BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INFORMATION SYSTEMS
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

HTTPS SECURE COMMUNICATION ANALYSIS
ANALÝZA HTTPS ZABEZPEČENÉ KOMUNIKACE

IBS - COMPUTER COMMUNICATION ANALYSIS
IBS - ANALÝZA POČÍTAČOVÉ KOMUNIKACE

AUTHOR ANDREI SHCHAPANIAK
AUTOR PRÁCE

BRNO 2022

Contents

Chapter 1

Introduction

A lot of people have been spending their free time surfing the internet and visiting their favorite websites. When they type the URL of the website in the address bar of the Internet browser, they will have the desired web page in a moment. But not many of them know, how it works on the backend and how their devices transfer information to the server.

A network is a big and complex system that contains at least two computers, either by a cable or a wireless connection. To describe a network system were developed different models. One of them is Open Systems Interconnection[1] (**OSI**) model. It contains 7 different layers and all of them work collaboratively to transmit the data from one device to another across the network. This article covers a small part of the application layer namely the protocols for communication between a web browser and a server.

The Hypertext Transfer Protocol[2] (**HTTP**) is an application layer protocol. It allows users to communicate data on the Internet. But this protocol does not encrypt connections. That means data is visible to anyone. Therefore, using unencrypted HTTP can pose serious security risks, if the user wants to send sensitive data like credentials.

For secure purposes is used HyperText Transfer Protocol Secure[3] (**HTTPS**), which can protect all sensitive information. It is possible because it uses the Transport Layer Security protocol[4] (**TLS**). This protocol provides the following features also:

- Authentication of the accessed website.
- Protection of the privacy.
- Integrity of the exchanged data.

In the next chapter you will know more interesting things about HTTPS protocol in practice.

[1] https://en.wikipedia.org/wiki/OSI_model

[2] https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol

[3] https://en.wikipedia.org/wiki/HTTPS

[4] https://en.wikipedia.org/wiki/Transport_Layer_Security

Chapter 2

HTTPS connection analysis

This chapter will describe TLS handshake and HTTPS decryption. For these purposes, an open-source network protocol analysis software program, Wireshark[1] was used. It is a very powerful tool, which provides monitoring of the network traffic on the local network, decryption of many protocols and other useful functions for network analysis.

TLS handshake

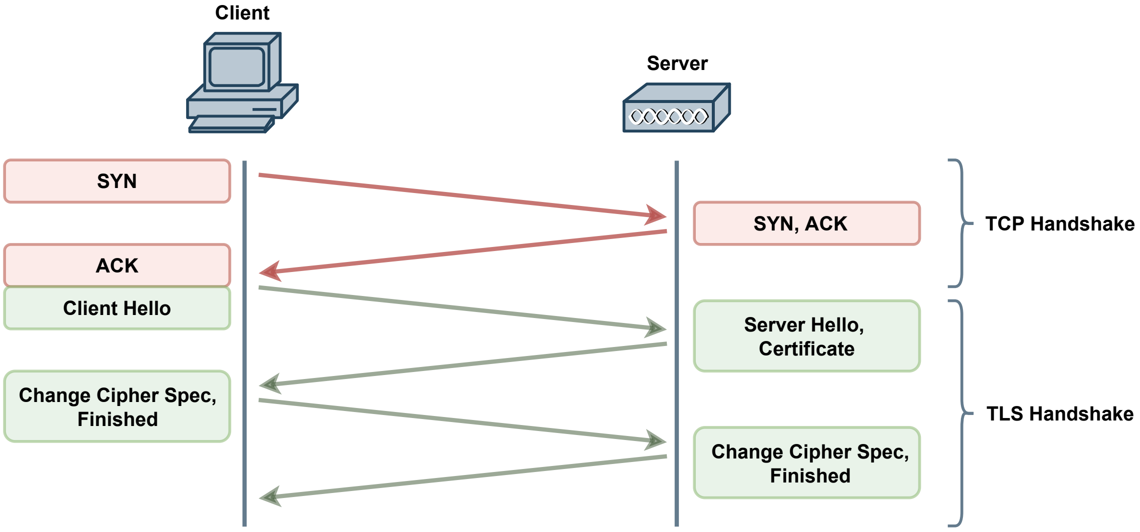
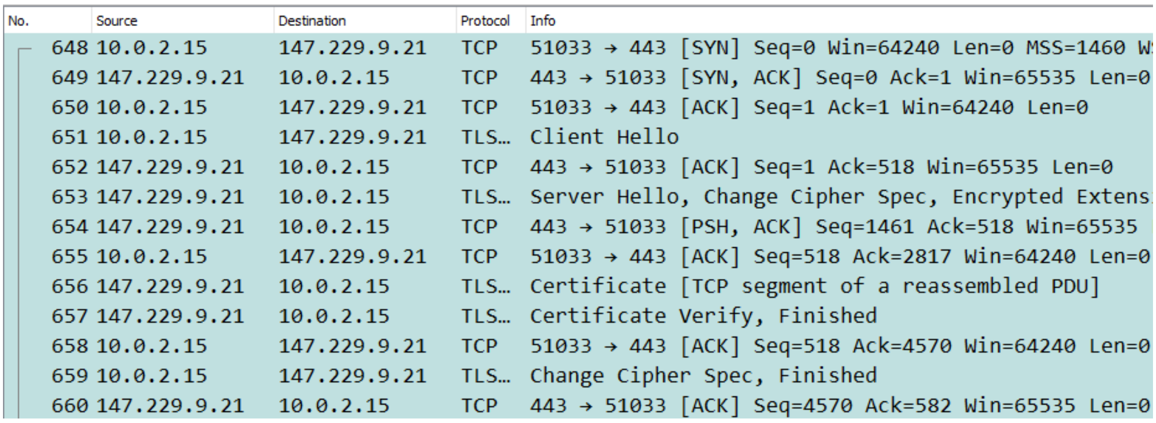


Figure 2.1: Overview of the TLS handshake.

The figure 2.1 shows how the TLS handshake works. First, a TCP 3-way handshake is performed to get an established and reliable connection. Then TLS handshake occurs because the connection uses HTTPS protocol and the browser first begins to query the website's origin server. It is based on asymmetric cryptography and is needed for session keys exchanging between server and client to make further conversation possible. A session key is a symmetric key, which is used to encrypt data for only one communication session. More information about the steps of TLS handshake is described here [2].

[1] https://www.wireshark.org



No.	Source	Destination	Protocol	Info
648	10.0.2.15	147.229.9.21	TCP	51033 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 W
649	147.229.9.21	10.0.2.15	TCP	443 → 51033 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0
650	10.0.2.15	147.229.9.21	TCP	51033 → 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
651	10.0.2.15	147.229.9.21	TLS…	Client Hello
652	147.229.9.21	10.0.2.15	TCP	443 → 51033 [ACK] Seq=1 Ack=518 Win=65535 Len=0
653	147.229.9.21	10.0.2.15	TLS…	Server Hello, Change Cipher Spec, Encrypted Extens
654	147.229.9.21	10.0.2.15	TCP	443 → 51033 [PSH, ACK] Seq=1461 Ack=518 Win=65535
655	10.0.2.15	147.229.9.21	TCP	51033 → 443 [ACK] Seq=518 Ack=2817 Win=64240 Len=0
656	147.229.9.21	10.0.2.15	TLS…	Certificate [TCP segment of a reassembled PDU]
657	147.229.9.21	10.0.2.15	TLS…	Certificate Verify, Finished
658	10.0.2.15	147.229.9.21	TCP	51033 → 443 [ACK] Seq=518 Ack=4570 Win=64240 Len=0
659	10.0.2.15	147.229.9.21	TLS…	Change Cipher Spec, Finished
660	147.229.9.21	10.0.2.15	TCP	443 → 51033 [ACK] Seq=4570 Ack=582 Win=65535 Len=0

Figure 2.2: Wireshark's result.

The figure 2.2 shows a connection in the Wireshark to a website that uses the HTTPS protocol. There you can see 2 handshakes described above.

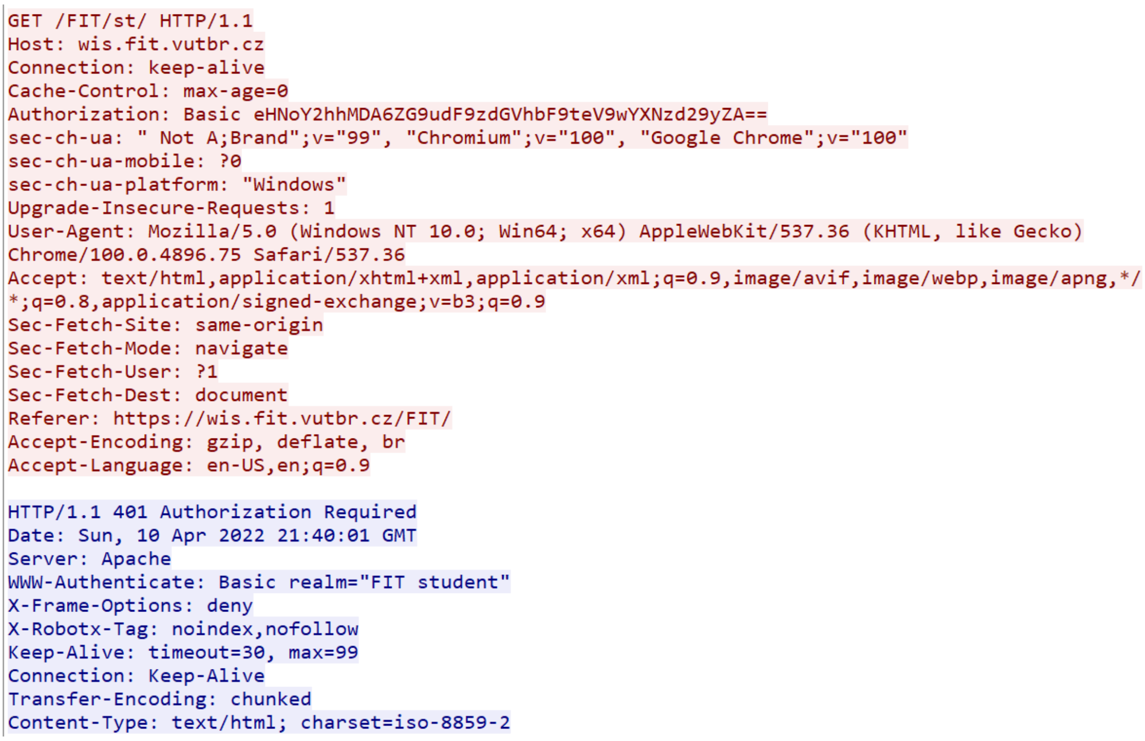
During this process, the following happened:

- An established connection was provided between server and client.
- Session keys were generated by client and server, and using asymmetric cryptography has happened the exchanging of session keys.
- A secure connection was provided, and communication continues using the session keys.

HTTPS decryption

As was described in the previous section, session keys help to encrypt data on the network traffic. Therefore, if we have a session key, we can decrypt data. This possibility provides a Wireshark analyzer. The full guide can be found here [1].

The figure 2.3 shows the HTTP stream with a detailed description of information about the connection and data that was transported over the network.



```
GET /FIT/st/ HTTP/1.1
Host: wis.fit.vutbr.cz
Connection: keep-alive
Cache-Control: max-age=0
Authorization: Basic eHNoY2hhMDA6ZG9udF9zdGVhbF9teV9wYXNzd29yZA==
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="100", "Google Chrome";v="100"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/100.0.4896.75 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/
*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://wis.fit.vutbr.cz/FIT/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9

HTTP/1.1 401 Authorization Required
Date: Sun, 10 Apr 2022 21:40:01 GMT
Server: Apache
WWW-Authenticate: Basic realm="FIT student"
X-Frame-Options: deny
X-Robotx-Tag: noindex,nofollow
Keep-Alive: timeout=30, max=99
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html; charset=iso-8859-2
```

Figure 2.3: Encrypted HTTPS stream.

As you can see, all information, including the type of system, name of the website, and other useful data has become available and readable to us. A line `Referer` confirms that a visited website uses the secure HTTPS protocol.

The most interesting line is `Authorization`. For the common user, this seems like a random set of characters. But for the hacker, it is credentials that are encrypted by the simple cipher Base64[2]. You can try to decode this message and see the credentials. There are no valid credentials for security purposes.

[2] https://en.wikipedia.org/wiki/Base64

Chapter 3

Conclusion

In this article you read about:

- secure communication between web browser and server.
- difference between HTTP and HTTPS protocols.
- providing of the secure connection using two handshakes.
- decryption method of HTTPS protocol in Wireshark.

Decrypt HTTPS is possible if you know session keys. In real life it is very difficult to steal these keys, but it does no mean, that you have 100% of security. There are other ways to get sensitive data.

For better understanding next examples I have recommended read about HTTP Strict Transport Security[1] policy mechanism (**HSTS**).

When the user requests a secure session, the web-server responds by sending its certificate to the user's browser where the certificate will be checked for validity. There is tool `gensslcert` which can create a certificate with two keys, which is called „self-signed". The attacker could replace the original certificates with these self-signed certificates generated on the local proxy server. After that, the user's computer will accept the certificate, because it trusts the proxy server, where the certificate was signed. This attack is called Man-in-the-middle[2] (**MITM**) attack and can be performed in open Wi-Fi networks, because people are connected to an open and unsecured network.

Another situation may be on websites, where an HTTP connection is redirected to the corresponding HTTPS. There is `sslstrip` tool, which prevents the HTTP protocol being upgraded to HTTPS. If you want to read a bit more about MITM attacks on the HTTPS connection, visit this page [3].

[1] https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security

[2] https://en.wikipedia.org/wiki/Man-in-the-middle_attack

Bibliography

[1] Aaron Phillips. *How to Decrypt SSL with Wireshark – HTTPS Decryption Guide* [online]. February 2022 [cit. 2022-04-11]. Available at: https://www.comparitech.com/net-admin/decrypt-ssl-with-wireshark.

[2] CloudFlare. *What happens in a TLS handshake? | SSL handshake* [online]. [cit. 2022-04-11]. Available at: https://www.cloudflare.com/learning/ssl/what-happens-in-a-tls-handshake.

[3] Paul Mutton. *95% of HTTPS servers vulnerable to trivial MITM attacks* [online]. March 2016 [cit. 2022-04-11]. Available at: https://news.netcraft.com/archives/2016/03/17/95-of-https-servers-vulnerable-to-trivial-mitm-attacks.html.