جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

قسم الامن السيبراني
DEPARTMENT OF CYBER SECURITY

SUBJECT:

AUTHENTICATION AND ACCESS CONTROL

CLASS:

SECOND

LECTURER:

DR. SUHA ALHUSSIENY

LECTURE: (6)

TRANSPORT LAYER SECURITY (TLS) AND SECURE SOCKETS LAYER (SSL)

Transport Layer Security (TLS) and Secure Sockets Layer (SSL)

TLS and SSL are cryptographic protocols designed to provide communications security over a computer network. Several versions of the protocols find widespread use in applications such as web browsing, email, instant messaging, and voice over IP (VoIP).

Websites can use TLS to secure all communications between their servers and web browsers.

✓ Secure Socket Layer (SSL) provides security services between TCP and applications that use TCP. The Internet standard version is called Transport Layer Service (TLS).

✓ SSL/TLS provides confidentiality using symmetric encryption and message integrity using a message authentication code(MAC).

✓ SSL/TLS includes protocol mechanisms to enable two TCP users to determine the security mechanisms and services they will use.

✓ HTTPS (HTTP over SSL) refers to the combination of HTTP and SSL to implement secure communication between a Web browser and a Web server.

✓ Secure Shell (SSH) provides secure remote logon and other secure client/server facilities.

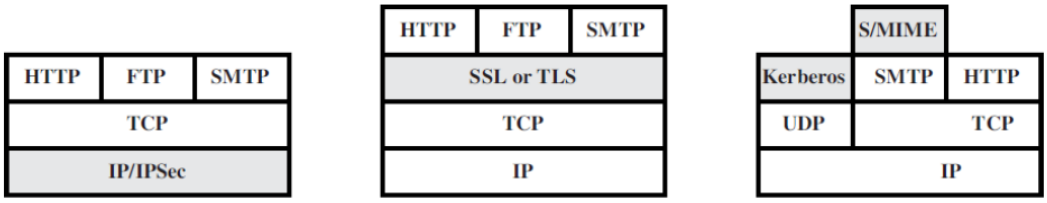
Web Traffic Security Approaches

Several approaches to providing Web security are possible. The various approaches that have been considered are similar in the services they provide and, to some extent, in the mechanisms that they use, but they differ with respect to their scope of applicability and their relative location within the TCP/IP protocol stack.

One way to provide Web security is to use IP security (IPsec). The advantage of using IPsec is that it is transparent to end users and applications and provides a general-purpose solution. Furthermore, IPsec includes a filtering capability so that only selected traffic need incur the overhead of IPsec processing.

The foremost example of this approach is the Secure Sockets Layer (SSL) and the follow-on Internet standard known as Transport Layer Security (TLS). At this level, there are two implementation choices. For full generality, SSL (or TLS) could be provided as part of the underlying protocol suite and therefore be transparent to applications. Alternatively, SSL can be embedded in specific packages. For example, Netscape and Microsoft Explorer browsers come equipped with SSL, and most Web servers have implemented the protocol.

Application-specific security services are embedded within the particular application. Figure below shows examples of this architecture. The advantage of this approach is that the service can be tailored to the specific needs of a given application.



(a) Network level (b) Transport level (c) Application level

SSL Architecture

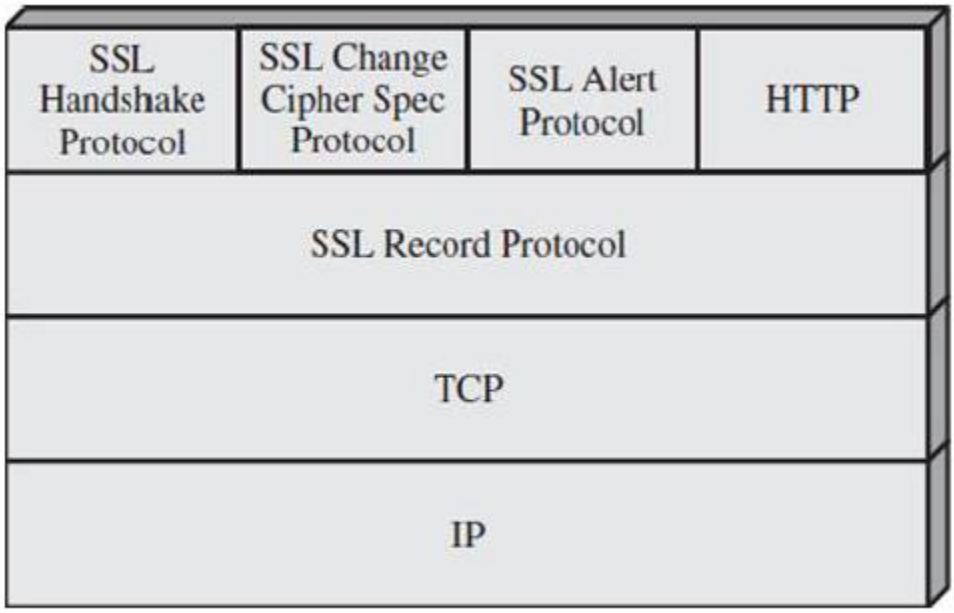
SSL is designed to make use of TCP to provide a reliable end-to-end secure service. SSL is not a single protocol but rather two layers of protocols, as illustrated in Figure below.

The SSL Record Protocol provides basic security services to various higher layer protocols. In particular, the Hypertext Transfer Protocol (HTTP), which provides the transfer service for Web client/server interaction, can operate on top of SSL. Three higher-layer protocols are defined as part of SSL: the Handshake Protocol, The Change Cipher Spec Protocol, and the Alert Protocol. These SSL-specific protocols are used in the management of SSL exchanges and are examined later in this section.

Two important SSL concepts are the SSL session and the SSL connection, which are defined in the specification as follows.

Connection: A connection is a transport (in the OSI layering model definition) that provides a suitable type of service. For SSL, such connections are peer-to-peer relationships. The connections are transient. Every connection is associated with one session.

Session: An SSL session is an association between a client and a server. Sessions are created by the Handshake Protocol. Sessions define a set of cryptographic security parameters which can be shared among multiple connections. Sessions are used to avoid the expensive negotiation of new security parameters for each connection.



Between any pair of parties (applications such as HTTP on client and server), there may be multiple secure connections. In theory, there may also be multiple simultaneous sessions between parties, but this feature is not used in practice.

There are a number of states associated with each session. Once a session is established, there is a current operating state for both read and write (i.e., receive and send). In addition, during the Handshake Protocol, pending read and write

states are created. Upon successful conclusion of the Handshake Protocol, the pending states become the current states.

A session state is defined by the following parameters:-

- **Session identifier**: An arbitrary byte sequence chosen by the server to identify an active or resumable session state.
- **Peer certificate**: An X509.v3 certificate of the peer. This element of the state may be null.
- **Compression method**: The algorithm used to compress data prior to encryption.
- **Cipher spec**: Specifies the bulk data encryption algorithm (such as null, AES, etc.) and a hash algorithm (such as MD5 or SHA-1) used for MAC calculation. It also defines cryptographic attributes such as the hash size.
- **Master secret**: 48-byte secret shared between the client and server.
- **Is resumable**: A flag indicating whether the session can be used to initiate new connections.

A connection state is defined by the following parameters.

- **Server and client random**: Byte sequences that are chosen by the server and client for each connection.
- **Server write MAC secret**: The secret key used in MAC operations on data sent by the server.

- **Client write MAC secret**: The secret key used in MAC operations on data sent by the client.
- **Server write key**: The secret encryption key for data encrypted by the server and decrypted by the client.
- **Client write key**: The symmetric encryption key for data encrypted by the client and decrypted by the server.
- **Initialization vectors**: When a block cipher in CBC mode is used, an initialization vector (IV) is maintained for each key. This field is first initialized by the SSL Handshake Protocol. Thereafter, the final ciphertext block from each record is preserved for use as the IV with the following record.
- **Sequence numbers**: Each party maintains separate sequence numbers for transmitted and received messages for each connection. When a party sends or receives a change cipher spec message, the appropriate sequence number is set to zero. Sequence numbers may not exceed $2^{64} - 1$.

SSL/TLS Handshake

The **Handshake Protocol** is the process by which the **client** (usually a web browser) and the **server** (a website) establish a **secure, encrypted communication channel**. It happens *before* any actual data is exchanged.

Main Goals of the Handshake:

1. Authenticate the communicating parties (especially the server).

2. Agree on a set of cryptographic algorithms to use.
3. Generate a shared secret key to encrypt data securely

Step-by-Step Process

1- **Client Hello :** The client sends a "hello" message containing supported cryptographic algorithms, a random number, and protocol version (e.g., TLS 1.3).
2- **Server Hello :** The server responds with its own random number, chosen cryptographic algorithms, and its **digital certificate (X.509)** that contains its public key.
3- **Server Authentication :** The client verifies the server's certificate using a trusted **Certificate Authority (CA)**. If valid, it confirms the server's identity.
4- **Key Exchange :** The client and server exchange data to generate a **session key** (e.g., using RSA, Diffie–Hellman, or ECDHE).
5- **Client Authentication (optional) :** Some servers may also request a client certificate for two-way authentication (mutual TLS).
6- **Finished Messages :** Both parties send an encrypted "Finished" message to confirm that the handshake succeeded.
7- **Secure Session Established :** From now on, all communication is encrypted using the shared session key.

SSL Record Protocol

The **SSL Record Protocol** is the core layer of SSL/TLS responsible for **securing and managing the actual data transmission** between the client and the server. While the **Handshake Protocol** establishes a secure *session* and negotiates the

cryptographic parameters, the **Record Protocol** operates during the *connection phase* to ensure that all application data (e.g., web pages, emails, login information) is transmitted **confidentially and intact**.

Main Goal:

To provide ***confidentiality***, ***integrity***, and ***authentication*** for the data exchanged after the handshake is completed.

Step-by-Step Process

1- **Fragmentation** : The protocol divides large messages into smaller blocks called *records* (typically up to 16 KB) to simplify encryption and transmission.
2- **Compression (optional) :** Each record can be compressed to reduce its size and improve efficiency. However, in modern TLS versions, compression is usually disabled to avoid security attacks (e.g., CRIME attack).
3- **Message Authentication Code (MAC) or AEAD Tag :** A *MAC* or an *Authenticated Encryption Tag* is added to ensure **integrity** — that no attacker has modified the message.
4- **Encryption :** The record is encrypted using symmetric encryption (e.g., AES, ChaCha20, or 3DES) with the keys generated during the Handshake phase.
5- **Transmission :** The encrypted records are sent over the TCP connection to the receiver.

On the receiving side: The Record Protocol reverses these steps — decrypts the data, verifies the MAC, decompresses (if needed), and reassembles the message before delivering it to the application layer.

Figure below indicates the overall operation of the SSL Record Protocol.

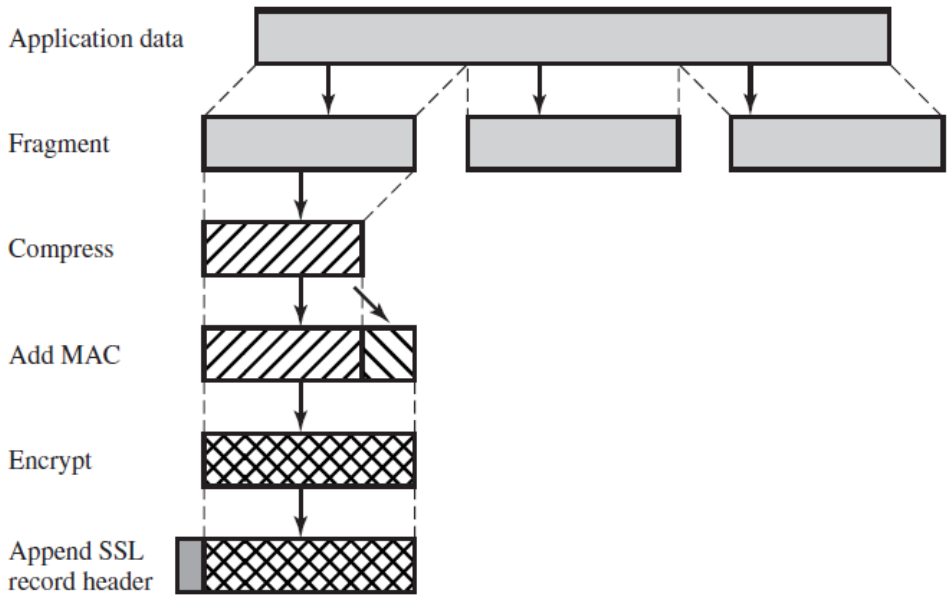


Figure /SSL Record Protocol Operation

H.W/

1. What is the main purpose of TLS and SSL protocols?
2. Which protocol provides security services between TCP and applications that use TCP?
3. What type of encryption is used by SSL/TLS for confidentiality?
4. Which of the following provides message integrity in SSL/TLS?
5. HTTPS stands for:
6. Which protocol provides secure remote logon and other secure client/server facilities?
7. The SSL Record Protocol operates during which phase?
8. Which layer protocol does SSL use to provide reliable service?
9. What is an SSL session?
10. What does the SSL Handshake Protocol mainly establish?
11. In the Handshake Protocol, the message 'Client Hello' contains:
12. The certificate used for server authentication in TLS is:
13. The session key in SSL/TLS is generated using which phase?
14. What does MAC stand for in SSL/TLS?
15. What happens after the SSL handshake is completed successfully?
16. The SSL Record Protocol divides large messages into smaller units called:
17. Which of the following is TRUE about TLS compression?

18. The Handshake Protocol uses which type of encryption algorithms?
19. The master secret in SSL session is how many bytes long?
20. Which field in SSL maintains sequence numbers for transmitted messages?