LAB- Enabling Https

You will be working on 13-usinghttps project

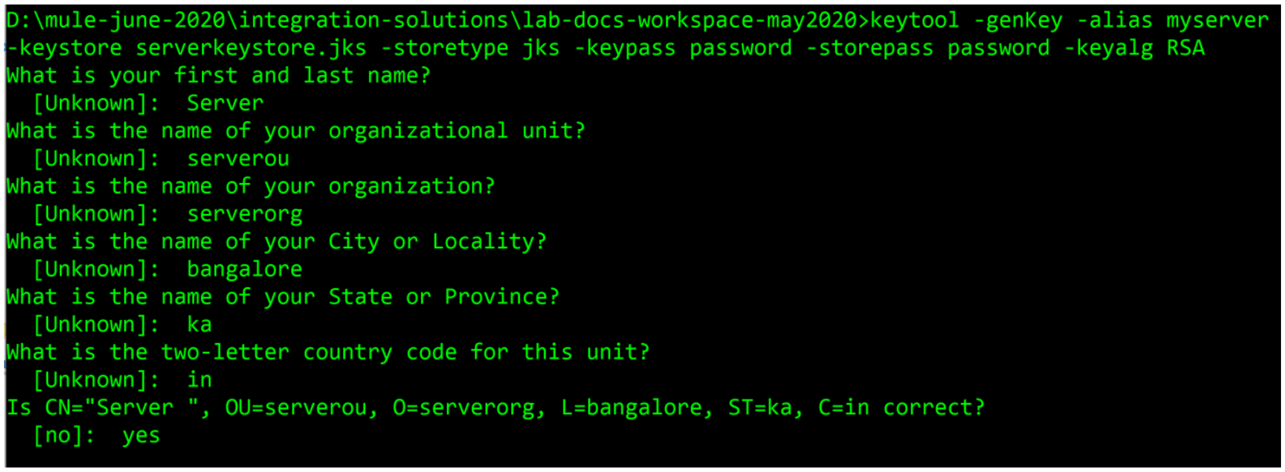
We want to enable https on the serverside.

So, we need to generate server keystore first.

Use the below command to generate server keystore file

keytool -genKey -alias myserver   -keystore serverkeystore.jks -storetype jks -keypass password -storepass password -keyalg RSA

Give the values as shown below:

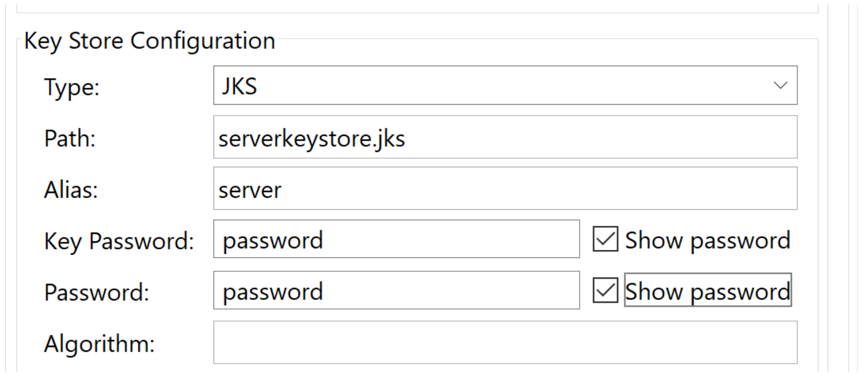


```
D:\mule-june-2020\integration-solutions\lab-docs-workspace-may2020>keytool -genKey -alias myserver
-keystore serverkeystore.jks -storetype jks -keypass password -storepass password -keyalg RSA
What is your first and last name?
  [Unknown]:  Server
What is the name of your organizational unit?
  [Unknown]:  serverou
What is the name of your organization?
  [Unknown]:  serverorg
What is the name of your City or Locality?
  [Unknown]:  bangalore
What is the name of your State or Province?
  [Unknown]:  ka
What is the two-letter country code for this unit?
  [Unknown]:  in
Is CN="Server ", OU=serverou, O=serverorg, L=bangalore, ST=ka, C=in correct?
  [no]:  yes
```

Copy the generated serverkeystore.jks into src/main/resources

Open httpsserver.xml and edit the global element “Http Listener”

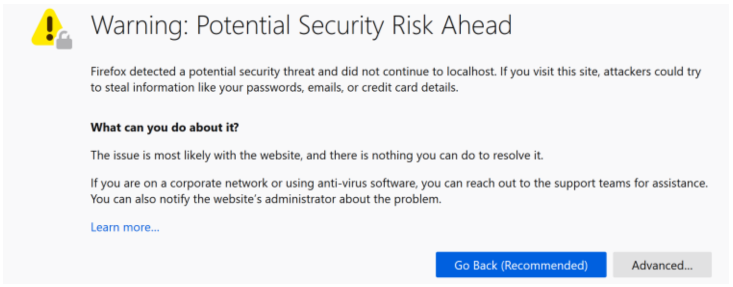
Change the protocol to HTTPS
Click on TLS tab, select edit inline and configure “Key Store Configuration” as shown below:



Key Store Configuration

Type:	JKS
Path:	serverkeystore.jks
Alias:	server
Key Password:	password ✓ Show password
Password:	password ✓ Show password
Algorithm:	

Deploy the application and give a request to https://localhost:8081/test
If you are using fire fox   browser , it should show a warning as shown below:



Warning: Potential Security Risk Ahead

Firefox detected a potential security threat and did not continue to localhost. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.

What can you do about it?

The issue is most likely with the website, and there is nothing you can do to resolve it.

If you are on a corporate network or using anti-virus software, you can reach out to the support teams for assistance. You can also notify the website's administrator about the problem.

Click on advanced and click on View certificate. You should see the server certificate which we generated.

Now click on “Accept the Risk and continue”. You should be able to see the response “Hello!!”

Now we want to consume this https flow using “Http Request component”.

Open httpsclient.xml and observe that there is a flow which is making http request now to https flow.

In this flow, there is a http listener at http://localhost:8082/clienttest

Just give a request to http://localhost:8082/clienttest and observe that u will get an error because client is trying to make http call to https flow.

So, we want modify client to make https call to server.

On the client side, we need a client trust store which has server certificate imported.

So, first export the server certificate using the below command:

```
keytool -exportcert -alias myserver -keystore serverkeystore.jks -file servercert.cer
-storepass password
```

Now import this server certificate into client trust store using below command

```
keytool -importcert   -keystore clienttruststore.jks -file servercert.cer -alias myserver
-storepass password
```

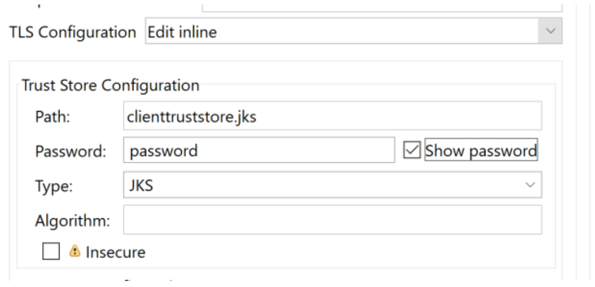
Now copy clienttruststore.jks into src/main/resources

Open HttpsClient.xml and edit the global element   “Http Request Configuration”

Select Protocol as “Https”

Go down and Select “Edit Inline” for TLS Configuration

Configure the client truststore as shown below:



TLS Configuration: Edit inline

Trust Store Configuration

Path: clienttruststore.jks

Password: password ☑ Show password

Type: JKS

Algorithm:

☐ Insecure

Now redeploy the application and give a request to client at http://localhost:8082/clienttest

You should get the response.

So, Till now, we implemented on ONE way SSL. IN one way SSL, client is checking of the certificate sent by server is in client’s truststore. But server is not requesting for client certificate and not validating it.

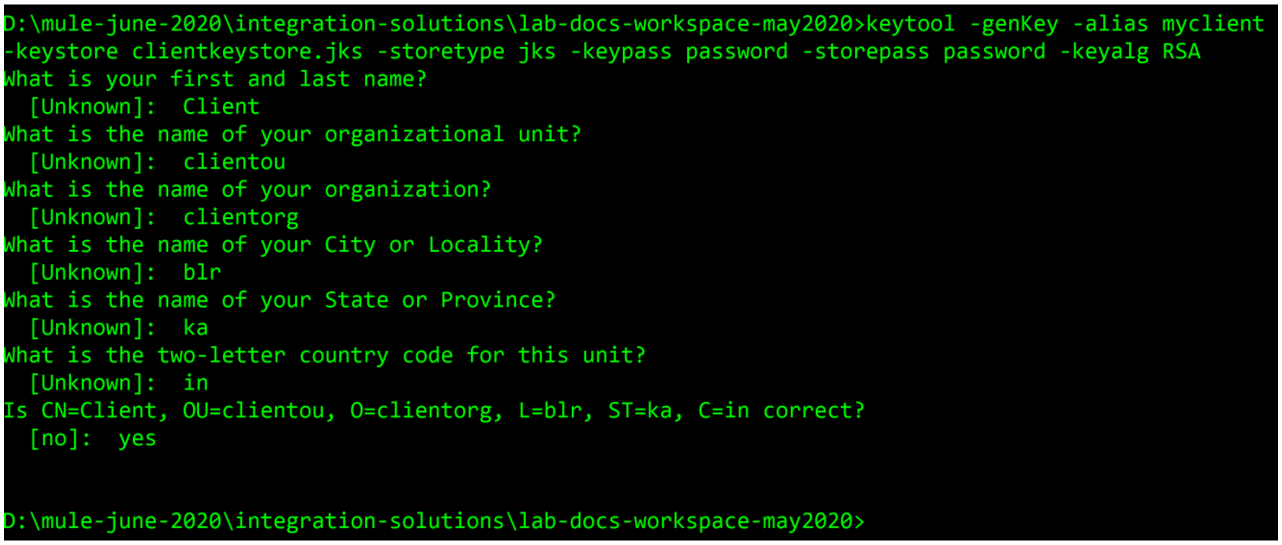
Now we want 2 way SSL where server will also need to validate Client certificate.

So, we need to configure a trust store on server and import client certificate into server truststore.

Also we need to configure client keystore

Firstly, Create a client keystore using below command as shown below:

keytool -genKey -alias myclient -keystore clientkeystore.jks -storetype jks -keypass password -storepass password -keyalg RSA



```
D:\mule-june-2020\integration-solutions\lab-docs-workspace-may2020>keytool -genKey -alias myclient
-keystore clientkeystore.jks -storetype jks -keypass password -storepass password -keyalg RSA
What is your first and last name?
  [Unknown]:  Client
What is the name of your organizational unit?
  [Unknown]:  clientou
What is the name of your organization?
  [Unknown]:  clientorg
What is the name of your City or Locality?
  [Unknown]:  blr
What is the name of your State or Province?
  [Unknown]:  ka
What is the two-letter country code for this unit?
  [Unknown]:  in
Is CN=Client, OU=clientou, O=clientorg, L=blr, ST=ka, C=in correct?
  [no]:  yes


D:\mule-june-2020\integration-solutions\lab-docs-workspace-may2020>
```

Export client certificate using below command:

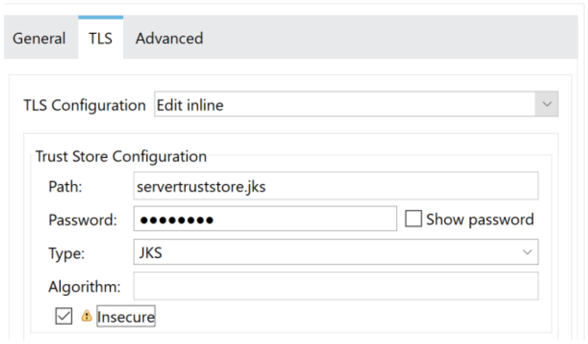
keytool -exportcert -alias myclient -keystore clientkeystore.jks -file clientcert.cer -storepass password

Import client certificate into server trust store using the below command:

keytool -importcert -keystore servertruststore.jks -file clientcert.cer -alias myclient -storepass password

Copy clientkeystore.jks and servertruststore.jks into src/main/resources

Open httpsclient.xml . Edit http listener config and configure it trust store as shown below:



TLS Configuration: Edit inline

Trust Store Configuration
Path: servertruststore.jks
Password: ••••••••
☐ Show password
Type: JKS
Algorithm:
☑ Insecure

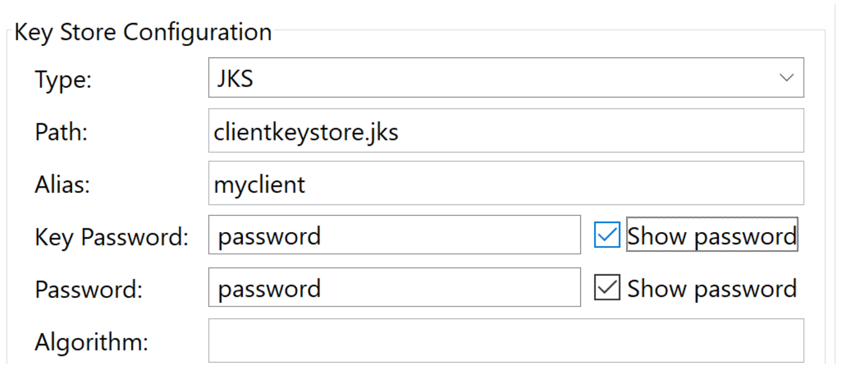
Make Sure that you select “Insecure” check box.

We need to configure this insecure=true becase we don’t want to validate the certificate as we are using self signed certificate.

In production when we have a CA Signed certificate, we can make insecure=false (Don’t select the check box “Insecure”)

Now open httpsclient.xml and edit “Http request configuration”.

Configure client keystore as shown below:



Key Store Configuration

Type:	JKS	
Path:	clientkeystore.jks	
Alias:	myclient	
Key Password:	password	✓ Show password
Password:	password	✓ Show password
Algorithm:		

That’s all . We have configured 2 way SSL.

Deploy the application and make a request to client application at http://localhost:8082/clienttest

You should get proper response.