HTTPS--HTTP Server and Client with SSL 3.0

The HTTPS--HTTP Server and Client with SSL 3.0 feature provides Secure Socket Layer (SSL) version 3.0 support for the HTTP 1.1 server and HTTP 1.1 client within Cisco IOS XE software. SSL provides server authentication, encryption, and message integrity to allow secure HTTP communications. SSL also provides HTTP client authentication. HTTP over SSL is abbreviated as HTTPS.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About HTTPS--HTTP Server and Client with SSL 3.0

To configure the HTTP with SSL 3.0 (HTTPS) feature, you should understand the following concepts:

Secure HTTP Server and Secure HTTP Client

A secure HTTP connection means that data sent to and received from an HTTP server are encrypted before being sent out over the Internet. HTTP with SSL encryption provides a secure connection to allow such functions as configuring a router from a web browser. Cisco's implementation of the secure HTTP server and secure HTTP client uses an implementation of the SSL version 3.0. Application layer encryption provides an alternative to older methods such as having to set up a tunnel to the HTTP server for remote management.

HTTP over SSL is abbreviated as HTTPS; the URL of a secure connection will begin with https:// instead of http://.

The Cisco IOS XE HTTP secure server's primary role is to listen for HTTPS requests on a designated port (the default HTTPS port is 443) and to pass the request to the HTTP 1.1 web server. The HTTP 1.1 server processes requests and passes responses (served pages) back to the HTTP secure server, which, in turn, responds to the original request.

The Cisco IOS XE HTTP secure client's primary role is to respond to Cisco IOS XE application requests for HTTPS User Agent services, perform HTTPS User Agent services on the application's behalf, and pass the response back to the application.

Certificate Authority Trustpoints

Certificate authorities (CAs) are responsible for managing certificate requests and issuing certificates to participating IPSec network devices. These services provide centralized security key and certificate management for the participating devices. Specific CA servers are referred to as "trustpoints."

The HTTPS server provides a secure connection by providing a certified X.509v3 certificate to the client when a connection attempt is made. The certified X.509v3 certificate is obtained from a specified CA trustpoint. The client (usually a web browser), in turn, has a public key that allows it to authenticate the certificate.

Configuring a CA trustpoint is highly recommended for secure HTTP connections. However, if a CA trustpoint is not configured for the routing device running the HTTPS server, the server will certify itself and generate the needed RSA key pair. Because a self-certified (self-signed) certificate does not provide adequate security, the connecting client will generate a notification that the certificate is self-certified, and the user will have the opportunity to accept or reject the connection. This option is available for internal network topologies (such as testing).

The HTTPS--HTTP Server and Client with SSL 3.0 feature also provides an optional command (**ip http secure-client-auth**) that, when enabled, has the HTTPS server request an X.509v3 certificate from the client. Authenticating the client provides more security than server authentication by itself.

For additional information on certificate authorities, see the "Configuring Certification Authority Interoperability" chapter in the Cisco IOS XE Security Configuration Guide .

CipherSuites

A CipherSuite specifies the encryption algorithm and digest algorithm to use on an SSL connection. Web browsers offer a list of supported CipherSuites when connecting to the HTTPS server, and the client and server will negotiate the best encryption algorithm to use from those that are supported by both. For example, Netscape Communicator 4.76 supports U.S. security with RSA Public Key Cryptography, MD2, MD5, RC2-CBC, RC4, DES-CBC, and DES-EDE3-CBC.

For the best possible encryption, you should use a browser that supports 128-bit encryption, such as Microsoft Internet Explorer version 5.5 (or later), or Netscape Communicator version 4.76 (or later). The SSL_RSA_WITH_DES_CBC_SHA CipherSuite provides less security than the other CipherSuites, because it does not offer 128-bit encryption.

In terms of router processing load (speed), the following list ranks the CipherSuites from fastest to slowest (slightly more processing time is required for the more secure and more complex CipherSuites):

1. SSL_RSA_WITH_DES_CBC_SHA
2. SSL_RSA_WITH_RC4_128_MD5

3. SSL_RSA_WITH_RC4_128_SHA
4. SSL_RSA_WITH_3DES_EDE_CBC_SHA

How to Configure the HTTPS--HTTP Server and Client with SSL 3.0

Declaring a Certificate Authority Trustpoint

Configuring a CA trustpoint is highly recommended for secure HTTP connections. The certified X.509v3 certificate for the secure HTTP server (or client) is obtained from the specified CA trustpoint. If you do not declare a CA trustpoint, then a self-signed certificate will be used for secure HTTP connections. The self-signed certificate is generated automatically.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **hostname** *name*
4. **ip domain-name** *name*
5. **crypto key generate rsa usage-keys**
6. **crypto ca trustpoint** *name*
7. **enrollment url** *url*
8. **enrollment http-proxy** *host-name port-number*
9. **crl** {**query** *url* | **optional** | **best-effort**}
10. **primary**
11. **exit**
12. **crypto ca authenticate** *name*
13. **crypto ca enrollment** *name*
14. Do one of the following:
    - **copy running-config startup-config**
    -
    - **copy system:running-config nvram:startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**hostname** *name* **Example:** `Device(config)# hostname Router`	Specifies the hostname of the Device. • This step is needed only if you have not previously configured a hostname for your Device. The hostname is required because a fully qualified domain name is needed for security keys and certificates.
Step 4	**ip domain-name** *name* **Example:** `Device(config)# ip domain-name example.com`	Specifies the IP domain name of the Device. • This step is needed only if you have not previously configured an IP domain name for your Device. The domain name is required because a fully qualified domain name is needed for security keys and certificates.
Step 5	**crypto key generate rsa usage-keys** **Example:** `Device(config)# crypto key generate rsa usage-keys`	(Optional) Generates an RSA key pair. • The **usage-keys** keyword specifies that two RSA special-usage key pairs should be generated (that is, one encryption pair and one signature pair) instead of one general-purpose key pair. • RSA key pairs are used to sign and encrypt Internet key exchange (IKE) key management messages and are required before you can obtain a certificate for your Device. • RSA key pairs are generated automatically. This command can be used to regenerate the keys, if needed. **Note** There are other keywords and arguments for this command, but they do not pertain to this feature.
Step 6	**crypto ca trustpoint** *name* **Example:** `Device(config)# crypto ca trustpoint TP1`	Specifies a local configuration name for the CA trustpoint and enters CA trustpoint configuration mode. **Note** The **crypto ca identity** command was replaced by the **crypto ca trustpoint** command.
Step 7	**enrollment url** *url* **Example:** `Device(ca-trustpoint)# enrollment url http://example.com`	Specifies a URL of the CA where your Device should send certificate requests. • If you are using Simple Certificate Enrollment Protocol (SCEP) for enrollment, the URL argument

	Command or Action	Purpose
		must be in the form **http://***CA-name* , where *CA-name* is the host Domain Name System (DNS) name or IP address of the CA trustpoint.
Step 8	**enrollment http-proxy** *host-name port-number* **Example:** `Device(ca-trustpoint)# enrollment http-proxy example.com 8080`	(Optional) Configures the Device to obtain certificates from the CA through an HTTP proxy server.
Step 9	**crl** {**query** *url* | **optional** | **best-effort**} **Example:** `Device(ca-trustpoint)# crl query ldap://example.com`	Configures the Device to request a certificate revocation list (CRL), make CRL checking optional, or perform CRL checking on a "best-effort" basis. • CRLs ensure that the certificate of the peer has not been revoked. • The **crl optional** command configures the Device to accept certificates even if the appropriate CRL cannot be downloaded. • Use the **crl query** *url* command to specify the Lightweight Directory Access Protocol (LDAP) URL of the CA server; for example, ldap://another-server.
Step 10	**primary** **Example:** `Device(ca-trustpoint)# primary`	(Optional) Specifies that this trustpoint should be used as the primary (default) trustpoint for CA requests. • Use this command if more than one CA trustpoint will be configured on this Device.
Step 11	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits CA trustpoint configuration mode and returns to global configuration mode.
Step 12	**crypto ca authenticate** *name* **Example:** `Device(config)# crypto ca authenticate TP1`	Authenticates the CA by getting the public key of the CA. • Use the same name that you used when declaring the CA in the **crypto ca trustpoint** command.
Step 13	**crypto ca enrollment** *name* **Example:** `Device(config)# crypto ca enrollment TP1`	Obtains the certificate from the specified CA trustpoint. • This command requests a signed certificate from the CA for each RSA key pair.
Step 14	Do one of the following:	Saves the configuration to NVRAM.

	Command or Action	Purpose
	• **copy running-config startup-config** • • **copy system:running-config nvram:startup-config** **Example:** `Device(config)# copy running-config startup-config`	• This command is required to save the certificates into NVRAM. If not used, the certificates would be lost at Device reload. **Note** To execute EXEC mode commands in global configuration mode, you can add the **do** keyword before the command. For example, instead of **copy running-config startup-config**, you could enter **do copy running-config startup-config**.

Configuring the HTTPS Server with SSL 3.0

To disable the standard HTTP server and configure the HTTPS server with SSL 3.0, complete the procedure in this section.

Before you begin

If a certificate authority is to be used for certification, you should declare the CA trustpoint on the routing device before enabling the secure HTTP server.

SUMMARY STEPS

1. **enable**
2. Device# **show ip http server status**
3. **configure terminal**
4. **no ip http server**
5. **ip http secure-server**
6. **ip http secure-port** *port-number*
7. **ip http secure-ciphersuite** [**3des-ede-cbc-sha**] [**rc4-128-sha**] [**rc4-128-md5**] [**des-cbc-sha**]
8. **ip http secure-client-auth**
9. **ip http secure-trustpoint** *name*
10. **end**
11. **show ip http server secure status**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Device# **show ip http server status** **Example:**	(Optional) Displays the status of the HTTP server. • If you are unsure whether the secure HTTP server is supported in the software image you are running,

	Command or Action	Purpose
	`Device# show ip http server status` **Example:**	enter this command and look for the line "HTTP secure server capability: {Present | Not present}". • This command displays the status of the standard HTTP server (enabled or disabled).
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**no ip http server** **Example:** `Device(config)# no ip http server`	Disables the standard HTTP server. **Note** When enabling the HTTPS server you should always disable the standard HTTP server to prevent insecure connections to the same services. This is a precautionary step (typically, the HTTP server is disabled by default).
Step 5	**ip http secure-server** **Example:** `Device(config)# ip http secure-server`	Enables the HTTPS server.
Step 6	**ip http secure-port** *port-number* **Example:** `Device(config)# ip http secure-port 1025`	(Optional) Specifies the port number that should be used for the HTTPS server. The default port number is 443. Valid options are 443 or any number in the range 1025 to 65535.
Step 7	**ip http secure-ciphersuite** [**3des-ede-cbc-sha**] [**rc4-128-sha**] [**rc4-128-md5**] [**des-cbc-sha**] **Example:** `Device(config)# ip http secure-ciphersuite rc4-128-sha rc4-128-md5`	(Optional) Specifies the CipherSuites (encryption algorithms) that should be used for encryption over the HTTPS connection. • This command allows you to restrict the list of CipherSuites that the server offers the connecting clients. For example, you may want to allow only the most secure CipherSuite to be used. • Unless you have a reason to specify the CipherSuites that should be used, or you are unfamiliar with the details of these CipherSuites, you should leave this command unconfigured and let the server and client negotiate the CipherSuite that they both support (this is the default).
Step 8	**ip http secure-client-auth** **Example:** `Device(config)# ip http secure-client-auth`	(Optional) Configures the HTTP server to request an X.509v3 certificate from the client in order to authenticate the client during the connection process.

	Command or Action	Purpose
		• In the default connection and authentication process, the client requests a certificate from the HTTP server, but the server does not attempt to authenticate the client. Authenticating the client provides more security than server authentication by itself, but not all clients may be configured for CA authentication.
Step 9	**ip http secure-trustpoint** *name* **Example:** `Device(config)# ip http secure-trustpoint trustpoint-01`	Specifies the CA trustpoint that should be used to obtain an X.509v3 security certificate and to authenticate the connecting client's certificate. • Use of this command assumes you have already declared a CA trustpoint using the **crypto ca trustpoint** command and associated submode commands. • Use the same trustpoint name that you used in the associated **crypto ca trustpoint** command.
Step 10	**end** **Example:** `Device(config)# end`	Ends the current configuration session and returns you to privileged EXEC mode.
Step 11	**show ip http server secure status** **Example:** `Device# show ip http server secure status`	Displays the status of the HTTP secure server configuration.

Verifying the Configuration of the HTTPS Server

To verify the configuration of the HTTPS server, connect to the router running the HTTPS server with a web browser by entering **https://***url* , where *url* is the IP address or hostname of the router. Successful connection using the **https** prefix (instead of the standard **http**) indicates that the HTTPS server is configured properly. If a port other than the default port is configured (using the **ip http secure-port** command), you must also specify the port number after the URL. For example:

```
https://209.165.202.129:1026
```

or

```
https://host.domain.com:1026
```

Generally, you can verify that the HTTPS server is configured and that you have a secure connection by locating an image of a padlock at the bottom of your browser window. Also note that secure HTTP connections have a URL that starts with "https:" instead of "http:".

Providing Additional Security and Efficiency

The configuration of the standard HTTP server applies to the secure HTTP server as well. To provide additional security and efficiency to both the standard HTTP server and the HTTPS server, complete the procedure in this section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip http path** *path-name*
4. **ip http access-class** *access-list-number*
5. **ip http max-connections** *value*
6. **ip http timeout-policy idle** *seconds* **life** *seconds* **requests** *value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip http path** *path-name* **Example:** `Device(config)# ip http path slot1:`	(Optional) Sets the base HTTP path for HTML files. • The base path is used to specify the location of the HTTP server files (HTML files) on the local system. Generally, the HTML files are located in system flash memory.
Step 4	**ip http access-class** *access-list-number* **Example:** `Device(config)# ip http access-class 20`	(Optional) Specifies the access list that should be used to allow access to the HTTP server.
Step 5	**ip http max-connections** *value* **Example:** `Device(config)# ip http max-connections 10`	(Optional) Sets the maximum number of concurrent connections to the HTTP server that will be allowed. The default value is 5.
Step 6	**ip http timeout-policy idle** *seconds* **life** *seconds* **requests** *value* **Example:**	(Optional) Sets the characteristics that determine how long a connection to the HTTP server should remain open. The characteristics are:

	Command or Action	Purpose
	`Device(config)# ip http timeout-policy idle 30 life 120 requests 100`	• **idle** --The maximum number of seconds the connection will be kept open if no data is received or response data cannot be sent out on the connection. Note that a new value may not take effect on any already existing connections. If the server is too busy or the limit on the **life** time or the number of **requests** is reached, the connection may be closed sooner. The default value is 180 seconds (3 minutes). • **life** --The maximum number of seconds the connection will be kept open, from the time the connection is established. Note that the new value may not take effect on any already existing connections. If the server is too busy or the limit on the idle time or the number of requests is reached, it may close the connection sooner. Also, because the server will not close the connection while actively processing a request, the connection may remain open longer than the specified **life** time if processing is occurring when the life maximum is reached. In this case, the connection will be closed when processing finishes. The default value is 180 seconds (3 minutes). The maximum value is 86,400 seconds (24 hours). • **requests** --The maximum limit on the number of requests processed on a persistent connection before it is closed. Note that the new value may not take effect on any already existing connections. If the server is too busy or the limit on the idle time or the life time is reached, the connection may be closed before the maximum number of requests are processed. The default value is 1. The maximum value is 86,400.

Configuring the HTTPS Client with SSL 3.0

To configure the HTTPS client with SSL 3.0, complete the procedure in this section.

Before you begin

The standard HTTP client and the secure HTTP client are always enabled.

A certificate authority is required for secure HTTP client certification; the following steps assume that you have previously declared a CA trustpoint on the routing device. If a CA trustpoint is not configured, and the remote HTTPS server requires client authentication, connections to the secure HTTP client will fail.

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **ip http client secure-trustpoint** *trustpoint-name*
4. **ip http client secure-ciphersuite** [**3des-ede-cbc-sha**] [**rc4-128-sha**] [**rc4-128-md5**] [**des-cbc-sha**]
5. **end**
6. **show ip http client secure status**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip http client secure-trustpoint** *trustpoint-name* **Example:** `Device(config)# ip http client secure-trustpoint trustpoint01`	(Optional) Specifies the CA trustpoint that should be used if the remote HTTP server requests client authentication. • Use of this command assumes you have already declared a CA trustpoint using the **crypto ca trustpoint** command and associated submode commands. • Use the same trustpoint name that you used in the associated **crypto ca trustpoint** command. • This command is optional if client authentication is not needed, or if a primary trustpoint has been configured. If the **ip http client secure-trustpoint** command is not used, the Device will use the primary trustpoint, as specified by the **primary**CA trustpoint configuration mode command.
Step 4	**ip http client secure-ciphersuite** [**3des-ede-cbc-sha**] [**rc4-128-sha**] [**rc4-128-md5**] [**des-cbc-sha**] **Example:** `Device(config)# ip http client secure-ciphersuite rc4-128-sha rc4-128-md5`	(Optional) Specifies the CipherSuites (encryption algorithms) that should be used for encryption over the HTTPS connection. • This command allows you to restrict the list of CipherSuites that the client offers when connecting to a secure HTTP server. For example, you may want to allow only the most secure CipherSuites to be used. • Unless you have a reason to specify the CipherSuites that should be used, or you are unfamiliar with the details of these CipherSuites, you should leave this command unconfigured and let the server and client negotiate the CipherSuite that they both support (this is the default).

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config)# end`	Ends the current configuration session and returns to privileged EXEC mode.
Step 6	**show ip http client secure status** **Example:** `Device# show ip http client secure status`	Displays the status of the HTTP secure server configuration.

Configuration Examples for the HTTPS--HTTP Server and Client with SSL 3.0 feature

The following example shows a configuration session in which the secure HTTP server is enabled, the port for the secure HTTP server is configured as 1025, and the remote CA trustpoint server “CA-trust-local” is used for certification.

```
Device# show ip http server status

HTTP server status: Disabled
HTTP server port: 80
HTTP server authentication method: enable
HTTP server access class: 0
HTTP server base path:
Maximum number of concurrent server connections allowed: 5
Server idle time-out: 600 seconds
Server life time-out: 600 seconds
Maximum number of requests allowed on a connection: 1
HTTP secure server capability: Present
HTTP secure server status: Disabled
HTTP secure server port: 443
HTTP secure server ciphersuite: 3des-ede-cbc-sha des-cbc-sha rc4-128-md5 rc4-12a
HTTP secure server client authentication: Disabled
HTTP secure server trustpoint:
Device# configure terminal

Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# ip http secure-server

Device(config)# ip http client secure-trustpoint CA-trust-local

Device(config)# ip http secure-port 1024

Invalid secure port value.
Device(config)# ip http secure-port 1025

Device(config)# ip http secure-ciphersuite rc4-128-sha rc4-128-md5
Device(config)# end


Device# show ip http server secure status
```

```
HTTP secure server status: Enabled

HTTP secure server port: 1025

HTTP secure server ciphersuite: rc4-128-md5 rc4-128-sha

HTTP secure server client authentication: Disabled

HTTP secure server trustpoint: CA-trust-local
```

In the following example, the CA trustpoint CA-trust-local is specified, and the HTTPS client is configured to use this trustpoint for client authentication requests:

```
Device# config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# crypto ca trustpoint CA-trust-local

Device(ca-trustpoint)# enrollment url http://example.com

Device(ca-trustpoint)# crl query ldap://example.com

Device(ca-trustpoint)# primary

Device(ca-trustpoint)# exit

Device(config)# ip http client secure-trustpoint CA-trust-local

Device(config)# end

Device# copy running-config startup-config
```

Additional References

The following sections provide references related to the HTTPS--HTTP Server and Client with SSL 3.0 feature.

Related Documents

Related Topic	Document Title
SSL 3.0	The SSL Protocol Version 3.0 *This document is available from various sources online.*
Standard Cisco Web Client	HTTP 1.1 Web Client
Standard Cisco Web Server	HTTP 1.1 Web Server

Related Topic	Document Title
Certification Authority Interoperability	• Configuring Certification Authority Interoperability • Certificate Autoenrollment • Certificate Enrollment Enhancements • Trustpoint CLI • Source Interface Selection for Outgoing Traffic with Certificate Authority

Standards

Standard	Title
No new or modified standards are supported by this feature.	--

Related MIBs

MIBs	MIBs Link
• No new or modified MIBs are supported by this feature and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Related RFCs

RFCs	Description
RFC 2616	Cisco's implementation of HTTP is based on RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1 .

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for HTTPS--HTTP Server and Client with SSL 3.0

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1: Feature Information for HTTPS--HTTP Server and Client with SSL 3.0

Feature Name	Releases	Feature Information
HTTPS--HTTP Server and Client with SSL 3.0	Cisco IOS XE Release 2.1	This feature provides Secure Socket Layer (SSL) version 3.0 support for the HTTP 1.1 server and HTTP 1.1 client within Cisco IOS XE software. SSL provides server authentication, encryption, and message integrity to allow secure HTTP communications. SSL also provides HTTP client authentication. This feature is supported only in Cisco software images that support SSL. Specifically, SSL is supported in “IPSec 56” and “IPSec 3DES” images (contains “k8” or “k9” in the image name). The following commands are introduced or modified in the feature or features documented in this module. • **debug ip http ssl error** • **ip http client secure-ciphersuite** • **ip http client secure-trustpoint** • **ip http secure-ciphersuite** • **ip http secure-client-auth** • **ip http secure-port** • **ip http secure-server** • **ip http secure-trustpoint** • **show ip http client secure status** • **show ip http server secure status**

Glossary

RSA--RSA is a widely used Internet encryption and authentication system that uses public and private keys for encryption and decryption. The RSA algorithm was invented in 1978 by Ron Rivest, Adi Shamir, and Leonard Adleman. The abbreviation RSA comes from the first letter of the last names of the three original

developers. The RSA algorithm is included in many applications, such as the web browsers from Microsoft and Netscape. The RSA encryption system is owned by RSA Security.

SHA --The Secure Hash Algorithm. SHA was developed by NIST and is specified in the Secure Hash Standard (SHS, FIPS 180). Often used as an alternative to Digest 5 algorithm.

signatures, digital--In the context of SSL, “signing” means to encrypt with a private key. In digital signing, one-way hash functions are used as input for a signing algorithm. In RSA signing, a 36-byte structure of two hashes (one SHA and one MD5) is signed (encrypted with the private key).

SSL 3.0--Secure Socket Layer version 3.0. SSL is a security protocol that provides communications privacy over the Internet. The protocol allows client and server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery. SSL uses a program layer located between the Internet’s HTTP and TCP layers. SSL is included as part of most web server products and as part of most Internet browsers.