2020

SOFTENG 364: Lab 8 SSL/TLS

Craig Sutherland

Contents

Overview

By the end of this lab you should be able to:

1. Explain how SSL establishes a connection to a host.
2. List the parts of an ICMP message.

To receive credit for completing the worksheet, please complete the Lab 8 Quiz on Canvas. You'll receive feedback immediately afterwards, and may choose to redo the quiz if you wish. This worksheet and the associated quiz comprise Lab 8 and contribute 1% to the final mark. The due date (for the quiz) is 11:59 pm, Friday 12th June.

Several activities on the lab worksheet are framed as "questions", but responses needn't be submitted (i.e. the on-line quiz is the only submission required). Nonetheless, please don't hesitate to speak to a member of the 364 team during the lab if you are unsure of what a suitable response might be.

Preparation

The software we need this week is installed in the Engineering computer labs. If you're working on your own PC, you will need to download and install Wireshark[1] (https://www.wireshark.org/download.html). The remaining tool should already be available on your PC as part of the operating system. We will be using `curl`, which is a command-line utility that can be accessed via a command/terminal window.

Capturing a Trace

The first part of this lab is to capture a trace that we can analyse. Normally, we would use a browser to generate SSL traffic, but web browsers have complex behaviours for sending and receiving requests. Using a browser would give us a complex trace that would be difficult to analyse. Instead, we will use `curl` to fetch a single HTTPS resource.

Note: close all browser tabs and windows before running the trace. These will generate additional network traffic and we want to minimize the amount of activity captured. Even with your browser(s) closed, you may find your machine is still generating additional traffic. If you find too much unexpected traffic, you may need to redo your trace to simplify the trace log.

1. Launch WireShark and start a capture with a filter of "`tcp port 443`". We need to specific the filter as there is no shorthand for SSL. However, as specific by RFC 7230, port 443 is the default port used for SSL[2] traffic.
2. Open a command window or terminal[3] and request the main google.com page using the following command:

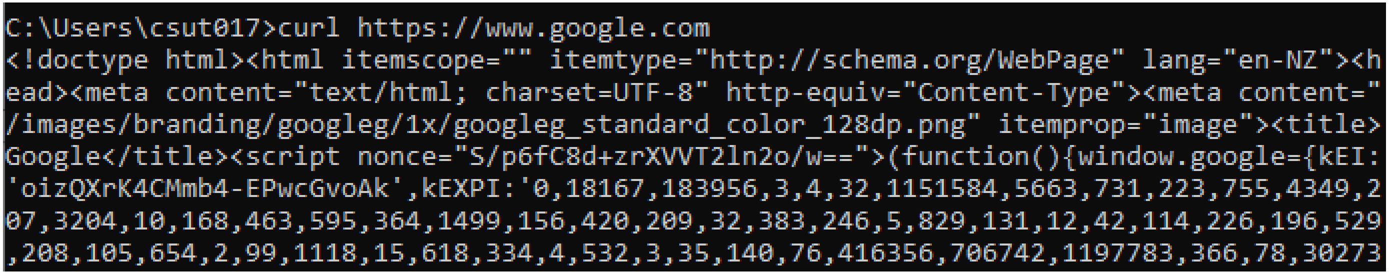
```
curl https://google.com
```

[1] This Lab manual is using version 3.2.3 – you can use other versions but the screen shots may be slightly different.

[2] Technically, we are actually capturing TLS traffic rather than SSL. TLS is the open standard version of SSL and comes after SSL 3.0. As SSL is no longer recommended, most sites are using TLS. It is common practise to refer to TLS and SSL as simply SSL.

[3] Command window for Windows OSes, a terminal for Mac or Linux-based OSes.

This command should display output similar to the following:



```
C:\Users\csut017>curl https://www.google.com
<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="en-NZ"><h
ead><meta content="text/html; charset=UTF-8" http-equiv="Content-Type"><meta content="
/images/branding/googleg/1x/googleg_standard_color_128dp.png" itemprop="image"><title>
Google</title><script nonce="S/p6fC8d+zrXVVT2ln2o/w==">(function(){window.google={kEI:
'oizQXrK4CMmb4-EPwcGvoAk',kEXPI:'0,18167,183956,3,4,32,1151584,5663,731,223,755,4349,2
07,3204,10,168,463,595,364,1499,156,420,209,32,383,246,5,829,131,12,42,114,226,196,529
,208,105,654,2,99,1118,15,618,334,4,532,3,35,140,76,416356,706742,1197783,366,78,30273
```

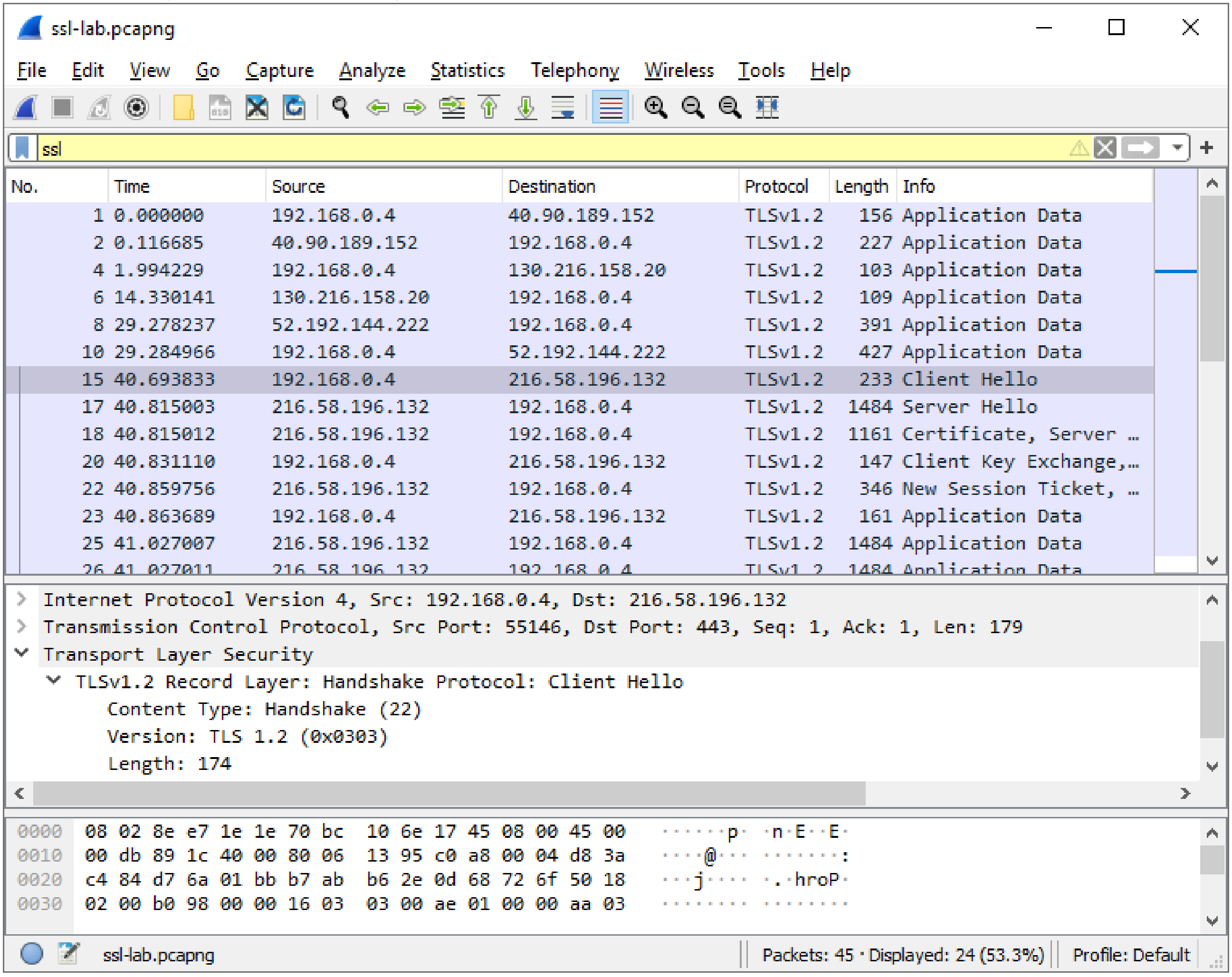
You may recognise this output as the content of an HTML page. For this lab, the actual content is not important, just that you have received an SSL response.

3. Stop the capture once the HTTPS request is complete and save your trace log.

Inspecting the Trace

Now that you have a trace, let's break it open and see what is inside.

1. Apply a display filter of "`ssl`". This filter will simplify your analysis by removing non SSL messages, such as Acks and connection open/close messages. After applying the filter you should have a display like the following:



ssl-lab.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ssl

```
No.  Time         Source          Destination     Protocol Length Info
   1 0.000000     192.168.0.4     40.90.189.152   TLSv1.2   156 Application Data
   2 0.116685     40.90.189.152   192.168.0.4     TLSv1.2   227 Application Data
   4 1.994229     192.168.0.4     130.216.158.20  TLSv1.2   103 Application Data
   6 14.330141    130.216.158.20  192.168.0.4     TLSv1.2   109 Application Data
   8 29.278237    52.192.144.222  192.168.0.4     TLSv1.2   391 Application Data
  10 29.284966    192.168.0.4     52.192.144.222  TLSv1.2   427 Application Data
  15 40.693833    192.168.0.4     216.58.196.132  TLSv1.2   233 Client Hello
  17 40.815003    216.58.196.132  192.168.0.4     TLSv1.2  1484 Server Hello
  18 40.815012    216.58.196.132  192.168.0.4     TLSv1.2  1161 Certificate, Server …
  20 40.831110    192.168.0.4     216.58.196.132  TLSv1.2   147 Client Key Exchange,…
  22 40.859756    216.58.196.132  192.168.0.4     TLSv1.2   346 New Session Ticket, …
  23 40.863689    192.168.0.4     216.58.196.132  TLSv1.2   161 Application Data
  25 41.027007    216.58.196.132  192.168.0.4     TLSv1.2  1484 Application Data
  26 41.027011    216.58.196.132  192.168.0.4     TLSv1.2  1484 Application Data
```

```
> Internet Protocol Version 4, Src: 192.168.0.4, Dst: 216.58.196.132
> Transmission Control Protocol, Src Port: 55146, Dst Port: 443, Seq: 1, Ack: 1, Len: 179
v Transport Layer Security
   v TLSv1.2 Record Layer: Handshake Protocol: Client Hello
        Content Type: Handshake (22)
        Version: TLS 1.2 (0x0303)
        Length: 174
```

```
0000  08 02 8e e7 1e 1e 70 bc  10 6e 17 45 08 00 45 00   ······p· ·n·E··E·
0010  00 db 89 1c 40 00 80 06  13 95 c0 a8 00 04 d8 3a   ····@··· ·······:
0020  c4 84 d7 6a 01 bb b7 ab  b6 2e 0d 68 72 6f 50 18   ···j···· ·.·hroP·
0030  02 00 b0 98 00 00 16 03  03 00 ae 01 00 00 aa 03   ········ ········
```

ssl-lab.pcapng | Packets: 45 · Displayed: 24 (53.3%) | Profile: Default

Note: my trace has captured some additional SSL traffic. The first few lines are additional SSL messages that my machine is either sending or receiving. You may find you have similar additional traffic. These can be ignored as long.

2. Select an SSL message somewhere in the middle that has "`Application Data`" in the Info column. Once selected, expand the Transport Layer Security block by clicking on the ">" if it is not already expanded.

3. 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.
    - 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[4].

Questions

Record the answers for these questions. You will use these results to answer the questions on the Canvas quiz.

What is the Content-Type for a record containing "`Application Data`"?

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

Does the Length cover the Record Layer header as well as payload, or only the payload?

The SSL Handshake

An important part of SSL is the initial handshake to establish a secure connection. The handshake proceeds in several phases. While there are slight differences for different versions of TLS, the usual outline for a new connection is:

a) Client (the browser) and Server (the web server) both send their Hellos
b) Server sends its certificate to Client to authenticate (and optionally asks for Client Certificate)
c) Client sends keying information and signals a switch to encrypted data.
d) Server signals a switch to encrypted data.
e) Both Client and Server send encrypted data.
f) An Alert is used to tell the other party that the connection is closing.

Note: 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. We will not cover session resumption in this lab.

1. Locate the Client Hello and Server Hello messages in the log and view the messages all the way to the Encrypted Alert. These messages should roughly follow the usual outline above.

Task

Draw a timeline showing and naming the SSL messages sent between the client and the server. Draw vertical lines to represent the client and the server, with time running down the page. To work out the names of the SSL messages you should include, use the description given in the Info column of your Wireshark display. You should cover the entire connection, but you can omit most of the encrypted SSL messages in the middle of the connection (Application Data) to keep the figure simple.

2. Find and inspect the details of both the Client Hello and the Server Hello. Expand the Handshake Protocol block to examine the data inside these blocks. As the encryption scheme has not been established yet, we can still see the contents of these blocks.

[4] As data is encrypted, we cannot see inside this block. It is possible to configure WireShark so it can decrypt these blocks, but it is outside the scope of this lab.

Questions

How long, in bytes, is the random data in the Hellos?

How long, in bytes, is the session identifier sent by the server?

What Cipher method is chosen by the Server? Give its name and value.

3. Next, find and inspect the details of the Certificate message. You will notice that this message contains several TLS blocks. As with the Hellos, the contents of this message are also visible.

Questions

What are the TLS blocks that are included in this message?

Who sends the certificate? The client, the server or both?

How many certificates are being sent? Think about why this number of certificates is being sent.

4. Find and inspect the details of the Client Key Exchange message. Again, it is un-encrypted and also contains several blocks. The last message is an encrypted handshake, indicating the client is now ready to start encrypting future messages.

Questions

What are the TLS blocks that are included in this message?

At the Record Layer, what Content-Type values are used to indicate each of these messages? Are these values the same or different than that used for the Hello and Certificate messages? *Note that this question is asking you to look at the Record Layer and not an inner Hand-shake Protocol.*

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

What are the contents carried inside the Change Cipher Spec message?

5. Find and inspect the details of the New Session Ticket message. This message is the last un-encrypted message that is sent form the server. The blocks in this message serve two purposes. First, they confirm that the connection is now secure and all future messages will be encrypted. Second, they prepare session resumption for future connections. As session resumptions are outside the scope of this lab, we won't be covering this message.
6. Finally, find and inspect the details of an Alert message at the end of the trace. Alerts are used to signal a condition has changed for the connection.

Questions

At the Record Layer, what Content-Type value is used to signal an alert?

How are the contents of the alert sent? Are they encrypted or plain-text?

You have now finished lab 8. Remember to complete the Canvas quiz for this lab (https://canvas.auckland.ac.nz/courses/47894/quizzes/50658).