CMR INSTITUTE OF TECHNOLOGY

USN | | | | | | | | | | |

Internal Assesment Test – I

Sub:	Network and Cyber Security		Sec	*ALL*		Code:	15EC835
Date:	07 / 03 / 2019	Duration:	90 mins	Max Marks:	50	Sem: VIII	Branch: ECE

Scheme of Evaluation

Marks	OBE	
	CO	RBT

1 Write a note on Web security considerations and threats associated with it. — 10 | CO1 | L2

The World Wide Web is fundamentally a client/server application running over the Internet and TCP/IP intranets. The following characteristics of Web usage suggest the need for tailored security tools: — *05*

1. 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 and may be are vulnerable to a variety of security attacks.
2. 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.
3. 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

Threats — *05*

	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

2 List and briefly define the parameters that define an SSL Connection state and Session. — 10 | CO1 | L2

Connection state parameters — *05*

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

3. Client write MAC secret: The secret key used in MAC operations on data sent by the client.
4. Server write key: The secret encryption key for data encrypted by the server and decrypted by the client.
5. Client write key: The symmetric encryption key for data encrypted by the client and decrypted by the server.
6. 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.
7. 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 $2^{64} - 1$.

Session state parameters: **05**

1. Session identifier: An arbitrary byte sequence chosen by the server to identify an active or resumable session state.
2. Peer certificate: An X509.v3 certificate of the peer. This element of the state may be null.
3. Compression method: The algorithm used to compress data prior to encryption.
4. 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.
5. Master secret: 48-byte secret shared between the client and the server.
6. Is resumable: A flag indicating whether the session can be used to initiate new connections.

3 What services are provided by the ssl record protocol? Explain with the help of a neat diagram ssl record protocol operation. | 10 | CO1 | L3

Two Services **1X2=02**

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).

Diagram of SSL Record Protocol Operation **05**

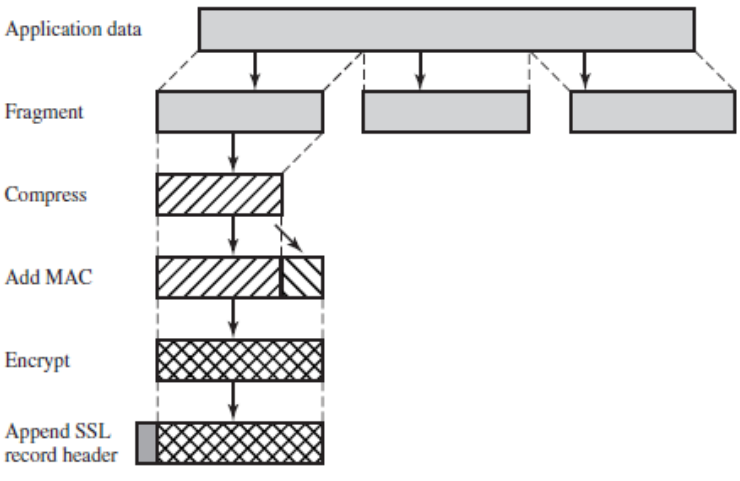


Figure 17.3 SSL Record Protocol Operation

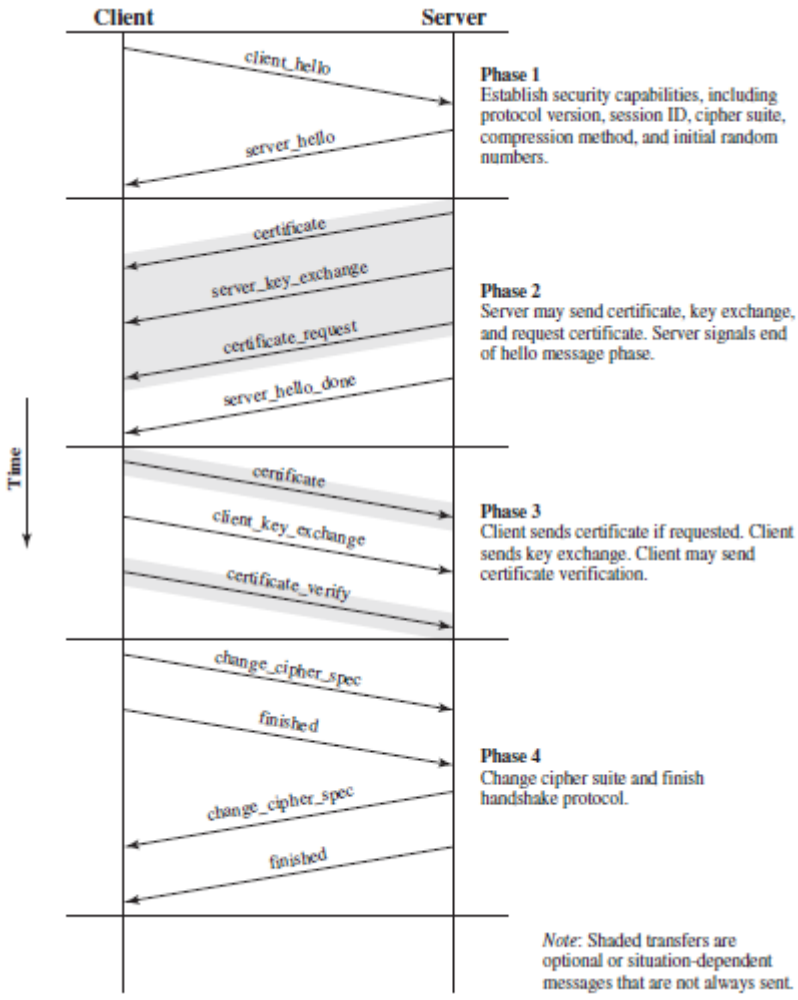
- Explanation for the same
- The first step is fragmentation. Each upper-layer message is fragmented into blocks of 214 bytes (16384 bytes) or less.
- Compression is optionally applied. Compression must be lossless and may not increase the content length by more than 1024 bytes.
- The next step in processing is to compute a message authentication code over the compressed data. For this purpose, a shared secret key is used. The calculation is defined as hash(MAC_write_secret || pad_2 ||
  hash(MAC_write_secret || pad_1 || seq_num ||
  SSLCompressed.type || SSLCompressed.length ||
  SSLCompressed.fragment))

  where
  || = concatenation
  MAC_write_secret = shared secret key
  hash = cryptographic hash algorithm; either MD5 or SHA-1.
  pad_1 = the byte 0x36 repeated 48 times for MD5 and 40 times for SHA-1
  pad_2 = the byte 0x5C repeated 48 times for MD5 and 40 times for SHA-1
  seq_num = the sequence number for this message
  SSLCompressed.type = the higher-level protocol used to process this fragment
  SSLCompressed.length = the length of the compressed fragment
  SSLCompressed.fragment = the compressed fragment (if compression is not used, this is the plaintext fragment)

03

4 With a neat diagram the Handshake Protocol Action. | 10 | CO1 | L3

Diagram the Handshake Protocol Action. — 05



Explanation for the same — 05

Phase 1. Establish Security Capabilities: used to initiate a logical connection and to establish the security capabilities that will be associated with it. The exchange is initiated by the client, which sends a client hello message with the following parameters: Version, Random, Session ID, Cipher suite, Compression method and initial random number.

Phase 2. Server Authentication and Key Exchange : The server begins this Phase by sending its certificate if it needs to be authenticated; the message contains one or a chain of X.509 certificates.

A **server_key_exchange message** may be sent, it is not required in two instances

(1) The server has sent a certificate with fixed Diffie-Hellman parameters

(2) A RSA key exchange is to be used.

A **Certificate request message** includes two parameters: Certificate type and Certificate authority.

The final message in phase 2 is server done message.

Phase 3. Client Authentication and Key Exchange: If the server has requested a certificate, the client begins this phase by sending a certificate message.

Next is the client_key_exchange message, which must be sent in this phase. Finally, in this phase, the client may send a certificate_verify message to provide explicit verification of a client certificate.

Phase 4. Finish : This phase completes the setting up of a secure connection. The client sends a change_cipher_spec message and copies the pending CipherSpec into the current CipherSpec. The client then immediately sends the finished message under the new algorithms, keys, and secrets. The finished message verifies that the key exchange and authentication processes were successful

5 What is the of HTTPS? Breifly explain Connection Initiation and Closure. — 10 | CO1 | L2

Breif explanation of HTTPS — 02

It refers to the combination of HTTP and SSL to implement a secur conncection between a web broweser and a web server. The principal difference seen by a user of a Web browser is that URL addresses begin with https rather than http. HTTPS uses port number 443, whereas HTTP uses port number 80.

Connection Initiation — 04

For HTTPS, the agent acting as the HTTP client also acts as the TLS client. The client initiates a connection to the server on the appropriate port and then sends the TLS ClientHello to begin the TLS handshake. When the TLS handshake has finished, the client may then initiate the first HTTP request. All HTTP data is to be sent as TLS application data. Normal HTTP behavior, including retained connections, should be followed.

There are three levels of awareness of a connection in HTTPS.

1. At the HTTP level, an HTTP client requests a connection to an HTTP server by sending a connection request to the next lowest layer.

2. Typically, the next lowest layer is TCP, but it also may be TLS/SSL. At the level of TLS, a session is established between a TLS client and a TLS server.
3. TLS request to establish a connection begins with the establishment of a TCP connection between the TCP entity on the client side and the TCP entity on the server side.

Connection Closure 04

An HTTP client or server can indicate the closing of a connection by including the following line in an HTTP record: Connection: close. This indicates that the connection will be closed after this record is delivered.

The closure of an HTTPS connection requires that TLS close the connection with the peer TLS entity on the remote side, which will involve closing the underlying TCP connection. At the TLS level, the proper way to close a connection is for each side to use the TLS alert protocol to send a **close_notify** alert. TLS implementations must initiate an exchange of closure alerts before closing a connection.

A TLS implementation may, after sending a closure alert, close the connection without waiting for the peer to send its closure alert, generating an "incomplete close".

HTTP clients also must be able to cope with a situation in which the underlying TCP connection is terminated without a prior close notify alert and without a Connection: close indicator. Such a situation could be due to a programming error on the server or a communication error that causes the TCP connection to drop. However, the unannounced TCP closure could be evidence of some sort of attack. So the HTTPS client should issue some sort of security warning when this occurs.

6 Explain with the help of neat diagram the SSH Transport Layer Protocol Packet Exchange. 10 CO1 L2

Diagram the SSH Transport Layer Protocol Packet Exchange 05

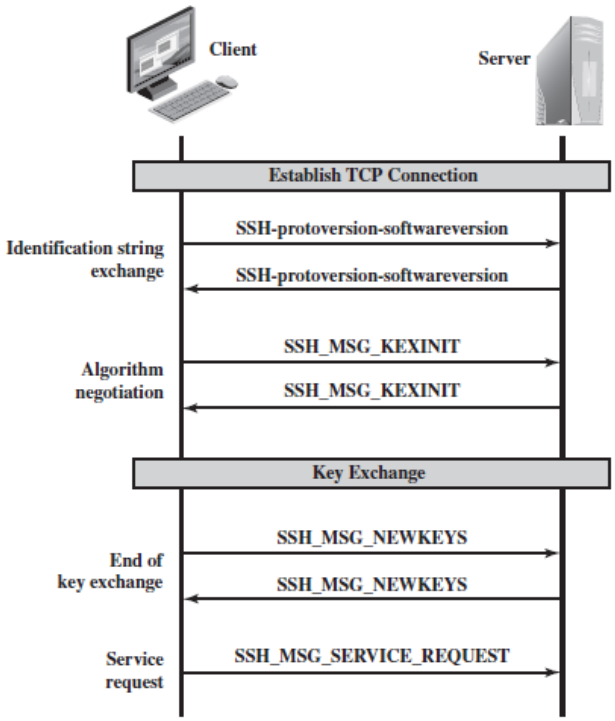


Figure 17.9 SSH Transport Layer Protocol Packet Exchanges

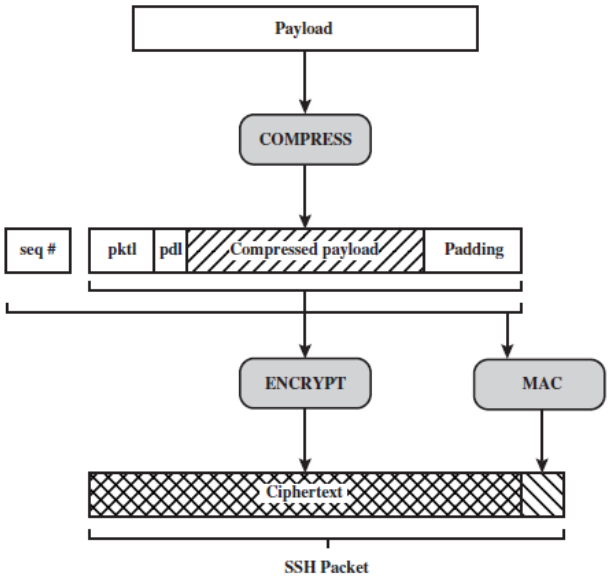


Figure 17.10 SSH Transport Layer Protocol Packet Formation

Explanation for the same 03

Fig 17.9 above illustrates the the sequence of events in the SSH Transport Layer Protocol. First, the client establishes a TCP connection to the server. This is done via the TCP protocol and is not part of the Transport Layer Protocol. Once the connection is established, the client and server exchange data referred to as packets, in the data field of a TCP segment.

Each packet is in the format shown in figure 17.10 and contains the following format

- Packet Length
- Padding Length
- Payload
- Random padding

MAC value is computed over the entire packet plus the sequence number. Once an encryption algorithm has been negotiated, the entire packet is encrypted after the MAC value is calculated.

First step: Identification string exchange

SSH-protoversion-softwareversion SP comments CR LF where SP, CR, and LF are space character, carriage return, and line feed, respectively.

Second step: algorithm negotiation:

Each side sends an SSH_MSG_KEXINIT containing lists of supported algorithms in the order of preference to the sender. There is one list for each type of cryptographic algorithm. The algorithms include key exchange, encryption, MAC algorithm, and compression algorithm.

Third step :Key exchange The specification allows for alternative methods of key exchange, but at present, only two versions of Diffie-Hellman key exchange are specified. The end of key exchange is signalled by the exchange of SSH_MSG_NEWKEYS packets.

The **final step is service request.** The client sends an SSH_MSG_SERVICE_ REQUEST packet to request either the User Authentication or the Connection Protocol. Subsequent to this, all data is exchanged as the payload of an SSH Transport Layer packet, protected by encryption and MAC.