

Lab: Manage Storage for Application Configuration and Data

Deploy a web application and its database that share database credentials from a secret. The database should use the default storage for the cluster. Also, you deploy a file-sharing application that runs with multiple replicas and shares its storage volume with a file uploader application. The file sharing and file uploader applications take configuration files from a configuration map and should use NFS file storage for shareability. The database should use local storage for increased performance.

Outcomes

- Deploy a database server.
- Deploy a web application.
- Create a secret that contains the database server credentials.
- Create a configuration map that contains an SQL file.
- Add and remove a volume on the database server and the web application.
- Expose the database server and the web application.
- Scale up the web application.
- Mount the configuration map as a volume.

As the student user on the workstation machine, use the `lab` command to prepare your system for this exercise. This command ensures that the cluster is accessible and that all resources are available for this exercise.

```
[student@workstation ~]$ lab start storage-review
```

Instructions

In this exercise, you work in the `storage-review` project that the `lab` command created. The command also created files in the `/home/student/DO180/labs/storage-review` directory that you use in the exercise.

The API URL of your OpenShift cluster is `https://api.ocp4.example.com:6443`, and the `oc` command is already installed on your workstation machine.

1. Log in to the OpenShift cluster as the `developer` user with `developer` as the password, and change to the `storage-review` project. You use this project for all your work in this exercise.

Log in to the OpenShift cluster.

```
[student@workstation ~]$ oc login -u developer -p developer \
https://api.ocp4.example.com:6443
...output omitted...
```

Change to the storage-review project.

```
[student@workstation ~]$ oc project storage-review
...output omitted...
```

2. Create a secret named `world-cred` that contains the following data:

| Field | Value |
|----------|-----------|
| user | redhat |
| password | redhat123 |
| database | world_x |

Create a secret that contains the database credentials.

```
[student@workstation]$ oc create secret generic world-cred \
--from-literal user=redhat \
--from-literal password=redhat123 \
--from-literal database=world_x
secret/world-cred created
```

Confirm the creation of the secret.

```
[student@workstation ~]$ oc get secrets world-cred
NAME          TYPE      DATA   AGE
world-cred    Opaque    3       2m34s
```

3. Create a configuration map named `dbfiles` by using the `~/D0180/labs/storage-review/insertdata.sql` file.

Create a configuration map named `dbfiles` by using the `insertdata.sql` file in the `~/D0180/labs/storage-review` directory.

```
[student@workstation ~]$ oc create configmap dbfiles \
--from-file ~/D0180/labs/storage-review/insertdata.sql
configmap/dbfiles created
```

Verify the creation of the configuration map.

```
[student@workstation]$ oc get configmaps
NAME      DATA  AGE
dbfiles   1      11s
...output omitted...
```

4. Create a database server deployment named `dbserver` by using the `registry.ocp4.example.com:8443/redhattraining/mysql-app:v1` container image. Then, set the missing environment variables by using the `world-cred` secret. Use the `MYSQL_` prefix for the variables.

Create the database server deployment.

```
[student@workstation ~]$ oc create deployment dbserver \
  --image \
  registry.ocp4.example.com:8443/redhattraining/mysql-app:v1
deployment.apps/dbserver created
```

Set the missing environment variables.

```
[student@workstation ~]$ oc set env deployment/dbserver \
  --from secret/world-cred --prefix MYSQL_
deployment.apps/dbserver updated
```

Verify that the `dbserver` pod is in the `RUNNING` state. The pod name might differ in your cluster.

```
[student@workstation ~]$ oc get pods
NAME                                READY   STATUS   ...
dbserver-6d5bf5d86c-ptrb2          1/1     Running  ...
```

5. Add a volume to the `dbserver` deployment by using the following information:

| Field | Value |
|-------------|-----------------------|
| Name | dbserver-lvm |
| Type | persistentVolumeClaim |
| Claim mod | rwo |
| Claim size | 1Gi |
| Mount path | /var/lib/mysql |
| Claim class | lvms-vg1 |
| Claim name | dbserver-lvm-pvc |

Add a volume to the dbserver deployment.

```
[student@workstation ~]$ oc set volume deployment/dbserver \
--add --name dbserver-lvm --type persistentVolumeClaim \
--claim-mode rwo --claim-size 1Gi \
--mount-path /var/lib/mysql --claim-class lvms-vg1 \
--claim-name dbserver-lvm-pvc
deployment.apps/dbserver volume updated
```

Verify the deployment status.

```
[student@workstation ~]$ oc get pods
NAME                                READY   STATUS    ...
dbserver-5bc6bd5d7b-7z7lv          1/1     Running   ...
```

Verify the volume status.

```
[student@workstation ~]$ oc get pvc
NAME              STATUS    VOLUME             CAPACITY ...
dbserver-lvm-pvc  Bound    pvc-2cb8...5025    1Gi       ...
```

6. Create a service for the dbserver deployment by using the following information:

| Field | Value |
|-------------|---------------|
| Name | mysql-service |
| Port | 3306 |
| Target port | 3306 |

Expose the dbserver deployment.

```
[student@workstation ~]$ oc expose deployment dbserver \
--name mysql-service --port 3306 --target-port 3306
service/mysql-service exposed
```

Verify the service configuration. The endpoint IP address might differ in your output.

```
[student@workstation ~]$ oc get services
NAME              TYPE        CLUSTER-IP    EXTERNAL-IP  PORT(S)
mysql-service     ClusterIP   172.30.240.100 <none>       3306/TCP
[student@workstation ~]$ oc get endpoints
NAME              ENDPOINTS             AGE
mysql-service     10.8.1.36:3306        20s
```

7. Create a web application deployment named `file-sharing` by using the `registry.ocp4.example.com:8443/redhattraining/php-webapp-mysql:v1` container image. Scale the deployment to two replicas. Then, expose the deployment by using the following information:

| Field | Value |
|-------------|--------------|
| Name | file-sharing |
| Port | 8080 |
| Target port | 8080 |

Create a route named `file-sharing` to expose the `file-sharing` web application to external access. Access the `file-sharing` route in a web browser to test the connection between the web application and the database server.

Create a web application deployment.

```
[student@workstation ~]$ oc create deployment file-sharing \
  --image registry.ocp4.example.com:8443/redhattraining/php-webapp-mysql:v1
deployment.apps/file-sharing created
```

Verify the deployment status. Verify that the `file-sharing` application pod is in the `RUNNING` state on your system.

```
[student@workstation ~]$ oc get pods
NAME                                READY   STATUS    ...
dbserver-5bc6bd5d7b-7z7lv          1/1     Running   ...
file-sharing-789c5948c8-gdrlz      1/1     Running   ...
```

Scale the deployment to two replicas.

```
[student@workstation ~]$ oc scale deployment file-sharing \
  --replicas 2
deployment.apps/file-sharing scaled
```

Verify the replica status and retrieve the pod name. The pod names might differ on your system.

```
[student@workstation ~]$ oc get pods
NAME                                READY   STATUS    ...
dbserver-5bc6bd5d7b-7z7lv          1/1     Running   ...
file-sharing-789c5948c8-62j9s      1/1     Running   ...
file-sharing-789c5948c8-gdrlz      1/1     Running   ...
```

Expose the `file-sharing` deployment.

```
[student@workstation ~]$ oc expose deployment file-sharing \
  --name file-sharing --port 8080 --target-port 8080
service/file-sharing exposed
```

Verify the service configuration. The endpoint IP address might differ in your output.

```
[student@workstation ~]$ oc get services
NAME          TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)
file-sharing   ClusterIP     172.30.139.210 <none>         8080/TCP
mysql-service  ClusterIP     172.30.240.100 <none>         3306/TCP
[student@workstation ~]$ oc get endpoints
NAME          ENDPOINTS                                AGE
file-sharing   10.8.1.37:8080,10.8.1.38:8080          14s
mysql-service  10.8.1.36:3306                          5m23s
```

Expose the file-sharing service.

```
[student@workstation ~]$ oc expose service/file-sharing
route.route.openshift.io/file-sharing exposed
[student@workstation ~]$ oc get routes
NAME          HOST/PORT    ...
file-sharing   file-sharing-storage-review.apps.ocp4.example.com
```

Test the connectivity between the web application and the database server. In a web browser, access `storage-review.apps.ocp4.example.com`, and verify that the Connected successfully message is displayed.

8. Mount the dbfiles configuration map to the file-sharing deployment as a volume named config-map-pvc. Set the mount path to the `/home/database-files` directory. Then, verify the content of the `insertdata.sql` file.

Mount the dbfiles configuration map to the file-sharing deployment.

```
[student@workstation ~]$ oc set volume deployment/file-sharing \
  --add --name config-map-pvc --type configmap \
  --configmap-name dbfiles \
  --mount-path /home/database-files
deployment.apps/file-sharing volume updated
```

Verify the deployment status.

```
[student@workstation ~]$ oc get pods
NAME                                READY   STATUS    ...
dbserver-5bc6bd5d7b-7z7lv           1/1     Running   ...
file-sharing-7f77855b7f-949lg        1/1     Running   ...
file-sharing-7f77855b7f-9zvq         1/1     Running   ...
```

Verify the content of the `/home/database-files/insertdata.sql` file.

```
[student@workstation ~]$ oc exec -it \
  pod/file-sharing-7f77855b7f-949lg -- \
  head /home/database-files/insertdata.sql
-- MySQL dump 10.13  Distrib 8.0.19, for osx10.14 (x86_64)
--
-- Host: 127.0.0.1    Database: world_x
--
-- Server version 8.0.19-debug
...output omitted...
```

9. Add a shared volume to the file-sharing deployment. Use the following information to create the volume:

| Field | Value |
|-------------|-----------------------|
| Name | shared-volume |
| Type | persistentVolumeClaim |
| Claim mode | rwo |
| Claim size | 1Gi |
| Mount path | /home/sharedfiles |
| Claim class | nfs-storage |
| Claim name | shared-pvc |

Next, connect to one of the file-sharing deployment pods, and use the `cp` command to copy the `/home/database-files/insertdata.sql` file to the `/home/sharedfiles` directory. Then, remove the `config-map-pvc` volume from the file-sharing deployment.

Add the shared-volume volume to the file-sharing deployment.

```
[student@workstation ~]$ oc set volume deployment/file-sharing \
  --add --name shared-volume --type persistentVolumeClaim \
  --claim-mode rwo --claim-size 1Gi \
  --mount-path /home/sharedfiles --claim-class nfs-storage \
  --claim-name shared-pvc
deployment.apps/file-sharing volume updated
```

Verify the deployment status. Your pod names might differ on your system.

```
[student@workstation ~]$ oc get pods
NAME                                READY   STATUS   ...
dbserver-5bc6bd5d7b-7z7lv          1/1     Running  ...
file-sharing-65884f75bb-92fxf      1/1     Running  ...
file-sharing-65884f75bb-gsghk      1/1     Running  ...
```

Verify the volume status.

```
[student@workstation ~]$ oc get pvc
```

| NAME | STATUS | VOLUME | CAPACITY | ... |
|------------------|--------|-----------------|----------|-----|
| dbserver-lvm-pvc | Bound | pvc-2cb8...5025 | 1Gi | ... |
| shared-pvc | Bound | pvc-cf2d...de48 | 1Gi | ... |

Copy the /home/database-files/insertdata.sql file to the /home/sharedfiles path.

```
[student@workstation ~]$ oc exec -it \
  pod/file-sharing-65884f75bb-92fxf -- \
  cp /home/database-files/insertdata.sql /home/sharedfiles/
```

```
[student@workstation ~]$ oc exec -it \
  pod/file-sharing-65884f75bb-92fxf -- \
  ls /home/sharedfiles/
insertdata.sql
```

Remove the config-map-pvc volume from the file-sharing deployment.

```
[student@workstation ~]$ oc set volume deployment/file-sharing \
  --remove --name=config-map-pvc
deployment.apps/file-sharing volume updated
```

10. Add the shared-volume PVC to the dbserver deployment. Then, connect to the dbserver deployment pod and verify the content of the /home/sharedfiles/insertdata.sql file.

Add the shared-volume volume to the dbserver deployment.

```
[student@workstation ~]$ oc set volume deployment/dbserver \
  --add --name shared-volume \
  --claim-name shared-pvc \
  --mount-path /home/sharedfiles
deployment.apps/dbserver volume updated
```

Verify the deployment status. The pod names might differ on your system.

```
[student@workstation ~]$ oc get pods
```

| NAME | READY | STATUS | ... |
|-------------------------------|-------|---------|-----|
| dbserver-6676fbf5fc-n9hpk | 1/1 | Running | ... |
| file-sharing-5fdb44cf57-2hhwj | 1/1 | Running | ... |
| file-sharing-5fdb44cf57-z4n7g | 1/1 | Running | ... |

Verify the content of the /home/sharedfiles/insertdata.sql file.


```
[student@workstation ~]$ oc exec -it \
  pod/dbserver-6676fbf5fc-n9hpk -- \
  head /home/sharedfiles/insertdata.sql
-- MySQL dump 10.13  Distrib 8.0.19, for osx10.14 (x86_64)
--
-- Host: 127.0.0.1    Database: world_x
-- -----
-- Server version 8.0.19-debug
...output omitted...
```

11. Connect to the database server and execute the `/home/sharedfiles/insertdata.sql` file to add data to the `world_x` database. You can execute the file by using the following command:

```
mysql -u$MYSQL_USER -p$MYSQL_PASSWORD world_x < \
/home/sharedfiles/insertdata.sql
```

Then, confirm connectivity between the web application and the database server by accessing the file-sharing route in a web browser.

Connect to the database server and execute the `/home/sharedfiles/insertdata.sql` file

```
[student@workstation ~]$ oc rsh dbserver-6676fbf5fc-n9hpk
```

```
sh-4.4$ mysql -u$MYSQL_USER -p$MYSQL_PASSWORD world_x < \
/home/sharedfiles/insertdata.sql
mysql: [Warning] Using a password on the command line interface can be insecure.
sh-4.4$ exit
exit
```

Test the connectivity between the web application and the database server. In a web browser, access `storage-review.apps.ocp4.example.com`, and verify that the application retrieves data from the database.

Connected successfully

| | | | |
|---|----------------|-------------------|-------------------------|
| 1 | Kabul | AFG Kabul | {"Population": 1780000} |
| 2 | Qandahar | AFG Qandahar | {"Population": 1780000} |
| 3 | Herat | AFG Herat | {"Population": 1780000} |
| 4 | Mazar-e-Sharif | AFG Balkh | {"Population": 1780000} |
| 5 | Amsterdam | NLD Noord-Holland | {"Population": 1780000} |
| 6 | Rotterdam | NLD Zuid-Holland | {"Population": 1780000} |

Evaluation

As the student user on the workstation machine, use the `lab` command to grade your work. Correct any reported failures and rerun the command until successful.

```
[student@workstation ~]$ lab grade storage-review
```

Finish

As the student user on the workstation machine, use the `lab` command to complete this exercise. This step is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish storage-review
```