Set DefaultRoute to True:

oc patch configs.imageregistry.operator.openshift.io/cluster --patch '{"spec":{"defaultRoute":true}}' --type=merge

HOST=$(oc get route default-route -n openshift-image-registry --template='{{ .spec.host }}')

docker login -u $(oc whoami) -p $(oc whoami -t) $HOST

** Get openldap**

Download openldap image from git

*$ git clone https://github.com/ibmgcgmikechang/openldap.git*

This should create a directory called openldap and create the content within it.

** Create required users and groups**

Change directory to openldap/bootstrap/ldif

*$ cd openldap/bootstrap/ldif*

Edit the openldap-default.ldif file

*$vi openldap-default.ldif*

Edit the ldif file to create users and groups as per your requirements

** Build the docker Image with the ldif file**

Change directory to the base folder

$ cd ../..

Build the docker image

$ make

Check that the image is created

$ docker images

Create ldap project

$oc new-project ldap

Tag Image

$ docker tag osixia/extend-osixia-openldap:0.1.0 default-route-openshift-image-registry.roks43-c0cf31a7105ba2e43673410c8b780e74-0000.hkg02.containers.appdomain.cloud/ldap/openldap:1.0.0

Push image

$docker push default-route-openshift-image-registry.roks43-c0cf31a7105ba2e43673410c8b780e74-0000.hkg02.containers.appdomain.cloud/ldap/openldap:1.0.0

Give required permission to the Container

$ oc adm policy add-scc-to-group anyuid system:serviceaccounts:ldap

oc adm policy add-scc-to-user anyuid -z default -n ldap

oc policy add-role-to-user system:image-puller system:serviceaccount:ldap:default --namespace=ldap

Create an image pull secret that ldap can use to pull an image from the image registry:

$ kubectl -n ldap create secret docker-registry **ldapsecret** --docker-server=default-route-openshift-image-registry.roks43-c0cf31a7105ba2e43673410c8b780e74-0000.hkg02.containers.appdomain.cloud --docker-username=$(oc whoami) --docker-password=$(oc whoami -t) [--docker-email=mjchang@tw.ibm.com](mailto:--docker-email=mjchang@tw.ibm.com)

Create a yaml file with the following content and name it openldap.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: openldap

labels:

app: openldap

spec:

replicas: 1

selector:

matchLabels:

app: openldap

template:

metadata:

labels:

app: openldap

spec:

containers:

- image: default-route-openshift-image-registry.roks43-c0cf31a7105ba2e43673410c8b780e74-0000.hkg02.containers.appdomain.cloud/ldap/openldap:1.0.0

imagePullPolicy: Always

imagePullSecrets:

- name: ldapsecret

name: openldap

ports:

- containerPort: 389

protocol: TCP

---

apiVersion: v1

kind: Service

metadata:

labels:

app: openldap

name: ldap-service

spec:

ports:

- name: openldap

port: 389

selector:

app: openldap

Create the OpenLDAP container and the service

oc secrets link default ldapsecret --for=pull

$ oc create -f openldap.yaml

Get all info

$ oc get all

NAME READY STATUS RESTARTS AGE

pod/openldap-6ffbd7bf69-stjcz 1/1 Running 0 4m42s

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

service/ldap-service ClusterIP **172.21.90.217**  <none> **389/TCP** 4m42s

NAME READY UP-TO-DATE AVAILABLE AGE

deployment.apps/openldap 1/1 1 1 4m42s

NAME DESIRED CURRENT READY AGE

replicaset.apps/openldap-6ffbd7bf69 1 1 1 4m43s

NAME IMAGE REPOSITORY TAGS UPDATED

imagestream.image.openshift.io/openldap image-registry.openshift-image-registry.svc:5000/ldap/openldap 1.0.0 About an hour ago

* *Connection name – LocalOpenLDAP*
* *Server Type – Custom*
* *Base DN – dc=ibm,dc=com*
* *Bind DN – cn=admin,dc=ibm,dc=com*
* *Bind DN password – Passw0rd*
* *url – ldap://LDAP\_SERVICE\_URL (The url you have determined in the previous step)*
* *Group filter: (&(cn=%v)(objectclass=groupOfUniqueNames))*
* *User filter: (&(uid=%v)(objectclass=inetOrgPerson))*
* *Group ID map: \*:cn*
* *User ID map: \*:uid*
* *Group member ID map: groupOfUniqueNames:uniqueMember*

Click on Test Connection. If everything has been configured appropriately, a success message is displayed in the URL text box itself.

Click on Create buttion at top right corner to create the LDAP connection