SSL Socket Communication

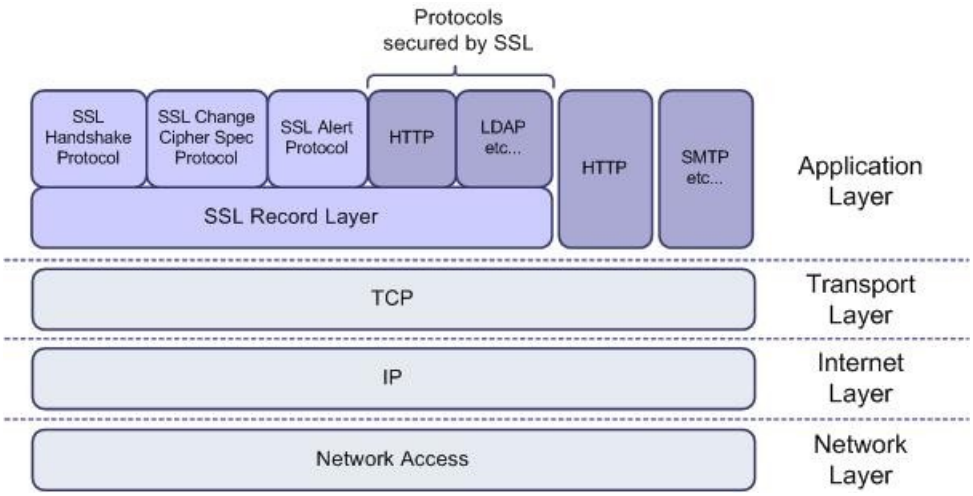
Secure Sockets Layer (SSL) is the most widely used protocol for implementing cryptography on the Web. SSL uses a combination of cryptographic processes to provide secure communication over a network. It negotiates the cryptography algorithms and keys between two sides of a communication, and establishes an encrypted tunnel through which other protocols (like HTTP) can be transported. Optionally, SSL can also authenticate both sides of communication through the use of certificates.

SSL Protocol Stack

SSL is a layered protocol and consists of four sub-protocols:

- •SSL Handshake Protocol
- •SSL Change Cipher Spec Protocol
- •SSL Alert Protocol
- •SSL Record Layer

The position of the above protocols according to the TCP/IP model has been illustrated on the following diagram



As the above diagrams shows, SSL is found in the application layer of the TCP/IP model. By dint of this feature, SSL can be implemented on almost every operating system that supports TCP/IP, without the need to modify the system kernel or the TCP/IP stack. This gives SSL a very strong advantage over other protocols like IPSec (IP Security Protocol), which requires kernel support and a modified TCP/IP stack. SSL can also be easily passed through firewalls and proxies, as well as through NAT (Network Address Translation) without issues.

How Does SSL Work?

There are manly two kinds SSL communications out there.

SSL without Client Authentication

In this case client does not need to prove its identity to the server. It basically says, client can be anyone but client and server communication will be confidential. In this case client does not have their own certificate. It relies on the server's certificate to encrypt the message in this communication. Please note we are not talking about client's user name and password. User name and password are still may be required by the web application to authenticate the user. Here we are talking in terms of SSL. This is the most common in internet world. As an example, when you visit your bank's website and try to login, the bank web server does not need any certificate from you. At a high level this is how it works when there is no client authentication.

1. Server presents client a server certificate and client authenticates server;
2. Client generates premaster secret and encrypt it with the server’s public key (contained in the server certificate), and send it to server.
3. Server decrypts the encrypted premaster secret.

4. Client and server both generates a master secret from the premaster secret, and then generate session keys from the master secret. The session keys are symmetric keys used to encrypt and decrypt information exchanged during the SSL session and to verify its integrity.

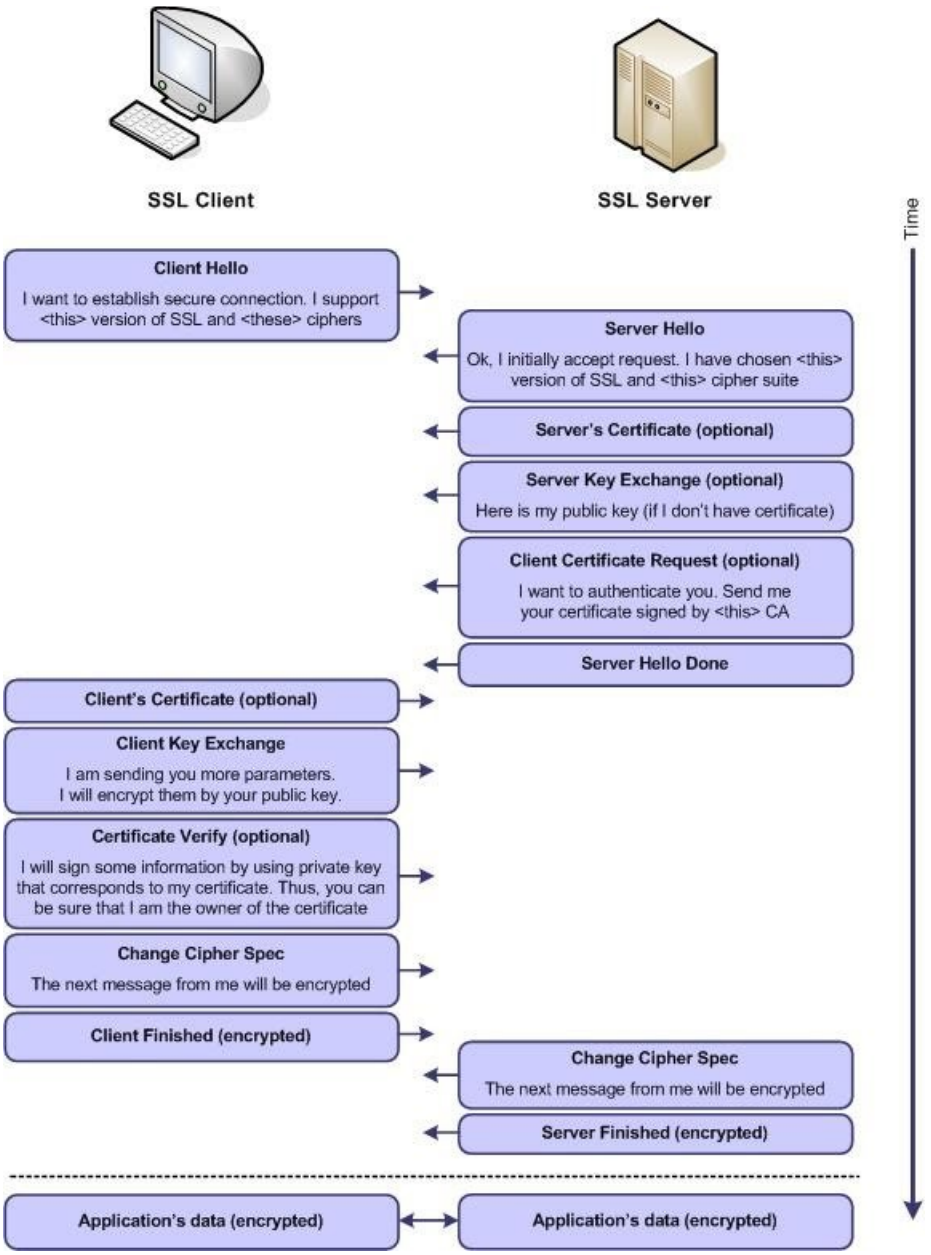
After this handshake, the server and client can communicate using encrypted messages that can only be decrypted by each other. Please note these above steps are subset of the steps shown in above diagram in "How Does SSL Work?" section.

SSL with Client Authentication

The server requests a certificate from the client, so that the connection can be mutually authenticated. Client needs to present its own digital certificate to server and also sign a handshake messages using the client's certificate's private key. This signature can be verified by using the client's certificate's public key. This lets the server know that the client has access to the private key of the certificate and thus owns the certificate. Apart form this client certificate validation the rest of the steps remain same. In this time type of communication client's user name and password verification becomes redundant.

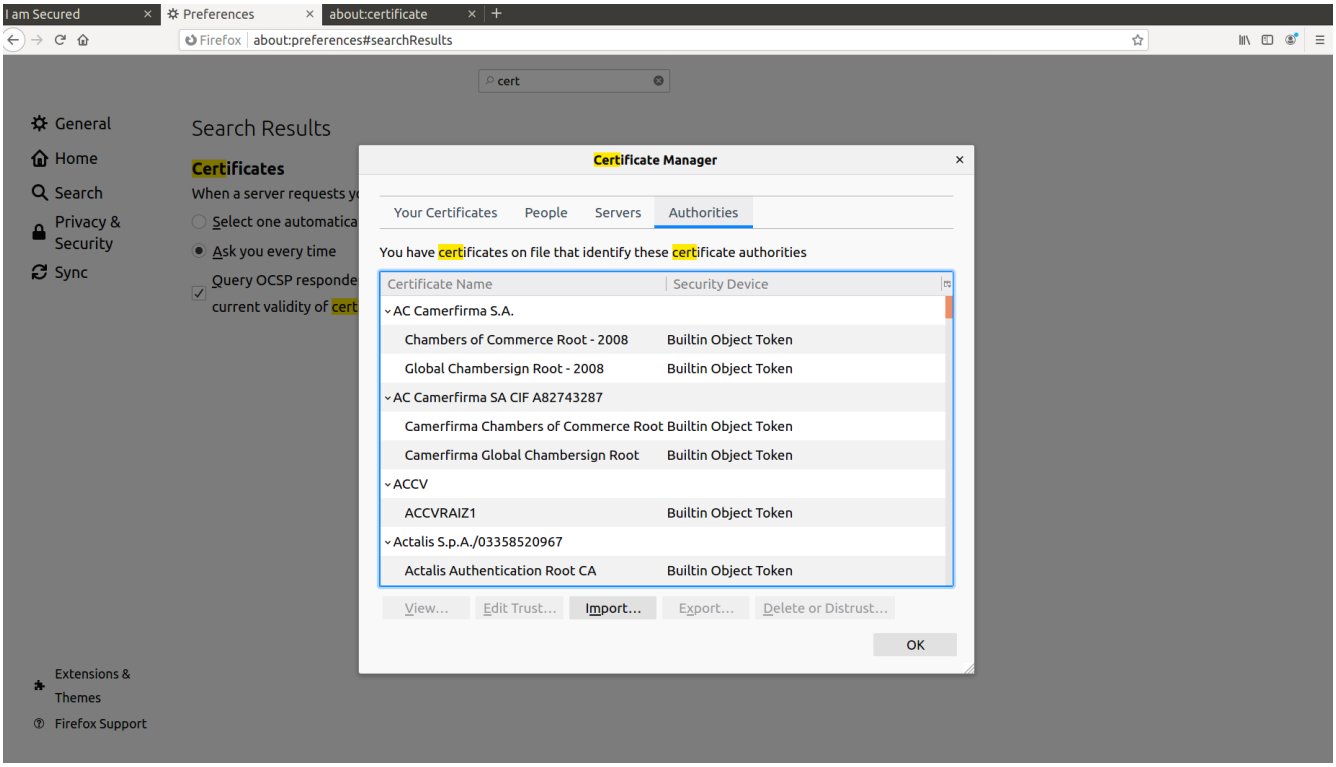
Simplified SSL Handshake Diagram

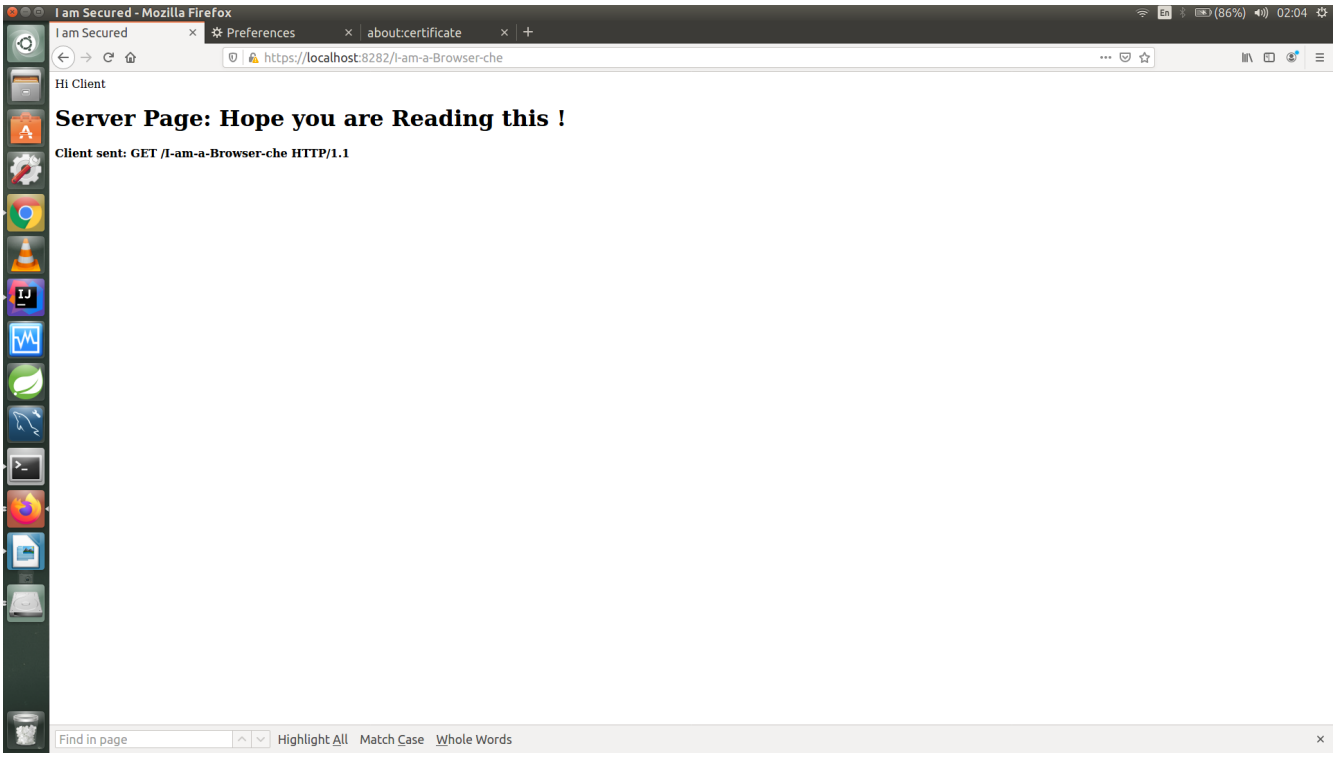
The diagram below shows the simplified, step-by-step process of establishing each new SSL connection between the client (usually a web browser) and the server (usually an SSL web server).



As you can see, the process of establishing each new SSL connection starts with exchanging encryption parameters and then optionally authenticating the servers (using the SSL Handshake Protocol). If the handshake is successful and both sides agree on a common cipher suite and encryption keys, the application data (usually HTTP, but it can be another protocol) can be sent through encrypted tunnel (using the SSL Record Layer).

In reality, the above process is in fact a little bit more complicated. To avoid unnecessary handshakes, some of the encryption parameters are being cached. Alert messages may be sent. Ciphers suites can be changed as well. However, regardless of the SSL specification details, the most common way this process actually works is very similar to the above.


I am Secured
Preferences
about:certificate
Firefox
about:preferences#searchResults
cert
General
Home
Search
Privacy & Security
Sync
Search Results
Certificates
When a server requests y
Select one automatica
Ask you every time
Query OCSP responde
current validity of cert
Certificate Manager
Your Certificates
People
Servers
Authorities
You have certificates on file that identify these certificate authorities
Certificate Name
Security Device
AC Camerfirma S.A.
Chambers of Commerce Root - 2008
Builtin Object Token
Global Chambersign Root - 2008
Builtin Object Token
AC Camerfirma SA CIF A82743287
Camerfirma Chambers of Commerce Root
Builtin Object Token
Camerfirma Global Chambersign Root
Builtin Object Token
ACCV
ACCVRAIZ1
Builtin Object Token
Actalis S.p.A./03358520967
Actalis Authentication Root CA
Builtin Object Token
View...
Edit Trust...
Import...
Export...
Delete or Distrust...
OK
Extensions & Themes
Firefox Support


I am Secured - Mozilla Firefox
I am Secured
Preferences
about:certificate
https://localhost:8282/I-am-a-Browser-che
Hi Client
Server Page: Hope you are Reading this !
Client sent: GET /I-am-a-Browser-che HTTP/1.1
Find in page
Highlight All
Match Case
Whole Words