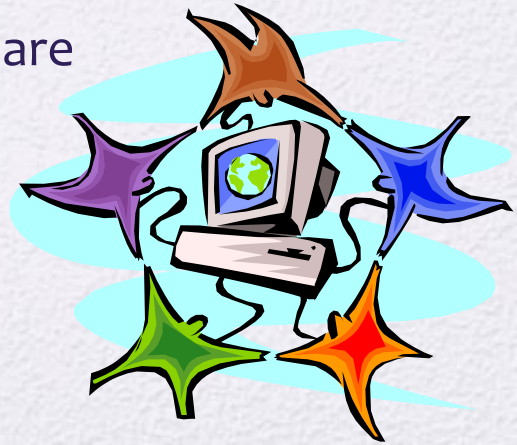
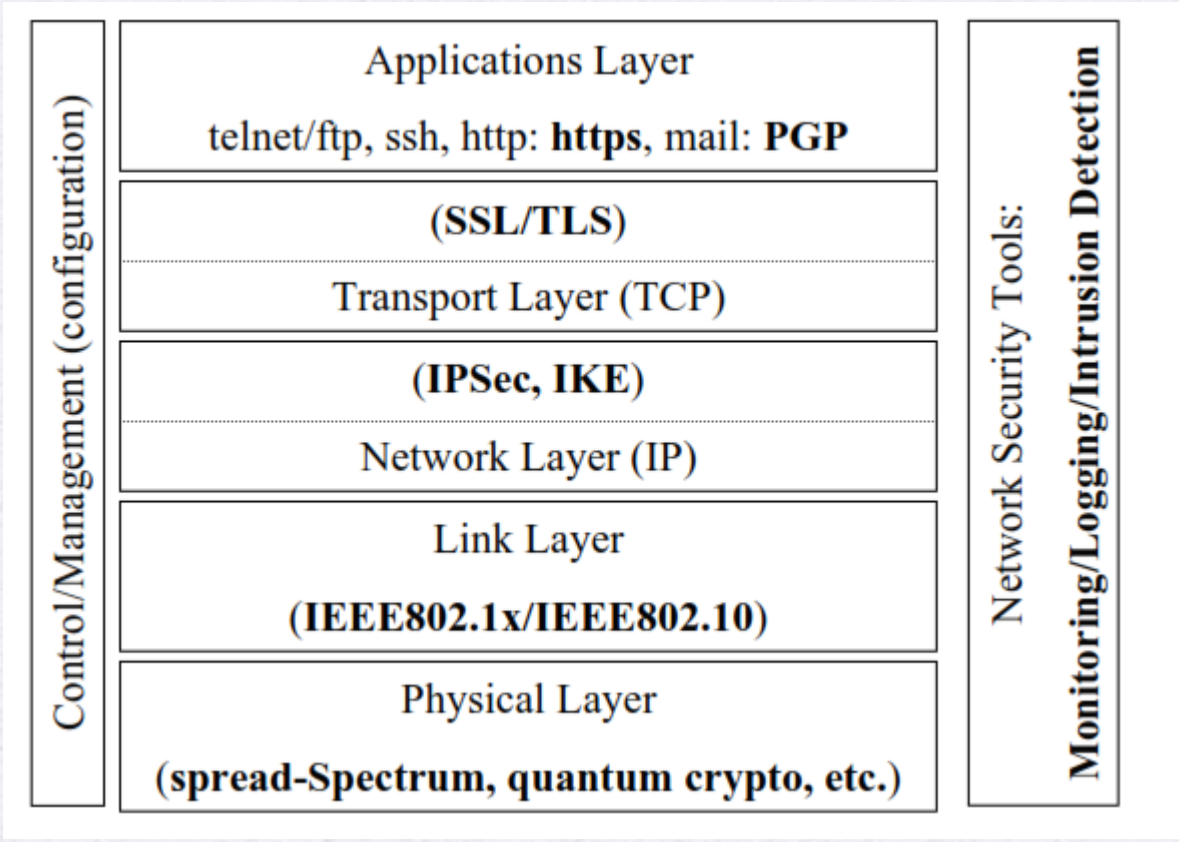
Web Security Considerations

- 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:
  - Web servers are relatively easy to configure and manage
  - Web content is increasingly easy to develop
  - The underlying software is extraordinarily complex
    - May hide many potential security flaws
  - A Web server can be exploited as a launching pad into the corporation's or agency's entire computer complex
  - 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




Control/Management (configuration)
Applications Layer
telnet/ftp, ssh, http: **https**, mail: **PGP**
(SSL/TLS)
Transport Layer (TCP)
(IPSec, IKE)
Network Layer (IP)
Link Layer
(IEEE802.1x/IEEE802.10)
Physical Layer
(spread-Spectrum, quantum crypto, etc.)
Network Security Tools:
Monitoring/Logging/Intrusion Detection

- SSL:
  - Avoids modifying "TCP stack" and requires minimum changes to the application
  - Mostly used to authenticate servers
- IPsec
  - Transparent to the application and requires modification of the network stack
  - Authenticates network nodes and establishes a secure channel between nodes
  - Application still needs to authenticate the users

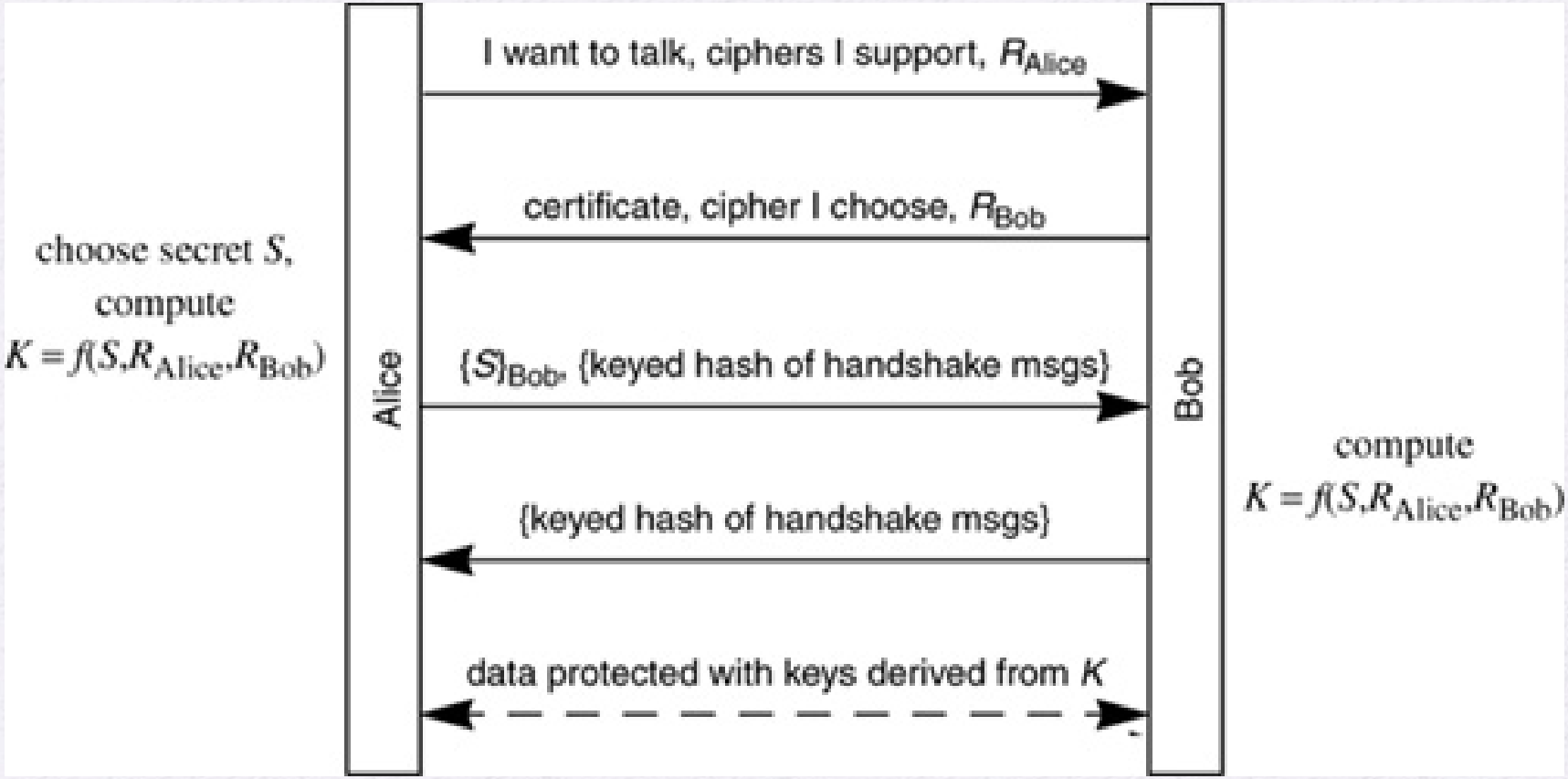
SSL/TLS

- SSL/TLS allows two parties to authenticate and establish a session key that is used to cryptographically protect the remainder of the session.
- Version 3

- SSL/TLS is designed to run in a user-level process, and runs on top of TCP.
- SSL/TLS a little simpler, because it doesn't have to worry about timing out and retransmitting lost data using TCP.
- UDP
- user-level process rather than requiring OS changes

- SSL (Secure Sockets Layer) version 2 (version 1 was never deployed) was originally deployed in Netscape Navigator in 1995.
- Microsoft improved upon SSLv2, fixing some security problems
- Netscape - version3
- SSLv3 is the most commonly deployed

SSL v3 / TLS


choose secret S,
compute
$K = f(S, R_{Alice}, R_{Bob})$
Alice
I want to talk, ciphers I support, R_{Alice}
certificate, cipher I choose, R_{Bob}
$\{S\}_{Bob}$, {keyed hash of handshake msgs}
{keyed hash of handshake msgs}
data protected with keys derived from K
Bob
compute
$K = f(S, R_{Alice}, R_{Bob})$

SSL

- Using the reliable octet stream service provided by TCP, SSL/TLS partitions this octet stream into records, with headers and cryptographic protection, to provide a reliable, encrypted, and integrity-protected stream of octets to the application.

Record Types

- four types of records:
  - user data,
  - handshake messages,
  - Alerts
  - change cipher spec

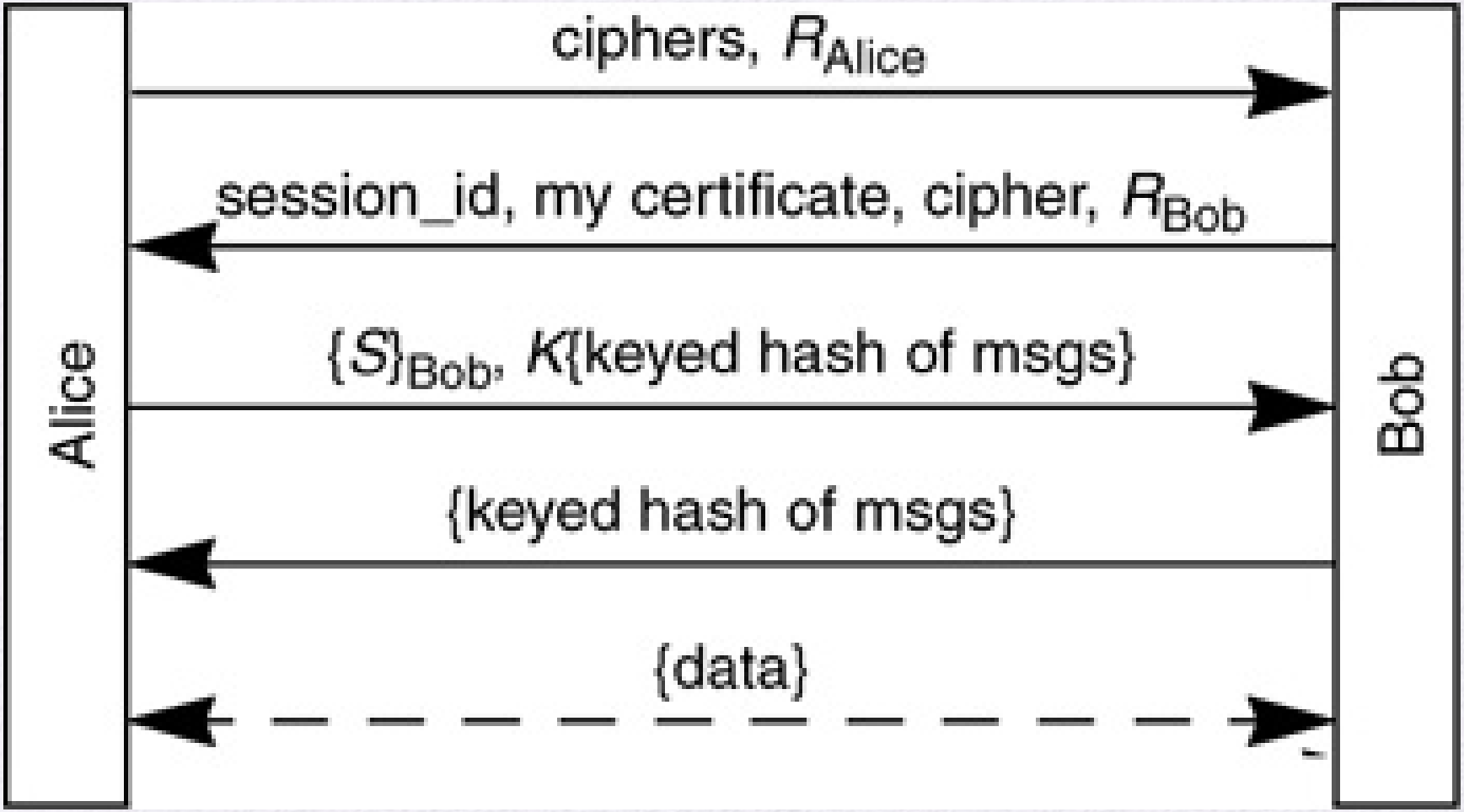
Working

- The client (Alice) initiates contact with the server (Bob).
- Then Bob sends Alice his certificate.
- Alice verifies the certificate, extracts Bob's public key, picks a random number S from which the session keys will be computed,
- and sends S to Bob, encrypted with Bob's public key. Then the remainder of the session is encrypted and integrity-protected with those session keys

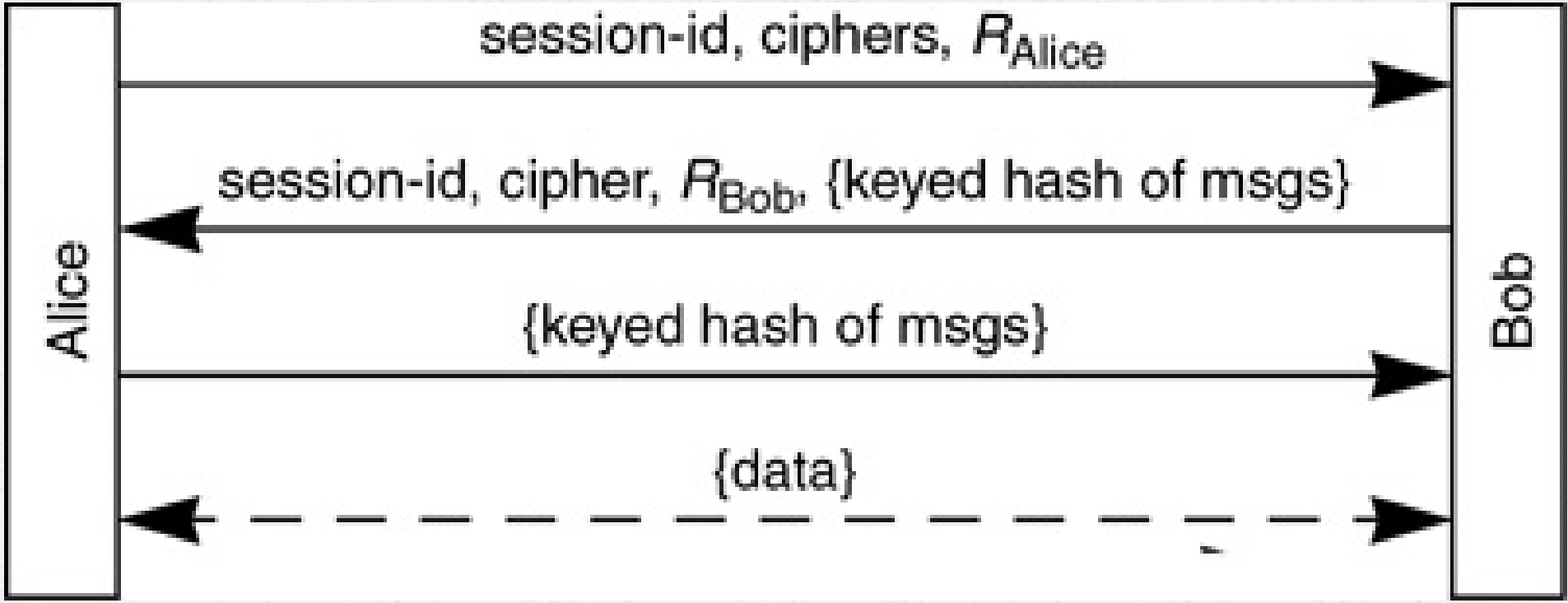
Mutual Authentication

- Alice has authenticated Bob, but Bob has no idea to whom he's talking.
- As deployed today, authentication is seldom mutual – the client authenticates the server but the server does not authenticate the client.
- It is usually done by having the user Alice send her name and password to the server Bob, cryptographically protected with the session keys.

Session initiation if no previous state



Session resumption if both sides remember session-id



Secret Key S

- The secret S sent in the first exchange is the pre-master secret.
- It is shuffled with the two Rs to produce the master secret K. In other words K $=f(S,R_{Alice},R_{Bob})$.
- The Rs are 32 octets long, and it is recommended (but not enforced) that the first 4 octets not be randomly chosen, but instead be the Unix time (seconds since January 1, 1970) when the message was generated

- with Diffie-Hellman using a fixed Diffie-Hellman number as your public key, the same two parties (Alice and Bob) will always compute the same S. If S were kept around in memory, it's possible that malicious software might steal it.

- the master secret is shuffled with the two Rs to produce the six keys used for that connection (for each side: encryption, integrity, and IV).

Keys: derived by hashing K and R_A and R_B

- 3 keys in each direction: encryption, integrity and IV
- Write keys (to send: encrypt, integrity protect)
- Read keys (to receive: decrypt, integrity check)

Client Authentication

- In SSL/TLS the server (Bob) to request that the client (Alice) authenticate herself. This is done by having Bob send a "certificate request" in message 2.
- Alice, upon seeing the request, sends her certificate, and her signature on a hash of the handshake messages, proving she knows the private key associated with the public key in the certificate.

Server Authentication

- The server sends a certificate to the client, and if it's signed by one of the CAs on the client's list, the client will accept the certificate.
- If the server presents a certificate signed by someone not on the list, certificate couldn't be verified because it was signed by an unknown authority,

Version Numbers

- it is possible to distinguish a v2 message from a v3 or TLS message (if they hadn't moved the VERSION NUMBER field it would have been easy). For instance, it happens that the first octet of the v2 client-hello will be greater than 128, and the first octet of the v3 client-hello will be something between 20 and 23

Cipher Suite

- A cipher suite is a complete package (encryption algorithm, key length, integrity checksum algorithm - specify all the crypto SSL/TLS will need).
- Cipher suites are predefined and each is assigned a numeric value. In SSLv2 the value is 3 octets long, but in SSLv3 and TLS, it's 2 octets.
- The number of suites that could potentially be defined in SSLv3/TLS to about 65000.

Choosing Cipher

- Bob returns the subset of Alice's suggested cipher suites that he is willing to support, but lets Alice make the final choice and announce it in message 3.
- Client.

- SSL means SSLv3, (the SSLv2 cipher suites have names starting with SSL2, e.g., SSL2_RC4_128_WITH_MD5),
- RSA means RSA,
- EXPORT means it's exportable (i.e., weak crypto, exportable before the rules were relaxed),
- WITH means the names weren't long enough,
- DES40 means DES with 40 bit keys,
- CBC means CBC-mode encryption, and
- SHA means HMAC-SHA is used for the MAC

- Compression is NULL =0, because of patent issues.

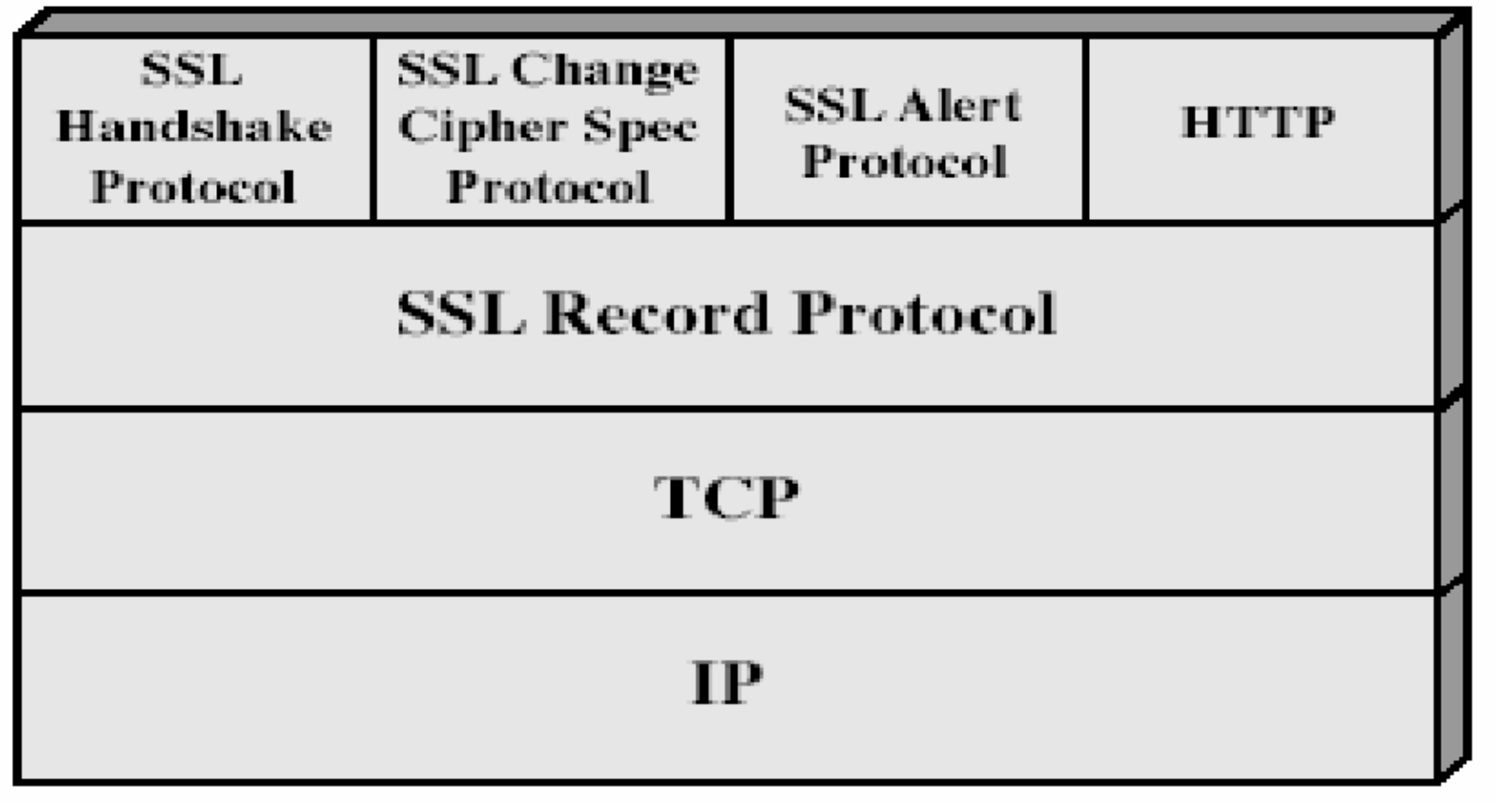
Downgrade attack

- In SSLv2, there is no integrity protection for the initial handshake, so an active attacker can remove the cipher suites with strong encryption from the list of requested cipher suites, causing Alice and Bob to agree upon a weaker cipher.
- In SSLv3 this was fixed by adding a finished message to the end of the initial handshake in which each side sends a digest of the messages in the handshake.

Truncation attack

- SSLv3 added a finished message to indicate that there is no more data to send. V2 depended on the TCP connection closing. But the TCP connection close is not cryptographically protected

SSL Stack


SSL Handshake Protocol
SSL Change Cipher Spec Protocol
SSL Alert Protocol
HTTP
SSL Record Protocol
TCP
IP

- SSL Record Protocol defines these two services for SSL connections:
  - **Confidentiality**
    - Using symmetric encryption with a shared secret key defined by Handshake Protocol
    - IDEA, RC2-40, DES-40, DES, 3DES, Fortezza, RC4-40, RC4-128
    - CBC mode (except for RC4)
    - Message is compressed before encryption
  - **Message integrity**
    - Using a MAC with shared secret key
    - Based on HMAC and MD5 or SHA (with a padding difference due to a typo in an early draft of HMAC RFC2104)
- Records sent after *ChangeCipherSpec* record are cryptographically protected
- Record header:
  - [record type, version number, length]
    - ChangeCipherSpec = 20, Alert = 21, Handshake = 22, Application_data = 23

Alert

- Severity
  - warning or fatal
- Specific alerts
  - Unexpected message, bad record mac, decompression failure, handshake failure, illegal parameter
  - Close notify, no certificate, bad certificate, unsupported certificate, certificate revoked, certificate expired, certificate unknown
- Compressed & encrypted

Handshake

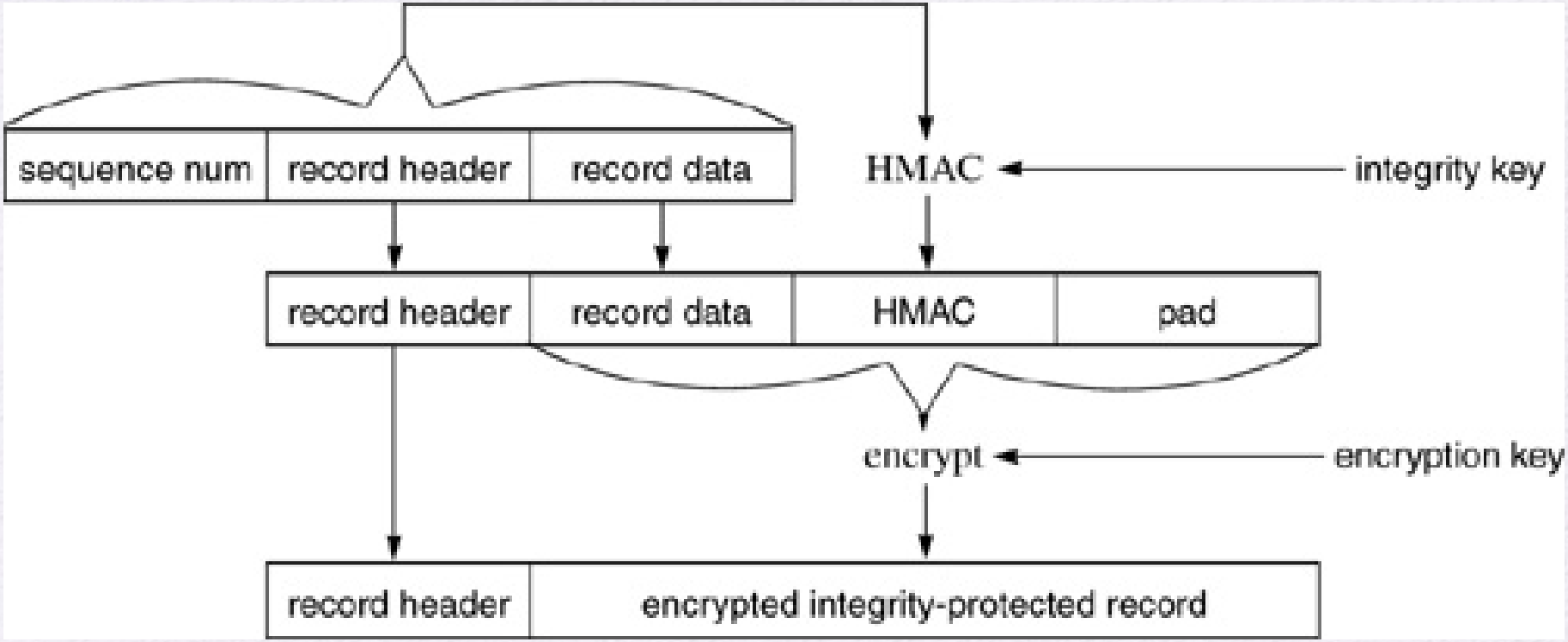
- Allows server & client to:
  - Authenticate each other
  - Negotiate encryption & MAC algorithms
  - Negotiate cryptographic keys to be used
- Comprises a series of messages in phases
  - Establish Security Capabilities
  - Server Authentication and Key Exchange
  - Client Authentication and Key Exchange
  - Finish

Exportable issues

- Exportable suites in SSLv2:
  - 40 secret bits out of 128 in symmetric keys
  - 512-bits RSA keys
- Exportability in SSLv3:
  - Integrity keys computed the same way
  - Encryption keys: 40 bits secret
  - IV non-secret
  - When a domestic server (e.g., 1024-bit RSA key) communicates with an external client the server creates an ephemeral key of 512-bits and signs it with it's 1024-bit key

Encoding

- 4 record types


sequence num
record header
record data
HMAC
integrity key
record header
record data
HMAC
pad
encrypt
encryption key
record header
encrypted integrity-protected record

	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 •Vulnerabilty 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

Table 17.1 A Comparison of Threats on the Web

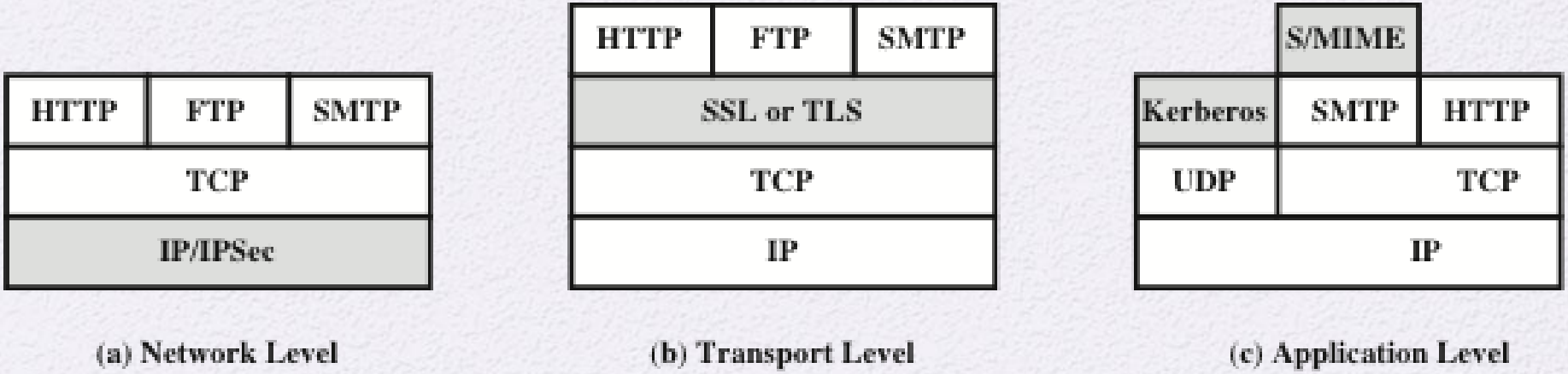


Figure 17.1 Relative Location of Security Facilities in the TCP/IP Protocol Stack

Secure Sockets Layer (SSL)

- One of the most widely used security services
- A general purpose service implemented as a set of protocols that rely on TCP
  - Could be provided as part of the underlying protocol suite and therefore be transparent to applications
  - Can be embedded in specific packages

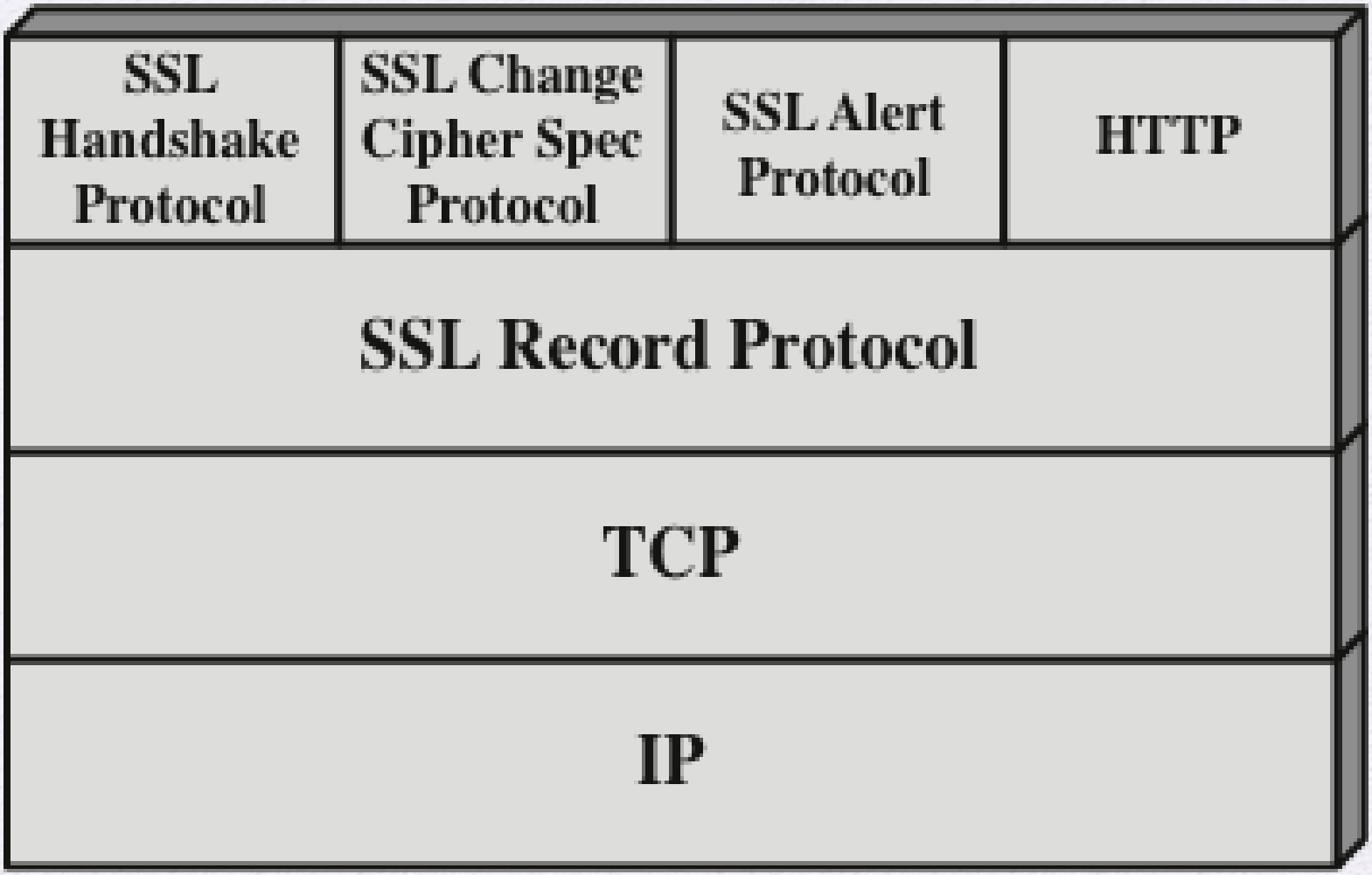


Figure 17.2 SSL Protocol Stack

SSL Architecture

- Two important SSL concepts are:

SSL connection

- A transport that provides a suitable type of service
- For SSL such connections are peer-to-peer relationships
- Connections are transient
- Every connection is associated with one session

SSL session

- An association between a client and a server
- Created by the Handshake Protocol
- Define a set of cryptographic security parameters which can be shared among multiple connections
- Are used to avoid the expensive negotiation of new security parameters for each connection

A session state is defined by the following parameters:

Session identifier	Peer certificate	Compression method	Cipher spec	Master secret	Is resumable
An arbitrary byte sequence chosen by the server to identify an active or resumable session state	An X509.v3 certificate of the peer; this element of the state may be null	The algorithm used to compress data prior to encryption	Specifies the bulk data encryption algorithm and a hash algorithm used for MAC calculation; also defines cryptographic attributes such as the hash_size	48-byte secret shared between the client and the server	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

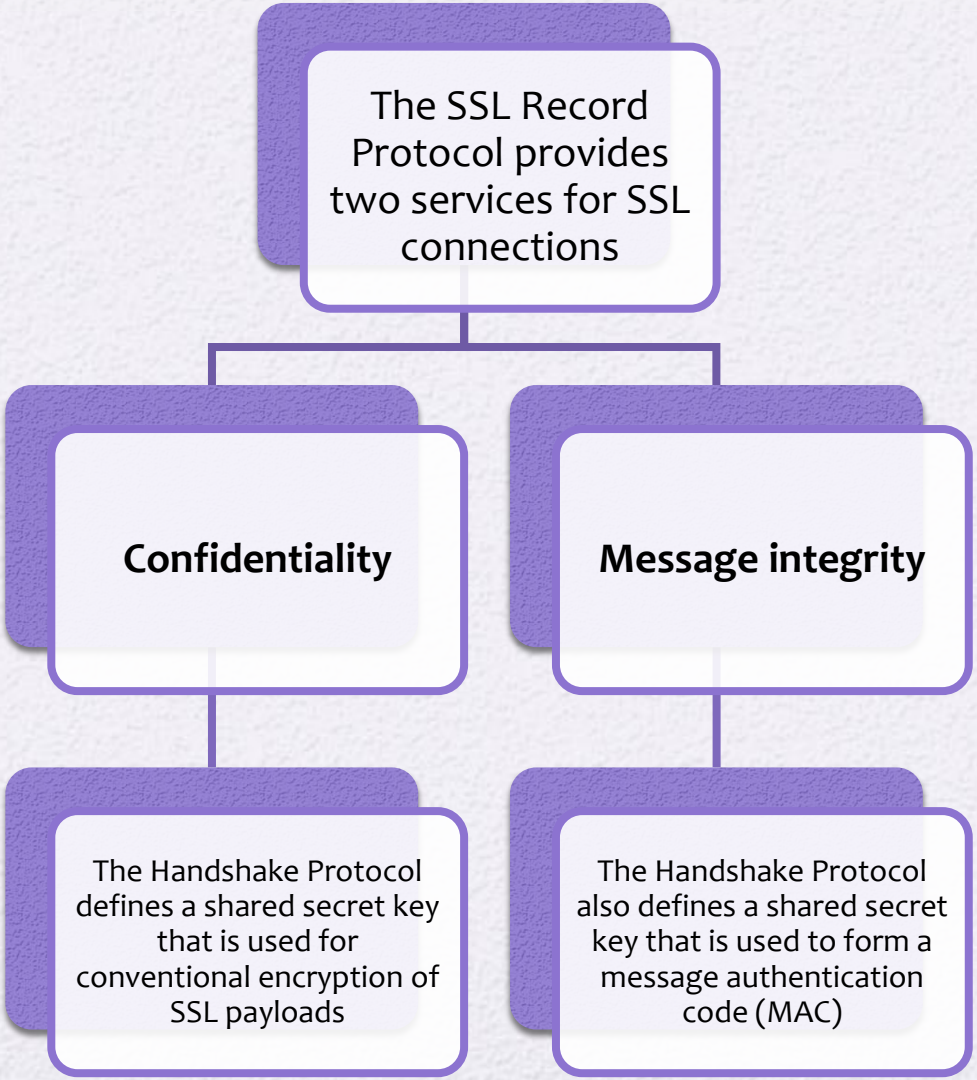
Server write key
- The secret encryption key for data encrypted by the server and decrypted by the client

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
- 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 Record Protocol



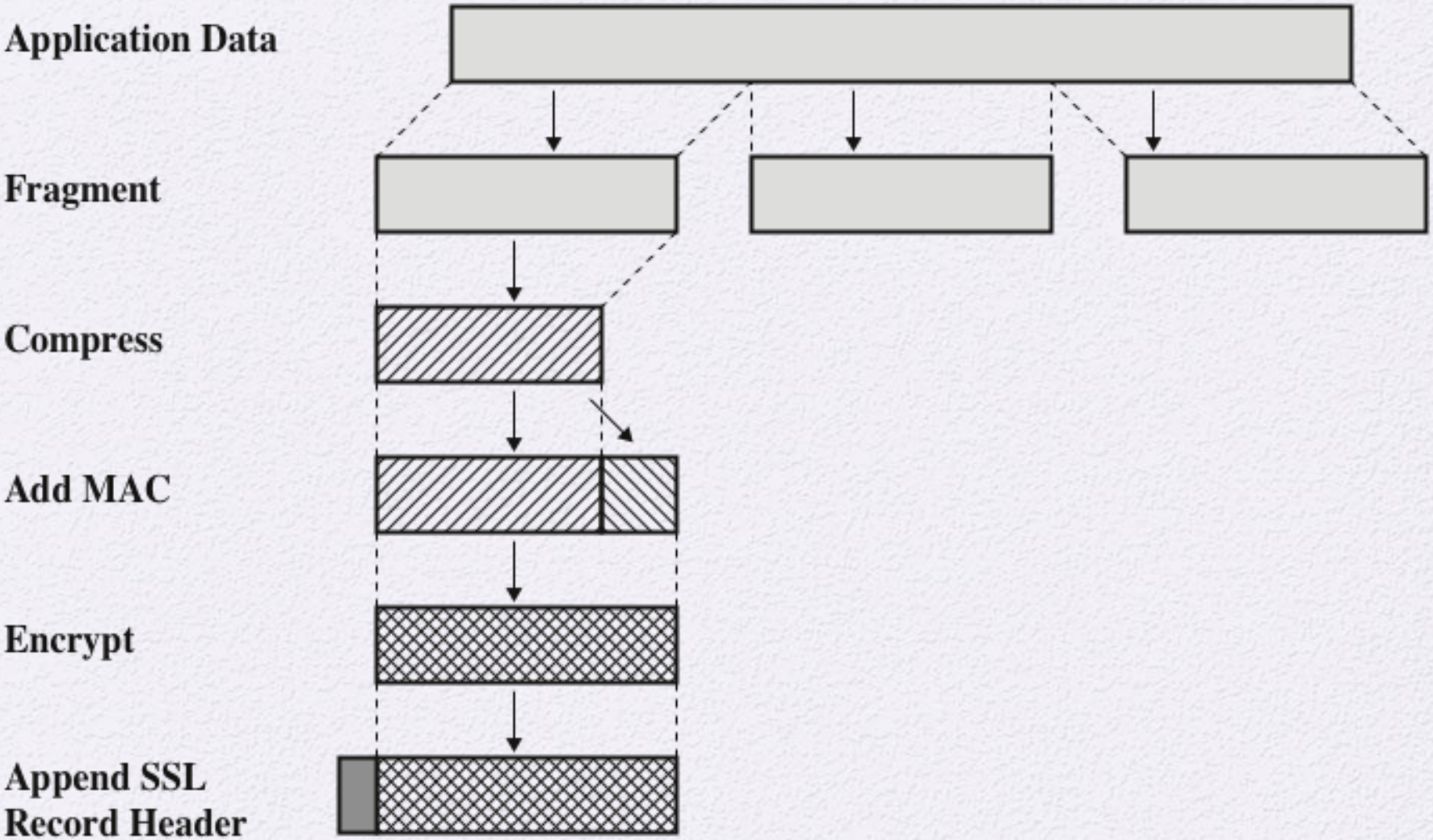


Figure 17.3 SSL Record Protocol Operation

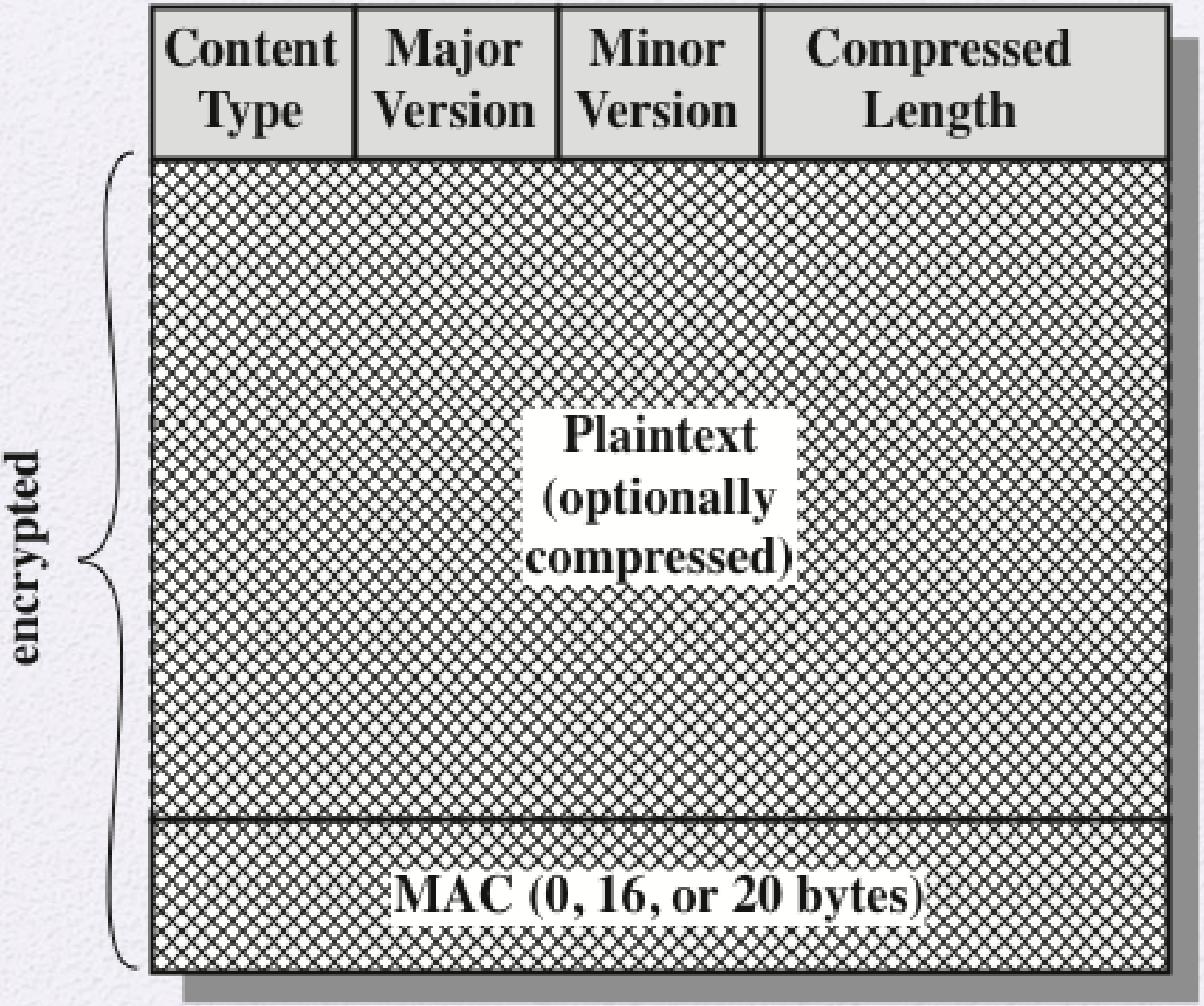


Figure 17.4 SSL Record Format

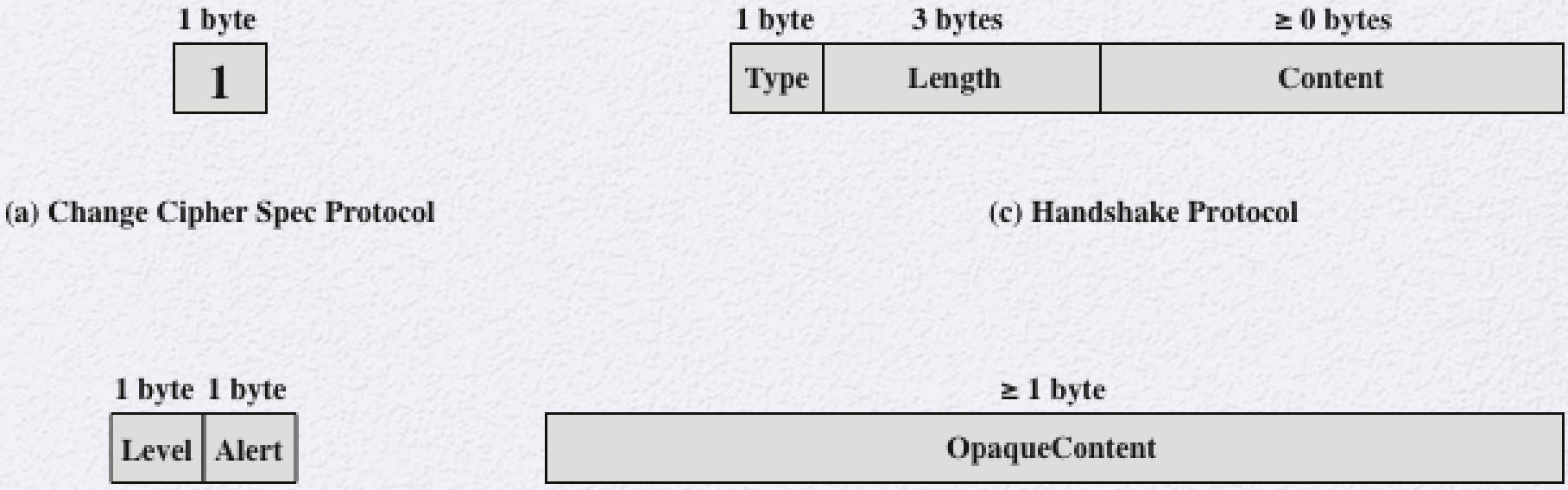


Figure 17.5 SSL Record Protocol Payload

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

Table 17.2 SSL Handshake Protocol Message Types

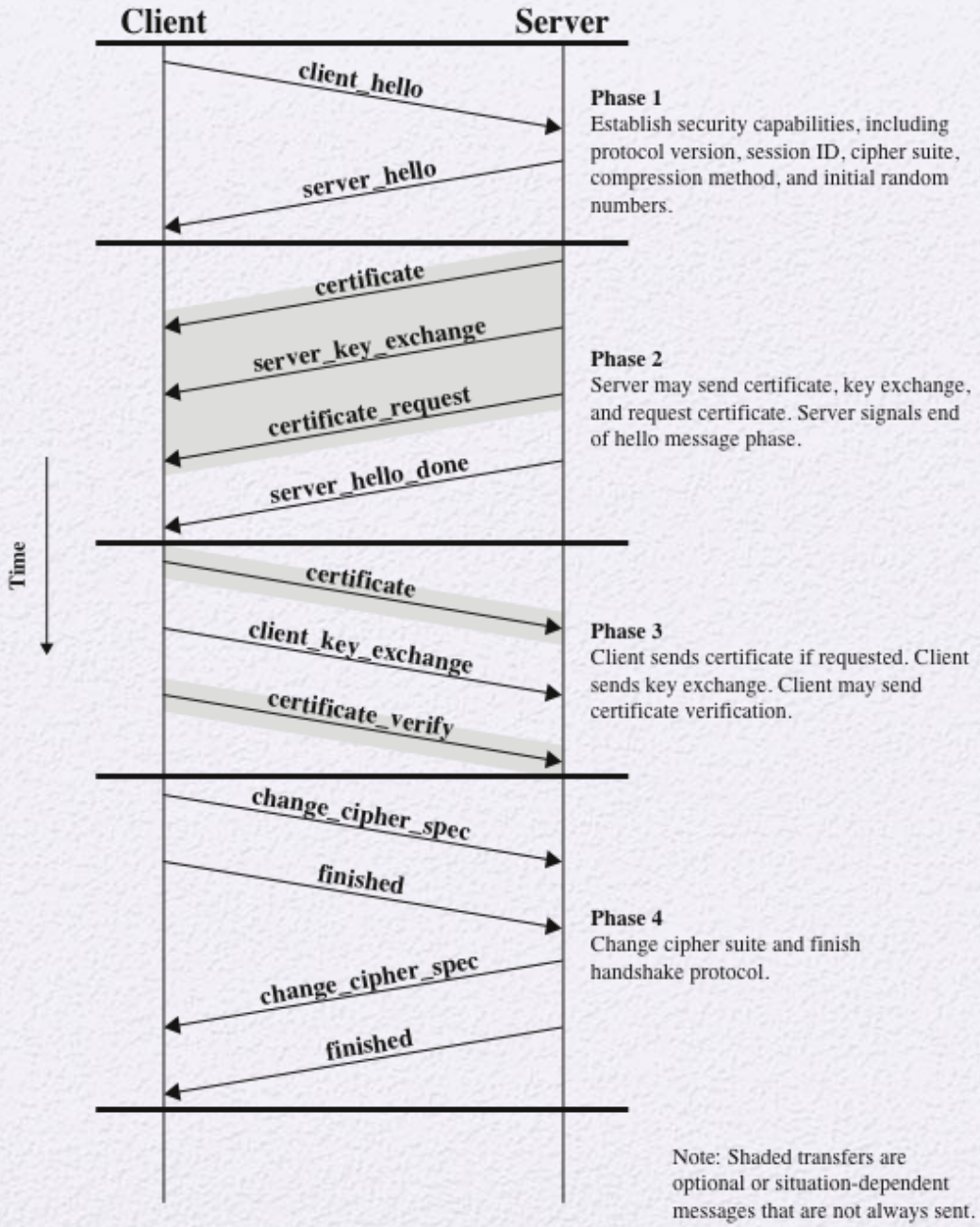


Figure 17.6 Handshake Protocol Action

Cryptographic Computations

- Two further items are of interest:
  - The creation of a shared master secret by means of the key exchange
    - The shared master secret is a one-time 48-byte value generated for this session by means of secure key exchange
  - The generation of cryptographic parameters from the master secret
    - CipherSpecs require a client write MAC secret, a server write MAC secret, a client write key, a server write key, a client write IV, and a server write IV which are generated from the master secret in that order
      - These parameters are generated from the master secret by hashing the master secret into a sequence of secure bytes of sufficient length for all needed parameters

Transport Layer Security (TLS)

- An IETF standardization initiative whose goal is to produce an Internet standard version of SSL
- Is defined as a Proposed Internet Standard in RFC 5246
  - RFC 5246 is very similar to SSLv3
  - Differences include:
    - Version number
    - Message Authentication Code
    - Pseudorandom function
    - Alert keys
    - Cipher suites
    - Client certificate types
    - Certificate_verify and Finished Messages
    - Cryptographic computations
    - Padding

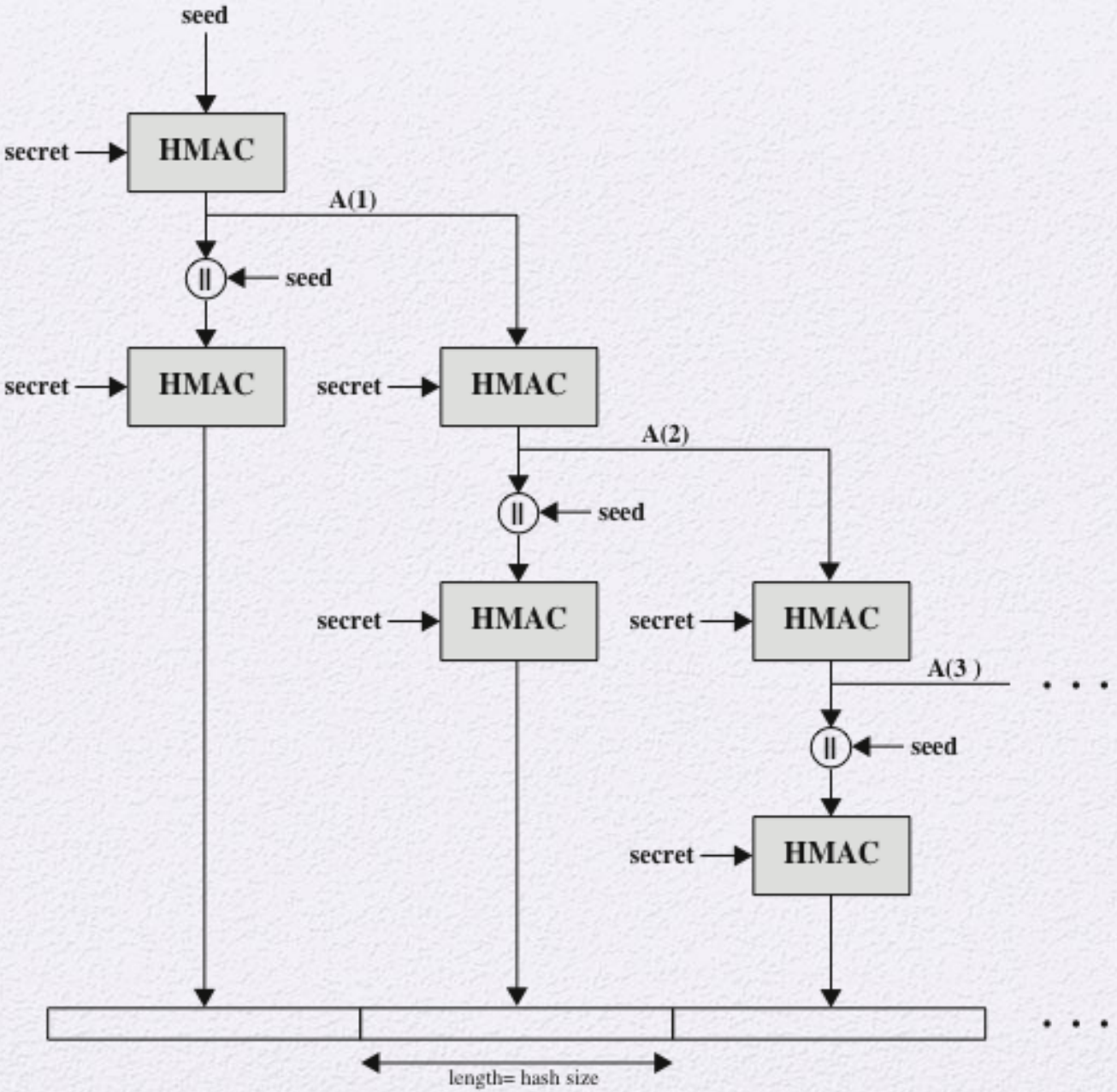


Figure 17.7 TLS Function P_hash (secret, seed)

HTTPS (HTTP over SSL)

- Refers to the combination of HTTP and SSL to implement secure communication between a Web browser and a Web server
- The HTTPS capability is built into all modern Web browsers
- A user of a Web browser will see URL addresses that begin with https:// rather than http://
- If HTTPS is specified, port 443 is used, which invokes SSL
- Documented in RFC 2818, *HTTP Over TLS*
  - There is no fundamental change in using HTTP over either SSL or TLS and both implementations are referred to as HTTPS
- When HTTPS is used, the following elements of the communication are encrypted:
  - URL of the requested document
  - Contents of the document
  - Contents of browser forms
  - Cookies sent from browser to server and from server to browser
  - Contents of HTTP header

Connection Initiation

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

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

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

- Typically the next lowest layer is TCP, but is may also be TLS/SSL

At the level of TLS, a session is established between a TLS client and a TLS server

- This session can support one or more connections at any time

A 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

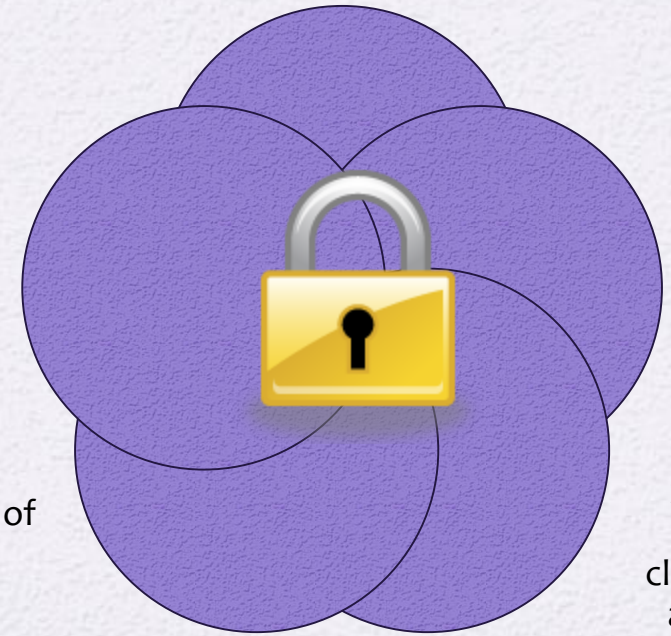
- An HTTP client or server can indicate the closing of a connection by including the line `Connection: close` in an HTTP record
- The closure of an HTTPS connection requires that TLS close the connection with the peer TLS entity on the remote side, which will involve closing the underlying TCP connection
- 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"
- An 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

Secure Shell (SSH)

A protocol for secure network communications designed to be relatively simple and inexpensive to implement

SSH client and server applications are widely available for most operating systems

- Has become the method of choice for remote login and X tunneling
- Is rapidly becoming one of the most pervasive applications for encryption technology outside of embedded systems



The initial version, SSH1 was focused on providing a secure remote logon facility to replace TELNET and other remote logon schemes that provided no security

SSH2 fixes a number of security flaws in the original scheme

- Is documented as a proposed standard in IETF RFCs 4250 through 4256

SSH also provides a more general client/server capability and can be used for such network functions as file transfer and e-mail

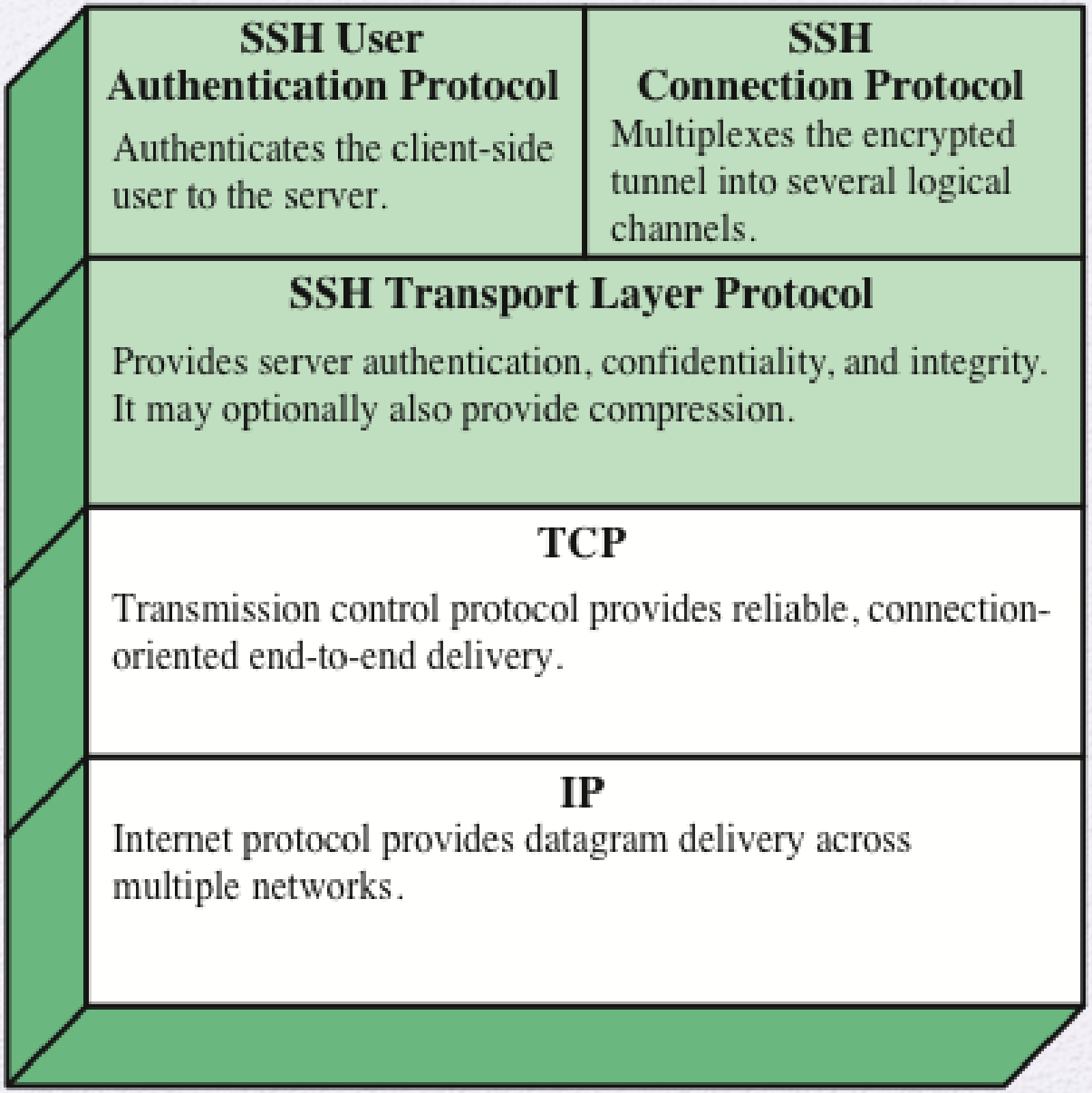


Figure 17.8 SSH Protocol Stack

Transport Layer Protocol

- Server authentication occurs at the transport layer, based on the server possessing a public/private key pair
- A server may have multiple host keys using multiple different asymmetric encryption algorithms
- Multiple hosts may share the same host key
- The server host key is used during key exchange to authenticate the identity of the host
- RFC 4251 dictates two alternative trust models:
  - The client has a local database that associates each host name with the corresponding public host key
  - The host name-to-key association is certified by a trusted certification authority (CA); the client only knows the CA root key and can verify the validity of all host keys certified by accepted CAs

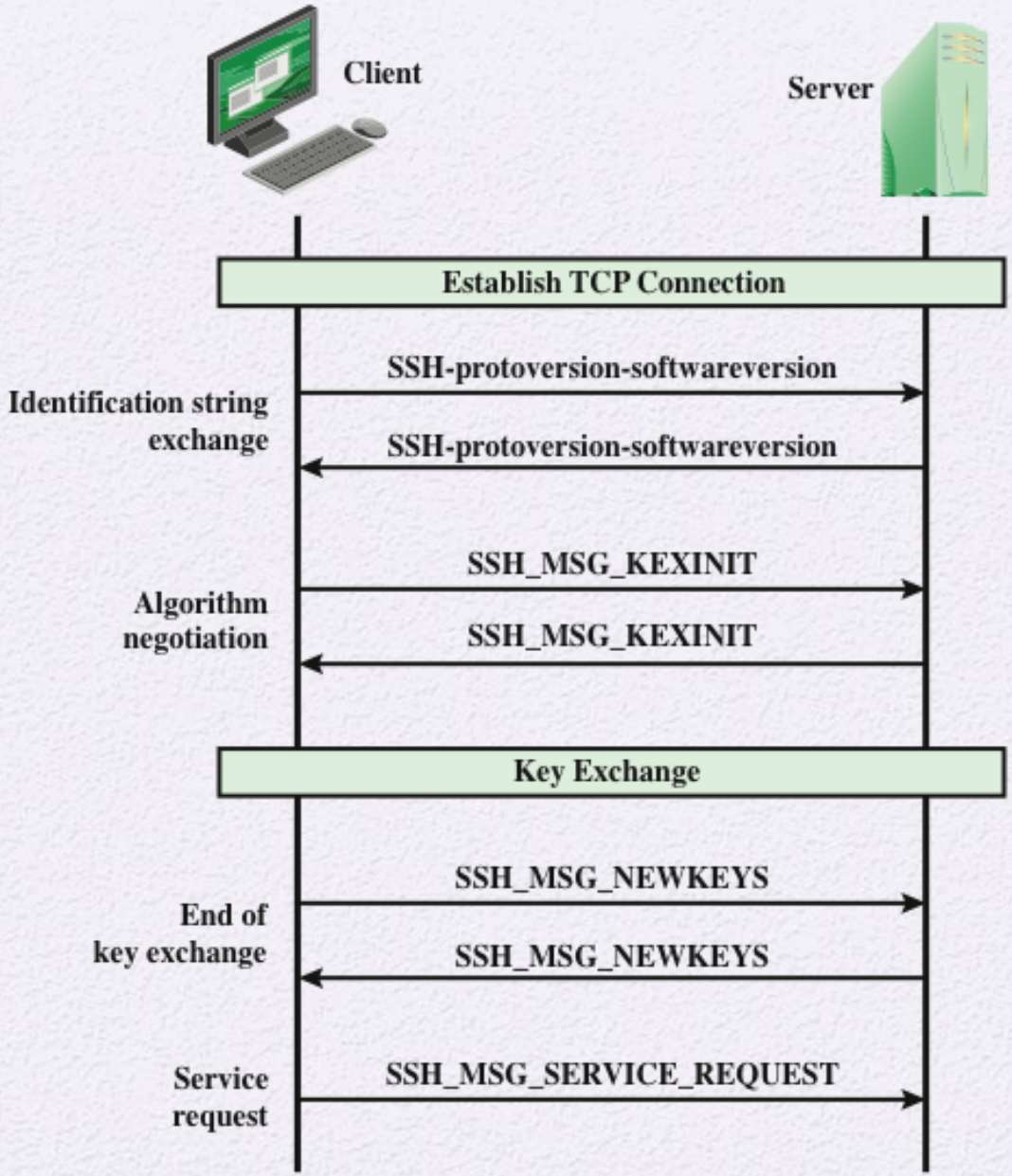


Figure 17.9 SSH Transport Layer Protocol Packet Exchanges

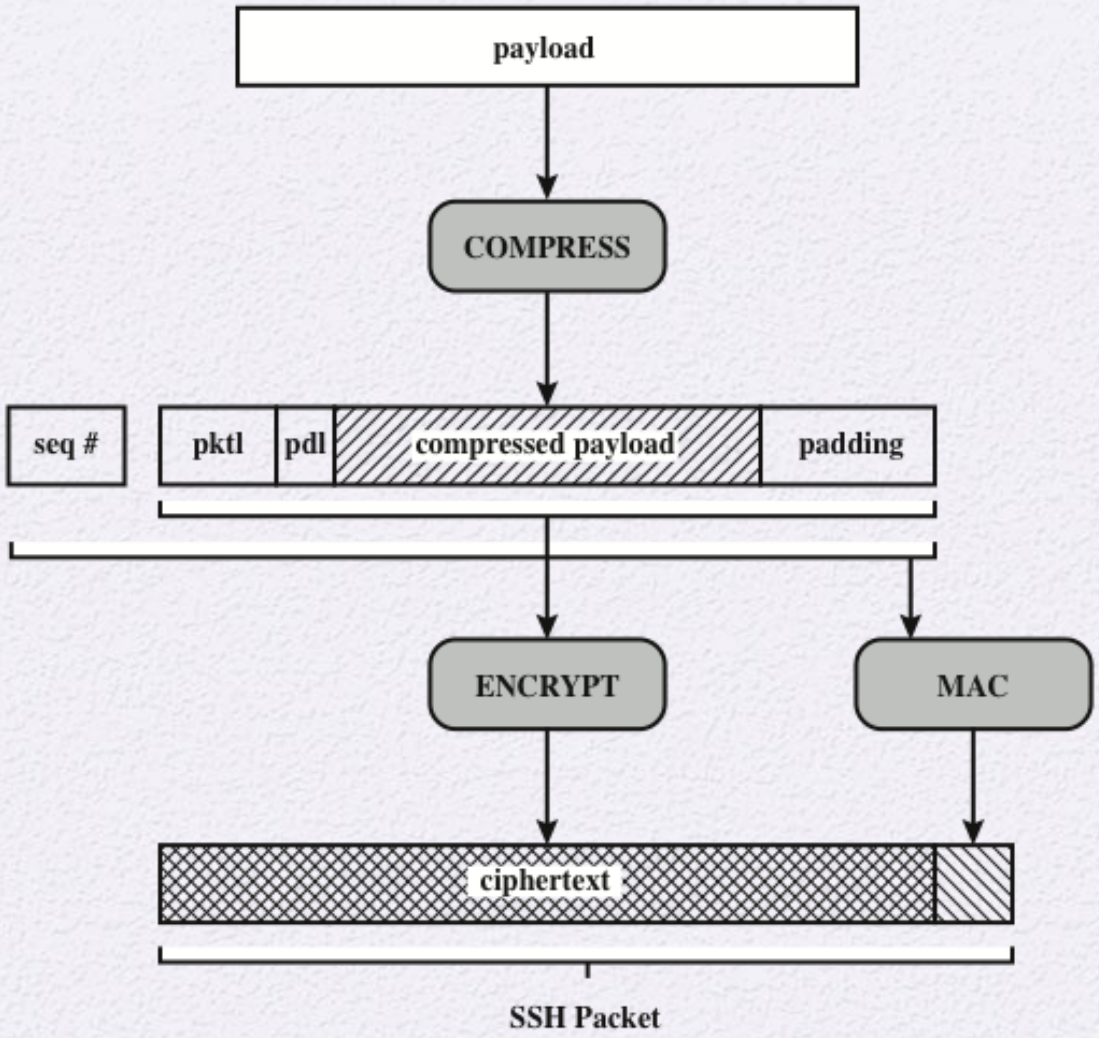


Figure 17.10 SSH Transport Layer Protocol Packet Formation

Cipher	
3des-cbc*	Three-key 3DES in CBC mode
blowfish-cbc	Blowfish in CBC mode
twofish256-cbc	Twofish in CBC mode with a 256-bit key
twofish192-cbc	Twofish with a 192-bit key
twofish128-cbc	Twofish with a 128-bit key
aes256-cbc	AES in CBC mode with a 256-bit key
aes192-cbc	AES with a 192-bit key
aes128-cbc**	AES with a 128-bit key
Serpent256-cbc	Serpent in CBC mode with a 256-bit key
Serpent192-cbc	Serpent with a 192-bit key
Serpent128-cbc	Serpent with a 128-bit key
arcfour	RC4 with a 128-bit key
cast128-cbc	CAST-128 in CBC mode

MAC algorithm	
hmac-sha1*	HMAC-SHA1; digest length = key length = 20
hmac-sha1-96**	First 96 bits of HMAC-SHA1; digest length = 12; key length = 20
hmac-md5	HMAC-MD5; digest length = key length = 16
hmac-md5-96	First 96 bits of HMAC-MD5; digest length = 12; key length = 16

Compression algorithm	
none*	No compression
zlib	Defined in RFC 1950 and RFC 1951

* = Required
** = Recommended

Table 17.3 SSH Transport Layer Cryptographic Algorithms

Authentication Methods

- Publickey
  - The client sends a message to the server that contains the client's public key, with the message signed by the client's private key
  - When the server receives this message, it checks whether the supplied key is acceptable for authentication and, if so, it checks whether the signature is correct
- Password
  - The client sends a message containing a plaintext password, which is protected by encryption by the Transport Layer Protocol
- Hostbased
  - Authentication is performed on the client's host rather than the client itself
  - This method works by having the client send a signature created with the private key of the client host
  - Rather than directly verifying the user's identity, the SSH server verifies the identity of the client host

Connection Protocol

- The SSH Connection Protocol runs on top of the SSH Transport Layer Protocol and assumes that a secure authentication connection is in use
  - The secure authentication connection, referred to as a *tunnel,* is used by the Connection Protocol to multiplex a number of logical channels
- Channel mechanism
  - All types of communication using SSH are supported using separate channels
  - Either side may open a channel
  - For each channel, each side associates a unique channel number
  - Channels are flow controlled using a window mechanism
  - No data may be sent to a channel until a message is received to indicate that window space is available
  - The life of a channel progresses through three stages: opening a channel, data transfer, and closing a channel

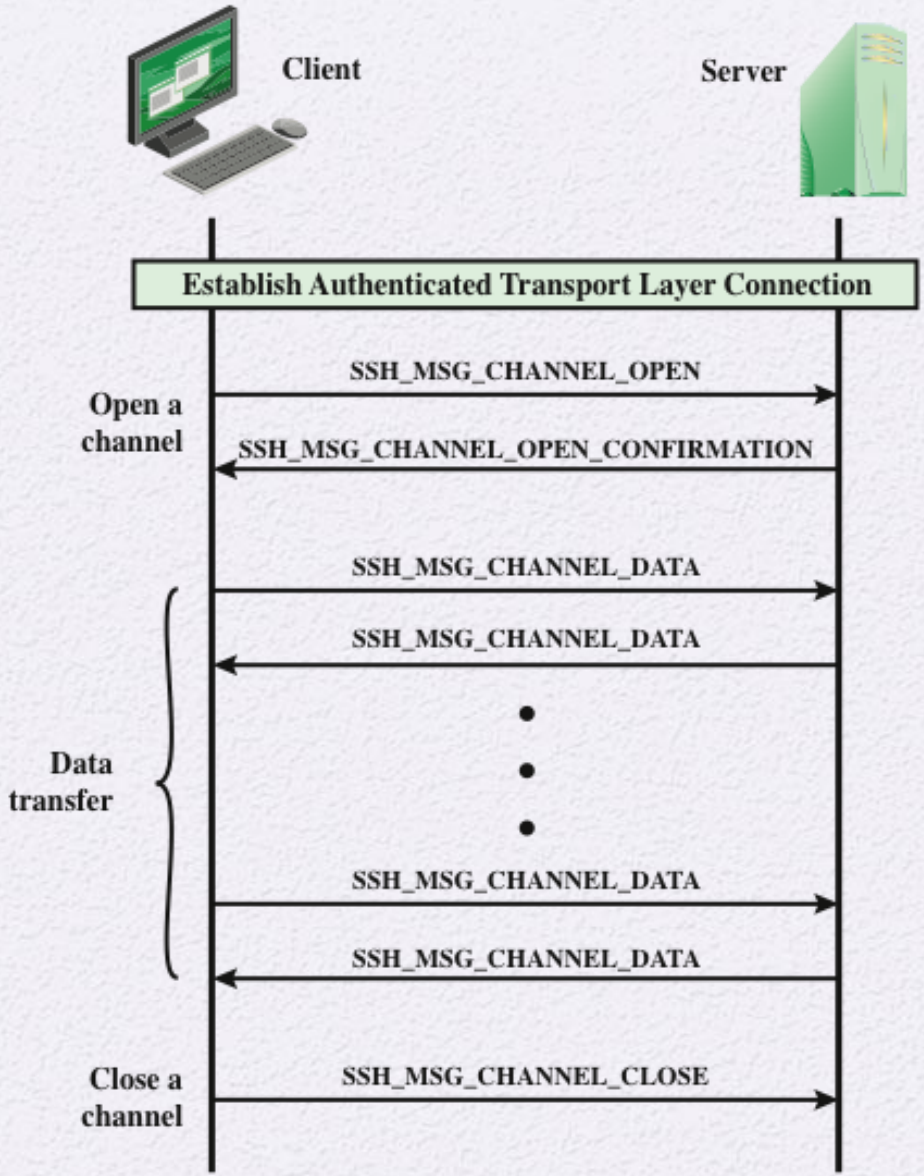


Figure 17.11 Example SSH Connection Protocol Message Exchange

Channel Types

Four channel types are recognized in the SSH Connection Protocol specification

Session

- The remote execution of a program
- The program may be a shell, an application such as file transfer or e-mail, a system command, or some built-in subsystem
- Once a session channel is opened, subsequent requests are used to start the remote program

X11

- Refers to the X Window System, a computer software system and network protocol that provides a graphical user interface (GUI) for networked computers
- X allows applications to run on a network server but to be displayed on a desktop machine

Forwarded-tcpip

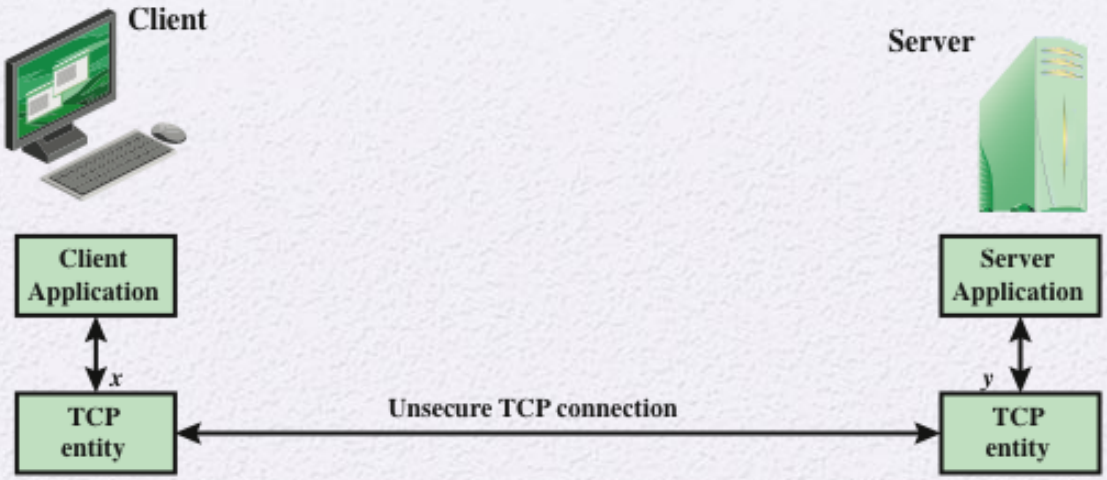
- Remote port forwarding

Direct-tcpip

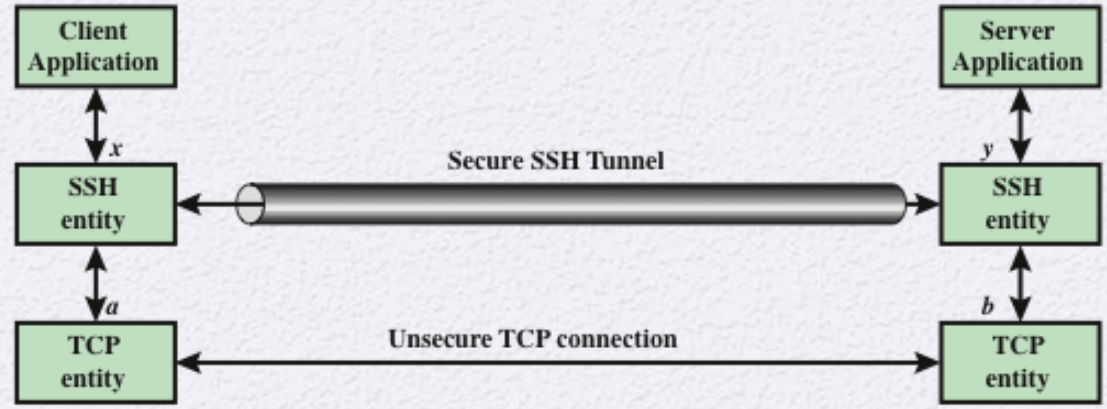
- Local port forwarding

Port Forwarding

- One of the most useful features of SSH
- Provides the ability to convert any insecure TCP connection into a secure SSH connection (also referred to as SSH tunneling)
- Incoming TCP traffic is delivered to the appropriate application on the basis of the port number (a port is an identifier of a user of TCP)
- An application may employ multiple port numbers



(a) Connection via TCP



(b) Connection via SSH Tunnel

Figure 17.12 SSH Transport Layer Packet Exchanges

Summary

- Web security considerations
  - Web security threats
  - Web traffic security approaches
- Secure sockets layer
  - SSL architecture
  - SSL record protocol
  - Change cipher spec protocol
  - Alert protocol
  - Handshake protocol
  - Cryptographic computations
- HTTPS
  - Connection initiation
  - Connection closure

- Transport layer security
  - Version number
  - Message authentication code
  - Pseudorandom function
  - Alert codes
  - Cipher suites
  - Client certificate types
  - Certificate_verify and finished messages
  - Cryptographic computations
  - Padding
- Secure shell (SSH)
  - Transport layer protocol
  - User authentication protocol
  - Communication protocol