Introduction to Secure Sockets Layer

Introduction

Originally developed by Netscape Communications to allow secure access of a browser to a Web server, Secure Sockets Layer (SSL) has become the accepted standard for Web security.[1] The first version of SSL was never released because of problems regarding protection of credit card transactions on the Web. In 1994, Netscape created SSLv2, which made it possible to keep credit card numbers confidential and also authenticate the Web server with the use of encryption and digital certificates. In 1995, Netscape strengthened the cryptographic algorithms and resolved many of the security problems in SSLv2 with the release of SSLv3. SSLv3 now supports more security algorithms than SSLv2.

Scope

This paper is intended to serve as a primer for learning the basic concepts of how SSL operates. Overview information on how SSL termination devices are deployed in a Web server environment also is included. Because this paper is intended for a technical audience, a basic understanding of network infrastructure and security concepts is assumed.

SSL Basics

SSL Element

The main role of SSL is to provide security for Web traffic. Security includes confidentiality, message integrity, and authentication. SSL achieves these elements of security through the use of cryptography, digital signatures, and certificates.

Cryptography

SSL protects confidential information through the use of cryptography. Sensitive data is encrypted across public networks to achieve a level of confidentiality. There are two types of data encryption: symmetric cryptography and asymmetric cryptography (refer to Table 1).

Symmetric cryptography uses the same key for encryption and decryption. An example of symmetric cryptography is a decoder ring. Alice has a ring and Bob has the same ring. Alice can encode messages to Bob using her ring as the cipher. Bob can then decode the sent message using his ring. In cryptography, the “decoder ring” is considered a preshared key. The key is agreed upon by both sides and can remain static. Both sides must know each other already and have agreed upon what key to use for the encryption and decryption of messages. Remember that the same key is used for encoding as well as decoding messages—thus the term *symmetric cryptography.*

[1] Wireless Security (p.367) Nichols, Lekkas

Asymmetric algorithms use one key for encryption of data, and then a separate key for decryption. Asymmetric algorithms are more favorable than symmetric algorithms because even if the encryption key is learned in one direction, the third party still needs to know the other key in order to decrypt the message in the other direction.

Table 1 Symmetric Cryptography vs. Asymmetric Cryptography

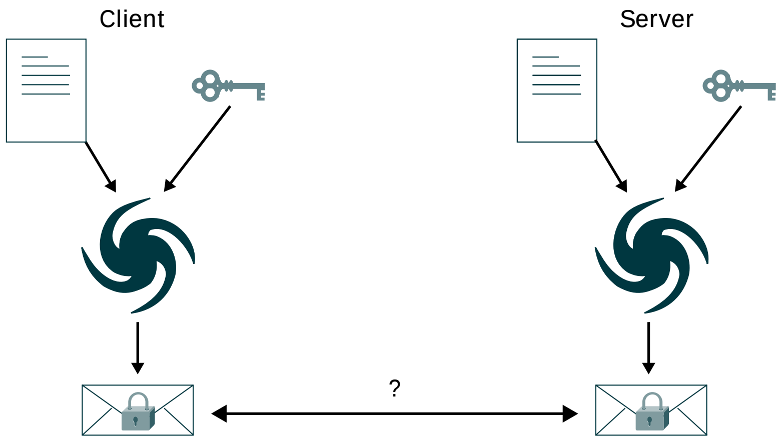
Symmetric Cryptography
• Symmetric cryptography uses a single key for encryption and decryption. • Symmetric cryptography requires that both parties have the key. • Key distribution is the inherent weakness in symmetric cryptography. • Minimal CPU cycles are required to verify keys. • Symmetric ciphers are fortified by algorithmic strength and key lengths. • SSL symmetric key lengths range from 40 to 168 bits.
Asymmetric Cryptography (PKI)
• Asymmetric cryptography was designed in response to the limitations of symmetric cryptography. • Information encrypted with one key can be decrypted only with another key. • Public key infrastructure (PKI) cryptography is up to 1000 times more CPU intensive than symmetric cryptography. • The Rivest, Shamir, Adelman (RSA) algorithm uses modular arithmetic to enable the concept of public and private keys. • All SSL transactions begin with an asymmetric key exchange.

With asymmetric encryption, both sides can spontaneously spawn a transaction without ever having met. This is achieved by the use of a public and private key pair. The public key of the entity is public knowledge and is used for encryption, whereas the private key of the entity remains secret and is used for decryption. PKI is the more common name for asymmetric cryptography. Although PKI is more secure, it also is more expensive in terms of processing speed. The encryption and decryption of the PKI can take up to 1000 times the processing than symmetric cryptography.

Digital Signatures

To ensure message integrity, each message exchanged in SSL has a digital signature attached to it. A digital signature is a hashed message digest with public key information. The message digest is based on the checksum of the message. The message digest is difficult to reverse. Both parties compute the message digest separately and compare the hashed results. Matching results means that the checksum was unaltered during transit, minimizing the chance of a compromised message (refer to Figure 1).

Figure 1. Digital Signatures[2]



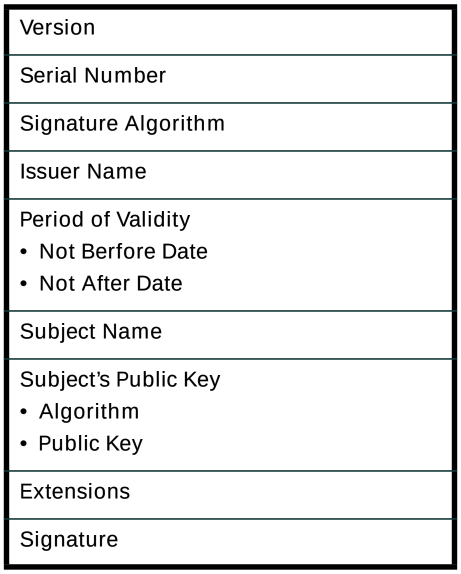
1. Client sends a message
2. Client has message and a public key
3. Client hashes message with public key
4. Server takes random message and knows public key
5. Server hashes message with public key
6. Server sends hashed message
7. Client compares its own hashed message to server's message
8. If the two match, then the message has not been tampered

[2] W3C Working Draft: "Digital Signature Label Architecture" WD-DSIG-label-arch-970610

Certificates

How do you trust the person to whom you are sending your message? SSL uses digital certificates to authenticate servers. (SSL also includes an optional authentication for clients.) Certificates are digital documents that will attest to the binding of a public key to an individual or other entity. They allow verification of the claim that a specific public key does, in fact, belong to the specified entity. Certificates help prevent someone from impersonating the server with a false key. SSL uses X.509 certificates to validate identities. X.509 certificates contain information about the entity, including public key and name. A certificate authority then validates this certificate (refer to Figure 2).

Figure 2. An X.509 Certificate



Version
Serial Number
Signature Algorithm
Issuer Name
Period of Validity • Not Berfore Date • Not After Date
Subject Name
Subject's Public Key • Algorithm • Public Key
Extensions
Signature

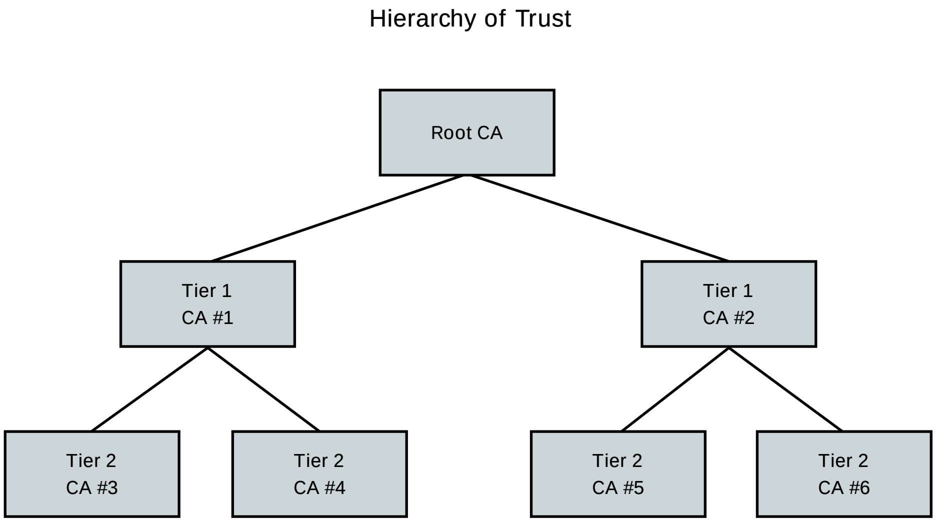
Certificate Authority

When you go to a bar or nightclub, security checks your ID to verify who you are. Your driver's license validates your ability to drive; more importantly, however, your driver's license is a trusted form of identity because your license was issued by a trusted third party. In the same way, a digital certificate is a mere statement of the identity of the body or individual who wishes to be authenticated. A trusted third party outside the server and client pair is needed to validate the certificate. This third party is the certificate authority. Reputable certificate authorities, such as VeriSign, are responsible for ensuring the trust of all World Wide Web entities.

Certificate Chaining

In some cases it may be necessary to create a chain of certificates, each one certifying the previous one until the parties involved are confident of the identity in question. This process is called certificate chaining. Certificate chaining is important in situations where the first line of certificate authorities may not be as well known or trusted as another certificate authority. A hierarchy of trust is formed (refer to Figure 3). This hierarchy of trust is vital to the authentication of an entity.

Figure 3. Hierarchy of Trust



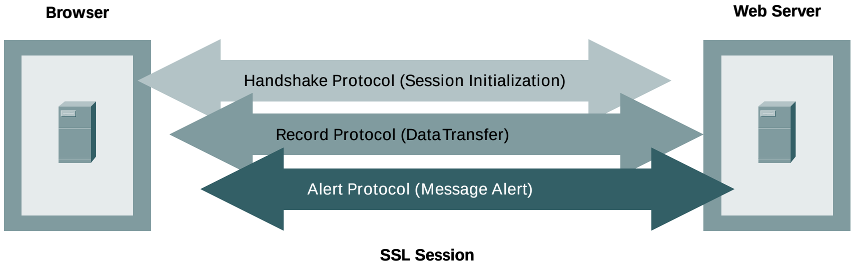
How SSL works

SSL Roles

SSL has two distinct entities, server and client. The client is the entity that initiates the transaction, whereas the server is the entity that responds to the client and negotiates which cipher suites are used for encryption. In SSL, the Web browser is the client and the Web-site server is the server.

Three protocols lie within SSL, the Handshake Protocol, the Record Protocol, and the Alert Protocol. The client authenticates the server during the Handshake Protocol. When the session is initiated and the handshake is complete, the data transfer is encrypted during the Record Protocol phase. If there are any alarms at any point during the session, the alert is attached to the questionable packet and handled according to the Alert Protocol (refer to Figure 4).

Figure 4. SSL Protocol Stack


Browser
Web Server
Handshake Protocol (Session Initialization)
Record Protocol (DataTransfer)
Alert Protocol (Message Alert)
SSL Session

SSL Handshake

The client always authenticates the server, and the server has the option of also authenticating the client. In general, Web servers do not authenticate the client during the Handshake Protocol because the server has other ways to verify the client other than SSL. For e-commerce, the Web-site server can verify the credit card number externally from the SSL session. In this way, the server can reserve precious processing resources for encrypted transactions.

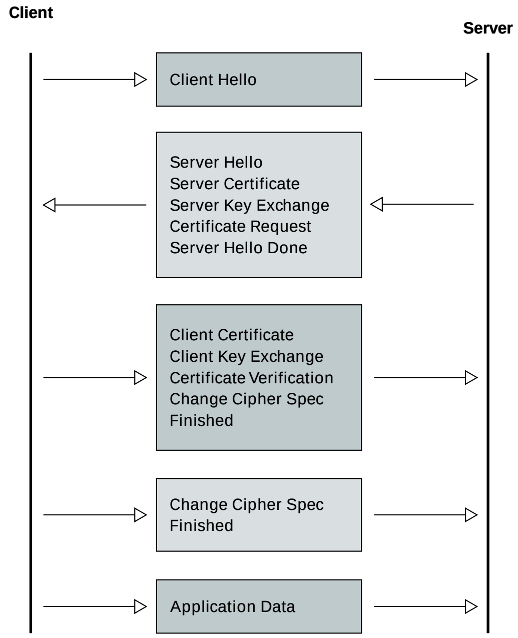
During the Handshake Protocol, the following important steps take place: the session capabilities are negotiated, meaning the encryption (ciphers) algorithms are negotiated; and the server is authenticated to the client.

SSL uses symmetric cryptography for the bulk data encryption during the transfer phase; however, asymmetric cryptography, (that is, PKI) is used to negotiate the key used for that symmetric encryption. This exchange is critical to the Handshake Protocol. Note that the server may optionally ask the client to authenticate itself. However, it is not necessary to the protocol. Table 2 and Figure 5 give the steps of the Handshake Protocol.

Table 2 Handshake Protocol

1.	Client sends *ClientHello* message.
2.	Server acknowledges with *ServerHello* message
3.	Server sends its certificate
4.	Optional: Server requests client's certificate
5.	Optional: Client sends its certificate
6.	Client sends *ClientKeyExhcange* message
7.	Client sends *Certificate Verify* message
8.	Both send *ChangeCipherSpec* messages
9.	Both send *Finished* messages

Figure 5. Handshake Protocol

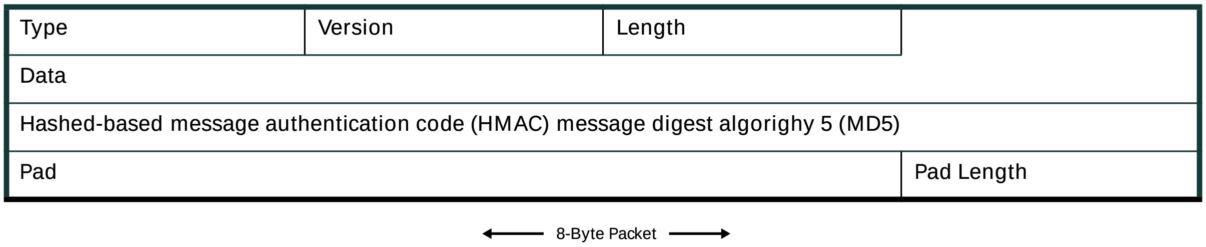


SSL Records

The encryption for all messaging in SSL is handled in the Record Protocol. This protocol provides a common format to frame all Alert, ChangeCiperSpec, Handshake, and application protocol messages.[3]

SSL records consist of the encapsulated data, digital signature, message type, version, and length. SSL records are 8 bytes long. Because the record length is fixed, encrypted messages sometimes include padding and pad length in the frame, as shown in Figure 6.

Figure 6. An Example of an SSL Record[4]



Type	Version	Length	
Data			
Hashed-based message authentication code (HMAC) message digest algorighy 5 (MD5)			
Pad			Pad Length

← 8-Byte Packet →

SSL Alert Protocol

As mentioned earlier, the Alert Protocol handles any questionable packets. If either the server or client detects an error, it sends an alert containing the error. There are three types of alert messages: warning, critical, and fatal. Based on the alert message received, the session can be restricted (warning, critical) or terminated (fatal).

Deploying SSL Termination Devices

Traditional Deployment

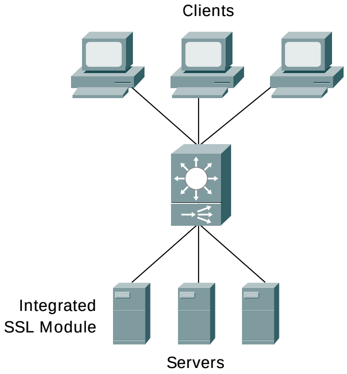
Servers with SSL NIC

The traditional deployment of SSL in a Web environment consisted of a Web server with an integrated SSL module (an SSL-enabled network interface card [NIC]). The client initiated a session with the server, and the server was directly responsible for the SSL termination. This process adds load to the server, which is already responsible for all Hypertext Transfer Protocol (HTTP) information that is sent to and received from the client. The Web server processor is shared across both the SSL processing and HTTP processing. Figure 7 displays a traditional SSL deployment.

[3] SSL and TLS Essentials: Securing the Web (p. 69) Thomas

[4] SSL and TLS Designing and Building Secure Systems (p.89) Rescorla

Figure 7. Traditional Deployed SSL



Bottleneck

Server performance is predicated on the performance of the system as a whole. Cryptography is expensive. The cost of using SSL can dramatically slow down the Web servers and, therefore, interfere with the Web site itself. The two relevant cryptographic operations of SSL occur during the data transfer phase. Remember that the data transfer phase occurs at the record protocol level in SSL. The SSL records are encrypted, and a digital signature of the Media Access Control (MAC) is included with each record transferred. The record encryption and the record MAC signature operations account for most of the cost during data transfer.[5]

The other process that slows down SSL is the public key cryptographic operations that are associated with the SSL Handshake Protocol. An SSL handshake occurs at the initiation of every SSL session requested of the server. The session key exchange accounts for most of the cost. One way to cut down on the number of handshakes is to use session resumption. This way, the server maintains a cache of clients. However, this cache can grow to be very large if the server is talking to a large number of clients. In this case, the memory and CPU can be at maximum capacity just to maintain session caches because a Web server may be handling hundreds of transactions a minute, meaning possibly hundreds of session resumptions as well. The ramifications include a slowdown in the Web server overall, in both HTTP and SSL transactions, which could turn into down time for the server as well. This slowdown—or even down time—translates into lost revenue that potentially can be generated during that time.

SSL Termination Devices

Offload Existing Servers

SSL accelerators can be implemented in a Web server infrastructure or deployment in order to offload the servers from the expensive part of SSL transactions. The SSL accelerator serves as a central point for negotiating handshakes and also encrypting and decrypting data. This allows the servers to process other HTTP processes unrelated to SSL and remove the load off the Web server. In most cases the SSL accelerator is typically connected to a content switch in the path between the client and the server. This external accelerator is commonly referred to as an SSL termination device. Other SSL accelerator modules can be integrated into the content switch for the termination of SSL traffic.

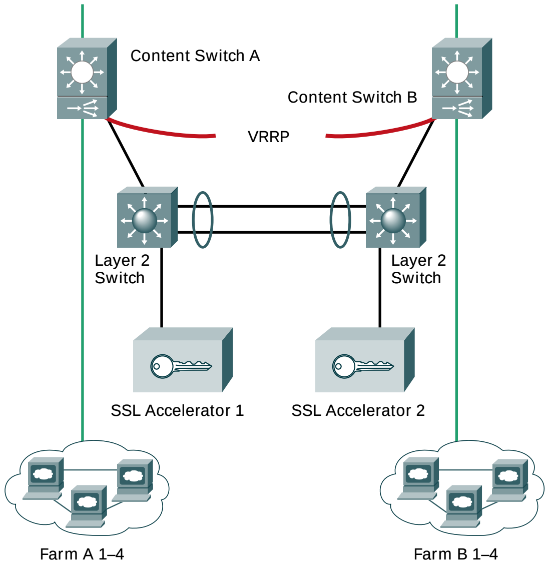
[5] SSL and TLS Designing and Building Secure Systems (p.180) Rescorla

The most recommended SSL design to deploy is called nontransparent, which is also referred to as "proxy mode SSL" and "nontransparent proxy mode." Transparent is sometimes referred to as "transparent proxy mode." Nontransparent mode scales well and is a more secure deployment design. In a nontransparent design, the client source address does not get to the server. This setup is beneficial because it provides privacy on the client side and scalability on the server side.

High Availability

Uptime for servers is critical for all business. For example, it is important to the business of an online shopping site to have high availability for its servers at all times—any down time can be considered a loss in revenue because purchase transactions cannot be completed. If an SSL accelerator fails, the connection should be load balanced to the next accelerator. In the high-availability configuration, shown in Figure 8, the user should experience no noticeable change in his or her session because any failure in equipment is automatically rerouted to the backup SSL accelerator.

Figure 8. High Availability



SSL in the Data Center

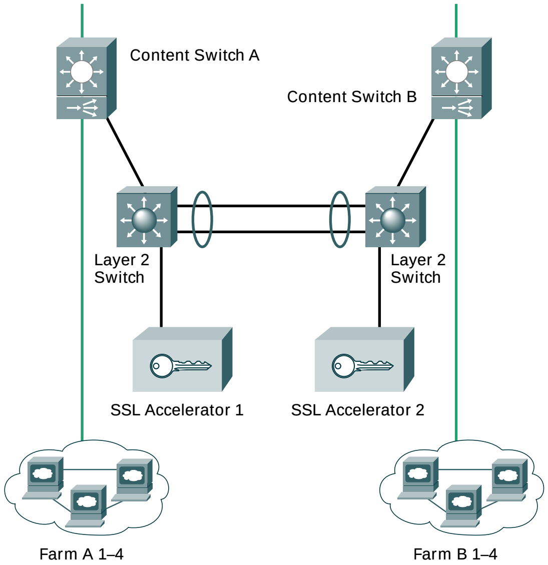
As stated previously, the most recommended design for deploying SSL is nontransparent because this architecture has the most flexibility and scalability. User tracking is done through cookies and optionally by having the SSL accelerator log the client addresses to a syslog server. This cache of client addresses allows the capability to not only track but also resume sessions that have been terminated gracefully. This saves processing overhead, thereby freeing more bandwidth for even more SSL transactions. SSL accelerators can also be shared between applications and content switches.

The reason it is called nontransparent is that the source addresses of all the packets decrypted by the SSL accelerator have a source address of that SSL accelerator. From the server perspective, the request came from the SSL accelerator. Some customers initially had a problem with this setup because they usually track the client's source address on the server. Some SSL termination devices still have the ability to send the client's source address to a syslog server for tracking purposes. However, with cookies, you can gather much more granular information about a client. In addition to IP address information, cookies can track where the client goes to and comes from on a Web site, as well as personal information (that is, passwords, shopping lists, and user preferences). Table 3 shows the packet flow for the nontransparent design, and Figure 9 shows the nontransparent design architecture.

Table 3 Packet Flow for Nontransparent Design Architecture

Packet Flow for Nontransparent Design Architecture
1. The client constructs a request (that is, builds a shopping cart).
2. The client clicks on a button that returns the client to the same virtual IP address on port 443.
3. The client comes back in on port 443 and hits an SSL content rule.
4. The traffic destination is port translated to the appropriate port within the accelerator.
5. Traffic is decrypted and sent out on port 80 to the original virtual IP.
6. The cookie that was set in the clear is used to get the client back to the same server on which that client's shopping cart resides.

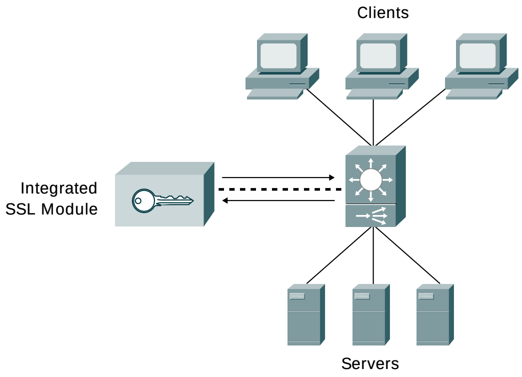
Figure 9. Nontransparent Design Architecture



SSL Termination Module

An SSL termination module is an external SSL accelerator that is co-located within a device other than the Web server system itself. The module is still solely responsible for all SSL transactions, but it resides within another device with ample memory and CPU processing speed. This other device can be either another router or a switch within the network. Today content switches that have SSL termination modules fully integrated are available (refer to Figure 10).

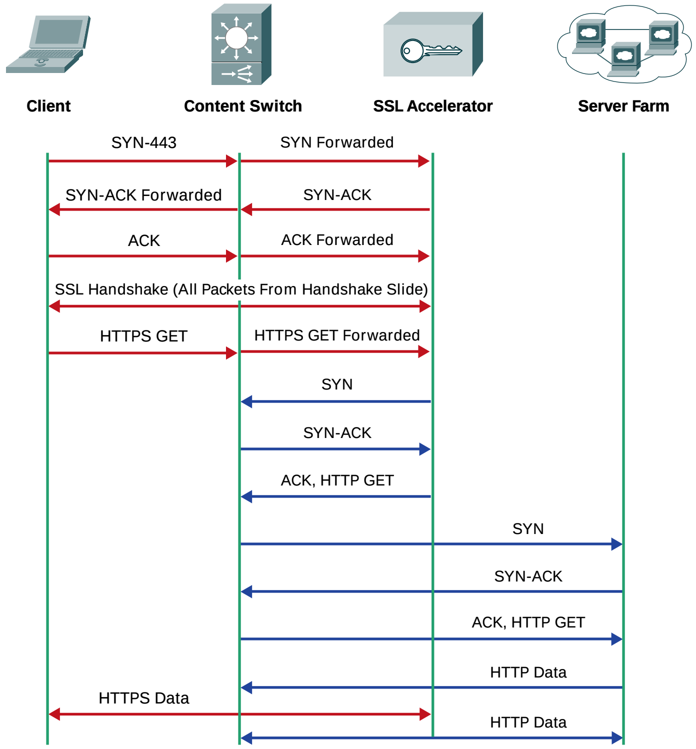
Figure 10. Switch with Integrated SSL Module



SSL Termination Appliance

An SSL accelerator that is connected to the content switch in the Web server environment is a termination device. This device is external from the other servers in the network and is solely responsible for SSL transactions. This is another way to offload the network by using an external device with more available resources (both memory and processing power) to handle SSL transactions. In this way, the termination appliance can be dedicated to handle the SSL transactions alone. This does not interfere with any network processing time on the switch or Web server. Figure 11 shows the packet flow using an external SSL termination device.

Figure 11. Packet Flow of SSL with External SSL Termination Appliance



Conclusion

SSL is vital to Web security. It provides a strong sense of confidentiality, message integrity, and server authentication to users. The business of e-commerce is tied closely to consumer confidence in the operation of SSL across the net. In the future, SSL termination devices will be able to handle more transactions at a faster rate. The encryption of key lengths and the cipher suites used will also continue to evolve in order to ensure the security of sensitive information over the Web. This way, e-commerce will be able to continue to grow in popularity as users grow more confidant in shopping and banking online, and embracing new online applications.

Cisco Systems

Corporate Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 526-4100

European Headquarters
Cisco Systems Europe
11 Rue Camille Desmoulins
92782 Issy-les-Moulineaux
Cedex 9
France
www-europe.cisco.com
Tel: 33 1 58 04 60 00
Fax: 33 1 58 04 61 00

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
www.cisco.com
Tel: 408 526-7660
Fax: 408 527-0883

Asia Pacific Headquarters
Cisco Systems, Inc.
Capital Tower
168 Robinson Road
#22-01 to #29-01
Singapore 068912
www.cisco.com
Tel: +65 317 7777
Fax: +65 317 7799

Cisco Systems has more than 200 offices in the following countries and regions. Addresses, phone numbers, and fax numbers are listed on the

Cisco Web site at www.cisco.com/go/offices

Argentina • Australia • Austria • Belgium • Brazil • Bulgaria • Canada • Chile • China PRC • Colombia • Costa Rica • Croatia
Czech Republic • Denmark • Dubai, UAE • Finland • France • Germany • Greece • Hong Kong SAR • Hungary • India • Indonesia • Ireland
Israel • Italy • Japan • Korea • Luxembourg • Malaysia • Mexico • The Netherlands • New Zealand • Norway • Peru • Philippines • Poland
Portugal • Puerto Rico • Romania • Russia • Saudi Arabia • Scotland • Singapore • Slovakia • Slovenia • South Africa • Spain • Sweden
Switzerland • Taiwan • Thailand • Turkey • Ukraine • United Kingdom • United States • Venezuela • Vietnam • Zimbabwe