

CYB650

Introduction to Cryptography

Transport-Level Security



Prepared by Dr. Esraa Saleh Alomari
Lecture #5



جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY

The Part 1 of the module 3 covers the Transport-Level Security are as follows:

- ❑ **Web Security Considerations**
 - **Web Security Threats**
 - **Web Traffic Security**
- ❑ **Secure Sockets Layer**
 - **SSL Architecture**
 - **SSL Record Protocol**
 - **Change Cipher Spec Protocol**
 - **Alert Protocol**
 - **Handshake Protocol**

- ❑ Virtually all businesses, most government agencies, and many individuals now have Web sites. The number of individuals and companies with Internet access is expanding rapidly and all of these have graphical Web browsers. As a result, businesses are enthusiastic about setting up facilities on the Web for electronic commerce.

❑ But the reality is that the Internet and the Web are extremely vulnerable to compromises of various sorts. As businesses wake up to this reality, the demand for secure Web services grows. The topic of Web security is a broad one and can easily fill a book. In this chapter, focus on security at the transport layer: SSL/TLS, HTTPS, and SSH.

The World Wide Web is fundamentally a client/server application running over the Internet and TCP/IP intranets. As such, the security tools and approaches discussed so far in this book are relevant to the issue of Web security. However, the following characteristics of Web usage suggest the need for tailored security tools:

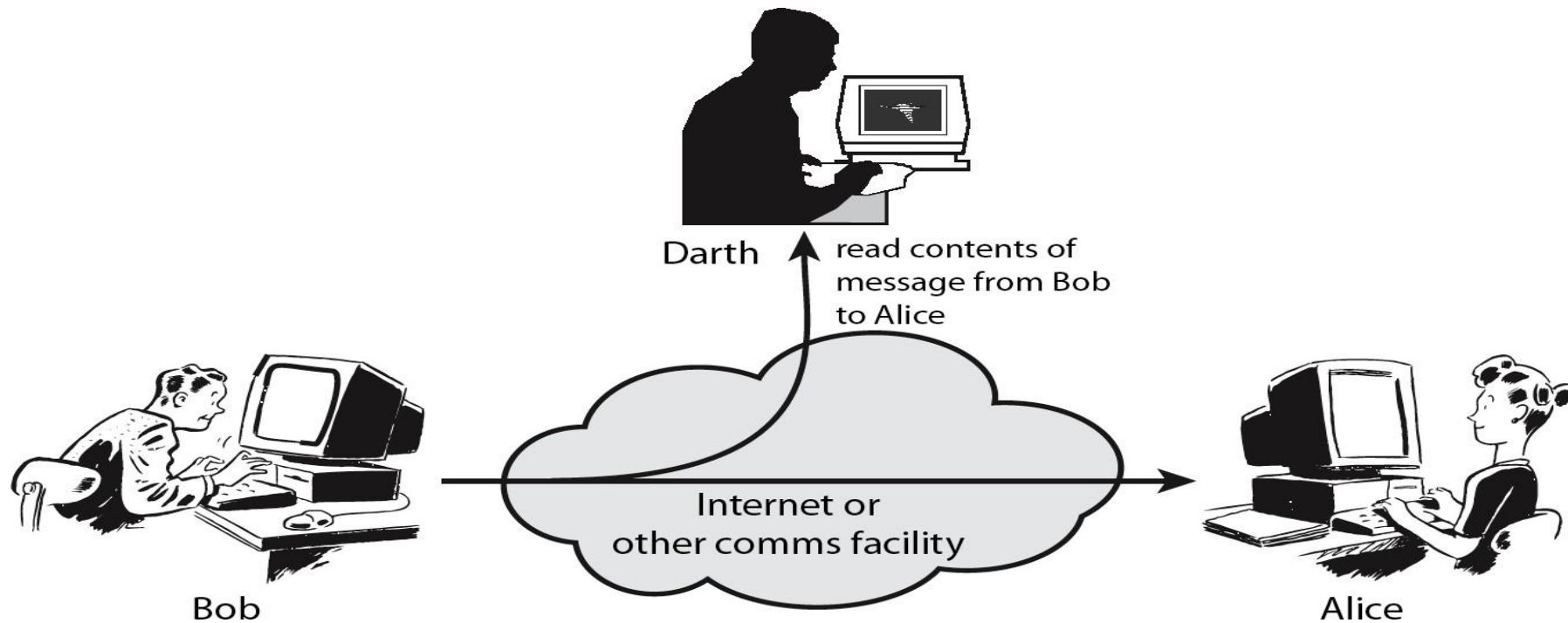
- Although Web browsers are very easy to use, Web servers are relatively easy to configure and manage, and Web content is increasingly easy to develop, the underlying software is extraordinarily complex. This complex software may hide many potential security flaws. The short history of the Web is filled with examples of new and upgraded systems, properly installed, that are vulnerable to a variety of security attacks.

- A Web server can be exploited as a launching pad into the corporation's or agency's entire computer complex. Once the Web server is subverted, an attacker may be able to gain access to data and systems not part of the Web itself but connected to the server at the local site

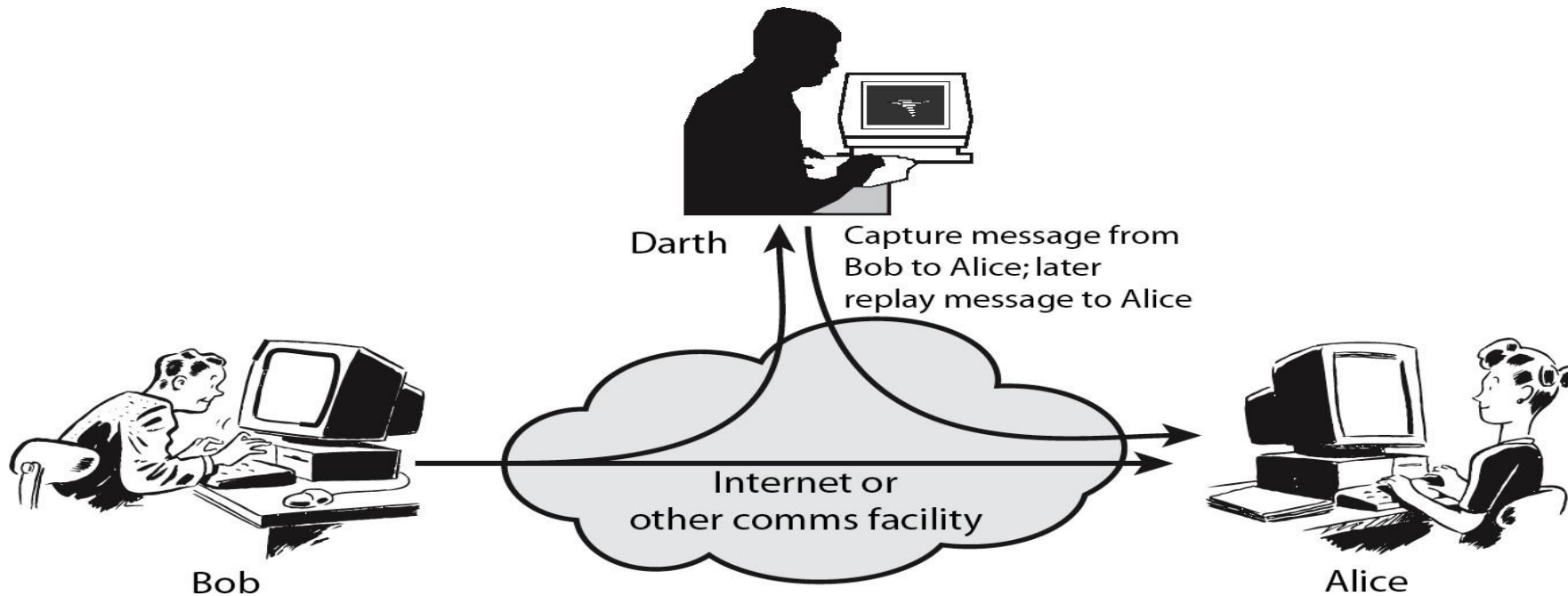
- Casual and untrained (in security matters) users are common clients for Web-based services. Such users are not necessarily aware of the security risks that exist and do not have the tools or knowledge to take effective countermeasures.

- Table 17.1 provides a summary of the types of security threats faced when using the Web. One way to group these threats is in terms of passive and active attacks.

- Passive attacks include eavesdropping on network traffic between browser and server and gaining access to information on a Web site that is supposed to be restricted.
- Active attacks include impersonating another user, altering messages in transit between client and server, and altering information on a Web site.



Passive Attack



Active Attack

- Another way to classify Web security threats is in terms of the location of the threat: Web server, Web browser, and network traffic between browser and server. Issues of server and browser security fall into the category of computer system security;

Web Security Threats

Table 17.1 A Comparison of Threats on the Web

	Threats	Consequences	Countermeasures
Integrity	• Modification of user data • Trojan horse browser • Modification of memory • Modification of message traffic in transit	• Loss of information • Compromise of machine • Vulnerability to all other threats	Cryptographic checksums
Confidentiality	• Eavesdropping on the net • Theft of info from server • Theft of data from client • Info about network configuration • Info about which client talks to server	• Loss of information • Loss of privacy	Encryption, Web proxies
Denial of Service	• Killing of user threads • Flooding machine with bogus requests • Filling up disk or memory • Isolating machine by DNS attacks	• Disruptive • Annoying • Prevent user from getting work done	Difficult to prevent
Authentication	• Impersonation of legitimate users • Data forgery	• Misrepresentation of user • Belief that false information is valid	Cryptographic techniques

- A number of 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.

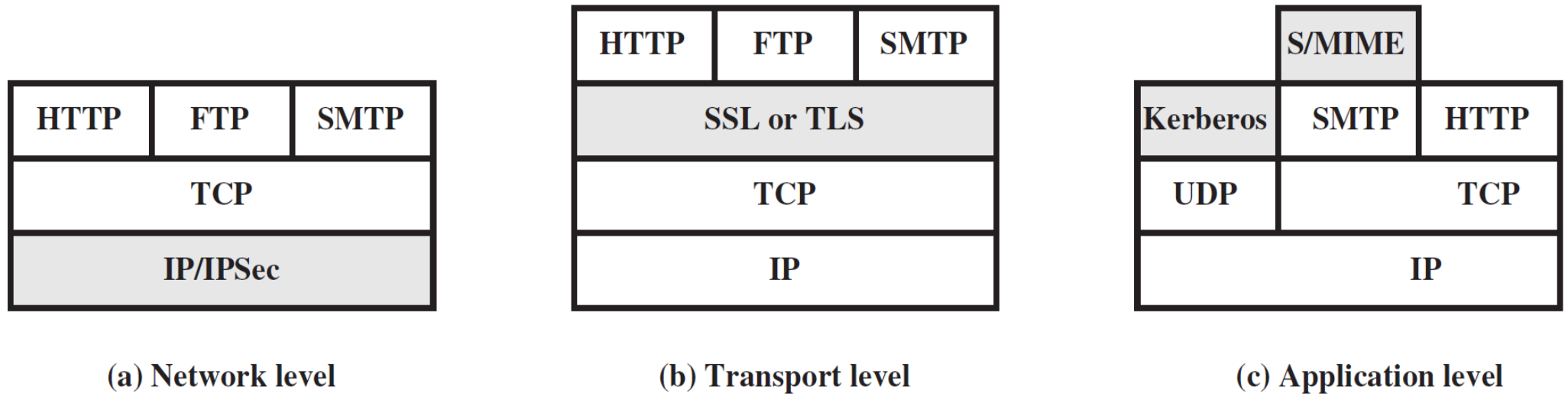


Figure 17.1 Relative Location of Security Facilities in the TCP/IP Protocol Stack

- Figure 17.1 illustrates this difference. One way to provide Web security is to use IP security (IPsec) (Figure 17.1a). 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.

- Another relatively general-purpose solution is to implement security just above TCP (Figure 17.1b). 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 17.1c 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.

- One of the most widely used security services is the Secure Sockets Layer (SSL) and the follow-on Internet standard known as Transport Layer Security (TLS), the latter defined in RFC 5246. SSL is a general-purpose service implemented as a set of protocols that rely on TCP. 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, most browsers come equipped with SSL, and most Web servers have implemented the protocol.

- 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 17.2.
- 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.

SSL Architecture

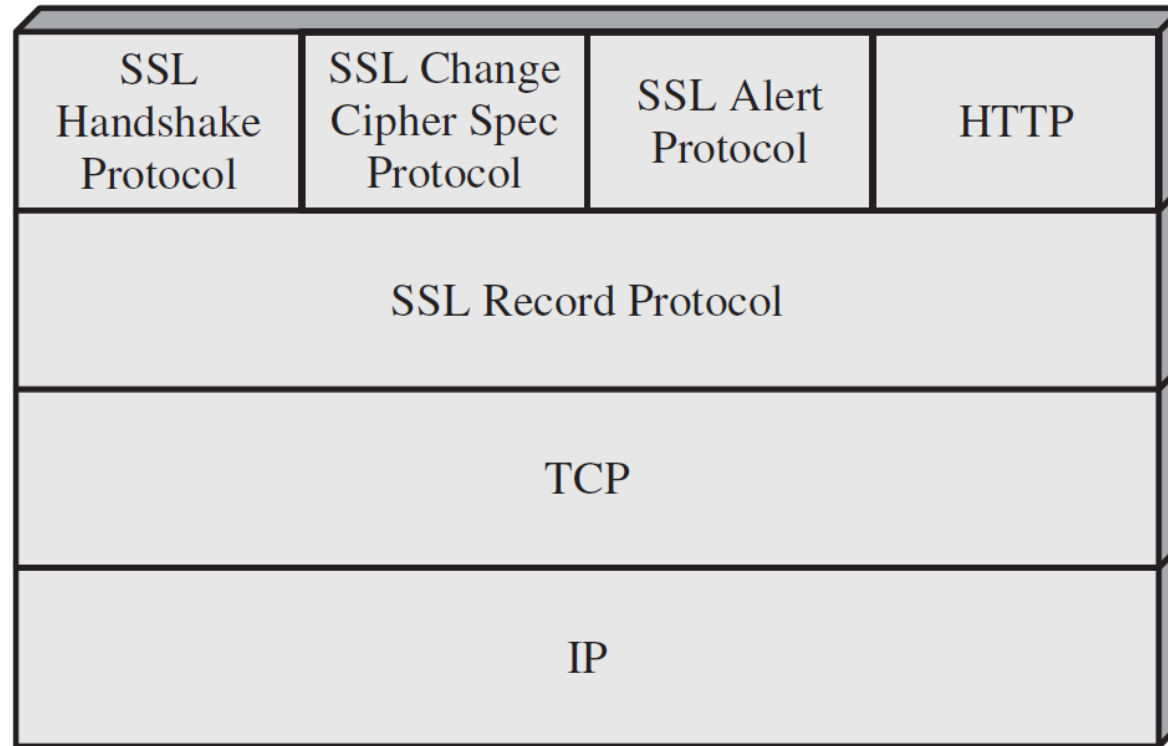
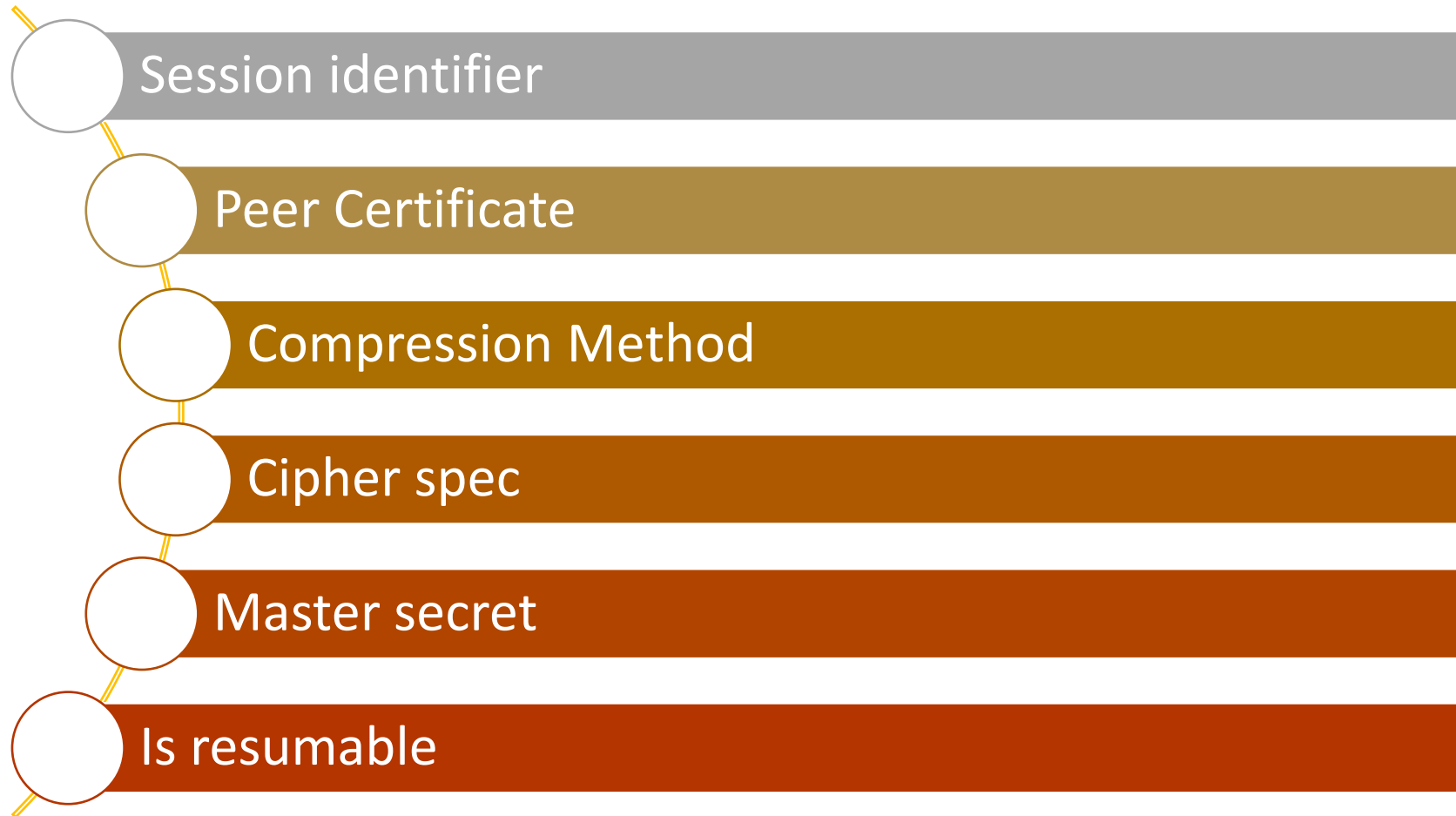
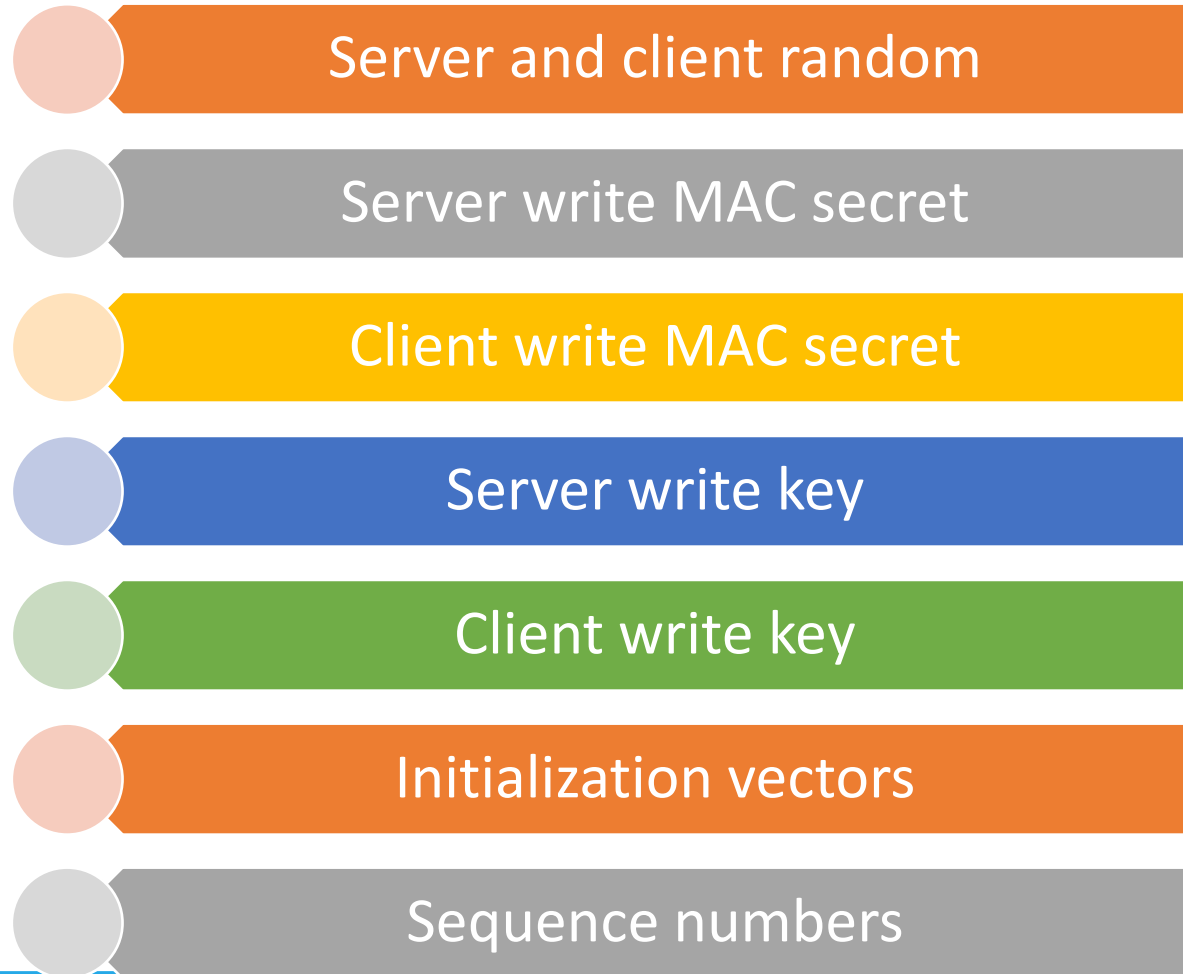


Figure 17.2 SSL Protocol Stack

SSL Architecture (Session State)



SSL Architecture (Connection State)



- The SSL Record Protocol provides two services for SSL connections:
- **Confidentiality:** The Handshake Protocol defines a shared secret key that is used for conventional encryption of SSL payloads.
- **Message Integrity:** The Handshake Protocol also defines a shared secret key that is used to form a message authentication code (MAC).

SSL Record Protocol

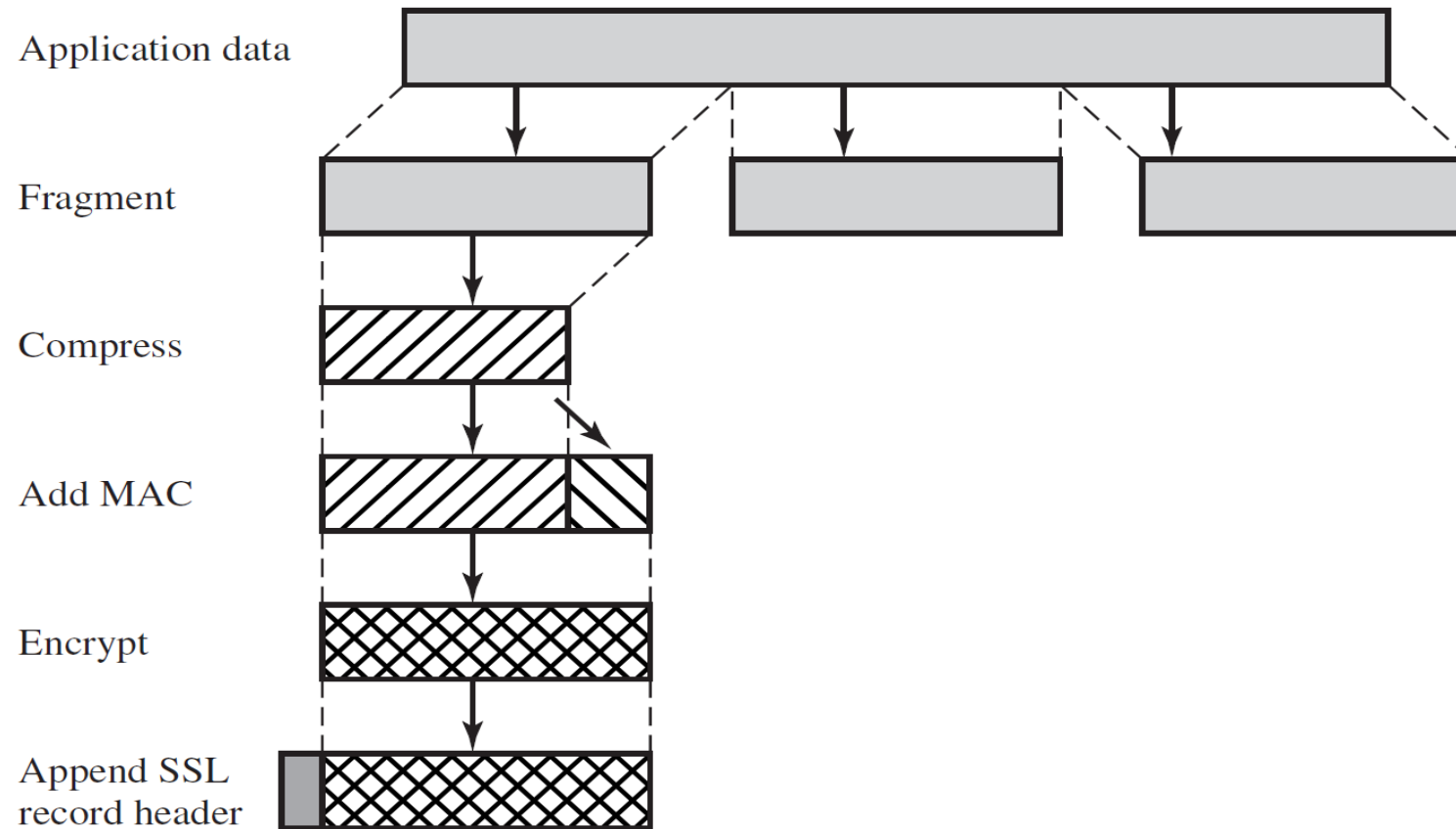


Figure 17.3 SSL Record Protocol Operation

- Figure 17.3 indicates the overall operation of the SSL Record Protocol. The Record Protocol takes an application message to be transmitted, fragments the data into manageable blocks, optionally compresses the data, applies a MAC, encrypts, adds a header, and transmits the resulting unit in a TCP segment. Received data are decrypted, verified, decompressed, and reassembled before being delivered to higher-level users.

- The first step is fragmentation. Each upper-layer message is fragmented into blocks of 214 bytes (16384 bytes) or less. Next, compression is optionally applied. Compression must be lossless and may not increase the content length by more than 1024 bytes.¹ In SSLv3 (as well as the current version of TLS), no compression algorithm is specified, so the default compression algorithm is null.

- The next step in processing is to compute a message authentication code over the compressed data. For this purpose, a shared secret key is used.
- Next, the compressed message plus the MAC are encrypted using symmetric encryption. Encryption may not increase the content length by more than 1024 bytes, so that the total length may not exceed $2^{14} + 2048$.

- The final step of SSL Record Protocol processing is to prepare a header consisting of the following fields:
- **Content Type (8 bits):** The higher-layer protocol used to process the enclosed fragment.
- **Major Version (8 bits):** Indicates major version of SSL in use. For SSLv3, the value is 3.

- **Minor Version (8 bits):** Indicates minor version in use. For SSLv3, the value is 0.
- **Compressed Length (16 bits):** The length in bytes of the plaintext fragment (or compressed fragment if compression is used). The maximum value is $2^{14} + 2048$.

- The Change Cipher Spec Protocol is one of the three SSL-specific protocols that use the SSL Record Protocol, and it is the simplest. This protocol consists of a single message (Figure 17.5a), which consists of a single byte with the value 1. The sole purpose of this message is to cause the pending state to be copied into the current state, which updates the cipher suite to be used on this connection.

- The Alert Protocol is used to convey SSL-related alerts to the peer entity. As with other applications that use SSL, alert messages are compressed and encrypted, as specified by the current state.

Alert Protocol

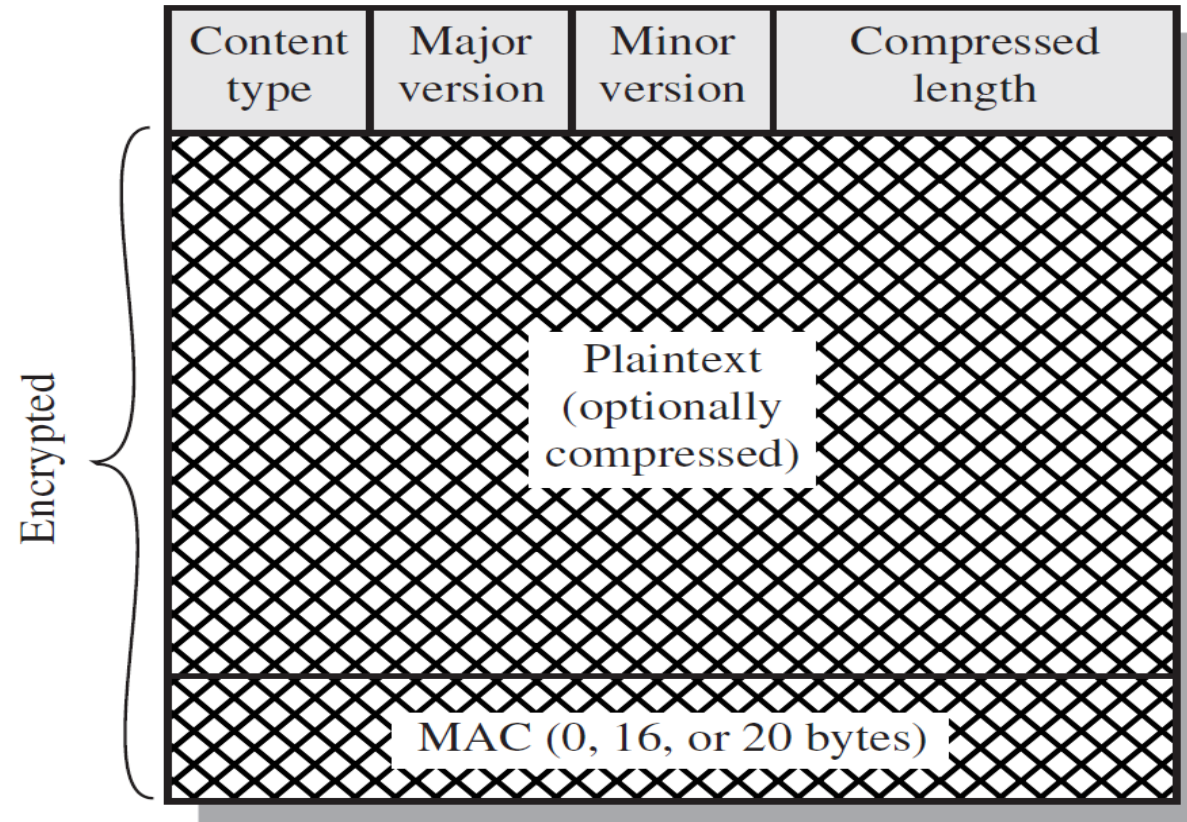


Figure 17.4 SSL Record Format

Alert Protocol

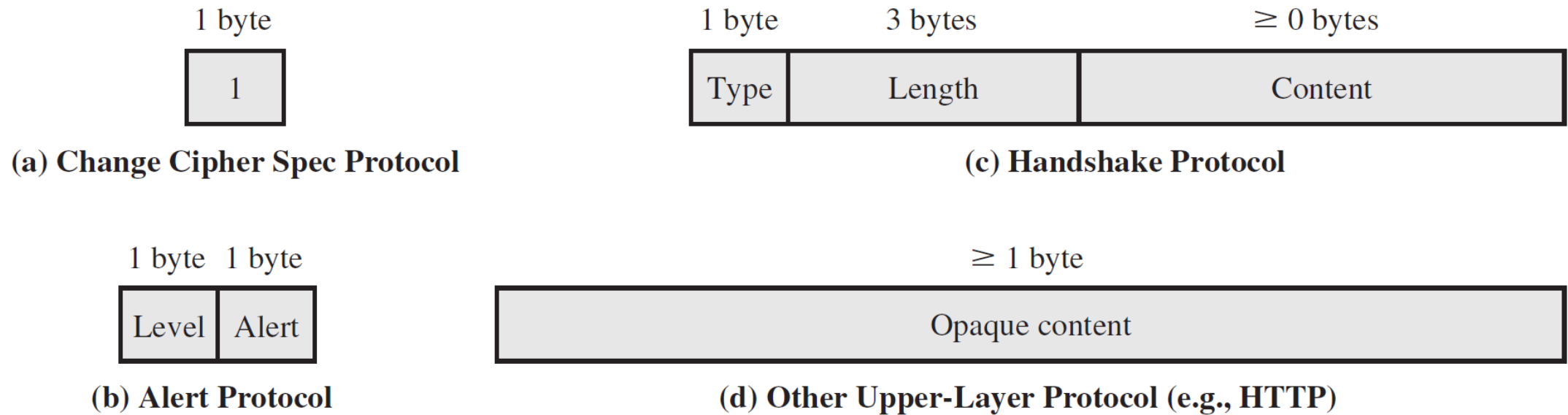


Figure 17.5 SSL Record Protocol Payload

- Each message in this protocol consists of two bytes (Figure 17.5b).

The first byte takes the value warning (1) or fatal (2) to convey the severity of the message. If the level is fatal, SSL immediately terminates the connection.

- Other connections on the same session may continue, but no new connections on this session may be established. The second byte contains a code that indicates the specific alert. First, we list those alerts that are always fatal (definitions from the SSL specification):

- **unexpected_message**: An inappropriate message was received.
- **bad_record_mac**: An incorrect MAC was received.
- **decompression_failure**: The decompression function received improper input (e.g., unable to decompress or decompress to greater than maximum allowable length).

- **handshake_failure:** Sender was unable to negotiate an acceptable set of security parameters given the options available.
- **illegal_parameter:** A field in a handshake message was out of range or inconsistent with other fields.

Handshake Protocol

- The most complex part of SSL is the Handshake Protocol. This protocol allows the server and client to authenticate each other and to negotiate an encryption and MAC algorithm and cryptographic keys to be used to protect data sent in an SSL record. The Handshake Protocol is used before any application data is transmitted.

Handshake Protocol

- The Handshake Protocol consists of a series of messages exchanged by client and server. All of these have the format shown in Figure 17.5c. Each message has three fields:
- **Type (1 byte):** Indicates one of ten messages. Table 17.2 lists the defined message types.
- **Length (3 bytes):** The length of the message in bytes.
- **Content (# 0 bytes):** The parameters associated with this message; these are listed in Table 17.2.

Handshake Protocol

Table 17.2 SSL Handshake Protocol Message Types

Message Type	Parameters
hello_request	null
client_hello	version, random, session id, cipher suite, compression method
server_hello	version, random, session id, cipher suite, compression method
certificate	chain of X.509v3 certificates
server_key_exchange	parameters, signature
certificate_request	type, authorities
server_done	null
certificate_verify	signature
client_key_exchange	parameters, signature
finished	hash value

Handshake Protocol

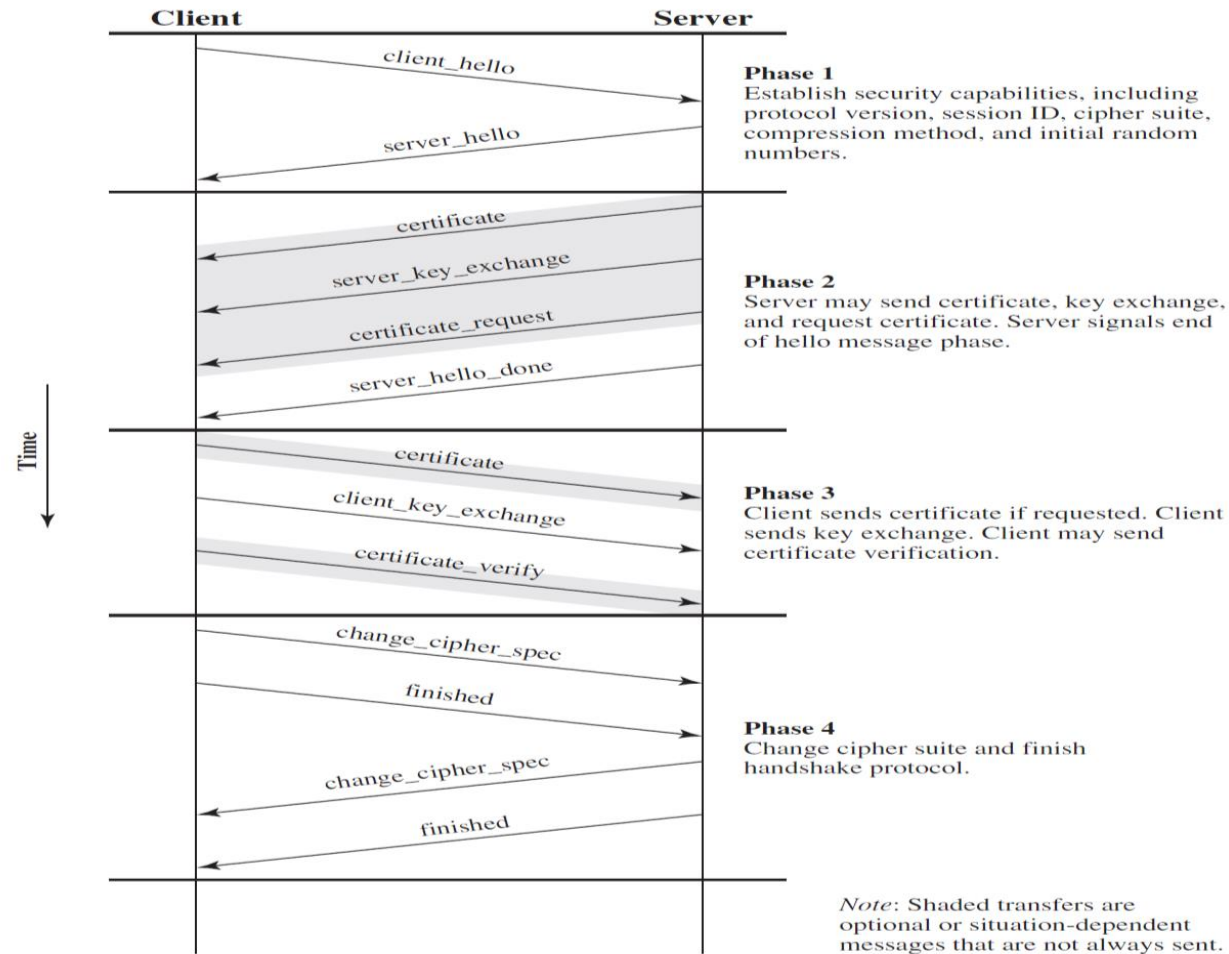


Figure 17.6 Handshake Protocol Action

Thank You

ANY
QUESTIONS?

