

Cryptography and Network Security Lab

Assignment 16: SSL/TLS Handshake Analysis using Wireshark

PRN: 2019BTECS00035

Date: 30/11/2022

SSL/TLS Lab

1 Objective

To observe SSL/TLS (Secure Sockets Layer/ Transport Layer Security) in action. SSL/TLS is used to secure TCP connections, and it is widely used as part of the secure web: HTTPS is SSL over HTTP

2 STEP 1: Open a Trace you should use a supplied trace file trace-ssl.pcap.

File → Open → open from folder containing file

3 STEP 2: Inspect the Trace

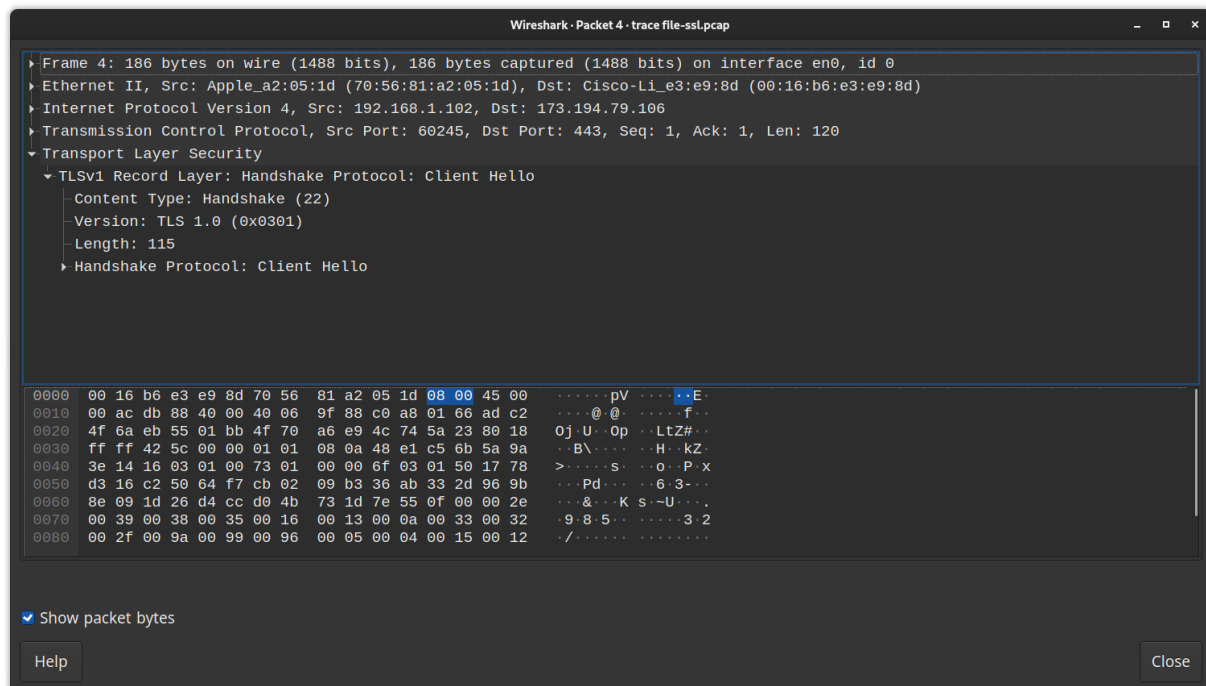
Now we are ready to look at the details of some SSL messages. To begin, enter and apply a display filter of ssl. This filter will help to simplify the display by showing only SSL and TLS messages. It will exclude other TCP segments that are part of the trace, such as Acks and connection open/close. Select a TLS message somewhere in the middle of your trace for which the Info field reads Application Data, and expand its Secure Sockets Layer block (by using triangular icon on left side). Application Data is a generic TLS message carrying contents for the application, such as the web page. It is a good place for us to start looking at TLS messages. Look for the following protocol blocks and fields in the message

- The lower layer protocol blocks are TCP and IP because SSL runs on top of TCP/IP.
- The SSL layer contains a TLS Record Layer. This is the foundational sublayer for TLS. All messages contain records. Expand this block to see its details.

- Each record starts with a Content Type field. This tells us what is in the contents of the record. Then comes a Version identifier. It will be a constant value for the SSL connection.
- It is followed by a Length field giving the length of the record.
- Last comes the contents of the record. Application Data records are sent after SSL has secured the connection, so the contents will show up as encrypted data.

Note that, unlike other protocols we will see such as DNS, there may be multiple records in a single message. Each record will show up as its own block. Look at the Info column, and you will see messages with more than one block.

Answer the following questions to show your understanding of SSL records:



1. What is the Content Type for a record containing Application Data?:

Content Type: Handshake(22)

2. What version constant is used in your trace, and which version of TLS does it represent?

Version : 1.0 (0x0301)

4 Step 3: The SSL Handshake

An important part of SSL is the initial handshake that establishes a secure connection. The handshake proceeds in several phases. There are slight differences for different versions of TLS and depending on the encryption scheme that is in use. The usual outline for a brand new connection is:

- Client (the browser) and Server(the web server) both send their Hellos
- Server sends its certificate to Client to authenticate (and optionally asks for Client Certificate)
- Client sends keying information and signals a switch to encrypted data.
- Server signals a switch to encrypted data.
- Both Client and Server send encrypted data.
- An Alert is used to tell the other party that the connection is closing. Note that there is also a mechanism to resume sessions for repeat connections between the same client and server to skip most of steps b and c.

4.1 Hello Message

Find and inspect the details of the Client Hello and Server Hello messages, including expanding the Hand- shake protocol block within the TLS Record. For these initial messages, an encryption scheme is not yet established so the contents of the record are visible to us. They contain details of the secure connection setup in a Handshake protocol format.

Answer the following questions.

1. How long in bytes is the random data in the Hellos? Both the Client and Server include this random data (a nonce) to allow the establishment of session keys.

→ **Length of random Bytes: 28**

```

Random: 501778d316c25064f7cb0209b336ab332d969b8e091d26d4ccd04b731d7e550f
  GMT Unix Time: Jul 31, 2012 11:48:59.000000000 IST
  Random Bytes: 16c25064f7cb0209b336ab332d969b8e091d26d4ccd04b731d7e550f
  Session ID Length: 0
  Cipher Suites Length: 46
  Cipher Suites (23 suites)
    Cipher Suite: TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x0039)
    Cipher Suite: TLS_DHE_DSS_WITH_AES_256_CBC_SHA (0x0038)
    Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
0000 00 16 b6 e3 e9 8d 70 56 81 a2 05 1d 08 00 45 00  . . . . . pV . . . . . E .
0010 00 ac db 88 40 00 40 06 9f 88 c0 a8 01 66 ad c2  . . . . . @ . @ . . . . . f .
0020 4f 6a eb 55 01 bb 4f 70 a6 e9 4c 74 5a 23 80 18  Oj . U . . Op . . LtZ# .
0030 ff ff 42 5c 00 00 01 01 08 0a 48 e1 c5 6b 5a 9a  . . B \ . . . . . H . . kZ .
0040 3e 14 16 03 01 00 73 01 00 00 6f 03 01 50 17 78  > . . . . . s . . o . . P . x
0050 d3 16 c2 50 64 f7 cb 02 09 b3 36 ab 33 2d 96 9b  . . Pd . . . . . 6 . 3 . .
0060 8e 09 1d 26 d4 cc d0 4b 73 1d 7e 55 0f 00 00 2e  . . & . . . K s . ~ U . .
0070 00 39 00 38 00 35 00 16 00 13 00 0a 00 33 00 32  . 9 . 8 . 5 . . . . . 3 . 2
0080 00 2f 00 9a 00 99 00 96 00 05 00 04 00 15 00 12  . / . . . . . . . . . . .
0090 00 09 00 14 00 11 00 08 00 06 00 03 00 ff 02 01  . . . . . . . . . . .
00a0 00 00 17 00 00 00 13 00 11 00 00 0e 77 77 77 2e  . . . . . . . . . www .
00b0 67 6f 6f 67 6c 65 2e 63 6f 6d  . . . . . google . c om

```

Bytes 81-108: Random Bytes (tls.handshake.random_bytes)

2. How long in bytes is the session identifier sent by the server? This identifier allows later resumption of the session with an abbreviated handshake when both the client and server indicate the same value. In our case, the client likely sent no session ID as there was nothing to resume.

→ **Session Identifier Length: 32 (Bytes 110 - 141)**

```

  GMT Unix Time: Jul 31, 2012 11:48:59.000000000 IST
  Random Bytes: d52d556ed20e072f638f0a51e9724d66ef5f13769d3a52e00161a893
  Session ID Length: 32
  Session ID: 8530bdac95116ccb343798b36cb2fd79c1e278cba1af41456c810c0cebfcccf4
  Cipher Suite: TLS_RSA_WITH_RC4_128_SHA (0x0005)
  Compression Method: null (0)
  Extensions Length: 9
  Extension: server_name (len=0)
  Extension: renegotiation_info (len=1)
  [JA3S Fullstring: 769,5,0-65281]
  [JA3S: d2e6f7ef558ea8036c7e21b163b2d1af]

```

3. What Cipher suite is chosen by the Server? Give its name and value. The Client will list the different cipher methods it supports, and the Server will pick one of these methods to use.

→ **Cipher Suite used by the Server is:**

Cipher Suite: TLS_RSA_WITH_RC4_128_SHA (0x0005)

```
Cipher Suite: TLS_RSA_WITH_RC4_128_SHA (0x0005)
Compression Method: null (0)
Extensions Length: 9
  ▶ Extension: server_name (len=0)
  ▼ Extension: renegotiation_info (len=1)
    Type: renegotiation_info (65281)
    Length: 1
    ▶ Renegotiation Info extension
    [JA3S Fullstring: 769,5,0-65281]
    [JA3S: d2e6f7ef558ea8036c7e21b163b2d1af]
```

4.2 Certificate Messages

Next, find and inspect the details of the Certificate message, including expanding the Handshake proto-col block within the TLS Record. As with the Hellos, the contents of the Certificate message are visible because an encryption scheme is not yet established. It should come after the Hello messages.

Answer the following questions:

1. Who sends the Certificate, the client, the server, or both? A certificate is sent by one party to let the other party authenticate that it is who it claims to be. Based on this usage, you should be able to guess who sends the certificate and check the messages in your trace.

→

In this packet trace only server is sending its certificate. But there might be the case that the server could ask the client to provide its own certificate for the identity of the client.

Source	Destination	Protocol	Length	Info
92.168.1.102	173.194.79.106	TLSv1	186	Client Hello
73.194.79.106	192.168.1.102	TLSv1	1484	Server Hello
73.194.79.106	192.168.1.102	TLSv1	377	Certificate, Server Hello Done
92.168.1.102	173.194.79.106	TLSv1	252	Client Key Exchange, Change Cipher Spec
73.194.79.106	192.168.1.102	TLSv1	113	Change Cipher Spec, Encrypted Handshake
92.168.1.102	173.194.79.106	TLSv1	239	Application Data
73.194.79.106	192.168.1.102	TLSv1	1416	Application Data
73.194.79.106	192.168.1.102	TLSv1	1416	Application Data
73.194.79.106	192.168.1.102	TLSv1	1416	Application Data, Application Data,
73.194.79.106	192.168.1.102	TLSv1	316	Application Data, Application Data

Wireshark · Packet 7 · trace file-ssl.pcap

Internet Protocol Version 4, Src: 173.194.79.106, Dst: 192.168.1.102

Transmission Control Protocol, Src Port: 443, Dst Port: 60245, Seq: 1419, Ack: 121, Len: 311

[2 Reassembled TCP Segments (1630 bytes): #6(1328), #7(302)]

Transport Layer Security

TLStv1 Record Layer: Handshake Protocol: Certificate

Content Type: Handshake (22)

Version: TLS 1.0 (0x0301)

Length: 1625

Handshake Protocol: Certificate

Handshake Type: Certificate (11)

Length: 1621

Certificates Length: 1618

Certificates (1618 bytes)

Certificate Length: 805

Certificate: 308203213082028aa00302010202104f9d96d966b0992b54c2957cb4157d4d300d06092a... (id-at-commonName=www.google...)

signedCertificate

algorithmIdentifier (sha1WithRSAEncryption)

Padding: 0

encrypted: 21acd5aeca34895ac2ab52d2b234669d7aabeee67cd57ec25c28bb7400c9101f4213fc69...

Certificate Length: 807

Certificate: 308203233082028ca003020102020430000002300d06092a864886f70d0101050500305f... (id-at-commonName=Thawte SGC ...)

Transport Layer Security

TLStv1 Record Layer: Handshake Protocol: Server Hello Done

Content Type: Handshake (22)

Version: TLS 1.0 (0x0301)

Frame (377 bytes) Reassembled TCP (1630 bytes)

Show packet bytes

Help Close

A Certificate message will contain one or more certificates, as needed for one party to verify the identity of the other party from its roots of trust certificates. You can inspect those certificates in your browser.

4.3 Client Key Exchange and Change Cipher Messages

Find and inspect the details of the Client Key Exchange and Change Cipher messages, expanding their various details. The key exchange message is sent to pass keying information so that both sides will have the same secret session key. The change cipher message signal a switch to a new encryption scheme to the other party. This means that it is the last unencrypted message sent by the party.

Answer the following questions:

1. Who sends the Change Cipher Spec message, the client, the server, or both?

→

The change cipher spec message is sent by both client and server. Client sent it first.

6 0.041694	173.194.79.106	192.168.1.102	TLSv1	108 Server Hello
7 0.041697	173.194.79.106	192.168.1.102	TLSv1	377 Certificate, Server Hello Done
9 0.088543	192.168.1.102	173.194.79.106	TLSv1	252 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
10 0.105145	173.194.79.106	192.168.1.102	TLSv1	113 Change Cipher Spec, Encrypted Handshake Message
12 0.105436	192.168.1.102	173.194.79.106	TLSv1	239 Application Data
13 0.136468	173.194.79.106	192.168.1.102	TLSv1	1416 Application Data
15 0.137903	173.194.79.106	192.168.1.102	TLSv1	1416 Application Data
17 0.138469	173.194.79.106	192.168.1.102	TLSv1	1416 Application Data, Application Data, Application Data

2. What are the contents carried inside the Change Cipher Spec message? Look past the Content Type and other headers to see the message itself.

▼ Transport Layer Security
▶ TLSv1 Record Layer: Handshake Protocol: Client Key Exchange
▼ TLSv1 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
Content Type: Change Cipher Spec (20)
Version: TLS 1.0 (0x0301)
Length: 1
Change Cipher Spec Message
▼ TLSv1 Record Layer: Handshake Protocol: Encrypted Handshake Message
Content Type: Handshake (22)
Version: TLS 1.0 (0x0301)
Length: 36
Handshake Protocol: Encrypted Handshake Message

→ The Change Cipher Spec Message contains following Fields:

- Content Type

- **Version**
- **Length**
- **change cipher spec message**