SSL: A discussion of the Secure Socket Layer
By Steve Fewer

Table Of Contents

1 Introduction

The secure socket layer is the protocol that gives e-commerce the confidence it needs to allow on-line banking and shopping. Netscape originally developed the SSL protocol that is now the de facto standard for normal users who wish to maintain secure transactions. The current version of the protocol is SSL v3.0. SSL provides and encrypted bi-directional data stream. Data is encrypted at the sender's end and decrypted at the receiver's end. SSL provides for a two-way verification of both the server and the client. Therefor each end of the stream can be verified before any data exchange may take place. Due to the methods SSL employs data on the stream can not by hijacked or altered without detection.

SSL is commonly used for secure HTTP connections where credit card information is going to be sent along a network. HTTP uses TCP/IP to establish and maintain a connection. TCP/IP employs no security measures to prevent data sniffing, hijacking or insertion and is unsuitable on its own for connections containing sensitive information. SSL sits in-between high level protocols (such as HTTP or IMAP) and the low-level protocols of TCP over IP providing the needed security for such transactions to take place. It uses TCP/IP on behalf of the higher level protocols and once the correct client/server software is installed can appear transparent. See figure 1.1 for diagram

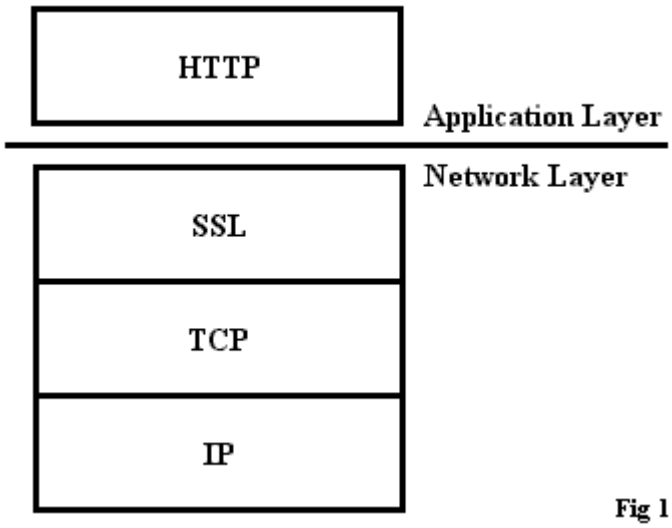


Fig 1.1

2 Encryption Techniques

The secure socket layer utilises many cryptographic techniques to provide an advanced level of security. It relies on every algorithm used being secure otherwise the entire sessions security is compromised. A full discussion on every cryptographic method available is beyond the scope of this paper however a brief overview is required. The two types of ciphers used are symmetrical and asymmetrical. A symmetrical cipher uses only one key for both encryption and decryption therefore both the parties must know the key and if found by an attacker would render the security void. Symmetrical ciphers are faster then asymmetrical ciphers and if employed properly with the aid of asymmetrical ciphers are a good solution. Examples of symmetrical encryption algorithms are the Data Encryption Standard (DES) and RC4. Asymmetrical ciphers use two keys, a public key and a private key. Only the owner knows the private key but the corresponding public key is available to everybody who wants to know. To send data securely to a host SSL encrypts it with the receivers public key and only the correct private key will decrypt it, the assumption lies in only the receiver knowing its own private key. An example of an asymmetrical encryption algorithm is RSA. A Hashing algorithm produces a checksum of a block of data. This can be used to check for any altercation or corruption during transit. An example of a hashing algorithm is MD4.

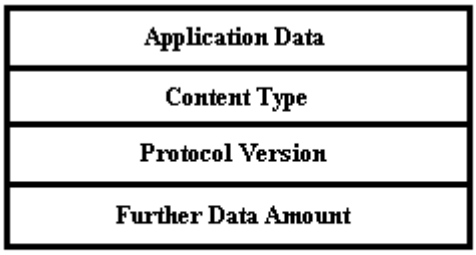
3 Protocol Overview

The secure socket layer protocol itself is made up from two sub protocols:

3.1 The SSL Record Protocol

The SSL Record protocol defines the method employed to transmit data. All data sent during a SSL session is transmitted in a series of records. A Record is a data structure with a maximum size of 16K bytes. The four fields of a record data structure are given in Fig 3.1. The content field indicates what the record data is for. There are four possible content types.

- A request to change the currently employed cipher.
- An alert message to indicate an invalid certificate or data corruption.
- SSL handshake information
- Application data



SSL Record Data Structure **Fig 3.1**

If compression is supported the record will be compressed. Every record transmitted to a host will be encrypted with an agreed cipher. Hashing the application data will generate a Message Authentication Code (MAC). This will aid detection of altered or corrupt data.

3.2 The SSL Handshake Protocol

The SSL Handshake protocol uses the record protocol to perform a two-way handshake. This handshake must be performed before the connection may continue. If the handshake fails the connection will be torn down.

4 Protocol Fundamentals

There are two fundamental features that make up the secure socket layer protocol, SSL Authentication and SSL Encrypted Connection.

4.1 SSL Authentication

SSL Authentication allows one end of the connection to verify the identity of the corresponding end. On the Internet ones identity can be verified by possessing a valid certificate and public ID which has been awarded by a company recognised by the software making the query to be trustworthy. Companies, known as trusted certificate authorities (e.g. Verisign) sell certificates to registered businesses whom wish to be able to prove their identity to clients.

4.2 SSL Encrypted Connection

This makes all data along the connection to be encrypted by the sending software and decrypted by the receiving software. The algorithms employed for the encryption process are defined during the handshake, as there are numerous possibilities available and each offering varying degrees of strength and speed. What algorithms are available to be used depends on where the software was purchased due to the American export restrictions on encryption. Encrypting the data allows for a high degree of confidentiality against it being sniffed along a network segment. The encrypted connection also employs methods to automatically determine whether the data has been tampered along route. This prevents attacks such as packet sniffing, connection hijacking and packet insertion.

5 The SSL Handshake

Each time an SSL session is initiated an exchange of messages, known as the handshake, must be performed. This handshake allows the server to authenticate itself to the client and optionally allows the client

to authenticate itself to the server. After authentication the client and server co-operate to generate symmetrical session keys which will be used for encryption, decryption and tamper detection throughout the session. The handshake may also be initiated at any time during a given session to re authenticate the two hosts and generate new cryptographic settings. The handshake uses public key encryption to communicate securely. The steps taken in performing the handshake are listed below:

1. A client wishing to initiate a secure socket layer connection with a server begins the session by connection to the server and sending its SSL version number, cipher settings, any compression methods available, session id and a structure of data containing a random 28 byte number and the system time. The session id will be zero if this is the first time the client has attempted to connect to the server, if it is not the first time the previous session id may be used to reduce the time take with the handshake as the old settings can be used. New session keys will still be generated. The server may choose to not allow old settings and will insist on a full handshake to take place.

2. The server will respond by sending its SSL version number, cipher settings and randomly generated data. The server will now transmit its own certificate to prove its identity. If the server wishes it will ask the client to send its certificate, this is optional however. The server will also transmit a session id for this connection.

3. Upon receiving the information from the server the client will proceed to authenticate the server. If the server fails to be authenticated the user is warned and a secure connection can not be established. If the servers identity can be authenticated the client will generate what is called the "premaster secret", a 48 byte random number. Once generated this key will be encrypted with the server's public key that can be obtained from its certificate and transmitted to the server. If the server requested client authentication in step two the client will also transmit its own certificate along with the premaster key. The client will now generate the "master secret" from the premaster secret using a hash algorithm.

4. If the server has requested to authenticate the client it will receive the clients certificate and proceed to authenticate the client's identity. If the server fails to authenticate the client the session will be dropped. If the clients identity can be confirmed the server will decrypt the premaster key that the client sent with the server's private key. The server will then proceed to generate the "master secret" from the premaster key.

5. At this stage both the client and the server will have a copy of the master secret, which itself has never been transmitted over the wire. Using this master secret both parties will generate the "session keys", which are symmetrical keys unique to this session. The session keys are used to encrypt and decrypt data being transmitted during the session. They also provide integrity checking so as altered data can be detected.

6. The client will now send a message to the server indicating that all further messages will now be encrypted using the newly generate session key. The client informs the server that it has finished its portion of the handshake by sending a further encrypted message.

7. The server will also send a message to the client saying that all further messages will be encrypted with the session key. The server will then finish the handshake by sending a final encrypted message to the client.

After these steps have been completed the handshake is complete and the SSL session has begun. All further data sent to either the client or server will be encrypted with the session key, which will also allow for the data to be decrypted, as it is a symmetrical key. The handshake relies on public key cryptography to ensure that the initial premaster secret remains a secret. Therefor the initial host authentication is vital to the session remaining secure.

6 Server Authentication

The most important part of the handshake is the authentication of the server. If this is corrupt all further generation of session keys will be corrupt and the entire SSL session will be insecure. The server is authenticated via a digital certificate that it sends the client in step two of the handshake. The clients will then proceed in step three of the handshake to validate the identity of the host that the certificate claims to

represent. A digital certificate will contain the fields as represented in Fig 6.1

Servers Certificate Fig 6.1

For the authentication to succeed four fundamental questions must be proved true. These questions are:

1. Is the current date within the certificates validity period? The client must check to see if the current date falls within the bounds of the certificates validity period. If it is not the handshake will fail and the session will be torn down. This is to check for expired certificates.

2. Is the issuing certificate authority a trusted certificate authority? Every client capable of using the SSL protocol has a list of trusted certificate authority certificates. This list will govern which server certificates the client will accept. If the servers issuing certificate authority is not present on the clients list the client will try to traverse a tree of certificate authorities in attempt to find the issuing certificate authority. Should this fail the handshake will fail and the session will be torn down.

3. Does the certificate authorities public key verify its own digital signature? Once the client has found the servers issuing certificate authority the client will validate its digital signature by using the public key from its own list and the digital signature from the certificate the server sent. If the client fails to validate the servers issuing certificate authority the handshake will fail and the session will be torn down. If the client can validate the issuing certificate authority the servers certificate is deemed to be valid and the handshake will continue.

4. Is the domain name in the certificate the actual domain name of the server? This step will simply check to see if the domain name given in the server's certificate is the actual domain name to which the client is connected. This step is to prevent a man-in-the-middle attack. If this step fails the session will be torn down.

Upon successful completion of these four steps the server's identity has been validated and generation of session keys can commence. The issuing of certificates also allows for public keys to be given to the client so the premaster secret can be encrypted before being transmitted to the server (step three of the handshake). Further reassurance of the server's identity lies in the knowledge that only the correct private key may decrypt the premaster secret. If the server wishes to authenticate the client the same four questions must be proved true, this is an optional procedure that would take place in stage four of the handshake.

7 Security Weaknesses in SSL

Although SSL is a solid protocol for transmitting sensitive information securely there are still a number of sophisticated attacks against it. There also have been a number of faulty implementations developed that lend themselves to specific attacks towards that implementation (e.g. Netscape developed a poor random number generator which could be predicted to revel the master key for a session.). Another security concern is the cryptographic strength of the ciphers being used. Due to severe export restrictions employed by America weak encryption is sold to countries such as Ireland. The maximum size allowed to be exported from the US is a 40-bit key. This can obviously be avoided if the cryptographic software is developed outside America. It is

also important to note that all versions of SSL below 3.0 have been cracked. But 128-bit encryption is employed in version three and has yet to be publicly cracked. There is also a more sophisticated network attack against SSL known as a man-in-the-middle attack. To prevent this the servers domain name must be verified as being its legitimate one as laid down in its digital certificate. The attack lets a rouge host sit in-between the client and server masquerading as the other while having full control over the session as all session keys are known to the attacker. Data may be observed and altered unnoticed to either legitimate host.

8 Bibliography

- An Introduction to SSL - Baltimore Security
  http://www.baltimore.com/products/jssl/sslintro.html

- SSL Torn Apart - Ankit Fadia
  http://neworder.box.sk

- Introduction to SSL - Netscape
  http://www.netscape.com