# Concepts

## TLS vs SSL

TLS is the successor to SSL. It is a protocol that ensures privacy between communicating applications. Unless otherwise stated, in this document consider TLS and SSL as interchangeable.

Certificate (cert)The public half of a public/private key pair with some additional metadata about who issued it etc. It may be freely given to anyone.

## Private Key

A private key can verify that its corresponding certificate/public key was used to encrypt data. It is never given out publicly.

## Certificate Authority (CA)

A company that issues digital certificates. For SSL/TLS certificates, there are a small number of providers (e.g., Symantec/Verisign/Thawte, Comodo, GoDaddy, LetsEncrypt) whose certificates are included by most browsers and Operating Systems. They serve the purpose of a “trusted third party”.

## Certificate Signing Request (CSR)

A file generated with a private key. A CSR can be sent to a CA to request to be signed. The CA uses its private key to digitally sign the CSR and create a signed cert. Browsers can then use the CA’s cert to validate the new cert has been approved by the CA.

## X.509

A specification governing the format and usage of certificates.

## Authentication with SSL

SSL is the standard security technology for establishing an encrypted link between a web server and a browser. Normally when a browser (the client) establishes an SSL connection to a secure web site, only the server certificate is checked. The browser either relies on itself or the operating system providing a list of certs that have been designated as root certificates and to be trusted as CAs.

## One-way SSL authentication (server -> client)

Client and server use 9 handshake messages to establish the encrypted channel prior to message exchanging:

# File Formats for Certs and Keys

## Privacy-Enhanced Mail (PEM)

PEM is just Distinguished Encoding Rules (DER) that has been Base64 encoded. Used for keys and certificates.

## PKCS12

PKCS12 is a password-protected format that can contain multiple certificates and keys.

## Java KeyStore (JKS)

Java version of PKCS12 and also password protected. Entries in a JKS file must have an “alias” that is unique. If an alias is not specified, “mykey” is used by default. It’s like a database for certs and keys.

# Tools

## OpenSSL

An open-source toolkit implementing the SSL (v2/v3) and TLS (v1) protocols, as well as a full-strength general purpose cryptography library.

## Keytool

Manages a Java KeyStore of cryptographic keys, X.509 certificate chains, and trusted certificates. Ships with the JDK.

# Authentication Flows

## One-way SSL authentication (server -> client)

Client and server use 9 handshake messages to establish the encrypted channel prior to message exchanging:

1. Client sends ClientHello message proposing SSL options.
2. Server responds with ServerHello message selecting the SSL options.
3. Server sends Certificate message, which contains the server’s certificate.
4. Server concludes its part of the negotiation with ServerHelloDone message.
5. Client sends session key information (encrypted with server’s public key) in ClientKeyExchange message.
6. Client sends ChangeCipherSpec message to activate the negotiated options for all future messages it will send.
7. Client sends Finished message to let the server check the newly activated options.
8. Server sends ChangeCipherSpec message to activate the negotiated options for all future messages it will send.
9. Server sends Finished message to let the client check the newly activated options.

## Two-way SSL authentication (server <-> client)

Client and server use 12 handshake messages to establish the encrypted channel prior to message exchanging:

1. Client sends ClientHello message proposing SSL options.
2. Server responds with ServerHello message selecting the SSL options.
3. Server sends Certificate message, which contains the server’s certificate.
4. Server requests client’s certificate in CertificateRequest message, so that the connection can be mutually authenticated.
5. Server concludes its part of the negotiation with ServerHelloDone message.
6. Client responds with Certificate message, which contains the client’s certificate.
7. Client sends session key information (encrypted with server’s public key) in ClientKeyExchange message.
8. Client sends a CertificateVerify message to let the server know it owns the sent certificate.
9. Client sends ChangeCipherSpec message to activate the negotiated options for all future messages it will send.
10. Client sends Finished message to let the server check the newly activated options.
11. Server sends ChangeCipherSpec message to activate the negotiated options for all future messages it will send.
12. Server sends Finished message to let the client check the newly activated options.