Guided Exercise: Limit Compute Capacity for Applications

Configure an application with compute resource limits that allow and prevent successful execution of its pods.

Outcomes

You should be able to monitor the memory usage of an application, and set a memory limit for a pod.

As the `student` user on the `workstation` machine, use the `lab` command to prepare your system for this exercise.

This command ensures that all resources are available for this exercise. It also creates the `reliability-limits` project and the `/home/student/DO180/labs/reliability-limits/resources.txt` file. The `resources.txt` file contains some commands that you use during the exercise. You can use the file to copy and paste these commands.

```
[student@workstation ~]$ lab start reliability-limits
```

Instructions

1. Log in to the OpenShift cluster as the `developer` user with the `developer` password. Use the `reliability-limits` project.

   Log in to the OpenShift cluster.

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

   Set the `reliability-limits` project as the active project.

   ```
   [student@workstation ~]$ oc project reliability-limits
   ...output omitted...
   ```

2. Create the `leakapp` deployment from the `~/DO180/labs/reliability-limits/leakapp.yml` file that the `lab` command prepared. The application has a bug, and leaks 1 MiB of memory every second.

   Review the `~/DO180/labs/reliability-limits/leakapp.yml` resource file. The memory limit is set to 35 MiB. Do not change the file.

   ```
   ...output omitted...
           resources:
             requests:
               memory: 20Mi
             limits:
               memory: 35Mi
   ```

   Use the `oc apply` command to create the application. Ignore the warning message.

   ```
   [student@workstation ~]$ oc apply -f \
     ~/DO180/labs/reliability-limits/leakapp.yml
   deployment.apps/leakapp created
   ```

   Wait for the pod to start. You might have to rerun the command several times for the pod to report a `Running` status. The name of the pod on your system probably differs.

   ```
   [student@workstation ~]$ oc get pods
   NAME                       READY   STATUS    RESTARTS   AGE
   leakapp-99bb64c8d-hk26k    1/1     Running   0          12s
   ```

3. Watch the pod. OpenShift restarts the pod after 30 seconds.

   Use the `watch` command to monitor the `oc get pods` command. Wait for OpenShift to restart the pod, and then press **Ctrl**+**C** to quit the `watch` command.

   ```
   [student@workstation ~]$ watch oc get pods
   Every 2.0s: oc get pods                    workstation: Thur Sep  4 09:04:47 2025

   NAME                       READY   STATUS    RESTARTS      AGE
   leakapp-99bb64c8d-hk26k    1/1     Running   1 (15s ago)   48s
   ```

   Retrieve the container status to verify that OpenShift restarted the pod due to an Out-Of-Memory (OOM) event.

```
[student@workstation ~]$ oc get pods leakapp-99bb64c8d-hk26k \
  -o jsonpath='{.status.containerStatuses[0].lastState}' | jq .
{
  "terminated": {
    "containerID": "cri-o://5800...1d04",
    "exitCode": 137,
    "finishedAt": "2025-09-04T09:04:47Z",
    "reason": "OOMKilled",
    "startedAt": "2025-09-04T09:04:17Z"
  }
}
```

4. Observe the pod status for a few minutes, until the `CrashLoopBackOff` status is displayed. During this period, OpenShift restarts the pod several times because of the memory leak.

   Between each restart, OpenShift sets the pod status to `CrashLoopBackOff`, waits an increasing amount of time between retries, and then restarts the pod. The delay between restarts gives the operator the opportunity to fix the issue.

   After various retries, OpenShift finally sets the `CrashLoopBackOff` wait timer to five minutes. During this wait time, the application is not available to your customers.

   ```
   [student@workstation ~]$ watch oc get pods
   Every 2.0s: oc get pods                    workstation: Thur Sep  4 09:04:47 2025

   NAME                       READY   STATUS             RESTARTS      AGE
   leakapp-99bb64c8d-hk26k    0/1     CrashLoopBackOff   4 (82s ago)   5m25s
   ```

   Press **Ctrl**+**C** to quit the `watch` command.

5. Fixing the memory leak would resolve the issue. However, it might take some time for the developers to fix the bug. In the meantime, set the memory limit to 600 MiB. With this setting, the pod can run for ten minutes before the application reaches the limit.

   Use the `oc set resources` command to set the new limit. Ignore the warning message.

   ```
   [student@workstation ~]$ oc set resources deployment/leakapp \
     --limits memory=600Mi
   deployment.apps/leakapp resource requirements updated
   ```

   Wait for the pod to start. You might have to rerun the command several times for the pod to report a `Running` status. The name of the pod on your system probably differs.

   ```
   [student@workstation ~]$ oc get pods
   NAME                       READY   STATUS    RESTARTS   AGE
   leakapp-6bc64dfcd-86fpc    1/1     Running   0          12s
   ```

   Wait two minutes to verify that OpenShift no longer restarts the pod every 30 seconds.

   ```
   [student@workstation ~]$ watch oc get pods
   Every 2.0s: oc get pods                    workstation: Thur Sep  4 09:04:47 2025

   NAME                       READY   STATUS    RESTARTS   AGE
   leakapp-6bc64dfcd-86fpc    1/1     Running   0          3m12s
   ```

   Press **Ctrl**+**C** to quit the `watch` command.

6. Review the memory that the pod consumes. The metrics for the pod are available a few minutes after the pod starts. You might have to rerun the command after the metrics are available, or if the command displays an error message that the metrics are not available. The memory usage on your system can differ.

   ```
   [student@workstation ~]$ oc adm top pods
   NAME                       CPU(cores)   MEMORY(bytes)
   leakapp-6bc64dfcd-86fpc    1m           133Mi
   ```

7. **Optional**. Wait about 10 minutes from the creation time until the application reaches the out of memory error. After this period, OpenShift restarts the pod, because it reached the 600 MiB memory limit.

   Open a new terminal window, and then run the `watch` command to monitor the `oc adm top pods` command.

```
[student@workstation ~]$ watch oc adm top pods
Every 2.0s: oc adm top pods            workstation: Thur Sep  4 09:04:47 2025

NAME                       CPU(cores)   MEMORY(bytes)
leakapp-6bc64dfcd-86fpc    0m           176Mi
```

Leave the command running and do not interrupt it.

> **NOTE**
>
> You might see a message that metrics are not yet available. If so, wait some time and try again.

In the first terminal, run the `watch` command to monitor the `oc get pods` command. Watch the output of the `oc adm top pods` command in the second terminal. When the memory usage reaches 600 MiB, the OOM subsystem kills the process inside the container, and OpenShift restarts the pod.

```
[student@workstation ~]$ watch oc get pods
Every 2.0s: oc get pods                    workstation: Thur Sep  4 09:04:47 2025

NAME                       READY   STATUS    RESTARTS     AGE
leakapp-6bc64dfcd-86fpc    1/1     Running   1 (3s ago)   9m58s
```

Press **Ctrl**+**C** to quit the `watch` command.

Press **Ctrl**+**C** to quit the `watch` command in the second terminal. Close this second terminal when done.

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 reliability-limits
```