology. An Ethernet frame must be at least **64 bytes** for collision detection to work, and can be a maximum of **1,518 bytes**.

- **Src** (6 bytes): Hardware address of the source network adapter.
- **Dst** (6 bytes): Hardware address of the destination network adapter

```
▶ Frame 8324: 1196 bytes on wire (9568 bits), 1196 bytes captured (9568 bits) on interface 0
▶ Ethernet II, Src: LcfcHefe_36:5c:dd (54:e1:ad:36:5c:dd), Dst: Cisco_74:60:43 (ec:44:76:74:60:43)
▶ Internet Protocol Version 4, Src: 10.19.6.117, Dst: 103.195.32.2
▶ Transmission Control Protocol, Src Port: 53109, Dst Port: 443, Seq: 344, Ack: 3158, Len: 1142
▼ Secure Sockets Layer
  ▶ TLSv1.2 Record Layer: Application Data Protocol: http-over-tls
```

Figure 1: Various Protocols

Q2.

TLSV1.2 :

Field	Value	Explanation
Content Type	Application Data	Type of the content, the protocol is carrying
Version	TLS 1.2 (0x0303)	Version of the Protocol
Length	891	Length of the data
Encrypted Application Data	0000000000000001...	Actual application data in encypted form.

```
▼ TLSv1.2 Record Layer: Application Data Protocol: http-over-tls
    Content Type: Application Data (23)
    Version: TLS 1.2 (0x0303)
    Length: 891
    Encrypted Application Data: 0000000000000000122423e3c38283a2cc18ea13e5871aaa9...
```

Figure 2: Sample TLSv1.2 Record

TCP :

Field	Value	Explanation
Source Port	53109	Port of the source side end point of the packet transfer.
Dst Port	443	Port of the destination side end point of the packet transfer.
Sequence Number	0	This shows the relative sequence number of the packet
Acknowledgement Number	0	This shows the acknowledgement number for the packet recieved
Header Length	32 bytes (8)	This shows the total length of the header attached to the packet
Flags	0x002 (SYN)	SYN flag is set to synchronize sequence numbers to initiate a connection

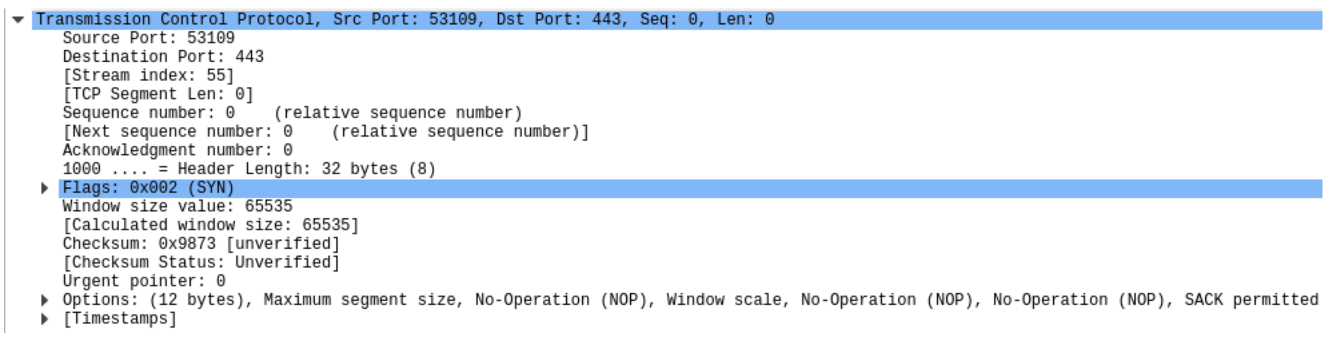

Figure 3: TCP Header Values

IPv4 :

Field	Value	Explanation
Src	10.19.6.117	IP address of the source
Dst	103.195.32.110	IP address of the destination
Version	4	Version of the IP protocol being used
Header Length	20 bytes (5)	This shows the total length of the header attached to the packet
Flags	0x4000, Don't Fragment	If the 'DF' bit is set on packets, a router which normally would fragment a packet larger than MTU, instead will drop the packet. The router is expected to send "ICMP Fragmentation Needed" packet, allowing the sending host to account for the lower MTU on the path to the destination host. The sending side will then reduce its estimate of the connection's Path MTU and re-send in smaller segments.This process is called PMTU-D
Time To Live	128	Specifies the maximum number of layer three hops (typically routers) that can be traversed on the path to their destination

```
▼ Internet Protocol Version 4, Src: 10.19.6.117, Dst: 103.195.32.110
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 91
    Identification: 0x0bd4 (3028)
  ▶ Flags: 0x4000, Don't fragment
    Time to live: 128
    Protocol: TCP (6)
    Header checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.19.6.117
    Destination: 103.195.32.110
```

Figure 4: IPv4 Header Values

Q3.

TLSv1.2 : It is used to authenticate and encrypt data exchanged between client and server. It uses Handshake protocols and Change Cipher Spec Messages to establish a secure connection. It helps by preventing intruder tampering with communication between client and server.Any unprotected HTTP request can potentially risk the identity of user like login credentials.

TCP :TCP (Transmission Control Protocol) is a standard that defines how to establish and maintain a network conversation through which application programs can exchange data. TCP is responsible for synchronization and initiating connection. TCP is used because it provides reliable data transfer which is essential while uploading and downloading videos.

Q4.

Handshake Protocols :

To establish a connection, **TCP** uses a **3-way handshake protocol**.

1. Client sends a TCP Packet with **SYN** flag set, which informs server that client is likely to start communication and with what sequence number it starts segments with
2. Server responds to the client by sending a TCP Packet with **SYN-ACK** signal set. ACK signifies the response of segment it received and SYN signifies with what sequence number it is likely to start the segments with.
3. Client acknowledges the response of server and they both establish a reliable connection with which they will start the actual data transfer

123 4.748591	10.19.6.117	195.8.215.136	TCP	66 53076 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
124 4.748798	195.8.215.136	10.19.6.117	TCP	66 443 → 53075 [SYN, ACK] Seq=0 Ack=1 Win=18352 Len=0 MSS=9176 SACK_PERM=1 WS=128
125 4.748865	10.19.6.117	195.8.215.136	TCP	54 53075 → 443 [ACK] Seq=1 Ack=1 Win=262144 Len=0

Figure 5: TCP Handshake Protocol

After the TCP Handshake, **TLS** connection is initiated using a sequence known as the **TLS handshake** :

1. TLS client sends a **client hello** message that lists cryptographic information
2. TLS server responds with a **server hello** message that contains the CipherSuite chosen by the server from the list provided by the client, the session ID, and another random byte string. The server also sends its digital certificate.
3. TLS client verifies the server's digital certificate.
4. TLS client sends the random byte string that enables both the client and the server to compute the secret key to be used for encrypting subsequent message data.

5. The client sends a random byte string encrypted with the client's private key, together with the client's digital certificate
6. TLS server verifies the client's certificate.
7. TLS client sends the server a finished message, which is encrypted with the secret key, indicating that the client part of the handshake is complete.
8. TLS server sends the client a finished message, which is encrypted with the secret key, indicating that the server part of the handshake is complete.
9. The server and client can now exchange messages that are symmetrically encrypted with the shared secret key.

145	4.754546	10.19.6.117	103.195.32.11	TLSv1.2	271	Client Hello
150	4.755161	103.195.32.11	10.19.6.117	TCP	60	443 → 53079 [ACK] Seq=1 Ack=218 Win=19456 Len=0
284	5.124854	103.195.32.11	10.19.6.117	TLSv1.2	1514	Server Hello
285	5.124854	103.195.32.11	10.19.6.117	TLSv1.2	1490	Certificate [TCP segment of a reassembled PDU]
286	5.124855	103.195.32.11	10.19.6.117	TLSv1.2	315	Server Key Exchange, Server Hello Done
287	5.124896	10.19.6.117	103.195.32.11	TCP	54	53079 → 443 [ACK] Seq=218 Ack=3158 Win=262144 Len=0
295	5.131607	10.19.6.117	103.195.32.11	TLSv1.2	180	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
296	5.131920	103.195.32.11	10.19.6.117	TCP	60	443 → 53079 [ACK] Seq=3158 Ack=344 Win=19456 Len=0
337	5.216676	103.195.32.11	10.19.6.117	TLSv1.2	240	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message

Figure 6: TLSv1.2 Handshake Protocol

Features :

Video Streaming :

While **streaming videos**, new data packets encrypted using **TSLv1.2** are sent by server and acknowledged by client time to time using TCP Protocol. **Play** and **Pause** happens at the client end so there is no specific behaviour observed in the connection.

Video Uploading :

New TCP connection is established between client and server. Data packets, encrypted using **TLSv1.2**, are sent from client to server (dailymotion server) and server sends acknowledgement time to time for each packet recieved. On upload completion, the TCP connection is broken. The client(personal PC) sends packet with FIN signal set. The server(dailymotion server) replies with FIN-ACK signal set. Finally the client acknowledges the recieved packet and TCP connection is closed.

Video Downloading :

New TCP connection is established between client and server. Data packets, encrypted using **TLSv1.2**, are sent from server (dailymotion server) to client and client sends acknowledgement time to time for each packet recieved. On download completion, the TCP connection is broken. The server (dailymotion server) sends packet with FIN signal set. The client(personal PC) replies with FIN-ACK signal set. Finally the server acknowledges the recieved packet and TCP connection is closed.

Q5.

Parameter	**1100 hours**	**1600 hours**	**2200 hours**
Throughput (kB/s)	1967	620	1340
RTT (ms)	0.293	0.244	0.163
Packet Size (B)	984	970	924
No. Of Packets Lost	1673 (2%)	0 (0%)	0 (0%)
No. Of TCP Packets	106534	25890	170914
No. Of UDP Packets	681	238	2173
Number of responses received with respect to one request sent	2.01722	2.08939	1.69596

Q6.

The whole content is being sent from multiple IPs. Some of them are listed below:
195.8.215.136, 195.8.215.224 : www.dailymotion.com
103.195.32.110, 103.195.32.33, 103.195.32.14 : st.dc3.dailymotion.com
198.54.201.91, 198.54.200.91 : st.nyc.dailymotion.com
and few more IPs.

Reasons :

1. **Load Balancing :** When using multiple servers, it is possible to balance the load of an influx of traffic across those servers. A spike in visitors to a website can quickly take a single server up to its capacity and cause the site to become very slow or even unavailable. By spreading the resources visitors use over two or more servers, the performance of the website can be more easily maintained.
2. **Geographic Location :** The ideal scenario is for a server to be as close as possible to the customer or end user.
3. **Maintenance Backup :** Updates to software and other tweaks are inevitable. But when you have more than one server, you can simply update one at a time and direct all traffic to the other server while the maintenance is being performed.
4. **Disaster Recovery :** A power cut to a data centre, technical issues in the data centre and human error are all things that can take server, and therefore website, offline. The quickest way to return to business as usual is if you already have a version of the same website hosted at a different location and can direct all traffic to that version.

```
103.53.14.131   dmotion.hs.llnwd.net
68.180.130.15   yf2.yahoo.com
91.209.196.50   ns2.as48447.net
198.51.44.5     dns1.p05.nsone.net
106.11.141.112  ns2.alidns.com
10.19.6.87      DESKTOP-CHU4DVU.local
69.28.143.13    dns13.llnwd.net
195.8.214.2     b.dailymotion.com
34.198.46.133   io-cookie-sync-1725936127.us-east-1.elb.amazonaws.com
216.239.34.10   ns2.google.com
216.239.34.21   x.mdhv.io
205.251.192.235 ns-235.awsdns-29.com
103.53.14.4     dmotion.hs.llnwd.net
23.61.199.131   a7-131.akadns.net
91.199.212.50   ns1.as48447.net
3.0.26.188      juretho.com
104.16.190.66   dmx.districtm.io
103.43.90.180   ib.sin1.geoadnxs.com
198.51.45.69    dns4.p05.nsone.net
8.43.72.97      pixel-us-east.rubiconproject.net.akadns.net
198.54.200.91   st.sv4.dailymotion.com
34.196.86.37    io-cookie-sync-1725936127.us-east-1.elb.amazonaws.com
203.198.20.166  n7e7.akamaiedge.net
205.251.193.101 ns-357.awsdns-44.com
69.173.144.138  pixel-eu.rubiconproject.net.akadns.net
216.239.34.106  ns-cloud-a2.googledomains.com
205.251.192.210 ns-210.awsdns-26.com
140.205.81.21   ns1.alidns.com
47.91.168.21    dailymotion-cs.vpadn.com
40.90.4.6       ns1-06.azure-dns.com
34.102.179.36   public-prod-dailymotion-addirector.dmxleo.com
95.100.170.111  n1e7.akamaiedge.net
184.85.248.128  a9-128.akadns.net
18.195.155.181  cs.emxdgt.com
205.251.195.136 ns-904.awsdns-49.net
60.254.173.205  n4e7.akamaiedge.net
216.239.32.10   ns1.google.com
216.239.32.21   x.mdhv.io
103.43.90.20    ib.sin1.geoadnxs.com
193.108.88.128  a1-128.akadns.net
103.43.90.53    ib.sin1.geoadnxs.com
96.7.49.129     a3-129.akadns.net
188.65.124.58   pebed.dm.gg
188.65.124.91   st.dc3.dailymotion.com
13.251.6.185    juretho.com
140.205.41.21   ns1.alidns.com
3.0.25.35       juretho.com
95.100.170.108  n2e7.akamaiedge.net
106.11.211.61   ns1.alidns.com
103.195.32.14   proxy-14.sg1.dailymotion.com
18.136.170.149  juretho.com
103.195.32.91   st.sg1.dailymotion.com
```

Figure 7: Resolved IP address list