Abdelwahid Benslimane
wahid.benslimane@gmail.com

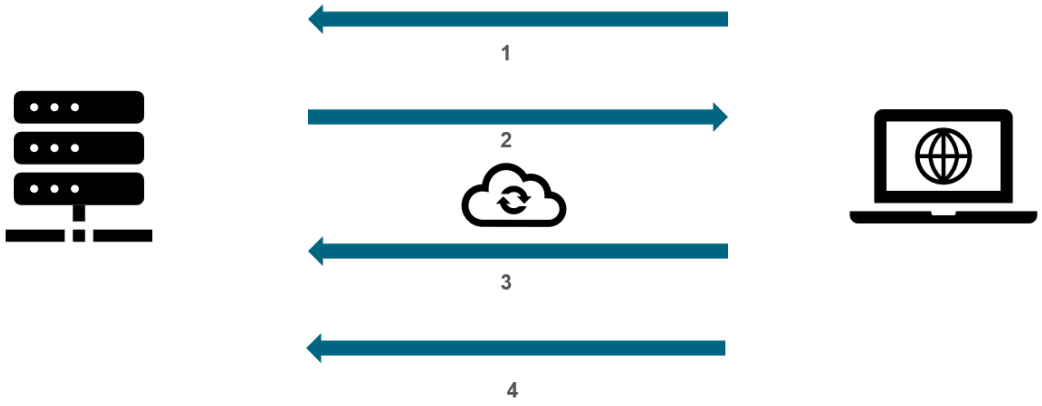
SSL/TLS Certification

Introduction

This document intends to give an overview of how the **SSL/TLS** certification works.

SSL stands for Secure Socket Layer and is a protocol that helps to guarantee the privacy of the data exchanged between a server and a client on Internet by encrypting it, so it won't be understandable for anyone outside of these two latter actors. This protocol is no longer used and has been replaced by a protocol with a similar purpose, TLS (Transport Layer Security). I won't dwell on the similarities or differences between these two protocols, but both serve the same purpose which is, to put it simply, somewhat creating a specific language ad hoc for the communication between two computing units and this is where the **SSL certificates** (these certificates are still called SSL certificates, even if TLS is now used) come into play as the key element for the translation.

Before going further into details, let's review the main steps of a secured communication's establishment described in the scheme below.



1. The clients request a connection secured by SSL/TLS.

2. The server sends its **public certificate** as well as its **Certification Authority** public certificate. The client checks the validity and signature.

3. The client sends its public encryption key, itself encrypted thanks to the **public key** contained in the server's public certificate.

4. The server decrypts the public key of the client thanks to its own **private key**. A secured connection can be established. The client will use the server's public key to encrypt messages to the server that will be decrypted thanks to the server's private key, and the server will use the client's public key to encrypt messages to the client that will be decrypted thanks to the client's private key.

What is a public SSL certificate?

A public certificate is somehow the numerical ID card that links an encryption key to the information describing in a unique way a server/website.

If we take the example of an asymmetric algorithm such as RSA (for A. Shamir, R. Rivests and L. Adleman the three inventors of this algorithm), an encryption key is created during the generation of the certificate and is used to encrypt the communication that only the server identified by the certificate is allowed to receive. This key is public and, thereby, meant to be distributed. The adjective asymmetric is related to the fact that two different keys are involved.

The communication is then decrypted thanks to the private key detained by the server. The private key must remain absolutely secret (known by its owner only).

Some mathematical considerations

The public and private keys are in fact the result of an arithmetic operation:

Let p and q be two prime numbers, we set $n = pq$ and e such that e is coprime to $(p-1)(q-1)$. The pair (n, e) is in fact the public key.

As e is coprime to $(p-1)(q-1)$, we can find an integer d such that $ed \equiv 1\ (mod\ (p-1)(q-1))$, which means that $ed - 1$ is a multiple of $(p-1)(q-1)$.

The pair (n, d) is the private key.

We can build d from e, p and q thanks to the **Euclid's algorithm** and this is what the RSA algorithm is doing.

Below, the RSA workflow:

1. Alice wants to send a message to Bob. This message is converted to an integer M $(M < n)$.
2. Alice knows Bob's public key and calculates $C \equiv M^e (mod\ n)$ and then sends C to Bob.
3. Bob receives the message C and calculates $D \equiv C^d (mod\ n)$ thanks to its private key.
4. Thanks to **Euler's theorem**, we know that $D \equiv C^d (mod\ n) \equiv M^{ed} (mod\ n) \equiv M\ (mod\ n)$. The original message has in fact been reconstituted, which would have been theoretically possible but materially very difficult or impossible without knowing the number d because of the time it would have taken to a computer to do the calculation, especially if the key length in bits is large.

Self-signed certificates vs CA signed certificates

A certificate, just as an ID card is issued by an official authority: the State of the card's owner, has to be issued and signed by a trustable **Certification Authority** (CA), established and recognized as such by all the Internet actors. However, for internal use and access, organizations may issue the certificates themselves. In this case we speak of **self-signed certificates**.

The information related to the issuer is gathered with all the other information available in the server's public certificate (duration of the certificate, algorithm used…).

In order to validate the CA's signature present in the server's certificate, the client needs to communicate with the Certification Authority, and to do so, another certificate is needed: the CA public certificate.

On its turn, the CA certificate can be signed by another Certification Authority and so on. We call it a chain of trust.

The client needs to be sent by the server all the certificates of the Certification Authorities forming the chain. The certificate at the end of the chain is of course a self-signed certificate since the **root CA** must issue a certificate for itself.

This scheme taken from Wikipedia explains clearly the principle of the chain of trust:

(source: https://en.wikipedia.org/wiki/Chain_of_trust#/media/File:Chain_Of_Trust.svg)