

Oracle® Communications

Network Analytics Data Director

Troubleshooting Guide



Release 23.2.0

F80272-04

January 2024

ORACLE®

Copyright © 2022, 2024, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

1 Introduction

1.1	Overview	1
1.2	Audience	1
1.3	References	1

2 Troubleshooting OCNADD

2.1	Generic Checklist	1
2.2	Helm Install and Upgrade Failure	6
2.2.1	Incorrect Image Name in ocnadd-custom-values.yaml File	6
2.2.2	Docker Registry is Configured Incorrectly	6
2.2.3	Continuous Restart of Pods	7
2.2.4	Adapter Pods Does Not Receive Update Notification	7
2.2.5	Adapter Deployment Removed during Upgrade or Rollback	7
2.2.6	ocnadd-custom-values.yaml File Parse Failure	8
2.2.7	Kafka Brokers Continuously Restart after Reinstallation	9
2.2.8	Kafka Brokers Continuously Restart After the Disk is Full	9
2.2.9	Kafka Brokers Restart on Installation	10
2.2.10	Database Goes into the Deadlock State	11
2.2.11	Readiness Probe Failure	12
2.2.12	Synthetic Feed Unable to Forward Packets After Rollback to 23.1.0	12
2.2.13	Rollback in Pending State due to Database Job Failure	12
2.2.14	Upgrade to 23.2.0 Fails due to Unsupported Changes	14
2.2.15	Upgrade Fails Due to Helm Hook Upgrade Database Job	15
2.2.16	Configuration Service Displays Error Logs	15
2.2.17	Webhook Failure During Installation or Upgrade	19

3 Logs

3.1	Log Levels	1
3.2	Configuring Log Levels	1
3.3	Collecting Logs	2
3.4	Collect logs using Deployment Data Collector Tool	2

4 Capturing tcpdump from CNE

5 OCNADD Alerts

5.1	Configuring Alerts	1
5.2	Alert Forwarding Using Simple Network Management Protocol (SNMP)	1
5.3	List of Alerts	6
5.3.1	System Level Alerts	6
5.3.2	Application Level Alerts	9

My Oracle Support (MOS)

My Oracle Support (<https://support.oracle.com>) is your initial point of contact for all product support and training needs. A representative at Customer Access Support can assist you with My Oracle Support registration.

Call the Customer Access Support main number at 1-800-223-1711 (toll-free in the US), or call the Oracle Support hotline for your local country from the list at <http://www.oracle.com/us/support/contact/index.html>. When calling, make the selections in the sequence shown below on the Support telephone menu:

- For Technical issues such as creating a new Service Request (SR), select **1**.
- For Non-technical issues such as registration or assistance with My Oracle Support, select **2**.
- For Hardware, Networking and Solaris Operating System Support, select **3**.

You are connected to a live agent who can assist you with My Oracle Support registration and opening a support ticket.

My Oracle Support is available 24 hours a day, 7 days a week, 365 days a year.

Acronyms

The following table provides information about the acronyms and the terminology used in the document.

Table Acronyms

Acronym	Description
CLI	Command Line Interface
MPS	Messages Per Second
OHC	Oracle Help Center
OSDC	Oracle Service Delivery Cloud
SCP	Oracle Communications Cloud Native Core, Service Communication Proxy
NRF	Oracle Communications Cloud Native Core, Network Repository Function
SEPP	Oracle Communications Cloud Native Core, Security Edge Protection Proxy
SVC	Services
URI	Uniform Resource Identifier
KPI	Key Performance Indicator
CNE	Cloud Native Environment
MPS	Messages Per Second

What's New in This Guide

This section lists the documentation updates for Release 23.2.x in *Oracle Communications Network Analytics Data Director Troubleshooting Guide*.

Release 23.2.0.0.3 - F80272-04, January 2024

There are no updates to this document in this release.

Release 23.2.0.0.2 - F80272-03, September 2023

There are no updates to this document in this release.

Release 23.2.0.0.1 - F80272-02, July 2023

There are no updates to this document in this release.

Release 23.2.0 - F80272-01, June 2023

- Added the following topics in the Helm Install and Upgrade Failure sections:
 - [Adapter Deployment Removed during Upgrade or Rollback](#)
 - [Upgrade to 23.2.0 Fails due to Unsupported Changes](#)
 - [Synthetic Feed Unable to Forward Packets After Rollback to 23.1.0](#)
 - [Upgrade Fails Due to Helm Hook Upgrade Database Job](#)
 - [Rollback in Pending State due to Database Job Failure](#)
 - [Configuration Service Displays Error Logs](#)
 - [Webhook Failure During Installation or Upgrade](#)
- Removed the topic "Helm Chart Duplicate Parameter Issues" in the "Helm Install and Upgrade Failure" section.
- Updated the following alerts:
 - [System Level Alerts](#)
 - [Application Level Alerts](#)

1

Introduction

This document provides Oracle Communications Network Analytics Data Director (OCNADD) troubleshooting information.

1.1 Overview

Oracle Communications Network Analytics Data Director (OCNADD) is a specialized Network Data Broker (NDB) that receives network data from various data sources (such as 5G NFs and Non-5G NFs) and sends the data securely to the subscribed consumers (such as third-party tools) after applying mechanisms such as data filtering, data replication, and data aggregation. All these mechanisms are configurable by the consumers.

OCNADD provides curated data (either filtered or replicated) for network analytics and monitoring. OCNADD supports robust, configurable filtering and aggregation options which enables the operator to sort data, create comprehensive dashboards, and generate Key Performance Indicators (KPIs) for all departments within the service provider framework. OCNADD also provides a GUI that enables users to create, edit, and delete data feed.

For more information about OCNADD architecture and features, see *Oracle Communications Network Analytics Data Director User Guide*.

1.2 Audience

The intended audiences for this document are the network administrators and the professionals responsible for OCNADD deployment and maintenance.

1.3 References

For more information about OCNADD, refer to the following documents:

- *Oracle Communications Network Analytics Data Director User Guide*
- *Oracle Communications Network Analytics Data Director Installation, Upgrade, and Fault Recovery Guide*
- *Oracle Communications Network Analytics Data Director Benchmarking Guide*
- *Oracle Communications Cloud Native Environment Installation, Upgrade, and Fault Recovery Guide*
- *Oracle Communications Cloud Native Core DBTier Installation, Upgrade, and Fault Recovery Guide*
- *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide*
- *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide*

2

Troubleshooting OCNADD

This chapter provides information to troubleshoot the common errors, which can be encountered during the preinstallation, installation, and upgrade procedures of OCNADD.

Note

kubectl commands might vary based on the platform deployment. Replace kubectl with Kubernetes environment-specific command line tool to configure Kubernetes resources through kube-api server. The instructions provided in this document are as per the Oracle Communications Cloud Native Environment (OCCNE) version of kube-api server.

2.1 Generic Checklist

The following sections provide a generic checklist for troubleshooting OCNADD.

Deployment Checklist

Perform the following pre-deployment checks:

- Failure in Certificate or Secret generation.
There may be a possibility of an error in certificate generation when the Country, State, or Organization name is different in CA and service certificates.

Error Code/Error Message:

The countryName field is different between
CA certificate (US) and the request (IN)

(similar error message will be reported for State or Org name)

To resolve this error:

1. Navigate to "ssl_certs/default_values/" and edit the "values" file.
2. Change the following values under "[global]" section:
 - countryName
 - stateOrProvinceName
 - organizationName
3. Ensure the values match the CA configurations, for example:
If the CA has country name as "US", state as "NY", and Org name as "ORACLE" then, set the values under [global] parameter as follows:

[global]

```
countryName=US
stateOrProvinceName=NY
localityName=BLR
organizationName=ORACLE
organizationalUnitName=CGBU
defaultDays=365
```

4. Rerun the script and verify the certificate and secret generation.

- Run the following command to verify if OCNADD deployment, pods, and services created are running and available:

```
# kubectl -n <namespace> get deployments,pods,svc
```

Verify the output, and check the following columns:

- READY, STATUS, and RESTARTS
- PORT(S) of service

Note

It is normal to observe the Kafka broker restart during deployment.

- Verify if the correct image is used and correct environment variables are set in the deployment.

To check, run the following command:

```
# kubectl -n <namespace> get deployment <deployment-name> -o yaml
```

- Check if the microservices can access each other through a REST interface.

To check, run the following command:

```
# kubectl -n <namespace> exec <pod name> -- curl <uri>
```

Example:

```
kubectl exec -it pod/ocnaddconfiguration-6ffc75f956-wnvzx -n ocnadd-
system -- curl 'http://ocnaddadminservice:9181/ocnadd-admin-svc/v1/topic'
```

Note

These commands are in their simple format and display the logs only if ocnaddconfiguration and ocnadd-admin-svc pods are deployed.

The list of URIs for all the microservices:

- `http://ocnaddconfiguration:<port>/ocnadd-configuration/v1/subscription`
- `http://ocnaddalarm:<port>/ocnadd-alarm/v1/alarm?&startTime=<start-time>&endTime=<end-time>`
Use off-set date time format, for example, 2022-07-12T05:37:26.954477600Z
- `<ip>:<port>/ocnadd-admin-svc/v1/topic/`

- `<ip>:<port>/ocnadd-admin-svc/v1/describe/topic/<topicName>`
- `<ip>:<port>/ocnadd-admin-svc/v1/alter`
- `<ip>:<port>/ocnadd-admin-svc/v1/broker/expand/entry`
- `<ip>:<port>/ocnadd-admin-svc/v1/broker/health`

Application Checklist

Logs Verification

Run the following command to check the application logs and look for exceptions:

```
# kubectl -n <namespace> logs -f <pod name>
```

Use the option `-f` to follow the logs or `grep` option to obtain a specific pattern in the log output.

Example:

```
# kubectl -n ocnadd-system logs -f $(kubectl -n ocnadd-system get pods -o name | cut -d '/' -f 2 | grep nrfaggregation)
```

Above command displays the logs of the `ocnaddnrfaggregation` service.

Run the following command to search for a specific pattern in the log output:

```
# kubectl logs -n ocnadd-system <pod name> | grep <pattern>
```

① Note

These commands are in their simple format and display the logs only if there is atleast one `nrfaggregation` pod deployed.

Kafka Consumer Rebalancing

The Kafka consumers can rebalance in the following scenarios:

- The number of partitions changes for any of the subscribed topics.
- A subscribed topic is created or deleted.
- An existing member of the consumer group is shutdown or fails.
- In the Kafka consumer application,
 1. Stream threads inside the consumer app skipped sending heartbeat to Kafka.
 2. The batch of messages took longer time to process and causes the time between the two polls to take longer.
 3. Any stream thread in any of the consumer application pods dies because of some error and it is replaced with a new Kafka Stream thread.
 4. Any stream thread is stuck and not processing any message.
- A new member is added to the consumer group (for example, new consumer pod spins up).

When the rebalancing is triggered, there is a possibility that offsets are not committed by the consumer threads as offsets are committed periodically. This can result in messages

corresponding to non-committed offsets being sent again or duplicated when the rebalancing is completed and consumers started consuming again from the partitions. This is a normal behavior in the Kafka consumer application. However, because of frequent rebalancing in the Kafka consumer applications, the counts of messages in the Kafka consumer application and 3rd party application can mismatch.

Data Feed not accepting updated endpoint

Problem

If a Data feed is created for synthetic packets with an incorrect endpoint, updating the endpoint afterward has no effect.

Solution

Delete and recreate the data feed for synthetic packets with the correct endpoint.

Kafka Performance Impact (due to disk limitation)

Problem

When source topics (SCP, NRF, and SEPP) and MAIN topic are created with Replication Factor = 1

For a low performance disk, the Egress MPS rate drops/fluctuates with the following traces in the Kafka broker logs:

```
Shrinking ISR from 1001,1003,1002 to 1001. Leader: (highWatermark: 1326,
endOffset: 1327). Out of sync replicas: (brokerId: 1003, endOffset: 1326)
(brokerId: 1002, endOffset: 1326). (kafka.cluster.Partition)
ISR updated to 1001,1003 and version updated to 28(kafka.cluster.Partition)
```

Solution

The following steps can be performed (or verified) to optimize the Egress MPS rate:

1. Try to increase the disk performance in the cluster where OCNADD is deployed.
2. If the disk performance cannot be increased, then perform the following steps for OCNADD:
 - a. Navigate to the Kafka helm charts values file (<helm-chart-path>/ocnadd/charts/ocnaddkafka/values.yaml)
 - b. Change the below parameter in the values.yaml:
 - i. offsetsTopicReplicationFactor: 1
 - ii. transactionStateLogReplicationFactor: 1
 - c. Scale down the Kafka and zookeeper deployment by modifying the following lines in the helm chart:

```
ocnaddkafka:
  enabled:false
```

- d. Perform helm upgrade for OCNADD:


```
helm upgrade <release name> <chart path> -n <namespace>
```
- e. Delete PVC for Kafka and Zookeeper using the following commands:

- i. `kubectl delete pvc -n <namespace> kafka-volume-kafka-broker-0`
- ii. `kubectl delete pvc -n <namespace> kafka-volume-kafka-broker-1`
- iii. `kubectl delete pvc -n <namespace> kafka-volume-kafka-broker-2`
- iv. `kubectl delete pvc -n <namespace> kafka-broker-security-zookeeper-0`
- v. `kubectl delete pvc -n <namespace> kafka-broker-security-zookeeper-1`
- vi. `kubectl delete pvc -n <namespace> kafka-broker-security-zookeeper-2`
- f. Modify the value of the following parameter to true:

```
ocnaddkafka:  
enabled:true
```

- g. Perform helm upgrade for OCNADD:
`helm upgrade <release name> <chart path> -n <namespace>`

Note

The following points are to be considered while applying the above procedure:

- 1. In case a Kafka broker becomes unavailable, then you may experience an impact on the traffic on the Ingress side.
- 2. Verify the Kafka broker logs or describe the Kafka/zookeeper pod which is unavailable and take the necessary action based on the error reported.

500 Server Error on GUI while creating/deleting the Data Feed

Problem

Occasionally, due to network issues, the user may observe a "500 Server Error" while creating/deleting the Data Feed.

Solution

The following actions generally resolve the issue:

- Delete and recreate the feed if it is not created properly.
- Retry the action after logging out from the GUI and login back again.
- Retry creating/deleting the feed after some time.

Kafka resources reaching more than 90% utilization

Problem

Kafka resources(CPU, Memory) reached more than 90% utilization due to a higher MPS rate or slow disk I/O rate

Solution

Add additional resources to the following parameters that are reaching high utilization.

File name: `ocnadd-custom-values.yaml`

Parameter name: `ocnaddkafka.ocnadd.kafkaBroker.resource`

```
kafkaBroker:
  name:kafka-broker
  resource:
    limit:
      cpu:5    ==> change it to require number of CPUs
      memory:24Gi ==> change it to require number of memory size
```

2.2 Helm Install and Upgrade Failure

This section describes the various helm installation or upgrade failure scenarios and the respective troubleshooting procedures:

2.2.1 Incorrect Image Name in *ocnadd-custom-values.yaml* File

Problem

`helm install` fails if an incorrect image name is provided in the `ocnadd-custom-values.yaml` file or if the image is missing in the image repository.

Error Code or Error Message

When you run `kubectl get pods -n <ocnadd_namespace>`, the status of the pods might be `ImagePullBackOff` or `ErrImagePull`.

Solution

Perform the following steps to verify and correct the image name:

1. Edit the `ocnadd-custom-values.yaml` file and provide the release specific image names and tags.
2. Run the `helm install` command.
3. Run the `kubectl get pods -n <ocnadd_namespace>` command to verify if all the pods are in **Running** state.

2.2.2 Docker Registry is Configured Incorrectly

Problem

`helm install` might fail if the Docker Registry is not configured in all primary and secondary nodes.

Error Code or Error Message

When you run `kubectl get pods -n <ocnadd_namespace>`, the status of the pods might be `ImagePullBackOff` or `ErrImagePull`.

Solution

Configure the Docker Registry on all primary and secondary nodes. For information about Docker Registry configuration, see *Oracle Communications Cloud Native Environment Installation, Upgrade, and Fault Recovery Guide*.

2.2.3 Continuous Restart of Pods

Problem

`helm install` might fail if MySQL primary or secondary hosts are not configured properly in `ocnadd-custom-values.yaml`.

Error Code or Error Message

When you run `kubectl get pods -n <ocnadd_namespace>`, the pods shows restart count increases continuously, or there is a Prometheus alert for continuous pod restart.

Solution

1. Verify MySQL connectivity.

MySQL servers may not be configured properly. For more information about the MySQL configuration, see *Oracle Communications Network Analytics Data Director Installation, Upgrade, and Fault Recovery Guide*.

2. Describe the POD to check more details on the error, troubleshoot further based on the reported error.
3. Check the POD log for any error, troubleshoot further based on the reported error.

2.2.4 Adapter Pods Does Not Receive Update Notification

Problem

Update notification from Configuration Service or GUI is not reaching Adapter pods.

Error Code or Error Message

On triggering update notification from GUI or Configuration Service, the configuration changes on Adapter does not take place.

Solution

This can be fixed by running the following command:

- Check the Adapter pod logs with:

```
"kubectl logs <adapter app name> -n <namespace>
```

- If the update notification logs are not present in the Adapter logs run the below command to reread the configurations with the following command:

```
kubectl rollout restart deploy <adapter app name> -n <namespace>
```

2.2.5 Adapter Deployment Removed during Upgrade or Rollback

Problem

Adapter(data feed) is deleted during upgrade or rollback.

Error Code or Error Message

Sample error message displayed:

Figure 2-1 Error Message

```
OCL 2023-04-26 14:07:11.421 [boundedElastic-3] ERROR com.oracle.cgbu.cne.ocnadd.admin.svc.service.consumerAdapter.ConsumerAdapterService - exception in deploying
io.fabric8.kubernetes.client.KubernetesClientException: Operation: [list] for kind: [Deployment] with name: [null] in namespace: [ocnadd-sg] failed.
  at io.fabric8.kubernetes.client.KubernetesClientException.lambda$throwable(KubernetesClientException.java:159) ~[kubernetes-client-api-6.3.1.jar!/:?]
  at io.fabric8.kubernetes.client.dsl.internal.BaseOperation.list(BaseOperation.java:420) ~[kubernetes-client-6.3.1.jar!/:?]
  at io.fabric8.kubernetes.client.dsl.internal.BaseOperation.list(BaseOperation.java:383) ~[kubernetes-client-6.3.1.jar!/:?]
  at io.fabric8.kubernetes.client.dsl.internal.BaseOperation.list(BaseOperation.java:93) ~[kubernetes-client-6.3.1.jar!/:?]
  at com.oracle.cgbu.cne.ocnadd.admin.svc.service.consumerAdapter.ConsumerAdapterService.deployConsumerAdapter(ConsumerAdapterService.java:677) ~[classes!/:1.0.72]
  at com.oracle.cgbu.cne.ocnadd.admin.svc.controller.AdminControllerImpl.lambda$createConsumerGroup$0(AdminControllerImpl.java:352) ~[classes!/:1.0.72]
  at reactor.core.publisher.MonoCallable.call(MonoCallable.java:92) ~[reactor-core-3.4.26.jar!/:3.4.26]
  at reactor.core.publisher.FluxSubscribeOnCallable$CallableSubscribeOnSubscription.run(FluxSubscribeOnCallable.java:227) ~[reactor-core-3.4.26.jar!/:3.4.26]
  at reactor.core.scheduler.SchedulerTask.call(SchedulerTask.java:68) ~[reactor-core-3.4.26.jar!/:3.4.26]
  at reactor.core.scheduler.SchedulerTask.call(SchedulerTask.java:28) ~[reactor-core-3.4.26.jar!/:3.4.26]
  at java.util.concurrent.FutureTask.run(FutureTask.java:264) ~[?:?]
  at java.util.concurrent.ScheduledThreadPoolExecutor$ScheduledFutureTask.run(ScheduledThreadPoolExecutor.java:304) ~[?:?]
  at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1136) ~[?:?]
  at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:635) ~[?:?]
  at java.lang.Thread.run(Thread.java:833) ~[?:?]
Caused by: java.io.InterruptedIOException
  at io.fabric8.kubernetes.client.dsl.internal.OperationSupport.waitForResult(OperationSupport.java:523) ~[kubernetes-client-6.3.1.jar!/:?]
  at io.fabric8.kubernetes.client.dsl.internal.BaseOperation.list(BaseOperation.java:418) ~[kubernetes-client-6.3.1.jar!/:?]
  ... 13 more
```

Solution

This can be fixed by running the following commands:

- Run the following command to verify the data feeds:

```
kubectl get po -n <namespace>
```

- If data feeds are missing, verify the above mentioned error message in the admin service log, by running the following command:

```
kubectl logs <admin svc pod name> -n <namespace>
```

- If the error message is present, run the following command:

```
kubectl rollout restart deploy <configuration app name> -n <namespace>
```

2.2.6 ocnadd-custom-values.yaml File Parse Failure

This section explains the troubleshooting procedure in case of failure while parsing the ocnadd-custom-values.yaml file.

Problem

Unable to parse the ocnadd-custom-values.yaml file or any other values.yaml while running Helm install.

Error Code or Error Message

Error: failed to parse ocnadd-custom-values.yaml: error converting YAML to JSON: yaml

Symptom

When parsing the ocnadd-custom-values.yaml file, if the above mentioned error is received, it indicates that the file is not parsed because of the following reasons:

- The tree structure may not have been followed.
- There may be a tab space in the file.

Solution

Download the latest OCNADD custom templates zip file from MoS. For more information, see *Oracle Communications Network Analytics Data Director Installation, Upgrade, and Fault Recovery Guide*.

2.2.7 Kafka Brokers Continuously Restart after Reinstallation

Problem

When re-installing OCNADD in the same namespace without deleting the PVC that was used for the first installation, Kafka brokers will go into crashloopbackoff status and keep restarting.

Error Code or Error Message

When you run, `kubectl get pods -n <ocnadd_namespace>` the broker pod's status might be Error/crashloopbackoff and it might keep restarting continuously, with "no disk space left on the device" errors in the pod logs.

Solution

1. Delete the Stateful set (STS) deployments of the brokers. Run `kubectl get sts -n <ocnadd_namespace>` to obtain the Stateful sets in the namespace.
2. Delete the STS deployments of the services with disk full issue. For example run the command `kubectl delete sts -n <ocnadd_namespace> kafka-broker1 kafka-broker2`.
3. Delete the PVCs in the namespace, which is used by kafka-brokers. Run `kubectl get pvc -n <ocnadd_namespace>` to get the PVCs in that namespace. The number of PVCs used is based on the number of brokers deployed. Therefore, select the PVCs that have the name kafka-broker or zookeeper, and delete them. To delete the PVCs, run `kubectl delete pvc -n <ocnadd_namespace> <pvcname1> <pvcname2>`.

For example:

For a three broker setup in the namespace ocnadd-deploy, delete the following PVCs:

```
kubectl delete pvc -n ocnadd-deploy broker1-pvc-kafka-broker1-0, broker2-pvc-kafka-broker2-0, broker3-pvc-kafka-broker3-0, kafka-broker-security-zookeeper-0, kafka-broker-security-zookeeper-1 kafka-broker-security-zookeeper-2
```

2.2.8 Kafka Brokers Continuously Restart After the Disk is Full

Problem

This issue occurs when the disk space is full on the broker or zookeeper.

Error Code or Error Message

When you run `kubectl get pods -n <ocnadd_namespace>`, the broker pod's status might be error, or crashloopbackoff and it might keep restarting continuously.

Solution

1. Delete the STS(stateful set) deployments of the brokers:

- a. Get the STS in the namespace with the following command:

```
kubectl get sts -n <ocnadd_namespace>
```

- b. Delete the STS deployments of the services with disk full issue:

```
kubectl delete sts -n <ocnadd_namespace> <sts1> <sts2>
```

For example, for three broker setup:

```
kubectl delete sts -n ocnadd-deploy kafka-broker1 kafka-broker2 kafka-broker3 zookeeper
```

2. Delete the PVCs in that namespace that is used by the removed kafka-brokers. To get the PVCs in that namespace:

```
kubectl get pvc -n <ocnadd_namespace>
```

The number of PVCs used will be based on the number of brokers you deploy. Select the PVCs that have the name kafka-broker or zookeeper and delete them.

- a. To delete PVCs, run:

```
kubectl delete pvc -n <ocnadd_namespace> <pvcname1> <pvcname2>
```

For example, for a three broker setup in namespace ocnadd-deploy, you must delete these PVCs;

```
kubectl delete pvc -n ocnadd-deploy broker1-pvc-kafka-broker1-0 broker2-pvc-kafka-broker2-0 broker3-pvc-kafka-broker3-0 kafka-broker-security-zookeeper-0 kafka-broker-security-zookeeper-1 kafka-broker-security-zookeeper-2
```

3. Once the STS and PVC's are deleted for the services, edit the respective broker's values.yaml to increase the PV size of the brokers at the location: <chartpath>/charts/ocnaddkafka/values.yaml.

If any formatting or indentation issues occur while editing, refer to the files in

```
<chartpath>/charts/ocnaddkafka/default
```

To increase the storage, edit the fields pvcClaimSize for each broker. For recommendation of PVC storage, see *Oracle Communications Network Analytics Data Director Benchmarking Guide*.

4. Upgrade the Helm chart after increasing the PV size

```
helm upgrade <release-name> <chartpath> -n <namespace>
```

5. Create the required topics.

2.2.9 Kafka Brokers Restart on Installation

Problem

Kafka brokers re-start during OCNADD installation.

Error Code or Error Message

The output of the command `kubectl get pods -n <ocnadd_namespace>` displays the broker pod's status as restarted.

Solution

The Kafka Brokers wait for a maximum of 3 minutes for the Zookeepers to come online before they are started. If the Zookeeper cluster does not come online within the given interval, the broker will start before the Zookeeper and will error out as it does not have access to the Zookeeper. This may Zookeeper may start after the 3 interval as the node may take more time to pull the images due to network issues. Therefore, when the zookeeper does not come online within the given time this issue may be observed.

2.2.10 Database Goes into the Deadlock State

Problem

MySQL locks get struck.

Error Code or Error Message

ERROR 1213 (40001): Deadlock found when trying to get lock; try restarting the transaction.

Symptom

Unable to access MySQL.

Solution

Perform the following steps to remove the deadlock:

1. Run the following command on each SQL node:

```
SELECT
CONCAT('KILL ', id, ';')
FROM INFORMATION_SCHEMA.PROCESSLIST
WHERE `User` = <DbUsername>
AND `db` = <DbName>;
```

This command retrieves the list of commands to kill each connection.

Example:

```
select
  CONCAT('KILL ', id, ';')
FROM INFORMATION_SCHEMA.PROCESSLIST
where `User` = 'ocnadduser'
      AND `db` = 'ocnadddb';
+-----+
| CONCAT('KILL ', id, ';') |
+-----+
| KILL 204491;              |
| KILL 200332;              |
| KILL 202845;              |
+-----+
3 rows in set (0.00 sec)
```

2. Run the kill command on each SQL node.

2.2.11 Readiness Probe Failure

The `helm install` might fail due to the readiness probe URL failure.

If this error appears, check for the readiness probe URL correctness in the particular microservice Helm charts under the charts folder.

Low Resources

`helm install` might fail due to low resources in the cluster.

In this case, check the CPU and memory availability in the Kubernetes cluster.

2.2.12 Synthetic Feed Unable to Forward Packets After Rollback to 23.1.0

Scenario

- Synthetic data feeds are created in release 23.1.0 and verified with end-to-end traffic flow.
- User performs an upgrade to release 23.2.0. Synthetic data feeds created in release 23.1.0 are working as expected.
- User has to perform a rollback to release 23.1.0 due to some unrecoverable error.
- The thread configuration of the TCP synthetic data feed is modified resulting in traffic failure to the third party consumers.

Error Code or Error Message

The following error message is observed in the synthetic adapter logs:

Figure 2-2 Error Message

```
OCL 2023-05-05 11:35:32.184 [Connection-Recreation] INFO
com.oracle.cgbu.cne.ocnadd.ConsumerAdapter.service.TlsServiceImpl -
Connection: ChannelOperations{PooledConnection{channel=[id: 0xcb310f49,
L:/10.233.100.90:60172 -
R:ocnaddthirdpartyconsumeroracle2.upg.svc.pv-11.lab.in.oracle.com/10.233.23.14
5:18990]}} created for ocnaddthirdpartyconsumeroracle2:18990
Exception in thread "Connection-Recreation" java.lang.IllegalStateException:
Deque full
```

Solution

1. Delete the existing synthetic data feed.
2. Create new synthetic feed with the same name and attributes, "Connection Failure setting" set to "Resume from point of failure".
3. Verify the end-to-end traffic flow between OCNADD and the corresponding third party application.

2.2.13 Rollback in Pending State due to Database Job Failure

Problem

After two successful upgrades (23.1.0 to 23.2.0 and 23.2.0 to 23.2.0) and one successful rollback (23.2.0 to 23.2.0), the second rollback is failing due to DB job failure. The rollback enters a pending rollback state.

Error Code or Error Message

The following error message is displayed:

Figure 2-3 Error Message

```
> helm history ocnadd -n <namespace>
```

REVISION	UPDATED	STATUS	CHART	APP VERSION	DESCRIPTION
1	Mon May 15 08:06:09 2023	superseded	ocnadd-0.0.0-23.1.0	23.1.0	Install complete
2	Mon May 15 08:08:19 2023	superseded	ocnadd-0.0.0-23.2.0-rc.10	23.2.0-rc.10	Upgrade complete
3	Mon May 15 08:13:54 2023	superseded	ocnadd-0.0.0-23.2.0-rc.10	23.2.0-rc.10	Upgrade complete
4	Mon May 15 08:16:10 2023	deployed	ocnadd-0.0.0-23.2.0-rc.10	23.2.0-rc.10	Rollback to 2

```
> helm rollback ocnadd 1 -n <namespace>
Error: job failed: BackoffLimitExceeded

> helm history ocnadd -n <namespace>
```

REVISION	UPDATED	STATUS	CHART	APP VERSION	DESCRIPTION
1	Mon May 15 08:06:09 2023	superseded	ocnadd-0.0.0-23.1.0	23.1.0	Install complete
2	Mon May 15 08:08:19 2023	superseded	ocnadd-0.0.0-23.2.0-rc.10	23.2.0-rc.10	Upgrade complete
3	Mon May 15 08:13:54 2023	superseded	ocnadd-0.0.0-23.2.0-rc.10	23.2.0-rc.10	Upgrade complete
4	Mon May 15 08:16:10 2023	deployed	ocnadd-0.0.0-23.2.0-rc.10	23.2.0-rc.10	Rollback to 2
5	Mon May 15 08:18:14 2023	pending-rollback	ocnadd-0.0.0-23.1.0	23.1.0	Rollback to 1

Note

It is observed that a rollback from revision 5 to revision 1 is successful, but the Zookeeper and Kafka enter *CrashLoopBackOff* state.

Solution

- Do not perform further rollbacks if any rollback procedure is in "pending-rollback" state.
- Run the following command to list the secrets and find the pending rollback secret:

```
> kubectl get secrets -n <namespace>
```

Sample output:

```
sh.helm.release.v1.ocnadd.v1 helm.sh/release.v1 1 13m
sh.helm.release.v1.ocnadd.v2 helm.sh/release.v1 1 11m
sh.helm.release.v1.ocnadd.v3 helm.sh/release.v1 1 5m33s
sh.helm.release.v1.ocnadd.v4 helm.sh/release.v1 1 3m18s
sh.helm.release.v1.ocnadd.v5 helm.sh/release.v1 1 74s
```

- Delete the pending-rollback secret, run the following command:

```
> kubectl delete <release revision secrets> -n <namespace>
```

Example:

```
kubectl delete secrets sh.helm.release.v1.ocnadd.v5 -n dd-vin
```

4. Restore the database backup taken prior to upgrade. For more information, see the procedure to "Create OCNADD Restore Job" in the *Oracle Communications Network Analytics Data Director Installation, Upgrade, and Fault Recovery Guide*.
5. Perform a rollback to 23.1.0 by running the following command:

```
> helm rollback <release name> <revision number> -n <namespace> --no-hooks
```

Example:

```
helm rollback ocnadd 1 -n dd-vin --no-hooks
```

6. Verify if the end-to-end traffic is running between the OCNADD and the corresponding third party application.

2.2.14 Upgrade to 23.2.0 Fails due to Unsupported Changes

Problem

Upgrade from 23.1.0 to 23.2.0 fails due to unsupported changes in 23.2.0 (for example, when upgrade database job is successful, but upgrade fails due to an error).

Note

This issue is not common. Occasionally, customers are unable to sync up target charts. In such cases, adopt the following procedure as a workaround.

Error Message

If there is a PVC size mismatch between 23.1.0 and 23.2.0, the following error message is displayed:

Figure 2-4 Error Message

Error: UPGRADE FAILED: cannot patch "zookeeper" with kind StatefulSet: StatefulSet.apps "zookeeper" is invalid: spec: Forbidden: updates to statefulset spec for fields other than 'replicas', 'template', 'updateStrategy', 'persistentVolumeClaimRetentionPolicy' and 'minReadySeconds' are forbidden

Run the following command:

```
-> helm history <release-name> -n <namespace>
```

Sample output:

Figure 2-5 Sample Output

REVISION	UPDATED	STATUS	CHART	APP VERSION	DESCRIPTION
1	Mon May 1 07:08:29 2023	deployed	ocnadd-0.0.0-23.1.0	23.1.0	Install complete
2	Mon May 1 07:24:08 2023	failed	ocnadd-0.0.0-23.2.0	23.2.0	Upgrade "ocnadd" failed: cannot patch "zookeeper" with kind StatefulSet: StatefulSet.apps "zookeeper" is invalid: spec: Forbidden: updates to statefulset spec for fields other than 'replicas', 'template', 'updateStrategy', 'persistentVolumeClaimRetentionPolicy' and 'minReadySeconds' are forbidden

Solution

Perform the following steps:

1. Correct the Helm charts, synchronize the 23.2.0 Helm charts with the 23.1.0 Helm charts.
2. Perform an upgrade. For more information about the upgrade procedure, see "Upgrading OCNADD" in the *Oracle Communications Network Analytics Data Director Installation, Upgrade, and Fault Recovery Guide*.

2.2.15 Upgrade Fails Due to Helm Hook Upgrade Database Job

Problem

Upgrade from 23.1.0 to 23.2.0 fails due to Helm hook upgrade database job in 23.2.0 (upgrade job fails).

Error Code or Error Message

Figure 2-6 Error Message

```
Error: UPGRADE FAILED: pre-upgrade hooks failed: job failed: BackoffLimitExceeded
```

Run the following command:

```
-> helm history <release-name> -n <namespace>
```

Sample Output:

```
REVISION UPDATED STATUS CHART APP VERSION DESCRIPTION 1 Mon May 1 07:08:29 2023  
deployed ocnadd-0.0.0-23.1.0 23.1.0 Install complete 2 Mon May 1 07:24:08 2023  
failed ocnadd-0.0.0-23.2.0 23.2.0 Upgrade "ocnadd" failed: pre-upgrade hooks  
failed: job failed: BackoffLimitExceeded
```

Solution

Rollback to 23.1.0, correct the error and run upgrade again.

1. Run the Helm rollback command:

```
helm rollback <helm release name> <revision number> -n <namespace> --no-  
hooks
```

2. Restore the Database backup taken before upgrade. For more information see, the procedure "Create OCNADD Restore Job" in the "Fault Recovery" section in the *Oracle Communications Network Analytics Data Director Installation, Upgrade, and Fault Recovery Guide*.
3. Correct the upgrade issue and run a fresh upgrade. For more information, see "Upgrading OCNADD" in the *Oracle Communications Network Analytics Data Director Installation, Upgrade, and Fault Recovery Guide*.

2.2.16 Configuration Service Displays Error Logs

Problem

On restarting ocnaddconfiguration and adapter or aggregation pods, notifications are reaching the adapter or aggregation pods, but error logs present in Configuration Service. This is observed if the old pods IP entry is not deleted from the database.

① Note

Ensure that configuration and aggregation or adapter pods are not deleted simultaneously.

Error Code or Error Message

Configuration Service Error Log

```
Error has been observed at the following site(s): *__checkpoint ? Request to POST
http://10.233.122.199:9182/ocnadd-consumeradapter/v2/notifications
[DefaultWebClient] Original Stack Trace: at
org.springframework.web.reactive.function.client.ExchangeFunctions$DefaultExchangeFunction.lambda$wrapException$9(ExchangeFunctions.java:136) ~[spring-webflux-6.0.7.jar!/:6.0.7] at
reactor.core.publisher.MonoErrorSupplied.subscribe(MonoErrorSupplied.java:55)
~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.Mono.subscribe(Mono.java:4485) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxOnErrorResume$ResumeSubscriber.onError(FluxOnErrorResume.java:103) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxPeek$PeekSubscriber.onError(FluxPeek.java:222)
~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxPeek$PeekSubscriber.onError(FluxPeek.java:222)
~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxPeek$PeekSubscriber.onError(FluxPeek.java:222)
~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.MonoNext$NextSubscriber.onError(MonoNext.java:93)
~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.MonoFlatMapMany$FlatMapManyMain.onError(MonoFlatMapMany.java:204) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.SerializedSubscriber.onError(SerializedSubscriber.java:124)
~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxRetryWhen$RetryWhenMainSubscriber.whenError(FluxRetryWhen.java:225) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxRetryWhen$RetryWhenOtherSubscriber.onError(FluxRetryWhen.java:274) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxContextWrite$ContextWriteSubscriber.onError(FluxContextWrite.java:121) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxConcatMapNoPrefetch$FluxConcatMapNoPrefetchSubscriber.maybeOnError(FluxConcatMapNoPrefetch.java:326) ~[reactor-core-3.5.4.jar!/:3.5.4]
at
reactor.core.publisher.FluxConcatMapNoPrefetch$FluxConcatMapNoPrefetchSubscriber.onNext(FluxConcatMapNoPrefetch.java:211) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxContextWrite$ContextWriteSubscriber.onNext(FluxContextWrite.java:107) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.SinkManyEmitterProcessor.drain(SinkManyEmitterProcessor.java:471) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.SinkManyEmitterProcessor$EmitterInner.drainParent(SinkManyEmitterProcessor.java:615) ~[reactor-core-3.5.4.jar!/:3.5.4] at
```

```

reactor.core.publisher.FluxPublish$PubSubInner.request(FluxPublish.java:602)
~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxContextWrite$ContextWriteSubscriber.request(FluxContextWrite.java:136) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxConcatMapNoPrefetch$FluxConcatMapNoPrefetchSubscriber.request(FluxConcatMapNoPrefetch.java:336) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxContextWrite$ContextWriteSubscriber.request(FluxContextWrite.java:136) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.Operators$DeferredSubscription.request(Operators.java:1717) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxRetryWhen$RetryWhenMainSubscriber.onError(FluxRetryWhen.java:192) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.MonoCreate$DefaultMonoSink.error(MonoCreate.java:201) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.netty.http.client.HttpClientConnect$MonoHttpConnect$ClientTransportSubscriber.onError(HttpClientConnect.java:311) ~[reactor-netty-http-1.1.5.jar!/:1.1.5] at
reactor.core.publisher.MonoCreate$DefaultMonoSink.error(MonoCreate.java:201) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.netty.http.client.Http2ConnectionProvider$DisposableAcquire.onError(Http2ConnectionProvider.java:281) ~[reactor-netty-http-1.1.5.jar!/:1.1.5] at
reactor.netty.http.client.Http2Pool$Borrower.fail(Http2Pool.java:813) ~[reactor-netty-http-1.1.5.jar!/:1.1.5] at
reactor.netty.http.client.Http2Pool.lambda$drainLoop$2(Http2Pool.java:443) ~[reactor-netty-http-1.1.5.jar!/:1.1.5] at
reactor.core.publisher.FluxDoOnEach$DoOnEachSubscriber.onError(FluxDoOnEach.java:186) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxMap$MapConditionalSubscriber.onError(FluxMap.java:265) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.MonoCreate$DefaultMonoSink.error(MonoCreate.java:201) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.netty.resources.DefaultPooledConnectionProvider$DisposableAcquire.onError(DefaultPooledConnectionProvider.java:162) ~[reactor-netty-core-1.1.5.jar!/:1.1.5] at
reactor.netty.internal.shaded.reactor.pool.AbstractPool$Borrower.fail(AbstractPool.java:475) ~[reactor-netty-core-1.1.5.jar!/:1.1.5] at
reactor.netty.internal.shaded.reactor.pool.SimpleDequePool.lambda$drainLoop$9(SimpleDequePool.java:429) ~[reactor-netty-core-1.1.5.jar!/:1.1.5] at
reactor.core.publisher.FluxDoOnEach$DoOnEachSubscriber.onError(FluxDoOnEach.java:186) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.MonoCreate$DefaultMonoSink.error(MonoCreate.java:201) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.netty.resources.DefaultPooledConnectionProvider$PooledConnectionAllocator$PooledConnectionInitializer.onError(DefaultPooledConnectionProvider.java:560) ~[reactor-netty-core-1.1.5.jar!/:1.1.5] at
reactor.core.publisher.MonoFlatMap$FlatMapMain.secondError(MonoFlatMap.java:241) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.MonoFlatMap$FlatMapInner.onError(MonoFlatMap.java:315) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxOnErrorResume$ResumeSubscriber.onError(FluxOnErrorResume.java:106) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.Operators.error(Operators.java:198) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.MonoError.subscribe(MonoError.java:53) ~[reactor-core-3.5.4.jar!/:3.5.4] at reactor.core.publisher.Mono.subscribe(Mono.java:4485)

```

```

~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.core.publisher.FluxOnErrorResume$ResumeSubscriber.onError(FluxOnErrorResume.java:103) ~[reactor-core-3.5.4.jar!/:3.5.4] at
reactor.netty.transport.TransportConnector$MonoChannelPromise.tryFailure(TransportConnector.java:587) ~[reactor-netty-core-1.1.5.jar!/:1.1.5] at
reactor.netty.transport.TransportConnector$MonoChannelPromise.setFailure(TransportConnector.java:541) ~[reactor-netty-core-1.1.5.jar!/:1.1.5] at
reactor.netty.transport.TransportConnector.lambda$doConnect$7(TransportConnector.java:265) ~[reactor-netty-core-1.1.5.jar!/:1.1.5] at
io.netty.util.concurrent.DefaultPromise.notifyListener0(DefaultPromise.java:590)
~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.DefaultPromise.notifyListeners0(DefaultPromise.java:583)
~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.DefaultPromise.notifyListenersNow(DefaultPromise.java:559) ~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.DefaultPromise.notifyListeners(DefaultPromise.java:492)
~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.DefaultPromise.setValue0(DefaultPromise.java:636)
~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.DefaultPromise.setFailure0(DefaultPromise.java:629)
~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.DefaultPromise.tryFailure(DefaultPromise.java:118)
~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.channel.nio.AbstractNioChannel$AbstractNioUnsafe$1.run(AbstractNioChannel.java:262) ~[netty-transport-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.PromiseTask.runTask(PromiseTask.java:98) ~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.ScheduledFutureTask.run(ScheduledFutureTask.java:153)
~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.AbstractEventExecutor.runTask(AbstractEventExecutor.java:174) ~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.AbstractEventExecutor.safeExecute(AbstractEventExecutor.java:167) ~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.SingleThreadEventExecutor.runAllTasks(SingleThreadEventExecutor.java:470) ~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.channel.nio.NioEventLoop.run(NioEventLoop.java:569) ~[netty-transport-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.SingleThreadEventExecutor$4.run(SingleThreadEventExecutor.java:997) ~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.internal.ThreadExecutorMap$2.run(ThreadExecutorMap.java:74)
~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.FastThreadLocalRunnable.run(FastThreadLocalRunnable.java:30) ~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
java.lang.Thread.run(Thread.java:833) ~[?:?] Caused by:
io.netty.channel.ConnectTimeoutException: connection timed out: /
10.233.122.199:9182 at
io.netty.channel.nio.AbstractNioChannel$AbstractNioUnsafe$1.run(AbstractNioChannel.java:261) ~[netty-transport-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.PromiseTask.runTask(PromiseTask.java:98) ~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.ScheduledFutureTask.run(ScheduledFutureTask.java:153)
~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at
io.netty.util.concurrent.AbstractEventExecutor.runTask(AbstractEventExecutor.java:174) ~[netty-common-4.1.90.Final.jar!/:4.1.90.Final] at

```

```
io.netty.util.concurrent.AbstractEventExecutor.safeExecute(AbstractEventExecutor.  
java:167) ~[netty-common-4.1.90.Final.jar!/4.1.90.Final] at  
io.netty.util.concurrent.SingleThreadEventExecutor.runAllTasks(SingleThreadEventE  
xecutor.java:470) ~[netty-common-4.1.90.Final.jar!/4.1.90.Final] at  
io.netty.channel.nio.NioEventLoop.run(NioEventLoop.java:569) ~[netty-  
transport-4.1.90.Final.jar!/4.1.90.Final] at  
io.netty.util.concurrent.SingleThreadEventExecutor$4.run(SingleThreadEventExecuto  
r.java:997) ~[netty-common-4.1.90.Final.jar!/4.1.90.Final] at  
io.netty.util.internal.ThreadExecutorMap$2.run(ThreadExecutorMap.java:74)  
~[netty-common-4.1.90.Final.jar!/4.1.90.Final] at  
io.netty.util.concurrent.FastThreadLocalRunnable.run(FastThreadLocalRunnable.java  
:30) ~[netty-common-4.1.90.Final.jar!/4.1.90.Final] at  
java.lang.Thread.run(Thread.java:833) ~[?:?]
```

Solution

Ensure the restart sequence is maintained:

1. Restart the adapter pods once they are running, then restart configuration pod.
2. If the previous step is unsuccessful, restart the configuration service again.
3. If issue persists, then it is a known issue and can be ignored as the error logs are not impacting the call flow.

2.2.17 Webhook Failure During Installation or Upgrade

Problem

Installation or upgrade unsuccessful due to webhook failure.

Error Code or Error Message

Sample error log:

```
Error: INSTALLATION FAILED: Internal error occurred: failed calling webhook  
"prometheusrulemutate.monitoring.coreos.com": failed to call webhook: Post  
"https://occne-kube-prom-stack-kube-operator.occne-infra.svc:443/admission-  
prometheusrules/mutate?timeout=10s": context deadline exceeded
```

Solution

Retry installation or upgrade using Helm.

3

Logs

This chapter explains the process to retrieve the logs and status that can be used for effective troubleshooting.

3.1 Log Levels

Logs register system events along with their date and time of occurrence. They also provide important details about a chain of events that could have led to an error or problem.

A log level helps in defining the severity level of a log message. For OCNADD, the log level of a microservice can be set to any one of the following valid values:

- **TRACE:** A log level that describes events, as a step by step execution of code. This can be ignored during the standard operation, but may be useful during extended debugging sessions.
- **DEBUG:** A log level used for events during software debugging when more granular information is needed.
- **INFO:** A standard log level indicating that something has happened, an application has entered a certain state, etc.
- **WARN:** A log level indicates that something unexpected has happened in the application, a problem, or a situation that might disturb one of the processes. But this does not mean that the application has failed. The WARN level should be used in situations that are unexpected, but the code can continue to work.
- **ERROR:** A log level that should be used when an application hits an issue preventing one or more functionalities from functioning.

Using this information, the logs can be filtered based on the system requirements. For instance, if you want to filter the critical information about your system from the informational log messages, set a filter to view messages with only WARN log level in Kibana.

3.2 Configuring Log Levels

To view logging configurations and update logging levels, check the respective service child `values.yaml`.

Following is an example from the Configurations service:

```
env:  
  CONFIGURATION_ROOT_LOG_LEVEL: INFO  
  CONFIGURATION_WEB_LOG_LEVEL: INFO
```

Once the service child `values.yaml` is modified, perform helm upgrade for the OCNADD charts.

3.3 Collecting Logs

This section describes the steps to collect logs from PODs or containers. Perform the following steps:

1. Run the following command to get the POD details:

```
$ kubectl -n <namespace_name> get pods
```

2. Collect the logs from the specific pods or containers:

```
$ kubectl logs <podname> -n <namespace>
```

Example:

```
$ kubectl logs ocnaddconfiguration-xxxxxxxxx-xxxxx -n ocnadd
```

3. Store the log in a file using the following command:

```
$ kubectl logs <podname> -n <namespace> > <filename>
```

Example:

```
$ kubectl logs ocnaddconfiguration-xxxxxxxxx-xxxxx -n ocnadd > logs.txt
```

4. (Optional) You can also use the following commands for the log stream with file redirection starting with the last 100 lines of the log:

```
$ kubectl logs <podname> -n <namespace> -f --tail <number of lines> >  
<filename>
```

Example:

```
$ kubectl logs ocnaddconfiguration-xxxxxxxxx-xxxxx -n ocnadd -f --tail  
100 > logs.txt
```

For information on the OCNADD GUI user logs, see *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide*.

3.4 Collect logs using Deployment Data Collector Tool

Perform this procedure to start the NF Deployment Data Collector module and generate the tarballs. If the user does not specify the output storage path, then this module generates the output in the same directory where the module is executed.

nfDataCapture.sh is a script which can be used for collecting all required logs from NF deployment for debugging issues. The script will collect logs from all Micro-Service PODs of specified helm input, helm deployment details, the status, description of all the kafka topics, offset details server properties and description of all the pods, services and events.

- Ensure that you have appropriate privileges to access the system and execute kubectl and helm commands.

- Perform this procedure on the same machine where the OCNADD is deployed using helm or kubectl.
- Execute the **chmod +x nfDataCapture.sh** command on the tool to provide the executable permission.
- Execute the following command to start the module:

```
./nfDataCapture.sh -n|--k8Namespace=[K8 Namespace] -u|--username=[User
Name] -p|--password=[Password] -k|--kubectl=[KUBE_SCRIPT_NAME] -h|--
helm=[HELM_SCRIPT_NAME]
-s|--size=[SIZE_OF_EACH_TARBALL] -d|--cnDBTierStatus=[CN DB TIER
STATUS]
-x|--kafkaDetails=[KAFKA DETAILS] -b|--binlogCollectionStatus=[BIN
LOG COLLECTION STATUS]
-o|--toolOutputPath -helm3=false
```

Examples:

```
./nfDataCapture.sh -k="kubectl --kubeconfig=admin.conf" -h="helm --kubeconfig
admin.conf" -n=ocnrf -s=5M -o=/tmp/
```

```
./nfDataCapture.sh -n=ocnadd -s=5M -o=/tmp/
```

```
./nfDataCapture.sh -n=ocnadd
```

```
./nfDataCapture.sh -n=ocnadd -x=false
```

```
./nfDataCapture.sh -n=ocnadd -helm3=true
```

```
./nfDataCapture.sh -n=ocnadd -u=username -p=password -s=5M -o=/home/root/
datacollector/data -helm3=true
```

```
./nfDataCapture.sh -n=ocnadd -u=username -p=password -s=5M -o=/home/root/
datacollector/data -helm3=true -b=false
```

```
./nfDataCapture.sh -n=ocnadd -u=username -p=password -s=5M -o=/home/root/
datacollector/data -helm3=true -d=true -b=false
```

Note

Default size of tarball generated will be 10M, if not provided, and default location of output will be tool working directory.

Kafka Detailed Status will be by default true and if we do not want to collect the details we have to pass it as false.

By default, helm2 is used. Use proper argument in command to use helm3.

Note

If the database is not in same namespace, then the script should be run again for the namespace in which database is deployed to capture the database related logs.

- Only if the size of the tar [example: ocnadd.debugData.2023.01.16_09.15.01.tar.gz] generated is greater than "SIZE_OF_EACH_TARBALL" specified in the command ,tar is split into several tarball based on the size specified.
- After execution of command, tar-balls will be created based on size specified in the following format:
<namespace>.debugData.<timestamp>

Example:

```
ocnadd.debugData.2023.01.16_09.15.01-part01
```

Each tarball can then be combined into one tarball with the following command:

```
cat ocnadd.debugData.2023.01.16_09.15.01-part* >  
onadd.debugData.2023.01.16_09.15.01-combined.tar.gz
```

4

Capturing `tcpdump` from CNE

Perform the following steps to capture `tcpdump` in CNE:

1. Run the following command to identify the worker node for the running pod:

```
$ kubectl get pods -n ocnadd -o wide
```

2. Login to the worker node and run the following command to search for the IP address of the pod:

```
$ ip a
```

3. Run the following command to start `tcpdump` on the identified network interface:

```
$ sudo tcpdump -n -s0 -i <interface> w <file-name>.pcap -Z <node-user-name>
```

4. Run the following command to change the file permissions:

```
$ chmod 777 <file-name>.pcap
```

5. Exit the worker node and run the following command to scp the file from the bastion host:

```
$ scp <user-name>@<worker node>:<path-in-workerNode-machine>
```

5

OCNADD Alerts

This section provides information on Oracle Communications Network Analytics Data Director (OCNADD) alerts and their configuration.

5.1 Configuring Alerts

This section describes how to configure alerts in OCNADD.

If OCNADD is installed in the CNE setup, all services are monitored by Prometheus by default. Therefore, there are no modifications required in the Helm chart. All the Prometheus alert rules present in helm chart are updated in the Prometheus Server.

Note

Here, the label used to update the Prometheus server is `role: cnc-alerting-rules`, which is added by default in helm charts.

If OCNADD is installed in the TANZU Setup, one of the files needs to be modified in Helm charts with the following parameters.

Note

Update the `release: prom-operator` label with `role: cnc-alerting-rules` in the `ocnadd-alerting-rules.yaml` file.

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  labels:
    release: prom-operator
  name: ocnadd-alerting-rules
  namespace: {{ .Values.global.cluster.namespace.name }}
```

5.2 Alert Forwarding Using Simple Network Management Protocol (SNMP)

OCNADD forwards the Prometheus alerts as Simple Network Management Protocol (SNMP) traps to the southbound SNMP servers. OCNADD uses two SNMP MIB files to generate the traps. The alert manager configuration is modified by updating the *alertmanager.yaml* file. In the *alertmanager.yaml* file, the alerts can be grouped based on podname, alertname, severity, namespace, and so on. The Prometheus alert manager is integrated with Oracle Communications Cloud Native Core, Cloud Native Environment (CNE) *snmp-notifier* service. The external SNMP servers are set up to receive the Prometheus alerts as SNMP traps. The

operator must update the MIB files along with the alert manager file to fetch the SNMP traps in their environment.

Note

- Only a user with *admin* privileges can perform the following procedures.

Alert Manager Configuration

- Run the following command to obtain the Alert Manager Secret configuration from the Bastion Host and save it to a file:

```
$ kubectl get secret alertmanager-occne-kube-prom-stack-kube-alertmanager -o yaml -n occne-infra > alertmanager-secret-k8s.yaml
```

Sample output:

```
apiVersion: v1
data:
  alertmanager.yaml: |
Z2xvYmFsOgogIHJlc29sdmVfdGltZW91dDogNW0KcmVjZWl2ZXJzOgotIG5hbWU6IGRlZmF1bHQ
tcmVjZWl2ZXIKICB3ZWJob29rX2NvbWZpZ3M6CiAgLSB1cmw6IGh0dHA6Ly9vY2NuZS1zbm1wLW
5vdGlmaWVyOjk0NjQvYWxlcnRzCnJvdXRlOgogIGdyb3VwX2J5OgogIC0gam9iCiAgZ3JvdXBfa
W50ZXJ2YWw6IDVtCiAgZ3JvdXBfd2FpdDogMzBzCiAgcmVjZWl2ZXI6IGRlZmF1bHQtcmVjZWl2
ZXIKICByZXBlYXRfaw50ZXJ2YWw6IDEyaAogIHJvdXRlczoKICAtIG1hdGNoOgogICAgaCBhbGV
ydG5hbWU6IFdhbGNoZG9nCiAgICByZWNLaxZlcljogZGVmYXVsdc1yZWNLaxZlclcgphZW1wbGF0ZX
M6Ci0gL2V0Yy9hbGVydgG1hbmFnZXIvY29uZmlnLyoudG1wbA==
kind: Secret
metadata:
  annotations:
    meta.helm.sh/release-name: occne-kube-prom-stack
    meta.helm.sh/release-namespace: occne-infra
  creationTimestamp: "2022-01-24T22:46:34Z"
  labels:
    app: kube-prometheus-stack-alertmanager
    app.kubernetes.io/instance: occne-kube-prom-stack
    app.kubernetes.io/managed-by: Helm
    app.kubernetes.io/part-of: kube-prometheus-stack
    app.kubernetes.io/version: 18.0.1
    chart: kube-prometheus-stack-18.0.1
    heritage: Helm
    release: occne-kube-prom-stack
  name: alertmanager-occne-kube-prom-stack-kube-alertmanager
  namespace: occne-infra
  resourceVersion: "5175"
  uid: a38eb420-a4d0-4020-a375-ab87421defde
type: Opaque
```

- Extract the Alert Manager configuration. The third line of the *alertmanager.yaml* file contains Alert Manager configuration encoded in base64 format. To extract the Alert Manager configuration, decode the *alertmanager.yaml* file. Run the following command:

```
echo
'Z2xvYmFsOgogIHJlc29sdmVfdGltZW91dDogNW0KcmVjZWl2ZXJzOgotIG5hbWU6IGRlZmF1bH
QtcVjZWl2ZXIKICB3ZWJob29rX2NvbWZpZ3M6CiAgLSB1cmw6IGh0dHA6Ly9vY2NuZS1zbm1wL
W5vdGlmawVyOjk0NjQvYWxlcnRzCnJvdXRlOgogIGdyb3VwX2J5OgogIC0gam9iCiAgZ3JvdXBf
aw50ZXJ2YWw6IDVtCiAgZ3JvdXBfd2FpdDogMzBzCiAgcmVjZWl2ZXI6IGRlZmF1bHQtcVjZWl
2ZXIKICByZXBlYXRfaw50ZXJ2YWw6IDEyaAogIHJvdXRlczoKICAtIG1hdGNoOgogICAgICBhbG
VydG5hbWU6IFdhGNoZG9nCiAgICByZWNaXZlcjogZGVmYXVsdC1yZWNaXZlcgp0ZW1wbGF0Z
XM6Ci0gl2V0Yy9hbGVydG1hbmFnZXIvY29uZmlnLyoudG1wbA==' | base64 --decode
```

Sample output:

```
global:
  resolve_timeout: 5m
receivers:
- name: default-receiver
  webhook_configs:
  - url: http://ocne-snmp-notifier:9464/alerts
route:
  group_by:
  - job
  group_interval: 5m
  group_wait: 30s
  receiver: default-receiver
  repeat_interval: 12h
  routes:
  - match:
    alertname: Watchdog
    receiver: default-receiver
templates:
- /etc/alertmanager/config/*.tmpl
```

- Update the *alertmanager.yaml* file, alerts can be grouped based on the following:
 - podname
 - alertname
 - severity
 - namespace

Save the changes to *alertmanager.yaml* file.

For example:

```
route:
  group_by: [podname, alertname, severity, namespace]
  group_interval: 5m
  group_wait: 30s
  receiver: default-receiver
  repeat_interval: 12h
```

- Encode the updated *alertmanager.yaml* file, run the following command:

```
$ cat alertmanager.yaml | base64 -w0
Z2xvYmFsOgogIHJlc29sdmVfdGltZW91dDogNW0KcmVjZwL2ZXJzOgotIG5hbWU6IGRlZmF1bHQ=
tcmVjZwL2ZXIKICB3ZWJob29rX2NvbmlmZPZ3M6CiAgLSB1cmw6IGh0dHA6Ly9vY2NuS1zbm1wLW
5vdGlmaWVyOjk0NjQvYyYxLnRzCi0gbmFtZTogbmV3LXLjY2VpdmdyTEKICB3ZWJob29rX2Nvb
mZPZ3M6CiAgLSB1cmw6IGh0dHA6Ly9vY2NuS1zbm1wLW5vdGlmaWVyLTE6OTQ2NS9hbGVydHMK
cm91dGU6CiAgZ3JvdXBfYnkybG6CiAgLSBqb2IKICBncm91cF9pbmRlcnZhbDogNW0KICBncm91cF9
3YWl0OiAzMHMKIChyZWNLaxZlcjogZGVmYXVsdc1yZWNLaxZlcgogIHJlcGVhdF9pbmRlcnZhbD
ogMTJoCiAgcm91dGVzOgogIC0gcmlmZwL2ZXI6IGRlZmF1bHQtcmlmZwL2ZXIKICAgIGdyb3VwX
3dhXQ6IDMwcwoGICAgZ3JvdXBfaw50ZXJ2Yww6IDVtciAgICByZXBlYXRfaW50ZXJ2Yww6IDEy
aAoGIC0gcmlmZwL2ZXI6IG5ldy1yZWNLaxZlcic0xciAgICBNcm91cF93YWl0OiAzMHMKIChyAGI
yb3VwX2ludGVydmFsOiA1bQoGICAgcmVwZWFOX2ludGVydmFsOiAxMmgKICAtIG1hdGNoOgogIC
AgICBhbGVydG5hbWU6IFdhGNoZG9nciAgICByZWNLaxZlcjogZGVmYXVsdc1yZWNLaxZlcgogI
C0gbWF0Y2g6CiAgICAgIGFSZXJ0bmFtZTogV2F0Y2hkbc2KiChyAHJlY2VpdmdyOiBiZuXctcmVj
ZwL2ZXItMQp0ZW1wbGF0ZXMKCi0gL2V0Yy9hbGVydg1hbmFnZXIvY29uZmlnLyoudG1wboAo=
```

- Edit the `alertmanager-secret-k8s.yaml` file created in step 1. Replace the `alertmanager.yaml` encoded content with the output generated in the previous step. For example:

```
$ vi alertmanager-secret-k8s.yaml
apiVersion: v1
data:
  alertmanager.yaml: <paste here the encoded content of alertmanager.yaml>
kind: Secret
metadata:
  annotations:
    meta.helm.sh/release-name: occne-kube-prom-stack
    meta.helm.sh/release-namespace: occne-infra
  creationTimestamp: "2023-02-16T09:44:58Z"
  labels:
    app: kube-prometheus-stack-alertmanager
    app.kubernetes.io/instance: occne-kube-prom-stack
    app.kubernetes.io/managed-by: Helm
    app.kubernetes.io/part-of: kube-prometheus-stack
    app.kubernetes.io/version: 36.2.0
    chart: kube-prometheus-stack-36.2.0
    heritage: Helm
    release: occne-kube-prom-stack
  name: alertmanager-occne-kube-prom-stack-kube-alertmanager
  namespace: occne-infra
  resourceVersion: "8211"
  uid: 9b499b32-6ad2-4754-8691-70665f9daab4
type: Opaque
```

- Run the following command:

```
$ kubectl apply -f alertmanager-secret-k8s.yaml -n occne-infra
```

Integrate the Alert Manager with snmp-notifier Service

- Update the SNMP client destination in the *occne-snmp-notifier* service with the SNMP destination client IP.

Note

For a persistent client configuration, edit the values of the *snmp-notifier* in Helm charts and perform a Helm upgrade.

Add "warn" in alert severity to receive warning alerts from OCNADD. Run the following command:

```
$ kubectl edit deployment -n occne-infra occne-snmp-notifier
```

1. update the field "--snmp.destination=<IP>:<port>" inside the args of container and add the snmp-client destination IP.

Example:

```
spec:
  containers:
  - args:
    - --snmp.destination=10.20.30.40:162
```

2. "warn" should also be added to the severity list as some of the DD alerts are raised with severity: warn.

Example:

```
- --alert.severities=critical,major,minor,warning,info,clear,warn
```

- To verify the SNMP notification, see the new notifications in the pod logs of *occne snmp notifier*. Run the following command to see the logs:

```
$ kubectl logs -n occne-infra <occne-snmp-notifier-pod-name>
```

Sample output:

```
10.20.30.50 - - [26/Mar/2023:13:58:14 +0000] "POST /alerts HTTP/1.1" 200 0
10.20.30.60 - - [26/Mar/2023:14:02:51 +0000] "POST /alerts HTTP/1.1" 200 0
10.20.30.70 - - [26/Mar/2023:14:03:14 +0000] "POST /alerts HTTP/1.1" 200 0
10.20.30.80 - - [26/Mar/2023:14:07:51 +0000] "POST /alerts HTTP/1.1" 200 0
10.20.30.90 - - [26/Mar/2023:14:08:14 +0000] "POST /alerts HTTP/1.1" 200 0
```

OCNADD MIB Files

Two OCNADD MIB files are used to generate the traps. The operator has to update the MIB files and the alert manager file to obtain the traps in their environment. The files are:

- *OCNADD-MIB-TC-23.2.0.mib*: This is a top level mib file, where the objects and their data types are defined.
- *OCNADD-MIB-23.2.0.mib*: This file fetches the objects from the top level mib file and based on the alert notification, the objects are selected for display.

Note

MIB files are packaged along with OCNADD Custom Templates. Download the files from [MOS](#). See *Oracle Communications Cloud Native Core Network Analytics Data Director Installation, Upgrade, and Fault Recovery Guide* for more information.

5.3 List of Alerts

This section provides detailed information about the alert rules defined for OCNADD.

5.3.1 System Level Alerts

This section lists the system level alerts for OCNADD.

OCNADD_POD_CPU_USAGE_ALERT

Table 5-1 OCNADD_POD_CPU_USAGE_ALERT

Field	Details
Triggering Condition	POD CPU usage is above set threshold (default 70%)
Severity	Major
Description	OCNADD Pod High CPU usage detected for the continuous period of 5 min

Table 5-1 (Cont.) OCNADD_POD_CPU_USAGE_ALERT

Field	Details
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: CPU usage is {{ "{{" }} \$value printf "%.2f" }} which is above threshold {{ .Values.global.cluster.cpu_threshold }} %' Expression: expr: <pre>(sum(rate(container_cpu_usage_seconds_total{im age!="" , pod=~".*ocnadd.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*2) or (sum(rate(container_cpu_usage_seconds_total{im age!="" , pod=~".*aggregation.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*3) or (sum(rate(container_cpu_usage_seconds_total{im age!="" , pod=~".*kafka.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*6) or (sum(rate(container_cpu_usage_seconds_total{im age!="" , pod=~".*zookeeper.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}) or (sum(rate(container_cpu_usage_seconds_total{im age!="" , pod=~".*egw.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*6) or (sum(rate(container_cpu_usage_seconds_total{im age!="" , pod=~".*adapter.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*3)</pre>
OID	1.3.6.1.4.1.323.5.3.51.29.4002
Metric Used	container_cpu_usage_seconds_total Note: This is a Kubernetes metric used for instance availability monitoring. If the metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the CPU utilization is below the critical threshold. Note: The threshold is configurable in the ocnadd-custom-values.yaml file. If guidance is required, contact My Oracle Support (MOS).

OCNADD_POD_MEMORY_USAGE_ALERT

Table 5-2 OCNADD_POD_MEMORY_USAGE_ALERT

Field	Details
Triggering Condition	POD Memory usage is above set threshold (default 70%)
Severity	Major
Description	OCNADD Pod High Memory usage detected for the continuous period of 5 min
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Memory usage is {{ "{{" }} \$value printf "%.2f" }} which is above threshold {{ .Values.global.cluster.memory_threshold }} % ' Expression: expr: (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnadd.*"}) by (pod,namespace) > 2*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*kafka.*"}) by (pod,namespace) > 10*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*zookeeper.*"}) by (pod,namespace) > 1*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*egw.*"}) by (pod,namespace) > 6*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*adapter.*"}) by (pod,namespace) > 4*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100)
OID	1.3.6.1.4.1.323.5.3.51.29.4005
Metric Used	container_memory_working_set_bytes Note : This is a Kubernetes metric used for instance availability monitoring. If the metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the memory utilization is below the critical threshold. Note: The threshold is configurable in the ocnadd/values.yaml file. If guidance is required, contact My Oracle Support (MOS) .

OCNADD_POD_RESTARTED

Table 5-3 OCNADD_POD_RESTARTED

Field	Details
Triggering Condition	A POD has restarted
Severity	Minor
Description	A POD has restarted in last 2 min
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: A Pod has restarted' Expression: expr: kube_pod_container_status_restarts_total{namespace="{{ .Values.global.cluster.nameSpace.name }}" } > 1
OID	1.3.6.1.4.1.323.5.3.51.29.5006
Metric Used	kube_pod_container_status_restarts_total Note: This is a Kubernetes metric used for instance availability monitoring. If the metric is not available, use the similar metric asexposed by the monitoring system.
Resolution	The alert is cleared automatically if the specific pod is up. Steps: 1. Check the application logs. Check for database related failures such as connectivity, Kubernetes secrets, and so on. 2. Run the following command to check orchestration logs for liveness or readiness probe failures: <pre>kubectl get po -n <namespace></pre> Note the full name of the pod that is not running, and use it in the following command: <pre>kubectl describe pod <desired full pod name> -n <namespace></pre> 3. Check the database status. For more information, see "Oracle Communications Cloud Native Core DBTier User Guide". 4. If the issue persists, capture all the outputs from the above steps and contact My Oracle Support (MOS). If guidance is required.

5.3.2 Application Level Alerts

This section lists the application level alerts for OCNADD.

OCNADD_ADMIN_SVC_DOWN

Table 5-4 OCNADD_ADMIN_SVC_DOWN

Field	Details
Triggering Condition	The OCNADD Admin service is down or not accessible.
Severity	Critical
Description	OCNADD Admin service not available for more than 2 min
Alert Details	Summary: "namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddadminservice service is down' Expression: expr: up{service="ocnaddadminservice"} != 1
OID	1.3.6.1.4.1.323.5.3.51.30.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Admin service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: kubectl -n <namespace> get pod If it is not in running state, capture the pod logs and events. Run the following command to fetch the events as follows: kubectl get events --sortby=.metadata.creationTimestamp -n <namespace> 3. Refer to the application logs and check for database related failures such as connectivity, invalid secrets, and so on. 4. Run the following command to check Helm status and make sure there are no errors: helm status <helm release name of data director> -n<namespace> If it is not in "STATUS: DEPLOYED", then again capture logs and events. 5. If the issue persists, capture all the outputs from the above steps and contact My Oracle Support (MOS) , If guidance is required.

OCNADD_ALARM_SVC_DOWN

Table 5-5 OCNADD_ALARM_SVC_DOWN

Field	Details
Triggering Condition	The alarm service is down or not accessible.
Severity	Critical
Description	OCNADD Alarm service not available for more than 2 min
Alert Details	Summary: namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddalarm service is down' Expression: expr: up{service="ocnaddalarm"} != 1
OID	1.3.6.1.4.1.323.5.3.51.24.2002
Metric Used	'up'  Note This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.

Table 5-5 (Cont.) OCNADD_ALARM_SVC_DOWN

Field	Details
Resolution	The alert is cleared automatically when the OCNADD Alarm service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: <code>kubectl -n <namespace> get pod</code> If it is not in running state, capture the pod logs and events. Run the following command to fetch the events as follows: <code>kubectl get events --sortby=.metadata.creationTimestamp -n <namespace></code> 3. Refer to the application logs and check for database related failures such as connectivity, invalid secrets, and so on. 4. Run the following command to check Helm status and make sure there are no errors: <code>helm status <helm release name of data director> -n<namespace></code> If it is not in "STATUS: DEPLOYED", then again capture logs and events. 5. If the issue persists, capture all the outputs from the above steps and contact My Oracle Support (MOS), If guidance is required.

OCNADD_CONFIG_SVC_DOWN

Table 5-6 OCNADD_CONFIG_SVC_DOWN

Field	Details
Triggering Condition	The configuration service is down or not accessible.
Severity	Critical
Description	OCNADD Configuration service not available for more than 2 min
Alert Details	Summary: <pre>'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }': ocnaddconfiguration service is down'</pre> Expression: <pre>expr: up{service="ocnaddconfiguration"} != 1</pre>
OID	1.3.6.1.4.1.323.5.3.51.20.2002

Table 5-6 (Cont.) OCNADD_CONFIG_SVC_DOWN

Field	Details
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Configuration service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: <code>kubectl -n <namespace> get pod</code> If it is not in running state, capture the pod logs and events. Run the following command to fetch the events as follows: <code>kubectl get events --sortby=.metadata.creationTimestamp -n <namespace></code> 3. Refer to the application logs and check for database related failures such as connectivity, invalid secrets, and so on. 4. Run the following command to check Helm status and make sure there are no errors: <code>helm status <helm release name of data director> -n<namespace></code> If it is not in "STATUS: DEPLOYED", then again capture logs and events. 5. If the issue persists, capture all the outputs from the above steps and contact My Oracle Support (MOS). If guidance is required..

OCNADD_CONSUMER_ADAPTER_SVC_DOWN**Table 5-7 OCNADD_CONSUMER_ADAPTER_SVC_DOWN**

Field	Details
Triggering Condition	The OCNADD Consumer Adapter service is down or not accessible
Severity	Critical
Description	OCNADD Consumer Adapter service not available for more than 2 min

Table 5-7 (Cont.) OCNADD_CONSUMER_ADAPTER_SVC_DOWN

Field	Details
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Consumer Adapter service is down' Expression: expr: up{service=~".*adapter.*"} != 1
OID	1.3.6.1.4.1.323.5.3.51.25.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Consumer Adapter service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: kubect! -n <namespace> get pod If it is not in running state, capture the pod logs and events. Run the following command to fetch the events as follows: <pre>kubect! get events -- sortby=.metadata.creationTimestamp -n <namespace></pre> 3. Refer to the application logs and check for database related failures such as connectivity, invalid secrets, and so on. 4. Run the following command to check Helm status and make sure there are no errors: helm status <helm release name of data director> -n<namespace> If it is not in "STATUS: DEPLOYED", then again capture logs and events. 5. If the issue persists, capture all the outputs from the above steps and contact My Oracle Support (MOS), If guidance is required.

OCNADD_E2E_AVG_RECORD_LATENCY_CRITICAL_THRESHOLD_CROSSED

Table 5-8 OCNADD_E2E_AVG_RECORD_LATENCY_CRITICAL_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total observed latency is above the configured critical threshold alert level of 100%
Severity	Critical
Description	Average E2E Latency is above configured critical threshold alert level (100%) for the period of 5 min
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: E2E Latency is above 100% of Maximum permissible latency {{ .Values.global.cluster.max_latency }} ms' Expression: expr: (sum (irate(ocnadd_egress_e2e_request_processing_lat ency_seconds_sum[5m])) by (namespace)) /(sum (irate(ocnadd_egress_e2e_request_processing_lat ency_seconds_count[5m])) by (namespace)) > 1.0*{{ .Values.global.cluster.max_latency }}
OID	1.3.6.1.4.1.323.5.3.51.29.5010
Metric Used	ocnadd_egress_e2e_request_processing_latency_ seconds_sum, ocnadd_egress_e2e_request_processing_latency_ seconds_count
Resolution	The alert is cleared automatically when the average latency goes below the critical threshold alert level of permissible latency.

OCNADD_E2E_AVG_RECORD_LATENCY_MAJOR_THRESHOLD_CROSSED

Table 5-9 OCNADD_E2E_AVG_RECORD_LATENCY_MAJOR_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total observed latency is above the configured major threshold alert level of 95%
Severity	Major
Description	Average E2E Latency is above configured minor threshold alert level (95%) for the period of 5 min

Table 5-9 (Cont.)

OCNADD_E2E_AVG_RECORD_LATENCY_MAJOR_THRESHOLD_CROSSED

Field	Details
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: E2E Latency is above 95% of Maximum permissible latency {{ .Values.global.cluster.max_latency }} ms' Expression: expr: (sum (irate(ocnadd_egress_e2e_request_processing_lat ency_seconds_sum[5m])) by (namespace)) / (sum (irate(ocnadd_egress_e2e_request_processing_lat ency_seconds_count[5m])) by (namespace)) > .95*{{ .Values.global.cluster.max_latency }} <= 1.0*{{ .Values.global.cluster.max_latency }}
OID	1.3.6.1.4.1.323.5.3.51.29.5010
Metric Used	ocnadd_egress_e2e_request_processing_latency_ seconds_sum, ocnadd_egress_e2e_request_processing_latency_ seconds_count
Resolution	The alert is cleared automatically when the average latency goes below the major threshold alert level of 95% of permissible latency.

OCNADD_E2E_AVG_RECORD_LATENCY_MINOR_THRESHOLD_CROSSED

Table 5-10 OCNADD_E2E_AVG_RECORD_LATENCY_MINOR_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total observed latency is above the configured minor threshold alert level of 90%.
Severity	Minor
Description	Average E2E Latency is above configured minor threshold alert level (90%) for the period of 5 min
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: E2E Latency is above 90% of Maximum permissible latency {{ .Values.global.cluster.max_latency }} ms' Expression: expr: (sum (irate(ocnadd_egress_e2e_request_processing_lat ency_seconds_sum[5m])) by (namespace)) / (sum (irate(ocnadd_egress_e2e_request_processing_lat ency_seconds_count[5m])) by (namespace)) > .90*{{ .Values.global.cluster.max_latency }} <= 0.95*{{ .Values.global.cluster.max_latency }}
OID	1.3.6.1.4.1.323.5.3.51.29.5010

Table 5-10 (Cont.)

OCNADD_E2E_AVG_RECORD_LATENCY_MINOR_THRESHOLD_CROSSED

Field	Details
Metric Used	ocnadd_egress_e2e_request_processing_latency_seconds_sum, ocnadd_egress_e2e_request_processing_latency_seconds_count
Resolution	The alert is cleared automatically when the average latency goes below the minor threshold alert level of 90% of permissible latency.

OCNADD_E2E_AVG_RECORD_LATENCY_WARNING_THRESHOLD_CROSSED

Table 5-11 OCNADD_E2E_AVG_RECORD_LATENCY_WARNING_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total observed latency is above the configured warning threshold alert level of 80%.
Severity	Warn
Description	Average E2E Latency is above configured warning threshold alert level (80%) for the period of 5 min
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: E2E Latency is above 80% of Maximum permissible latency {{ .Values.global.cluster.max_latency }} ms' Expression: expr: (sum (irate(ocnadd_egress_e2e_request_processing_latency_seconds_sum[5m])) by (namespace)) / (sum (irate(ocnadd_egress_e2e_request_processing_latency_seconds_count[5m])) by (namespace)) > .80*{{ .Values.global.cluster.max_latency }} <= .90*{{ .Values.global.cluster.max_latency }}
OID	1.3.6.1.4.1.323.5.3.51.29.5010
Metric Used	ocnadd_egress_e2e_request_processing_latency_seconds_sum, ocnadd_egress_e2e_request_processing_latency_seconds_count
Resolution	The alert is cleared automatically when the average latency goes below the warning threshold alert level of 80% of permissible latency.

OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_0.1PERCENT**Table 5-12 OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_0.1PERCENT**

Field	Details
Triggering Condition	The Egress adapter failure rate towards the third party application is above the configured threshold of 0.1% of total supported MPS.
Severity	Info
Description	Egress external connection failure rate towards 3rd party application is crossing info threshold of 0.1% in the period of 5 min
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Egress external connection Failure Rate detected above 0.1 Percent of Total Egress external connections' Expression: expr: (sum(rate(ocnadd_egress_failed_request_total[5m])) by (namespace))/ (sum(rate(ocnadd_egress_requests_total[5m])) by (namespace)) *100 >= 0.1 < 10
OID	1.3.6.1.4.1.323.5.3.51.29.5012
Metric Used	ocnadd_egress_failed_request_total, ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the failure rate towards third party consumer goes below the threshold (0.1%) alert level of supported MPS.

OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_10PERCENT**Table 5-13 OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_10PERCENT**

Field	Details
Triggering Condition	The Egress adapter failure rate towards the third party application is above the configured threshold of 10% of total supported MPS.
Severity	Minor
Description	Egress external connection failure rate towards 3rd party application is crossing minor threshold of 10% in the period of 5 min

Table 5-13 (Cont.) OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_10PERCENT

Field	Details
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Egress external connection Failure Rate detected above 10 Percent of Total Egress external connections' Expression: expr: (sum(rate(ocnadd_egress_failed_request_total[5m])) by (namespace))/ (sum(rate(ocnadd_egress_requests_total[5m])) by (namespace)) *100 >= 10 < 25
OID	1.3.6.1.4.1.323.5.3.51.29.5012
Metric Used	ocnadd_egress_failed_request_total, ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the failure rate towards third party consumer goes below the threshold (10%) alert level of supported MPS.

OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_1PERCENT

Table 5-14 OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_1PERCENT

Field	Details
Triggering Condition	The Egress adapter failure rate towards the third party application is above the configured threshold of 1% of total supported MPS.
Severity	Warn
Description	Egress external connection failure rate towards 3rd party application is crossing warning threshold of 1% in the period of 5 min
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Egress external connection Failure Rate detected above 1 Percent of Total Egress external connections' Expression: expr: (sum(rate(ocnadd_egress_failed_request_total[5m])) by (namespace))/ (sum(rate(ocnadd_egress_requests_total[5m])) by (namespace)) *100 >= 1 < 10
OID	1.3.6.1.4.1.323.5.3.51.29.5012
Metric Used	ocnadd_egress_failed_request_total, ocnadd_egress_requests_total

Table 5-14 (Cont.) OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_1PERCENT

Field	Details
Resolution	The alert is cleared automatically when the failure rate towards third party consumer goes below the threshold (1%) alert level of supported MPS.

OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_25PERCENT**Table 5-15 OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_25PERCENT**

Field	Details
Triggering Condition	The Egress adapter failure rate towards the third party application is above the configured threshold of 25% of total supported MPS.
Severity	Major
Description	Egress external connection failure rate towards 3rd party application is crossing major threshold of 25% in the period of 5 min
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Egress external connection Failure Rate detected above 25 Percent of Total Egress external connections' Expression: expr: (sum(rate(ocnadd_egress_failed_request_total[5m])) by (namespace))/ (sum(rate(ocnadd_egress_requests_total[5m])) by (namespace)) *100 >= 25 < 50
OID	1.3.6.1.4.1.323.5.3.51.29.5012
Metric Used	ocnadd_egress_failed_request_total, ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the failure rate towards third party consumer goes below the threshold (25%) alert level of supported MPS.

OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_50PERCENT**Table 5-16 OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_50PERCENT**

Field	Details
Triggering Condition	The Egress adapter failure rate towards the 3rd party application is above the configured threshold of 50% of total supported MPS.
Severity	Critical
Description	Egress external connection failure rate towards 3rd party application is crossing critical threshold of 50% in the period of 5 min.

Table 5-16 (Cont.) OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_50PERCENT

Field	Details
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Egress external connection Failure Rate detected above 50 Percent of Total Egress external connections' Expression: expr: (sum(rate(ocnadd_egress_failed_request_total[5m]) by (namespace))/ (sum(rate(ocnadd_egress_requests_total[5m])) by (namespace)) *100 >= 50
OID	1.3.6.1.4.1.323.5.3.51.29.5012
Metric Used	ocnadd_egress_failed_request_total, ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the failure rate towards 3rd party consumer goes below the threshold (50%) alert level of supported MPS.

OCNADD_HEALTH_MONITORING_SVC_DOWN

Table 5-17 OCNADD_HEALTH_MONITORING_SVC_DOWN

Field	Details
Triggering Condition	The health monitoring service is down or not accessible.
Severity	Critical
Description	OCNADD Health monitoring service not available for more than 2 min.
Alert Details	Summary: summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddhealthmonitoring service is down' Expression: expr: up{service="ocnaddhealthmonitoring"} != 1
OID	1.3.6.1.4.1.323.5.3.51.28.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.

Table 5-17 (Cont.) OCNADD_HEALTH_MONITORING_SVC_DOWN

Field	Details
Resolution	The alert is cleared automatically when the OCNADD Health monitoring service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: <code>kubectl -n <namespace> get pod</code> If it is not in running state, capture the pod logs and events. Run the following command to fetch the events as follows: <code>kubectl get events --sortby=.metadata.creationTimestamp -n <namespace></code> 3. Refer to the application logs and check for database related failures such as connectivity, invalid secrets, and so on. 4. Run the following command to check Helm status and make sure there are no errors: <code>helm status <helm release name of data director> -n<namespace></code> If it is not in "STATUS: DEPLOYED", then again capture logs and events. 5. If the issue persists, capture all the outputs from the above steps and contact My Oracle Support (MOS) if guidance is required.

OCNADD_INGRESS_TRAFFIC_RATE_DECREASE_SPIKE_10PERCENT**Table 5-18 OCNADD_INGRESS_TRAFFIC_RATE_DECREASE_SPIKE_10PERCENT**

Field	Details
Triggering Condition	The ingress traffic decrease is more than 10% of the supported MPS.
Severity	Major
Description	The ingress traffic decrease is more than 10% of the supported MPS in last 5 min.

Table 5-18 (Cont.)
OCNADD_INGRESS_TRAFFIC_RATE_DECREASE_SPIKE_10PERCENT

Field	Details
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Ingress MPS decrease is more than 10 Percent of current supported MPS' Expression: expr: sum(irate(kafka_stream_processor_node_process_total{service=~".*aggregation.*"}[5m])) by (namespace)/ sum(irate(kafka_stream_processor_node_process_total{service=~".*aggregation.*"}[5m] offset 5m)) by (namespace) <= 0.9
OID	1.3.6.1.4.1.323.5.3.51.29.5013
Metric Used	kafka_stream_processor_node_process_total
Resolution	The alert is cleared automatically when the decrease in MPS comes back to lower than 10% of the supported MPS.

OCNADD_INGRESS_TRAFFIC_RATE_INCREASE_SPIKE_10PERCENT

Table 5-19 OCNADD_INGRESS_TRAFFIC_RATE_INCREASE_SPIKE_10PERCENT

Field	Details
Triggering Condition	The ingress traffic increase is more than 10% of the supported MPS.
Severity	Major
Description	The ingress traffic increase is more than 10% of the supported MPS in last 5 min.
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Ingress MPS increase is more than 10 Percent of current supported MPS' Expression: expr: sum(irate(kafka_stream_processor_node_process_total{service=~".*aggregation.*"}[5m])) by (namespace)/ sum(irate(kafka_stream_processor_node_process_total{service=~".*aggregation.*"}[5m] offset 5m)) by (namespace) >= 1.1
OID	1.3.6.1.4.1.323.5.3.51.29.5013
Metric Used	kafka_stream_processor_node_process_total

Table 5-19 (Cont.)

OCNADD_INGRESS_TRAFFIC_RATE_INCREASE_SPIKE_10PERCENT

Field	Details
Resolution	The alert is cleared automatically when the increase in MPS comes back to lower than 10% of the supported MPS.

OCNADD_KAFKA_PACKET_DROP_THRESHOLD_10PERCENT_MPS

Table 5-20 OCNADD_KAFKA_PACKET_DROP_THRESHOLD_10PERCENT_MPS

Field	Details
Triggering Condition	The packet drop rate in Kafka streams is above the configured critical threshold of 10% of total supported MPS.
Severity	Critical
Description	The packet drop rate in Kafka streams is above the configured critical threshold of 10% of total supported MPS in the period of 5 min.
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Packet Drop rate is above 10% threshold of Max messages per second:{{ .Values.global.cluster.mps }}' Expression: expr: sum(rate(kafka_stream_task_dropped_records_total{service=~".*aggregation.*"}[5m])) by (namespace) > 0.1*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5011
Metric Used	kafka_stream_task_dropped_records_total
Resolution	The alert is cleared automatically when the packet drop rate goes below the critical threshold (10%) alert level of supported MPS.

OCNADD_KAFKA_PACKET_DROP_THRESHOLD_1PERCENT_MPS

Table 5-21 OCNADD_KAFKA_PACKET_DROP_THRESHOLD_1PERCENT_MPS

Field	Details
Triggering Condition	The packet drop rate in Kafka streams is above the configured major threshold of 1% of total supported MPS.
Severity	Major
Description	The packet drop rate in Kafka streams is above the configured major threshold of 1% of total supported MPS in the period of 5 min.

Table 5-21 (Cont.) OCNADD_KAFKA_PACKET_DROP_THRESHOLD_1PERCENT_MPS

Field	Details
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Packet Drop rate is above 1% threshold of Max messages per second:{{ .Values.global.cluster.mps }}' Expression: expr: sum(rate(kafka_stream_task_dropped_records_tot al{service=~".*aggregation.*"}[5m])) by (namespace) > 0.01*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5011
Metric Used	kafka_stream_task_dropped_records_total
Resolution	The alert is cleared automatically when the packet drop rate goes below the major threshold (1%) alert level of supported MPS.

OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED

Table 5-22 OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total egress MPS crossed the critical threshold alert level of 100% of the supported MPS.
Severity	Critical
Description	Total Egress Message Rate is above configured critical threshold alert (100%) for the period of 5 min.
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above supported Max messages per second:{{ .Values.global.cluster.mps }}' Expression: expr: sum (irate(ocnadd_egress_requests_total[5m])) by (namespace) > 1.0*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5008
Metric Used	ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the critical threshold alert level of 100% of supported MPS.

OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED_FOR_A_CONSUMER

Table 5-23 OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED_FOR_A_CONSUMER

Field	Details
Triggering Condition	The total egress MPS crossed the critical threshold alert level of 100% of the supported MPS for a consumer.
Severity	Critical
Description	Total Egress Message Rate is above configured critical threshold alert (100%) for the period of 5 min.
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above supported Max messages per second:{{ .Values.global.cluster.mps }}' Expression: expr: sum (rate(ocnadd_egress_requests_total[5m])) by (namespace, instance_identifier) > 1.0*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5009
Metric Used	ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the critical threshold alert level of 100% of supported MPS.

OCNADD_MPS_CRITICAL_INGRESS_THRESHOLD_CROSSED

Table 5-24 OCNADD_MPS_CRITICAL_INGRESS_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total ingress MPS crossed the critical threshold alert level of 100% of the supported MPS.
Severity	Critical
Description	Total Ingress Message Rate is above configured critical threshold alert (100%) for the period of 5 min.

Table 5-24 (Cont.) OCNADD_MPS_CRITICAL_INGRESS_THRESHOLD_CROSSED

Field	Details
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above the supported Max messages per second:{{ .Values.global.cluster.mps }}' Expression: expr: sum(irate(kafka_stream_processor_node_process _total{service=~".*aggregation.*"}[5m])) by (namespace) > 1.0*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5007
Metric Used	kafka_stream_processor_node_process_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the critical threshold alert level of 100% of supported MPS.

OCNADD_MPS_MAJOR_EGRESS_THRESHOLD_CROSSED

Table 5-25 OCNADD_MPS_MAJOR_EGRESS_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total egress MPS crossed the major threshold alert level of 95% of the supported MPS.
Severity	Major
Description	Total Egress Message Rate is above configured major threshold alert (95%) for the period of 5 min.
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above 95% of Max messages per second: {{ .Values.global.cluster.mps }}' Expression: expr: sum (irate(ocnadd_egress_requests_total[5m])) by (namespace) > 0.95*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5008
Metric Used	ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the major threshold alert level of 95% of supported MPS.

OCNADD_MPS_MAJOR_INGRESS_THRESHOLD_CROSSED

Table 5-26 OCNADD_MPS_MAJOR_INGRESS_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total ingress MPS crossed the major threshold alert level of 95% of the supported MPS.
Severity	Major
Description	Total Ingress Message Rate is above configured major threshold alert (95%) for the period of 5 min.
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above 95 Percent of Max messages per second:{{ .Values.global.cluster.mps }}' Expression: expr: sum(irate(kafka_stream_processor_node_process _total{service=~".*aggregation.*"}[5m])) by (namespace) > 0.95*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5007
Metric Used	kafka_stream_processor_node_process_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the major threshold alert level of 95%.

OCNADD_MPS_MINOR_EGRESS_THRESHOLD_CROSSED

Table 5-27 OCNADD_MPS_MINOR_EGRESS_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total egress MPS crossed the minor threshold alert level of 90% of the supported MPS.
Severity	Minor
Description	Total Egress Message Rate is above configured minor threshold alert (90%) for the period of 5 min.
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above 90% of Max messages per second: {{ .Values.global.cluster.mps }}' Expression: expr: sum (irate(ocnadd_egress_requests_total[5m])) by (namespace) > 0.90*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5008

Table 5-27 (Cont.) OCNADD_MPS_MINOR_EGRESS_THRESHOLD_CROSSED

Field	Details
Metric Used	ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the minor threshold alert level of 90% of supported MPS.

OCNADD_MPS_MINOR_INGRESS_THRESHOLD_CROSSED**Table 5-28 OCNADD_MPS_MINOR_INGRESS_THRESHOLD_CROSSED**

Field	Details
Triggering Condition	The total ingress MPS crossed the minor threshold alert level of 90% of the supported MPS.
Severity	Minor
Description	Total Ingress Message Rate is above configured minor threshold alert (90%) for the period of 5 min.
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above 90 Percent of Max messages per second:{{ .Values.global.cluster.mps }}' Expression: expr: sum(irate(kafka_stream_processor_node_process _total{service=~".*aggregation.*"}[5m])) by (namespace) > 0.9*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5007
Metric Used	kafka_stream_processor_node_process_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the minor threshold alert level of 90%.

OCNADD_MPS_WARNING_EGRESS_THRESHOLD_CROSSED**Table 5-29 OCNADD_MPS_WARNING_EGRESS_THRESHOLD_CROSSED**

Field	Details
Triggering Condition	The total egress MPS crossed the warning threshold alert level of 80% of the supported MPS.
Severity	Warn
Description	Total Egress Message Rate is above configured warning threshold alert (80%) for the period of 5 min.

Table 5-29 (Cont.) OCNADD_MPS_WARNING_EGRESS_THRESHOLD_CROSSED

Field	Details
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above 80% of Max messages per second: {{ .Values.global.cluster.mps }}' Expression: expr: sum (irate(ocnadd_egress_requests_total[5m])) by (namespace) > 0.80*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5008
Metric Used	ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the warning threshold alert level of 80% of supported MPS.

OCNADD_MPS_WARNING_INGRESS_THRESHOLD_CROSSED

Table 5-30 OCNADD_MPS_WARNING_INGRESS_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total ingress MPS crossed the warning threshold of 80% of the supported MPS.
Severity	Warn
Description	Total Ingress Message Rate is above configured warning threshold (80%) for the period of 5 min.
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above 80 Percent of Max messages per second:{{ .Values.global.cluster.mps }}' Expression: expr: sum(irate(kafka_stream_processor_node_process _total{service=~".*aggregation.*"}[5m])) by (namespace) > 0.8*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5007
Metric Used	kafka_stream_processor_node_process_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the warning threshold level of 80%.

OCNADD_NRF_AGGREGATION_SVC_DOWN

Table 5-31 OCNADD_NRF_AGGREGATION_SVC_DOWN

Field	Details
Triggering Condition	The NRF Aggregation service is down or not accessible
Severity	Critical
Description	OCNADD NRF Aggregation service not available for more than 2 min
Alert Details	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddnrfaggregation service is down' Expression: expr: up{service="ocnaddnrfaggregation"} != 1
OID	1.3.6.1.4.1.323.5.3.53.1.31.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD NRF Aggregation service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: kubectl -n <namespace> get pod If it is not in running state, capture the pod logs and events. Run the following command to fetch the events as follows: <pre>kubectl get events -- sortby=.metadata.creationTimestamp -n <namespace></pre> 3. Refer to the application logs and check for database related failures such as connectivity, invalid secrets, and so on. 4. Run the following command to check Helm status and make sure there are no errors: helm status <helm release name of data director> -n<namespace> If it is not in "STATUS: DEPLOYED", then again capture logs and events. 5. If the issue persists, capture all the outputs from the above steps and contact My Oracle Support (MOS). If guidance is required.

OCNADD_SCP_AGGREGATION_SVC_DOWN

Table 5-32 OCNADD_SCP_AGGREGATION_SVC_DOWN

Field	Details
Triggering Condition	The SCP Aggregation service is down or not accessible
Severity	Critical
Description	OCNADD SCP Aggregation service not available for more than 2 min.
Alert Details	Summary: <pre>'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddscpaggregation service is down'</pre> Expression: <pre>expr: up{service="ocnaddscpaggregation"} != 1</pre>
OID	1.3.6.1.4.1.323.5.3.51.22.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD SCP Aggregation service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: <pre>kubectl -n <namespace> get pod</pre> If it is not in running state, capture the pod logs and events. Run the following command to fetch the events as follows: <pre>kubectl get events -- sortby=.metadata.creationTimestamp -n <namespace></pre> 3. Refer to the application logs and check for database related failures such as connectivity, invalid secrets, and so on. 4. Run the following command to check Helm status and make sure there are no errors: <pre>helm status <helm release name of data director> - n<namespace></pre> If it is not in "STATUS: DEPLOYED", then again capture logs and events. 5. If the issue persists, capture all the outputs from the above steps and contact My Oracle Support (MOS), If guidance is required.

OCNADD_SEPP_AGGREGATION_SVC_DOWN

Table 5-33 OCNADD_SEPP_AGGREGATION_SVC_DOWN

Field	Details
Triggering Condition	The SEPP Aggregation service is down or not accessible
Severity	Critical
Description	OCNADD SEPP Aggregation service not available for more than 2 min.
Alert Details	Summary: <pre>'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddseppaggregation service is down'</pre> Expression: <pre>expr: up{service="ocnaddseppaggregation"} != 1</pre>
OID	1.3.6.1.4.1.323.5.3.53.1.32.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD SEPP Aggregation service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: <pre>kubectl -n <namespace> get pod</pre> If it is not in running state, capture the pod logs and events. Run the following command to fetch the events as follows: <pre>kubectl get events -- sortby=.metadata.creationTimestamp -n <namespace></pre> 3. Refer to the application logs and check for database related failures such as connectivity, invalid secrets, and so on. 4. Run the following command to check Helm status and make sure there are no errors: <pre>helm status <helm release name of data director> -n<namespace></pre> If it is not in "STATUS: DEPLOYED", then again capture logs and events. 5. If the issue persists, capture all the outputs from the above steps and contact My Oracle Support (MOS). If guidance is required.