Guided Exercise: Troubleshoot Containers and Pods

Troubleshoot and fix a failed MySQL pod and manually initialize a database with test data.

Outcomes

- Investigate errors with creating a pod.
- View the status, logs, and events for a pod.
- Copy files into a running pod.
- Connect to a running pod by using port forwarding.

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 and all exercise resources are available.

```
[student@workstation ~]$ lab start pods-troubleshooting
```

Instructions

1. Log in to the OpenShift cluster and create the `pods-troubleshooting` project.

   Open a new terminal window and log in to the OpenShift cluster as the `developer` user with `developer` as the password.

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

   Create the `pods-troubleshooting` project.

   ```
   [student@workstation ~]$ oc new-project pods-troubleshooting
   Now using project "pods-troubleshooting" on server "https://api.ocp4.example.com:6443".
   ...output omitted...
   ```

2. Create the MySQL `mysql-server` pod. Use the `registry.ocp4.example.com:8443/rhel9/mysql-80:1228` container image for the pod. Specify the following environment variables:

Variable	Value
MYSQL_USER	redhat
MYSQL_PASSWORD	redhat123
MYSQL_DATABASE	world

   Use the `oc run` command to create the `mysql-server` pod. Specify the environment values with the `--env` option.

   ```
   [student@workstation ~]$ oc run mysql-server \
     --image registry.ocp4.example.com:8443/rhel9/mysql-80:1228 \
     --env MYSQL_USER=redhat \
     --env MYSQL_PASSWORD=redhat123 \
     --env MYSQL_DATABASE=world
   pod/mysql-server created
   ```

   Retrieve the status of the pod by using the `oc get pods` command. Execute the `oc get pods` command a few times to view the status updates for the pod.

   ```
   [studet@workstation ~]$ oc get pods
   NAME           READY   STATUS             RESTARTS   AGE
   mysql-server   0/1     ImagePullBackoff   0          45s
   ```

   The pod fails to start.

3. Identify the root cause of the pod's failure to start.

   Use the `oc logs` command to retrieve the pod's logs.

   ```
   [student@workstation ~]$ oc logs mysql-server
   Error from server (BadRequest): container "mysql-server" in pod "mysql-server" is waiting to start: trying and failing to
   pull image
   ```

The logs state that the pod cannot pull the container image.

Retrieve the events log with the `oc get events` command.

```
student@workstation ~]$ oc get events
LAST SEEN   TYPE      REASON           OBJECT             MESSAGE
33s         Normal    Scheduled        pod/mysql-server   Successfully assigned pods-troubleshooting/mysql-server to maste
r01
31s         Normal    AddedInterface   pod/mysql-server   Add eth0 [10.8.0.68/23] from ovn-kubernetes
16s         Normal    Pulling          pod/mysql-server   Pulling image "registry.ocp4.example.com:8443/rhel9/mysql-80:122
8"
16s         Warning   Failed           pod/mysql-server   Failed to pull image
"registry.ocp4.example.com:8443/rhel9/mysql-80:1228": initializing source
docker://registry.ocp4.example.com:8443/rhel9/mysql-80:1228: reading manifest 1228 in
registry.ocp4.example.com:8443/rhel9/mysql-80: manifest unknown
16s         Warning   Failed           pod/mysql-server   Error: ErrImagePull
4s          Normal    BackOff          pod/mysql-server   Back-off pulling image "registry.ocp4.example.com:8443/rhel9/mys
ql-80:1228"
4s          Warning   Failed           pod/mysql-server   Error: ImagePullBackOff
```

The output states that the image pull failed because the `1228` manifest is unknown. This failure could mean that the manifest, or image tag, does not exist in the repository.

Authenticate to the container repository with the `skopeo login` command as the `developer` user with `developer` as the password.

```
[student@workstation ~]$ skopeo login registry.ocp4.example.com:8443
Username: developer
Password: developer
Login Succeeded!
```

Use the `skopeo inspect` command to retrieve the available manifests in the `registry.ocp4.example.com:8443/rhel9/mysql-80` repository.

```
[student@workstation ~]$ skopeo inspect \
  docker://registry.ocp4.example.com:8443/rhel9/mysql-80
...output omitted...
    "Name": "registry.ocp4.example.com:8443/rhel9/mysql-80",
    "Digest": "sha256:5098...72a1",
    "RepoTags": [
        "1",
        "1-224",
        "1-224-source",
        "1-228",
        "1-228-source",
        "1-237",
        "latest"
    ],
...output omitted...
```

The `1228` manifest, or tag, is not available in the repository, which means that the `registry.ocp4.example.com:8443/rhel9/mysql-80:1228` image does not exist. However, the `1-228` tag does exist.

4. Update the pod's configuration to use the `registry.ocp4.example.com:8443/rhel9/mysql-80:1-228` container image. Confirm that the pod is re-created after editing the resource.

   Locate the `.spec.containers.image` object. Use the `oc edit` command to update the value to the `registry.ocp4.example.com:8443/rhel9/mysql-80:1-228` container image, and save the change.

```
[student@workstation ~]$ oc edit pod/mysql-server
...output omitted...
apiVersion: v1
kind: Pod
metadata:
...output omitted...
spec:
  containers:
  - image: registry.ocp4.example.com:8443/rhel9/mysql-80:1-228
...output omitted...
```

   Verify the status of the `mysql-server` pod with the `oc get` command. The pod's status might take a few moments to update after the resource edit. Repeat the `oc get` command until the pod's status changes.

```
[student@workstation ~]$ oc get pods
NAME           READY   STATUS     RESTARTS      AGE
mysql-server   1/1     Running    0             10m
```

The `mysql-server` pod successfully pulled the image and created the container. The pod now shows a `Running` status.

5. Copy the `~/DO180/labs/pods-troubleshooting/world_x.sql` file to the `mysql-server` pod. Then, connect to the pod and execute the `world_x.sql` file to populate the `world` database.

   Use the `oc cp` command to copy the `world_x.sql` file in the `~/DO180/labs/pods-troubleshooting` directory to the `/tmp/` directory on the `mysql-server` pod.

```
[student@workstation ~]$ oc cp ~/DO180/labs/pods-troubleshooting/world_x.sql \
  mysql-server:/tmp/
```

   Confirm that the `world_x.sql` file is accessible within the `mysql-server` pod with the `oc exec` command.

```
[student@workstation ~]$ oc exec -it pod/mysql-server -- ls -la /tmp/
total 548
drwxrwxrwx. 1 root       root  25     Aug  15 01:53 .
dr-xr-xr-x. 1 root       root  50     Aug  15 01:50 ..
-rw-r--r--. 1 1000750000 root  558791 Aug  15 01:53 world_x.sql
```

   Connect to the `mysql-server` pod with the `oc rsh` command.

```
[student@workstation ~]$ oc rsh mysql-server
sh-5.1$
```

   Log in to MySQL as the `redhat` user with `redhat123` as the password.

```
sh-5.1$ mysql -u redhat -p
Enter password: redhat123
Welcome to the MySQL monitor.  Commands end with ; or \g.
...output omitted...
mysql>
```

   From the MySQL prompt, select the `world` database.

```
mysql> USE world
Database changed
```

   Source the `world_x.sql` script inside the pod to initialize and populate the `world` database.

```
mysql> SOURCE /tmp/world_x.sql
Query OK, 1 row affected (0.00 sec)
...output omitted...
```

   Execute the `SHOW TABLES;` command to confirm that the database now contains tables. Then, exit the database and the pod.

```
mysql> SHOW TABLES;
-----------------
| Tables_in_test  |
-----------------
| city            |
| country         |
| countryinfo     |
| countrylanguage |
-----------------
4 rows in set (0.00 sec)
mysql> exit
Bye
sh-5.1$ exit
exit
[student@workstation ~]$
```

6. Configure port forwarding and then use the MySQL client on the terminal to connect to the `world` database on the `mysql-server` pod. Confirm that you can access data within the `world` database from the terminal.

   From the terminal, use the `oc port-forward` command to forward the `3306` local port to the `3306` port on the `mysql-server` pod.

```
[student@workstation ~]$ oc port-forward mysql-server 3306:3306
Forwarding from 127.0.0.1:3306 -> 3306
Forwarding from [::1]:3306 -> 3306
```

Open another terminal window on the `workstation` machine. Connect to the `world` database with the local MySQL client on the `workstation` machine. Log in as the `redhat` user with the `redhat123` password. Specify the host as the localhost `127.0.0.1` IP address and use `3306` as the port.

```
[student@workstation ~]$ mysql -u redhat -p -h 127.0.0.1 -P 3306
Enter password: redhat123
Welcome to the MySQL monitor.  Commands end with ; or \g.
...output omitted...
mysql>
```

Select the `world` database.

```
mysql> USE world
Reading table information for completion of table and columns names
You can turn off this feature to get a quicker startup with -A

Database changed
```

Use the `SHOW TABLES;` command to list the tables in the `world` database.

```
mysql> SHOW TABLES;
-----------------
| Tables_in_test  |
-----------------
| city            |
| country         |
| countryinfo     |
| countrylanguage |
-----------------
4 rows in set (0.01 sec)
```

Confirm that you can retrieve data from the `country` table. Execute the `SELECT COUNT(*) FROM country` command to retrieve the number of countries within the `country` table.

```
mysql> SELECT COUNT(*) FROM country;
----------
| COUNT(*) |
----------
|      239 |
----------
1 row in set (0.01 sec)
```

Exit the database.

```
mysql> exit
Bye
[student@workstation ~]$
```

Close the second terminal window.

Return to the terminal that is executing the `oc port-forward` command. Press **Ctrl**+**C** to end the connection.

```
[student@workstation ~]$ oc port-forward mysql-server 3306:3306
Forwarding from 127.0.0.1:3306 -> 3306
Forwarding from [::1]:3306 -> 3306
Handling connection for 3306
Handling connection for 3306
^C[student@workstation ~]$
```

Finish

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 pods-troubleshooting
```