Using HTTPS (Aspire 2)

Enabling HTTPS in Apache Felix

For more information on how to enable secure HTTP for Aspire, see apache-felix-http-service here

Creating a keystore

A 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 –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.

To view the certificate explicitly, 'keytool' has a command to export the certificate from the keystore. We can do so with the following command:

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

For example:

```
C:\Users\user.SEARCH> keytool -export -alias aspire -file aspire.crt -keystore myKeystore
Enter keystore password: myKeystorePassword
Certificate stored in file <aspire.crt>
```

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.

How to request a CA to sign our certificate?

The following command will generate the .CSR (Certificate Signing Request) for the key/certificate already in the keystore:

```
 keytool -certreq -alias [domain] -keystore [keystore file name] -file [.csr file name]
```

For example:

```
keytool -certreq -alias aspire -keystore myKeystore -file aspire.csr
```

It will generate a CSR file which should contain a certificate request:

```
-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBuTCCASICAQAweTEQMA4GA1UEBhMHVW5rbm93bjEQMA4GA1UECBMHVW5rbm93bjEQMA4GA1UE
BxMHVW5rbm93bjEQMA4GA1UEChMHVW5rbm93bjEQMA4GA1UECxMHVW5rbm93bjEdMBsGA1UEAxMU
c29maS1wYy5zZWFyY2gubG9jYWwwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAInYjLYyB4Wh
LRJKxk4JxpYiDSWuJrmRDjxdXwHlXK3/X7JDqOU7JVrMIQD9oqxGMTjWssZ1fkG43+Efvg5zNwjH
PJRKuYQIonaAmJJINyI7nr4eUT7u4Sla9SxEoT8LudYf6LdTgLFL614xmjZ0Yziz+0XS/FiqTtlO
Tj58oN/PAgMBAAGgADANBgkqhkiG9w0BAQUFAAOBgQA9paeqnnYn+eGsSdk5Jzkz5iYeYB1a4/JK
72QE36zXRGQ78iVzyTMfHWmajnVW1h3E7EFtigEDq0WAEsCokBcmjq6b06bug6AERER2mb7AXxKP
fbPZGB5Id3GsOICT1RF5QG/xAIHXjQhD3tedZ15cDnPj6qGecLrZFgY90aijSg==
-----END NEW CERTIFICATE REQUEST-----
```

Then you can send the certificate request to a proper Certificate Authority (see example list above). The exact procedure depends on the Certificate Authority you choose.

Configuring Felix Properties

The config\felix.properties file should be edited to enable HTTPS.

If we leave this property in the configuration, it will still work with HTTPS enabled. So, http://localhost:50505/aspire/ will access the admin interface with HTTP (see notes below).

```
 org.osgi.service.http.port=50505
```

To enable HTTPS, we must set the following values:

```
 org.apache.felix.https.enable=true
 org.osgi.service.http.port.secure=50443
```

The default secure port is 443, and if you use that port then you do not need to put the port number in the URL. However it seems like further configuration is required to avoid the browser "SSL connection error".

Optionally, we could disable unsecured HTTP traffic with:

```
 org.apache.felix.http.enable=false
```

And specify the keystore file and passwords to access it:

```
 org.apache.felix.https.keystore=config/myKeystore  <<<< NOTE UNIX STYLE PATH SEPARATOR IS REQUIRED EVEN ON
WINDOWS, THIS PATH IS RELATIVE TO ASPIRE_HOME BY DEFAULT
 org.apache.felix.https.keystore.password=myKeystorePassword
 org.apache.felix.https.keystore.key.password=myKeyPassword
```

Finally, Aspire can be started and the admin page can now be accessed using HTTPS: https://my-pc.search.local:50443/aspire

Access to other components should be through HTTPS as well, such as the HTTP Feeder:

```
https://my-pc.search.local:50443/submitFiles?test1=hello&test2=world

<doc>
  <feederLabel>HttpFeeder</feederLabel>
  <test1 source="HttpFeeder">hello</test1>
  <test2 source="HttpFeeder">world</test2>
</doc>
```

Notes:

- In some instances (possibly all) if you don't turn *http* off when you turn *https* on, requests to the *https* port will hang. It's probably best to turn *http* off when you turn *https* on
- The URL must match the server host entered into the certificate. In other words, it should match the "Issuer" name in the certificate. If accessed through another such as https://localhost:50443/aspire it will still work but you will get the error message: "Server's certificate does not match the URL".
- If the certificate has not been signed by a valid CA, the 'lock' icon will appear in red and will show a "Server's certificate is not trusted" message.

blocked URL