

Practical TLS – Lab Guide

A deep dive into SSL and TLS – the protocols that secure the Internet

Lab 6.1 – Inspecting a TLS Handshake with Wireshark

In this lab you will be inspecting a full TLS handshake in Wireshark. You will be looking at a packet capture of an actual TLS handshake and picking apart the pieces we illustrated in the course.

This lab uses freely available software known as Wireshark. Wireshark lets you capture packets sent to or from your computer. In this lab, you will be studying a saved packet capture of a TLS conversation. Wireshark can be downloaded for free for any platform here: <https://www.wireshark.org/>

Create working directory and acquire lab files

1. Navigate to the folder you created in Lab 0.0 named Practical-TLS
 - `$ cd Practical-TLS`
2. Create and navigate to a new directory for Lab 6.1 files:
 - `$ mkdir "LAB6.1 - Inspecting a TLS Handshake"`
 - `$ cd "LAB6.1 - Inspecting a TLS Handshake"`
3. Download one of the following lab files and place it in the newly created folder:
 - **LAB 6.1 Files - Inspecting a TLS Handshake - Practical TLS.tar.gz**
 - **LAB 6.1 Files - Inspecting a TLS Handshake - Practical TLS.zip**

The lab files are in .zip or .tar.gz format – the contents within are identical, use whichever file is easier for you
4. Unzip or Untar the file you downloaded into your LAB 6.1 working directory:
 - For the TAR.GZ file, you will use the Linux command: `tar -xvzf "LAB 6.1 ..."`
 - For the ZIP file, it should be something like right clicking the file and selecting "Extract All..."
5. When finished, you should have the following **1 file** in the LAB 6.1 directory:

Practical TLS - RSA KX.pcap

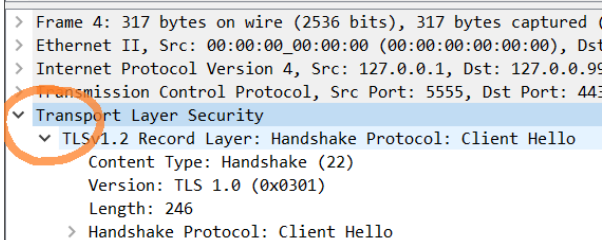
Inspecting a TLS Handshake with Wireshark

Follow the instructions below to explore the TLS handshake. **Step 5 and beyond will include questions** to anchor topics we discussed in the course lessons. **Write down the answers** to all the questions and compare them to the answer key at the end of this guide.

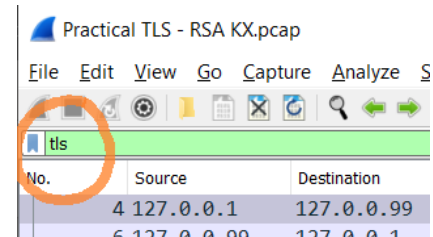
1. Open the file **Practical TLS - RSA KX.pcap** in Wireshark
 - Every line in this file is captured packet from a short TLS web browsing session
 - (optional) If you are familiar with TCP, try to locate the TCP three-way-handshake

2. Filter the display to only show you the TLS conversation
 - In the display filter, type **tls** and press enter

3. Click on a packet and look at the Packet Details pane
 - Expand the Transport Layer Security header
 - Expand the TLS Record Layer header



> Frame 4: 317 bytes on wire (2536 bits), 317 bytes captured (2536 bits) on interface 0
> Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.99
> Transmission Control Protocol, Src Port: 5555, Dst Port: 443
▼ Transport Layer Security
 ▼ TLS 1.2 Record Layer: Handshake Protocol: Client Hello
 Content Type: Handshake (22)
 Version: TLS 1.0 (0x0301)
 Length: 246
 > Handshake Protocol: Client Hello



Practical TLS - RSA KX.pcap

No.	Source	Destination
4	127.0.0.1	127.0.0.99
5	127.0.0.99	127.0.0.1

4. Identify the three fields in the Record Header:
 - Content Type *(in the course lessons this was described as the Record Type)*
 - Version
 - Length
5. Identify the IP address of the Client and the IP address of the Server
 - Hint: Consider the Source IP address of the Client Hello and the Server Hello
6. Using the Content Type field in the Record Header, list the Packet Numbers which include each of the four SSL Records we discussed:
 - Content Type: 20 – Change Cipher Spec
 - Content Type: 21 – Alert
 - Content Type: 22 – Handshake
 - Content Type: 23 – Application Data
7. Locate each Handshake record Subtype below and identify the Packet Number in the capture:
 - Client Hello
 - Server Hello
 - Certificate
 - Server Hello Done
 - Client Key Exchange

8. Look inside the Client Hello and Server Hello for each of these fields:
 - Version
 - Random Number
 - Session ID
 - Cipher Suite(s)
 - Extensions
9. Using the fields above, answer the following questions:
 - What was the highest version of TLS supported by the Client?
 - What was the highest version of TLS supported by the Server?
 - What Cipher Suites were suggested by the Client?
 - What Cipher Suite was selected by the Server?
 - What protocols are being used for Authentication, Key Exchange, Symmetric Encryption, and HMAC?
 - How many Extensions were suggested by the Client?
10. Inspect the contents of the **Handshake: Certificate** record:
 - What was the total length in Bytes of the Certificate record?
 - Would this entire record have fit in a typical IP Packet?
 - How many Certificates were sent?

Note: For the following questions, you will need to drill down and expand as needed within each Certificate

 - What is the Common Name of the Subject of each Certificate?
 - What is the Common Name of the Issuer of each Certificate?
 - What is the Validity duration for each Certificate?
 - What x509 extensions were included in each Certificate?
11. Inspect the contents of the **Handshake: Client Key Exchange** record:
 - What value is contained within this Record?
 - Is this value sent as Plain Text or Cipher Text?
 - What key was used to Encrypt this value?
 - What will be the ultimate purpose of this value?
12. Inspect the contents of *both* **Change Cipher Spec** records:
 - What is the purpose of this record?
 - What should we expect of every record sent *after* the Change Cipher Spec?
 - What record do you see sent immediately after each Change Cipher Spec record?
 - According to the Course Lessons, what Record should we see immediately after C.C.S.?
13. Inspect the content of any **Application Data** records:
 - What Record Headers can you see in plain text?
 - Can you tell which part of the application data is the HMAC Digest or Padding?
14. Locate any **Alert** records you see:
 - What Record Headers can you see in plain text?
 - Can you see the Severity or Description of this Alert? Why or Why Not?
15. Does this TLS Handshake provide Forward Secrecy?
 - What are the implications of providing or not providing Forward Secrecy?

This page intentionally left blank

Answers for the questions above are on the next page. Check your answers.

If you are unclear about a question and answer, ask about it in the Discord server.

Answer Key

Answers are in Black. Text from the original questions are in Gray.

5. Client IP Address: 127.0.0.1 Server IP Address: 127.0.0.99
6. Using the Content Type field in the Record Header, list the Packet Numbers which include each of the four SSL Records we discussed:
- Content Type: 20 – Change Cipher Spec – Packets 8 and 10
 - Content Type: 21 – Alert – Packet 17
 - Content Type: 22 – Handshake – Packets 4,6,8, and 10
 - Content Type: 23 – Application Data – Packets 12 and 14
7. Locate each Handshake record Subtype below and identify the Packet Number in the capture:
- Client Hello – Packet 4
 - Server Hello – Packet 6
 - Certificate – Packet 6
 - Server Hello Done – Packet 6
 - Client Key Exchange – Packet 8
8. Look inside the Client Hello and Server Hello for each of these fields:
- Version
 - Random Number
 - Session ID
 - Cipher Suite(s)
 - Extensions
9. Using the fields above, answer the following questions:
- TLS v1.2
 - TLS v1.2
 - Client suggested 17 Cipher Suites:
- | | |
|---|-----------------------------------|
| TLS_AES_256_GCM_SHA384 | TLS_DHE_RSA_WITH_AES_128_CBC_SHA |
| TLS_CHACHA20_POLY1305_SHA256 | TLS_RSA_WITH_AES_256_GCM_SHA384 |
| TLS_AES_128_GCM_SHA256 | TLS_RSA_WITH_AES_128_GCM_SHA256 |
| TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 | TLS_RSA_WITH_AES_256_CBC_SHA256 |
| TLS_DHE_RSA_WITH_CHACHA20_POLY1305_SHA256 | TLS_RSA_WITH_AES_128_CBC_SHA256 |
| TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 | TLS_RSA_WITH_AES_256_CBC_SHA |
| TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 | TLS_RSA_WITH_AES_128_CBC_SHA |
| TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 | TLS_EMPTY_RENEGOTIATION_INFO_SCSV |
| TLS_DHE_RSA_WITH_AES_256_CBC_SHA | |
- Server selected: TLS_RSA_WITH_AES_128_CBC_SHA256 (0x003c)
 - Authentication: RSA Key Exchange: RSA
 - Symmetric Encryption: AES_128_CBC HMAC: SHA256
 - 8 extensions were suggested by the Client

10. Inspect the contents of the **Handshake: Certificate** record:

- 2646 Bytes
- No – IP Packets are typically 1500 Bytes maximum
- 2 Certificates were sent:

<u>CA Certificate</u>	<u>End-Entity Certificate</u>
OrangeCA.com	Blue.com
OrangeCA.com	OrangeCA.com
02/25/2021 – 02/23/2031	02/25/2021 – 02/25/2022
Subject Key Identifier	Basic Constraints
Authority Key Identifier	
Basic Constraints	
- CN of Subject:
- CN of Issuer:
- Validity:
- x509 Extensions:

11. Inspect the contents of the **Handshake: Client Key Exchange** record:

- Pre Master Secret
- Cipher Text
- Server's Public Key
- This is the SEED value which will be used to generate a Master Secret, which will then be used to generate the Session Keys which will protect the Application Data

12. Inspect the contents of *both* **Change Cipher Spec** records:

- CCS is an indication that the sender has everything it needs to speak securely (i.e., the Session Keys have been calculated)
- Every record after the CCS is protected / encrypted by the Session Keys
- Encrypted Handshake Message
- Finished Record

13. Inspect the content of any **Application Data** records:

- Content/Record Type, Version, Length
- No. The padding and HMAC digest are included in the encryption.

14. Locate any **Alert** records you see:

- Content Type (Record Type), Version, Length
- No. Since the Alert was sent after the Change Cipher Spec, the contents are encrypted.

15. Does this TLS Handshake provide Forward Secrecy? No

- What are the implications of providing or not providing Forward Secrecy?
If you were to acquire the Server's Private Key, you could apply it to this Handshake to extract the Pre-Master-Secret, calculate the Master Secret, calculate the Session Keys, and then decrypt all the Application Data.