TLS Implementation

Ext.prc.002.02 | 01.06.2024

UPM 24.x

ZINFI Confidential & Proprietary
Shared Under NDA

 |

Contents

Introduction

Transport layer security (TLS) is likely the most widely deployed security protocol in ZINFI's Unified Partner Management (UPM) platform, particularly for securing web traffic (HTTPS) to UPM instances. To achieve reliable security for TLS, we have developed an implementation of the Internet standard of TLS 1.2. The result is an extensive application of security (the UPM) at the TLS interface down to the cryptographic algorithms selected by TLS 1.2 cipher-suites. Thus, we address application security, protocol security, and cryptographic security in a common implementation framework.

Our primary goals establish:

Standard compliance – Following the details of the implementation we verify the concrete message parsing and processing of TLS, including support of TLS 1.2, as well as cipher-suites, protocol extensions, sessions and connections (with re-handshakes and resumptions). The TLS standard targets messages exchanged over the network. Since this is critical for using TLS securely, we have designed our own REST API based on TLS 1.2 with an emphasis on precision. Our REST API is similar to those provided by popular implementations, but gives more control to the application with TLS 1.2, providing even stronger security properties.

We also implement .NET streams on top of TLS, and program minimal web clients and servers, to ensure our implementation is interoperable with mainstream implementations and offers reasonable usability and performance.

Verified security – Following the proven security approach of computational cryptography, we integrate the privacy and integrity of byte streams sent over TLS, provided their connection keys are established using a strong cipher-suite between principals using secure long-term keys. In brief, results are expressed using indistinguishability algorithms, whereby the communication content is replaced with zeros before sending and restored by table lookups after receiving. In the process of verifying our implementation, we also establish functional properties, logical authentication goals and state machine invariants.

Organization of TLS 1.2 at UPM

The following steps are involved in the implementation of TLS 1.2 for UPM's client/server communication:

- Handshake and cipher suite negotiation
- Authentication of parties
- Key-related information exchange
- Application data exchange

TLS 1.2 is primarily divided into two protocols that, together, provide connection security:

- TLS 1.2 Handshake Protocol
- TLS 1.2 Record Protocol

UPM's TLS Handshake Protocol Implementation

In the UPM platform, the Transport Layer Security (TLS 1.2) Handshake Protocol is responsible for the authentication and key exchange necessary to establish or resume secure sessions between the UPM instance and its clients. When establishing a secure session, the Handshake Protocol manages the following:

- Cipher suite negotiation
- Authentication of the server and optionally, the client
- Session key information exchange.

UPM's Cipher Suite Negotiation

The UPM client and server communicate and choose the cipher suite that will be used throughout the message exchange cycle.

Authentication of UPM Client

The UPM server and the client proves their individual identity to each other, through the usage of PKI, the public/private key pairs which becomes the basis of this authentication. The exact method used for authentication is determined by the cipher suite negotiated by the two parties.

Key Exchange between UPM Client and Server

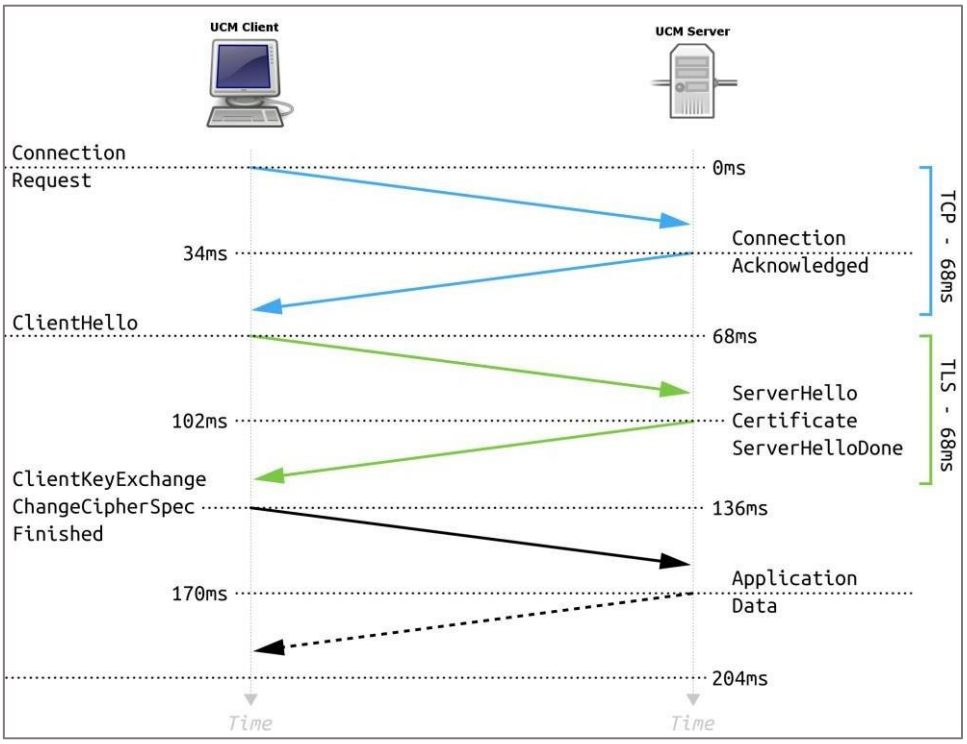
The UPM client and server exchange random numbers and a special number called the Pre-Master Secret. These numbers are combined with additional data permitting that specific client and server to create their shared secret, called the Master Secret. The Master Secret is used by the specific UPM client and UPM server to generate the write MAC secret, which is the session key used for hashing, and the write key, which is the session key used for encryption.

Establishing a Secure Session across UPM by Implementing TLS 1.2

The TLS Handshake Protocol integration involves the following steps:

- The UPM client sends a "Client hello" message to the UPM server, along with the client's random value and supported cipher suites.
- The UPM server responds by sending a "Server hello" message to the UPM client, along with the server's random value.
- The UPM server sends its certificate to the client for authentication and may request a certificate from the client. The server sends the "Server hello done" message.
- If the UPM server has requested a certificate from the client, the client sends it.

- The UPM client creates a random Pre-Master Secret and encrypts it with the public key from the server's certificate, sending the encrypted Pre-Master Secret to the server.
- The UPM server receives the Pre-Master Secret. The UPM server and client each generate the Master Secret and session keys based on the Pre-Master Secret.
- The UPM client sends “Change cipher spec” notification to server to indicate that the client will start using the new session keys for hashing and encrypting messages. The UPM client also sends a “Client finished” message.
- The UPM server receives “Change cipher spec” and switches its record layer security state to symmetric encryption using the session keys. The server sends "Server finished" message to the client.
- Both, UPM Client and server can now exchange application data over the secured channel they have established. All messages sent from the UPM client to UPM server and vice-versa are encrypted using session key.



Resuming a Secure Session at UPM by Implementing TLS 1.2

- The UPM client sends a “Client hello” message using the Session ID of the session to be resumed.
- The UPM server checks its session cache for a matching Session ID. If a match is found, and the server is able to resume the session, it sends a “Server hello” message with the Session ID.

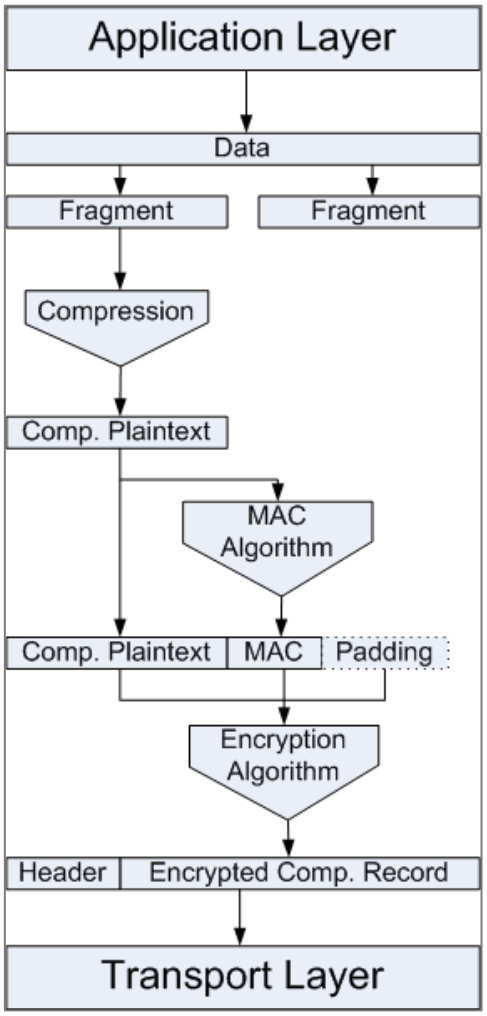
  Note: If a session ID match is not found, the UPM server generates a new session ID and the TLS UPM client and server perform a full handshake.

- The UPM client and server must exchange “Change cipher spec” messages and send “Client finished” and “Server finished” messages to resume UPM application data exchange over the secure channel.

UPM’s TLS Record Protocol Implementation

The Transport Layer Security (TLS 1.2) Record protocol secures UPM’s application data using the keys created during the Handshake. The Record Protocol is responsible for securing UPM’s application data and verifying its integrity and origin. It manages the following:

- Dividing outgoing messages into manageable blocks and reassembling incoming messages.
- Compressing outgoing blocks and decompressing incoming blocks (optional).
- Applying a Message Authentication Code (MAC) to outgoing messages, and verifying incoming messages using the MAC.
- Encrypting outgoing messages and decrypting incoming messages.



When the Record Protocol is complete, the outgoing encrypted data is passed down to the Transmission Control Protocol (TCP) layer for transport.

Key Exchange or Key Agreement

Before a UPM client and server can begin to exchange information protected by TLS 1.2, they must securely exchange or agree upon an encryption key and a cipher to use when encrypting data. Among the methods used for key exchange/agreement, the pre-dominantly used technique is: public and private keys generated with RSA (denoted TLS_RSA in the TLS 1.2 handshake protocol) (SHA-256 With RSA Encryption). Cipher utilized is commonly AES.

