**Root SSL certificate**A **Root SSL certificate** is a certificate issued by a **trusted**[**certificate authority (CA)**](https://support.dnsimple.com/articles/what-is-certificate-authority)

**Steps to create HTTPS :**

We use HTTS to secure communication between server and client.

As HTTP will transfer data as plain text.

HTTPS is protocol to encrypt request and response

Is an application specific implementation that is a combination of the Hypertext Transfer Protocol (**HTTP**) with the **SSL**/TLS

**What is SSL ; secure socket layer**

It is standard security technology for establishing encrypted link between a web server and a Client

**Difference between SSL.2 and SSL.3**

SSL 3.0 uses HMAC, which is more powerful than MAC.

HMAC is 128-bit of encryption but MAC is 40 bit

MAC supports only RSA key exchange and not none-RSA

SSL 3 supports also chain certificate

Steps to create HHTPS in server

1. Create certificate CA\_ certificate using keytool to save it in out keystore
2. We caret keystore.p12  file

Configure Spring boot

server.port: 8443

server.ssl.key-store: keystore.p12

server.ssl.key-store-password: mypassword

server.ssl.keyStoreType: PKCS12

server.ssl.keyAlias: tomcat

1. **First we create CA certificate**

CA-Certificate is an entity that issues X509 ( standard entity trusted third party

Open

ssl req -config c:\openssl\bin\openssl.cnf -new -x509 -keyout ca-key.pem.txt -out ca-certificate.pem.txt -days 365

**2. Creating the Keystore ( -genkey )**

The keys that server needs to use for secure communication are stored in a file called Keystore

keytool –keystore clientkeystore –genkey –alias client

Clinetkeystore file will be gerenarted

KeyTool will ask to enter password and company info

1. **Create CSR certificate request ( -certreq)**This file will be used by certificate to generate SLL that is presented to other parties during the handshake

keytool –keystore clientkeystore –certreq –alias client –keyalg rsa

–file client.csr

1. **Create CA-Key for CSR ( - CAkey)**

Openssl x509 -req -CA ca-certificate.pem.txt -CAkey ca-key.pem.txt

-in client.csr -out client.cer -days 365 -CAcreateserial

1. **Import CA certificate in to keystore file**

keytool -import -keystore clientkeystore -file ca-certificate.pem.txt

-alias theCARoot

keytool -genkey -keystore c:\ arashkeystore -alias arashtomcat -keyalg RSA -storetype PKCS12 -keyalg RSA -keysize 2048 -keystore c:\test\keystore.p12 -validity 3650

1. First create certificate or obtain certificate file from trusted certificate authority
2. If we create ourselves we use keytool
3. We caret keystore which is a file that our certificate will be imported in there.
4. For inboud connection we just add that to folder and configuare it in server connector
5. For outboutd server(boot that needs to call another boot) it will be acted as client java so then we need to import the certificate in to the JVM/JRE /certificate location by the following :

keytool -import -trustcacerts -keystore cacerts -storepass changeit -noprompt -alias mycert -file mycert.cer

1. Get an SSL certificate
   * Generate a self-signed SSL certificate
   * Use an existing SSL certificate
2. Enable HTTPS in Spring Boot
3. Redirect HTTP requests to HTTPS
4. Distribute the SSL certificate to clients.
5. <https://www.baeldung.com/spring-boot-https-self-signed-certificate>
6. PKCS12: [**Public Key Cryptographic Standards**](https://en.wikipedia.org/wiki/PKCS_12) is a password protected format that can contain multiple certificates and keys; it’s an industry-wide used format
7. JKS: [**Java KeyStore**](https://en.wikipedia.org/wiki/Keystore) is similar to PKCS12; it’s a proprietary format and is limited to the Java environment.

<https://www.quora.com/Whats-the-difference-between-a-jks-and-a-cer-certificate-file-Is-it-possible-to-convert-from-one-to-the-other>