OpenShift RH-SSO migration when connecting to an external Oracle Database

Presentation

This paper describes how it is possible to perform the migration from RH-SSO 7.5 to RH-SSO 7.6 when RH-SSO is deployed on OpenShift and connected to an external Oracle 19c database.

There are 2 different possible scenarios allowing you to connect to an external database.
The upgrade from RH-SSO 7.5 to RH-SSO 7.6 will depend on the scenario retained.

The both scenarios are:

Scenario 1 :

- Build a custom RH-SSO 7.5 image that embeds the external database driver.
- Push this new image to an intermediate repository such as *quay.io.*
- Update your deployment config to pick this new image.

Note:
This scenario corresponds to the scenario which is described within RH-SSO documentation on OpenShift: 3.4. Using Custom JDBC Driver.

Scenario 2:

In this scenario, a new custom template *sso75-x509-ojdbc8.json* is created with:

- A BuildConfig using the *docker strategy* to download the Oracle JDBC 19c Driver and pointing to *sso75-openshift-rhel8:7.5* imagestream.
- A configmap (for sso-extensions.cli script).

- Secrets (to retrieve Oracle Database connection parameters)

The main difference between scenario 1 and scenario 2 is:
- in scenario 1, you build and deploy a custom image corresponding to a precise RH-SSO version.
- In scenario 2, a new custom RH-SSO image is built and redeployed upon each RH-SSO pod restart, whenever a new RH-SSO image is available.

Scenario 1 - Building a custom RH-SSO 7.5 image

The goal of this section is to describe how it is possible to deploy a custom RH-SSO 7.5 image on Openshift with RH-SSO connecting to an external Oracle database.

Deploying RH-SSO 7.5 on Openshift

You first need to connect to Your Openshift cluster as an administrator, having cluster-admin privileges.

The way to download RH-SSO 7.5 templates.is described in RH-SSO documentation at:
2.1. Using the Red Hat Single Sign-On for OpenShift Image Streams and application templates.

As Openshift administrative cluster user, run the following commands to update the core set of Red Hat Single Sign-On 7.5.2 resources for OpenShift in the `openshift` project:

```
$ for resource in sso75-image-stream.json \
  sso75-https.json \
  sso75-postgresql.json \
  sso75-postgresql-persistent.json \
  sso75-x509-https.json \
  sso75-x509-postgresql-persistent.json
do
  oc replace -n openshift --force -f \

https://raw.githubusercontent.com/jboss-container-images/redhat-sso-7-openshift-image/sso75-dev/templates/${resource};

done
```

It is possible that the execution of this command might fail.

This described throughout 2 different JIRAs:

- RHSSO-2168: Command to download templates on Openshift RH-SSO 7.5 templates is failing.
- **Bug 2068138 -** Timed out waiting for the condition while replacing the template

Despite RH #**2068138**, it is possible to download manually the only template we are interested in:

```
oc replace -f
https://raw.githubusercontent.com/jboss-container-images/redhat-sso-7-openshift-image/sso75-dev/templates/sso75-ocp4-x509-https.json -n openshift
template.template.openshift.io/sso75-ocp4-x509-https replaced
```

Install RH-SSO OpenShift image streams in the `openshift` project:

```
$ oc -n openshift import-image rh-sso-7/sso75-openshift-rhel8:7.5
--from=registry.redhat.io/rh-sso-7/sso75-openshift-rhel8:7.5 --confirm
```

Create a sso-75-external-db project:

```
$ oc new-project sso-75-external-db
```

Deploy the original/old RH-SSO 7.5 container image using the "sso75-ocp4-x509-https" OpenShift template. We use the *oc process* command to provide *user admin credentials* which are *admin/password.*

```
$ oc process sso75-ocp4-x509-https SSO_ADMIN_USERNAME=admin
SSO_ADMIN_PASSWORD=password -n openshift -o yaml >
sso75-ocp4-x509-https.yaml

$ oc create -f sso75-ocp4-x509-https.yaml
```

Testing

Once SSO pod has reached ready state 1/1, we can use the admin console to connect to it.

```
oc get pods
NAME          READY   STATUS      RESTARTS   AGE
sso-1-6bz9x    1/1     Running     0          31s
sso-1-deploy   0/1     Completed   0          8m21s
```

To get the admin console URL, you can either run the command *oc get routes*, or *oc status*.

```
$ oc status
In project sso-75-external-db on server https://api.example.redhat.com:6443

svc/sso-ping (headless):8888
https://sso-sso-75-external-db.apps.example.redhat.com (reencrypt) (svc/sso)
  dc/sso deploys openshift/sso75-openshift-rhel8:7.5
    deployment #1 deployed 6 minutes ago - 1 pod
```

You should now be able to connect to the admin console URL with admin/password

If the connection is failing, see Tip-3: Application is not available - Admin console.

Building a sso-75 image with Oracle Driver and push it to quay.io

You need to build an image connecting to an external database as described in RH-SSO documentation:
3.4. Using Custom JDBC Driver

Create a new directory called build-image

```
mkdir build-image

cd build-image
```

Download into this directory the Oracle JDBC driver odjbc8.jar driver

```
wget https://download.oracle.com/otn-pub/otn_software/jdbc/1916/ojdbc8.jar
```

Check *sso-extensions.cli* script
Update the variables *DB_JDBC_URL, DB_USERNAME, DB_PASSWORD*

```
Update the variables DB_JDBC_URL, DB_USERNAME, DB_PASSWORD
set DB_JDBC_URL=<existing-external-oracle-jdbc-url>
set DB_USERNAME=<oracle-db-username>
set DB_PASSWORD=<oracle-db-password>
...
set FILE=/opt/eap/extensions/ojdbc8.jar
set DB_DRIVER_NAME=oracle
....
.....
....
```

Note:
You may find most of the *sso-extensions.cli* skeleton in the appendix.

Create a Dockerfile that points to the sso-75 image.

```
FROM rh-sso-7/sso75-openshift-rhel8:latest

COPY sso-extensions.cli /opt/eap/extensions/

COPY ojdbc8.jar /opt/eap/extensions/ojdbc8.jar
```

Build the image

```
$ podman build -t localhost/docker-registry-default/project/sso75-external-db-oracle:1.0 .

STEP 1/3: FROM rh-sso-7/sso75-openshift-rhel8:latest
STEP 2/3: COPY sso-extensions.cli /opt/eap/extensions/
--> a619499e63d
STEP 3/3: COPY ojdbc8.jar /opt/eap/extensions/ojdbc8.jar
COMMIT localhost/docker-registry-default/project/sso75-external-db-oracle:1.0
--> b6cda72079d
Successfully tagged localhost/docker-registry-default/project/sso75-external-db-oracle:1.0
```

Push image to quay.io:

```
podman login -u <username> -p <password> quay.io
Login Succeeded!

podman push localhost/docker-registry-default/project/sso75-external-db-oracle:1.0
quay.io/<username>/sso:sso_75_1_30_08

Copying blob 6fb638e16655 done
Copying blob 45b6ae52fd47 done
Copying blob 2613a7d0a222 skipped: already exists
Copying blob e7a656684252 skipped: already exists
Copying blob 725111d3ca9a skipped: already exists
Copying config b6cda72079 done
Writing manifest to image destination
Storing signatures
```

Deploy sso-75 image connecting to the Oracle database

Ensure that you are using sso-75-external-db namespace, and check that the pod is running:

```
oc project sso-75-external-db
Using project "sso-75-external-db" on server "https://api.example.redhat.com:6443".

$ oc get pods
NAME          READY   STATUS      RESTARTS   AGE
sso-1-6bz9x    1/1     Running     0          77m
sso-1-deploy   0/1     Completed   0          85m
```

Scale down the replica to zero.

```
oc scale --replicas=0 dc/sso
```

Update the sso deployment with the three following modifications using the *oc edit command:*

- Update image field to point to the sso-75 image build previously.
- Totally remove the *triggers* and *update* sections from this yaml file edition.
- Add initialDelaySeconds: 600 to liveness and readiness probes.

```
$ oc edit dc/sso

image: quay.io/<username>/sso:sso_75_1_30_08
….
….
livenessProbe:
        exec:
          command:
          - /bin/bash
          - -c
          - /opt/eap/bin/livenessProbe.sh
        failureThreshold: 3
        initialDelaySeconds: 600
        periodSeconds: 10
        successThreshold: 1
        timeoutSeconds: 1
….
….


readinessProbe:
        exec:
          command:
          - /bin/bash
          - -c
          - /opt/eap/bin/readinessProbe.sh
        failureThreshold: 3
        initialDelaySeconds: 600
        periodSeconds: 10
        successThreshold: 1
        timeoutSeconds: 1
```

Totally remove both the trigger and status sections. Or remove both the trigger na status sections altogether.

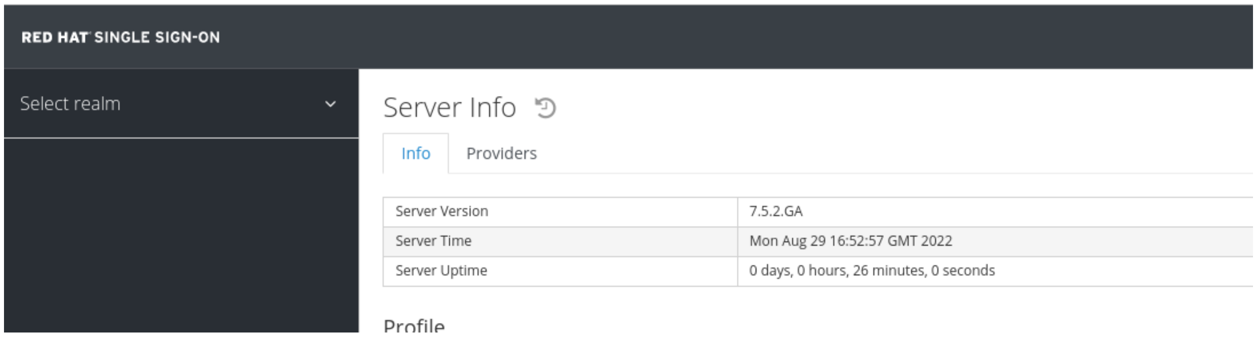
Testing

Check that sso pod trace shows that you connect to the external Oracle database

```
$ oc logs -f <sso-pod-name>

16:26:47,494 INFO  [org.jboss.as] (MSC service thread 1-1) WFLYSRV0050: WildFly Core
15.0.8.Final-redhat-00001 stopped in 52ms
INFO Running rh-sso-7/sso75-openshift-rhel8 image, version 7.5

….
….
16:26:59,152 INFO  [org.keycloak.connections.jpa.DefaultJpaConnectionProviderFactory]
(ServerService Thread Pool -- 87) Database info:
{databaseUrl=<external-oracle-url-db>:1521:dballo, databaseUser=<oracle-db-username>,
databaseProduct=Oracle Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 -
Production
Version 19.3.0.0.0, databaseDriver=Oracle JDBC driver 19.16.0.0.0}
16:27:00,045 INFO  [org.hibernate.jpa.internal.util.LogHelper] (ServerService Thread Pool --
87) HHH000204: Processing PersistenceUnitInfo [
        name: keycloak-default
        ...]
…..
…..
```

```
oc get pods
NAME          READY   STATUS      RESTARTS   AGE
sso-1-deploy   0/1     Completed   0          115m
sso-2-deploy   0/1     Completed   0          18m
sso-3-6h8rn    1/1     Running     0          10m
sso-3-deploy   0/1     Completed   0          13m
$ oc status
In project sso-75-external-db on server https://api.example.redhat.com:6443

svc/sso-ping (headless):8888
https://sso-sso-75-external-db.apps.exampleredhat.com (reencrypt) (svc/sso)
  dc/sso deploys quay.io/orivat/sso:sso_75_1_30_08
    deployment #3 deployed 11 minutes ago - 1 pod
    deployment #2 deployed 16 minutes ago
    deployment #1 deployed 2 hours ago
```

Now you shall be able to connect to the admin console. When displaying serverinfo, it indicates that 7.5.2 is in use
(serververinfo is available from Admin menu, top left hand corner)



Server Version	7.5.2.GA
Server Time	Mon Aug 29 16:52:57 GMT 2022
Server Uptime	0 days, 0 hours, 26 minutes, 0 seconds

Scenario 1 - Migrating to RH-SSO 7.6

The usual migration process is described in detail in section 4.1. Updating a database for a new Red Hat Single Sign-On for OpenShift image version, where you have the choice between **Automatic and manual Migration**.

The approach that we will take here is kind of similar to Automatic *migration* but we will adapt it to meet our needs.

In our case, we will adapt the migration process, and need to do the following steps:

- Build an external RH-SSO 7.6 image able to connect to external Oracle Database
- Stop all Pods running SSO 7.5 version (`$ oc scale --replicas=0 dc/sso)`
- `Update the SSO deployment config to points to 7.6 image`
- `Restart SSO pod ($ oc scale --replicas=1 dc/sso)`

Building a sso76 image

Create a Docker_76 file, which points to sso76 release.

```
FROM rh-sso-7/sso76-openshift-rhel8:latest
```

```
COPY sso-extensions.cli /opt/eap/extensions/

COPY ojdbc8.jar /opt/eap/extensions/ojdbc8.jar
```

Build the new image

```
podman build -f Dockerfile_76 -t
localhost/docker-registry-default/project/sso76-external-db-oracle:1.0 .
STEP 1/3: FROM rh-sso-7/sso76-openshift-rhel8:latest
STEP 2/3: COPY sso-extensions.cli /opt/eap/extensions/
--> 6cb7a1d0135
STEP 3/3: COPY ojdbc8.jar /opt/eap/extensions/ojdbc8.jar
COMMIT localhost/docker-registry-default/project/sso76-external-db-oracle:1.0
--> 7451dbb59d9
Successfully tagged localhost/docker-registry-default/project/sso76-external-db-oracle:1.0
7451dbb59d9d9fe89a104ebe2def1bdbdd0154fa5a024c84bb2be2b80d7a7396
```

Push the image sso76 to quay.io:

```
podman push localhost/docker-registry-default/project/sso76-external-db-oracle:1.0
quay.io/<username>/sso:sso_76_1_30_08
Getting image source signatures
Copying blob 8f1ae4f45dd0 done
Copying blob ecf2e2fc44ff done
Copying blob 725111d3ca9a skipped: already exists
Copying blob f682c28ad812 skipped: already exists
Copying blob e7a656684252 skipped: already exists
Copying config 7451dbb59d done
Writing manifest to image destination
Storing signatures
```

Deploying the new sso-76 image

Stop the SSO pod:

```
oc scale --replicas=0 dc/sso
```

Update the deployment config to point to the sso76 image

```
$ oc edit dc/sso
```

```
image: quay.io/<username>/sso:sso_76_1_30_08
….
….
```

Restart the SSO

```
oc scale --replicas=1 dc/sso
```

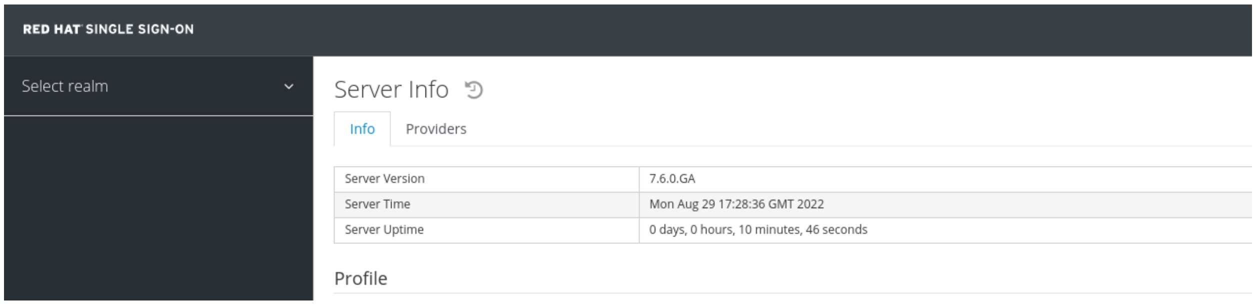
Testing - checking sso log file

The log trace displays that you are using sso76 image.

```
17:11:33,311 INFO  [org.jboss.as] (MSC service thread 1-1) WFLYSRV0050: WildFly Core
15.0.8.Final-redhat-00001 stopped in 25ms
INFO Running rh-sso-7/sso76-openshift-rhel8 image, version 7.6

….
….

17:11:44,336 INFO  [org.keycloak.connections.jpa.DefaultJpaConnectionProviderFactory]
(ServerService Thread Pool -- 77) Database info:
{databaseUrl=jdbc:<<external-oracle-url-db>:1521:dballo,
databaseUser=<oracle-db-username>, databaseProduct=Oracle Oracle Database 19c
Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0, databaseDriver=Oracle JDBC driver 19.16.0.0.0}
17:11:45,312 INFO  [org.hibernate.jpa.internal.util.LogHelper] (ServerService Thread Pool --
77) HHH000204: Processing PersistenceUnitInfo
```

Testing - checking admin console

Connecting to the admin console URL, You can check that server info displays 7.6 information.



Server Version	7.6.0.GA
Server Time	Mon Aug 29 17:28:36 GMT 2022
Server Uptime	0 days, 0 hours, 10 minutes, 46 seconds

Scenario 2: Deploying RH-SSO 7.5 using a Buildconfig

Advantage of using a buildconfig for RH-SSO 7.5

The advantage of using a *BuildConfig* with the *docker strategy* versus a static image build (scenario 1) is that upon each RH-SSO instance restart, the buildconfig (bc) will check if the imageStream *sso75-openshift-rhel8:7.5* has been updated with a new RH-SSO image.

Preparing sso75-x509-ojdbc8.json template and sso-extensions.cli environment

A new template *sso75-x509-ojdbc8.json* is created. It is derived from the *sso75-x509-https.json* template.

Below are described in detail the different steps:.

Step 1: Create a json template file

```
oc process sso75-x509-https SSO_ADMIN_USERNAME=admin
SSO_ADMIN_PASSWORD=password -n openshift -o json > sso75-x509-ojdbc8.json
```

Step 2: Update *sso-extensions.cli script* with *Oracle DB information* (Oracle Database connection URL, Oracle Database username, oracle Database password)

```
batch

set DB_JDBC_URL=<external-oracle-url-db>
set DB_USERNAME=<oracle-db-username>
set DB_PASSWORD=<oracle-db-password>

set FILE=/home/jboss/ojdbc8.jar

set DB_DRIVER_NAME=oracle


module add --name=com.oracle --resources=$FILE
--dependencies=javax.api,javax.resource.api


/subsystem=datasources/jdbc-driver=$DB_DRIVER_NAME:add( \
  driver-name=$DB_DRIVER_NAME, \
  driver-module-name=com.oracle, \
  driver-class-name=oracle.jdbc.driver.OracleDriver, \
  driver-xa-datasource-class-name=oracle.jdbc.xa.client.OracleXADataSource \
)

/subsystem=datasources/data-source=KeycloakDS:remove()

/subsystem=datasources/data-source=KeycloakDS:add( \
  jndi-name=java:jboss/datasources/KeycloakDS, \
  enabled=true, \
  use-java-context=true, \
  connection-url=$DB_JDBC_URL, \
  driver-name=$DB_DRIVER_NAME, \
  user-name=$DB_USERNAME, \
  password=$DB_PASSWORD \
)




/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=background-validati
on, value=true)


/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=background-validati
on-millis, value=30000)
```

```
 /subsystem=datasources/data-source=KeycloakDS:write-attribute(name=validate-on-match,
value=false)


/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=check-valid-connecti
on-sql, value=select 1)



/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=valid-connection-che
cker-class-name,
value=org.jboss.jca.adapters.jdbc.extensions.oracle.OracleValidConnectionChecker)


/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=stale-connection-ch
ecker-class-name,
value=org.jboss.jca.adapters.jdbc.extensions.oracle.OracleStaleConnectionChecker)


/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=exception-sorter-cla
ss-name, value=org.jboss.jca.adapters.jdbc.extensions.oracle.OracleExceptionSorter)


/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=min-pool-size,
value=30)

/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=max-pool-size,
value=30)

/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=prepared-statement
s-cache-size, value=100)


/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=flush-strategy,
value=IdleConnections)


run-batch
```

Step 3: Create a configmap from the *sso-extensions.cli* script

```
oc create cm ssoextensionscli --from-file=sso-extensions.cli
```

Step 4: Create an *oracle-db-secret* secret containing the values of Oracle Database connection parameters

```
oc create secret generic oracle-db-secret
--from-literal=EXTERNAL_DB_JDBC_URL=<external-oracle-url-db>
--from-literal=EXTERNAL_DB_USERNAME=<oracle-db-username>
--from-literal=EXTERNAL_DB_PASSWORD=<oracle-db-password>
```

Updating the sso75-x509-ojdbc8.json template

The *sso75-x509-ojdbc8.json* template is updated with:

- A BuildConfig (to import the Oracle 19c JDBC driver).
- A configmap (for sso-extensions.cli script).
- Secrets (to retrieve the Oracle Database connection parameter).

Preparing the buildconfig

The added buildconfig contains a new imageStream called sso-oracle-sso75-ojdbc8:latest

If there is a new SSO image available for *sso75-openshift-rhel8:7.5* imagestream, the Buildconfig source docker strategy will build a new image image called *sso-oracle-sso75-ojdbc8:latest*.

The buildconfig section to be added is:

```
{
            "apiVersion": "build.openshift.io/v1",
            "kind": "BuildConfig",
            "metadata": {
                "labels": {
                    "name": "sso-oracle-sso75-ojdbc8-build"
                },
                "name": "sso-oracle-sso75-ojdbc8-build"
                },
            "spec": {
                "failedBuildsHistoryLimit": 5,
            "nodeSelector": null,
            "output": {
            "to": {
               "kind": "ImageStreamTag",
                  "name": "sso-oracle-sso75-ojdbc8:latest"
               }
             },
            "postCommit": {},
            "resources": {},
            "runPolicy": "Serial",
            "source": {
               "dockerfile": "FROM openshift/sso75-openshift-rhel8:7.5\n\n RUN curl -k -L
https://download.oracle.com/otn-pub/otn_software/jdbc/1916/ojdbc8.jar -o
/home/jboss/ojdbc8.jar \n\nCMD [\"/opt/eap/bin/openshift-launch.sh\"]",
               "type": "Dockerfile"
            },
            "strategy": {
               "dockerStrategy": {
                  "from": {
                     "kind": "ImageStreamTag",
                     "name": "sso75-openshift-rhel8:7.5",
                     "namespace": "openshift"
                     }
                  },
               "type": "Docker"
            },
            "successfulBuildsHistoryLimit": 2,
            "triggers": [
               {
                  "imageChange": {},
                  "type": "ImageChange"
               },
               {
               "type": "ConfigChange"
               }
            ]
         }
      },
```

adding the configmap

You should add the configmap as a mounted volume, so the *sso-extensions.cli* script can be executed at boot time.

```
"volumeMounts": [
            ...
            ...
            {
                   "mountPath": "/opt/eap/extensions",
                   "name": "ssoextensionscli-configmap",
                   "readOnly": true
            }
        ]


 "volumes": [
     ...
     ...
     {
         "name": "ssoextensionscli-configmap",
             "configMap": {
             "name": "ssoextensionscli"
             }
     }
 ]
```

Adding oracle-db-secret

You should add those 3 oracle-db-secret to your json file, so RH-SSO deployment will be able to connect to the Oracle database.

```
{
                    "name": "EXTERNAL_DB_JDBC_URL",
                    "valueFrom": {
                        "secretKeyRef": {
                        "name": "oracle-db-secret",
                        "key":  "EXTERNAL_DB_JDBC_URL"
                        }
                    }
                },
                {
                    "name": "EXTERNAL_DB_USERNAME",
                    "valueFrom": {
                        "secretKeyRef": {
                        "name": "oracle-db-secret",
                        "key": "EXTERNAL_DB_USERNAME"
                        }
                    }
                },
                {
                    "name": "EXTERNAL_DB_PASSWORD",
                        "valueFrom": {
                        "secretKeyRef": {
                            "name": "oracle-db-secret",
                        "key": "EXTERNAL_DB_PASSWORD"
                        }
                    }
                }
            ],
```

Deploy the Buildconfig image

Check that *sso75-x509-ojdbc8.json* template syntax is correct.

To deploy and image, run command

```
oc create -f sso75-x509-https.json
```

Deployment and testing

A specific ephemeral pod is created at runtime to run the buildconfig and create the new image (*sso-oracle-sso75-ojdbc8-build-1-build here)*

```
oc get pods

NAME                              READY   STATUS      RESTARTS   AGE
```

```
sso-oracle-1-79m45                  1/1     Running     0         4d3h
sso-oracle-1-deploy                 0/1     Completed   0         4d3h
sso-oracle-sso75-ojdbc8-build-1-build   0/1     Completed   0         4d3h
```

The command *oc status* provides the RH-SSO admin console URL to connect to.

Scenario 2: Migration to RH-SSO 7.6

When using scenario 2 (Using a buildconfig with a docker to embed the oracle driver), the upgrade has to be done in 3 steps:

- scale down the replica to zero.
- Update the buildconfig, so it will pick the sso-76 image instead of the sso-75 image.
- up scale the replica to one.

Below are described the different steps to be performed for the the upgrade process

Scale down you sso replica

```
oc get dc

oc scale --replicas=0 dc/<sso-dc-name>
```

Example:

```
oc get dc
NAME         REVISION   DESIRED   CURRENT   TRIGGERED BY
sso-oracle   2          1         1         config,image(sso-oracle-sso75-ojdbc8:latest)
```

```
oc scale --replicas=0 dc/sso-oracle
```

Update the buildconfig

You need to update the buildconfig so that the source docker is picking its image from **openshift/sso76-openshift-rhel8:7.6 image** stream **openshift/sso75-openshift-rhel8:7.5 image** stream**.**

```
oc get bc
NAME                          TYPE     FROM         LATEST
sso-oracle-sso75-ojdbc8-build   Docker   Dockerfile   2
```

```
oc edit bc/sso-oracle-sso75-ojdbc8-build

source:
   dockerfile: "FROM openshift/sso76-openshift-rhel8:7.6\n\n RUN curl -k -L
https://download.oracle.com/otn-pub/otn_software/jdbc/1916/ojdbc8.jar
    -o /home/jboss/ojdbc8.jar \n\nCMD [\"/opt/eap/bin/openshift-launch.sh\"]"
   type: Dockerfile
  strategy:
   dockerStrategy:
    from:
     kind: ImageStreamTag
     name: sso76-openshift-rhel8:7.6
     namespace: openshift
   type: Docker
```

As soon as the buildconfig is updated, a new build is automatically triggered.

```
oc get pods
```

```
NAME                          READY   STATUS      RESTARTS   AGE
sso-oracle-1-deploy               0/1     Completed   0         44m
sso-oracle-2-deploy               0/1     Completed   0         13m
sso-oracle-sso75-ojdbc8-build-1-build   0/1     Completed   0         45m
sso-oracle-sso75-ojdbc8-build-2-build   0/1     Completed   0         13m
```

Restart the pod

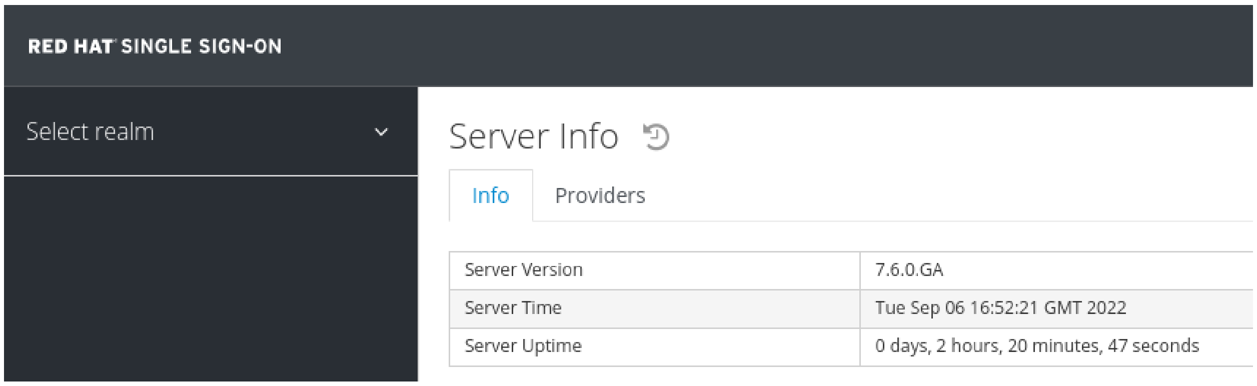
```
oc scale --replicas=1 dc/sso-oracle
```

Testing

When connecting to the admin console, you will see that the admin console is showing 7.6 server info.

You can also check the logs, and verify the schema checking that it is corresponding to RH-SSO 7.6

When connecting to the admin console, you can verify that the admin console is showing 7.6 server info.



Server Version	7.6.0.GA
Server Time	Tue Sep 06 16:52:21 GMT 2022
Server Uptime	0 days, 2 hours, 20 minutes, 47 seconds

Troubleshooting section

Tip-1 Json file edition

Syntax error detection

Using the following command allows to quickly get invalid syntax detection messages

```
cat <filename>.json | jq

--> it will display any syntax error.
```

Using a json editor

Using a json editor,such as VisualCode Studio allows a pretty nice json indenting, and json error detection/display as well.

Tip-2: Getting events ordered by timestamp

```
oc get events --sort-by='.lastTimestamp'
```

Tip-3: Application is not available - Admin console

You may get following screenshot:

```


Application is not available
```

The application is currently not serving requests at this endpoint. It may not have been started or is still starting.

Possible reasons you are seeing this page:

- **The host doesn't exist.** Make sure the hostname was typed correctly and that a route matching this hostname exists.
- **The host exists, but doesn't have a matching path.** Check if the URL path was typed correctly and that the route was created using the desired path.
- **Route and path matches, but all pods are down.** Make sure that the resources exposed by this route (pods, services, deployment configs, etc) have at least one pod running.

To overcome this issue, you have to restart the RH-SSO application.

```
oc scale --replicas=0 dc/sso
```

```
oc scale --replicas=1 dc/sso
```

Appendix

sso-extensions.cli

To update the **sso-extensions.cli** script, you just need to replace the 3 followings variables:

- DB_JDBC_URL
- DB_USERNAME
- set DB_PASSWORD

Check also that ALL the specific CLI oracle instructions match what you need.

```
batch
```

```
set DB_JDBC_URL=<external-oracle-url-db>
set DB_USERNAME=<oracle-db-username>
set DB_PASSWORD=<oracle-db-password>

set FILE=/home/jboss/ojdbc8.jar

set DB_DRIVER_NAME=oracle


module add --name=com.oracle --resources=$FILE
--dependencies=javax.api,javax.resource.api


/subsystem=datasources/jdbc-driver=$DB_DRIVER_NAME:add( \
  driver-name=$DB_DRIVER_NAME, \
  driver-module-name=com.oracle, \
  driver-class-name=oracle.jdbc.driver.OracleDriver, \
  driver-xa-datasource-class-name=oracle.jdbc.xa.client.OracleXADataSource \
)

/subsystem=datasources/data-source=KeycloakDS:remove()

/subsystem=datasources/data-source=KeycloakDS:add( \
  jndi-name=java:jboss/datasources/KeycloakDS, \
  enabled=true, \
  use-java-context=true, \
  connection-url=$DB_JDBC_URL, \
  driver-name=$DB_DRIVER_NAME, \
  user-name=$DB_USERNAME, \
  password=$DB_PASSWORD \
)






/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=background-validati
on, value=true)


/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=background-validati
on-millis, value=30000)

 /subsystem=datasources/data-source=KeycloakDS:write-attribute(name=validate-on-match,
value=false)


/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=check-valid-connecti
on-sql, value=select 1)

/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=valid-connection-che
cker-class-name,
value=org.jboss.jca.adapters.jdbc.extensions.oracle.OracleValidConnectionChecker)


/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=stale-connection-ch
ecker-class-name,
value=org.jboss.jca.adapters.jdbc.extensions.oracle.OracleStaleConnectionChecker)


/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=exception-sorter-cla
ss-name, value=org.jboss.jca.adapters.jdbc.extensions.oracle.OracleExceptionSorter)


/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=min-pool-size,
value=30)

/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=max-pool-size,
value=30)

/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=prepared-statement
s-cache-size, value=100)


/subsystem=datasources/data-source=KeycloakDS:write-attribute(name=flush-strategy,
value=IdleConnections)


run-batch
```

Checking databasechangelog schema

You can verify that the database changelog schema has been updated to Keycloak 18 (RH-SSO is mapped on Keycloak 18).

See Liquibase documentation for more info about more info about liquibase schema .

```
Sqlplus <username>/<password>@<external-db-oracle-URL>

SELECT FILENAME, ID, EXECTYPE FROM DATABASECHANGELOG ORDER BY
DATEEXECUTED;
….
```

```
….

META-INF/jpa-changelog-14.0.0.xml
KEYCLOAK-17267-add-index-to-user-attributes

FILENAME
--------------------------------------------------------------------------------
ID
--------------------------------------------------------------------------------
EXECTYPE
----------
EXECUTED

META-INF/jpa-changelog-14.0.0.xml
KEYCLOAK-18146-add-saml-art-binding-identifier
EXECUTED

META-INF/jpa-changelog-15.0.0.xml

FILENAME
--------------------------------------------------------------------------------
ID
--------------------------------------------------------------------------------
EXECTYPE
----------
15.0.0-KEYCLOAK-18467
EXECUTED

META-INF/jpa-changelog-17.0.0.xml
17.0.0-9562
EXECUTED


FILENAME
--------------------------------------------------------------------------------
ID
--------------------------------------------------------------------------------
EXECTYPE
----------
META-INF/jpa-changelog-18.0.0.xml
18.0.0-10625-IDX_ADMIN_EVENT_TIME
EXECUTED


106 rows selected.

SQL>
```