Configuring a Certificate Store

A certificate store or keystore is a database of keys. Private keys in a keystore have a certificate chain associated with them, which authenticates the corresponding public key. A keystore also contains certificates from trusted entities.

The keystore must contain a key pair with a certificate signed by a trusted Certification Authority (CA).

How to create a keystore?

We will be using the JDK 'keytool', which is a key and certificate management utility. It allows users to administer their own public/private key pairs and associated certificates for use in self-authentication (user authenticates himself/herself to the service).

To generate the keystore, open a command line and enter the following to generate a key pair and certificate directly into it:

```
keytool  -keystore  [keystore file name]  -alias [domain]  -genkey  -keyalg  RSA
```

For example:

```
keytool  -keystore  myKeystore.jks  -alias aspire  -genkey  -keyalg  RSA
```

This command will prompt for information about the certificate and for passwords to protect both the keystore and the keys within it. The only mandatory response is to provide the fully qualified host name of the server at the "first and last name" prompt.

Certificate information, for example:

```
Enter keystore password: myKeystorePassword
Re-enter new password: myKeystorePassword

What is your first and last name?
  [Unknown]:  my-pc.search.local

What is the name of your organizational unit?
  [Unknown]:

What is the name of your organization?
  [Unknown]:

What is the name of your City or Locality?
  [Unknown]:

What is the name of your State or Province?
  [Unknown]:

What is the two-letter country code for this unit?
  [Unknown]:

Is CN=my-pc.search.local, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown correct?
  [no]:  yes

Enter key password for <aspire>
        (RETURN if same as keystore password): myKeyPassword
Re-enter new password: myKeyPassword
```

A keystore file is generated with the content encrypted.

Get the Certificate

For SSL to work, you need to install the certificate in the machine. To get the certificate there are 2 options: extracting the certificate from the keystore (the easiest and the one we recommend) or generate a self-signed certificate and import it into the keystore

The certificate needs to be installed in the server machine, otherwise the SSL will not work. the installation of the certificate varies among operating systems

This certificate is enough to run SSL. However, this certificate we generated will not be trusted by the browser unless we request a well known Certificate Authority (CA) to sign our key/certificate. Among them are: AddTrust, Entrust, GeoTrust, RSA Data Security, Thawte, ,VISA, ValiCert, Verisign and beTRUSTed.

Extracting the Certificate from the Keystore

Using the command below, you can list the certificates inside the keystore, identify the one to be extracted via the Alias

```
keytool -list -v -keystore [keystore file name]
```

Example

```
keytool -list -v -keystore keystore.jks
Enter keystore password:


Keystore type: PKCS12
Keystore provider: SUN

Your keystore contains 1 entry

Alias name: aspire
Creation date: Mar 19, 2020
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=loacalhost, OU=Content Analytics, O=Accenture, L=Heredia, ST=Heredia, C=CR
Issuer: CN=loacalhost, OU=Content Analytics, O=Accenture, L=Heredia, ST=Heredia, C=CR
Serial number: 6d395d90
Valid from: Thu Mar 19 10:44:56 CST 2020 until: Wed Jun 17 10:44:56 CST 2020
Certificate fingerprints:
         SHA1: E8:E9:3D:BB:C6:BD:E8:B2:11:96:56:07:53:39:DD:42:44:E4:3F:F5
         SHA256: A3:3E:7B:D8:76:18:A2:D0:19:13:94:AB:4F:CB:6A:98:59:39:53:C8:E1:21:14:FF:87:0C:86:05:BE:DF:16:D2
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 3D F7 1B 37 94 DA F9 34   03 84 FE E2 33 3E F3 7B  =..7...4....3>..
0010: D9 C1 62 74                                        ..bt
]
]



*******************************************
*******************************************
```

Once you have identified the certificate's alias; to extract it you can use the following command:

```
keytool -export -alias [domain]  -file [filename for certificate] -keystore [keystore file name]
```

Example

```
keytool -export -alias aspire -file aspire.crt -keystore keystore.jks
Enter keystore password:
Certificate stored in file <aspire.crt>
```

Generating Self-Signed Certificates

This solution requires OpenSSL to be installed in the machine

There is a utility script available for Windows (**generate-ssl-certs.bat**) and Linux (**generate-ssl-certs.sh**), which generates the self-signed CA, server, and client certificate. Also regardless of the file picked you need to also download the openssl config (**openssl.cnf**) in the same folder as the script file

The command receives the client certificate name, the certificate password and a destination folder (if destination is not added, then current folder will be the destination).

```
generate-ssl-certs.bat --generate [domain] [password] [folder_path]
```

generate-ssl-certs.bat must be run from the *bin* directory. If you run from another directory, it appears to work but will not write all the certificate files (in fact it writes the key files only)

```
generate-ssl-certs.bat --generate aspire-cert 123456
Location: C:\Users\test\Desktop


Generating CA
Generating Server Certificate
Generating Client Certificate
```

Importing the Certificate into the Keystore

You can generate a keystore for the client certificate and a truststore for the CA certificate.

Make sure the keystore file is not placed in "/config/certs" folder of your distribution. Only X509 certificates should be placed in this location.

Keystore

```
keytool -v -importkeystore -srckeystore client/client.p12 -srcstoretype PKCS12 -destkeystore myKeystore.jks -
deststoretype JKS
```

Truststore

```
keytool -import -file ca/ca.crt -keystore myKeystore.jks
```