Guided Exercise: Examining Cluster Metrics

In this exercise, you will examine the metrics page and dashboard within the web console.

**Outcomes**

You should be able to use the Red Hat OpenShift web console to:

* View cluster, project, pod, and node metrics.
* Identify a pod consuming large amounts of memory or CPU.

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 API is reachable and creates the resources required for this exercise.

**[student@workstation ~]$ lab console-metrics start**

1. As the admin user, locate and navigate to the OpenShift web console.
   1. Log in to your OpenShift cluster as the admin user.
   2. **[student@workstation ~]$ oc login -u admin -p redhat \**
   3. **>**  **https://api.ocp4.example.com:6443**
   4. Login successful.

*...output omitted...*

* 1. Identify the URL for the web console.
  2. **[student@workstation ~]$ oc whoami --show-console**

https://console-openshift-console.apps.ocp4.example.com

* 1. Open a web browser and navigate to https://console-openshift-console.apps.ocp4.example.com.
  2. Click **Advanced** to reveal the untrusted certificate message, and then click **Add Exception**. In the **Add Security Exception** dialog box, click **Confirm Security Exception**.

You will need to do this twice to skip the warnings about the self-signed SSL certificate for both the console-openshift-console and oauth-openshift subdomains.

* 1. Click **localusers** and log in as the admin user with the password of redhat.

1. In this guided exercise, you will see how changes in load are displayed in the web console. Start by observing baseline healthy metrics on the Overview, Pod Details, and Project Details pages.
   1. Click **Home** → **Overview** to display the **Overview** page. Scroll down to the **Cluster Utilization** section, which displays a time-series historical graph of the cluster's CPU, memory, and disk usage.
   2. For each resource in the table, such as **CPU**, **Memory**, or **Filesystem**, click the link in the **Usage** column to view the **Top Consumers** of that resource. By default, the window filters top consumers by project, but you can filter by pod or by node instead.
   3. Click the usage link for **Memory**, filter the top consumers by pod, and then click the name of the pod that consumes the most memory resources.

|  |
| --- |
|  |

* 1. Figure 8.21: Memory breakdown: Top consumers by pod
  2. The **Pod Details** page displays **Memory Usage**, **CPU Usage**, and **Filesystem** time-series historical graphs at the top of the page.
  3. Click **Home** → **Projects**, and then click **console-apps** to display the console-apps **Project Details** page.

Notice the **Utilization** section, which displays the metrics for the workloads running in the console-apps project. The links in the **Usage** column open windows displaying the pods that consume the most resources. The workloads are running safely within their limits.

* 1. Scroll down to the **Resource Quotas** section, which displays the current CPU and memory usage compared to the allotted quota.

1. Find and review the baseline health metrics of a compute node.
   1. Click **Compute** → **Nodes**, then click any of the nodes in the list.
   2. On the **Node Details** page, notice the time-series graphs that display the metrics for the individual node that you selected.

|  |
| --- |
|  |

* 1. Figure 8.22: Time-series graphs showing various metrics for a node.

1. On workstation, execute the load.sh script to generate load on the example books deployment. The application intentionally contains a memory leak that consumes multiple megabytes of RAM with every request to its /leak path.
   1. In a terminal on the workstation machine, run the following command.

**[student@workstation ~]$ ~/DO280/labs/console-metrics/load.sh**

1. In the OpenShift web console, observe the change in metrics and identify the problematic pod. The data displayed in the web console automatically refreshes, so there is no need to reload the page.
   1. Click **Home** → **Projects**, and then click **console-apps** to display the console-apps **Project Details** page. Watch the **Memory Usage** time-series graph to monitor for changes.

The memory leak may take a minute or two before it is significant enough to be visible. Although both CPU and memory increase, the total CPU usage remains low.

|  |
| --- |
|  |

Figure 8.23: Utilization graphs indicating a possible memory leak.

* 1. Click **Home** → **Overview** to display the **Overview** page. The memory consumed by the load test may be too small to notice across a large cluster, but the **Memory breakdown** window (sorted by pod) provides a convenient list of pods using the most memory. View the **Memory breakdown** window by clicking the usage link for **Memory**. Sort the top consumers by pod.

|  |
| --- |
|  |

* 1. Figure 8.24: The books pod is a top memory consumer.
  2. The books pod appears at or near the top of the list. If its not on the list, you may need to wait a minute longer for the load script to complete.
  3. Click the books pod link in the **Memory breakdown** window to navigate to the **Pod Details** page. Notice the climbing memory leak visible in the **Memory Usage** time-series graph.

|  |
| --- |
|  |

* 1. Click **Monitoring** → **Metrics** to display the web console **Metrics** page. Type the following Prometheus query in the expression input field:

avg(container\_memory\_working\_set\_bytes{namespace='console-apps'}) BY (pod)

Click **Run Queries** to view the results in the OpenShift web console.

|  |
| --- |
|  |

1. Delete the console-apps project and stop the load test.
   1. Click **Home** → **Projects**, and then click **Delete Project** in the menu at the end of the console-apps row.

|  |
| --- |
|  |

* 1. In the Delete Project dialog box, type console-apps and then click **Delete**.
  2. If load.sh is still running on the workstation terminal, press **Ctrl**+**C** in the terminal to stop the load test.

**Finish**

On the workstation machine, use the lab command to complete this exercise. This is important to ensure that resources from previous exercises do not impact upcoming exercises.

**[student@workstation ~]$ lab console-metrics finish**

This concludes the section.