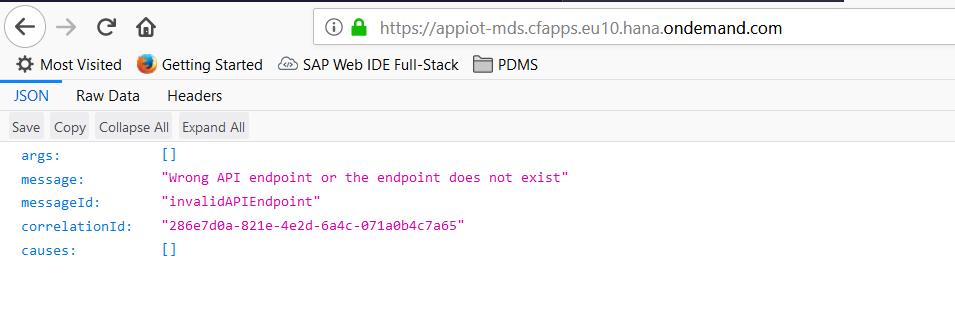
How to apply a Javascript Extension Policy to parameterize an API Proxy from API Management

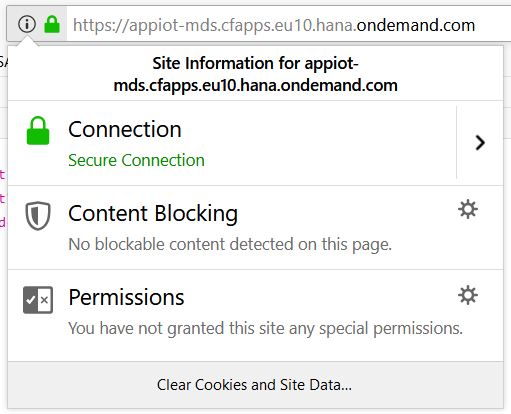
To access the IoT AE APIs from SAP Cloud Platform API Management it requires certificates containing private key and root certificates in PEM format.

The following steps show how to download the Client Certificate using browser and convert certificate to PEM format using OpenSSL.

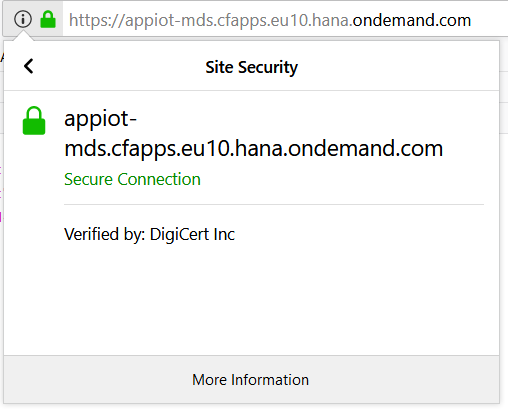
1. To download the certificate open the url <https://appiot-mds.cfapps.eu10.hana.ondemand.com> in browser and you will get the following screen.



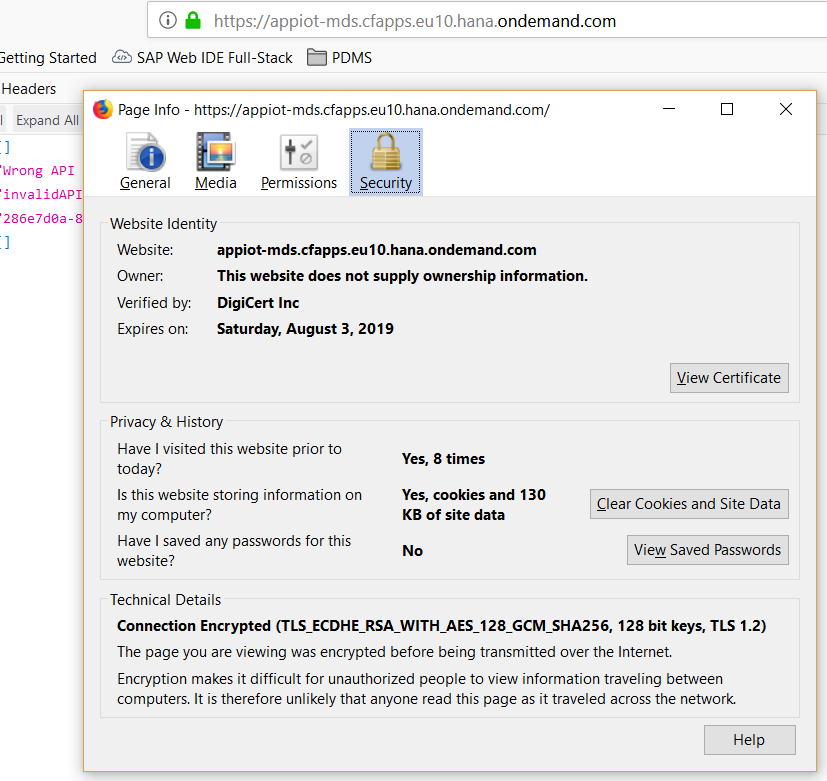
1. Open the site information popover by clicking the green lock icon and navigate inside Connection



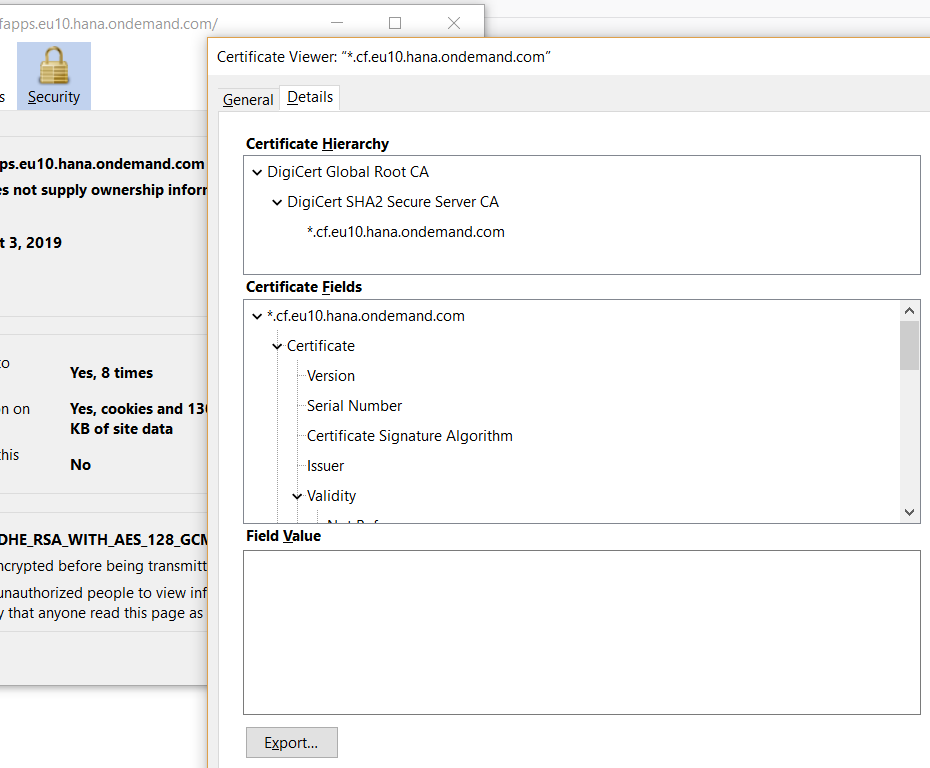
1. Once inside Site Security click on “More Information”



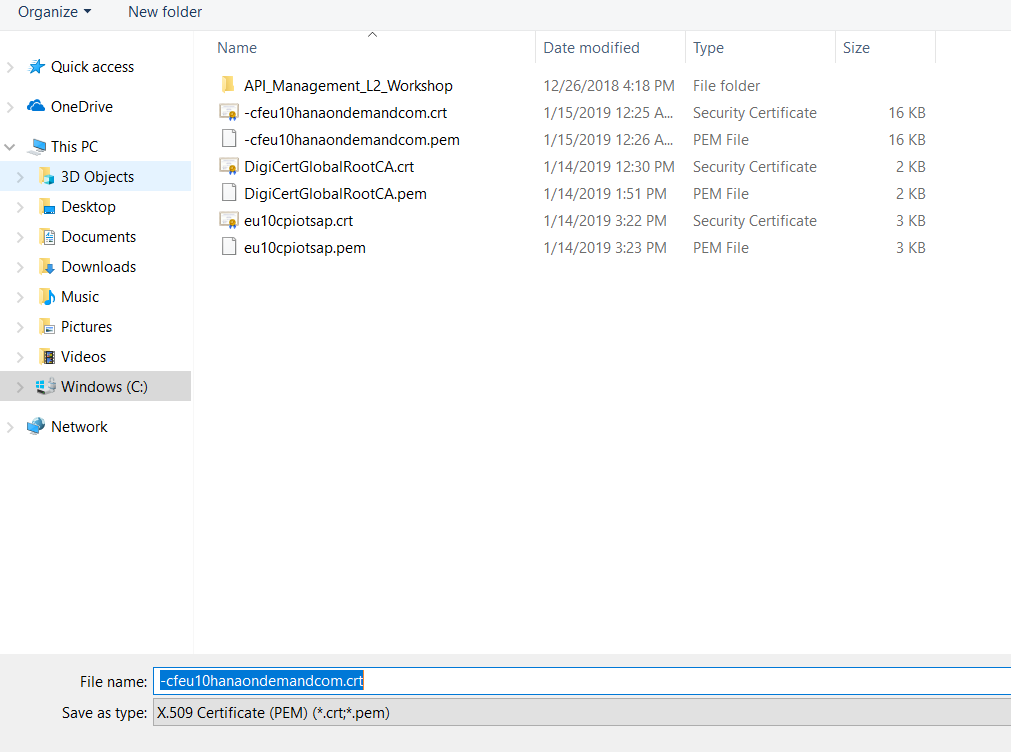
1. Click on “More Information” will open the Page Info popup as shown below.



1. Now, click on View Certificate and go to the Details tab as shown below.



1. Click on the Export button to export the Client Certificate and save it in your local folder.

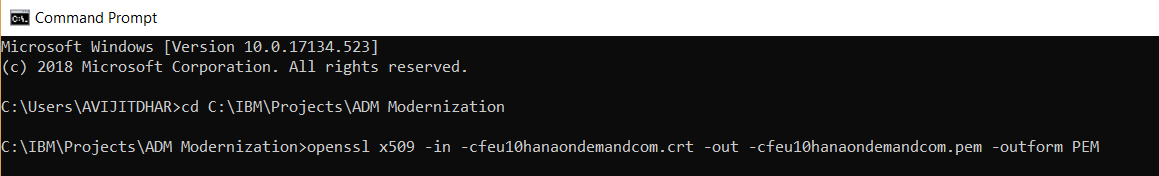


1. **Install** **OpenSSL**

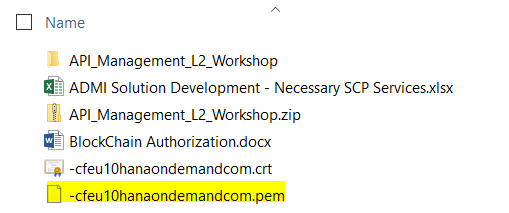
* Install Open SSL from - <https://sourceforge.net/projects/openssl/?source=typ_redirect> and add the path to Open SSL.exe in system variable PATH
* To check the installation Open SSL open Command Prompt and type **openssl** and the command must be recognized by the system.

1. Now using OpenSSL we will extract certificate with private key in .**pem** format as shown below.

*openssl x509 -in -cfeu10hanaondemandcom.crt -out -cfeu10hanaondemandcom.pem -outform PEM*

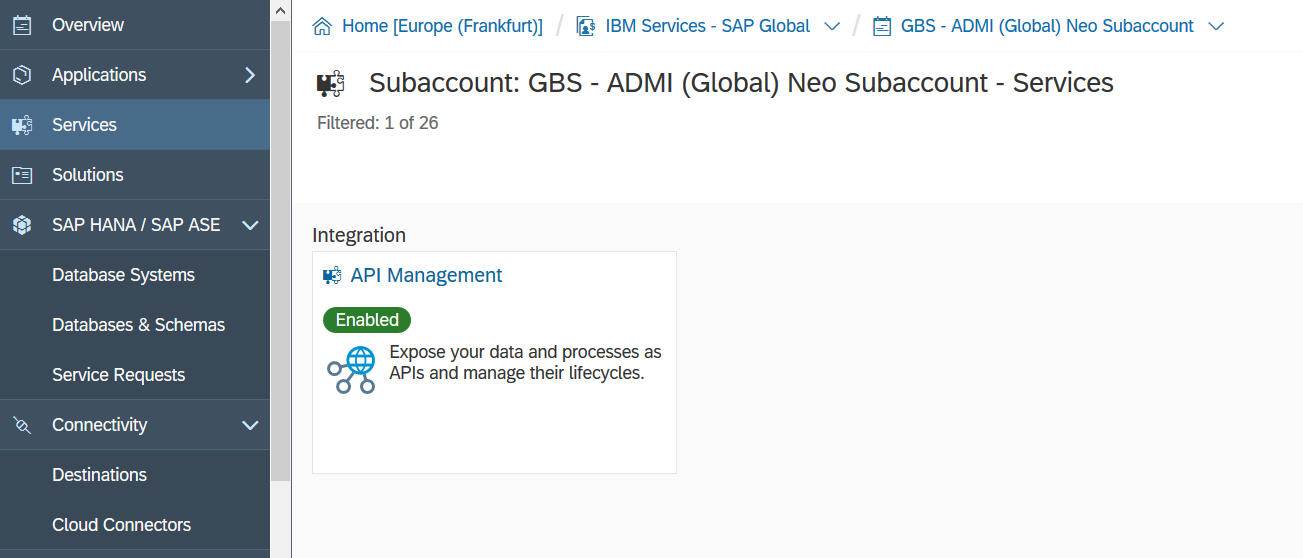


1. Once executed, this command will generate the .pem certificate in the designated folder as shown below.

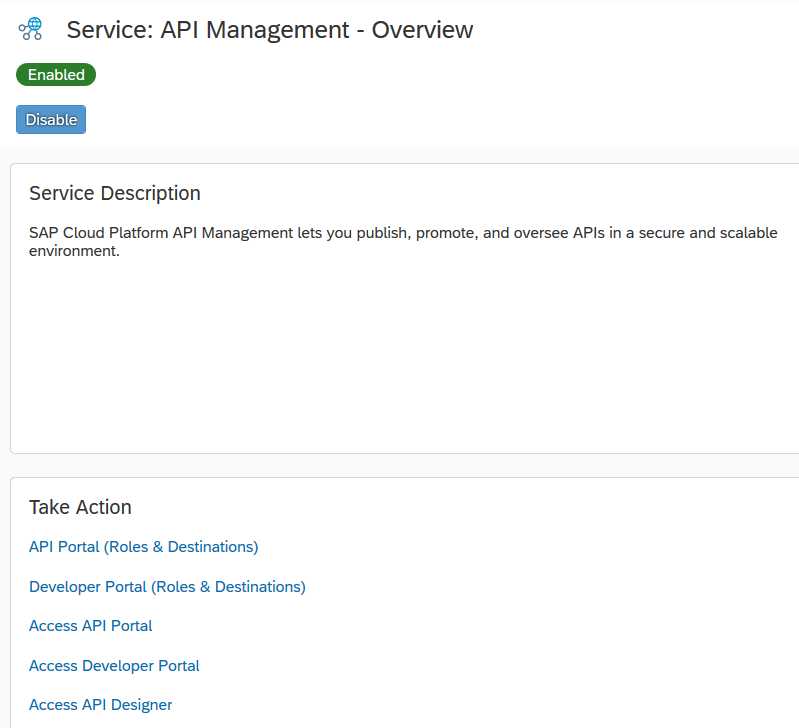


So far, we have followed the steps to generate a self-signed X509 certificate with private key in PEM which can then be used for Client Certificate authentication.

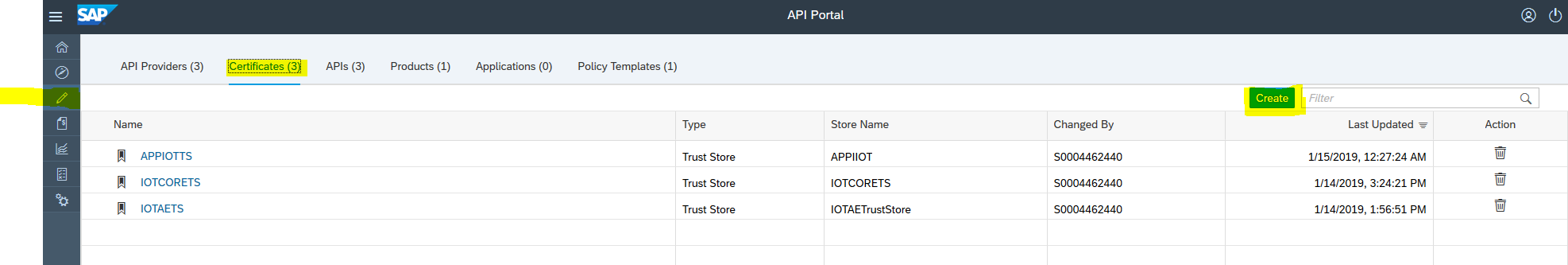
1. Now we will import this certificate into SAP Cloud Platform API Management – API Portal and use this from creating the required proxies.
2. Logon to your SAP Cloud Platform account and navigate to the Services tab, search for API Management service tile and click to open API Management service.



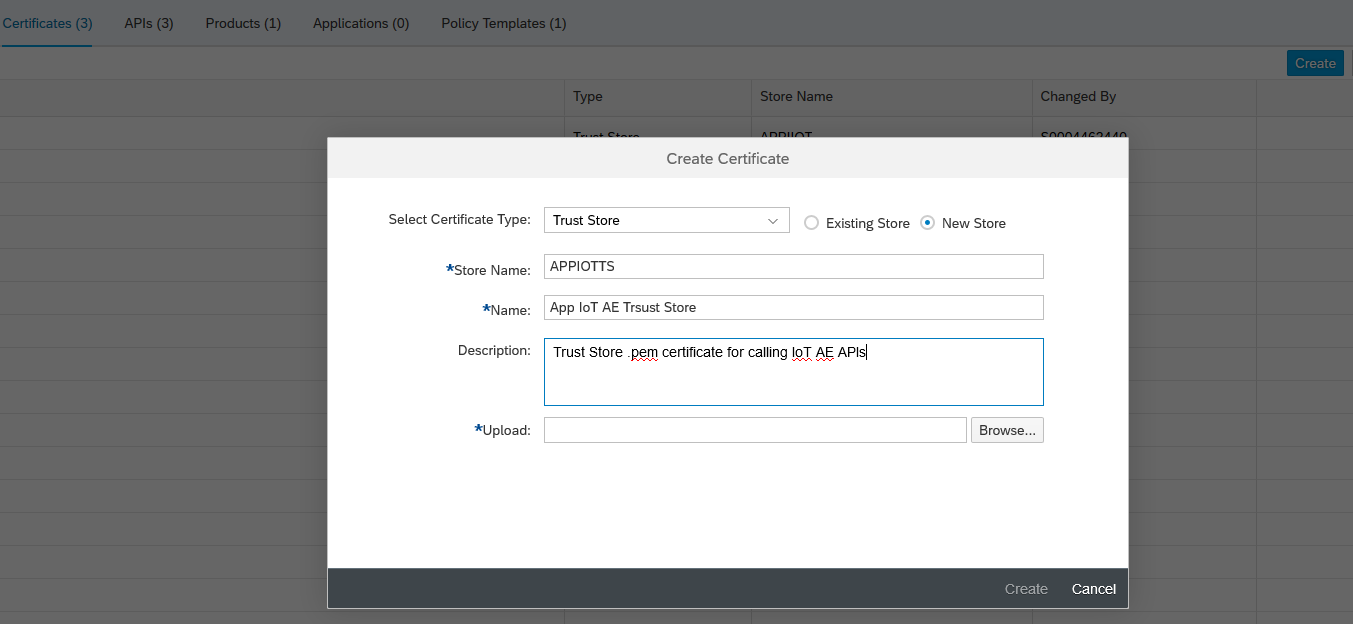
1. Click on the link **Access API Portal**to open API Portal.



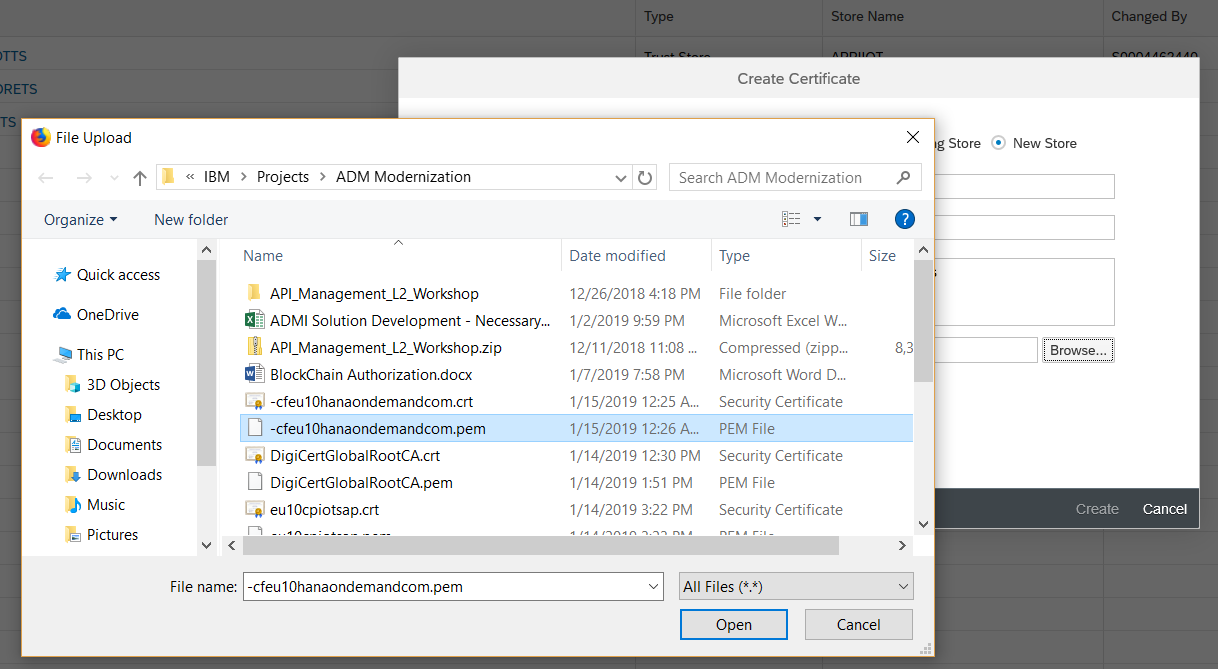
1. Select the option **Certificate** and click on the option **Create.**



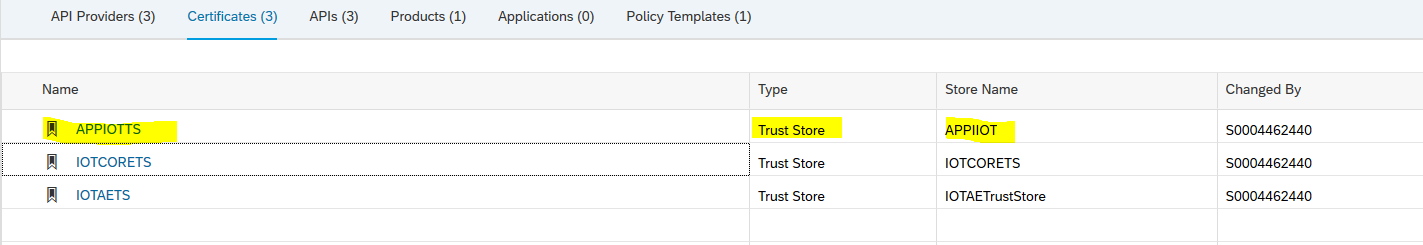
1. Click on Create will open the follow window. In the UI, select the certificate type as **Key Store**, select the option **New Store**.



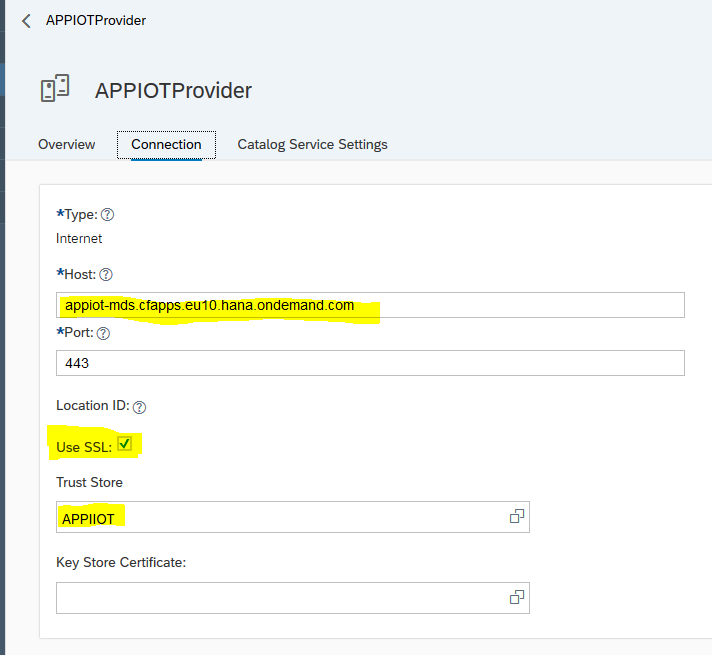
1. Enter the relevant details as shown above in the given fields and click browser to upload the .pem certificate which we generated at step 8 & step 9.

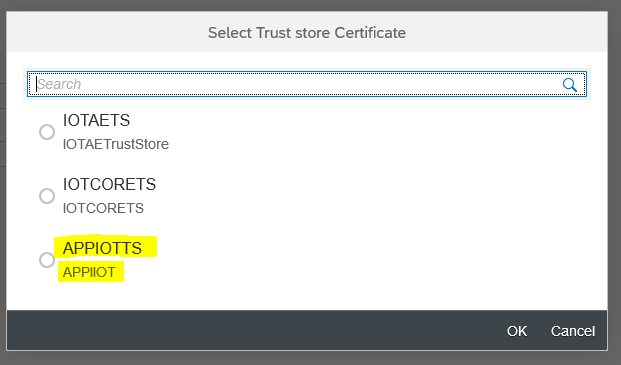


1. Once done click on **Create** button to save the Certificate inside API Portal.



1. Next, we will create a API Provider which will basically use the uploaded certificate.
2. Go to **API Provider** tab and then click on **Create** button.
3. In the create API Provider screen fill system details (important is to check **use ssl** flag) and then Go to tab **AUTHENTICATION** and Select previously uploaded key store in parameter Key Store Certificate.

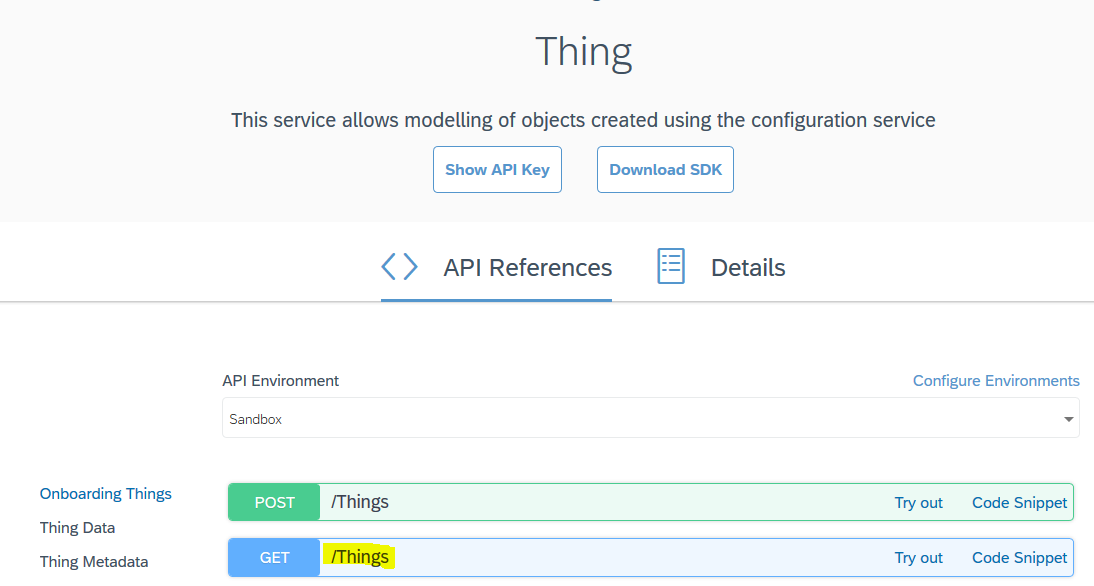




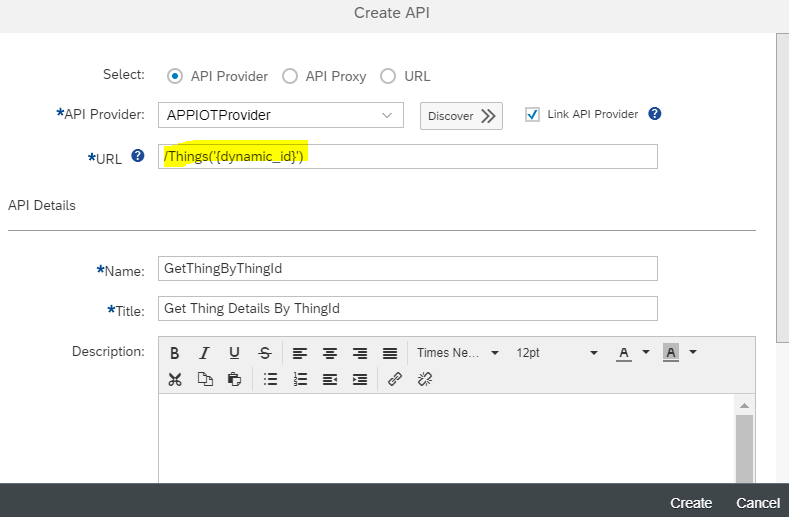
1. Once you save the API provider you can click the “Test Connection” to check the connectivity of the **API Provider**.

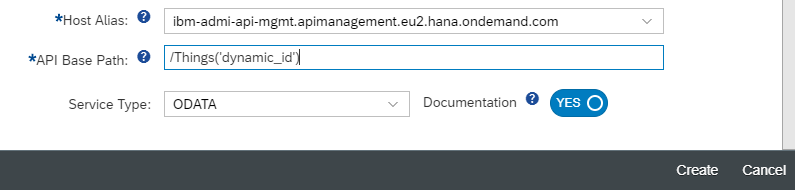


1. We are the final step and will now use this **API Provider** to create a new API. Select the APIs tab and then click **Create** button to create an API Proxy which requires the Certificate Authentication and from the **API Provider** drop down select the provider created and enter the details for your API Proxy and then click **Create**.
2. In this exercise we are trying to create a proxy out of IoT AE Rest API called Things. This API return can return thing details upon passing a specific Thing Id as a parameter.



1. To pass the parameter dynamically we need to create the proxy as shown below where the url should look like this /Things(‘{**dynamic\_id**}’). In place of **dynamic\_id** you can use any variable name.

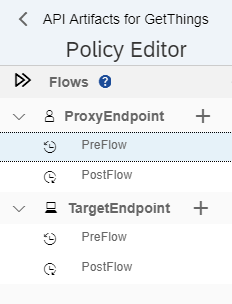




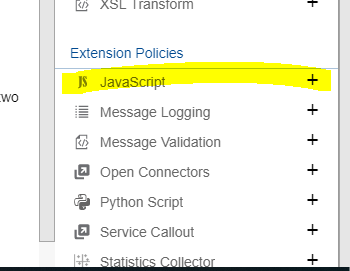
1. Once the proxy is created click on Policies link at the top right corner of the screen.



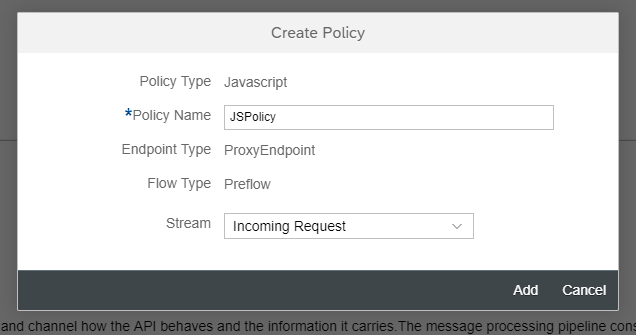
1. We will next create a **ProxyEndpoint** **PreFlow** **Javascript** extension policy to add the necessary logic.



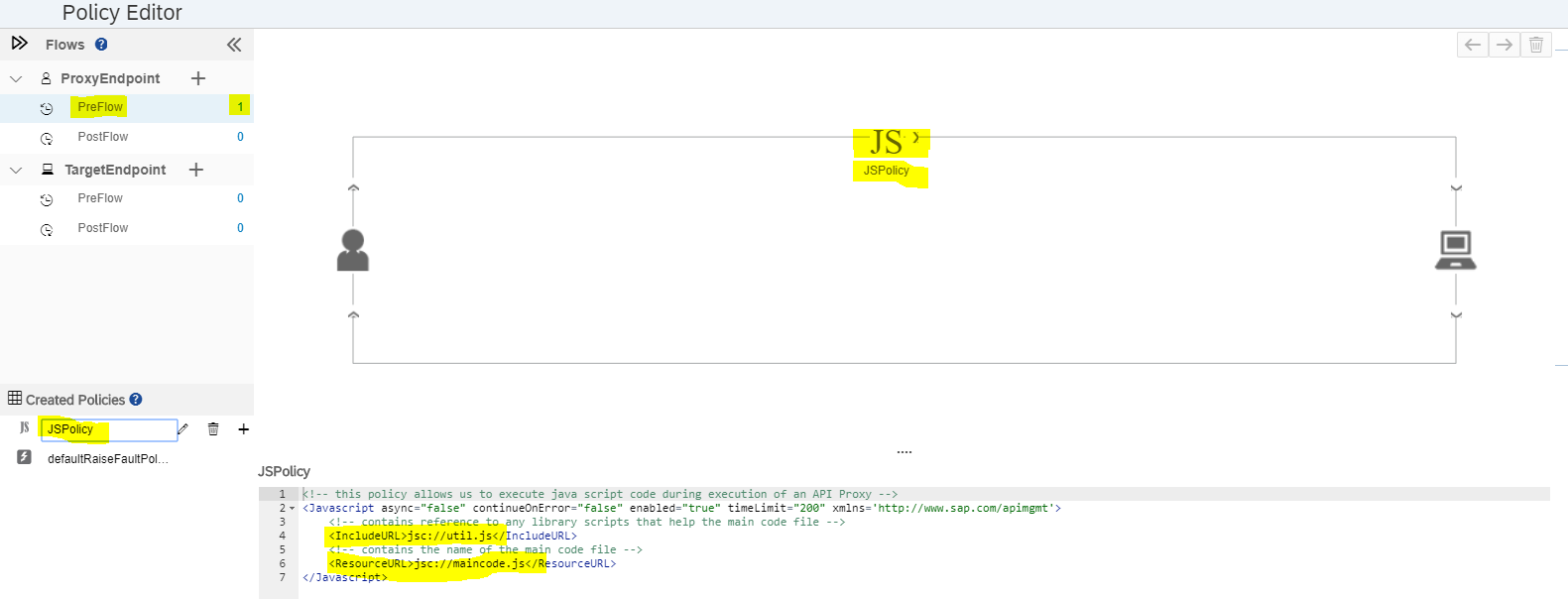
1. First select the PreFlow as shown above from top left corner and click the + symbol next to Javascript Extension Policy from the bottom right corner of the screen shown here.



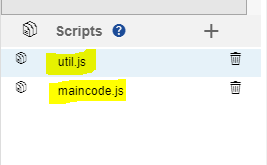
1. Once the Create Policy window is open enter the following details as shown here and click on Add.



1. Once created the changes in the screen looks like below –



1. The newly created JSPolicy xml includes the name of 2 .js files including **util.js** and **maincode.js**. Next, we need add these 2 scripts file (.js) with the same name inside the Scripts segment at the bottom left corner of the screen.



1. Once done open the **util.js** file and add the following code –

*function appendSuffix(){*

*var thingid = context.getVariable("request.queryparam.thingid")*

*context.setVariable("****dynamic\_id****", thingid);*

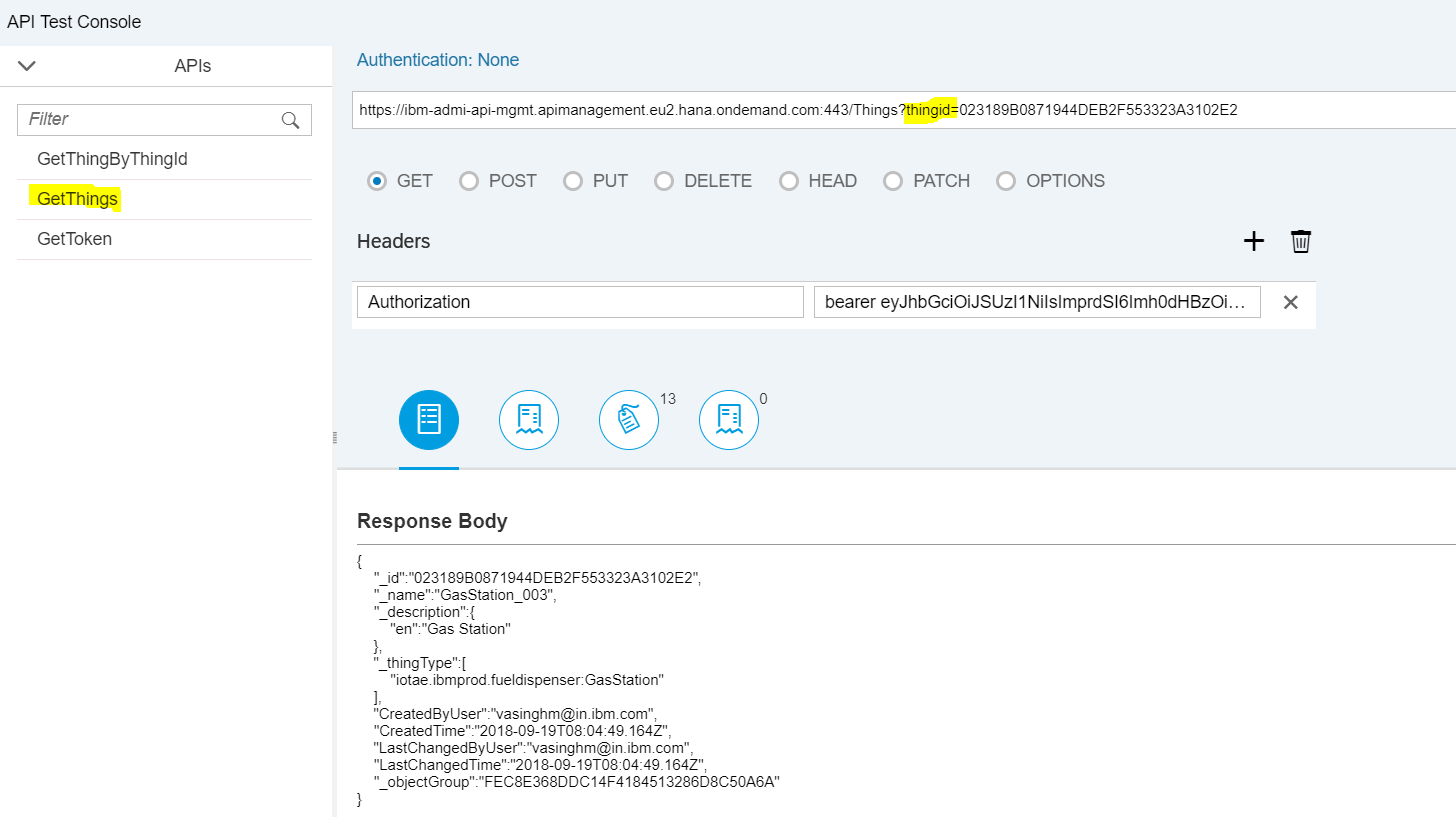
*}*

In the above code ***dynamic\_id*** is the dynamic parameter to be replaced with actual Thing Id at runtime.

1. And add the following inside **maincode.js** andclick **Update** to save the policy changes

*appendSuffix();*

1. Let’s now test the service using API Management Test Console and see if it works!



1. As shown above we are passing thingid as query parameter which is getting extracted and set to the context via Javascript policy defined above. Additionally we are passing Authorization token as a Header parameter which need to done every time you invoke this API from any application.