An Overview of the TLS Handshaking Protocol

December 5, 2019

Duncan Freeman

Contents

List of Figures

Introduction

Transport Layer Security (TLS) is a protocol that facilitates privacy and authentication of messages transmitted over a network (RFC 5246, 1). This document provides an overview of the TLS handshake subprocedure. This document does not cover implementation details, message transport, specific ciphers, or application layer protocols. This document is arranged as follows: First, background information on TLS will be provided, followed by the two stages of the handshake procedure: Method Agreement, and Key Exchange & Authentication. The conclusion will describe the utilization of the secure channel that the handshake establishes.

Background information

TLS is a protocol used to secure private and untampered communications on the web (RFC 5246, 6, 78). For example, HTTPS is the original HTTP (Hyper-Text Transfer Protocol) transmitted over a TLS transport instead of TCP (HTTP over TLS, 1). TLS is not a specific cipher or encryption method in itself; it is a standard container format that facilitates the use of a set of existing encryption methods (RFC 8446, 7).

The TLS handshaking protocol

First, a protocol version, cipher suite, and compression method is established. Then, keys are exchanged and digital certificates are verified.

Stage I: Method agreement

The client presents an array of protocol versions and ciphers it is willing to use, and the server selects one in response. Optionally, the pair may resume a prior session instead.

1. The client begins the conversation by sending a ‘ClientHello’ message (See Figure 1), which contains information about what protocols it is capable of communicating over (RFC 8446, 12, RFC 5246, 34). The message includes the specific version of TLS that is intended to be used, encryption schemes the client supports, and compression methods available (RFC 8446, 12, RFC 5246, 34). Because encryption has not been established at this stage, this message is unencrypted; a MitM (Man-in-the-Middle) attack could be used to weaken the protocol by selecting for older versions of TLS or specific broken ciphers (RFC 5246, 34, 97). It is the responsibility of the applications using TLS to ensure that they do not allow communication over outdated

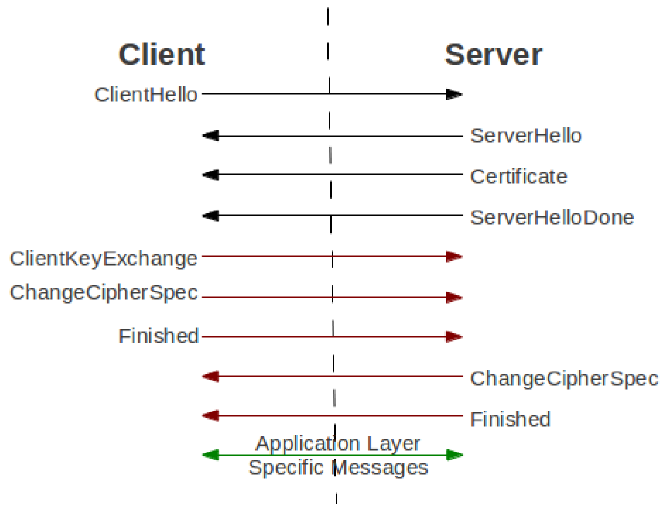

Figure 1: TLS Handshake

The TLS Handshake procedure is composed of a small number of negotiating messages that together form the configuration of the resultant message layer. ©*Carnegie Mellon University 2012*

protocols deemed insecure (RFC 5246, 34). Newer versions of TLS help to mitigate these attacks (See step 3).

2. Optionally, the client sends a specialized 'Resume' message that describes a previous connection from which encryption and protocol parameters are to be reused (RFC 5246, 36, RFC 8446, 39). Following this message, no further authentication is needed for the session, and communication can begin immediately (RFC 5246, 35, 36). This can help to decrease the overhead of TLS in contexts where several concurrent or successive connections are established (RFC 8446, 74).

3. The server picks from the protocol versions, ciphers, and compression methods offered by the client and sends its choice for each. Among the choices given, the server always picks the safest available (RFC 5246, 41, 97). In TLS 1.3, if a MitM protocol downgrade attack is suspected, the server will attempt to expose it by sending the client some false info that signals that it actually supports a safer protocol and that communication should terminate (RFC 8446, 32).

Stage II: Key exchange and authentication

The server sends its certificate and key, and the client (having verified said certificate) sends its own key and optionally its certificate if requested.

1. The server sends its certificate. The certificate proves mathematically that the public key contained in the following message originates from the server the client intended to connect to and that the server's key was signed by a Certificate Authority (See Figure 2) with which the client is familiar and trusts (What Are Certificate Authorities & Trust Hierarchies?, 1). For a server to be trusted by clients, it must obtain

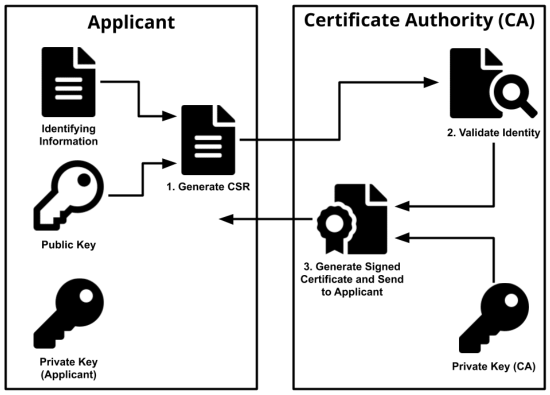

Figure 2: Certificate Authority

The Certificate Authority validates the host's and/or client's public key using it's own private key, which protects the Applicant's connection against tampering. ©*SSL.com 2019*

a certificate from a trusted source with which to sign its key (GlobalSign, 1). It is the client's responsibility to verify the server's authenticity using this certificate (RFC 5246, 85).

2. The server sends its key exchange message. This message contains the cryptographic information the client needs to compute a shared key with which to cipher and decipher information (RFC 5246, 50, RFC 8446, 31). This message may be omitted if there was enough information in the last transmission, to avoid overhead (RFC 5246, 50).

3. Optionally, the server sends the client an authentication request (RFC 5246, 35, RFC 8446, 12). This request is an additional security measure requiring trust on the client side, but does incur overhead in the form of another message from the client side (RFC 8446, 12).

4. The client sends its key exchange message. This message contains cryptographic information the server needs to compute a shared key with which to cipher and decipher information (RFC 5246, 35, 57). In TLS 1.3, all extra or nonstandard handshake communications following this are encrypted with the agreed-upon protocol for additional security (RFC 8446, 8). Barring any custom requirements, the handshake is over and the private channel is now ready for use (RFC 8446, 8).

Conclusion

Following the success of the above procedure, the TLS connection is now initialized and ready for messaging. Messaging is handled over the TLS record protocol (RFC 8446, 71). Application-level protocols such as FTP, HTTP, SMTP and others may then utilize the record protocol to send messages transparently as if using the underlying TCP stream, but with the guarantee of encryption and authentication initialized by the handshake procedure (RFC 5246, 78).

Works cited

"HTTP Over TLS." IETF Tools, May 2000.

"What Are Certificate Authorities & Trust Hierarchies?" What is a Certificate Authority? Globalsign. Accessed November 14, 2019.

"The Transport Layer Security (TLS) Protocol Version 1.3." IETF Tools. IETF, August 2018.

"The Transport Layer Security (TLS) Protocol Version 1.2." IETF Tools. IETF, August 2008.

Bibliography

"A Type-safe, TPM Backed TLS Infrastructure" Type-safe, TPM Backend TLS. Carnegie Mellon University. Accessed November 15, 2019.

"An Overview of TLS 1.3 and Q&A" The CloudFare Blog. CloudFare. Accessed November 16, 2019.

"HTTP Over TLS." IETF Tools, May 2000.

"What Are Certificate Authorities & Trust Hierarchies?" What is a Certificate Authority? Globalsign. Accessed November 14, 2019.

"The Transport Layer Security (TLS) Protocol Version 1.3." IETF Tools. IETF, August 2018.

"The Transport Layer Security (TLS) Protocol Version 1.2." IETF Tools. IETF, August 2008.

"The SSL/TLS Handshake: an Overview" Globalsign. Accessed November 14, 2019.

"Transport Layer Security" Wikipedia. Accessed November 13, 2019.