Container CPU/MEMORY Usage % is too high

Created by Hernan Escalera, last modified on May 02, 2023

gi_sre

3 Comments

Hernan Escalera

Title: Container CPU Usage % is too high

Symptom:

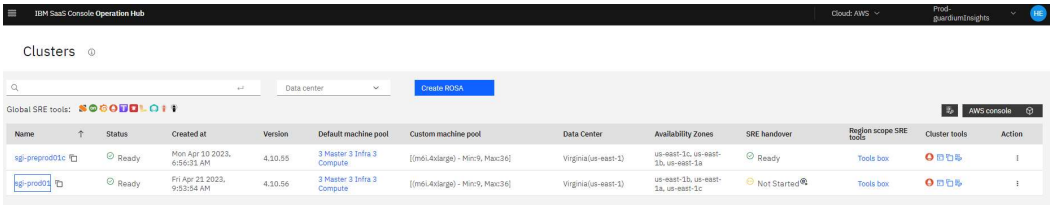
The CPU usage of a container is consistently high, which may cause performance issues or lead to resource exhaustion on the node.

Impact:

High CPU usage can cause application performance issues and may lead to reduced availability or even downtime of the application. It can also impact the performance of other applications running on the same node.

Diagnosis:

1.- Identify the affected container(s): Connect through https://sre-console.saas.ibm.com/sre-console/clusters, select the appropiate environment prod-GuardiumInsights:



2.- Go through sgi-prod01:

3.- Select IBM-ID-Login in the Red-Hat screen:

4.- Check the CPU usage of the container by running the following command: `oc exec <pod-name> -- top -bn1 | grep -E "^%?(Cpu|CPU)"` This command will show you the CPU usage of the container as a percentage.

5.- Check the CPU limits and requests set for the container by running the following command: `oc describe pod <pod-name> | grep -E "(CPU|cpu)(limits|Limits|requests|Requests)"` This command will show you the CPU limits and requests set for the container.

6.- Check the logs of the application running inside the container for any errors related to CPU usage by running the following command: `oc logs <pod-name> --container=<container-name> | grep -i cpu` This command will show you any log entries that contain the string "cpu".

Remediation:

1.- If the CPU/MEMORY usage is consistently higher than the limits and requests set for the container, consider increasing the CPU/MEMORY limits and/or requests to allocate more CPU resources to the container. To adjust the container's CPU/MEMORY limits and requests. By the following commands:

- `oc get guardiuminsights` The "get" command is used to retrieve information about resources.
- `oc edit guardiuminsights <Custome Resource>`
- Under spec section, add for example: Look for the `resources` section and adjust the `limits.cpu` and/or `requests.cpu` fields and MEMORY as needed. Save the changes and exit the editor.

```
<microservice-name>:
   resources:
     limits:
       cpu: "4"
       memory: 16Gi
     requests:
       cpu: "4"
       memory: 16Gi
```

Note: After you update the CR with the new cpu/ram values, should wait ~15/20 min, then the deployment will be updated, and the pods will be restarted on their own.

2.- If the CPU usage is consistently high but within the limits and requests set for the container, consider scaling the application horizontally by adding more replicas or deploying it on multiple nodes to distribute the load. By the following command:

NOTE: This section, scaling through the deployment is for the MCV's (Microservices-based Applications) that doesn't have an HPA enabled.

```
-oc scale deployment <deployment name> --replicas=<higher value>
```

```
or
```

```
-oc scale statefulset <statefulset name> --replicas=<higher value>
```

For the MCV's that have hHPA, need to increase the max replicas through the GI (Guardium Insights) CR (Custome Resource).

Check the HPA by running the following commands:

```
-oc get guardiuminsights
-oc describe hpa <Custom Resource>
-oc edit guardiuminsights <Custome Resource>
under spec section add:
<microservice-name>:
    hpa:
    replicaCountMax: <new max>
```

3.- If the high CPU usage is caused by a specific process or application running inside the container, consider optimizing or tuning the application to reduce its CPU usage. This may involve profiling the application, identifying the bottlenecks, and optimizing the code or configuration to reduce CPU/MEMORY usage.

Post Remediation Steps:

1.- Monitor the CPU usage of the container and the node over time to ensure that the changes you made have effectively reduced the CPU usage and that the container is not exceeding the available CPU resources on the node.

2.- If the CPU usage is still high or if the container is still experiencing performance issues, repeat the diagnosis and remediation steps as needed until the issue is resolved.

3.- If the issue persists or if you are unable to identify the cause of the high CPU usage, consider reaching out to the application developers or support teams for further assistance.

Joshua Ho

We should NOT be using "`-oc autoscale deployment <deployment-name> --min <min-replicas> --max <max-replicas> --cpu-percent <cpu-percent>`"

```
Instead use "oc scale deployment <deployment name> --replicas=<higher value>"
```

```
or
```

```
"oc scale statefulset <statefulset name> --replicas=<higher value>"
```

Hernan Escalera

Done Joshua