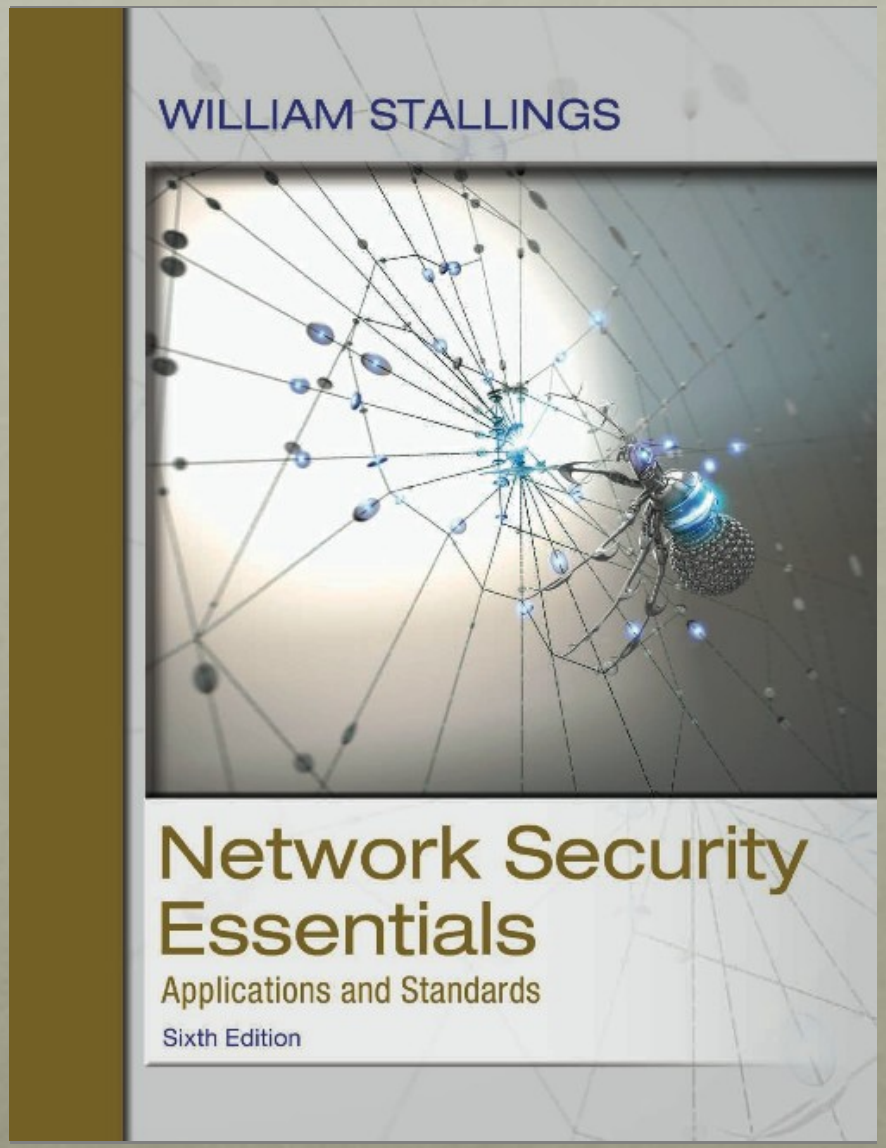
Network Security Essentials

Sixth Edition

by William Stallings



© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Chapter 6

Transport-Level Security

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

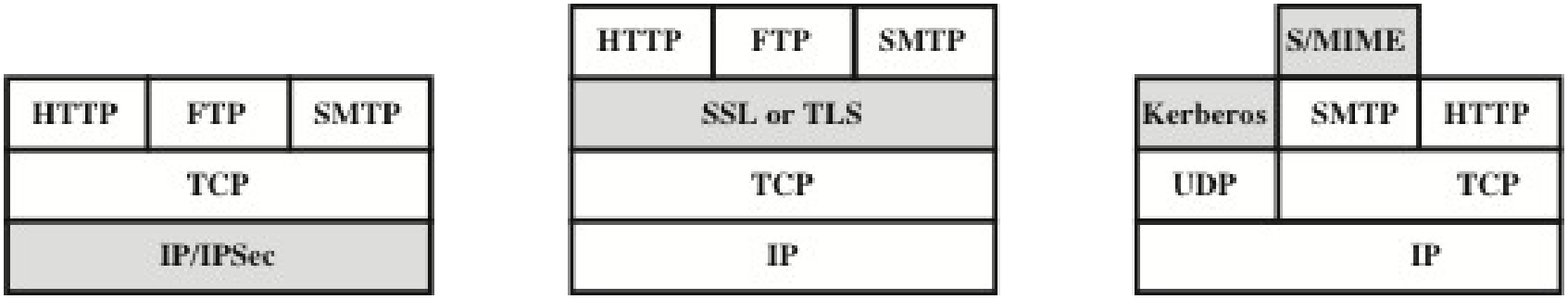


Figure 6.1 Relative Location of Security Facilities in the TCP/IP Protocol Stack

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Transport Layer security (TLS)

- One of the most widely used security services
- TLS is an Internet standard that evolved from a commercial protocol known as *Secure Sockets Layer* (SSL)
- TLS is a general purpose service implemented as a set of protocols that rely on TCP
  - TLS could be provided as part of the underlying protocol suite and therefore be transparent to applications
  - Alternatively, TLS can be embedded in specific packages

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

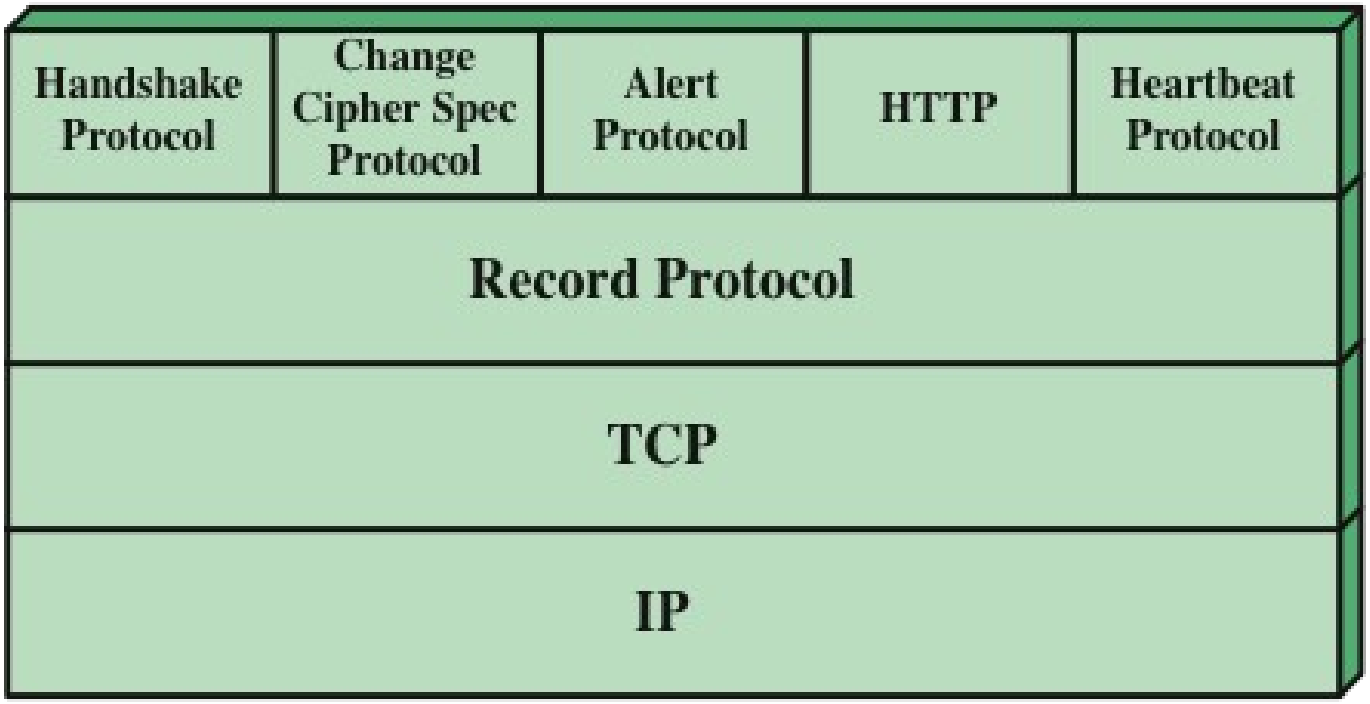


Figure 6.2 SSL/TLS Protocol Stack

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

TLS Architecture

Two important TLS concepts are:

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

■ Session: A TLS 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.

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

A session state is defined by the following parameters:

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 the server.
- Is resumable: A flag indicating whether the session can be used to initiate new connections.

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

A connection state is defined by the following parameters:

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 send or receives a change cipher spec message, the appropriate sequence number is set to zero. Sequence numbers may not exceed $2^{64} - 1$.

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

TLS Record Protocol

TLS Record Protocol defines two services for TLS connections:

- Confidentiality: The Handshake Protocol defines a shared secret key that is used for conventional encryption of TLS payloads.
- Message Integrity: The Handshake Protocol also defines a shared secret key that is used to form a message authentication code (MAC)

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

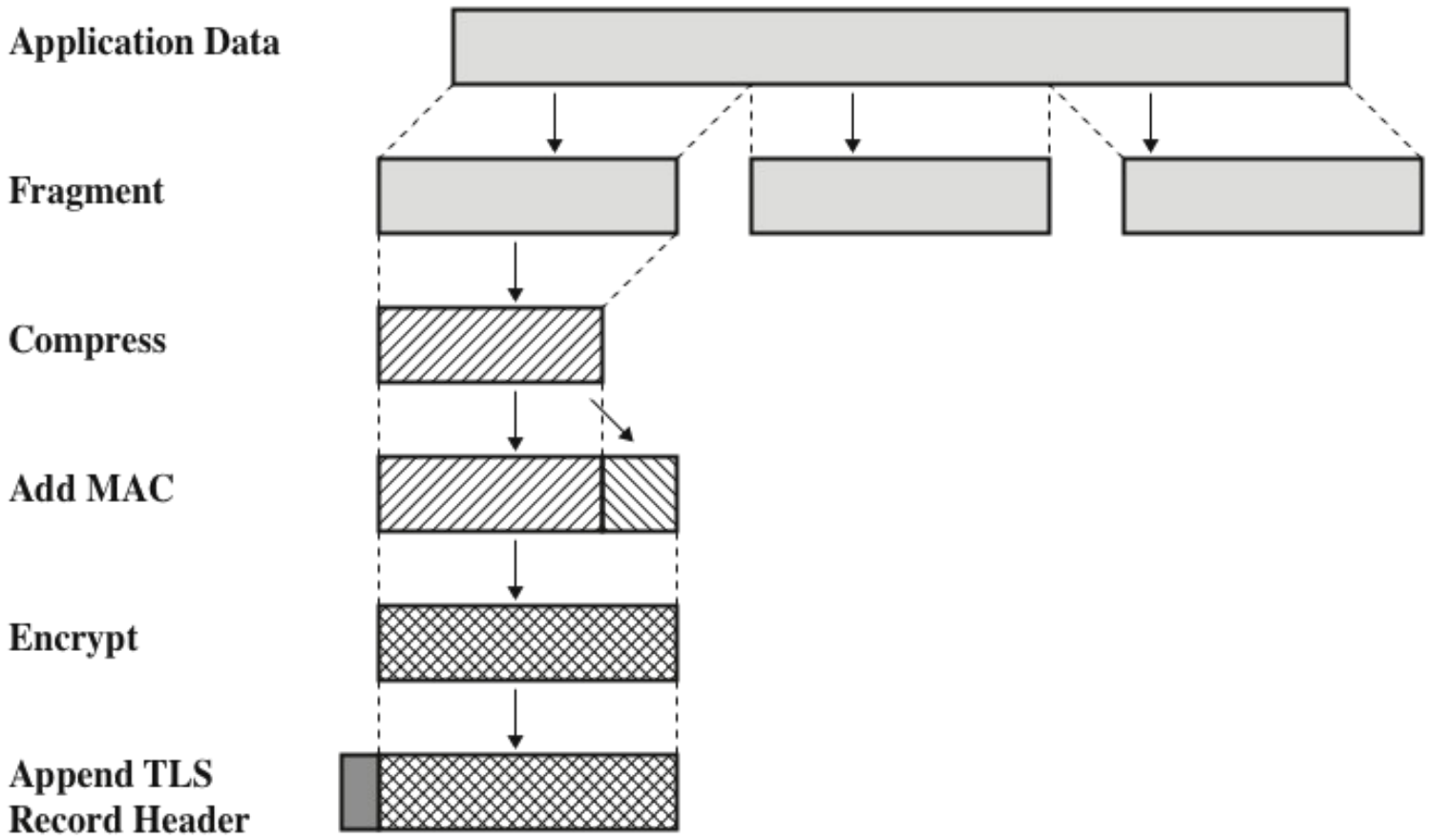


Figure 6.3 TLS Record Protocol Operation

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

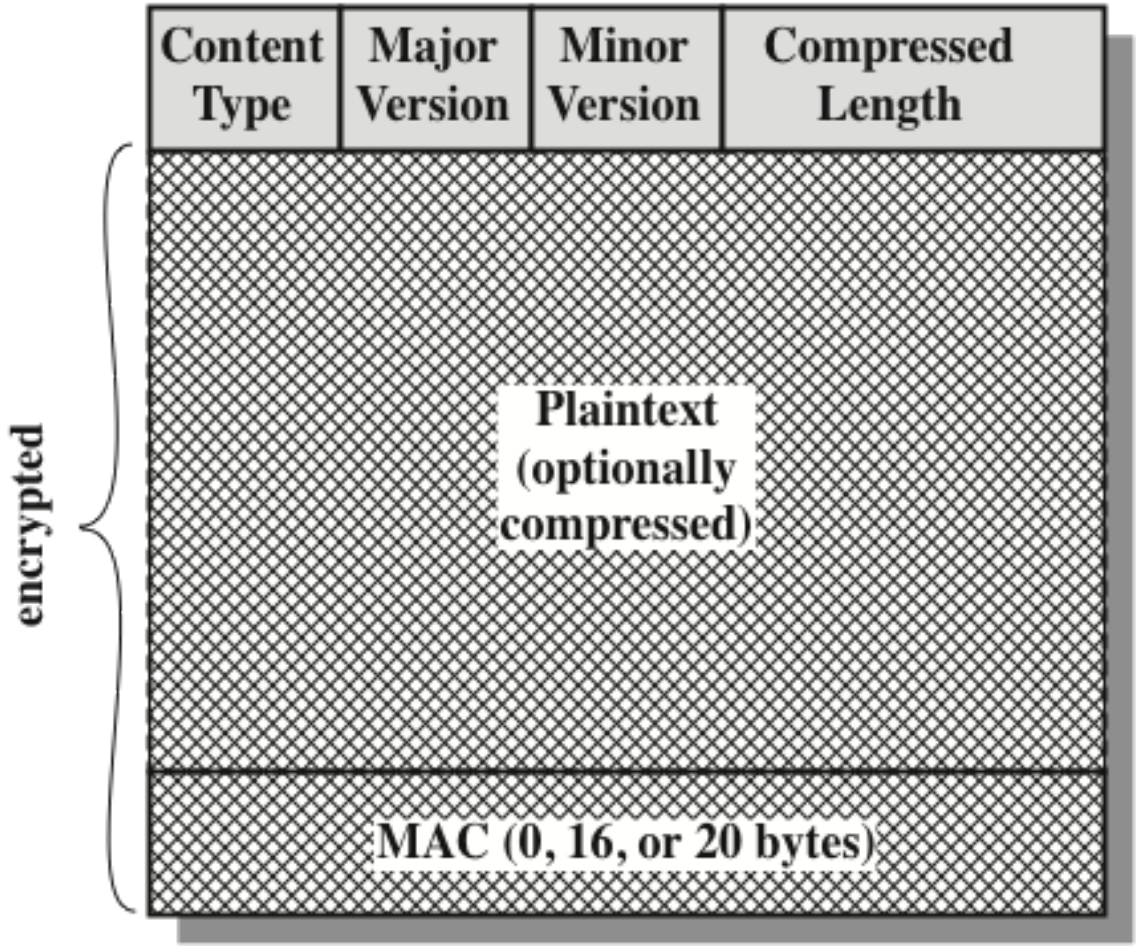


Figure 6.4 SSL Record Format

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

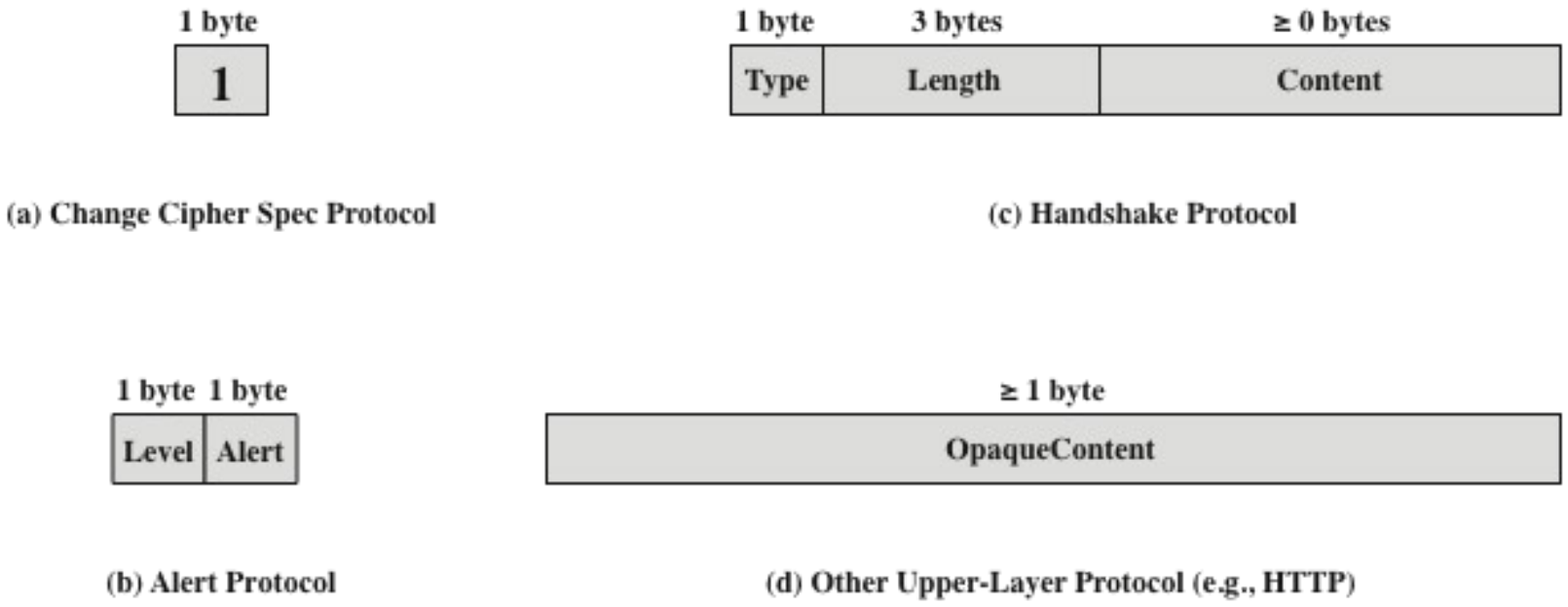


Figure 6.5 TLS **Record Protocol Payload**

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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 6.2 TLS Handshake Protocol Message Types

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

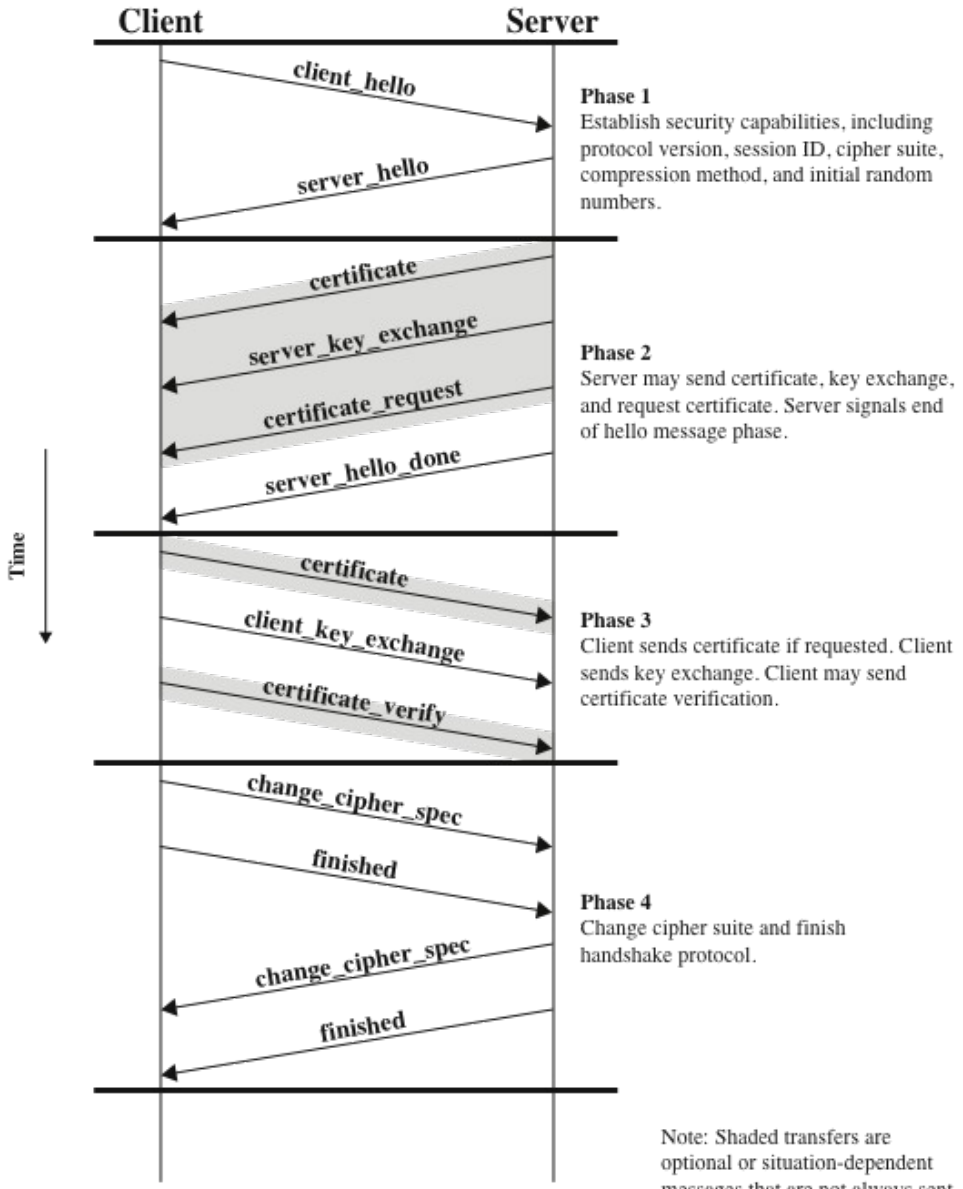


Figure 6.6 Handshake Protocol Action

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Heartbeat Protocol

- A heartbeat is a periodic signal generated by hardware or software to indicate normal operation or to synchronize other parts of a system
- A heartbeat protocol is typically used to monitor the availability of a protocol entity
- The heartbeat protocol runs on top of the TLS Record Protocol
  - Consists of two message types: heartbeat_request and heartbeat_response
- The heartbeat serves two purposes:
  - It assures the sender that the recipient is still alive, even though there may not have been any activity over the underlying TCP connection
  - It generates activity across the connection during idle periods, which avoids closure by a firewall that does not tolerate idle connections

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

SSL/TLS Attacks

Attack Categories We can group the attacks into four general categories:

- Attacks on the handshake protocol: As early as 1998, an approach to compromising the handshake protocol based on exploiting the formatting and implementation of the RSA encryption scheme was presented [BLEI98]. As countermeasures were implemented the attack was refined and adjusted to not only thwart the countermeasures but also speed up the attack [e.g., BARD12].

- Attacks on the record and application data protocols: A number of vulnerabilities have been discovered in these protocols, leading to patches to counter the new threats. As a recent example, in 2011, researchers Thai Duong and Juliano Rizzo demonstrated a proof of concept called BEAST (Browser Exploit Against SSL/TLS) that turned what had been considered only a theoretical vulnerability into a practical attack [GOOD11]. BEAST leverages a type of cryptographic attack called a chosen-plaintext attack. The attacker mounts the attack by choosing a guess for the plaintext that is associated with a known ciphertext. The researchers developed a practical algorithm for launching successful attacks.

- Attacks on the PKI: Checking the validity of X.509 certificates is an activity subject to a variety of attacks, both in the context of SSL/TLS and elsewhere. For example, [GEOR12] demonstrated that commonly used libraries for SSL/TLS suffer from vulnerable certificate validation implementations. The authors revealed weaknesses in the source code of OpenSSL, GnuTLS, JSSE, ApacheHttpClient, Weberknecht, cURL, PHP, Python and applications built upon or with these products.

- Other attacks: [MEYE13] lists a number of attacks that do not fit into any of the preceding categories. One example is an attack announced in 2011 by the German hacker group The Hackers Choice, which is a DoS attack [KUMA11]. The attack creates a heavy processing load on a server by overwhelming the target with SSL/TLS handshake requests. Boosting system load is done by establishing new connections or using renegotiation. Assuming that the majority of computation during a handshake is done by the server, the attack creates more system load on the server than on the source device, leading to a DoS. The server is forced to continuously recompute random numbers and keys.

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Secure Shell (SSH)

Secure Shell (SSH) is a protocol for secure network communications designed to be relatively simple and inexpensive to implement. 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. SSH also provides a more general client/server capability and can be used for such network functions as file transfer and e-mail. A new version, SSH2, fixes a number of security flaws in the original scheme. SSH2 is documented as a proposed standard in IETF RFCs 4250 through 4256.

SSH client and server applications are widely available for most operating systems. It has become the method of choice for remote login and X tunneling and is rapidly becoming one of the most pervasive applications for encryption technology outside of embedded systems.

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

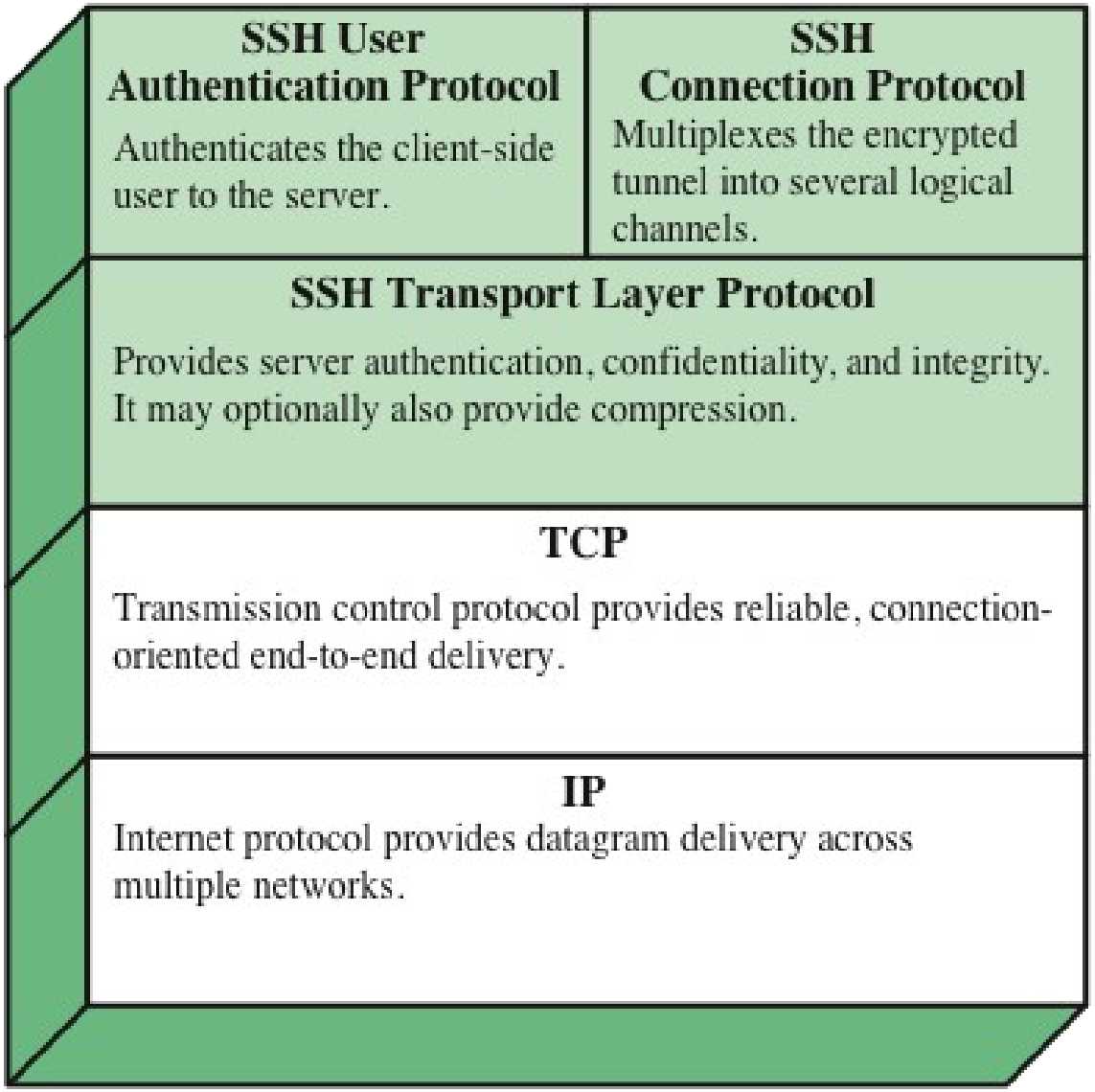


Figure 6.8 SSH Protocol Stack

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

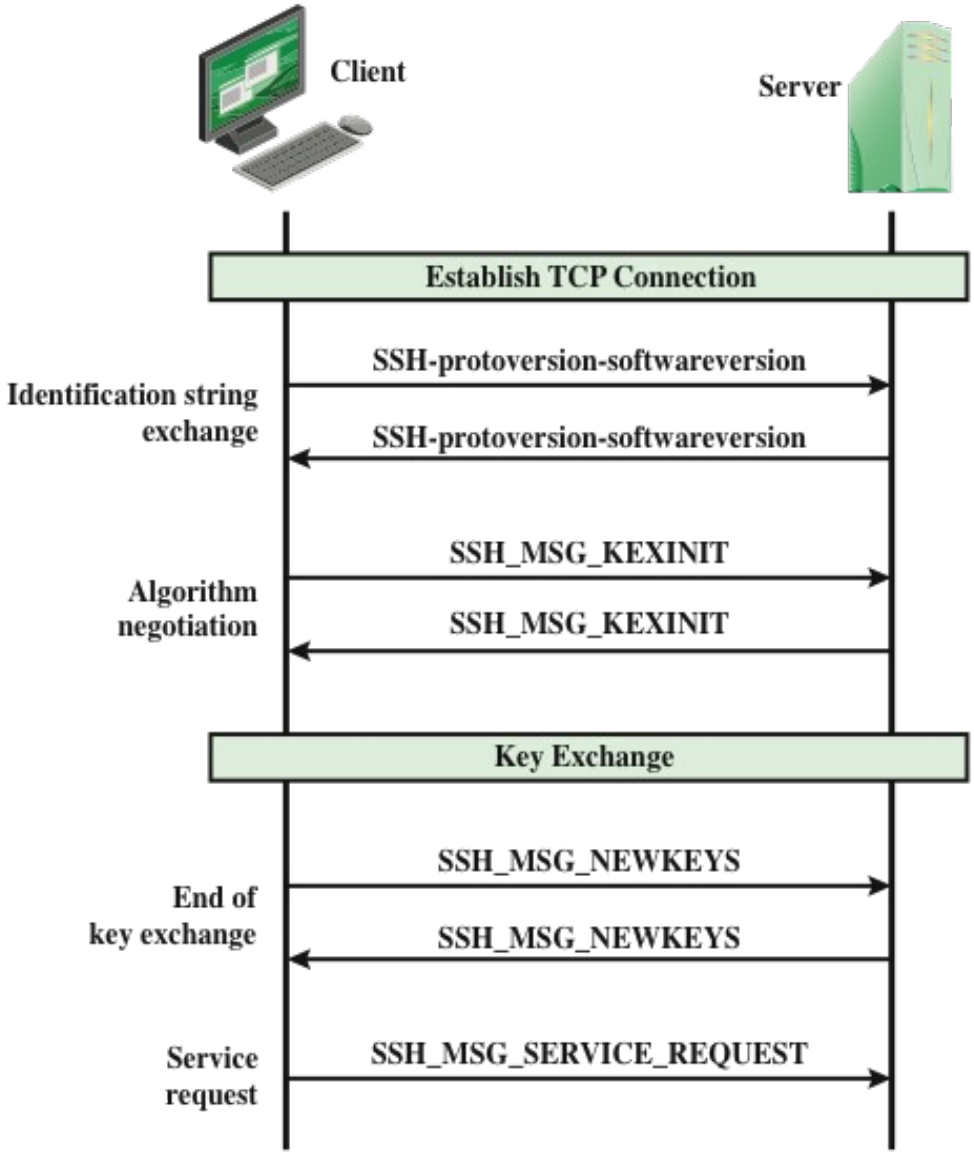


Figure 6.9 SSH Transport Layer Protocol Packet Exchanges

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

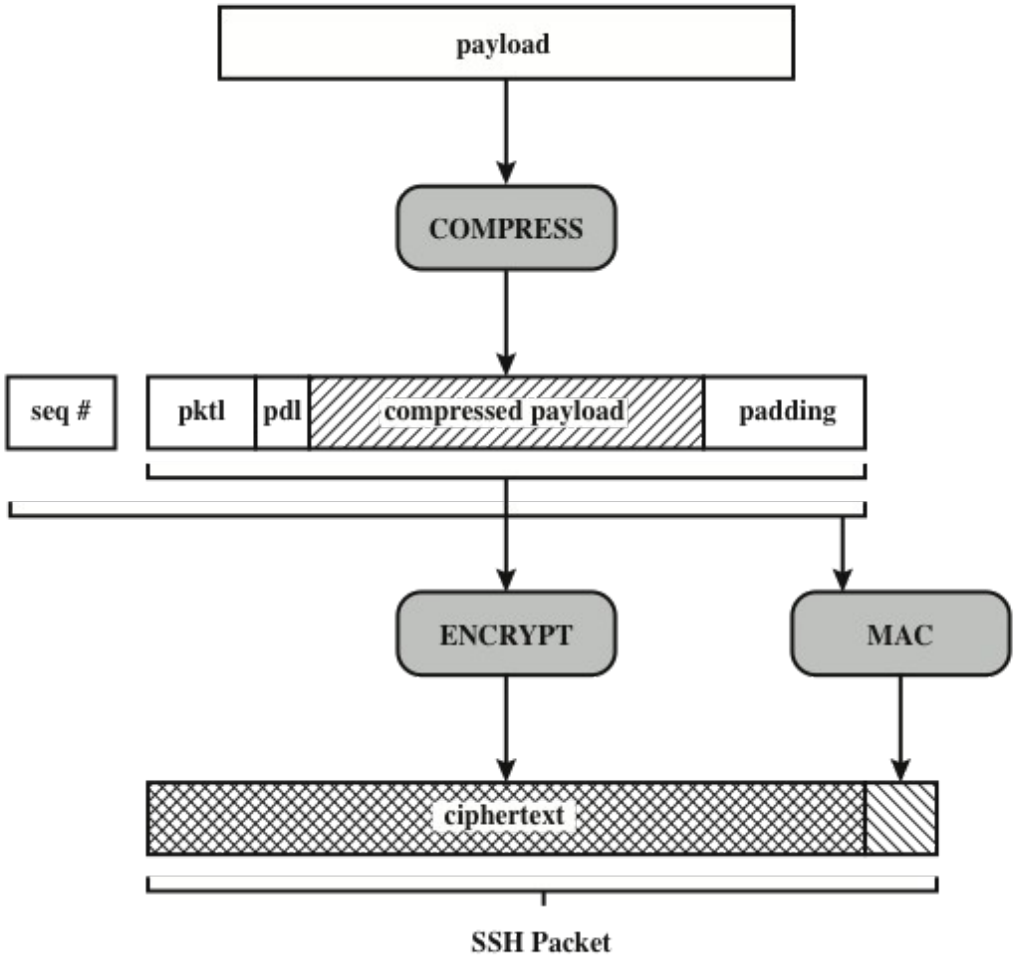


Figure 6.10 SSH Transport Layer Protocol Packet Formation

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Table 6.3 SSH Transport Layer Cryptographic Algorithms

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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Authentication Methods

The server may require one or more of the following authentication methods:

• public key: The details of this method depend on the public-key algorithm chosen. In essence, 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.

• host based: Authentication is performed on the client's host rather than the client itself. Thus, a host that supports multiple clients would provide authentication for all its clients. This method works by having the client send a signature created with the private key of the client host. Thus, rather than directly verifying the user's identity, the SSH server verifies the identity of the client host—and then believes the host when it says the user has already authenticated on the client side.

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

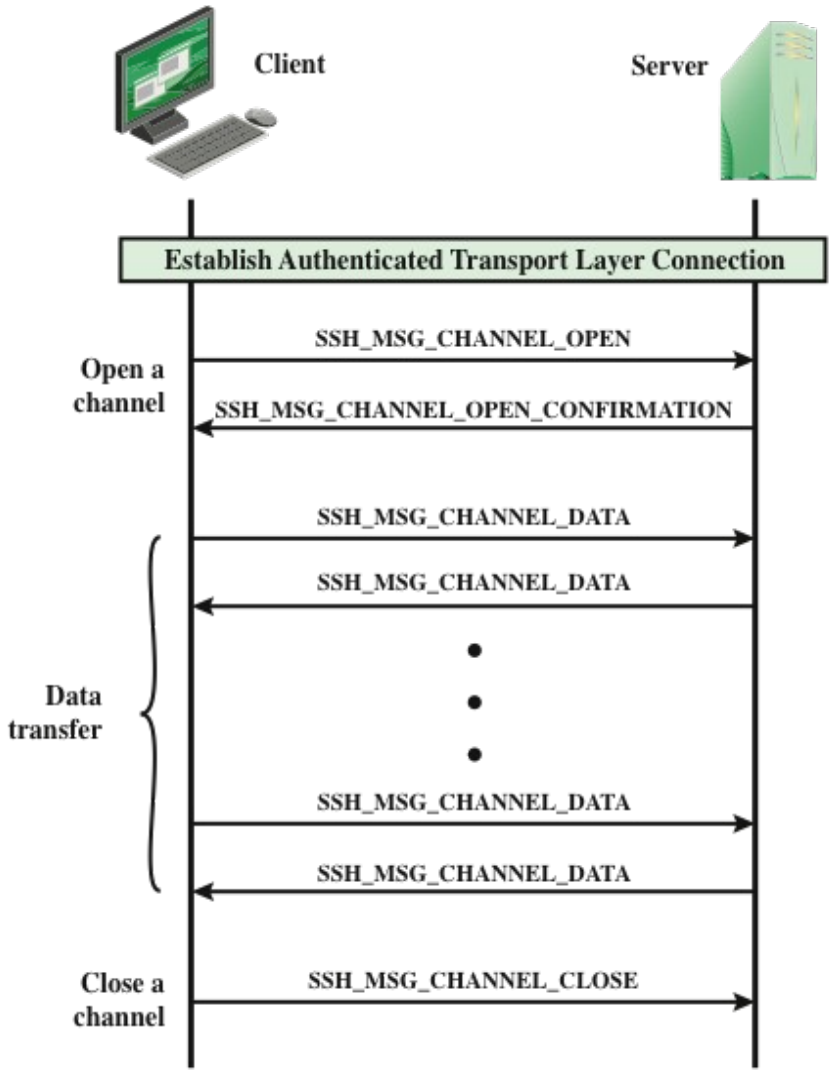


Figure 6.11 Example SSH Connection Protocol Message Exchange

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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: This 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: This is remote port forwarding, as explained in the next subsection.

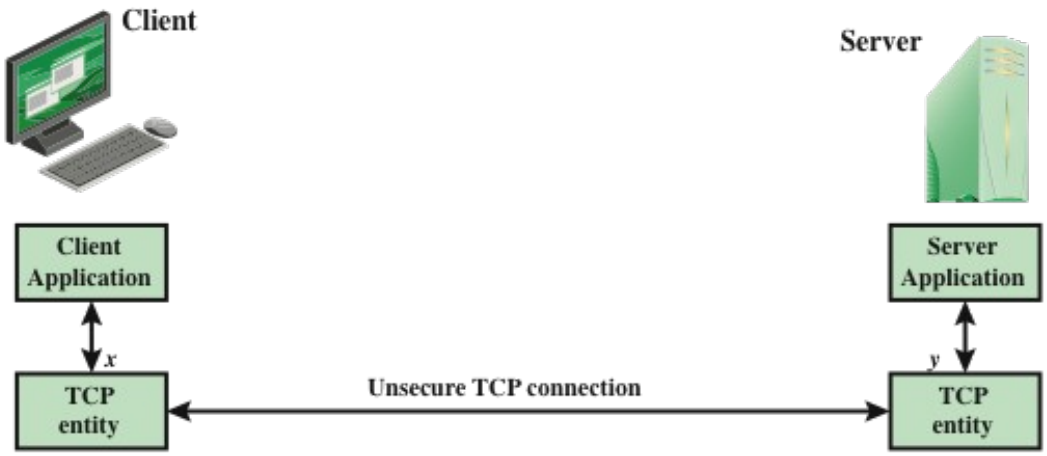
• direct-tcpip: This is local port forwarding, as explained in the next subsection.

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

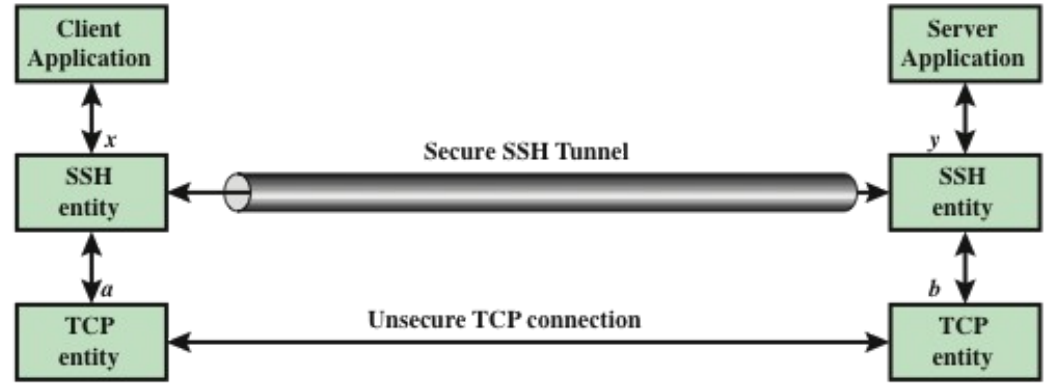
Port Forwarding

- 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

© 2017 Pearson Education, Inc., Hoboken, NJ.
All rights reserved.



(a) Connection via TCP



(b) Connection via SSH Tunnel

Figure 6.12 SSH Transport Layer Packet Exchanges

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Summary

- Transport Layer Security
  - TLS architecture
  - TLS record protocol
  - Change cipher spec protocol
  - Alert protocol
  - Handshake protocol
  - Cryptographic computations
  - Heartbeat protocol
  - SSL/TLS attacks
  - TLSv1.3
- Web security considerations
  - Web security threats
  - Web traffic security approaches
- HTTPS
  - Connection initiation
  - Connection closure
- Secure shell (SSH)
  - Transport layer protocol
  - User authentication protocol
  - Communication protocol

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Table 6.1 A Comparison of Threats on the Web

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Cryptographic Computations

- 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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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. Normal HTTP behavior, including retained connections, should be followed.

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 it also may 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. As we have seen, 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.

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

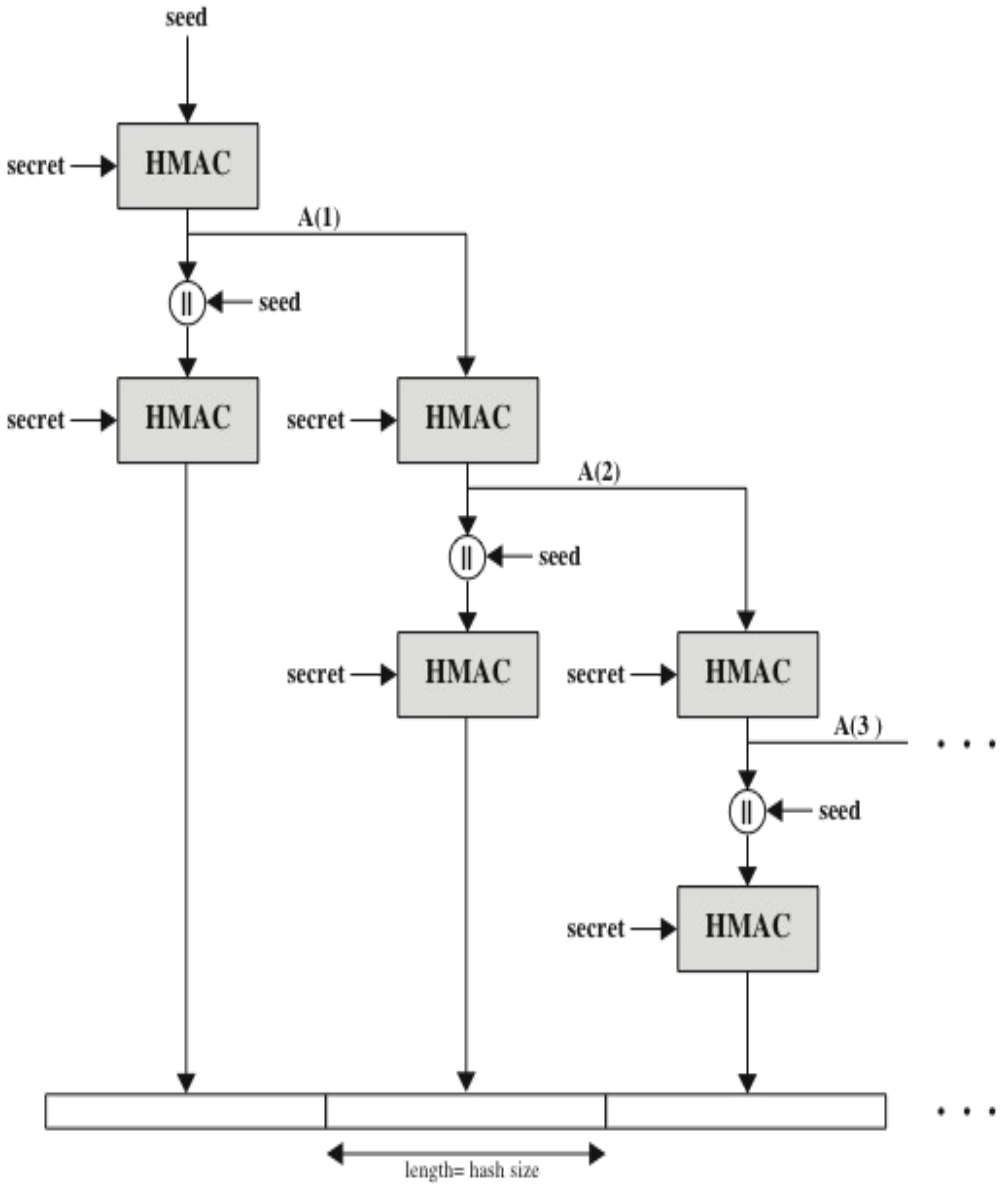


Figure 6.7 TLS Function P_hash (secret, seed)

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.