Troubleshoot Ops-center Pod in CrashLoopBackOff State

Contents

Introduction

This document describes how to identify and recover the ops-center pod in CrashLoopBackOff state.

Acronyms

RCM – Redundancy Configuration Manager

YYYY-MM-DD hh:mm:ss – Year-Month-Day Hour:Minute:second

CPU – Central Processing Unit

Logs Required

RCM command outputs required for troubleshooting:

```
1. kubectl get pods --namespace <namespace>
2. kubectl describe pods <podname> --namespace <namespace>
3. journalctl --since "YYYY-MM-DD hh:mm:ss" --until "YYYY-MM-DD hh:mm:ss" > /tmp/<filename>
4. kubectl --namespace rcm logs --previous <pod name> --container <container name> > /tmp/<filename>
```

Sequence to Troubleshoot

1. Verify if the affected ops-center pod is in a MASTER RCM or BACKUP RCM by executing the command in the high-availability pair:

<#root>

```
# rcm show-status
```

Example :

```
[unknown] rcm# rcm show-status
message :
{"status”: “MASTER"}
```

2. Collect the pod description of the affected op-centre pod and review the restart count and which exit codes in the containers are in a problematic state. For instance, the containers confd and confd notifications are currently in a problematic state, as indicated:

<#root>

Example:

```
rcm # kubectl describe pods ops-center-rcm-ops-center --namespace rcm
Name:          ops-center-rcm-ops-center
Namespace:     rcm
…
Containers:
  confd:
    …
```

Last State: Terminated

```
      Reason:       Error
```

Exit Code: 137

```
      Started:      Fri, 01 Dec 2023 12:44:13 +0530
      Finished:     Fri, 01 Dec 2023 12:46:09 +0530
    Ready:          False
```

Restart Count: 8097

```
  …
  confd-api-bridge:
  …
    State:          Running
      Started:      Tue, 09 May 2023 02:36:37 +0530
    Ready:          True
    Restart Count:  0
  …
  product-confd-callback:
  …
    State:          Running
      Started:      Tue, 09 May 2023 02:36:38 +0530
    Ready:          True
    Restart Count:  0
  …
  confd-notifications:
    …
    State:          Running
      Started:      Fri, 01 Dec 2023 12:46:14 +0530
```

```
Last State:     Terminated
```

```
      Reason:       Error
```

```
Exit Code:    1
```

```
      Started:      Fri, 01 Dec 2023 12:40:50 +0530
      Finished:     Fri, 01 Dec 2023 12:46:00 +0530
    Ready:          True
```

```
Restart Count:  5278
```

…

3. Examine the exit code to understand the cause of initial container restart.

Example:

Exit code 137 indicates that the containers/pod do not have sufficient memory.

Exit code 1 indicates a container shutdown due to an application error.

4. Review the **journalctl** to verify the issue timeline and understand from when the issue is observed. Logs indicating the restart of the container confd notifications, as shown here, can be used to identify the start of the issue time:

<#root>

```
Nov 29 00:00:01 <nodename> kubelet[30789]: E1129 00:00:01.993620   30789 pod_workers.go:190] "Error syn
```

```
restarting failed container=confd-notifications
```

```
 pod=ops-center-rcm-ops-center (<podUID>)\"" pod="rcm/ops-center-rcm-ops-center" podUID=<podUID>
```

5. Review the container logs of restarted containers and verify the cause for the continuous container restart loop. In this example, the container logs indicate a failure in loading the restoration configuration:

<#root>

```
Example:
```

```
rcm # kubectl --namespace rcm logs --previous ops-center-rcm-ops-center --container confd
ConfD started
Failed to connect to server
All callpoints are registered - exiting
ConfD restore
Failure loading the restore configuration
ConfD load nodes config
DEBUG Failed to connect to ConfD: Connection refused
```

```
confd_load: 290: maapi_connect(sock, addr, addrlen) failed: system call failed (24): Failed to connect
…
Failure loading the nodes config
ConfD load day-N config
Failure loading the day-N config
…
Failure in starting confd - see previous errors - killing 1

rcm # kubectl --namespace rcm logs --previous ops-center-rcm-ops-center --container confd-notifications
…
Checking that ConfD is running.
Checking that ConfD is running.
ConfD is up and running
Failed to load schemas from confd
```

Warning:

If container logs are executed with the option **--previous** on a container that has not restarted or terminated, it returns an error:

```
rcm:~# kubectl --namespace rcm logs --previous ops-center-rcm-ops-center --container confd-api-br
Error from server (BadRequest): previous terminated container "confd-api-bridge" in pod "ops-cent
```

Possible Scenarios Leading to an Issue with Subsequent Configuration Restoration

Configuration Unavailability

- The confd-api-bridge container has the function to read configuration from confd and create a backup every second. The confd-api-bridge stores it in the configmap **ops-center-confd-<opscenter-name>**.
- If the confd container is stopped and subsequently, the confd-api-bridge receives no reply for the configuration, it stores an empty configuration in the configmap.
- When the confd container attempts to restore from the backup configuration available, it fails and causes the CrashLoopBackOff state. This can be verified from the confd container logs:

```
confd_load: 660: maapi_candidate_commit_persistent(sock, NULL) failed: notset (12): /cisco-mobile-produ
```

This behavior is addressed by a Cisco bug ID CSCwi15801.

CPU Cycle Constraints

- When the confd container attempts to recover, if the startup is not completed within thirty seconds, the container is restarted.
- The startup is delayed if it does not receive the required CPU cycles due to the high CPU load on the RCM.
- If the RCM CPU continues in an occupied state due to load by other pods such as rcm-checkpointmgr, the confd container continues to restart and cause the CrashLoopBackOff state.

This behavior is addressed by a Cisco bug ID CSCwe79529.

Note:

- If the MASTER RCM is affected, perform an RCM switchover to the BACKUP RCM and then troubleshoot further. And If no BACKUP RCM is available, continue to troubleshoot the MASTER RCM.
- It is recommended to consult with Cisco TAC before performing any workarounds if an ops-center pod is observed in CrashLoopBackOff state.