SSL Session

1. Difference between HTTP and HTTPS

HTTP	HTTPS
HTTP stands for HyperText Transfer Protocol.	HTTPS stands for HyperText Transfer Protocol Secure
Uses port number 80 for communication	Uses port number 443 for communication
It works at application layer	It works at transport layer
In HTTP, encryption is absent	In HTTPS, encryption is present
HTTP is faster than HTTPS	HTTPS is slower than HTTP

2. A SSL session is consists of the following steps.

2.1. **Connect**: When a client sends connection request to the application server. If the application server is on a secure port, the TCP/IP server sends this request to the SSL server labeling things which identify the certificate to ensure a secure connection. Once verified the SSL server sends the request to the application server. Hence, in total 2 connections are formed, client to SSL server and SSL server to application server. The intervention of SSL server is transparent to the application server and the client. To them it looks like they are communicating directly.

2.2. **Handshake**: After successful connection the client initiates handshake protocol to produce the cryptographic parameters for the session. In respond to his the SSL server sends the application server's certificate to the client, asking for clients certificate in return to authenticate it.

2.3. **Data transmission**: Once the handshake is complete, the client sends encrypted data over the network. The SSL server receives it, decrypts it and sends unencrypted data to the application server. The application server responds by sending unencrypted data to the SSL server then encrypts it and sends it to the client.

2.4. **Close**: When a close request is initiated from either side, the SSL certificate sends a close request to the other party and cleans up the connection.

3. Where are SSL certificates stored?

3.1. **Windows**: The certificate store is located in the registry under the HKEY_LOCAL_MACHINE root. This type of certificate store is local to a user account on the computer. This certificate store is located in the registry under the HKEY_CURRENT_USER root.

3.2. **Linux**: The default location to store certificates is /etc/ssl/certs.

4. Private keys and personal certificates are stored in keystores. Public keys and CA certificates are stored in truststores. A truststore is a keystore that by convention contains only trusted keys and certificates.

5. SSL one way and two way handshakes

5.1. SSL one way handshake

5.1.1. In this process the client posts a request to the server to connect.
5.1.2. Server in return shares their public certificate to client for authentication.
5.1.3. Client on receiving shares the certificate to the respected CA's for authentication.
5.1.4. Once the certificate is authenticated by the CA's, the SSL/TLS client shares a random byte string to client and server which will be used for secure connection. The random byte is itself encrypted.
5.1.5. After successful authentication the client and server starts communicating.

5.2. SSL two way handshake

5.2.1. In this process, the client posts a request to the server to connect.
5.2.2. Server shares their public certificate to client for authentication.
5.2.3. Client on receiving shares the certificate to the respected CA's for authentication.
5.2.4. Once the certificate is authenticated by the CA's, the SSL/TLS client shares their public certificate to the server and the server authenticates it by sharing it to the respective CA's.
5.2.5. On successful authentication of both the parties, they connect with each other and data sharing takes place.

6. SSL/TLS versions and ciphers:

6.1. TLS 1.3 Supported Ciphers

6.1.1. AEAD-AES128-GCM-SHA256
6.1.2. AEAD-AES256-GCM-SHA384
6.1.3. AEAD-CHACHA20-POLY1305-SHA256

6.2. TLS 1.2 supported ciphers

6.2.1. CDHE-ECDSA-AES128-GCM-SHA256
6.2.2. ECDHE-ECDSA-CHACHA20-POLY1305
6.2.3. ECDHE-RSA-AES128-GCM-SHA256
6.2.4. ECDHE-RSA-CHACHA20-POLY1305
6.2.5. ECDHE-ECDSA-AES128-SHA256
6.2.6. ECDHE-ECDSA-AES128-SHA
6.2.7. ECDHE-RSA-AES128-SHA256

6.2.8. ECDHE-RSA-AES128-SHA
6.2.9. AES128-GCM-SHA256
6.2.10. AES128-SHA256
6.2.11. ES128-SHA
6.2.12. ECDHE-ECDSA-AES256-GCM-SHA384
6.2.13. ECDHE-ECDSA-AES256-SHA384
6.2.14. ECDHE-RSA-AES256-GCM-SHA384
6.2.15. ECDHE-RSA-AES256-SHA384
6.2.16. ECDHE-RSA-AES256-SHA
6.2.17. AES256-GCM-SHA384
6.2.18. AES256-SHA256
6.2.19. AES256-SHA

7. Use SSL to sniff SSL handshake