What is SSL?

The Secure Socket Layer Protocol (SSL) is a protocol designed to establish private communications over a computer network. The newest rendition, Transport Layer Security (TLS) is also referred to as SSL. SSL is the technology which allows the sending and receiving of sensitive information such as email and private messages, or shopping and banking over the internet with (relative) security. Without SSL, the use of these features over the internet would be severely discouraged. However, an encrypted connection does not mean it is secure.

Arguably, the most important aspect of SSL is verification of the server. That is to say, the server is who it says it is. During the connection negotiation, the server *usually* sends its Certificate, which is then challenged by the client. Using its list of trusted Certificates provided by Certificate Authorities, the client can verify whether or not the server is trustworthy.

How does SSL work?

SSL uses symmetric cryptography (the same key is used for encrypting and decrypting) in order to disguise meaningful bytes transmitted over a computer network into something that looks more or less like random noise. The challenge is, in order to establish a connection both parties must agree on a key to use for encrypting and decrypting data, and this is where things get interesting. A naive solution to establish a connection where both parties have the same key might be to simply pass a key from one device to another over the network. However, once the key is transmitted over the network it cannot be assumed that the intended party was the only one that received the key. Since the key is used for both encryption and decryption, an uninvited party could decipher data that was not intended for it. So then, how is it possible for two devices to have the same key without one exchanging it with another?

How does SSL negotiate a connection?

SSL uses a handshake protocol (TLS Handshake) in order to exchange the needed information to establish a secure connection.

The steps are generally as follows:

The handshake starts with the client who sends a Client Hello message. Within this message is the TLS protocol version the client supports, a random number, a list of cipher suites and supported compression algorithms.

The server then responds with its Server Hello message, of which specifies the chosen highest TLS protocol version it supports (capped by the client), another random number, the cipher suite it chose, and

The compression method chosen.

The server *should* then sends its Certificate, so that the client may validate it.

The server then sends its Server Hello Done message which indicates the end of the servers portion of the handshake.

The client then responds with a Client Key Exchange message, which contains the Pre-master secret which is encrypted with the Server's public key.

Using the random numbers and the Pre-master secret, the client and server compute the Master Secret.

The client then sends a Change Cipher Spec message which signals that messages from now on will be encrypted.
Finally the client sends a Finished message which is encrypted and contains a hash and MAC of the previous messages.

The server then deciphers the Finished message and verifies the hash and MAC.

Then the server sends a Change Cipher Spec message, which signals that from now on all messages will be encrypted.

Finally, the server sends its Finished message.

What kinds of certificates are there?

Aside from normal (boring) SSL certificates, there are certificate types called Wildcard, Multi-Domain and Subject Alternative Name (SAN). These types of Certificates allow multiple domains to be included under a single Certificate.

What is a CA trust chain?

A trust chain is a chain of trusted Certificate Authorities, where the top of the chain is a Certificate Root Authority, and the bottom is the Certificate to be validated. When a client wants to validate a Certificate, it must work its way up the trust chain in order to determine that the Certificate can be trusted.

What is the validation process?

Suppose you are browsing https://www.google.com. Your web browser (or sometimes your operating system) comes with a list of Trusted Certificates, which contain the public key of the corresponding domain. If at the end of the CA chain is a trusted Certificate Root Authority, then the Certificate should be valid and should be trustworthy.

The following is the certificate Hierarchy as presented in Firefox.
The Fields exist nested under the Certificate 'tab'.

USERTrust RSA Certification Authority
- InCommon RSA Server CA
- laulima.hawaii.edu
- Certificate
  - Version
  - Serial Number
  - Certificate Signature Algorithm
  - Issuer
  - Validity
  - Not Before
  - Not After
  - Subject
  - Subject Public Key Info
  - Subject Public Key Algorithm

Certificate Basic Constraints
- Extended Key Usage
- Certificate Policies
- CRL Distribution Points
- Authority Information Access
- Certificate Subject Alt Name
- Certificate Signature Algorithm
- Certificate Signature Value