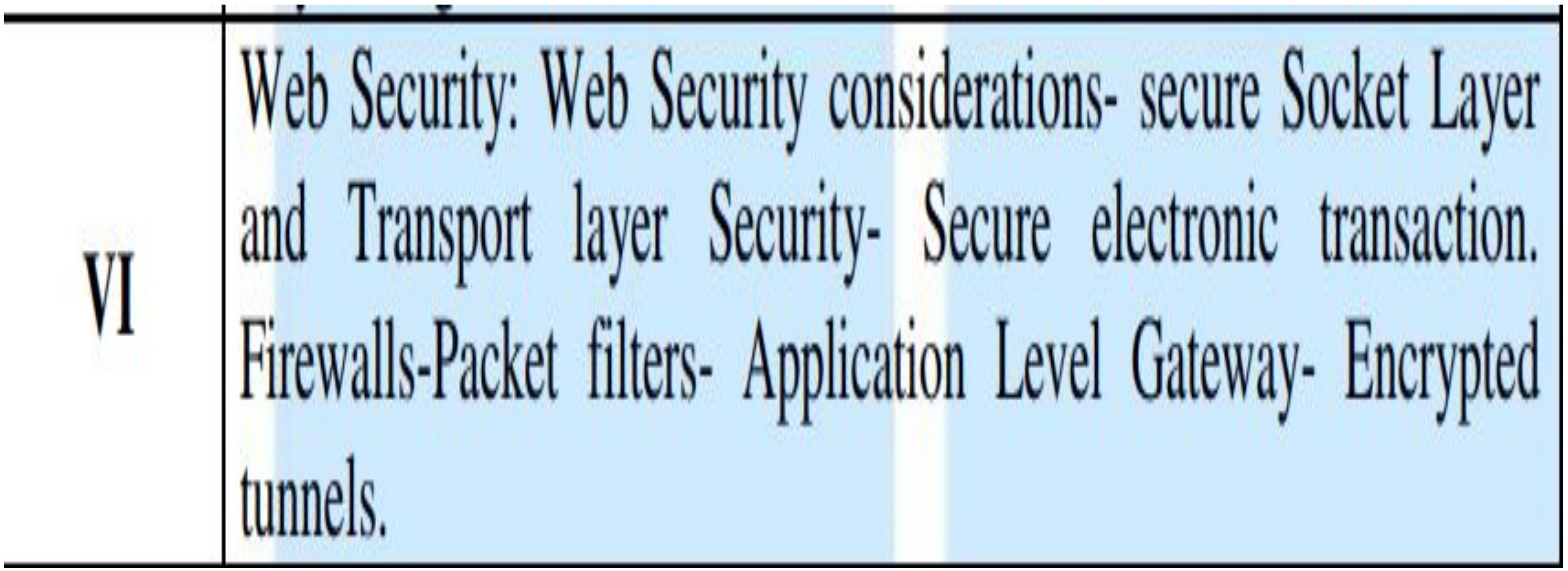


VI	Web Security: Web Security considerations- secure Socket Layer and Transport layer Security- Secure electronic transaction. Firewalls-Packet filters- Application Level Gateway- Encrypted tunnels.

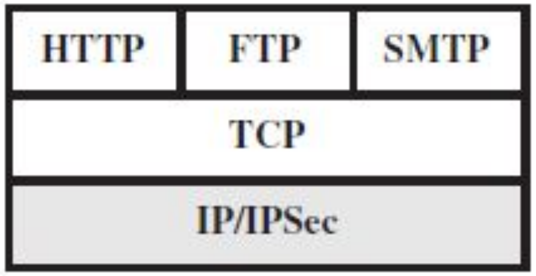
Web security considerations

World Wide Web

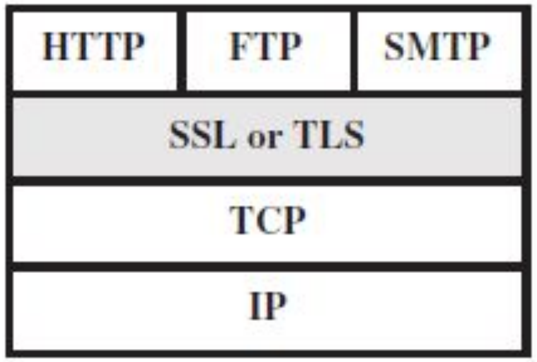
- A client/server application running over the Internet and TCP/IP intranets
- Are vulnerable to a variety of security attacks
  - integrity
  - confidentiality
  - denial of service
  - authentication
- Need added security mechanisms

	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

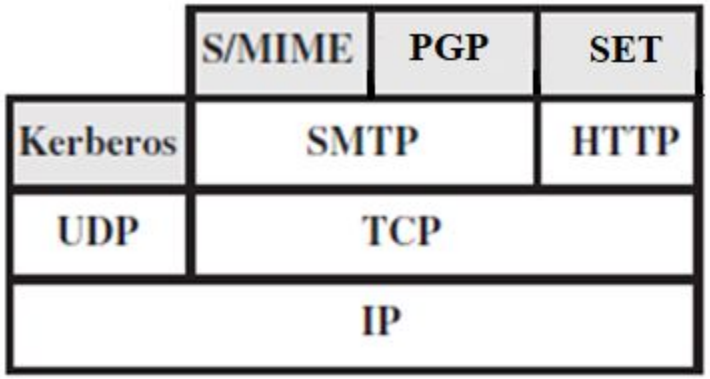
Web Traffic Security Approaches



(a) Network level



(b) Transport level



(c) Application level

Secure socket layer (SSL)

Secure Socket Layer (SSL)

- SSL is a general-purpose service implemented as a set of protocols that rely on TCP
- Transport layer security service, originally developed by Netscape
- Subsequently became internet standard known as TLS (transport layer security)
- Uses TCP to provide a reliable end-to-end service
- SSL has two layers of protocols

SSL Architecture

- SSL has two layers of protocols

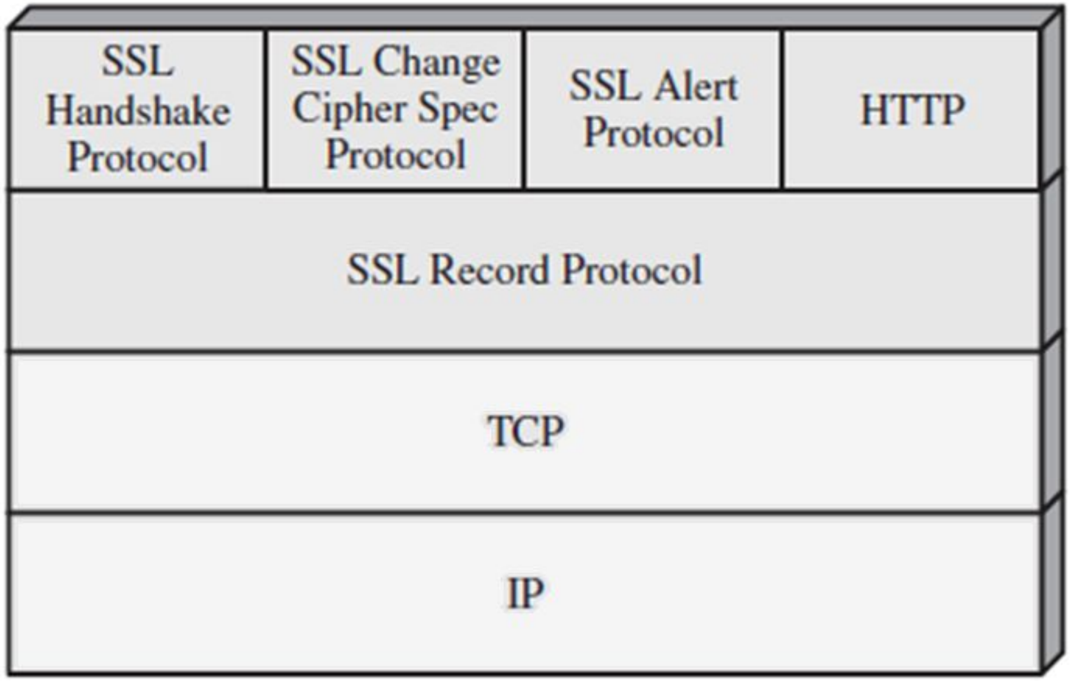


Figure 17.2 SSL Protocol Stack

SSL Concepts

- **SSL connection**
  - A transient, peer-to-peer, communications link that provides a suitable service
  - Associated with 1 SSL session

SSL Concepts

- **SSL session**
  - An association between client & server
  - Created by the handshake protocol
  - Define a set of cryptographic parameters
  - May be shared by multiple SSL connections

SSL Session state

- Session state defined by:
  - Session id
  - Peer certificate
  - Compression method
  - Cipher spec
  - Master secret
  - Is resumable

SSL Connection state

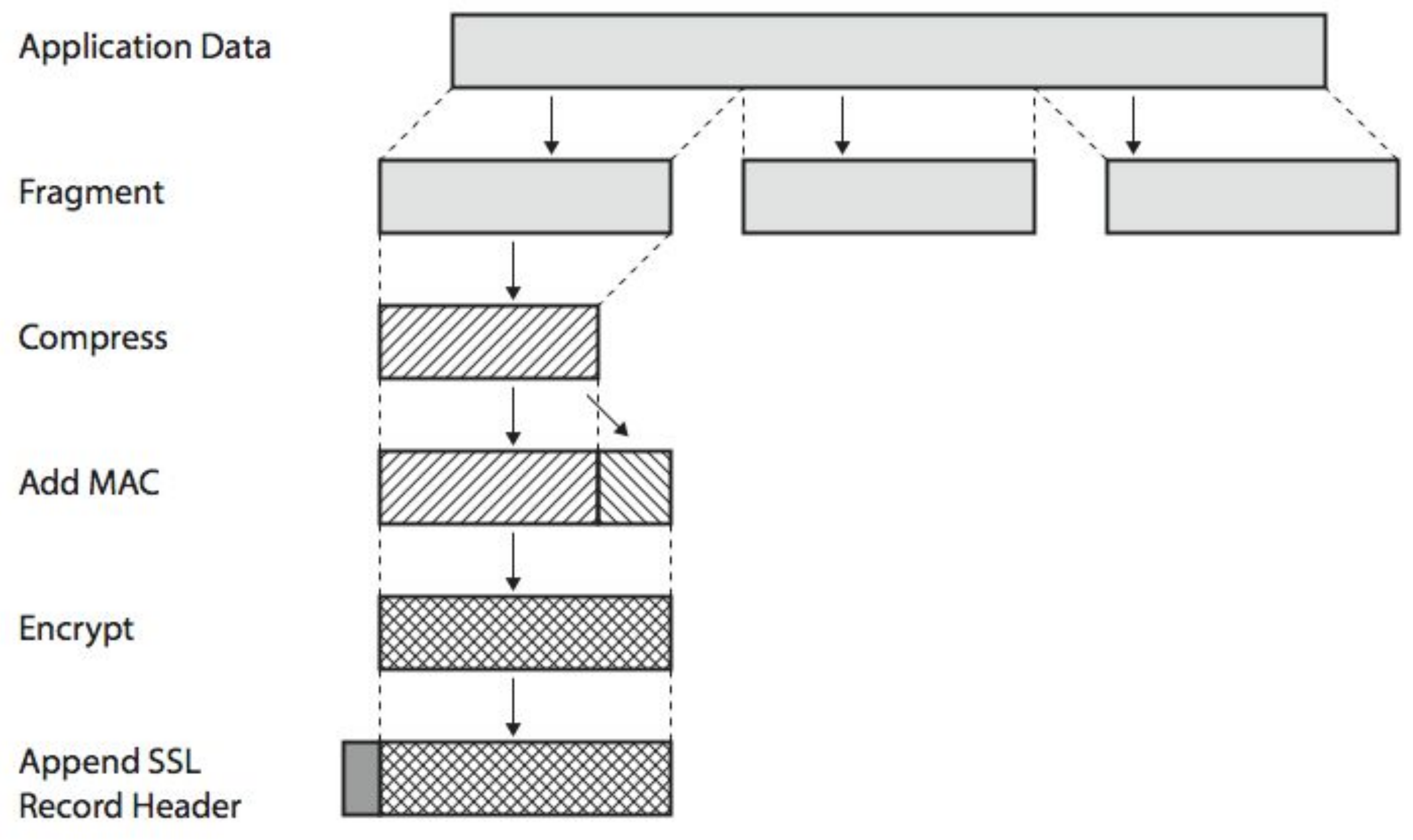
- Connection state identified by:
  - Server and client random
  - Server write MAC secret
  - Client write MAC secret
  - Server write key
  - Client write key
  - Initialization vectors
  - Sequence numbers

SSL Record Protocol

SSL Record Protocol provides two services:

- **Message integrity**
  - Using a MAC with shared secret key
- **Confidentiality**
  - Using symmetric encryption with a shared secret key defined by handshake protocol
  - AES, IDEA, RC2-40, DES-40, DES, 3DES, fortezza, RC4-40, RC4-128
  - Message is compressed before encryption

SSL Record Protocol Operation



- The record layer formats the upper layer protocol messages.
- It fragments the data into manageable blocks (max length 16 KB). It optionally compresses the data.
- Encrypts the data.
- Provides a header for each message and a hash (Message Authentication Code (MAC)) at the end.
- Hands over the formatted blocks to TCP layer for transmission

- MAC Computation:

hash(MAC_write_secret || pad_2 || hash(MAC_write_secret || pad_1 || seq_num || SSL Compressed.type || SSLCompressed.length || SSLCompressed.fragment))

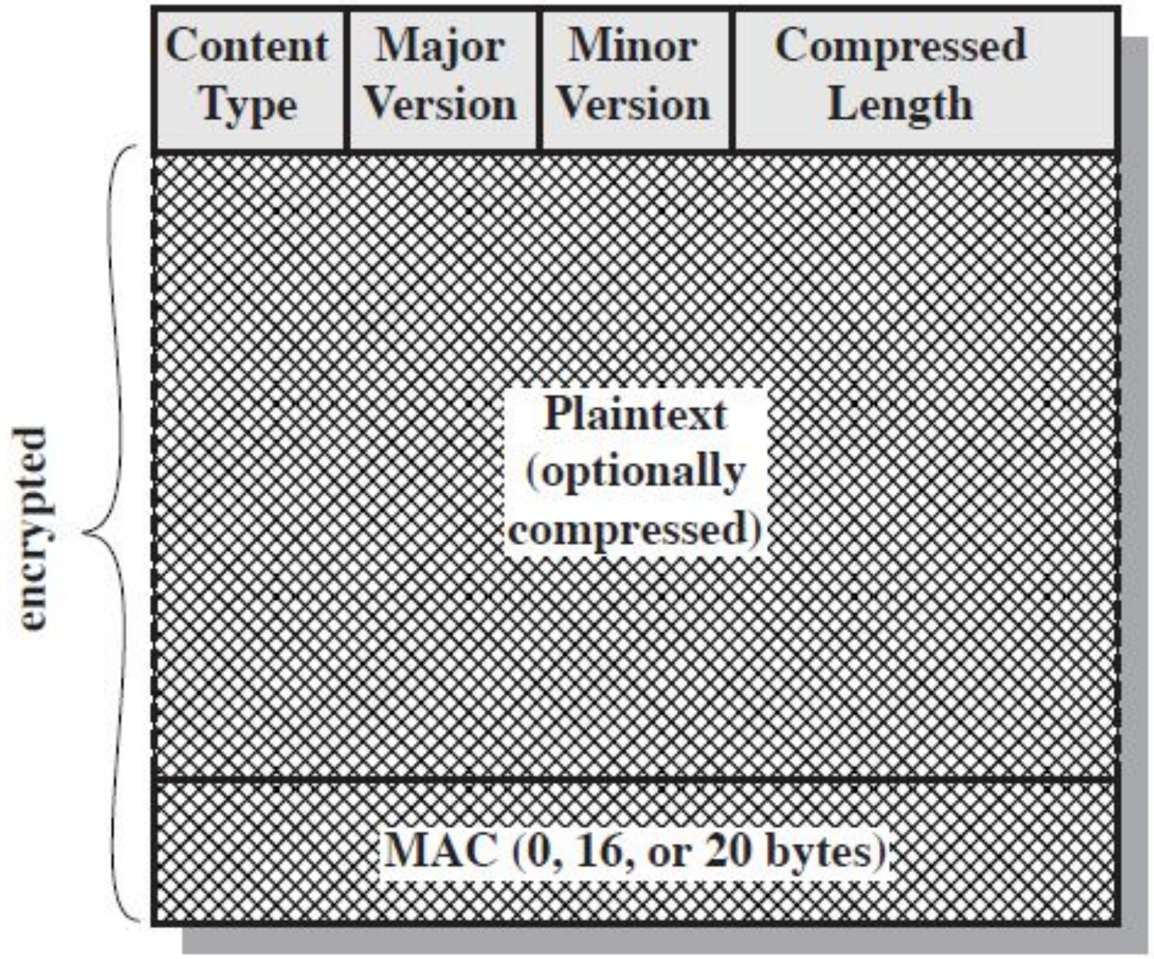
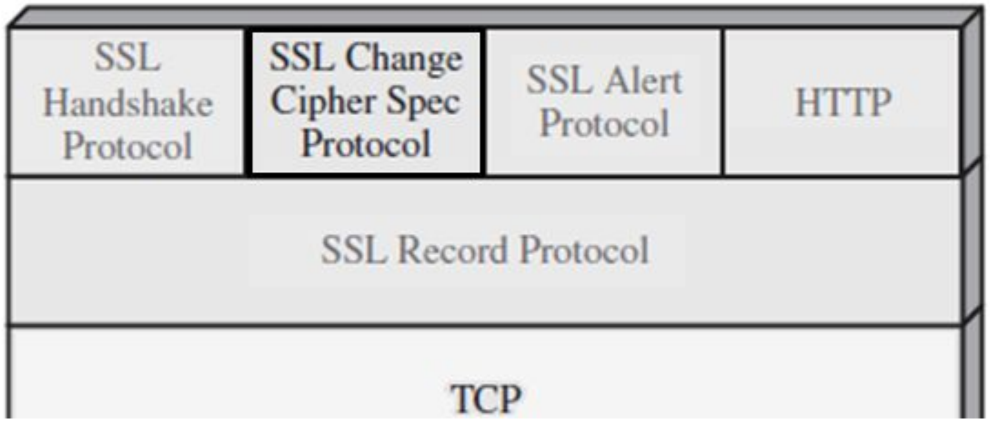


Figure 17.4 SSL Record Format

SSL Change Cipher Spec Protocol

- One of 3 SSL specific protocols which use the SSL record protocol
- A single message, single byte with value 1
- Causes pending state to become current
- Hence updating the cipher suite in use



- Simplest part of SSL protocol. It comprises of a single message exchanged between two communicating entities, the client and the server.
- As each entity sends the ChangeCipherSpec message, it changes its side of the connection into the secure state as agreed upon.
- The cipher parameters pending state is copied into the current state.
- Exchange of this Message indicates all future data exchanges are encrypted and integrity is protected.

SSL Alert Protocol

- Conveys SSL-related alerts to peer entity
- Severity
  - Warning or fatal
- Specific alert
  - Fatal: unexpected message, bad record mac, decompression failure, handshake failure, illegal parameter
  - Warning: close notify, no certificate, bad certificate, unsupported certificate, certificate revoked, certificate expired, certificate unknown
- Compressed & encrypted like all SSL data

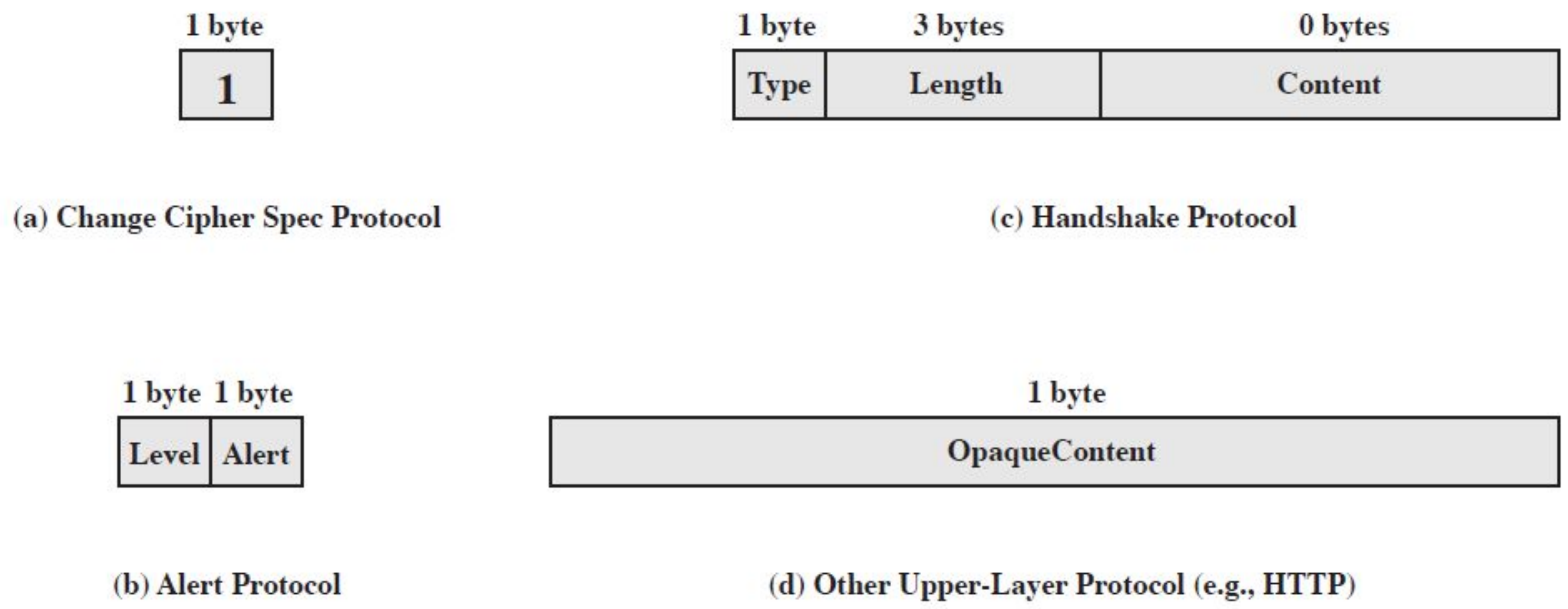


Figure 17.5 SSL Record Protocol Payload

•CHANGE CIPHER SPEC PROTOCOL

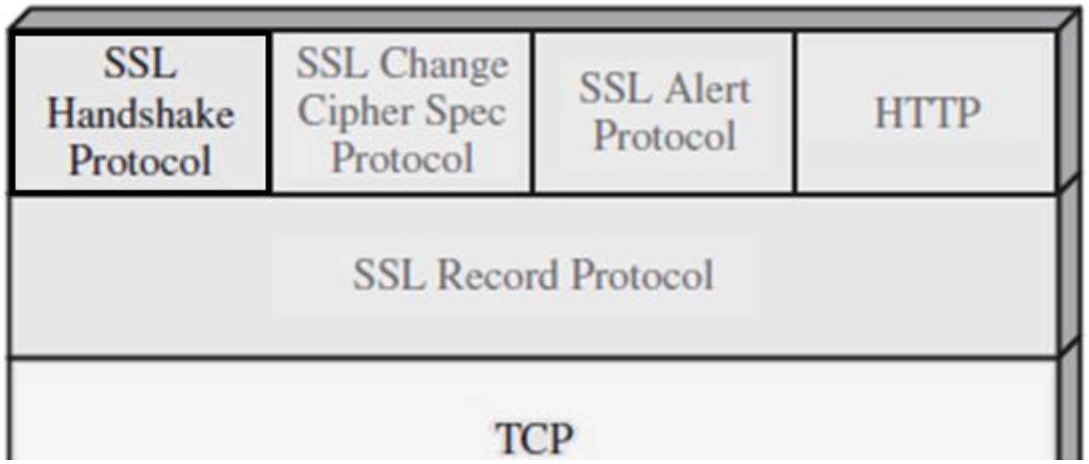
• The Change Cipher Spec Protocol is one of the three SSL-specific protocols that use the SSL Record Protocol, and it is the simplest.
• This protocol consists of a single message which consists of a single byte with the value 1.
• The purpose of this message is to cause the pending state to be copied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

•ALERT PROTOCOL

- The Alert Protocol is used to convey SSL-related alerts to the peer entity.
- Each message in this protocol consists of two bytes
- 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.

SSL Handshake Protocol

- Allows server & client to:
  - Authenticate each other
  - To negotiate encryption & MAC algorithms
  - To negotiate cryptographic keys to be used

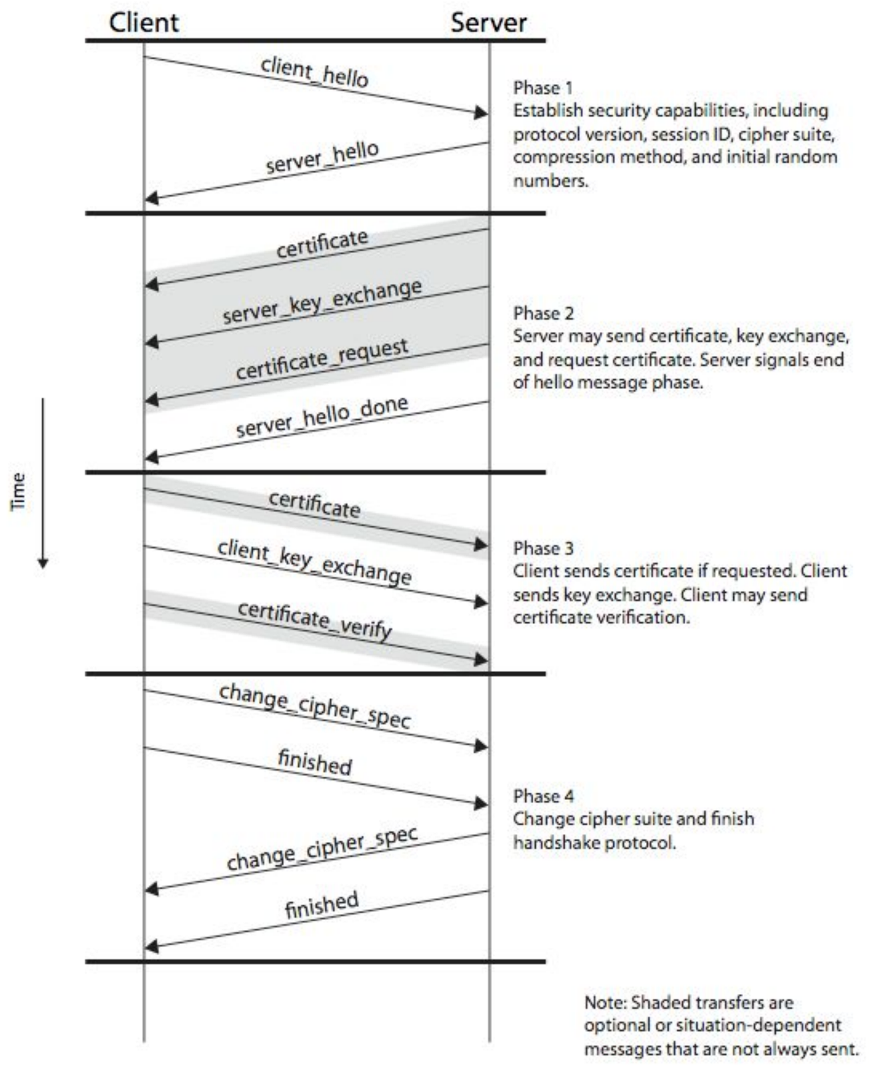


SSL Handshake Protocol

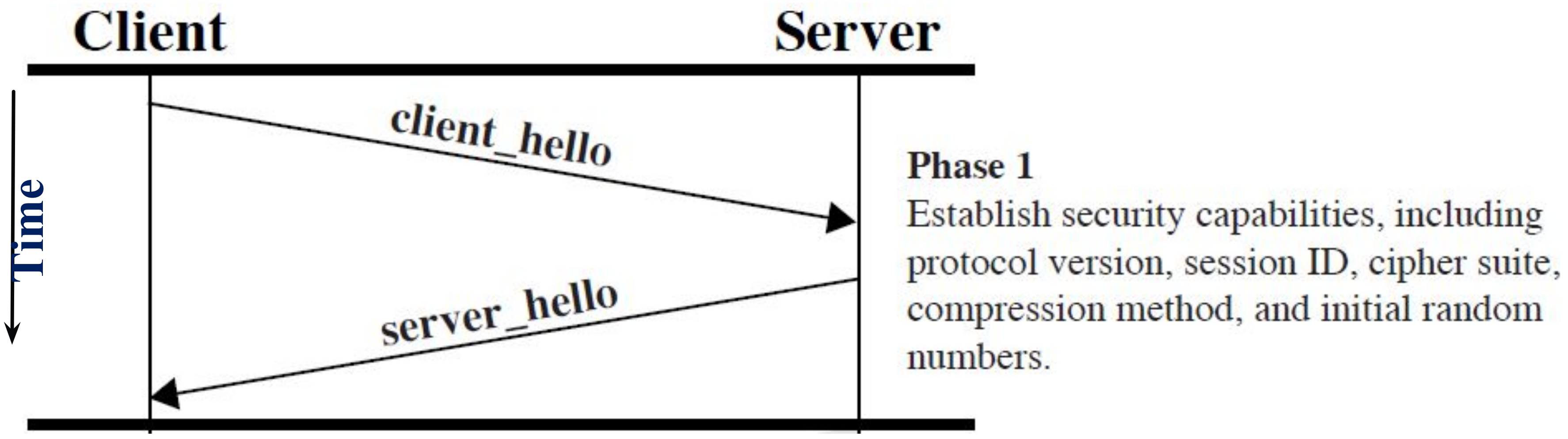
- Comprises a series of messages in phases

1. Establish security capabilities
2. Server authentication and key exchange
3. Client authentication and key exchange
4. Finish

SSL Handshake Protocol


Client
Server
client_hello
server_hello
Phase 1
Establish security capabilities, including protocol version, session ID, cipher suite, compression method, and initial random numbers.
certificate
server_key_exchange
certificate_request
server_hello_done
Phase 2
Server may send certificate, key exchange, and request certificate. Server signals end of hello message phase.
Time
certificate
client_key_exchange
certificate_verify
Phase 3
Client sends certificate if requested. Client sends key exchange. Client may send certificate verification.
change_cipher_spec
finished
change_cipher_spec
finished
Phase 4
Change cipher suite and finish handshake protocol.
Note: Shaded transfers are optional or situation-dependent messages that are not always sent.

Phase 1

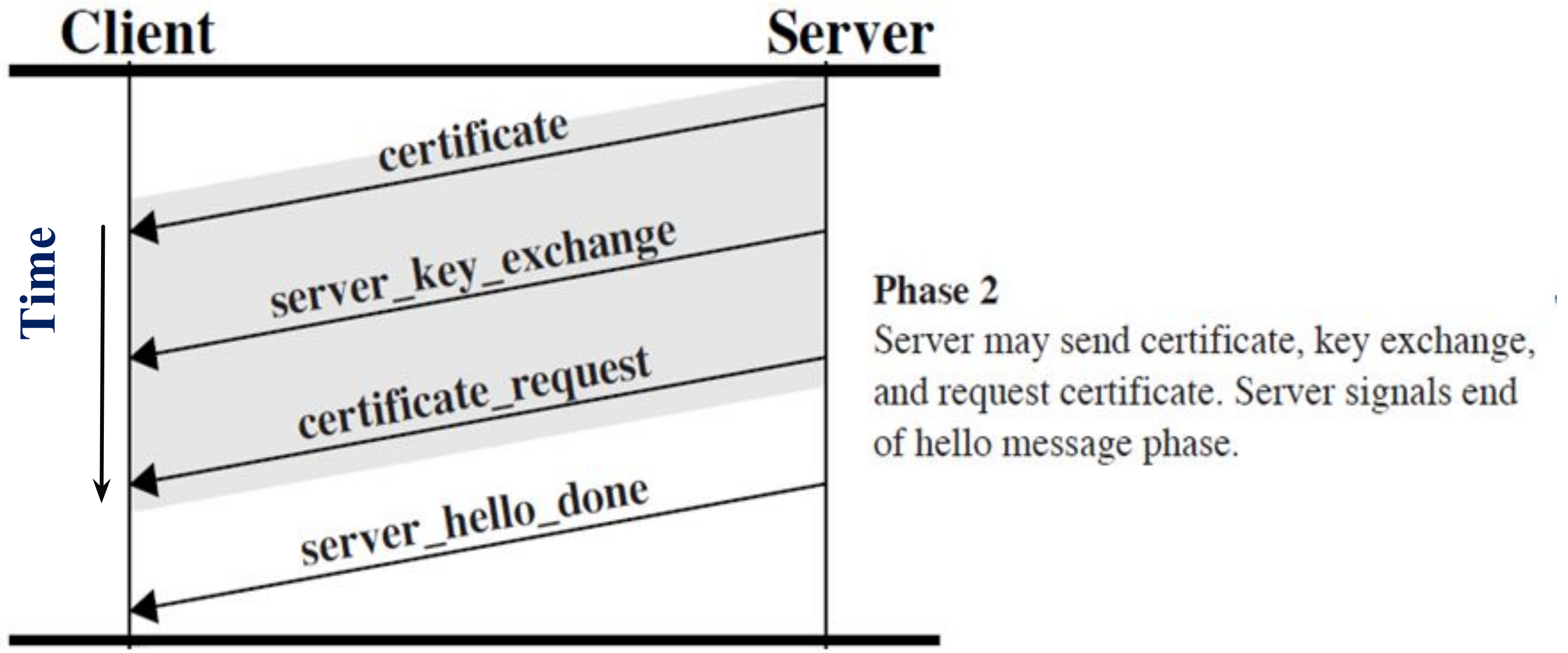

Client
Server
Time
client_hello
server_hello
Phase 1
Establish security capabilities, including protocol version, session ID, cipher suite, compression method, and initial random numbers.

- *Client_hello* contains of list of cryptographic algorithms supported by the client, in decreasing order of preference.
- *Server_hello* contains the selected Cipher Specification (CipherSpec) and a new *session_id*.

Phase 2

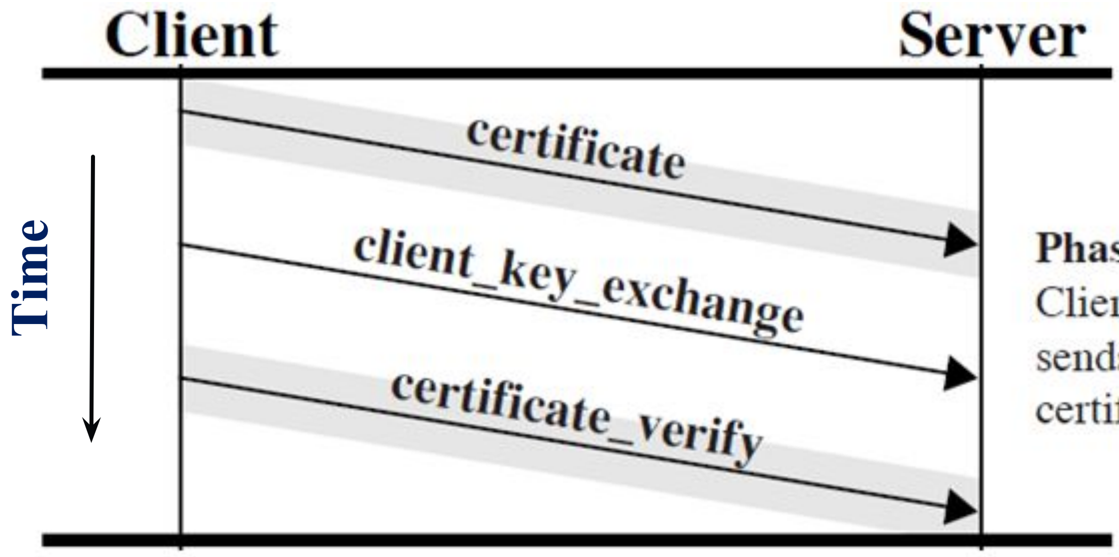
Server

Client


Client
Server
Time
certificate
server_key_exchange
certificate_request
server_hello_done
Phase 2
Server may send certificate, key exchange, and request certificate. Server signals end of hello message phase.

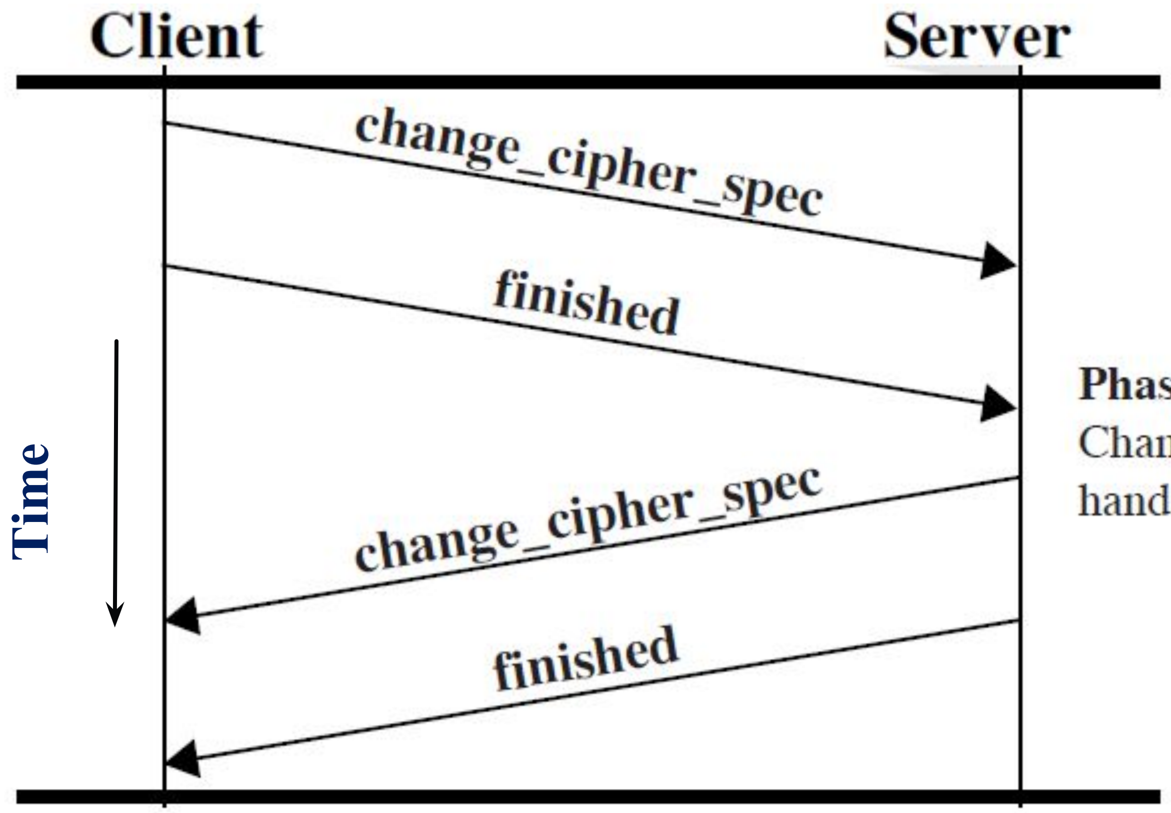
•Server sends certificate. Client software comes configured with public keys of various "trusted" organizations (CAs) to check certificate.
•Server sends chosen cipher suite.
•Server may request client certificate. Usually it is not done.
•Server indicates end of *Server_hello*.

Phase 3



Phase 3
Client sends certificate if requested. Client sends key exchange. Client may send certificate verification.

Phase 4



Phase 4
Change cipher suite and finish handshake protocol.

Transport Layer Security

TLS (Transport Layer Security)

The same record format as the SSL record format.

- IETF standard RFC 2246 similar to sslv3
- With minor differences
    - Version number
    - Uses HMAC for MAC
    - A pseudo-random function expands secrets
    - Has additional alert codes
    - Some changes in supported ciphers
    - Changes in certificate negotiations
    - Changes in use of padding

TLS MAC

$\mathrm{HMAC}_K(M) = \mathrm{H}[(K^+ \oplus \mathrm{opad}) || \mathrm{H}[(K^+ \oplus \mathrm{ipad}) || M]]$

where

- H = embedded hash function (either MD5 or SHA-1)
- *M = message input to HMAC*
- *K^+ = secret key padded with zeros on the left*
- ipad = 00110110 (36_H) repeated 64 times (512 bits)
- opad = 01011100 ($5C_H$) repeated 64 times (512 bits)

Pseudorandom Function

- PRF is based on P-hash(secret, seed) expansion function

P_hash(secret, seed) = HMAC_hash(secret,A(1) } seed) }
HMAC_hash(secret,A(2) } seed) }
HMAC_hash(secret,A(3) } seed) } ...

where `A()` is defined as

`A(0)` = seed

`A(i)` = HMAC_hash(secret, A(*i* - 1))

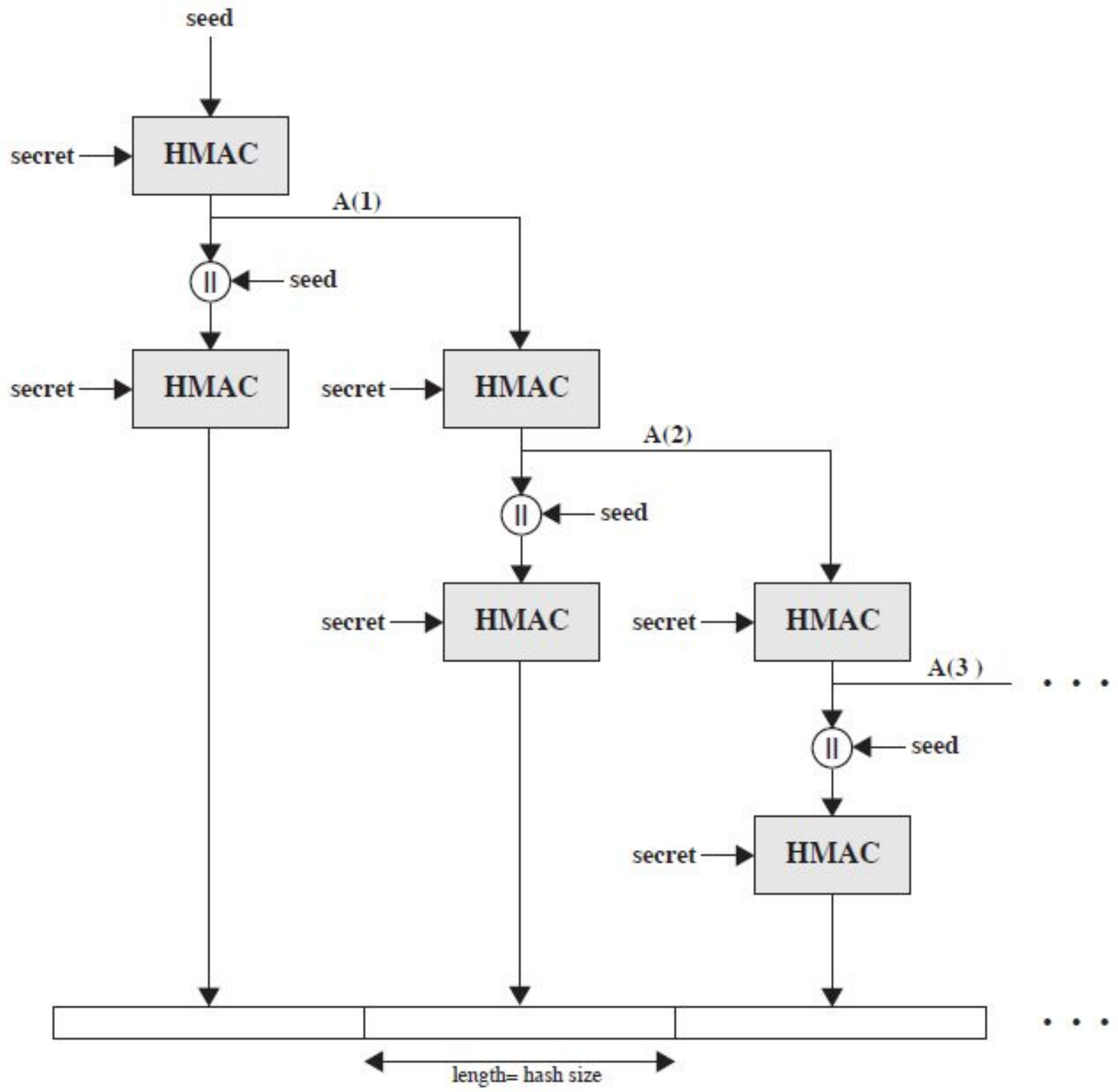


Figure 17.7 TLS Function P_hash (secret, seed)

Pseudorandom Function

- PRF is made secure by using two hash algorithms

$$PRF(secret, label, seed) = P\_MD5(S1, label \| seed) \oplus P\_SHA\text{-}1(S2, label \| seed)$$

Alert Codes

- Fatal: decryption_failed, record_overflow, unknown_ca, access_denied, decode_error, etc
- Warning: decrypt_error, user_cancelled, no_renegogiation