Description of the Secure Sockets Layer (SSL) Handshake

SUMMARY

The Secure Sockets Layer (SSL) protocol uses a combination of public-key and symmetric-key encryption. Symmetric-key encryption is much faster than public-key encryption; however, public-key encryption provides better authentication techniques. An SSL session always begins with an exchange of messages called the SSL handshake. The handshake allows the server to authenticate itself to the client by using public-key techniques, and then allows the client and the server to cooperate in the creation of symmetric keys used for rapid encryption, decryption, and tamper detection during the session that follows. Optionally, the handshake also allows the client to authenticate itself to the server.

MORE INFORMATION

The steps involved in the SSL handshake are as follows (note that the following steps assume the use of the cipher suites listed in Cipher Suites with RSA Key Exchange: Triple DES, RC4, RC2, DES):

1. The client sends the server the client's SSL version number, cipher settings, session-specific data, and other information that the server needs to communicate with the client using SSL.
2. The server sends the client the server's SSL version number, cipher settings, session-specific data, and other information that the client needs to communicate with the server over SSL. The server also sends its own certificate, and if the client is requesting a server resource that requires client authentication, the server requests the client's certificate.
3. The client uses the information sent by the server to authenticate the server (see Server Authentication for details). If the server cannot be authenticated, the user is warned of the problem and informed that an encrypted and authenticated connection cannot be established. If the server can be successfully authenticated, the client proceeds to step 4.
4. Using all data generated in the handshake thus far, the client (with the cooperation of the server, depending on the cipher being used) creates the pre-master secret for the session, encrypts it with the server's public key (obtained from the server's certificate, sent in step 2), and then sends the encrypted pre-master secret to the server.
5. If the server has requested client authentication (an optional step in the handshake), the client also signs another piece of data that is unique to this handshake and known by both the client and server. In this case, the client sends both the signed data and the client's own certificate to the server along with the encrypted pre-master secret.
6. If the server has requested client authentication, the server attempts to authenticate the client (see Client Authentication for details). If the client cannot be authenticated, the session ends. If the client can be successfully authenticated, the server uses its private key to decrypt the pre-master secret, and then performs a series of steps (which the client also performs, starting from the same pre-master secret) to generate the master secret.
7. Both the client and the server use the master secret to generate the session keys, which are symmetric keys used to encrypt and decrypt information exchanged during the SSL session and to verify its integrity (that is, to detect any changes in the data between the time it was sent and the time it is received over the SSL connection).
8. The client sends a message to the server informing it that future messages from the client will be encrypted with the session key. It then sends a separate (encrypted) message indicating that the client portion of the handshake is finished.
9. The server sends a message to the client informing it that future messages from the server will be encrypted with the session key. It then sends a separate (encrypted) message indicating that the server portion of the handshake is finished.
10. The SSL handshake is now complete and the session begins. The client and the server use the session keys to encrypt and decrypt the data they send to each other and to validate its integrity.
11. This is the normal operation condition of the secure channel. At any time, due to internal or external stimulus (either automation or user intervention), either side may renegotiate the connection, in which case, the process repeats itself.

Description of the Client Authentication Process During the SSL Handshake

SUMMARY

This article describes the client authentication process during the Secure Sockets Layer (SSL) handshake.

MORE INFORMATION

SSL-enabled servers can be configured to require client authentication, or cryptographic validation by the server of the client's identity. When a server that is configured this way requests client authentication, the client sends the server both a certificate and a separate piece of digitally signed data to authenticate itself. The server uses the digitally signed data to validate the public key in the certificate and to authenticate the identity that the certificate claims to represent.

The SSL protocol requires the client to create a digital signature by creating a one-way hash from data that is generated randomly during the handshake and that is known only to the client and server. The hash of the data is then encrypted with the private key that corresponds to the public key in the certificate that is being presented to the server.

An SSL-enabled server goes through the following steps to authenticate a user's identity:

1. Does the user's public key validate the user's digital signature? The server checks whether the user's digital signature can be validated with the public key in the certificate. If so, the server has established that the public key asserted to belong to the user matches the private key that is used to create the signature and that the data has not been tampered with since it was signed.

   At this point, however, the binding between the public key and the Distinguished Name (DN) that is specified in the certificate has not yet been established. The certificate might have been created by someone attempting to impersonate the user. To validate the binding between the public key and the DN, the server must also complete step 3 and step 4.

2. Is today's date within the validity period? The server checks the certificate's validity period. If the current date and time are outside of that range, the authentication process does not go any further. If the current date and time are within the certificate's validity period, the server goes on to Step 3.

3. Is the issuing Certificate Authority (CA) a trusted CA? Each SSL-enabled server maintains a list of trusted CA certificates. This list determines which certificates the server will accept. If the DN of the issuing CA matches the DN of a CA on the server's list of trusted CAs, the answer to this question is yes, and the server goes on to Step 4. If the issuing CA is not on the list, the client is not authenticated unless the server can verify a certificate chain ending in a CA that is on the list. Administrators can control which certificates are trusted or not trusted within their organizations by controlling the lists of CA certificates that are maintained by clients and servers.

4. Does the issuing CA's public key validate the issuer's digital signature? The server uses the public key from the CA's certificate (which it found in its list of trusted CAs in step 3) to validate the CA's digital signature on the certificate that is being presented. If the information in the certificate has changed since it was signed by the CA, or if the public key in the CA certificate does not correspond to the private key that was used by the CA to sign the certificate, the server does not authenticate the user's identity. If the CA's digital signature can be validated, the server treats the user's certificate as a valid "letter of introduction" from that CA and proceeds. At this point, the SSL protocol allows the server to consider the client authenticated and proceed with the connection in the next step.

5. Is the authenticated client authorized to access the requested resources? The server checks what resources the client is permitted to access according to the server's access control lists (ACLs) and

establishes a connection with appropriate access. If the server does not get to step 5 for any reason, the user that is identified by the certificate cannot be authenticated, and the user is not allowed to access any server resources that require authentication.

Description of the Server Authentication Process During the SSL Handshake

SUMMARY

This article describes the server authentication process during the Secure Sockets Layer (SSL) handshake.

MORE INFORMATION

During the SSL handshake, the server sends the client a certificate to authenticate itself. The client uses the certificate to authenticate the identity the certificate claims to represent.

An SSL-enabled client goes through these steps to authenticate a server's identity:

1. Is today's date within the validity period? The client checks the server certificate's validity period. If the current date and time are outside of that range, the authentication process does not go any further. If the current date and time are within the certificate's validity period, the client goes on to step 2.

2. Is the issuing Certificate Authority (CA) a trusted CA? Each SSL-enabled client maintains a list of trusted CA certificates. This list determines which server certificates the client will accept. If the distinguished name (DN) of the issuing CA matches the DN of a CA on the client's list of trusted CAs, the answer to this question is yes, and the client goes on to step 3. If the issuing CA is not on the list, the server is not authenticated unless the client can verify a certificate chain ending in a CA that is on the list.

3. Does the issuing CA's public key validate the issuer's digital signature? The client uses the public key from the CA's certificate (which it found in its list of trusted CAs in step 2) to validate the CA's digital signature on the server certificate that is being presented. If the information in the server certificate has changed since it was signed by the CA, or if the CA certificate's public key doesn't correspond to the private key that was used by the CA to sign the server certificate, the client does not authenticate the server's identity. If the CA's digital signature can be validated, the client treats the server's certificate as a valid "letter of introduction" from that CA and proceeds. At this point, the client has determined that the server certificate is valid. It is the client's responsibility to take step 4 before it takes step 5.

4. Does the domain name in the server's certificate match the domain name of the server itself? This step confirms that the server is actually located at the same network address that is specified by the domain name in the server certificate. Although step 4 is not technically part of the SSL protocol, it provides the only protection against a form of security attack known as a "Man-in-the-Middle Attack." Clients must perform this step and must refuse to authenticate the server or establish a connection if the domain names do not match. If the server's actual domain name matches the domain name in the server certificate, the client goes on to step 5.

5. The server is authenticated. The client proceeds with the SSL handshake. If the client does not get to step 5 for any reason, the server that is identified by the certificate cannot be authenticated, and the user is warned of the problem and informed that an encrypted and authenticated connection cannot be established.