**Sample Application Migration Approach on CP4I from ICP**

Jun Shen

[junshen@au1.ibm.com](mailto:junshen@au1.ibm.com)

Version: 0.6

Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number** | **Date** | **Section** | **Changed By** | **Comments** |
| 0.1 | 16/02/2023 |  | Jun Shen | Initial creation |
| 0.2 | 01/03/2023 |  | Jun Shen | Add MQ session |
| 0.3 | 07/03/2023 |  | Jun Shen | Add SpringBoot session |
| 0.4 | 08/03/2023 |  | Jun Shen | Updated based on peer review |
| 0.5 | 14/03/2023 |  | Jun Shen | Updated based on WSV review |
| 0.6 | 14/03/2023 |  | Jun Shen | Add ArgoCD session |

Reference Documents

|  |  |
| --- | --- |
| **Name** | **Link** |
| CP4I topology design | CP4I Topology design for WSV-v0.7.docx |
| CP4I Deployment | CP4I Deployment for WSV Non-Prod v0.5.docx |

Table of Contents

[1. Introduction 5](#_Toc129674691)

[1.1 Overview 5](#_Toc129674692)

[1.2 Purpose of Document 5](#_Toc129674693)

[1.3 Scope of Document 5](#_Toc129674694)

[2. APIc Migration 6](#_Toc129674695)

[2.1 API Migration 6](#_Toc129674696)

[2.1.1 Clone APIC artefacts from Bitbucket 6](#_Toc129674697)

[2.1.2 Import API yaml file 6](#_Toc129674698)

[2.1.3 Migrate API v5c to v10 API Gateway 9](#_Toc129674699)

[2.2 API Gateway Custom Configuration 15](#_Toc129674700)

[2.2.1 Configure jws-cryto 15](#_Toc129674701)

[2.2.2 Configure JSONSetting on APIGW 17](#_Toc129674702)

[2.3 API Gateway Configuration for API Development 18](#_Toc129674703)

[2.3.1 Enable WebUI on APIGW 18](#_Toc129674704)

[2.3.2 Enable offload analytics to Splunk 21](#_Toc129674705)

[2.3.3 Enable API probe on API Gateway 22](#_Toc129674706)

[3. ACE Integraton Server Deployment 23](#_Toc129674707)

[3.1 Clone ACE Artefacts from Bitbucket 23](#_Toc129674708)

[3.2 ACE configuration 23](#_Toc129674709)

[3.2.1 Create BarAuth configuration 23](#_Toc129674710)

[3.2.2 Create mqkey.kdb configuration 24](#_Toc129674711)

[3.2.3 Create truststore.jks configuration 25](#_Toc129674712)

[3.2.4 Create keystore.jks configuration 25](#_Toc129674713)

[3.2.5 Create server.conf.yaml configuration 25](#_Toc129674714)

[3.2.6 Create odbc.ini configuration 26](#_Toc129674715)

[3.2.7 Create mqkey.rdb configuration 27](#_Toc129674716)

[3.2.8 Create mqkey.sth configuration 27](#_Toc129674717)

[3.2.9 Create setdbparms.txt configuration 28](#_Toc129674718)

[3.2.10 Create dbkey.p12 configuration 28](#_Toc129674719)

[3.3 ACE Integration Server deployment 28](#_Toc129674720)

[3.3.1 Create a policy bar file 28](#_Toc129674721)

[3.3.2 Create Integration Server deployment file 30](#_Toc129674722)

[3.3.3 Create Integration Server route 31](#_Toc129674723)

[4. DataPower Migration 34](#_Toc129674724)

[4.1 DataPower Configuration Migration 34](#_Toc129674725)

[4.2 DataPower Deployment 35](#_Toc129674726)

[4.3 DataPower Route Configuration 38](#_Toc129674727)

[5. MQ Deployment 41](#_Toc129674728)

[5.1 Queue Manager Configuration Migration 41](#_Toc129674729)

[5.1.1 Create a qm.ini file 41](#_Toc129674730)

[5.1.2 Remove the listener definition 41](#_Toc129674731)

[5.1.3 Update Channel configurations 41](#_Toc129674732)

[5.1.4 Configure User “app” permission 41](#_Toc129674733)

[5.1.5 Create a ConfigMap for MQ 42](#_Toc129674734)

[5.2 Queue Manager Deployment 43](#_Toc129674735)

[5.3 Create Queue Manager Route 45](#_Toc129674736)

[6. SpringBoot Migration 47](#_Toc129674737)

[6.1 SpringBoot Helm Chart Migration 47](#_Toc129674738)

[6.2 SpringBoot Deployment 48](#_Toc129674739)

[7. ArgoCD deployment 49](#_Toc129674740)

[7.1 Sample deployment repository 49](#_Toc129674741)

[7.2 Create Namespaces for application deployment 49](#_Toc129674742)

[7.3 Application deployment 49](#_Toc129674743)

[7.3.1 API Gateway apigw-ts1 49](#_Toc129674744)

[7.3.2 DataPower Gateway dpgw-ts1 50](#_Toc129674745)

[7.3.3 DataPower Gateway dpgw-ts1-priv 50](#_Toc129674746)

[7.3.4 Integration Servers 50](#_Toc129674747)

[7.3.5 MQ 51](#_Toc129674748)

[7.3.6 SpringBoot 51](#_Toc129674749)

[8. Troubleshooting 52](#_Toc129674750)

[8.1 Platform UI is not available 52](#_Toc129674751)

[8.2 API Gateway High Queued Status 53](#_Toc129674752)

# **Introduction**

* 1. **Overview**

Six application scenarios are migrated to CP4I NonProd environment. The migration document will introduce the approaches on API migration and CP4I component deployment of ACE, DataPower and MQ.

* 1. **Purpose of Document**

This document describes the detailed steps of API migration and the major steps of CP4I component deployment, such as ACE, DataPower and MQ. One scenario is described for each type of component. The developer could follow the same approach to migrate or deploy other same type of components.

The document focuses on the migration approach. It’s not a training document on the API and ACE application development and APIC/ACE/MQ/DataPower usage and configuration.

The auditor and reader of the document should have the following skills:

* OpenShift administration
* API development and Test
* ACE message flow development
* ACE integration server configuration
* MQ administration
* DataPower administration
* APIC administration
  1. **Scope of Document**

This document describes the detailed migration and configuration of the following components.

* APIC
  + API source code migration from v5c to API Gateway
  + API Gateway configuration for API development and testing
  + Custom configuration on API Gateway
* DataPower
  + Custom configuration
  + DataPower deployment
* Integration Server
  + Integration Server configuration
  + Integration Server deployment
* MQ
  + MQ configuration
  + MQ deployment
* SpringBoot deployment

Also a troubleshoot session is provided to cover two runtime issues for CP4I environment.

* PlatformUI not available issue
* API gateway high queued status

# **APIc Migration**

* 1. **API Migration**

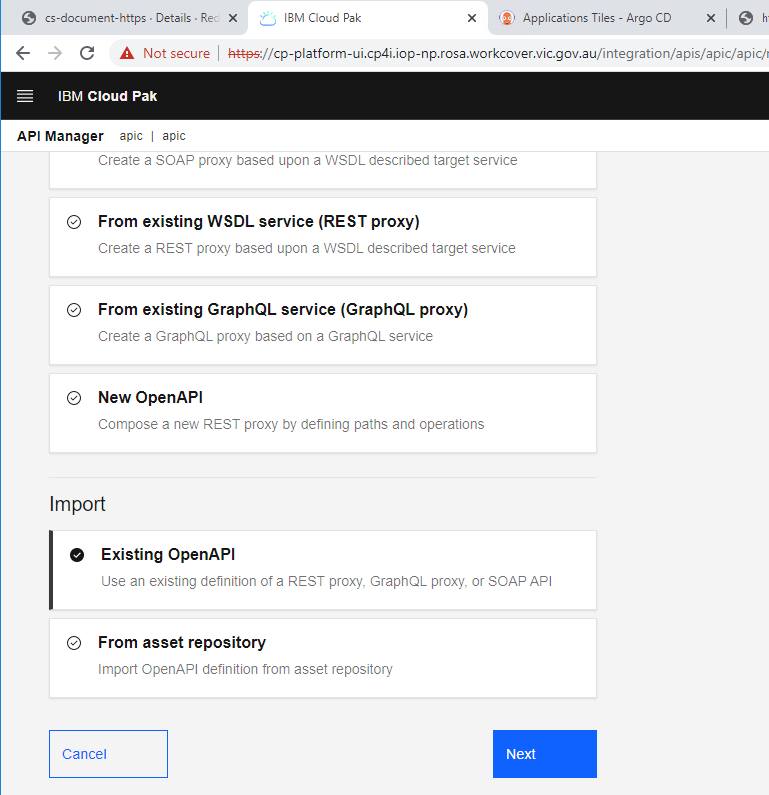
The API gateway type of ICP is v5 compatible, but the gateway type of CP4I is DataPower API Gateway. The scripts for the two types of gateway have a lot of differences, which require manual source code change for the API migration.

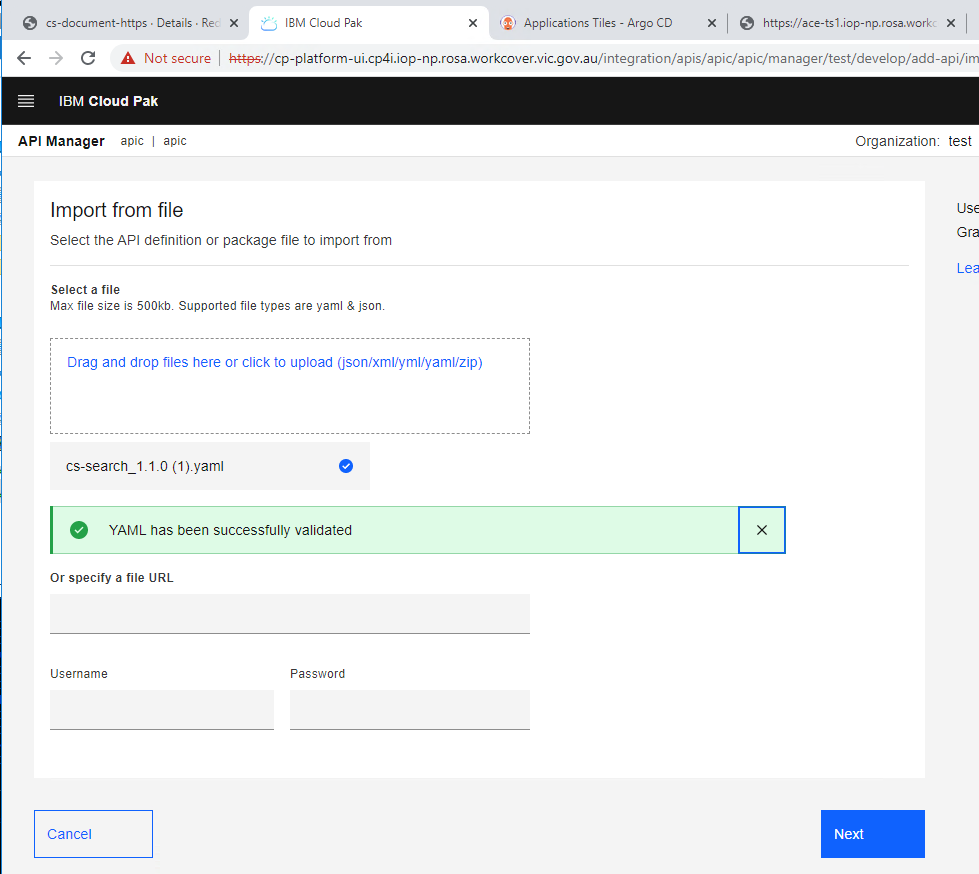
The following links provide some information on the difference between the two types of gateway:

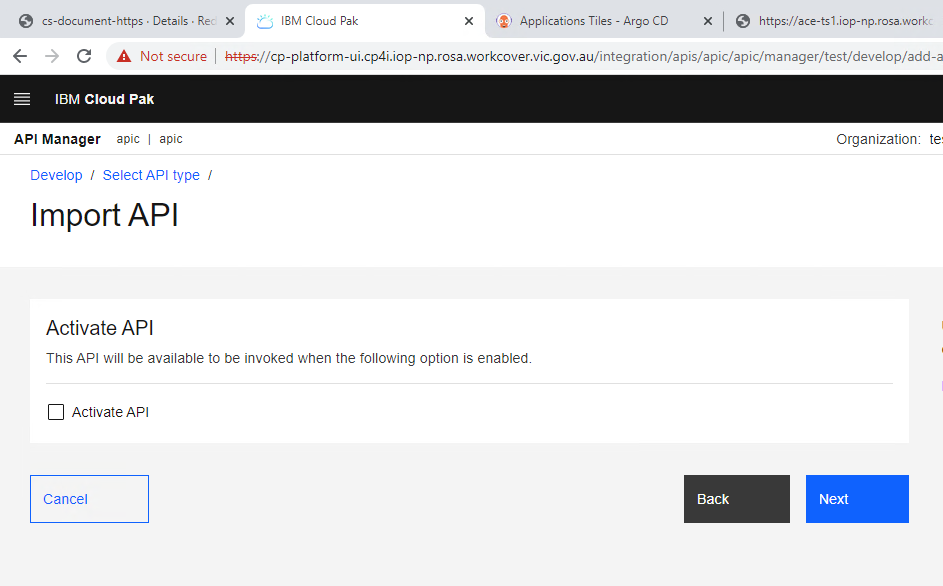
* [https://www.ibm.com/docs/en/api-connect/10.0.x?topic=overview-api-connect-gateway-types](https://aus01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fwww.ibm.com%2Fdocs%2Fen%2Fapi-connect%2F10.0.x%3Ftopic%3Doverview-api-connect-gateway-types&data=05%7C01%7Cjun_shen%40worksafe.vic.gov.au%7C9c82c5187af24cb1e5c408db22ffe9ed%7C7babb6c17c2f4b0e9da4589c4c4a3fca%7C0%7C0%7C638142254772966930%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C3000%7C%7C%7C&sdata=yPQULk1VtSASI5nPggQiF8rX2cKQ%2BfRYSh82bXSNWX0%3D&reserved=0)
* [https://www.ibm.com/docs/en/api-connect/10.0.5.x\_lts?topic=constructs-built-in-policies](https://aus01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fwww.ibm.com%2Fdocs%2Fen%2Fapi-connect%2F10.0.5.x_lts%3Ftopic%3Dconstructs-built-in-policies&data=05%7C01%7Cjun_shen%40worksafe.vic.gov.au%7C9c82c5187af24cb1e5c408db22ffe9ed%7C7babb6c17c2f4b0e9da4589c4c4a3fca%7C0%7C0%7C638142254772966930%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C3000%7C%7C%7C&sdata=qsN9XpeY%2BauagtRofNdJnI6pCGLYJ480iu9GglhRuto%3D&reserved=0)
  + 1. **Clone APIC artefacts from Bitbucket**

git clone <https://bitbucket-aws.itss.vic.gov.au/scm/rosa/rosa-app-migration.git>

* + 1. **Import API yaml file**







* + 1. **Migrate API v5c to v10 API Gateway**
* **Change the gateway type**

From

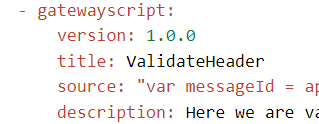


To

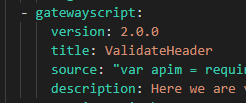


* **Change the policy version from 1.0.0 to 2.0.0**

Eg, from

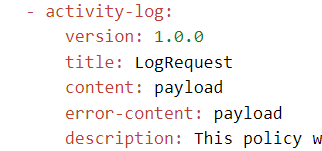


To

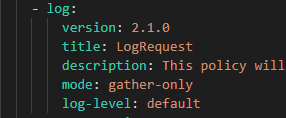


* **Change activity-log**

From

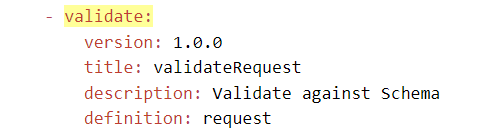


To

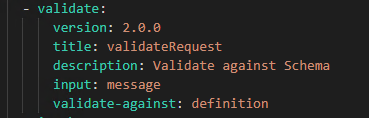


* **Update validate policy**

From



To



* **Add “apim” var before calling apim.getvariable**

From

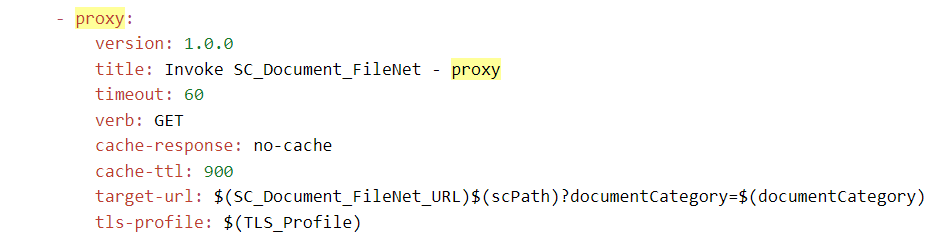


To

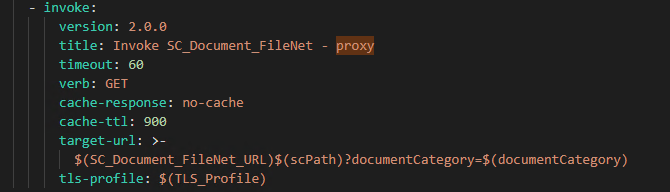


* **Change proxy policy**

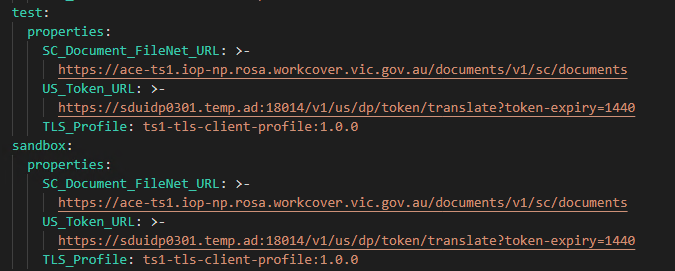
From



To

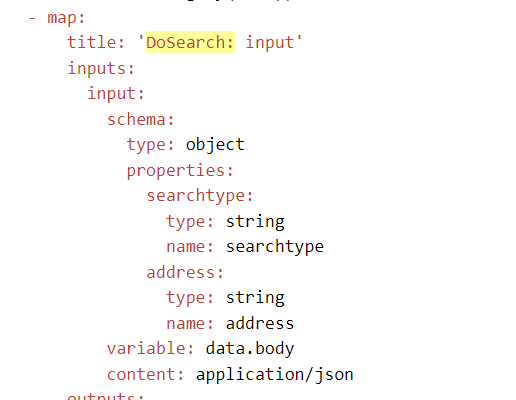


* **Add Sandbox and Test catalog properties**

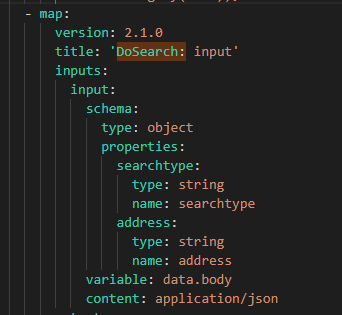


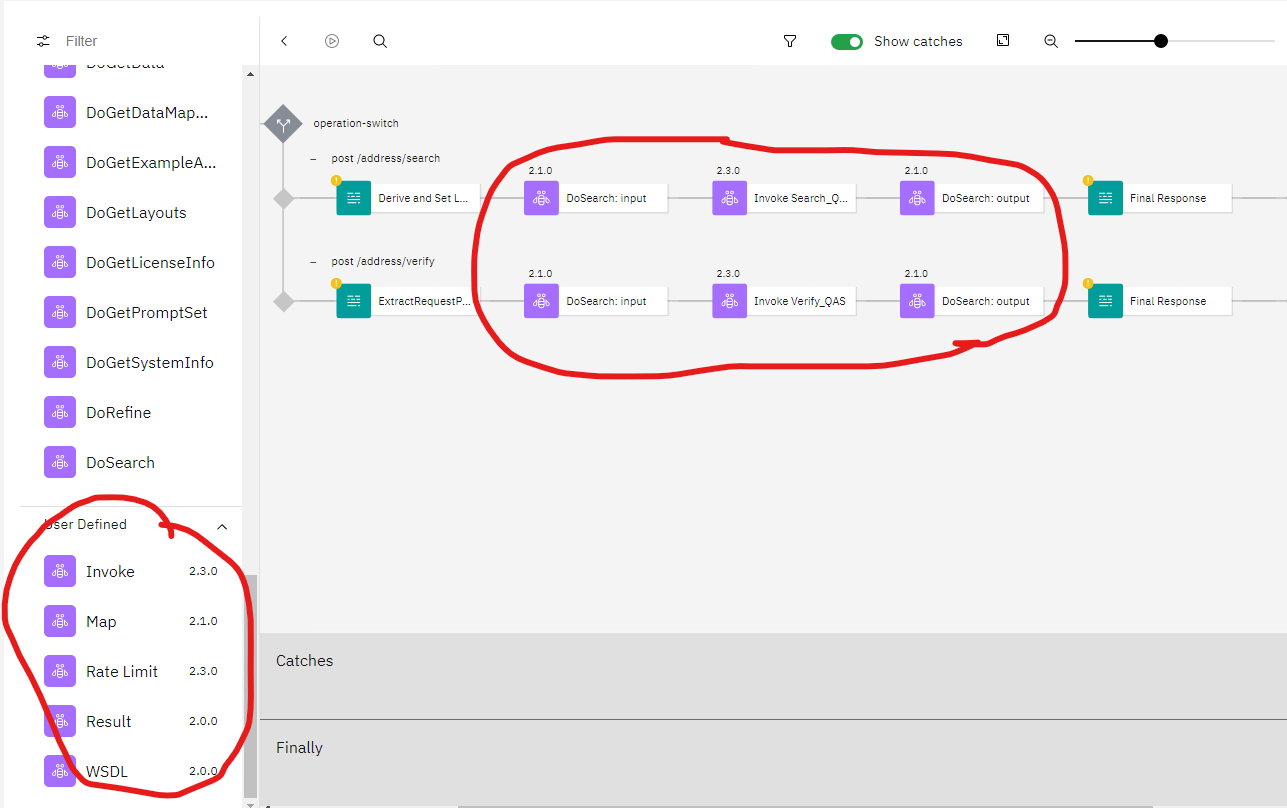
* **Set version number for User Defined Policy**

From

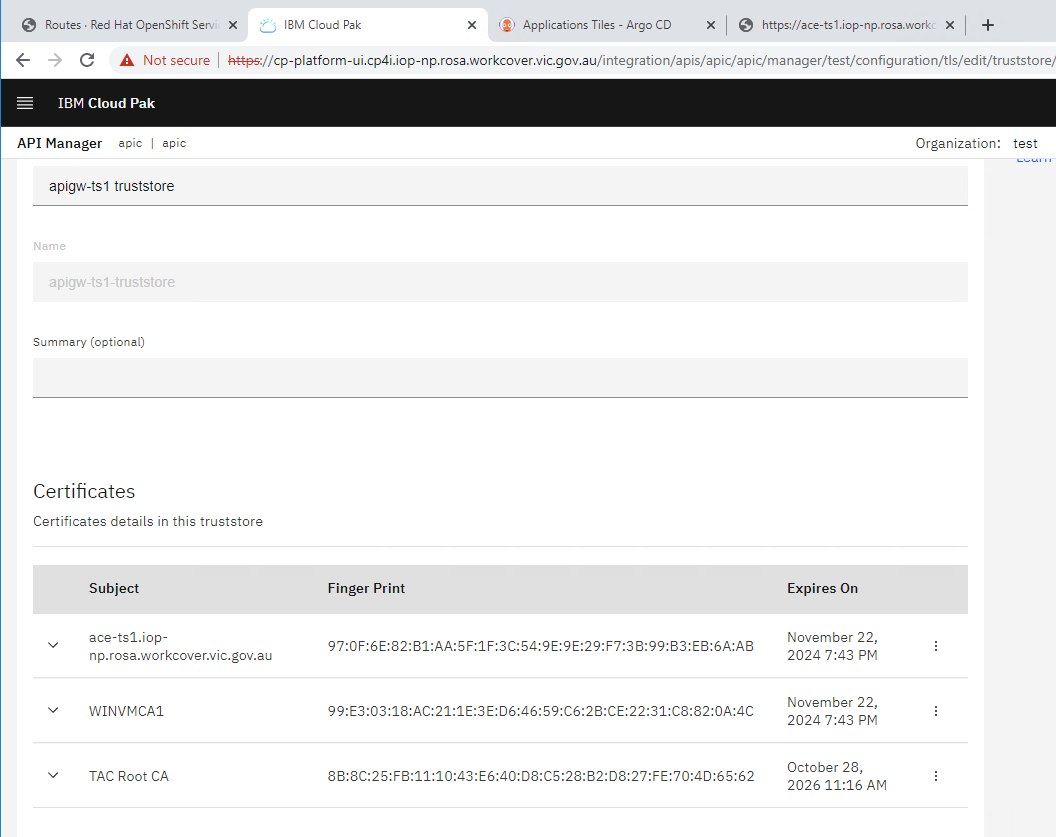


To

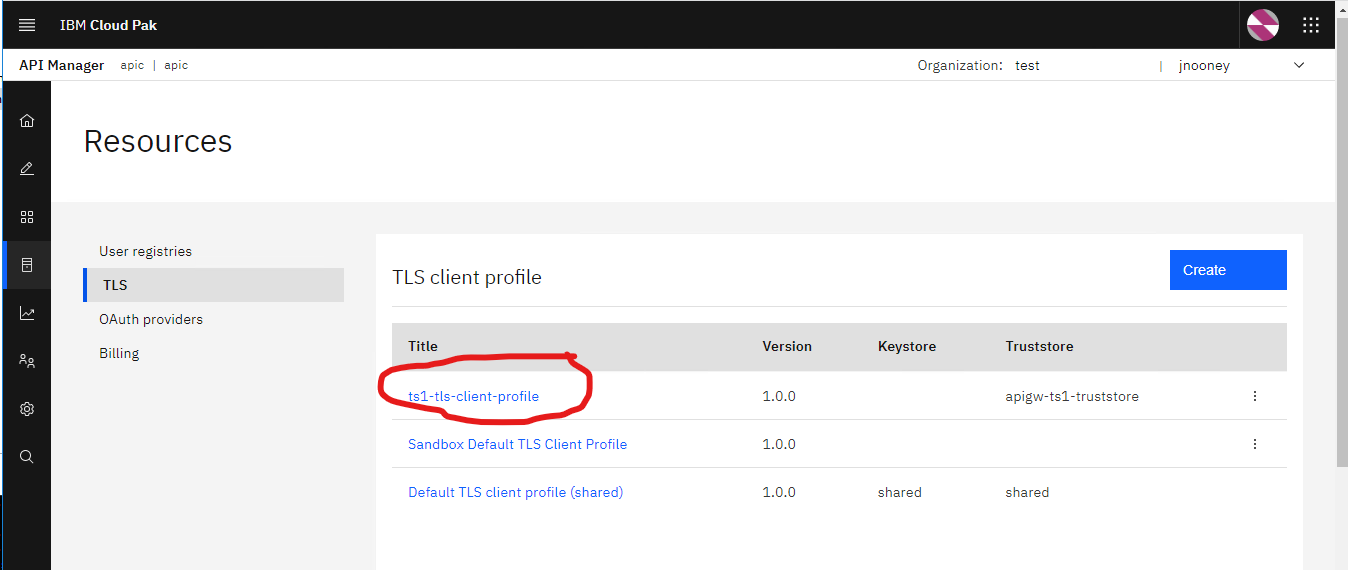




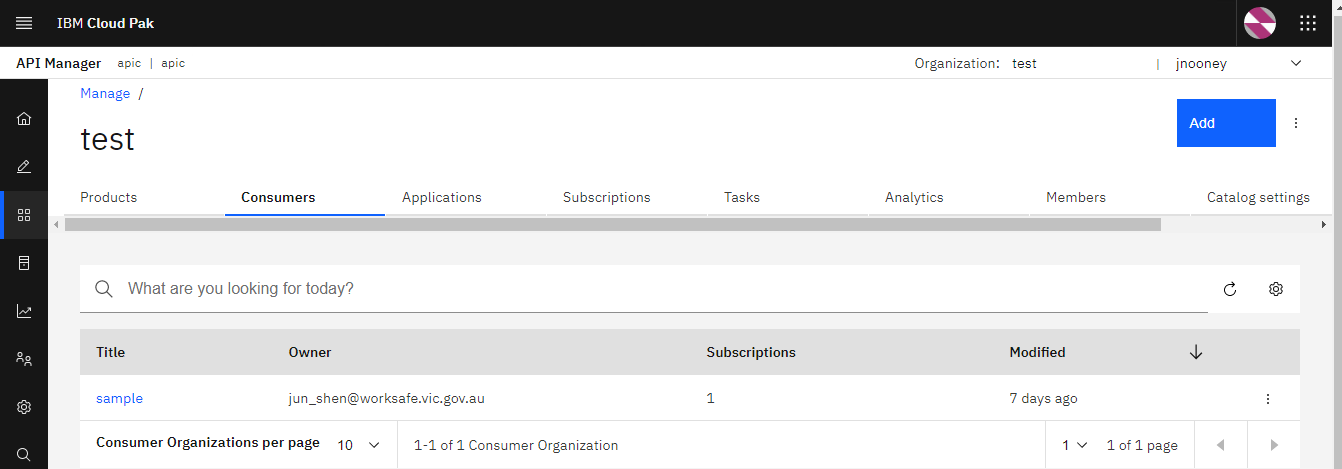
* **Create a truststore**



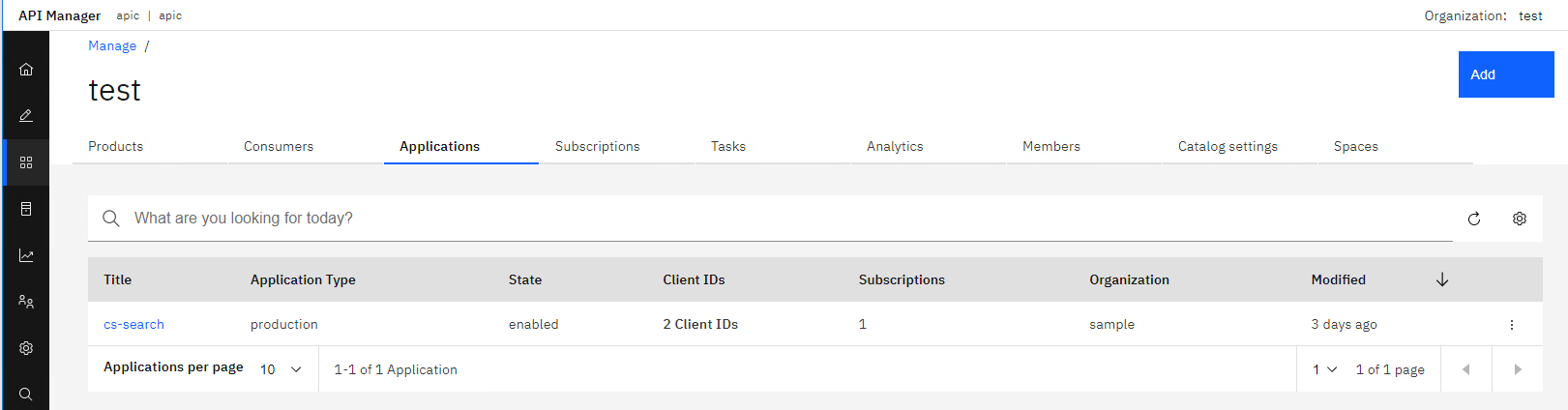
* **Create a TLS client profile**



* **Create a consumer org**



* **Create an application**

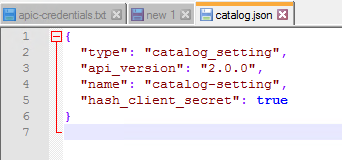


* **Import subscriptions**

1. **Login API Manager**

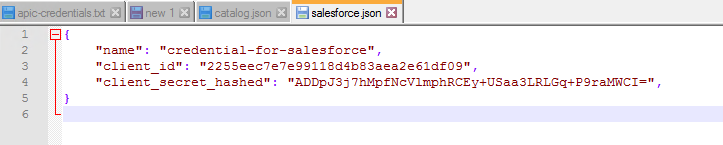
apic login --username jnooney --password Aditya@369 --server https://cp-platform-ui.cp4i.iop-np.rosa.workcover.vic.gov.au/integration/apis/apic/apic --realm provider/default-idp-2

1. **Enable “hash\_client\_secret” for Catalog**



apic catalog-settings:update --server https://cp-platform-ui.cp4i.iop-np.rosa.workcover.vic.gov.au/integration/apis/apic/apic --org test --catalog test catalog.json

1. **Import subscriptions**



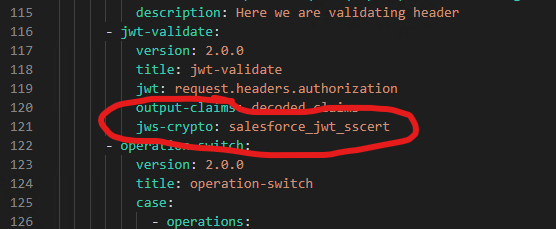
apic credentials:create --server https://cp-platform-ui.cp4i.iop-np.rosa.workcover.vic.gov.au/integration/apis/apic/apic --org test --catalog test --consumer-org sample --app cs\_search salesforce.json --format json --output -

1. **List subscriptions**

apic credentials:list --server https://cp-platform-ui.cp4i.iop-np.rosa.workcover.vic.gov.au/integration/apis/apic/apic --org test --catalog test --consumer-org sample --app cs\_search

* 1. **API Gateway Custom Configuration**
     1. **Configure jws-cryto**

If an API has jwt-validate policy, need deploy the jws-cryto to APIGW.



* **First create a secret for the certificate.**

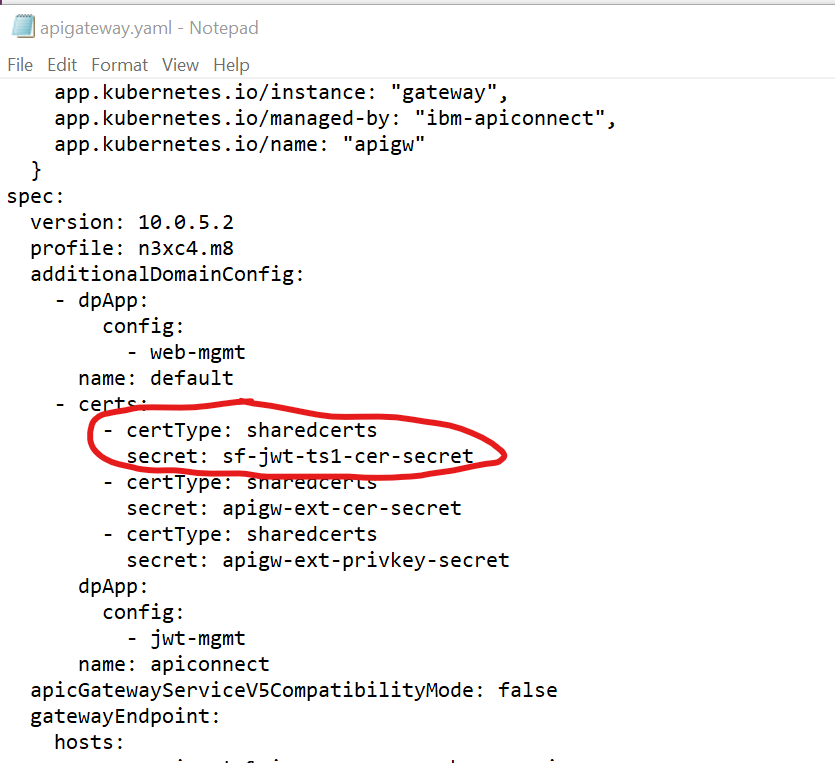
oc create secret generic sf-jwt-ts1-cer-secret --from-file=sf-jwt-ts1.cer -n apigw-ts1

* **Second create a configmap for it.**



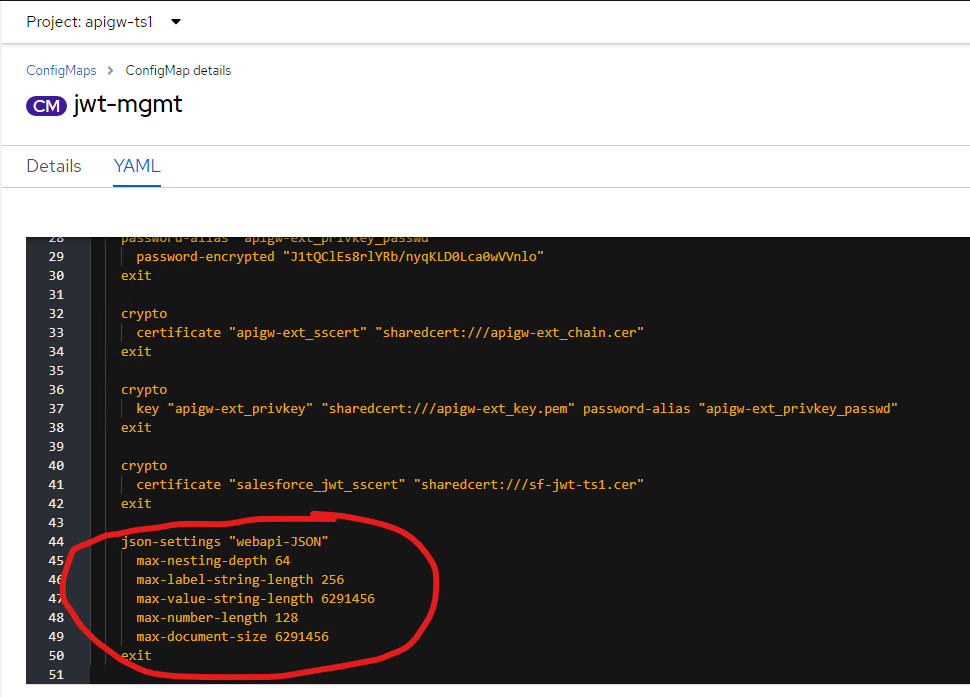
* **Last update APIGW CR.**

APIGW CR link: <https://console-openshift-console.apps.rosa-iop-np.b3hz.p1.openshiftapps.com/k8s/ns/apigw-ts1/clusterserviceversions/ibm-apiconnect.v3.2.1/gateway.apiconnect.ibm.com~v1beta1~GatewayCluster/apigw/yaml>



* + 1. **Configure JSONSetting on APIGW**
* **Add JSONSetting in APIGW jwt-mgmt configmap**

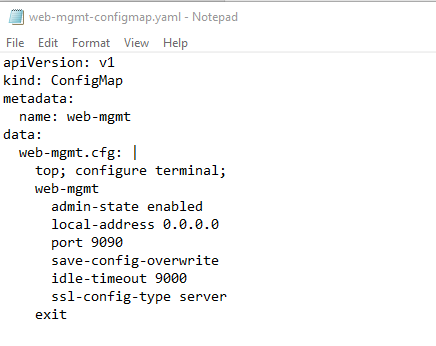
The jwt-mgmt configmap link: <https://console-openshift-console.apps.rosa-iop-np.b3hz.p1.openshiftapps.com/k8s/ns/apigw-ts1/configmaps/jwt-mgmt/yaml>



* 1. **API Gateway Configuration for API Development**

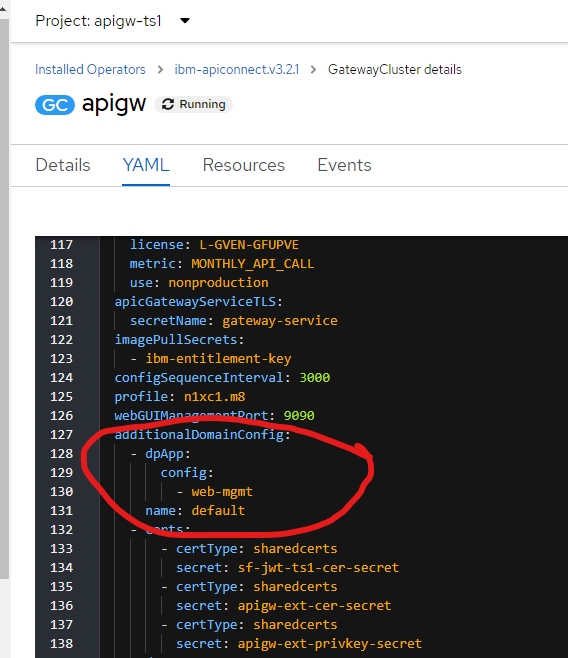
Some configurations are enabled on API gateway, which are helpful for API development and test.

* + 1. **Enable WebUI on APIGW**
* **Create a web-mgmt ConfigMap**

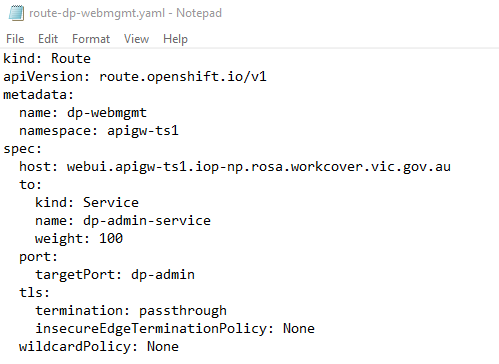


* **Update APIGW CR.**

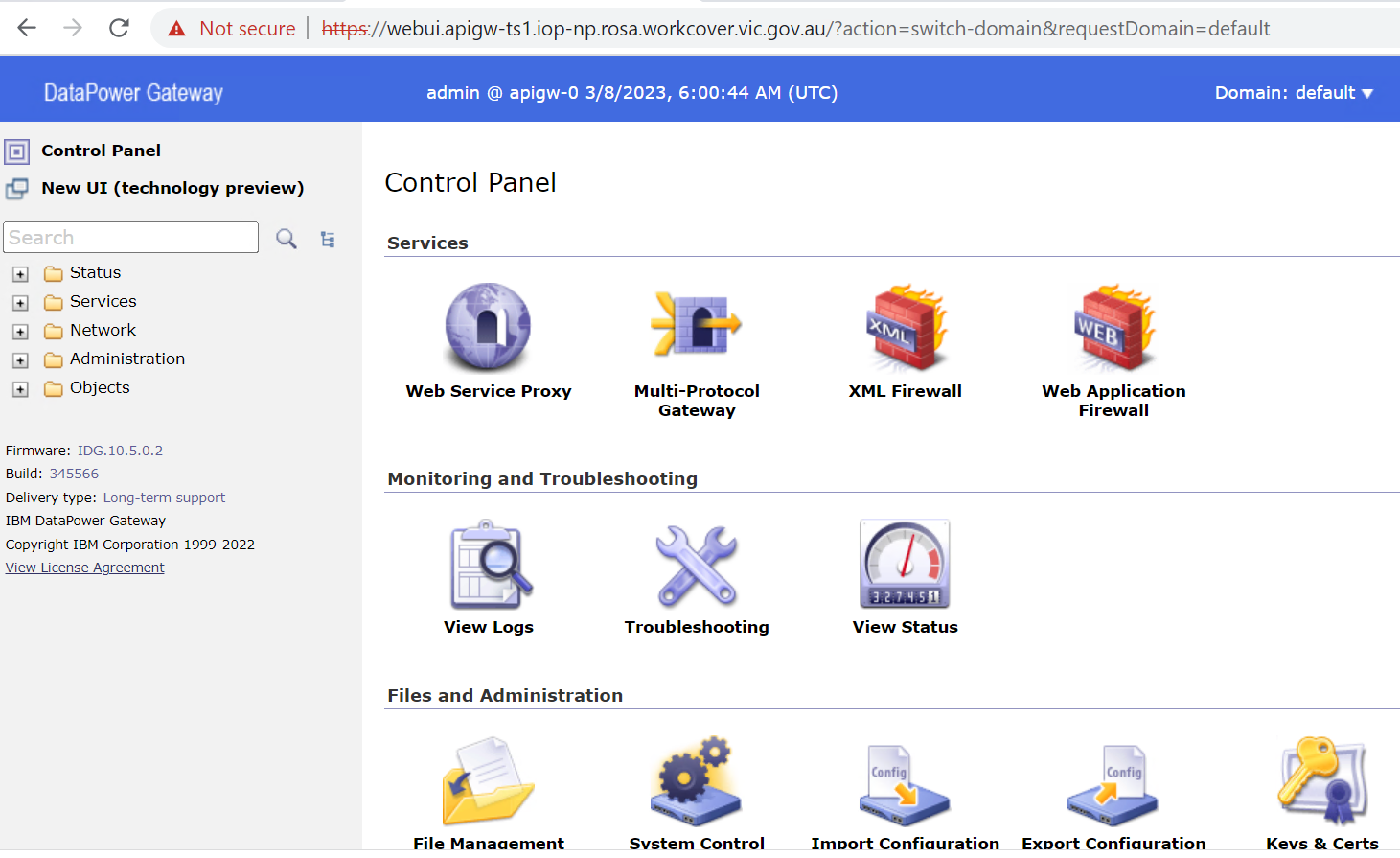
APIGW CR link: <https://console-openshift-console.apps.rosa-iop-np.b3hz.p1.openshiftapps.com/k8s/ns/apigw-ts1/clusterserviceversions/ibm-apiconnect.v3.2.1/gateway.apiconnect.ibm.com~v1beta1~GatewayCluster/apigw/yaml>



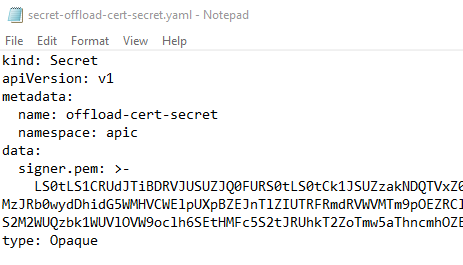
* **Create a route for WebUI.**



* **Open WebUI via** [**https://webui.apigw-ts1.iop-np.rosa.workcover.vic.gov.au**](https://webui.apigw-ts1.iop-np.rosa.workcover.vic.gov.au)

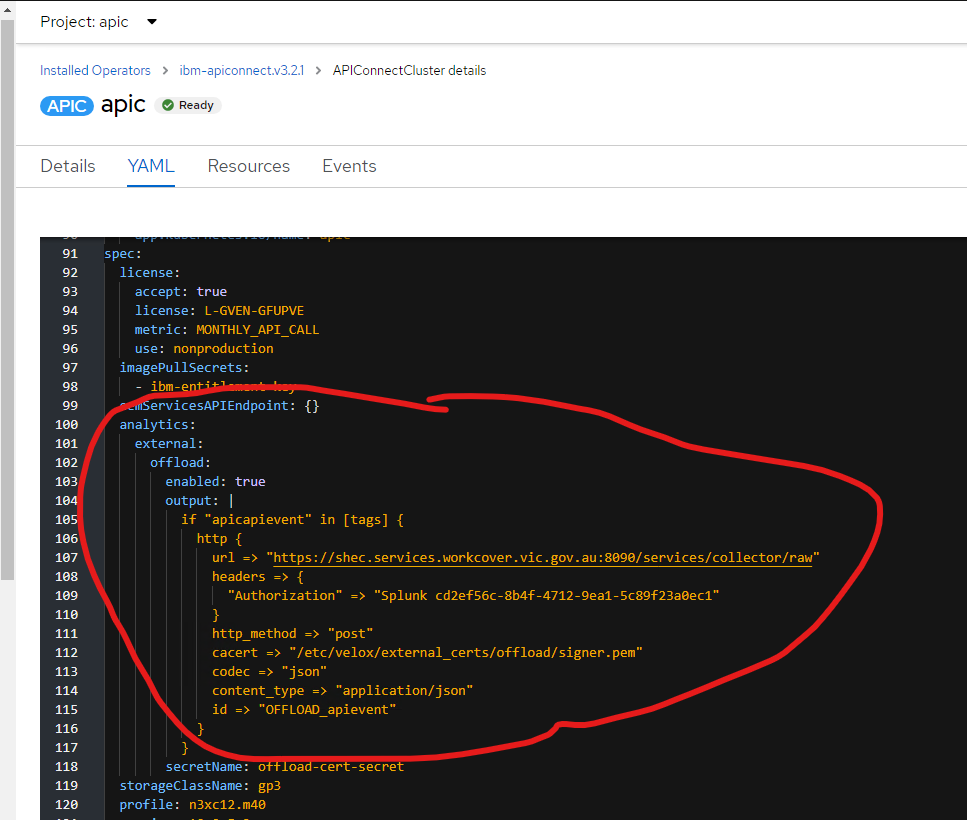


* + 1. **Enable offload analytics to Splunk**
* **Create a secret for offload certificate**



* **Update APIConnectCluster CR**

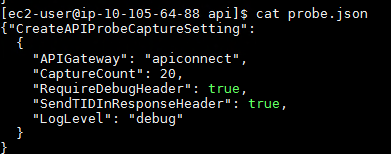
APIConnectCluster CR link: <https://console-openshift-console.apps.rosa-iop-np.b3hz.p1.openshiftapps.com/k8s/ns/apic/clusterserviceversions/ibm-apiconnect.v3.2.1/apiconnect.ibm.com~v1beta1~APIConnectCluster/apic/yaml>



* + 1. **Enable API probe on API Gateway**

When the API probe capture setting is configured to capture data only for transactions that have the **APIm-Debug: true** header

* **Create a probe.json file**



* **Creating an API probe capture setting via the following command**

curl -k -u admin:'`/)3TwOq' -H 'Content-type: application/json' https://rest.apigw-ts1.iop-np.rosa.workcover.vic.gov.au/mgmt/actionqueue/apiconnect -d @probe.json

# **ACE Integraton Server Deployment**

Not like ICP environment, ACE integration server is deployed via flying mode without building a custom ACE image. This approach is more efficient than building a custom image and it’s not necessary to maintain the ACE image version.

ACE message flows are not required to make any change. Only ACE configurations and the integration server CR are required to create and deploy on CP4I.

* 1. **Clone ACE Artefacts from Bitbucket**

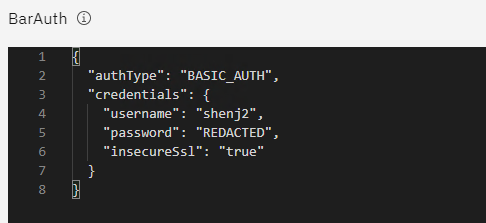
All existing ACE artefacts from ICP are located in Bitbucket. Run the follow command to clone its repository to your local machine.

git clone <https://bitbucket-aws.itss.vic.gov.au/scm/rosa/rosa-app-migration.git>

* 1. **ACE configuration**
     1. **Create BarAuth configuration**

Reference link: <https://www.ibm.com/docs/en/app-connect/container?topic=types-barauth-type>

* **Define the JSON content**



* **Create a secret**

oc create secret generic barauth-55dd8 --from-file=configuration=/mytempdir/barauth.json --namespace=iop-ts1

* **Create BarAuth yaml file**

apiVersion: appconnect.ibm.com/v1beta1

kind: Configuration

metadata:

  name: barauth

  namespace: iop-ts1

spec:

  secretName: barauth-55dd8

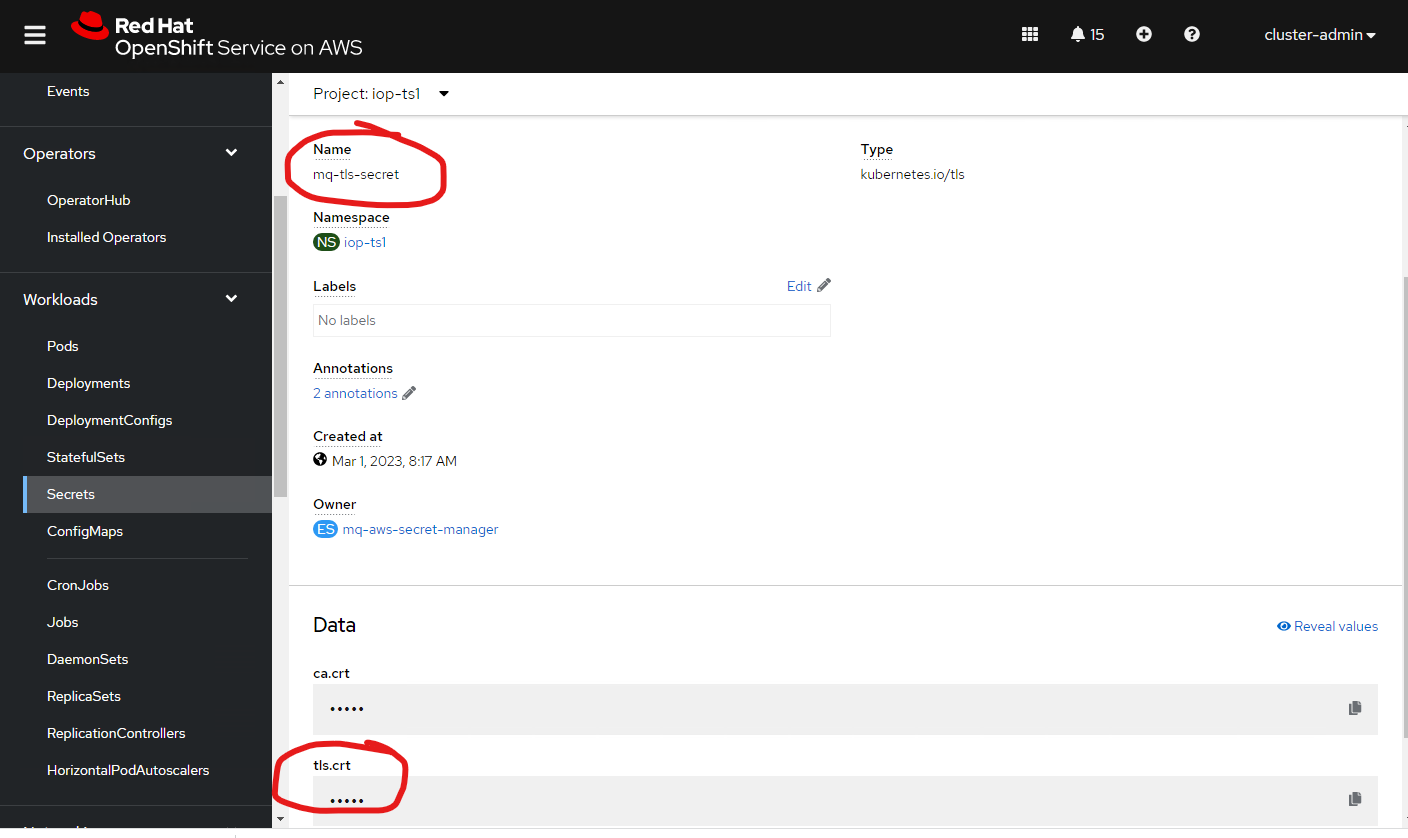
  type: barauth

  version: 12.0.7.0-r2

* **Apply the yaml file**

oc apply -f barauth.yaml

* + 1. **Create mqkey.kdb configuration**
* **Save MQ certificate as a file: TS1TM01.cer**



* **Import TS1TM01.cer to mqkey.kdb**

First copy mqkey.kdb, mqkey.rdb, mqkey.sth and TS1TM01.cer to a MQ server.

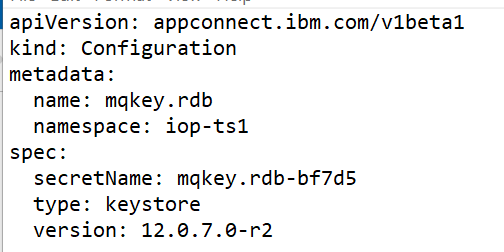
Then run the command on a MQ server.

runmqakm -cert -add -db mqkey.kdb -label mqservercert -file TS1TM01.cer -format ascii -stashed

* **Create a mqkey.kdb secret**

oc create secret generic mqkey.kdb-7sr6b --from-file=configuration=mqkey.kdb --namespace=iop-ts1

* **Create mqkey.kdb yaml file and apply the file via oc**



* + 1. **Create truststore.jks configuration**
* **Install keytool**

Download keytool from the link <https://sourceforge.net/projects/keytool/>, then install it on your development machine.

* **Create truststore.jks**

keytool -import -file ace-ts1-tls.crt -alias aceroot2 -keystore truststore.jks -storepass changeit

keytool -import -file truststore-tacroot.crt -alias tacroot -keystore truststore.jks -storepass changeit

keytool -import -file truststore-wsvroot.crt -alias wsvroot -keystore truststore.jks -storepass changeit

keytool -import -file truststore-filenet.crt -alias filenet -keystore truststore.jks -storepass changeit

keytool -import -file truststore-mdm.crt -alias mdm -keystore truststore.jks -storepass changeit

keytool -import -file truststore-mdmroot.crt -alias mdmroot -keystore truststore.jks -storepass changeit

keytool -import -file apigw-key.crt -alias apigwts1 -keystore truststore.jks -storepass changeit

* **Create configuration via the same approach as mqkey.kdb**
  + 1. **Create keystore.jks configuration**
* **Create keystore.jks**

openssl pkcs12 -export -in ace-ts1-tls.crt -inkey ace-ts1-tls.key -certfile ace-ts1-ca.crt -out ace.p12 -passout pass:changeit -name acekey

keytool -importkeystore -srckeystore ace.p12 -srcstoretype PKCS12 -srcstorepass changeit -destkeystore keystore.jks  -deststoretype JKS -deststorepass changeit

* **Create a configuration via the same approach as mqkey.kdb**
  + 1. **Create server.conf.yaml configuration**
* **Update the server.conf.yaml**

1. mqKeyRepository path

From



To



1. truststoreFile path

From



To



1. jvmSystemProperty path

From



To

jvmSystemProperty: '-Dlog4j.configurationFile=/home/aceuser/generic/iop-log4j2.xml'

1. KeystoreFile path

From



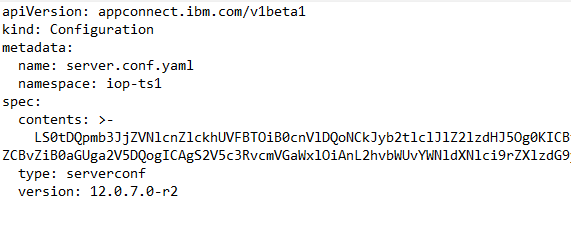
To



* **Convert the file content to base64**

cat server.conf.yaml | base64

* **Create a configuration yaml file and apply it on OpenShift**



* + 1. **Create odbc.ini configuration**
* **Update odbc.ini**

1. KeyStore=/home/aceuser/generic/dbkey.p12
2. TrustStore=/home/aceuser/generic/dbkey.p12

* **Convert the file content to base64**

cat odbc.ini | base64

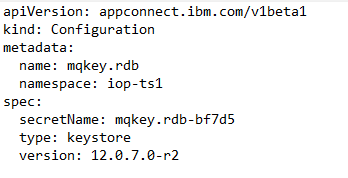
* **Create a configuration yaml file and apply it on OpenShift**



* + 1. **Create mqkey.rdb configuration**
* **Create a secret**

oc create secret generic mqkey.rdb-bf7d5 --from-file=configuration=mqkey.rdb --namespace=iop-ts1

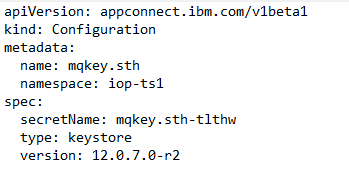
* **Create mqkey.rdb configuration yaml file and apply the file via oc**



* + 1. **Create mqkey.sth configuration**
* **Create a secret**

oc create secret generic mqkey.sth-tlthw --from-file=configuration=mqkey.sth --namespace=iop-ts1

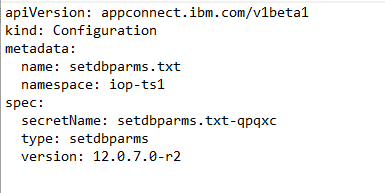
* **Create mqkey.sth configuration yaml file and apply the file via oc**



* + 1. **Create setdbparms.txt configuration**
* **Create a secret**

oc create secret generic setdbparms.txt-qpqxc --from-file=configuration= setdbparms.txt --namespace=iop-ts1

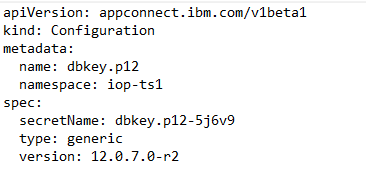
* **Create setdbparms.txt configuration yaml file and apply the file via oc**



* + 1. **Create dbkey.p12 configuration**
* **Create a secret**

oc create secret generic dbkey.p12-5j6v9 --from-file=configuration=dbkey.p12 --namespace=iop-ts1

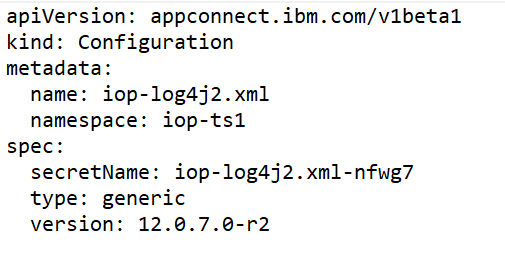
* **Create dbkey.p12 configuration yaml file and apply the file via oc**



* + 1. **Create iop-log4j2.xml configuration**
* **Create a secret**

oc create secret generic iop-log4j2.xml-nfwg7 --from-file=configuration=iop-log4j2.xml --namespace=iop-ts1

* **Create iop-log4j2.xml configuration yaml file and apply the file via oc**

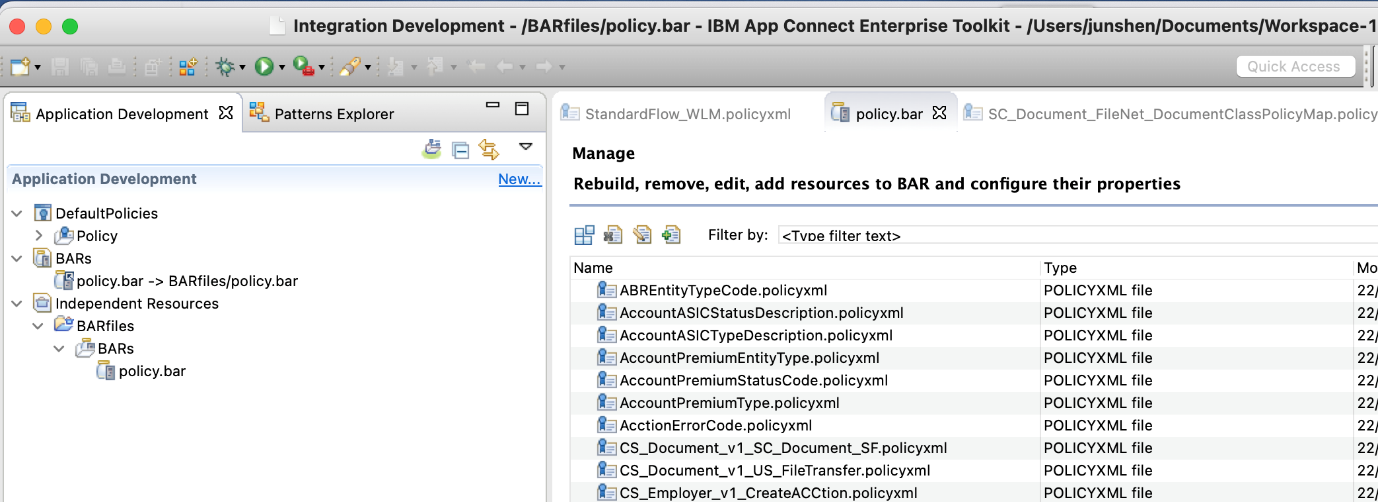


* 1. **ACE Integration Server deployment**
     1. **Create a policy bar file**

The ACE repository missed the policy bar file for Integration Servers. So have to create it for ACE deployment.

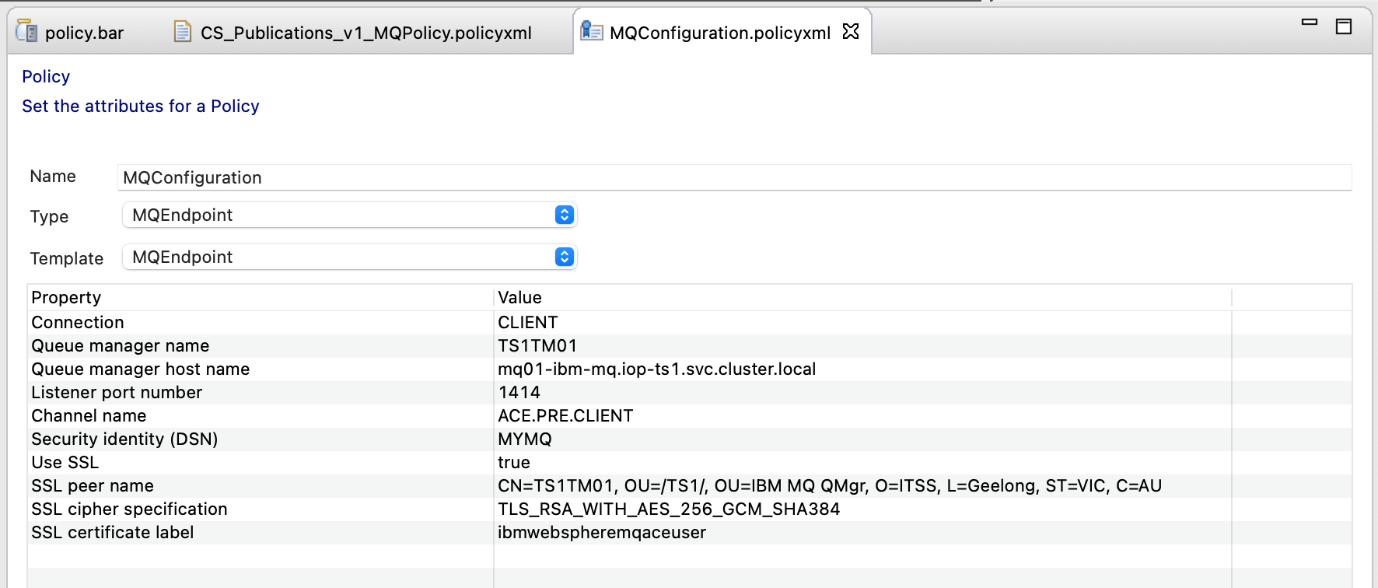
There are two approaches to create a policy bar file, ACE toolkit and the command mqsicreatebar. ACE toolkit is used in this document.

* **Create a policy project in ACE toolkit**

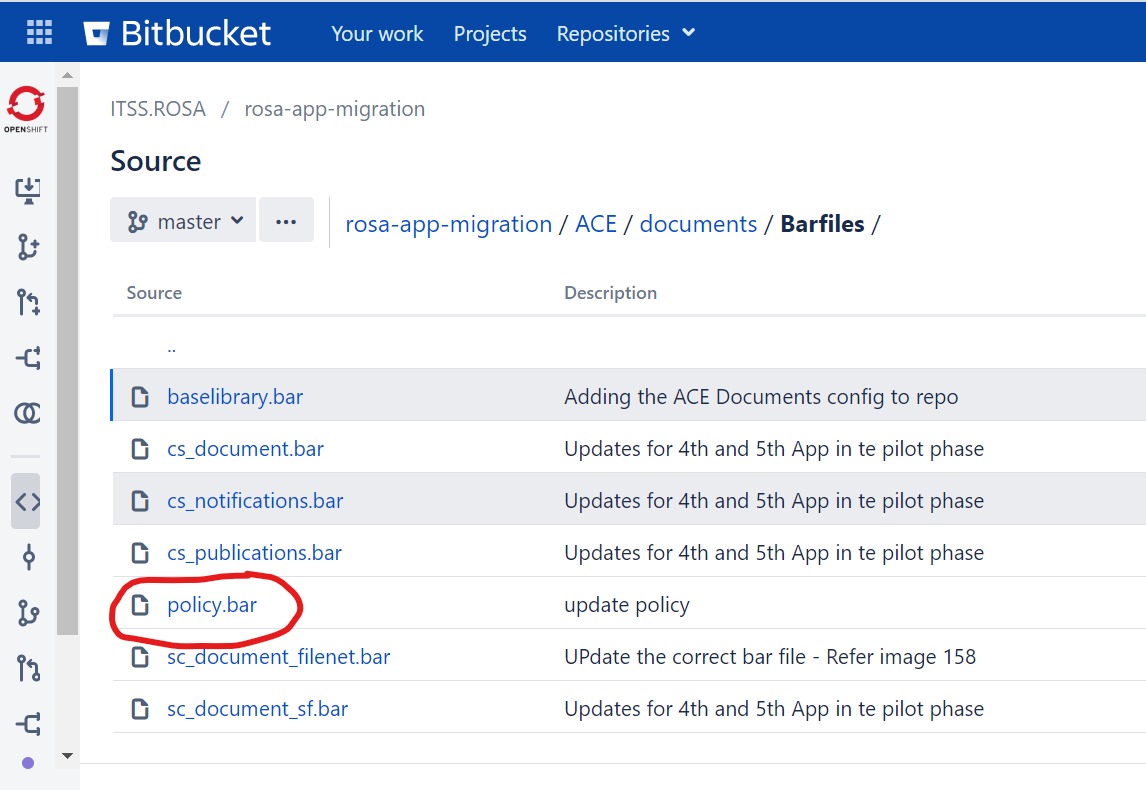


* **Update MQ policy**

1. Queue manager name
2. Host name
3. SSL peer name
4. SSL certificate label



* **Export the policy bar and upload to rosa-app-migration Bitbucket**



* + 1. **Create Integration Server deployment file**

apiVersion: appconnect.ibm.com/v1beta1

kind: IntegrationServer

metadata:

  name: is-cs-search

  namespace: iop-ts1

spec:

  enableMetrics: true

  license:

    accept: true

    license: L-APEH-CJUCNR

    use: CloudPakForIntegrationProduction

  pod:

    containers:

      runtime:

        resources:

          limits:

            cpu: 300m

            memory: 368Mi

          requests:

            cpu: 300m

            memory: 368Mi

  adminServerSecure: true

  router:

    timeout: 120s

  designerFlowsOperationMode: disabled

  createDashboardUsers: true

  service:

    endpointType: https

  version: 12.0.7.0-r2

  replicas: 1

  barURL: >-

    https://bitbucket-aws.itss.vic.gov.au/projects/ROSA/repos/rosa-app-migration/raw/ACE/search/Bar%20Files/baselibrary.bar,https://bitbucket-aws.itss.vic.gov.au/projects/ROSA/repos/rosa-app-migration/raw/ACE/search/Bar%20Files/sc\_search\_rds.bar

  configurations:

    - barauth

    - dbkey.p12

    - iop-log4j2.xml

    - keystore.jks

    - odbc.ini

    - setdbparms.txt

    - truststore.jks

    - server.conf.yaml

    - mqkey.kdb

    - mqkey.rdb

    - mqkey.sth

* **Apply the yaml file**

oc apply -f is-cs-search.yaml

* + 1. **Create Integration Server route**

There are two approaches to define the route URL of Integration Servers which are deployed in the same namespace.

1. All Integration Servers shared the same hostname, but use different paths to distinguish different services. Because OpenShift Passthrough Route doesn’t support this pattern (hostname/path), the Reencrypt Route is used to implement this route.
2. Different Integration Servers define different hostnames. The Passthrough Route could be used to implement this route.

* **Reencrypt Route**

The Reencrypt Route is the current implementation for the sample ACE applications because only one hostname, ace-ts1.iop-np.rosa.workcover.vic.gov.au, is defined all ACE servers in Namespace iop-ts1. The route CR file is shown below.

kind: Route

apiVersion: route.openshift.io/v1

metadata:

  name: cs-search-https

  namespace: iop-ts1

  annotations:

    haproxy.router.openshift.io/rewrite-target: /

spec:

  host: ace-ts1.iop-np.rosa.workcover.vic.gov.au

  path: /search

  to:

    kind: Service

    name: is-cs-search-is

    weight: 100

  port:

    targetPort: https

  tls:

    termination: reencrypt

    certificate: |-

      -----BEGIN CERTIFICATE-----

      -----END CERTIFICATE-----

    key: |-

      -----BEGIN PRIVATE KEY-----

      -----END PRIVATE KEY-----

    caCertificate: |-

      -----BEGIN CERTIFICATE-----

      -----END CERTIFICATE-----

    destinationCACertificate: |-

      -----BEGIN CERTIFICATE-----

      -----END CERTIFICATE-----

    insecureEdgeTerminationPolicy: Redirect

  wildcardPolicy: None

1. Certificate is the certificate of the ACE server
2. Key is the key of the ACE server
3. caCertificate is the CA certificate to sign the certificate of the ACE server, which is WINVMCA1 and TACROOT
4. destinationCACertificate is also the certificate of the ACE server

* **Passthrough Route**

Passthrough Route is recommended to implement for ACE HTTPS services because it provides better performance than Reencrypt Route. But it requires to change the ACE Endpoints and its related Certificates defined in <https://wiki.worksafe.vic.gov.au/pages/viewpage.action?spaceKey=AIPR&title=CP4I+Domain%2C+Endpoints+and+Certificates>

For example, the passthrough route for ACE cs-search service is defined as below.

kind: Route

apiVersion: route.openshift.io/v1

metadata:

  name: cs-search-route

  namespace: iop-ts1

spec:

  host: search.ace-ts1.iop-np.rosa.workcover.vic.gov.au

  to:

    kind: Service

    name: is-cs-search-is

    weight: 100

  port:

    targetPort: https

  tls:

    termination: passthrough

    insecureEdgeTerminationPolicy: Redirect

  wildcardPolicy: None

# **DataPower Migration**

* 1. **DataPower Configuration Migration**

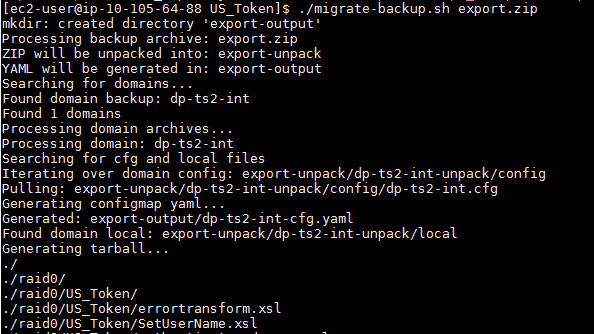
DataPowers in ICP environment are virtual machines, which are required to migrate to the Container environment in CP4I.

Follow the below steps to migrate DataPower:

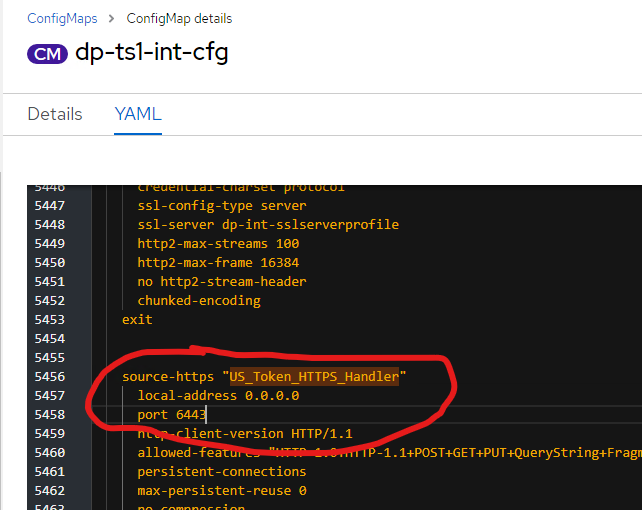
* **Export DataPower domain from ICP**
* **Download the transfer tool from the link:**

<https://github.com/IBM/datapower-operator-scripts#migrate-backupsh>

* **Transfer the DataPower domain export file to ConfigMap**



* **Update US\_Token\_HTTPS\_Handler local address and port in dp-ts1-int-cfg.yaml.**



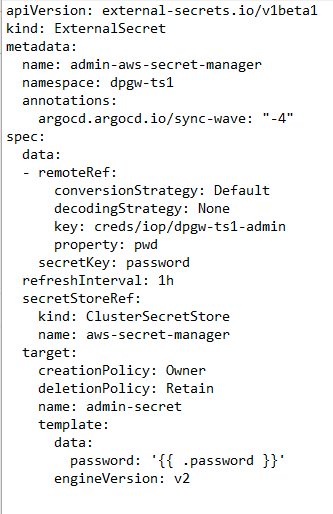
* **Creating ConfigMaps generated by the above tool**

cd backup-output

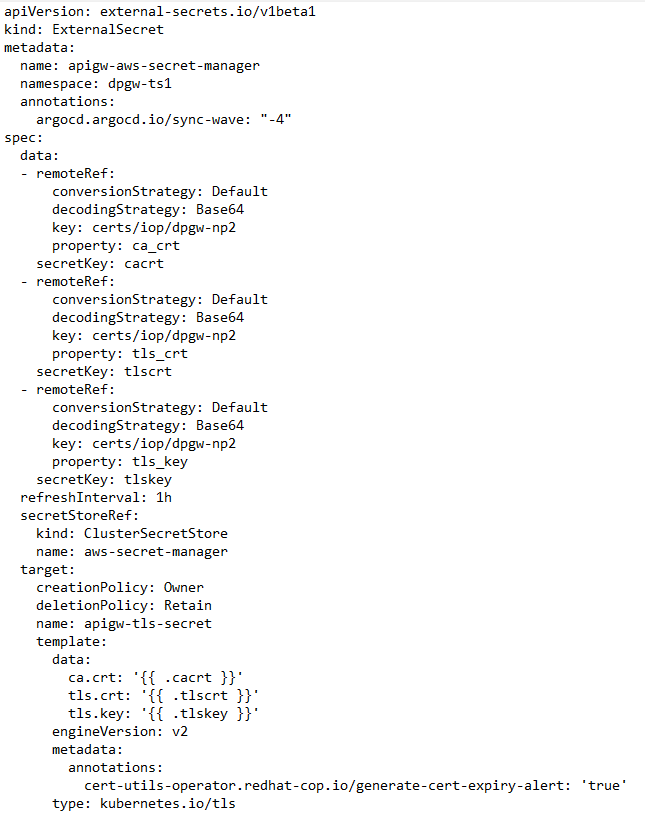
oc apply -f dp-ts1-int-cfg.yaml –n dpgw-ts1

oc apply -f dp-ts1-int-local.yaml –n dpgw-ts1

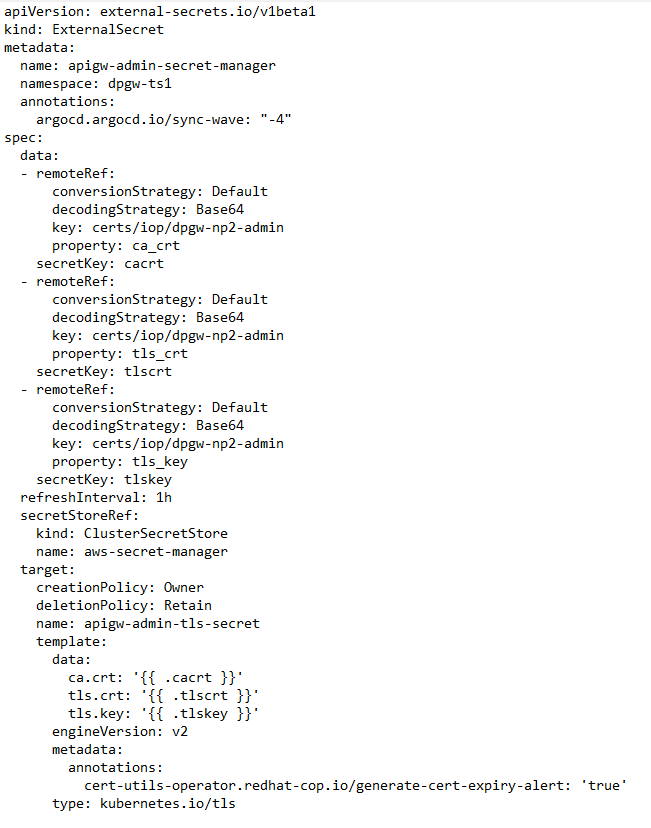
* 1. **DataPower Deployment**
* **Create a password secret**



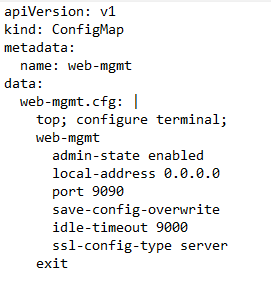
* **Create a TLS key**



* **Create WebUI admin TLS key**



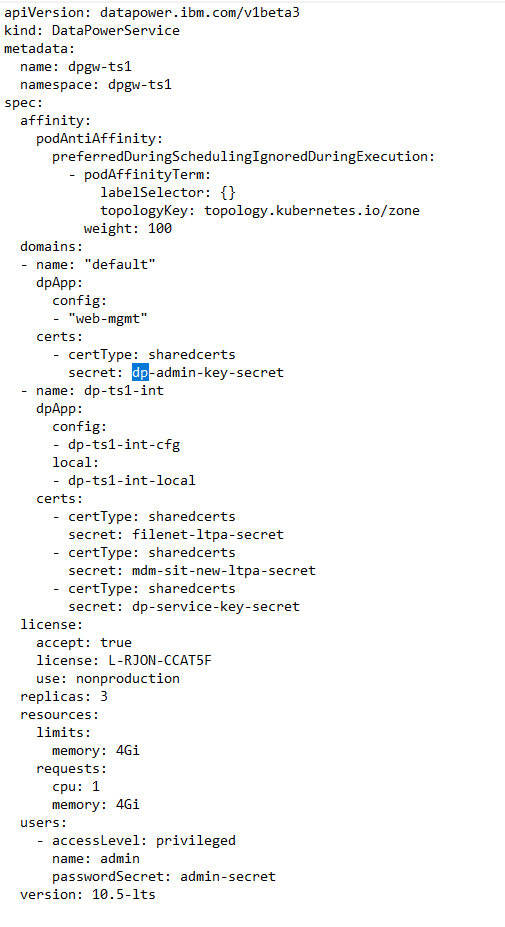
* **Create a web management ConfigMap**



* **Create secrets for the certificates and keys which could not be exported from the old Datapower, eg, DataPower service certificate and key.**

oc create secret generic dp-service-key-secret --from-file=apigw-np1\_key.pem --from-file=apigw-np1.cer

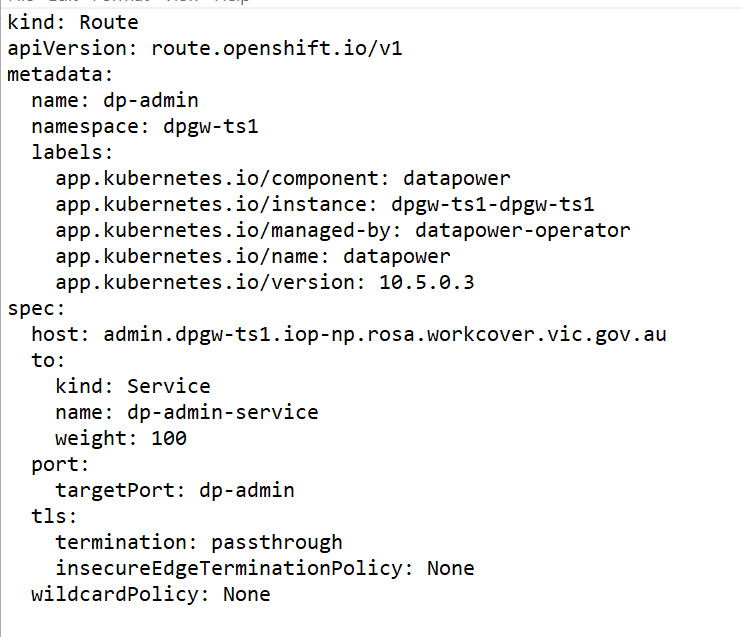
* **Create DataPower Service CR**



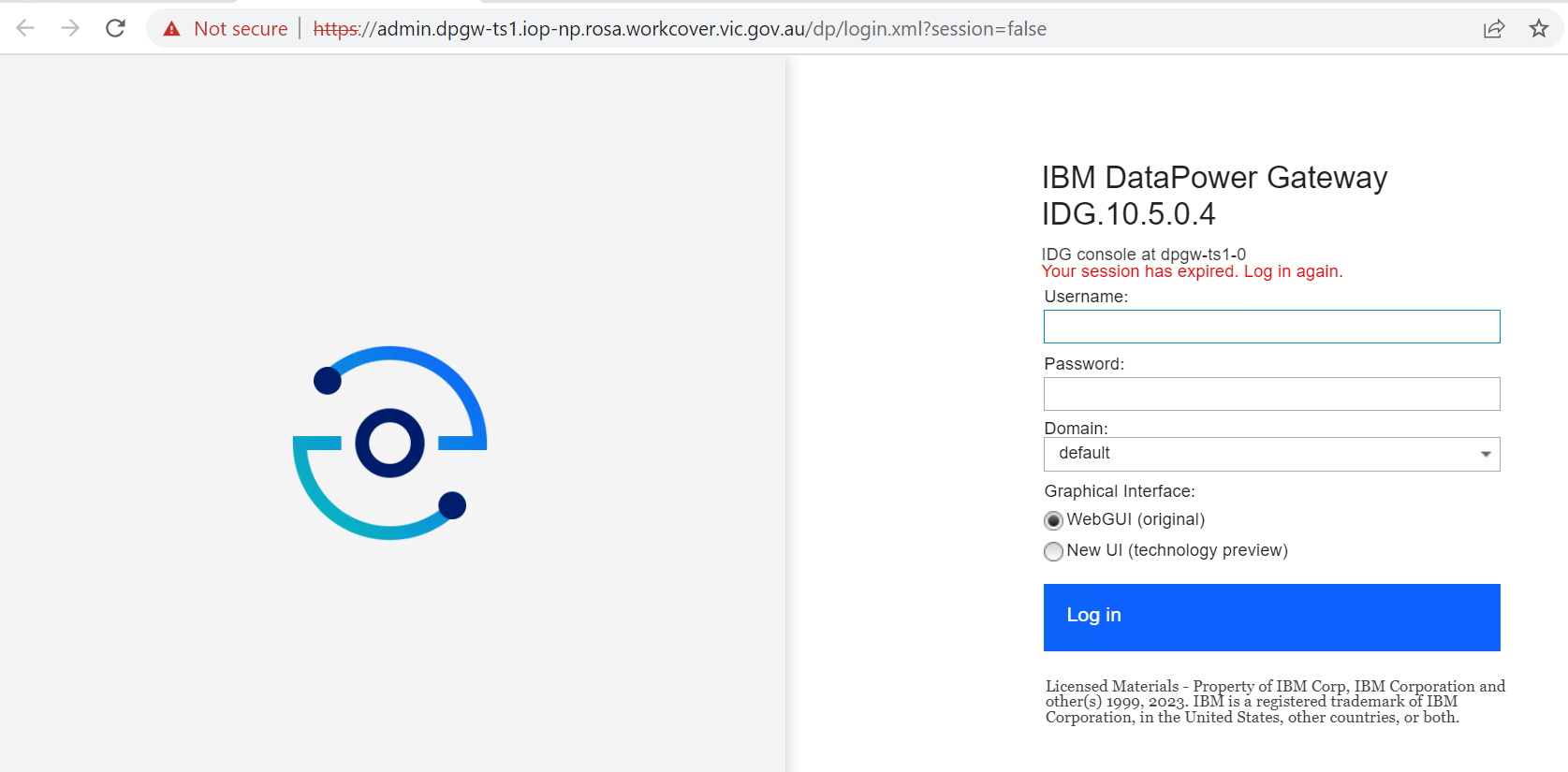
* 1. **DataPower Route Configuration**
* **Create a service yaml file**



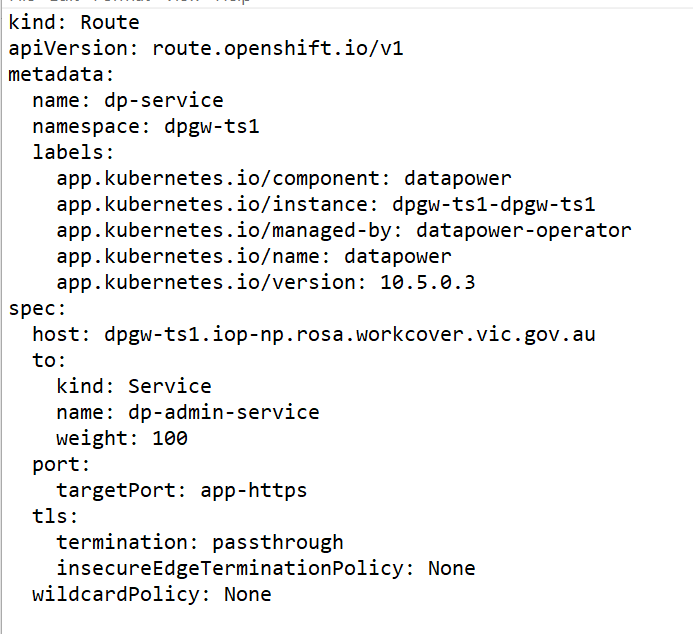
* **Create a route for WebUI**



* **Open the WebUI via** [**https://admin.dpgw-ts1.iop-np.rosa.workcover.vic.gov.au**](https://admin.dpgw-ts1.iop-np.rosa.workcover.vic.gov.au)



* **Create a route for DP service**



# **MQ Deployment**

A MQ custom image is built on ICP for MQ deployment. However on CP4I, IBM MQ image is directly used for MQ deployment without creating a custom image.

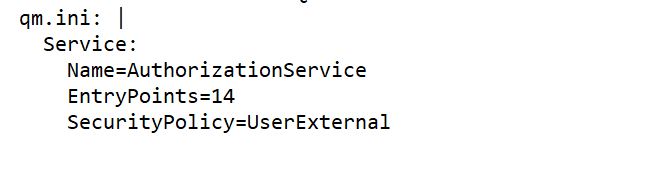
* 1. **Queue Manager Configuration Migration**

Some queue manager configurations should be changed on CP4I before MQ deployment.

* + 1. **Create a qm.ini file**

Local users are created on ICP custom MQ image. But this approach doesn’t work on CP4I because no MQ users are created at the OS level.

A qm.ini is created on CP4I to enable MQ to support users which are not on the local OS or LDAP.



* + 1. **Remove the listener definition**

A default listener will be created automatically on CP4I. Therefore it’s necessary to define a listener on queue manager.

Delete the follow lines in Queue manager configuration:

\*\* CONFIGURING LISTENER STARTS

DEFINE LISTENER('TS1DMA301.LISTENER') TRPTYPE(TCP) PORT(1414) CONTROL(QMGR) REPLACE

START LISTENER('TS1DMA301.LISTENER')

* + 1. **Update Channel configurations**

define channel('ACE.PRE.CLIENT') chltype(SVRCONN) trptype(TCP)

MCAUSER('app') sslcauth(REQUIRED) sslciph(TLS\_RSA\_WITH\_AES\_256\_GCM\_SHA384)

SSLPEER('CN=ACE-TS1 Client,OU=TS1/IBM MQ

Client,O=ITSS,L=Geelong,ST=VIC,C=AU') REPLACE

alter authinfo(SYSTEM.DEFAULT.AUTHINFO.IDPWOS) authtype(IDPWOS)

chckclnt(NONE)

* + 1. **Configure User “app” permission**

set authrec principal('app') objtype(qmgr) authadd(all)

set authrec profile('SC.\*\*') principal('app') objtype(queue) authadd(all)

set authrec profile('CS.\*\*') principal('app') objtype(queue) authadd(all)

set authrec profile('TEST.\*\*') principal('app') objtype(queue) authadd(all)

set authrec profile('TS1TM01.DLQ') principal('app') objtype(queue)

authadd(all)

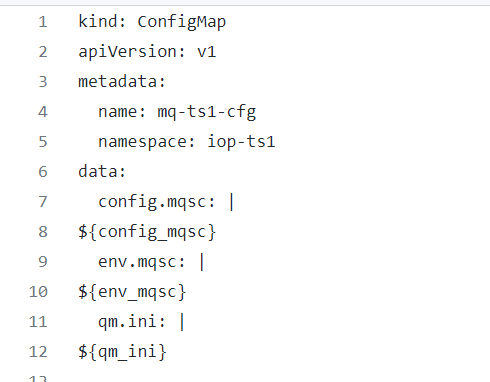
set authrec profile('SYSTEM.SALESFORCE.\*\*') principal('app') objtype(queue)

authadd(all)

set authrec profile('SF.\*\*') principal('app') objtype(topic) authadd(all)

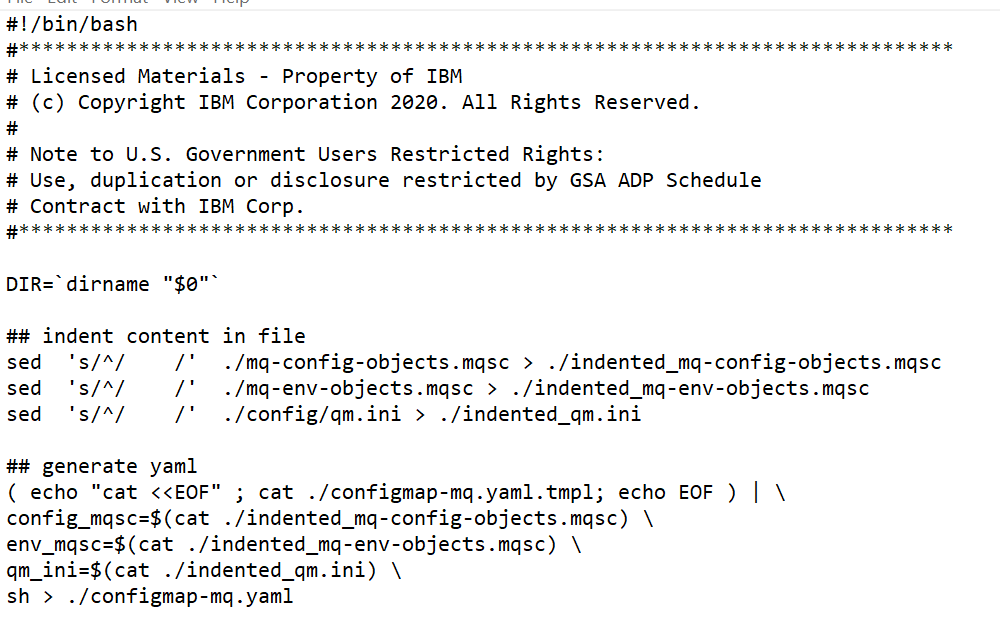
set authrec profile('IOP.\*\*') principal('app') objtype(topic) authadd(all)

* + 1. **Create a ConfigMap for MQ**
* **Create a ConfigMap template file configmap-mq.yaml.tmpl**



* **Create a ConfigMap file via gen-mq.sh**

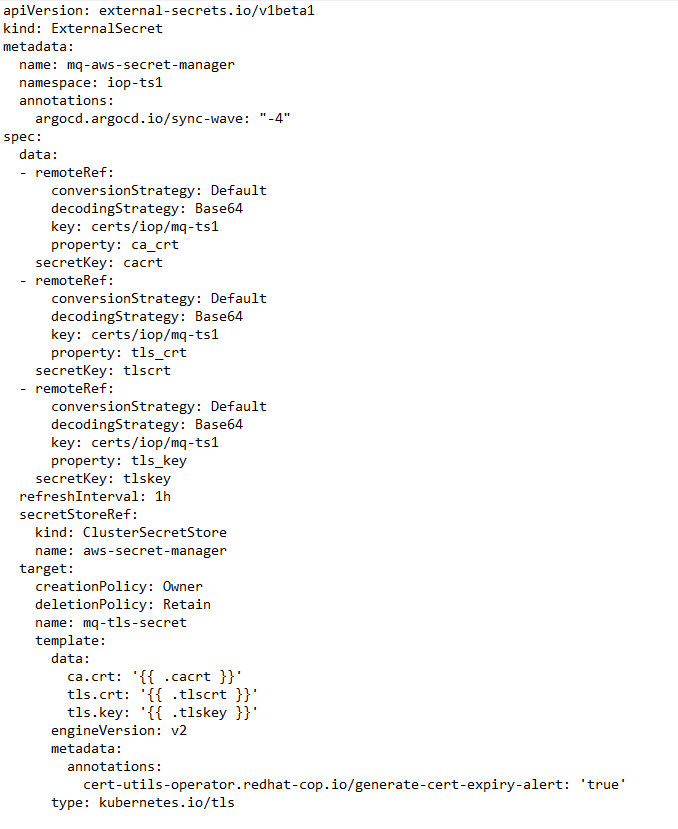
The script gen-mq.sh is shown below and run the script to generate the ConfigMap.



* **Apply the configMap file**

oc apply –f configmap-mq.yaml

* 1. **Queue Manager Deployment**
* **Create a MQ TLS secret**

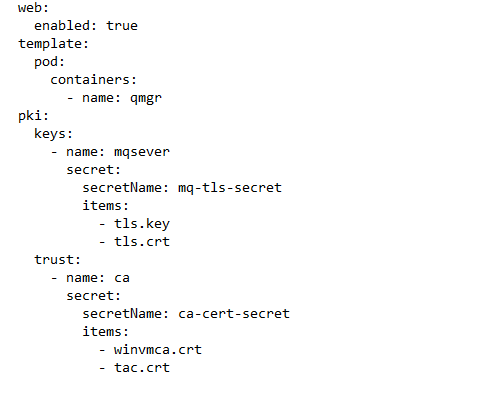


* **Create a MQ CA secret**

oc create secret generic ca-cert-secret --from-file=winvmca.crt --from-file=tac.crt –n iop-ts1

* **Create a MQ CR**





* **Deploy MQ**

oc apply –f mq.yaml

* 1. **Create Queue Manager Route**

kind: Route

apiVersion: route.openshift.io/v1

metadata:

  name: mq01-https

  namespace: iop-ts1

spec:

  host: mq-ts1.iop-np.rosa.workcover.vic.gov.au

  to:

    kind: Service

    name: mq01-ibm-mq

    weight: 100

  port:

    targetPort: qmgr

  tls:

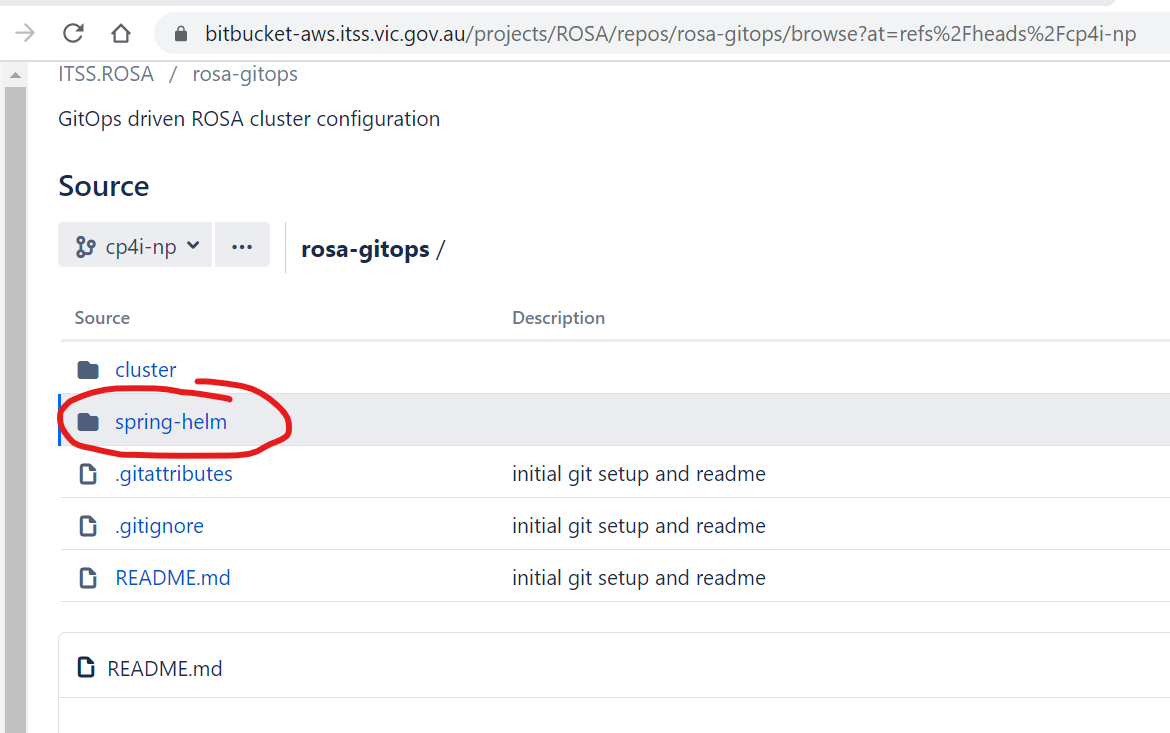
    termination: passthrough

    insecureEdgeTerminationPolicy: Redirect

  wildcardPolicy: None

# **SpringBoot Migration**

* 1. **SpringBoot Helm Chart Migration**
* **Upload Spring-helm directory to ITSS.ROSA/rosa-gitops bitbucket**



* **Clone rosa-springboot on Bastion server**

git clone <https://bitbucket-aws.itss.vic.gov.au/scm/rosa/rosa-springboot.git>

cd rosa-springboot/cicd-spring-01/scripts/csdecipha/ts1

* **Create Configmaps and Secrets**

chmod +x \*.sh

./generateAppConfigmap.sh jsb-dv1-csdecipha-config spring-ts1 /home/ec2-user/app-migration/rosa-app-migration/config-cs-decipha/config/ts1

./generateAppSecret.sh jsb-dv1-csdecipha-wsvsecret spring-ts1

./generatesslftpkeySecret.sh jsb-dv1-csdecipha-sslftpkeysecret spring-ts1 /home/ec2-user/app-migration/rosa-app-migration/config-cs-decipha/config/ts1

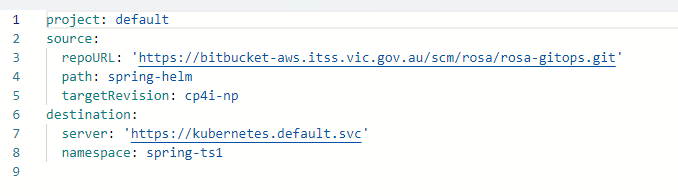
* **Update Helm Chart Deployment Template**

1. Change runAsUser: 1000
2. Remove fsGroup: 1001
3. Remove securityContext. Capabilities

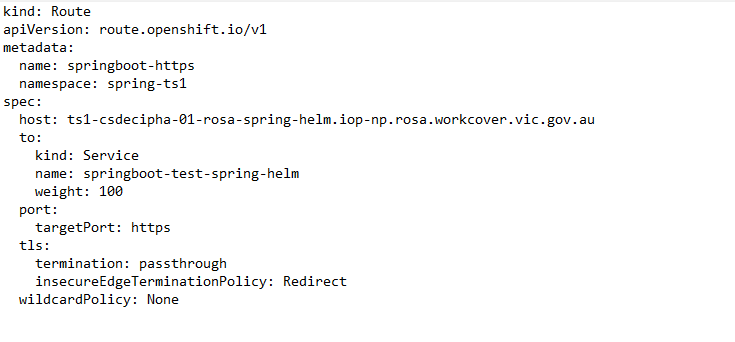
* **Update Helm Chart Value.yaml**

1. Update image. Repository Update image. Repository
2. Update image. Tag
3. Update springAppConfig, springAppSecret and sslftpkeySecret
4. Remove podAnnotations.prometheus.io/scrape
5. Remove nodeSelector.hostgroup and nodeSelector.ace
   1. **SpringBoot Deployment**

* **Create an ArgoCD app and Sync the app.**



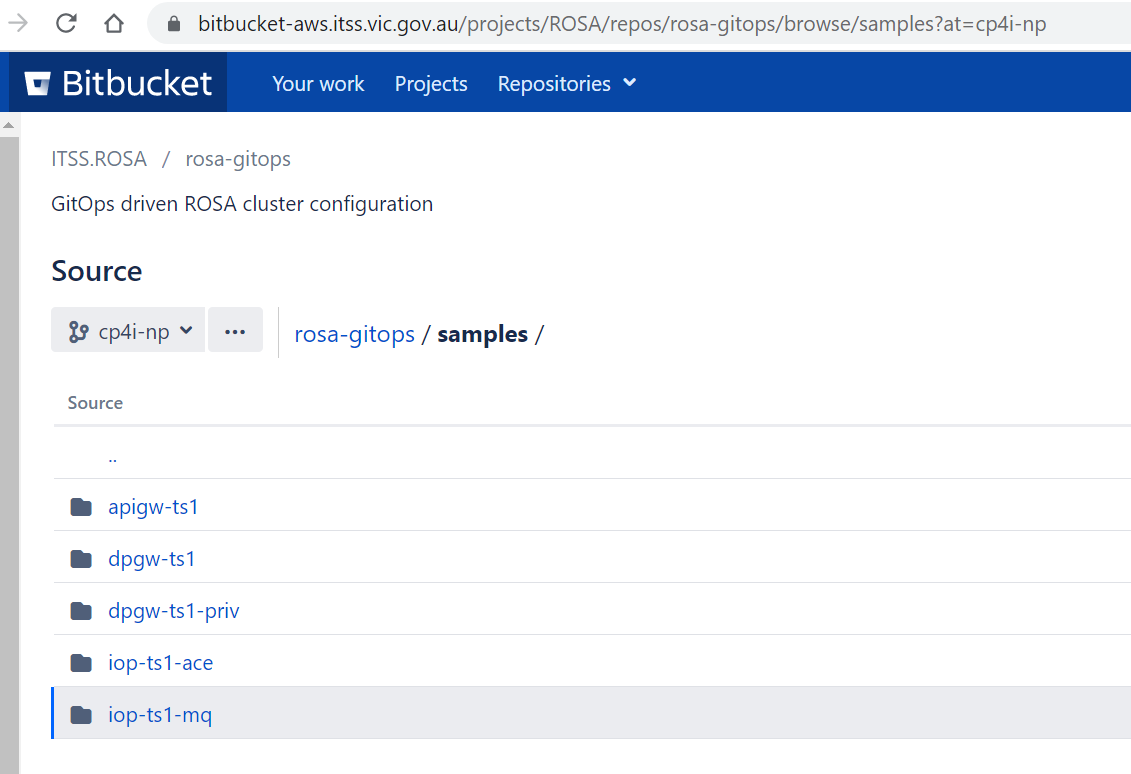
* **Create a route**



# **ArgoCD deployment**

* 1. **Sample deployment repository**

All deployment yaml files are uploaded into Bitbucket: <https://bitbucket-aws.itss.vic.gov.au/projects/ROSA/repos/rosa-gitops/browse/samples?at=refs%2Fheads%2Fcp4i-np>



* 1. **Create Namespaces for application deployment**

The follow namespaces are created by Integral Team:

1. apigw-ts1
2. dpgw-ts1
3. dpgw-ts1-priv
4. iop-ts1
   1. **Application deployment**
      1. **API Gateway apigw-ts1**

* **Create an ArgoCD app and Sync the app.**

project: default

source:

  repoURL: 'https://bitbucket-aws.itss.vic.gov.au/scm/rosa/rosa-gitops.git'

  path: samples/apigw-ts1

  targetRevision: cp4i-np

destination:

  server: 'https://kubernetes.default.svc'

  namespace: apigw-ts1

* + 1. **DataPower Gateway dpgw-ts1**
* **Create an ArgoCD app and Sync the app.**

project: default

source:

  repoURL: 'https://bitbucket-aws.itss.vic.gov.au/scm/rosa/rosa-gitops.git'

  path: samples/dpgw-ts1

  targetRevision: cp4i-np

destination:

  server: 'https://kubernetes.default.svc'

  namespace: dpgw-ts1

* + 1. **DataPower Gateway dpgw-ts1-priv**
* **Create an ArgoCD app and Sync the app.**

project: default

source:

  repoURL: 'https://bitbucket-aws.itss.vic.gov.au/scm/rosa/rosa-gitops.git'

  path: samples/dpgw-ts1-priv

  targetRevision: cp4i-np

destination:

  server: 'https://kubernetes.default.svc'

  namespace: dpgw-ts1-priv

* + 1. **Integration Servers**
* **Create an ArgoCD app and Sync the app.**

project: default

source:

  repoURL: 'https://bitbucket-aws.itss.vic.gov.au/scm/rosa/rosa-gitops.git'

  path: samples/iop-ts1-ace

  targetRevision: cp4i-np

destination:

  server: 'https://kubernetes.default.svc'

  namespace: iop-ts1

* + 1. **MQ**
* **Create an ArgoCD app and Sync the app.**

project: default

source:

  repoURL: 'https://bitbucket-aws.itss.vic.gov.au/scm/rosa/rosa-gitops.git'

  path: samples/iop-ts1-mq

  targetRevision: cp4i-np

destination:

  server: 'https://kubernetes.default.svc'

  namespace: iop-ts1

* + 1. **SpringBoot**
* **Create an ArgoCD app and Sync the app.**

project: default

source:

  repoURL: 'https://bitbucket-aws.itss.vic.gov.au/scm/rosa/rosa-gitops.git'

  path: spring-helm

  targetRevision: cp4i-np

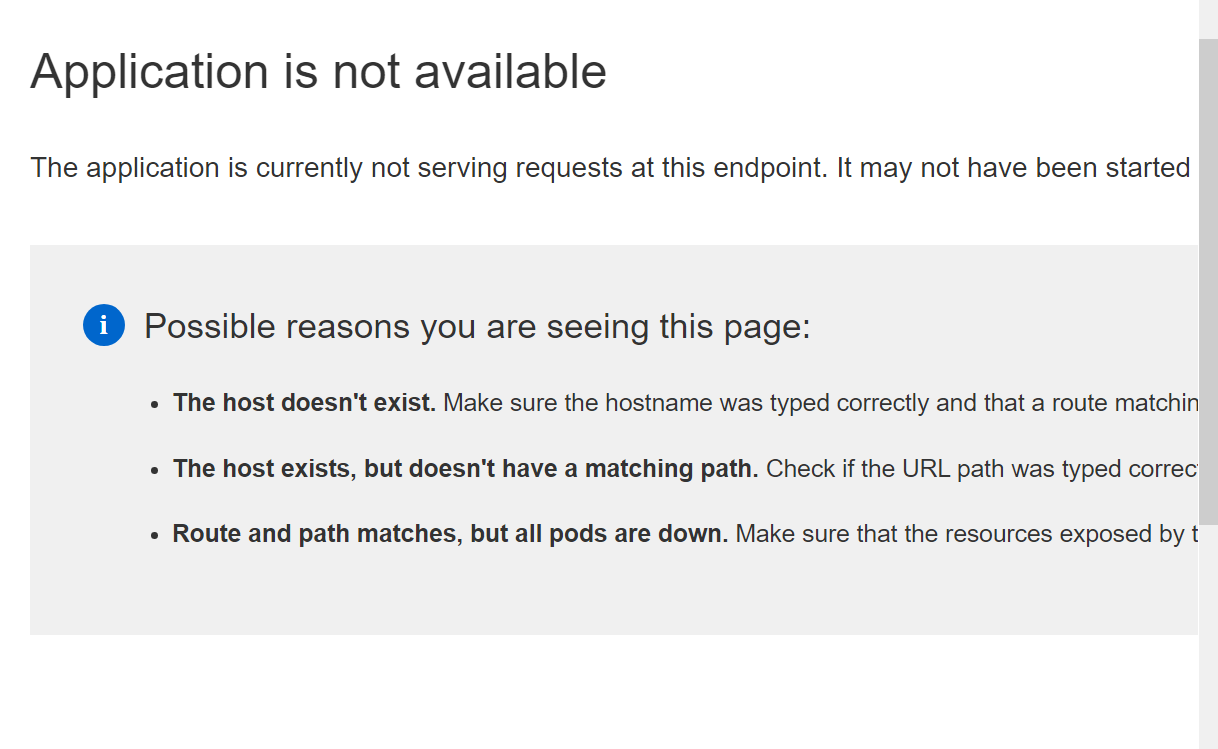
destination:

  server: 'https://kubernetes.default.svc'

  namespace: iop-ts1

# **Troubleshooting**

* 1. **Platform UI is not available**

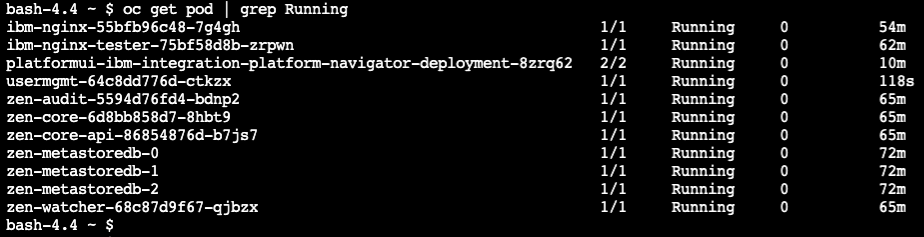


Solution: restart all platformUI pods under Namespace cp4i

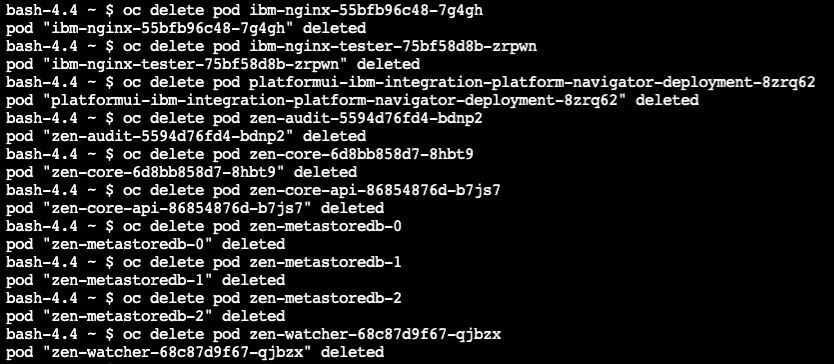
* **Check the pods under Namespace cp4i**

oc project cp4i

oc get pod | grep Running



* **Delete pods**

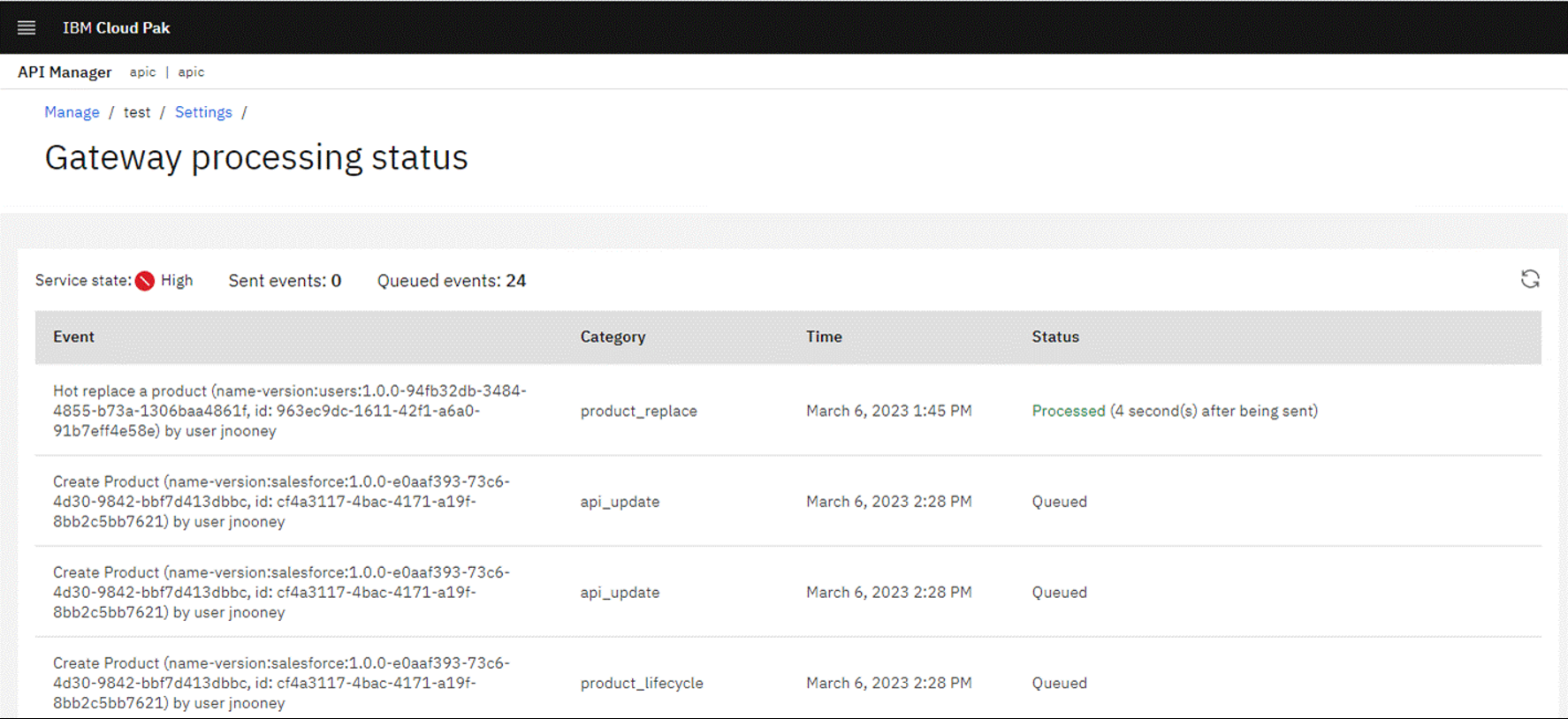


* **Check all pods are ready**

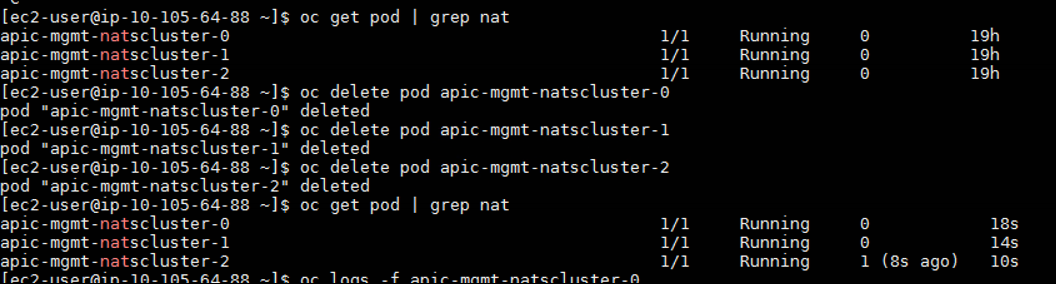
Text

Description automatically generated

* 1. **API Gateway High Queued Status**
* **Check APIGW processing status and found a lot of events are queued.**



* **Solution: restart apic-mgmt-natscluster pods in apic namespace.**



* **Recheck APIGW processing status and make sure all events are processed.**

