

Oracle® Communications

Network Analytics Data Director

Troubleshooting Guide



Release 25.2.200
G49000-01
December 2025

ORACLE®

Copyright © 2022, 2025, 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	References	1

2 Troubleshooting OCNADD

2.1	Generic Checklist	1
2.2	Helm Install and Upgrade Failure	14
2.2.1	Incorrect Image Name in ocnadd-<group-name>-custom-values.yaml File	14
2.2.2	Failed Helm Installation/Upgrade Due to Prometheus Rules Applying Failure	15
2.2.3	Docker Registry is Configured Incorrectly	16
2.2.4	Continuous Restart of Pods	16
2.2.5	Adapter Deployment Removed during Upgrade or Rollback	16
2.2.6	ocnadd-custom-values.yaml File Parse Failure	17
2.2.7	Kafka Brokers Continuously Restart after Reinstallation	18
2.2.8	Kafka Brokers Continuously Restart After the Disk is Full	18
2.2.9	Kafka Brokers Restart on Installation	20
2.2.10	Database Goes into the Deadlock State	20
2.2.11	Upgrade fails due to Database MaxNoOfAttributes exhausted	21
2.2.12	Webhook Failure During Installation or Upgrade	21
2.2.13	Data Feeds Do Not delete after Rollback	22
2.2.14	Data Feeds Do Not Restart after Rollback	22
2.2.15	Third-Party Endpoint DOWN State and New Feed Creation	23
2.2.16	Adapter Feed Not Coming Up After Rollback	23
2.2.17	Adapters pods are in the "init:CrashLoopBackOff" error state after rollback	24
2.2.18	Feed/Filter Configurations are Missing From Dashboard After Upgrade	24
2.2.19	Kafka Feed Cannot be Created in Dashboard After Upgrade	24
2.2.20	OCNADD Services Status Not Correct in Dashboard After Upgrade	25
2.2.21	Alarm for Unable to Transfer File to SFTP Server for Export Configuration	26
2.2.22	OCNADD Two-Site Redundancy Troubleshooting Scenarios	26
2.2.23	Resource Allocation Challenges During DD Installation on OCI	30
2.2.24	Transaction Filter Update Takes Few Seconds to Reflect Changes in Traffic Processing	31
2.2.25	Connection Timeout Errors in Configuration Service Logs	31

2.2.26	High Latency in Adapter Feed Due to High Disk Latency	32
2.2.27	Kafka Broker Goes into "CrashLoopBackOff" State During Very Low-Throughput Traffic	34
2.2.28	Database Space Full Preventing Configuration and Subscription from Being Saved in Database	35
2.2.29	Invalid Subscription Entry in the Subscription Table	37
2.2.30	Correlation Configuration Does Not Get Deployed Post Rollback	39
2.2.31	Cleanup of Redundant xDR Tables	41
2.2.32	Filter service does not receive any notification update after Migration	41
2.3	Troubleshooting Traffic Stability in OCNADD	41
2.4	CNLB Troubleshooting Scenarios	45
2.5	Extended Storage with Druid Troubleshooting Scenarios	48

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 List of Alerts

5.1	System Level Alerts	1
5.2	Application Level Alerts	17

Preface

- [Documentation Accessibility](#)
- [Diversity and Inclusion](#)
- [Conventions](#)

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

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

What's New in This Guide

This section lists the documentation updates for Release 25.2.2xx.

Release 25.2.200 - G49000-01, December 2025

Major Architecture Restructuring: Data Director Group Classification

This release introduces a fundamental transformation in the OCNADD architecture. The previously singular Worker Group has been decoupled and re-architected into two specialized functional groups: the Relay Agent Group and the Mediation Group.

This architectural shift impacts the entire system workflow and is reflected throughout this document.

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*.

Note

The performance and capacity of the OCNADD system may vary based on the call model, Feature/Interface configuration, and underlying CNE and hardware environment.

1.2 References

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

- *Oracle Communications Network Analytics Suite Release Notes*
- *Oracle Communications Network Analytics Suite Licensing Information User Manual*
- *Oracle Communications Network Analytics Automated Testing Suite Guide*
- *Oracle Communications Network Analytics Suite Security Guide*
- *Oracle Communications Network Analytics Data Director Installation, Upgrade, and Fault Recovery Guide*
- *Oracle Communications Network Analytics Data Director User Guide*
- *Oracle Communications Network Analytics Data Director Outbound Interface Specification Guide*
- *Oracle Communications Network Analytics Data Director Benchmarking Guide*
- *Oracle Communications Network Analytics Data Director Diameter User Guide*

- *Oracle Communications Network Analytics Data Director vCollector Installation Guide*
- *Oracle Communications Cloud Native Core, cnDBTier Installation, Upgrade, and Fault Recovery Guide*
- *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide*
- *Oracle Communications Cloud Native Core, OCI Deployment Guide*
- *Oracle Communication Certificate Manager Installation, Upgrade and Fault Recovery Guide*
- *Oracle Communication Certificate Manager User 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-apiserver. The instructions provided in this document are as per the Oracle Communications Cloud Native Environment (OCCNE) version of kube-apiserver.

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
```

Note

Here, the namespace could be of the management group, relayagent group or mediation group.

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
```

Note

The namespace should be used for the corresponding group management, relayagent or mediation

- 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>
```

Note

Here, the management group namespace should be used.

Example:

For relayagent:

```
<relayAgentGroupName> =  
<siteName>:<workerGroupName>:<relayAgentNamespace>:<relayAgentClusterName>
```

```
kubectl exec -it pod/ocnaddmanagementgateway-xxxx -n dd-mgmt -- curl  
'http://ocnaddmanagementgateway.dd-mgmt:12889/ocnadd-configuration/v2/  
topic?relayAgentGroup=BLR:ddworker1:dd-relay-ns:dd-relay-cluster'
```

If mTLS is enabled, then use the following command:

```
kubectl exec -it pod/ocnaddmanagementgateway-xxxx -n dd-mgmt --curl -k --  
location --cert-type P12 --cert /var/securityfiles/keystore/  
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD "https://  
ocnaddmanagementgateway.ddmgmt:12889/ocnadd-configuration/v2/topic?  
relayAgentGroup=BLR:ddworker1:dd-relay-ns:dd-relay-cluster"
```

For mediation group:

```
<mediationGroup> =  
<siteName>:<workerGroupName>:<mediationNamespace>:<mediationClusterName>
```

```
kubectl exec -it pod/ocnaddmanagementgateway-xxxx -n dd-mgmt -- curl  
'http://ocnaddmanagementgateway.dd-mgmt:12889/ocnadd-configuration/v2/  
topic?mediationGroup=BLR:ddworker1:dd-med-ns:dd-mediation-cluster'
```

If mTLS is enabled, then use the following command:

```
kubectl exec -it pod/ocnaddmanagementgateway-xxxx -n dd-mgmt --curl -k --  
location --cert-type P12 --cert /var/securityfiles/keystore/  
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD "https://  
ocnaddmanagementgateway.ddmgmt:12889/ocnadd-configuration/v2/topic?  
mediationGroup=BLR:ddworker1:dd-med-ns:dd-mediation-cluster"
```

Note

These commands are in their simple format and display the logs only if ocnaddconfiguration and ocnaddadmin pods are deployed.

The list of URIs for accessing Kafka Topic information:

For Relay Agent Kafka Cluster, the API endpoint is:

```
<MgmtGWIP:MgmtGW  
Port>.<mgmt-ns>/ocnadd-configuration/v2/topic?  
relayAgentGroup=<relayAgentGroupName>
```

where,

```
<relayAgentGroupName> =  
<siteName>:<workerGroupName>:<relayAgentNamespace>:<relayAgentClusterName>
```

For Mediation Kafka Cluster, the API endpoint is:

```
<MgmtGWIP:MgmtGW
  Port>.<mgmt-ns>/ocnadd-configuration/v2/topic?
mediationGroup=<mediationGroup>
```

where,

```
<mediationGroup> =
<siteName>:<workerGroupName>:<mediationNamespace>:<mediationClusterName>
```

```
<MgmtGWIP:MgmtGW Port>/ocnadd-configuration/v2/topic/?
relayAgentGroup=<relayAgentGroupName>
<MgmtGWIP:MgmtGW Port>/ocnadd-configuration/v2/topic/<topicName>?
relayAgentGroup=<relayAgentGroupName>
<MgmtGWIP:MgmtGW Port>/ocnadd-configuration/v2/broker/bootstrap-server?
relayAgentGroup=<relayAgentGroupName>
```

Similarly, other APIs can be accessed for the configuration.

Application Checklist

Logs Verification

Note

The below procedures should be verified or run corresponding to the applicable worker group (relayagent or mediation) or the management group.

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

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

Note

The namespace should be used for the corresponding group: management, relayagent, or mediation.

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

Example:

```
# kubectl -n dd-relay logs -f $(kubectl -n dd-relay get pods -o name | cut -
d'/' -f 2 | grep nrffaggregation)
```

The above command displays the logs of the `ocnaddnrffaggregation` service.

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

```
# kubectl logs -n dd-relay <nrffaggregation pod name> | grep <pattern>
```

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. Edit the Relay agent or Mediation custom values file.
 - b. Change the value of below parameter to 1:
 - i. `offsetsTopicReplicationFactor: 1`
 - ii. `transactionStateLogReplicationFactor: 1`
 - c. Scale down the Kafka and Kraft controller deployment by modifying the following lines in the corresponding worker group (relayagent or mediation) helm chart and custom values:

```
ocnaddkafka:
  enabled: false
```

- d. Perform helm upgrade for the worker group:
Refer to section "Update worker group service parameters" from *Oracle Communications Network Analytics Data Director User Guide*
- e. Delete PVCs for Kafka and Kraft controller using the following commands, where <namespace> is the relayagent or mediation group namespace:

```
kubectl delete pvc -n <namespace> kafka-volume-kafka-broker-0
kubectl delete pvc -n <namespace> kafka-volume-kafka-broker-1
kubectl delete pvc -n <namespace> kafka-volume-kafka-broker-2
kubectl delete pvc -n <namespace> kraft-broker-security-kraft-
controller-0
kubectl delete pvc -n <namespace> kraft-broker-security-kraft-
controller-1
kubectl delete pvc -n <namespace> kraft-broker-security-kraft-
controller-2
```

- f. Modify the value of the following parameter to true in the corresponding worker group (relayagent or mediation) helm chart and custom values:

```
ocnaddkafka:
  enabled: true
```

- g. Perform helm upgrade for the worker group:
Refer to section "Update worker group service parameters" from *Oracle Communications Network Analytics Data Director User Guide*

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/Kraft-controller 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 percent 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-mediation-custom-values.yaml or ocnadd-relayagent-custom-values.yaml corresponding to the relayagent or mediation group.

Parameter name:

```
ocnaddkafka.ocnadd.kafkaBroker.resource
```

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

Refer to section "Update worker group service parameters" from *Oracle Communications Network Analytics Data Director User Guide*.

Kafka ACLs: Identifying the Network IP "Host" ACLs in Kafka Feed**Problem**

User is unable to identify the network IP **Host** ACLs in Kafka Feed.

Solution

The following procedure can be referred to when the user is unable to identify the network IP address.

Adding network IP address for Host ACL

This set of instructions explains how to add a network IP address to the host ACL. The procedure is illustrated using the following example configuration:

- Kafka Feed Name: demofeed
- Kafka ACL User: joe
- Kafka Client Hostname: 10.1.1.15

① Note

The steps below should be run corresponding to the mediation group against which the Kafka feed is created or modified.

Retrieving Current ACLs

To obtain the current ACLs configured for the demofeed feed on the Kafka service, run the following command from managementgateway within the OCNADD management group deployment:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/keystore/
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --request GET 'https://
ocnaddmediationgateway.dd-mediation-ns:12890/ocnadd-admin/v2/<mediationGroup>/
acls'
```

Example:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/keystore/
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --request GET 'https://
ocnaddmediationgateway.dd-mediation-ns:12890/ocnadd-admin/v2/BLR:ddworker1:dd-
mediation-ns:dd-mediation-cluster/acls'
```

The output will be in JSON format and might look like this:

```
["(pattern=ResourcePattern(resourceType=GROUP, name=demofeed,
patternType=LITERAL), entry=(principal=User:joe, host=10.1.1.15,
operation=READ, permissionType=ALLOW))",
"(pattern=ResourcePattern(resourceType=TOPIC, name=MAIN,
patternType=LITERAL), entry=(principal=User:joe, host=10.1.1.15,
operation=READ, permissionType=ALLOW))"]
```

Authorization Error in Kafka Logs

With the Kafka feed demofeed configured with ACL user joe and client host IP address 10.1.1.15, Kafka reports the following authorization error:

```
[2023-07-31 05:34:22,063] INFO Principal = User:joe is Denied Operation =
Read from host = 10.1.1.0 on resource = Group:LITERAL:demofeed for request =
JoinGroup with resourceRefCount = 1 (kafka.authorizer.logger)
```

In the above output, host ACL is allowing the specific client IP address 10.1.1.15, whereas the Kafka server is expecting an ACL entry for the network IP 10.1.1.0 as well.

Steps to Create Network IP Address ACL

Step 1: Check Kafka Logs

To identify the network IP address that Kafka is denying against the configured feed:

Run:

```
kubectl logs -n <namespace> -c kafka-broker kafka-broker-1 -f
```

Example:

```
kubectl logs -n dd-med -c kafka-broker kafka-broker-1 -f
```

Look for traces similar to:

```
Principal = User:joe is Denied Operation = Read from host = 10.1.1.0 on
resource = Group:LITERAL:demofeed...
```

Identify the denied IP address; in this example, it is 10.1.1.0.

Step 2: Create Host ACL for Network IP

Access any ocnaddmanagementgateway pod:

```
kubectl exec -it ocnaddmanagementgateway-xxxx -n dd-mgmt -- bash
```

Run the curl commands to configure the host network IP ACLs:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/keystore/
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD \
--request POST 'https://ocnaddmanagementgateway.ddmgt:12889/ocnadd-
configuration/v3/client-acl?mediationGroup=<mediationGroup>' \
--header 'Content-Type: application/json' \
--data-raw '{ "principal": "joe", "hostName": "10.1.1.0", "resourceType":
"TOPIC", "resourceName": "MAIN", "aclOperation": "READ" }'
```

```
curl -k --location --cert-type P12 --cert /var/securityfiles/keystore/
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD \
--request POST 'https://ocnaddmanagementgateway.ddmgt:12889/ocnadd-
configuration/v3/client-acl?mediationGroup=<mediationGroup>' \
--header 'Content-Type: application/json' \
```

```
--data-raw '{ "principal": "joe", "hostName": "10.1.1.0", "resourceType":  
"GROUP", "resourceName": "demofeed", "aclOperation": "READ" }'
```

Step 3: Verify ACLs

Use the following command to verify updated ACLs:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/keystore/  
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --request GET 'https://  
ocnaddmediationgateway.dd-mediation-ns:12890/ocnadd-admin/v2/<mediationGroup>/  
acls'
```

Example:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/keystore/  
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --request GET 'https://  
ocnaddmediationgateway.dd-mediation-ns:12890/ocnadd-admin/v2/BLR:ddworker1:dd-  
mediation-ns:dd-mediation-cluster/acls'
```

Expected output example:

```
[{"(pattern=ResourcePattern(resourceType=GROUP, name=demofeed,  
patternType=LITERAL), entry=(principal=User:joe, host=10.1.1.0,  
operation=READ, permissionType=ALLOW))",  
"(pattern=ResourcePattern(resourceType=GROUP, name=demofeed,  
patternType=LITERAL), entry=(principal=User:joe, host=10.1.1.15,  
operation=READ, permissionType=ALLOW))",  
"(pattern=ResourcePattern(resourceType=TOPIC, name=MAIN,  
patternType=LITERAL), entry=(principal=User:joe, host=10.1.1.0,  
operation=READ, permissionType=ALLOW))",  
"(pattern=ResourcePattern(resourceType=TOPIC, name=MAIN,  
patternType=LITERAL), entry=(principal=User:joe, host=10.1.1.15,  
operation=READ, permissionType=ALLOW))"]
```

Producer Unable to Send traffic to OCNADD when an External Kafka Feed is enabled

Problem

Producer is unable to send traffic to OCNADD when an External Kafka Feed is enabled.

Solution

Follow the below steps to debug and investigate if the producer is unable to send traffic to OCNADD when ACL is enabled and there are authorization errors coming in producer NF logs.

Debug and Investigation Steps:

① Note

The below steps should be run corresponding to the Relay Agent group against which the Kafka feed is being enabled.

1. Begin by creating the `admin.properties` file within the Kafka broker, following Step 2 of "Update SCRAM Configuration with Users" as outlined in the *Oracle Communications Network Analytics Data Director User Guide*.
2. If the producer's security protocol is SASL_SSL (port 9094), verify whether the users have been created in SCRAM. Use the following command for verification in the mediation group:

```
./kafka-configs.sh --bootstrap-server kafka-broker:9094 --describe --
entity-type users --command-config ../../admin.properties
```

If no producer's SCRAM ACL users are found, see the "Prerequisites for External Consumers" section in the *Oracle Communications Network Analytics Data Director User Guide* to create the necessary Client ACL users.

3. In case the producer's security protocol is SSL (port 9093), ensure that the Network Function (NF) producer's certificates have been correctly generated as per the instructions provided in the *Oracle Communications Network Analytics Suite Security Guide*.
4. Check whether the producer client ACLs have been set up based on the configured security protocol (SASL_SSL or SSL) in the NF Kafka Producers. To verify this:
 - a. Access `ocnaddmanagementgateway` pod from the management group:

```
kubectl exec -it ocnaddmanagementgateway-xxxx -n dd-mgmt -- bash
```

- b. Run the following `curl` command to list all the ACLs:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/keystore/
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --request GET 'https://
ocnaddrelayagentgateway.dd-relayagent-ns:12888/ocnadd-admin/v2/
<relayAgentGroup>/acls'
```

Example:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/keystore/
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --request GET 'https://
ocnaddrelayagentgateway.dd-relayagent-ns:12888/ocnadd-admin/v2/
BLR:ddworker1:dd-relayagent-ns:dd-relayagent-cluster/acls'
```

The expected output might resemble the following example, indicating With Feed Name: demofeed, ACL user: joe, Host Name: 10.1.1.15, Network IP: 10.1.1.0:

```
["(pattern=ResourcePattern(resourceType=GROUP, name=demofeed,
patternType=LITERAL), entry=(principal=User:joe, host=10.1.1.0,
operation=READ,
permissionType=ALLOW))", "(pattern=ResourcePattern(resourceType=GROUP,
name=demofeed, patternType=LITERAL), entry=(principal=User:joe,
host=10.1.1.15, operation=READ,
permissionType=ALLOW))", "(pattern=ResourcePattern(resourceType=TOPIC,
name=MAIN, patternType=LITERAL), entry=(principal=User:joe,
host=10.1.1.0, operation=READ,
permissionType=ALLOW))", "(pattern=ResourcePattern(resourceType=TOPIC,
name=MAIN, patternType=LITERAL), entry=(principal=User:joe,
host=10.1.1.15, operation=READ, permissionType=ALLOW))"]
```

5. If no ACLs are found as observed in step 4, follow the "Create Client ACLs" section provided in the *Oracle Communications Network Analytics Data Director User Guide* to establish the required ACLs.

By following these steps, you will be able to diagnose and address issues related to the producer's inability to send traffic to OCNADD when an External Kafka Feed is enabled and ACL-related authorization errors are encountered.

External Kafka Consumer Unable to consume messages from DD

Problem

External Kafka Consumer is unable to consume messages from OCNADD.

Solution

If you are experiencing issues where an external Kafka consumer is unable to consume messages from OCNADD, especially when ACL is enabled and unauthorized errors are appearing in the Kafka feed's logs, follow the subsequent steps for debugging and investigation:

Debug and Investigation Steps:

① Note

The below steps should be run corresponding to the mediation group against which the Kafka feed is created or modified.

1. Verify that ACL Users created for the Kafka feed, along with SCRAM users, are appropriately configured in the JAAS config by executing the following command:

```
./kafka-configs.sh --bootstrap-server kafka-broker:9094 --describe --
entity-type users --command-config ../../admin.properties
```

2. Validate that the Kafka feed parameters have been correctly configured in the consumer client. If not, ensure proper configuration and perform an upgrade on the Kafka feed's consumer application.
3. Inspect the logs of the external consumer application.
 - If you encounter an error related to *XYZ Group authorization failure* in the consumer application logs, follow these steps:
 - Access ocnaddmanagementgateway pod from the management group:

```
kubectl exec -it ocnaddmanagementgateway-xxxx -n dd-mgmt -- bash
```

- Run the curl command below to retrieve ACLs information and verify the existence of ACLs for the Kafka feed:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/
keystore/serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --request
GET 'https://ocnaddmediationgateway.dd-mediation-ns:12890/ocnadd-
admin/v2/<mediationGroup>/acls'
```

Example:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/
keystore/serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --request
GET '
https://ocnaddmediationgateway.dd-mediation-ns:12890/ocnadd-
admin/v2/BLR:ddworker1:dd-mediation-ns:dd-mediation-cluster/acls'
```

Sample output with Feed Name: demofeed, ACL user: joe, Host Name: 10.1.1.15, Network IP: 10.1.1.0:

```
["(pattern=ResourcePattern(resourceType=GROUP, name=demofeed,
patternType=LITERAL), entry=(principal=User:joe, host=10.1.1.0,
operation=READ,
permissionType=ALLOW))", "(pattern=ResourcePattern(resourceType=GROUP
, name=demofeed, patternType=LITERAL), entry=(principal=User:joe,
host=10.1.1.15, operation=READ,
permissionType=ALLOW))", "(pattern=ResourcePattern(resourceType=TOPIC
, name=MAIN, patternType=LITERAL), entry=(principal=User:joe,
host=10.1.1.0, operation=READ,
permissionType=ALLOW))", "(pattern=ResourcePattern(resourceType=TOPIC
, name=MAIN, patternType=LITERAL), entry=(principal=User:joe,
host=10.1.1.15, operation=READ, permissionType=ALLOW))"]
```

- If no ACL is found for the Kafka feed with the resource type GROUP, run the following curl command to create the Group resource type ACLs:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/
keystore/serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --request
POST 'https://ocnaddmanagementgateway.ddmgmt:12889/ocnadd-
configuration/v3/client-acl?mediationGroup=<mediationGroup>' --
header 'Content-Type: application/json' --data-raw '{ "principal":
"<ACL-USER-NAME>", "resourceType": "GROUP", "resourceName": "<KAFKA-
FEED-NAME>", "aclOperation": "READ" }'
```

4. If you encounter an error related to *XYZ TOPIC authorization failure* in the consumer application logs, follow these steps:

- a. Access ocnaddmanagementgateway pod from the management group:

```
kubectl exec -it ocnaddmanagementgateway -n dd-mgmt -- bash
```

- b. Run the curl command below to retrieve ACLs information and verify the existence of ACLs for the Kafka feed:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/keystore/
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --request GET 'https://
ocnaddmediationgateway.dd-mediation-ns:12890/ocnadd-admin/v2/
<mediationGroup>/acls'
```

Example:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/keystore/
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --request GET 'https://
```

```
ocnaddmediationgateway.dd-mediation-ns:12890/ocnadd-admin/v2/
BLR:ddworker1:dd-mediation-ns:dd-mediation-cluster/acls'
```

Sample output with Feed Name: demofeed, ACL user: joe, Host Name: 10.1.1.15, Network IP: 10.1.1.0:

```
[{"pattern=ResourcePattern(resourceType=GROUP, name=demofeed,
patternType=LITERAL), entry=(principal=User:joe, host=10.1.1.0,
operation=READ,
permissionType=ALLOW))", {"pattern=ResourcePattern(resourceType=GROUP,
name=demofeed, patternType=LITERAL), entry=(principal=User:joe,
host=10.1.1.15, operation=READ,
permissionType=ALLOW))", {"pattern=ResourcePattern(resourceType=TOPIC,
name=MAIN, patternType=LITERAL), entry=(principal=User:joe,
host=10.1.1.0, operation=READ,
permissionType=ALLOW))", {"pattern=ResourcePattern(resourceType=TOPIC,
name=MAIN, patternType=LITERAL), entry=(principal=User:joe,
host=10.1.1.15, operation=READ, permissionType=ALLOW))"]
```

- c. If no ACL is found for the Kafka feed with the resource type TOPIC, run the following curl command to create the TOPIC resource type ACLs:

```
curl -k --location --cert-type P12 --cert /var/securityfiles/keystore/
serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --request POST 'https://
ocnaddmanagementgateway.ddmgmt:12889/ocnadd-configuration/v3/client-acl?
mediationGroup=<mediationGroup>' --header 'Content-Type: application/
json' --data-raw '{ "principal": "<ACL-USER-NAME>", "resourceType":
"TOPIC", "resourceName": "MAIN", "aclOperation": "READ" }'
```

Database Error During Kafka Feed Update

Problem:

When attempting to update a Kafka feed, you may encounter an error similar to the following:

Error Message: Updating the Kafka feed in the database has failed.

Solution:

To address this issue, retry the Kafka feed update using the Update Kafka feed option from OCNADD UI with the same information as in the previous attempt.

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-**<group-name>**-custom-values.yaml* File

Problem

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

where group-name can be management-group, relayagent-group, or mediation-group.

Error Code or Error Message

When you run `kubectl get pods -n <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 specific `ocnadd-<group-name>-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 <namespace>` command to verify if all the pods are in Running state.

2.2.2 Failed Helm Installation/Upgrade Due to Prometheus Rules Applying Failure

Scenario:

Helm installation or upgrade fails due to Prometheus rules applying failure.

Problem:

Helm installation or upgrade might fail if Prometheus service is down or Prometheus pods are not available during the helm installation or upgrade of the OCNADD.

Error Code or Error Message:

Error: UPGRADE FAILED: cannot patch "ocnadd-<mgmt|relayagent|mediation>-alerting-rules" with kind PrometheusRule: 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:

Perform the following steps to proceed with the OCNADD helm install or upgrade:

1. Move the `ocnadd-relayagent-alerting-rules.yaml`, `ocnadd-mediation-alerting-rules.yaml`, and `ocnadd-mgmt-alerting-rules.yaml` from the `<chart_path>/helm-charts/charts/<groupName>/templates` to some other directory outside the OCNADD charts.
2. Continue with the helm install or upgrade.
3. Run the following command to verify if the status of all the pods are running:

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

The OCNADD Prometheus alerting rules must be applied again when the Prometheus service and pods are available back in service. Ensure to apply the alerting rules using the Helm upgrade procedure itself by moving back the `ocnadd-relayagent-alerting-rules.yaml`, `ocnadd-mediation-alerting-rules.yaml`, and `ocnadd-mgmt-alerting-rules.yaml` files into

the templates folder of their respective chart directory (<chart_path>/helm-charts/charts/<groupName>/templates).

2.2.3 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`.

<ocnadd_namespace> can be of management, relayagent, or mediation group.

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*.

2.2.4 Continuous Restart of Pods

Problem

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

Error Code or Error Message

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

Solution

1. Verify MySQL connectivity.
2. 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*.
3. Describe the pod to check more details on the error, and troubleshoot further based on the reported error.
4. Check the pod log for any error, and troubleshoot further based on the reported error.

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:

1. Run the following command to verify the data feeds:

```
kubectl get po -n `ocnadd-med`
```

2. 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 `ocnadd-med`
```

3. If the error message is present, run the following command:

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

2.2.6 ocnadd-custom-values.yaml File Parse Failure

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

Problem

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

Error Code or Error Message

Error: failed to parse ocnadd-<group-name>-custom-values.yaml: error converting YAML to JSON: yaml

Symptom

When parsing the ocnadd-<group-name>-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 <group_namespace>` the broker pod's status might be Error or CrashLoopBackOff, and it might keep restarting continuously, with *no disk space left on the device* errors in the pod logs.

where, <group_namespace> is for relayagent or mediation group.

Solution

1. Delete the StatefulSet (STS) deployments of the brokers. Run `kubectl get sts -n <group_namespace>` to obtain the StatefulSets in the namespace.
2. Delete the STS deployments of the services with the disk full issue. For example, run:

```
kubectl delete sts -n <group_namespace> kafka-broker1 kafka-broker2
```

3. After deleting the STS of the brokers, delete the PVC. Delete the PVCs in the namespace that were used by the Kafka brokers. Run:

```
kubectl get pvc -n <group_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 `kraft controller`, and delete them. To delete the PVCs, run:

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

For example:

For a three-broker setup in mediation worker group namespace `dd-med`, you will need to delete these PVCs:

```
kubectl delete pvc -n dd-med kafka-volume-kafka-broker-0 kafka-volume-kafka-broker-1 kafka-volume-kafka-broker-2 kraft-broker-security-kraft-controller-0 kraft-broker-security-kraft-controller-1 kraft-broker-security-kraft-controller-2
```

2.2.8 Kafka Brokers Continuously Restart After the Disk is Full

Problem

There is no disk space left on either the broker or the KRaft controller in the corresponding worker group (Relay Agent or Mediation).

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.

where <ocnadd_namespace> is for relayagent or mediation group.

Solution

Note

The below steps should be run corresponding to the worker (relayagent or mediation) group against which Kafka is reporting the disk full error.

1. **Delete the StatefulSet (STS) deployments of the brokers** (explained using mediation group as an example).

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

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

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

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

For example, for a three-broker setup:

```
kubectl delete sts -n dd-med kafka-broker kraft-controller
```

2. **Delete the PVCs in the namespace that are used by the removed Kafka brokers.**

- To get the PVCs in the namespace:

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

- The number of PVCs used will be based on the number of brokers deployed. Select the PVCs that have the name kafka-broker or kraft-controller and delete them.
- To delete PVCs, run:

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

For example, for a three-broker setup in namespace dd-med, you must delete these PVCs:

```
kubectl delete pvc -n dd-med kafka-volume-kafka-broker-0 kafka-volume-kafka-broker-1 kafka-volume-kafka-broker-2 kraft-broker-security-kraft-controller-0 kraft-broker-security-kraft-controller-1 kraft-broker-security-kraft-controller-2
```

3. Once the STS and PVCs are deleted, edit the custom values file of mediation (or relay agent) group.

4. **To increase the storage**, edit the field `pvcClaimSize`. For PVC storage recommendations, see the *Oracle Communications Network Analytics Data Director Benchmarking Guide*.
5. **Upgrade the Helm chart after increasing the PVC size.**
Refer to the section *Update worker group service parameters* from the *Oracle Communications Network Analytics Data Director User Guide*.
6. **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.

where `<ocnadd_namespace>` is for relayagent or mediation group.

Solution

The Kafka brokers wait for a maximum of 3 minutes for the Kraft controllers to come online before they are started. If the Kraft controller cluster does not come online within the given interval, the broker will start before the Kraft controller and will error out as it does not have access to the Kraft controller. The Kraft controller may start after the 3-minute interval, as the node may take more time to pull the images due to network issues. Therefore, when the Kraft controller 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

The output of the command:

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 required to kill each connection.

For 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 generated KILL commands on each SQL node.

2.2.11 Upgrade fails due to Database MaxNoOfAttributes exhausted

Scenario:

Upgrade fails due to Database MaxNoOfAttributes exhausted

Problem:

Helm upgrade may fail due to maximum number for attributes allowed to be created has reached maximum limit.

Error Code or Error Message:

```
Executing::::: /tmp/230300001.sql mysql: [Warning] Using a password on the
command line interface can be insecure. ERROR 1005 (HY000) at line 51: Can't
create destination table for copying alter table (use SHOW WARNINGS for more
info). error in executing upgrade db scripts
```

Solution:

Delete few database schemas that are not being used or the ones which are stale.

For example, from MySQL prompt, drop database xyz;

Note

Dropping unused or stale database schemas is a valid approach. However, exercise caution when doing this to ensure you are not deleting important data. Make sure to have proper backups before proceeding.

2.2.12 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.

2.2.13 Data Feeds Do Not delete after Rollback

Scenario:

During an upgrade from one version to another, if any data feeds (Adapter/Correlation) are spawned in the new version, the newly created resources may not be deleted even after a rollback.

Problem:

Upon rolling back to an older version, all resources in OCNADD should revert to their previous states. Consequently, any new resources generated in the upgraded version should be deleted since they did not exist in older versions.

Solution:

If they failed to get deleted on their own, use the following command to delete all the resources manually::

```
$ kubectl delete service,deploy,hpa <adapter-name> -n dd-mediation
```

2.2.14 Data Feeds Do Not Restart after Rollback

Scenario:

When Data Feeds (Adapters/Correlation) are created in the source release, and an upgrade is performed to the latest release, later, if a rollback to the previous release is executed, the restart of respective data feed pods is expected.

Problem:

The Data feed pods do not restart, and the Admin Service reports the following exception:

```
io.fabric8.kubernetes.client.KubernetesClientException: Operation: [update]
for kind: [Deployment] with name: [app-http-adapter] in namespace: [ocnadd-
deploy] failed.
    at
io.fabric8.kubernetes.client.KubernetesClientException.launderThrowable(Kubern
etesClientException.java:159) ~[kubernetes-client-api-6.5.1.jar!/:?]
    at
io.fabric8.kubernetes.client.dsl.internal.HasMetadataOperation.update(HasMetad
ataOperation.java:133) ~[kubernetes-client-6.5.1.jar!/:?]
    at
io.fabric8.kubernetes.client.dsl.internal.HasMetadataOperation.update(HasMetad
```

```

ataOperation.java:109) ~[kubernetes-client-6.5.1.jar!/:?]
    at
io.fabric8.kubernetes.client.dsl.internal.HasMetadataOperation.update(HasMetad
ataOperation.java:39) ~[kubernetes-client-6.5.1.jar!/:?]
    at
com.oracle.cgbu.cne.ocnadd.admin.svc.service.consumerAdapter.ConsumerAdapterSe
rvice.lambda$upgradeConsumerAdapterOnStart$1(ConsumerAdapterService.java:1068)
~[classes!/:2.2.3]
    at java.util.ArrayList.forEach(ArrayList.java:1511) ~[?:?]
    at
com.oracle.cgbu.cne.ocnadd.admin.svc.service.consumerAdapter.ConsumerAdapterSe
rvice.upgradeConsumerAdapterOnStart(ConsumerAdapterService.java:1022)
~[classes!/:2.2.3]

```

Solution:

Restart the Admin Service after the rollback, which in turn will restart the data feed pods to revert them to the older version.

2.2.15 Third-Party Endpoint DOWN State and New Feed Creation

Scenario:

When a third-party endpoint is in a DOWN state and a new third-party Feed (HTTP/SYNTHETIC) is created with the "Proceed with Latest Data" configuration, data streaming is expected to resume once the third-party endpoint becomes available and connectivity is established.

Problem

A third-party endpoint is in a DOWN state and a new third-party Feed (HTTP/SYNTHETIC) is created.

Solution

It is recommended to use "Resume from Point of Failure" configuration in case of third-party endpoint unavailability during feed creation.

2.2.16 Adapter Feed Not Coming Up After Rollback

Scenario:

When Data Feeds are created in the previous release and an upgrade is performed to the latest release, and in the latest release both the data feeds are deleted and rollback was carried out to the previous release. Then the feeds that are created in the older Release should have come back after Rollback.

Problem

- After the Rollback only one of the Data Feeds is created back.
- The Data Feeds get created after the rollback, however, it is stuck in "Inactive" state.

Solution:

1. After the rollback, clone the feeds and delete the old feeds.
2. Update the Cloned Feeds with the Data Stream Offset as EARLIEST to avoid data loss.

2.2.17 Adapters pods are in the "init:CrashLoopBackOff" error state after rollback

Scenario:

This issue occurs when the adapter pods, created before or after the upgrade, encounter errors due to missing fixes related to data feed types available only in the latest releases.

Problem:

Adapter pods are stuck in the `init:CrashLoopBackOff` state after a rollback from the latest release to an older release.

Solution Steps:**1. Delete the Adapter Manually:**

Run the following command to delete the adapter:

```
kubectl delete service,deploy,hpa <adapter-name> -n dd-med
```

2. Clone or Recreate the Data Feed:

Clone or recreate the data feed again with the same configurations. While creating the data feed, the `Data Stream Offset` option can be set as `EARLIEST` to avoid data loss.

2.2.18 Feed/Filter Configurations are Missing From Dashboard After Upgrade

Scenario:

After OCNADD is upgraded from the previous release to the latest release, it may be observed that some feed or filter configurations are missing from dashboard.

Problem:

When this issue occurs after upgrade, the configuration service log will have the following message *"Table definition has changed, please retry transaction"*.

Solution:**1. Delete the configuration service pod manually:**

```
kubectl delete pod -n ocnadd-mgmt <configuration-name>
```

2. Check the logs of the new configuration service pod, if *"Table definition has changed, please retry transaction"* is still present in the log, repeat step 1.

2.2.19 Kafka Feed Cannot be Created in Dashboard After Upgrade

Scenario:

After OCNADD is upgraded from the previous release to the latest release, it may be observed that new Kafka feed configurations cannot be created in dashboard.

Problem:

When this issue occurs after upgrade, the configuration service log will have the following message *"Table definition has changed, please retry transaction"*.

Solution:

1. Delete the configuration service pod manually:

```
kubectl delete pod -n ocnadd-mgmt <configuration-name>
```

2. Check the logs of the new configuration service pod, if *"Table definition has changed, please retry transaction"* is still present in the log, repeat step 1.

2.2.20 OCNADD Services Status Not Correct in Dashboard After Upgrade

Scenario:

After upgrading OCNADD from the previous release to the latest release, you may observe that the OCNADD microservices do not reflect the correct status.

Problem:

After the upgrade, OCNADD microservices may fail to register their health profiles. The service logs will show repeated retry attempts similar to the following:

For StatefulSet deployments (e.g., kraft-controller, Kafka broker), use:

```
kubectl exec -it -n dd-med kafka-broker-0 -- bash  
cat extService.txt | grep -i retry
```

For standard deployments, use:

```
kubectl logs -n <namespace> <ocnadd-service-podname> -f | grep -i retry
```

(namespace may be management, relayagent, or mediation group)

Sample Service Log:

```
ERROR ... Health Profile Registration is not successful, retry number 0  
ERROR ... Health Profile Registration is not successful, retry number 1  
ERROR ... Health Profile Registration is not successful, retry number 2  
...  
ERROR ... Health Profile Registration is not successful, retry number 12
```

Solution:

1. **Restart all OCNADD microservices** using the following command.
Run this separately for *each worker group* and the *management group* if deployed in centralized mode:

```
kubectl rollout restart -n <namespace> deployment,sts
```

(namespace can be of management, relayagent or mediation group)

2. **Verify service health:**
Check the status of all OCNADD microservices in the OCNADD UI.

Each service should eventually show **Active** status.

2.2.21 Alarm for Unable to Transfer File to SFTP Server for Export Configuration

Scenario:

While performing the Export Configuration, the user encounters the following alarm:

Alarm: OCNADD02009: SFTP service is unreachable

Solution:

Verify the following points to manage this alarm:

1. **Check the SFTP server IP:** Ensure the SFTP server IP is correct. Verify that the provided IP is neither IPv6 nor FQDN.
2. **Check the Storage Path in the File Location:** Ensure the file path is correct. The file path should be relative; absolute paths are not supported.

2.2.22 OCNADD Two-Site Redundancy Troubleshooting Scenarios

General Troubleshooting

This section provides information to address troubleshooting issues faced post two-site redundancy configuration. The issues and fixes mentioned in this section are related to alarms. After the site configuration is done, the user should move to the Alarm Dashboard and verify if any newly generated alarms are raised for two-site redundancy.

OCNADD02006: Mate Site Down

TYPE: COMMUNICATION

SEVERITY: MAJOR

Scenario:

When either CNE cluster of the mate site or mate site worker group is down or not accessible.

Problem Description:

The CNE cluster or worker group in the mate site is down or inaccessible.

Troubleshooting Steps:

1. Check the status of worker group services in the mate site.
2. Check if the CNE cluster is accessible or not.
3. Verify if nodes are in an isolated zone or not.
4. Run sync after the issue has been fixed.

OCNADD02000: Loss of Connection

TYPE: COMMUNICATION

SEVERITY: MAJOR

Scenario:

Initial connection with Redundancy Agent of mate site is not established.

Problem Description:

Connection could not be established with the mate site Redundancy Agent service due to egress annotation not enabled or cluster communication not supported.

Troubleshooting Steps:

1. Check if LoadBalancer IP of mated site is able to communicate with each other.

- a. Run Exec command in the mate site Redundancy Agent pod:

```
kubectl exec -ti -n <namespace> <mate_redundancyagent_pod_name> -- bash
```

- b. Run the following cURL command:

```
curl -k --cert-type P12 --cert /var/securityfiles/keystore/
serverKeyStore.p12:$OCNADD_SSL_KEY_STORE_PASSWORD --location -X GET
https://<mate_redundancy_agent_external_ip>:13000
```

2. Verify if CNE version requires egress annotation or not.
3. Check the Redundancy Agent service pod logs in the mate site.
4. If there are error logs in Redundancy Agent, then:
 - a. Go to custom values of management charts.
 - b. Set the value of `global.ocnaddredundancyagent.egress` to `true`.
 - c. Run Helm chart upgrade:
 - i. Go to the custom values of the management charts.
 - ii. Edit `global.ocnaddmanagement.ocnaddredundancyagent.egress` and set it to `true`.
 - iii. Run the Helm chart upgrade:

```
helm upgrade <release_name> <management_chart_path> -f <common-
custom-values-path> -f <management-custom-values-path> -n
<management_namespace>
```

5. Post-upgrade error logs should not recur.

OCNADD02001: Loss of Heartbeat

TYPE: COMMUNICATION

SEVERITY: MINOR

Scenario:

When heartbeat from mate site Redundancy Agent is not sent to the primary site Redundancy Agent.

Problem Description:

When there is no heartbeat sent from mate site Redundancy Agent to primary site Redundancy Agent, then Loss of Heartbeat is detected.

Troubleshooting Steps:

1. Run curl request from primary site to secondary site Redundancy Agent to verify if calls are received at the Secondary Site:
 - a. Run Exec command in the mate site Redundancy Agent pod

```
kubectl exec -ti -n <namespace> <mate_redundancyagent_pod_name> -- bash
```
 - b. Run the following cURL command:

```
curl -k --cert-type P12 --cert /var/securityfiles/keystore/serverKeyStore.p12:$OCNADD_SSL_KEY_STORE_PASSWORD --location -X GET https://<mate_redundancy_agent_external_ip>:13000
```
2. This should return 4xx error; if 500 response is received, then the connection between two sites is lost.
3. If communication fails, then verify network communication between both setups.
4. Run sync after the issue has been fixed.

OCNADD050018: Consumer Feed Configuration Sync Discrepancy**TYPE: OPERATIONAL_ALARM****SEVERITY: MAJOR****Scenario:**

When the mate site configuration is created, both the sites have Consumer Feed Configuration with the same name, but Data Transfer Object (DTO) parameters are different.

Problem Description:

The consumer feed configuration is mismatched between mated worker group pair.

Troubleshooting Steps:

1. Check mate site consumer feed configuration.
2. Update the configuration with the mated site.
3. Run sync option to clear the alarm.

OCNADD050019: Kafka Feed Configuration Sync Discrepancy**TYPE: OPERATIONAL_ALARM****SEVERITY: MAJOR****Scenario:**

When the mate site configuration is created, both the sites have Kafka feed configuration with the same name, but DTO parameters are different.

Problem Description:

The Kafka feed configuration is mismatched between mated worker group pair.

Troubleshooting Steps:

1. Check mate site Kafka feed configuration.
2. Update the Kafka feed configuration with the mated site.
3. Run sync option to clear the alarm.

OCNADD050020: Filter Configuration Sync Discrepancy**TYPE: OPERATIONAL_ALARM****SEVERITY: MAJOR****Scenario:**

When the mate site configuration is created, both the sites have filter configuration with the same name, but DTO parameters are different.

Problem Description:

The filter configuration is mismatched between mated worker group pair.

Troubleshooting Steps:

1. Check mate site filter configuration.
2. Update the filter configuration with the mated site.
3. Run sync option to clear the alarm.

OCNADD050021: Correlation Configuration Sync Discrepancy**TYPE: OPERATIONAL_ALARM****SEVERITY: MAJOR****Scenario:**

When mate site configuration is created, both the sites have Correlation feed configuration with the same name, but DTO parameters are different.

Problem Description:

The correlation configuration is mismatched between mated worker group pair.

Troubleshooting Steps:

1. Check mate site correlation configuration.
2. Update the correlation configuration with the mated site.
3. Run sync option to clear the alarm.

Unable to Sync Consumer Feed/Filter/Correlation/Kafka Feed**Scenario:**

Unable to Sync Consumer Feed/Filter/Correlation/Kafka Feed

Problem Description:

Consumer Feed/Filter/Correlation/Kafka Feed configuration is not getting synced and is missing in the local site even after triggering manual sync. This issue persists despite resolving discrepancies encountered during the initial configuration.

Solution:

1. Delete the two-site redundancy configuration and create again.
2. Verify on secondary site if all the Consumer Feed/Filter/Correlation/Kafka Feed has been synced or not.

One of the mated sites is unavailable and user updates Consumer Feed/Filter/Correlation in the available Site**Scenario:**

One of the Mated Site is unavailable, and the user updates Consumer Feed/Filter/Correlation in the available Site.

Problem Description:

After creating mate site configuration, the user updates the Consumer Feed/Filter/Correlation when one of the mated sites is unavailable. On updating the Consumer Feed/Filter/Correlation, the change will be reflected only on the available site, and when the other mated site becomes available again, discrepancy alarms will be raised.

Solution:

1. Repeat the Consumer Feed/Filter/Correlation update operation in the primary site.
2. Clear the discrepancy alarm that was raised previously.

One of the mated sites is unavailable and user deletes Consumer Feed/Filter/Correlation in the available Site**Scenario:**

One of the mated site is unavailable, and the user deletes Consumer Feed/Filter/Correlation in the available Site.

Problem Description:

After creating mate site configuration, the user deletes the Consumer Feed/Filter/Correlation when one of the mated sites is unavailable. On deleting the Consumer Feed/Filter/Correlation, the change will be reflected only on the available site, and when the other mated site becomes available again, discrepancy alarms will be raised.

Solution:

1. Delete the mate site configuration.
2. Delete the Consumer Feed/Filter/Correlation in both sites if present.
3. Create the mate site configuration again.

2.2.23 Resource Allocation Challenges During DD Installation on OCI

Scenario:

When DD is installed in OCI, some of the services are stuck in a pending state for a long time.

Problem:

Describing pods of pending services using the below command shows insufficient resources available to start the services:

```
kubectl describe po <pod_name> -n <namespace>
```

Solution:

When encountering OKE cluster nodes CPU and memory issues during DD installation, it's essential to address them promptly. Follow these steps:

1. Begin by assessing the utilization of resources on the affected nodes. You can do this by executing the command:

```
kubectl get nodes  
kubectl describe node <node_name>
```

2. If the resource utilization exceeds 90%, it indicates potential congestion. To alleviate this, proceed with increasing the node's resources using the following steps:
 - a. Manually adjust the CPU and memory allocation for the instance node. Access the Oracle Cloud Infrastructure Console, navigate to "**Instances**," and select "**Instance Details**."
 - b. Under "**More Actions**," choose "**Edit**" and then "**Edit Shape**."
 - c. Expand the shape settings and adjust the CPU and memory allocation according to the specific requirements.
 - d. After making the changes, reboot the instance and wait until it transitions to the "**Running**" state.

Note

- One OCPU (Oracle Compute Unit) is equivalent to two vCPUs.
- Ensure that the allocated resources remain within the maximum limit of the instance shape.
- Increasing CPU and memory resources is subject to the limitations of the tenancy's assigned resource allocation.

2.2.24 Transaction Filter Update Takes Few Seconds to Reflect Changes in Traffic Processing

When a user updates a configuration for transaction filters, the DD services (filter service and consumer adapter) promptly receive a notification for the filter condition update; however, it may take a few seconds for the changes to be applied to the actual traffic processing.

2.2.25 Connection Timeout Errors in Configuration Service Logs

Scenario:

The "Connection Timeout Errors" are occasionally logged by the Configuration service when an instance of the service is deleted or goes down during pod restart and sends an unsubscribe request. However, the unsubscribe request to the configuration service may not always be successful, potentially due to network glitches. If the IP reported in these error logs does not belong to any pod, then these entries are treated as stale entries.

Error Message:

Following error logs will be reported in the configuration service logs:

```
reactor.core.Exceptions$ErrorCallbackNotImplemented:  
org.springframework.web.reactive.function.client.WebClientRequestException:  
connection timed out after 30000 ms: /10.233.92.24:9664
```

```
Caused by:
org.springframework.web.reactive.function.client.WebClientRequestException:
connection timed out after 30000 ms: /10.233.92.24:9664
    at
org.springframework.web.reactive.function.client.ExchangeFunctions$DefaultExchangeFunction.lambda$wrapException$9(ExchangeFunctions.java:136) ~[spring-webflux-6.1.2.jar!/:6.1.2]
...
...
Caused by: io.netty.channel.ConnectTimeoutException: connection timed out
after 30000 ms: /10.233.92.24:9664
...
```

Solution:

These error messages will not have any impact on the service processing the request. It can be ignored.

2.2.26 High Latency in Adapter Feed Due to High Disk Latency

Scenario:

The data director adapter feed reports high latency.

Problem:

- The Fluentd pod reports multiple restarts and frequently gets killed by OOMKilled.
- This results in high disk usage by the Fluentd pods, impacting the Kafka disk write access. The Ceph storage class is shared by all the pods deployed in the cluster.

occne-fluentd-opensearch-2hv5p	1/1	Running	2 (22d ago)	30d
occne-fluentd-opensearch-2kq5v	1/1	Running	2 (37h ago)	30d
occne-fluentd-opensearch-2lcrx	1/1	Running	0	30d
occne-fluentd-opensearch-2zvvg	1/1	Running	0	30d
occne-fluentd-opensearch-5wmnx	1/1	Running	2	30d
occne-fluentd-opensearch-65c85	1/1	Running	9 (3d16h ago)	30d
occne-fluentd-opensearch-6xjqg	1/1	Running	0	30d
occne-fluentd-opensearch-742wl	1/1	Running	31 (3h55m ago)	30d
occne-fluentd-opensearch-774s4	1/1	Running	6 (10d ago)	30d
occne-fluentd-opensearch-7ftvq	1/1	Running	6 (10d ago)	30d
occne-fluentd-opensearch-7thmq	1/1	Running	1 (5d13h ago)	30d
occne-fluentd-opensearch-85dfm	1/1	Running	11 (44h ago)	30d
occne-fluentd-opensearch-879nv	1/1	Running	3 (7d17h ago)	30d
occne-fluentd-opensearch-8d2mh	1/1	Running	1 (3d5h ago)	30d
occne-fluentd-opensearch-8j89p	1/1	Running	4 (3d ago)	30d
occne-fluentd-opensearch-9fhm9	1/1	Running	1 (14d ago)	30d
occne-fluentd-opensearch-9ll85	1/1	Running	0	30d
occne-fluentd-opensearch-9sq65	1/1	Running	5 (2d8h ago)	30d
occne-fluentd-opensearch-bxdrt	1/1	Running	2 (20d ago)	30d
occne-fluentd-opensearch-chl9h	1/1	Running	0	30d
occne-fluentd-opensearch-cwh7n	1/1	Running	3322 (25m ago)	30d
occne-fluentd-opensearch-cxnz4	1/1	Running	0	30d
occne-fluentd-opensearch-djt7f	1/1	Running	4369 (4m38s ago)	30d
occne-fluentd-opensearch-fjxk9	1/1	Running	2 (10d ago)	30d
occne-fluentd-opensearch-gw8wd	1/1	Running	5 (2d8h ago)	30d
occne-fluentd-opensearch-gxdkd	1/1	Running	1 (10d ago)	30d
occne-fluentd-opensearch-h6mps	1/1	Running	0	30d
occne-fluentd-opensearch-jqtxd	1/1	Running	54 (12h ago)	30d
occne-fluentd-opensearch-l8lfs	1/1	Running	0	30d
occne-fluentd-opensearch-p4fn2	1/1	Running	1 (27d ago)	30d
occne-fluentd-opensearch-phjwn	1/1	Running	1 (30d ago)	30d
occne-fluentd-opensearch-pzct5	1/1	Running	0	30d
occne-fluentd-opensearch-qfbrd	1/1	Running	4 (17d ago)	30d
occne-fluentd-opensearch-qx29v	1/1	Running	0	30d
occne-fluentd-opensearch-qx4zh	1/1	Running	28 (17h ago)	30d
occne-fluentd-opensearch-tn772	1/1	Running	1 (25d ago)	30d
occne-fluentd-opensearch-v7ql4	1/1	Running	2 (14d ago)	30d
occne-fluentd-opensearch-wt5pf	1/1	Running	1 (9d ago)	30d
occne-fluentd-opensearch-wx7qd	1/1	Running	5 (6d18h ago)	30d
occne-fluentd-opensearch-x8c29	1/1	Running	0	30d
occne-fluentd-opensearch-xt9q	1/1	Running	2 (7d12h ago)	30d
occne-fluentd-opensearch-zljpf	1/1	Running	23 (22m ago)	30d
occne-fluentd-opensearch-zm456	1/1	Running	1 (23d ago)	30d
occne-fluentd-opensearch-zps5n	1/1	Running	1 (21d ago)	30d

Details:

Containers:

fluentd:

```

Container ID:   containerd://
a586d8342a087b5f910134c0ed796e1acc6c5ababe1c6fcefc3516ac2fea7bb9
Image:         occne-repo-host:5000/docker.io/fluent/fluentd-kubernetes-
daemonset:v1.16.2-debian-opensearch-1.0
Image ID:      occne-repo-host:5000/docker.io/fluent/fluentd-kubernetes-
daemonset@sha256:021133690696649970204ed525f3395c17601b63dcf1fe9c269378dcd0d80
ae1
Port:          24231/TCP
Host Port:     0/TCP
State:         Running
Started:       Thu, 23 May 2024 10:42:00 +0000
Last State:    Terminated
Reason:        OOMKilled

```

Solution:

- Increase the memory allocation for the Fluentd pods to resolve the OOMKilled issue.
- Allow all the Fluentd pods to restart after the memory increase.

2.2.27 Kafka Broker Goes into "CrashLoopBackOff" State During Very Low-Throughput Traffic

Problem

The Kafka broker enters a CrashLoopBackoff state during very low-throughput traffic due to the *No space left on device* error. This occurs when the broker's Persistent Volume Claim (PVC) is full, preventing the broker from starting its process.

Error Code or Error Message

```
[2024-05-20 07:04:11,964] ERROR Error while loading log dir /kafka/logdir/
kafka-logs (kafka.log.LogManager)
java.io.IOException: No space left on device
    at java.base/sun.nio.ch.FileDispatcherImpl.write0(Native Method)
    at java.base/
sun.nio.ch.FileDispatcherImpl.write(FileDispatcherImpl.java:62)
    at java.base/sun.nio.ch.IOUtil.writeFromNativeBuffer(IOUtil.java:132)
    at java.base/sun.nio.ch.IOUtil.write(IOUtil.java:97)
    at java.base/sun.nio.ch.IOUtil.write(IOUtil.java:67)
    at java.base/sun.nio.ch.FileChannelImpl.write(FileChannelImpl.java:288)
    at
org.apache.kafka.storage.internals.log.ProducerStateManager.writeSnapshot(Prod
ucerStateManager.java:683)
    at
org.apache.kafka.storage.internals.log.ProducerStateManager.takeSnapshot(Produ
cerStateManager.java:437)
    at kafka.log.LogLoader.recoverSegment(LogLoader.scala:373)
    at kafka.log.LogLoader.recoverLog(LogLoader.scala:425)
    at kafka.log.LogLoader.load(LogLoader.scala:163)
    at kafka.log.UnifiedLog$.apply(UnifiedLog.scala:1804)
    at kafka.log.LogManager.loadLog(LogManager.scala:278)
    at kafka.log.LogManager.$anonfun$loadLogs$15(LogManager.scala:421)
    at java.base/
java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:539)
    at java.base/java.util.concurrent.FutureTask.run(FutureTask.java:264)
    at java.base/
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1136
)
    at java.base/
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:635
)
    at java.base/java.lang.Thread.run(Thread.java:842)
```

Solution

This issue can be resolved by following the steps below. The procedure is applicable to the relayagent or mediation group with CEPH-based Kafka storage; use the corresponding group's namespace.

1. Increase the PVC size of your Kafka broker so that the Kafka process can start normally. Refer to the section *Steps to increase the size of existing PVC* of the *Oracle Communications Network Analytics Data Director User Guide* for more details on how to increase the size of a PVC.

For example, if you have configured a small PVC size for your Kafka broker, such as 8Gi, you can increase the size to 11Gi for the Kafka broker process to start normally.

2. Run the following command to exec into the Kafka broker pod:

```
kubectl exec -ti kafka-broker-0 -n <namespace> -- bash
```

3. Navigate to the kafka/bin folder and apply the config segment.ms for your Kafka topics by running the appropriate command:

If ACL is not enabled, run:

```
./kafka-configs.sh --bootstrap-server localhost:9092 --alter --entity-type topics --entity-name <TOPIC_NAME> --add-config segment.ms=<Time in ms>
```

If ACL is enabled, run:

```
./kafka-configs.sh --bootstrap-server kafka-broker:9094 --alter --entity-type topics --entity-name <TOPIC_NAME> --add-config segment.ms=<Time in ms> --command-config ../../admin.properties
```

The config must be applied for all the OCNADD Kafka topics.

Time in ms can be any small value, for example 600000 (10 minutes) or 720000 (12 minutes).

2.2.28 Database Space Full Preventing Configuration and Subscription from Being Saved in Database

Scenario:

The Data Director has separate schemas for configuration, alarms, health monitoring, and xDRs. However, all these schemas share a common

cnDBTier

instance. The database space can become full in the following scenarios:

1. **Alarm Flooding:** Too many alarms are raised due to repetitive failure scenarios, such as network communication failure.
2. **Excessive xDR Generation:** Too many xDRs are generated either due to a traffic spike or network misconfiguration, leading to an increased number of xDRs.

Problem:

When this issue occurs, new entries in database tables within various schemas cannot be created. The following error might be seen in the logs of the Data Director export service:

```
SQL Error: 1114, SQLState: HY000 The table 'SUBSCRIPTION' is full
```

Solution:

- Free up space in the AlarmDB by deleting older alarm entries.
- Free up space in the xDRDB by deleting older xDRs.

Steps to Resolve:

Step 1: Run the following command to log in to the MySQL pod.

Note

Use the namespace in which the cnDBTier is deployed. For example, the occne-cndbtier namespace is used. The default container name is ndbmysqld-0.

```
kubectl -n occne-cndbtier exec -it ndbmysqld-0 -- bash
```

Step 2: Run the following command to log in to the MySQL server using the MySQL client

```
$ mysql -h 127.0.0.1 -uocdd -p
$ Enter password:
```

Step 3: Free up space in the Alarm DB by deleting older alarms and event entries.

1. Fetch the required events from the EVENTS table:

```
SELECT * FROM <alarm_db_name>.EVENTS
WHERE ALARM_ID IN (
    SELECT ALARM_ID FROM <alarm_db_name>.ALARM
    WHERE (ALARM_SEVERITY = 'INFO')
    OR (ALARM_STATUS = 'CLEARED' AND RAISED_TIME <= NOW() - INTERVAL 7 DAY)
)
LIMIT 5000;
```

2. Delete the required events from the EVENTS table:

```
DELETE FROM <alarm_db_name>.EVENTS
WHERE ALARM_ID IN (
    SELECT ALARM_ID FROM <alarm_db_name>.ALARM
    WHERE (ALARM_SEVERITY = 'INFO')
    OR (ALARM_STATUS = 'CLEARED' AND RAISED_TIME <= NOW() - INTERVAL 7 DAY)
)
LIMIT 5000;
```

3. Fetch the required alarms from the ALARM table:

```
SELECT * FROM <alarm_db_name>.ALARM
WHERE (ALARM_SEVERITY = 'INFO')
OR (ALARM_STATUS = 'CLEARED' AND RAISED_TIME <= NOW() - INTERVAL 7 DAY)
LIMIT 5000;
```

4. Delete the required alarms from the ALARM table:

```
DELETE FROM <alarm_db_name>.ALARM
WHERE (ALARM_SEVERITY = 'INFO')
OR (ALARM_STATUS = 'CLEARED' AND RAISED_TIME <= NOW() - INTERVAL 7 DAY)
LIMIT 5000;
```

5. Commit the changes:

```
commit;
```

Step 4: Free up space in the xDR DB by deleting older xDRs. The xDR table name is the same as the Correlation Configuration name (for example, <correlationname>XDR). Replace <xDR_table_name> with <correlationname>XDR in the steps below:

1. Fetch the required xDRs from the corresponding xDR table:

```
SELECT * FROM <storageadapter_schema>.<correlationname>XDR
WHERE beginTime <= NOW() - INTERVAL 6 HOUR
LIMIT 5000;
```

2. Delete the required xDRs from the corresponding xDR table:

```
DELETE FROM <storageadapter_schema>.<correlationname>XDR
WHERE beginTime <= NOW() - INTERVAL 6 HOUR
LIMIT 5000;
```

3. Commit the changes:

```
commit;
```

Note

The limit in the above queries can be increased from '5000' as needed.

2.2.29 Invalid Subscription Entry in the Subscription Table

Scenario:

OCNADD supports a subscription mechanism for the configuration information exchange within Data Director services. OCNADD configuration service maintains the subscription table, and all the other services like correlation, filter, export, consumer adapter, etc., subscribe to the configuration service on startup and receive configuration updates via notifications from the configuration service. The configuration service manages the subscription entry cleanup from the database and its local cache. It deletes the subscription entry of the service which is not available or which has been unregistered or unsubscribed with the subscription service.

In the older version of OCNADD, for example, 24.2.x/24.3.x, it is possible that the older subscription entry may remain in the subscription table as the cleanup mechanism was not available or the clean unsubscription could not happen. This results in the older entries remaining in the subscription table. These older entries may result in delayed configuration notifications sent to the services that have subscribed to the configuration service when the configuration service is restarted during an upgrade/rollback. The other services, like correlation, export, or adapter, may not receive the notification in a timely manner and could not perform their functionality for a prolonged time duration.

Problem:

When this issue occurs, older entries in the subscription table can be observed. This is typically expected in rollback scenarios to older releases (e.g., from release 25.1.0 to 24.2.x).

For example, the following is a stale subscription entry for the export service in the subscription table:

```
| 6c0b06a8-06e1-47f3-91b3-6e3828284cca | https://10.233.82.205:12595/ocnadd-  
export/v1/export/notification | EXPORTSERVICE-1 | EXPORT_SERVICE | EXPORT-1 |  
kp-mgmt:ocnadd-vcne3 |
```

```
mysql> select * from SUBSCRIPTION;  
+-----+  
+-----+  
+-----+-----+-----+  
+-----+  
| SUBSCRIPTION_ID |  
NOTIFICATION_URI |  
SERVICE_ID | SERVICE_TYPE | TARGET_CONSUMERNAME |  
WORKER_GROUP |  
+-----+  
+-----+  
+-----+-----+-----+  
+-----+  
| 980705eb-54ac-4148-9962-e587e80f4328 | https://10.233.113.164:9664/ocnadd-  
storageadapter/v1/notification | feed1-storage-adapter-1 |  
STORAGE_ADAPTER_SERVICE | feed1 | kp-mgmt:ocnadd-vcne3 |  
| 012852ec-97f2-483c-a625-02d3c7f2c32d | https://10.233.96.131:12585/ocnadd-  
egress-filter/v1/notifications | FILTER_SERVICE-1 |  
FILTER_SERVICE | egress-filter | kp-mgmt:ocnadd-vcne3 |  
| 467ef3b8-f26f-443e-a1e0-11b0a308e33a | https://10.233.75.161:9664/ocnadd-  
correlation/v1/notification | feed1-correlation-1 |  
CORRELATION_SERVICE | feed1 | kp-mgmt:ocnadd-vcne3 |  
| d75fc1f7-82fe-442c-a528-83b7b35d6078 | https://10.233.109.96:9664/ocnadd-  
correlation/v1/notification | feed1-correlation-1 |  
CORRELATION_SERVICE | feed1 | kp-mgmt:ocnadd-vcne3 |  
| 3c353e06-0635-4c4a-8c32-d13ac18069f9 | https://10.233.97.134:9182/ocnadd-  
consumeradapter/v5/notifications | A-ORA-ADAPTER-1 |  
CONSUMER_ADAPTER_SERVICE | a-ora | kp-mgmt:ocnadd-vcne3 |  
| c1f0541f-ae36-46a2-b41f-09e3fb247505 | https://10.233.89.232:9182/ocnadd-  
consumeradapter/v5/notifications | A-ORA-ADAPTER-2 |  
CONSUMER_ADAPTER_SERVICE | a-ora | kp-mgmt:ocnadd-vcne3 |  
| 6c0b06a8-06e1-47f3-91b3-6e3828284cca | https://10.233.82.205:12595/ocnadd-  
export/v1/export/notification | EXPORTSERVICE-1 |  
EXPORT_SERVICE | EXPORT-1 | kp-mgmt:ocnadd-vcne3 |  
| a38e52ef-fd69-4ffe-b4c0-b70ab05c9301 | https://10.233.109.77:9664/ocnadd-  
correlation/v1/notification | feed1-correlation-1 |  
CORRELATION_SERVICE | feed1 | kp-mgmt:ocnadd-vcne3 |  
+-----+  
+-----+  
+-----+-----+-----+  
+-----+  
8 rows in set (0.01 sec)
```

The following logs could be seen in the configuration service:

```
OCL [36m2024-12-04 07:31:03.237[0;39m [1;30m[parallel-1][0;39m [31mWARN  
[0;39m [35mc.o.c.c.o.c.e.s.n.ExportNotificationService[0;39m - Sending Export
```

Config notification to uri: https://10.233.82.205:12595/ocnadd-export/v1/export/notification is not successful. Retry number 1

```
OCL [36m2024-12-04 07:31:33.234[0;39m [1;30m[parallel-1][0;39m [31mWARN
[0;39m [35mc.o.c.c.o.c.e.s.n.ExportNotificationService[0;39m - Sending Export
Config notification to uri: https://10.233.82.205:12595/ocnadd-export/v1/
export/notification is not successful. Retry number 2
```

```
OCL [36m2024-12-04 07:32:03.240[0;39m [1;30m[parallel-1][0;39m [31mWARN
[0;39m [35mc.o.c.c.o.c.e.s.n.ExportNotificationService[0;39m - Sending Export
Config notification to uri: https://10.233.82.205:12595/ocnadd-export/v1/
export/notification is not successful. Retry number 3
```

The configuration service is not able to create the connection with the IP of the POD corresponding to the older subscription entry as that POD no more exists, the following connection timeout can also be seen in the configuration service log:

```
OCL [36m2024-12-04 07:33:02.252[0;39m [1;30m[default-nioEventLoopGroup-1-2]
[0;39m [1;31mERROR[0;39m [35mclient-logger[0;39m - Failed to connect to
remote io.netty.channel.ConnectTimeoutException: connection timed out after
30000 ms: /10.233.82.205:12595 at
io.netty.channel.nio.AbstractNioChannel$AbstractNioUnsafe$1.run(AbstractNioCha
nnel.java:263) at
io.netty.util.concurrent.PromiseTask.runTask(PromiseTask.java:98) at
io.netty.util.concurrent.ScheduledFutureTask.run(ScheduledFutureTask.java:153)
at
io.netty.util.concurrent.AbstractEventExecutor.runTask(AbstractEventExecutor.j
ava:173) at
io.netty.util.concurrent.AbstractEventExecutor.safeExecute(AbstractEventExecut
or.java:166) at
io.netty.util.concurrent.SingleThreadEventExecutor.runAllTasks(SingleThreadEve
ntExecutor.java:469) at
io.netty.channel.nio.NioEventLoop.run(NioEventLoop.java:569) at
io.netty.util.concurrent.SingleThreadEventExecutor$4.run(SingleThreadEventExec
utor.java:994) at
io.netty.util.internal.ThreadExecutorMap$2.run(ThreadExecutorMap.java:74) at
io.netty.util.concurrent.FastThreadLocalRunnable.run(FastThreadLocalRunnable.j
ava:30) at java.base/java.lang.Thread.run(Thread.java:1583)
```

Solution:

1. Delete the stale entry from the subscription table. For example, remove the entry corresponding to the subscription ID 6c0b06a8-06e1-47f3-91b3-6e3828284cca with the POD IP 10.233.82.205:12595 in the presented example.
2. Restart the configuration service after deleting the entry from the subscription table in the configuration database.

2.2.30 Correlation Configuration Does Not Get Deployed Post Rollback

Scenario:

Correlation configuration does not get deployed after the rollback.

Problem:

1. The UI is not accessible after the Configuration Service is restarted following a rollback from the current release to target release.
2. The Correlation Configuration does not get deployed after the Configuration Service is restarted following a rollback from the current release to target release.

Error Code or Error Message:

Figure 2-2 Error Code or Error Message

Below logs are observed in config service when UI is accessed :

```

OCL 2025-01-31 10:04:22.307 [virtual-executor89] ERROR i.m.h.s.RouteExecutor - Unexpected error occurred: failed to lazily initialize a collection of
role: com.oracle.cgbu.cne.ocnadd.configuration.correlation.entity.CorrelationConfiguration.supportedXdrContents: could not initialize proxy - no
Session
org.hibernate.LazyInitializationException: failed to lazily initialize a collection of role:
com.oracle.cgbu.cne.ocnadd.configuration.correlation.entity.CorrelationConfiguration.supportedXdrContents: could not initialize proxy - no Session
    at org.hibernate.collection.spi.AbstractPersistentCollection.throwLazyInitializationException(AbstractPersistentCollection.java:634)
    at org.hibernate.collection.spi.AbstractPersistentCollection.withTemporarySessionIfNeeded(AbstractPersistentCollection.java:217)
    at org.hibernate.collection.spi.AbstractPersistentCollection.initialize(AbstractPersistentCollection.java:613)
    at org.hibernate.collection.spi.AbstractPersistentCollection.read(AbstractPersistentCollection.java:136)
    at org.hibernate.collection.spi.PersistentBag.toString(PersistentBag.java:587)
    at java.base/java.lang.StringConcatHelper.stringOf(StringConcatHelper.java:467)
    at

```

Solution:

This issue can be addressed by applying the following workaround:

Workaround 1: Edit Deployment for Configuration Service

1. Edit the Configuration Service deployment and add the following parameter to the environment variables:

```

- name:
  logger.levels.com.oracle.cgbu.cne.ocnadd.configuration.correlation.service.
  configuration
  value: DEBUG

```

Workaround 2: Update the custom values of the management group and add the parameter in the ocnaddconfiguration service section, then perform a Helm upgrade

1. Update the parameter
logger.levels.com.oracle.cgbu.cne.ocnadd.configuration.correlation.service.configuration: DEBUG in the custom values of the management group.

```

ocnaddconfiguration:
  ocnaddconfiguration:
    name: ocnaddconfiguration
    env:

  logger.levels.com.oracle.cgbu.cne.ocnadd.configuration.correlation.service.
  configuration: DEBUG

```

2. Perform the Helm upgrade for the management group:

```

helm upgrade <management-release-name> -f ocnadd-common-custom-values.yaml
-f ocnadd-custom-values-<mgmt-group>.yaml --namespace <source-release-
namespace> <helm_chart>

```

For example:

```
helm upgrade mgmt -f custom-templates/ocnadd-common-custom-values.yaml -f  
custom-templates/ocnadd-custom-values-mgmt.yaml -n mgmt-ns ocnadd
```

2.2.31 Cleanup of Redundant xDR Tables

Scenario

xDR tables may remain in the Storage Adapter schema even after xDR storage has been disabled or removed. The following scenarios can lead to this condition:

- xDR storage is disabled in the correlation configuration.
- The correlation configuration has been deleted.
- The Storage Adapter was unavailable in a previous release, and a rollback was performed from the current release with xDR storage enabled.

Solution

If xDRs are no longer required, it is recommended to delete the xDR tables directly using the MySQL client.

2.2.32 Filter service does not receive any notification update after Migration

Scenario

After the migration has been completed in the current release, the filter service does not receive notification from the configuration service. The data processing from the filter service will be stopped, and any FILTERED / CORRELATED_FILTERED feed will stop working.

Solution

The solution is to restart the ocnaddfilter service in the mediation namespace. The same can be achieved by deleting the ocnaddfilter pod in the mediation namespace.

```
kubectl delete po ocnaddfilter-xxxx -n <mediation_namespace>
```

Example if mediation namespace is dd-med:

```
kubectl delete po ocnaddfilter-xxxx -n dd-med
```

2.3 Troubleshooting Traffic Stability in OCNADD

Verifying Pod Stability

To verify pod stability, run the following command:

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

Example

To verify pods in the management group namespace ocnadd-mgmt:

```
kubectl get po -n ocnadd-mgmt
```

To verify pods in the relay agent group namespace ocnadd-rea:

```
kubectl get po -n ocnadd-rea
```

To verify pods in the mediation group namespace ocnadd-med:

```
kubectl get po -n ocnadd-med
```

Output:

Any pod restarts or CrashLoopBackOff status will appear in the output.

Figure 2-3 Output

NAME	READY	STATUS	RESTARTS	AGE
ocnaddalarm-6497657997-8md8d	1/1	Running	0	8h
ocnaddcnlbannotationgen-w4lv4	0/1	Completed	0	3h12m
ocnaddconfiguration-68b9987db9-xxq5b	1/1	Running	0	8h
ocnaddexport-99594574d-xrggb	1/1	Running	0	8h
ocnaddgui-7d5f899fd6-lt2jq	1/1	Running	0	8h
ocnaddhealthmonitoring-787cbbf648-x7m5n	1/1	Running	0	8h
ocnaddmanagementgateway-84d5dd4fbb-p7mj7	1/1	Running	0	8h
ocnaddredundancyagent-6c75c8d64-62xv5	1/1	Running	0	8h
ocnadduirouter-ff4f7db4c-lrh1x	1/1	Running	0	8h

Throughput metrics

Analyzing Throughput Metrics

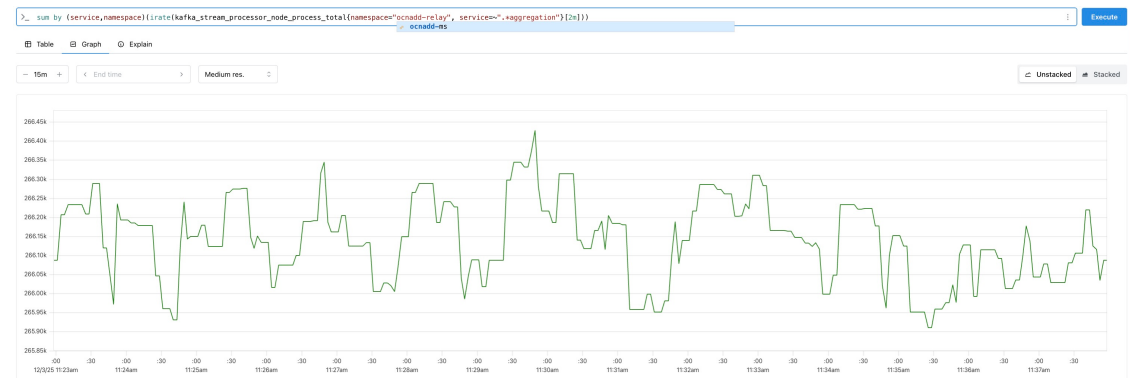
Analyze the throughput metrics to validate both ingress and egress traffic rates. The following metrics are used to measure them:

- **Ingress:** `kafka_stream_processor_node_process_total`
(Metric reported under the relay agent group namespace)
- **Egress:** `ocnadd_egress_requests_total`
(Metric reported under the mediation group namespace)

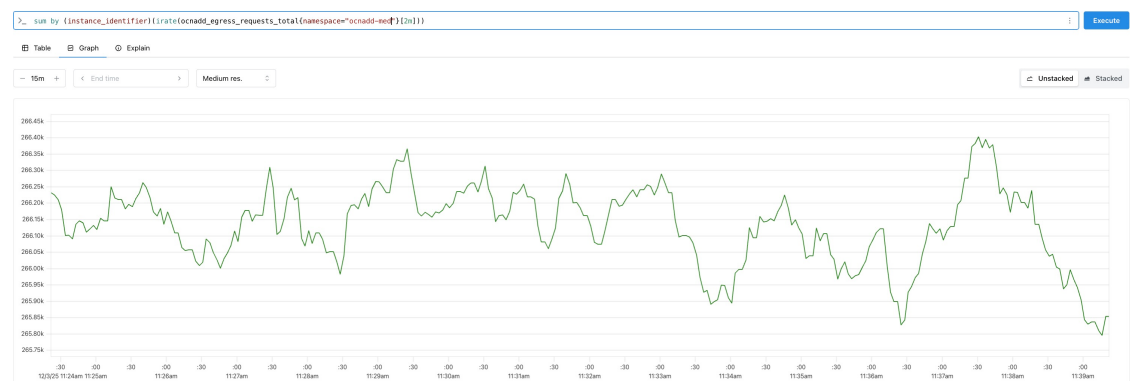
These metrics can be visualized in Prometheus to understand the current system throughput. For details on these metrics and corresponding Prometheus queries, refer to the “OCNADD Metrics, KPIs, and Alerts” section of the **Oracle Communications Network Analytics Data Director User Guide**.

Output:

Ingress traffic rate

Figure 2-4 Ingress traffic rate

Egress traffic rate

Figure 2-5 Egress traffic rate

Resource utilization

To check the pod CPU and memory utilisation, run the following command:

```
kubectl top po -n <ocnadd-group-namespace>
```

Example:

To verify resource utilisation for pods in the management group namespace ocnadd-mgmt:

```
> kubectl top po -n ocnadd-mgmt
```

To verify resource utilisation for pods in the relay agent group namespace ocnadd-relay:

```
> kubectl top po -n ocnadd-relay
```

To verify resource utilisation for pods in the mediation group namespace ocnadd-med:

```
> kubectl top po -n ocnadd-med
```

Output:

The output lists the resource utilisation for each pod in the specified namespace. Ensure no pod sustains CPU or memory utilisation above 85 percent for an extended duration.

Figure 2-6 Output

ocnaddalarm-6497657997-8md8d	3m	308Mi
ocnaddconfiguration-68b9987db9-xxq5b	7m	362Mi
ocnaddexport-99594574d-xrggb	9m	345Mi
ocnaddgui-7d5f899fd6-lt2jq	1m	31Mi
ocnaddhealthmonitoring-787cbbf648-x7m5n	5m	283Mi
ocnaddmanagementgateway-84d5dd4fbb-p7mj7	1m	188Mi
ocnaddredundancyagent-6c75c8d64-62xv5	1m	158Mi
ocnadduirouter-ff4f7db4c-lrh1x	1m	144Mi

Kafka consumer performance

Run the Kafka consumer group command in the Relay Agent Kafka cluster and the Mediation Kafka cluster to measure partition lag in the source NF topic and the MAIN topic respectively. There should not be significant lag in the partitions for a long period of time, and it should not continue to accumulate. Review the consumer group command output every 2–3 minutes to determine whether the lag is decreasing or building up across partitions.

```
./kafka-consumer-groups.sh --bootstrap-server <host>:<port> --describe --all-groups
```

Example:

```
./kafka-consumer-groups.sh --bootstrap-server kafka-broker:9092 --describe --all-groups
```

For Kafka port 9093 and 9094, the user must create admin.properties to run the command:

```
./kafka-consumer-groups.sh --bootstrap-server kafka-broker:9093 --describe --all-groups --command-config /path/to/admin.properties
```

Output:**Figure 2-7 output**

```
ocnadd@kafka-broker-0 bin]$ ./kafka-consumer-groups.sh --bootstrap-server kafka-broker:9092 --describe --all-groups
```

GROUP	TOPIC	PARTITION	CURRENT-OFFSET	LOG-END-OFFSET	LAG	CONSUMER-ID	HOST	CL
ocnaddscppaggregation-StreamThread-4-consumer	SCP	41	734040666	734051086	10340	ocnaddscppaggregation-StreamThread-4-consumer-3ac1c273-b9ca-4e8e-9fba-b6ee3e1bc36	/10.233.96.185	oc
ocnaddscppaggregation-StreamThread-1-consumer	SCP	4	733853257	733863952	10695	ocnaddscppaggregation-StreamThread-1-consumer-d9570884-b91a-4243-81f4-073f715478f3	/10.233.110.132	oc
ocnaddscppaggregation-StreamThread-5-consumer	SCP	49	733954486	733956586	2100	ocnaddscppaggregation-StreamThread-5-consumer-d472047-5c38-4958-8c85-4f76325cc07c	/10.233.83.31	oc
ocnaddscppaggregation-StreamThread-1-consumer	SCP	5	733984648	733991675	7027	ocnaddscppaggregation-StreamThread-1-consumer-ac6f1598-6fb4-4d45-951b-925934539bd3	/10.233.83.31	oc
ocnaddscppaggregation-StreamThread-6-consumer	SCP	59	733553436	733561161	7725	ocnaddscppaggregation-StreamThread-6-consumer-d2b2ea97-4b31-479a-9c9a-acf4b78c7a2e	/10.233.110.132	oc
ocnaddscppaggregation-StreamThread-3-consumer	SCP	29	734075162	734086172	11010	ocnaddscppaggregation-StreamThread-3-consumer-2e60b57f-cbcb-4da0-b034-0bf2afec8ca3	/10.233.69.164	oc
ocnaddscppaggregation-StreamThread-1-consumer	SCP	10	734179252	734179696	444	ocnaddscppaggregation-StreamThread-1-consumer-6bef32aa-9d94-48ad-9024-ea0a43338713	/10.233.82.17	oc
ocnaddscppaggregation-StreamThread-2-consumer	SCP	18	734208216	734212648	4432	ocnaddscppaggregation-StreamThread-2-consumer-97196b85-b94d-46e5-b0d1-fdeca08c1703	/10.233.69.164	oc
ocnaddscppaggregation-StreamThread-4-consumer	SCP	34	734129934	734134480	4546	ocnaddscppaggregation-StreamThread-4-consumer-199f5080-fe26-4681-8180-23ebdc1d2efe	/10.233.119.186	oc

Figure 2-8 output

```
[ocnadd@kafka broker-0 bin]$ ./kafka-consumer-groups.sh --bootstrap-server kafka-broker-9092 --describe --all-groups
```

GROUP	HOST	TOPIC	PARTITION	CURRENT-OFFSET	LOG-END-OFFSET	LAG	CONSUMER-ID
nim-perf-1-adapter-edbbeb	MAIN		135	182725694	182726624	930	nim-perf-1-adapter-edbbeb-StreamThread-5-consumer-cdd59afb-9b99-4422-96c3-0b6e35
3ad8e2	/10.233.96.193	nim-perf-1-adapter-edbbeb-StreamThread-5-consumer	34	182672159	182674014	1855	nim-perf-1-adapter-edbbeb-StreamThread-2-consumer-8bd32b66-3b75-4c85-9ce2-a1158e
936f43	/10.233.84.230	nim-perf-1-adapter-edbbeb-StreamThread-2-consumer	28	182724962	182728016	3854	nim-perf-1-adapter-edbbeb-StreamThread-1-consumer-98d5572c-be9d-4df5-89c5-b7a215
9c6dd3	/10.233.121.26	nim-perf-1-adapter-edbbeb-StreamThread-1-consumer	245	182728146	182728381	155	nim-perf-1-adapter-edbbeb-StreamThread-9-consumer-6e6f1999-84d6-4225-893a-e2ac36
3a5a3c	/10.233.123.172	nim-perf-1-adapter-edbbeb-StreamThread-9-consumer	144	182791694	182792570	876	nim-perf-1-adapter-edbbeb-StreamThread-5-consumer-5622d66e-bc2d-4084-b9fe-481425
ab6d8e	/10.233.121.26	nim-perf-1-adapter-edbbeb-StreamThread-5-consumer	115	182670718	182672779	2061	nim-perf-1-adapter-edbbeb-StreamThread-4-consumer-b3e0a6b7-94fb-4ca0-9d28-5118ac
de592c	/10.233.121.26	nim-perf-1-adapter-edbbeb-StreamThread-4-consumer	40	182710593	182713087	2494	nim-perf-1-adapter-edbbeb-StreamThread-2-consumer-4b7c3096-dd18-4aa5-93e5-78b882
378ae5	/10.233.96.193	nim-perf-1-adapter-edbbeb-StreamThread-2-consumer	40	182675768	182676045	277	nim-perf-1-adapter-edbbeb-StreamThread-2-consumer-0341523f-9d99-4bd9-9fb3-5ec694
948559	/10.233.79.184	nim-perf-1-adapter-edbbeb-StreamThread-2-consumer	156	182667818	182670060	2242	nim-perf-1-adapter-edbbeb-StreamThread-6-consumer-098ed4c7-4161-42f8-9b0d-7a0b99
c9954b	/10.233.79.184	nim-perf-1-adapter-edbbeb-StreamThread-6-consumer	147	182726760	182729872	3112	nim-perf-1-adapter-edbbeb-StreamThread-6-consumer-6b0e9f6a-1764-41ab-8843-4b3885
3e2f14	/10.233.114.109	nim-perf-1-adapter-edbbeb-StreamThread-6-consumer	85	182666628	182667647	1019	nim-perf-1-adapter-edbbeb-StreamThread-3-consumer-023285d1-9148-4ed1-8883-40a430
b92da5	/10.233.98.177	nim-perf-1-adapter-edbbeb-StreamThread-3-consumer					

End-to-end latency

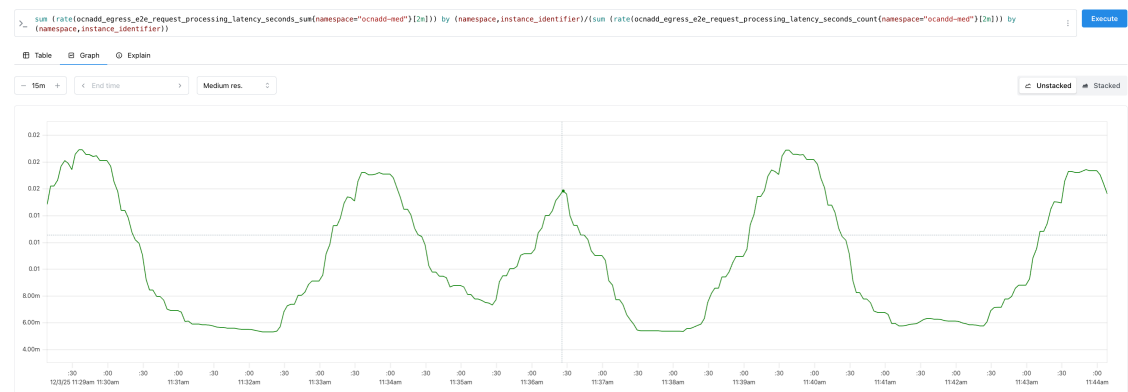
End-to-end metrics can be visualized to verify how quickly traffic is processed. This value should remain below the allowed threshold. Analyze the following metric to check end-to-end latency:

- `ocnadd_egress_e2e_request_processing_latency_seconds`
(Metric reported under the mediation group namespace)

For details on this metric and the Prometheus query, see the *Oracle Communications Network Analytics Data Director User Guide*, in the “OCNADD Metrics, KPIs, and Alert” section.

Output:

Figure 2-9 output



2.4 CNLB Troubleshooting Scenarios

This section should be referred to for troubleshooting issues encountered after OCNADD deployment and configuration in a CNLB-based CNE environment. The issues mentioned below are general problems that can be investigated to identify the root cause and find appropriate solutions.

Secondary Site Unable to Reach Primary Site RA on Load Balancer IP (CNLB IP):

Troubleshooting Steps:

1. Verify indentation and configuration of CNLB annotations first:

```
$ kubectl edit -n <namespace> deployment.apps/ocnaddrundancyagent
```

Indentation and configuration should be like below example:

```
template:
  metadata:
    annotations:
      k8s.v1.cni.cncf.io/networks: default/nf-oam-int1@nf-oam-int1
      oracle.com.cnc/cnlb: '[{"backendPortName": "rrbackendport",
"cnlbIp": "10.123.155.33", "cnlbPort": "13000"}]'
```

2. If the annotation indentation is correct, proceed to check the attached Network Attachment Definition (NAD) information:

```
$ kubectl get net-attach-def <NAD Name Configured in RA> -n default -o yaml
```

For example:

```
spec:
  config: '{"cniVersion": "0.4.0", "name": "nf-oam-int1", "plugins": [{"type": "macvlan",
"mode": "bridge", "master": "bond0.110", "ipam": {"type": "whereabouts", "range":
"172.16.110.0/24", "range_start": "172.16.110.129", "range_end": "172.16.110.190",
"gateway": "172.16.110.1"}}, {"type": "sbr"}]}'
```

3. Retrieve the service set information by running the following command:

```
$ kubectl exec -ti -n <ocne-infra namespace> $(kubectl -n <ocne-infra
namespace> get po --no-headers -l app=cnlb-manager -o custom-
columns=:.metadata.name) -- curl http://localhost:5001/service-info
```

Identify the serviceIpSet that contains the frontEndIP and frontEndPort of the configured CNLB annotation in the RA deployment.

For example: The RA is assigned to service set to 0. Ensure that rrbackendport and 10.121.44.157 are not repeated in more than one entry in any of service set.

```
{
  "backendIpList": ["172.16.113.131"],
  "backendPort": 13000,
  "backendPortName": "rrbackendport",
  "frontEndIp": "10.123.155.33",
  "frontEndPort": 13000,
  "gatewayIp": "172.16.113.1",
  "networkName": "sig3"
}
```

- a. If multiple entries are found, run the following command to check which other services are using the same frontEndIp (cnlbIP). Resolve the issue by following the *Oracle*

Communication Network Analytics Data Director User Guide to re-enable RA CNLB and assign a new IP.

```
$ kubectl get deploy,sts,ds -A -o json | jq -r '.items[] |
select(.spec.template.metadata.annotations."k8s.v1.cni.cncf.io/
networks" != null) | {kind, namespace: .metadata.namespace,
name: .metadata.name,
podAnnotations: .spec.template.metadata.annotations."k8s.v1.cni.cncf.io/
networks", cnlbIPS: .spec.template.metadata.annotations."oracle.com.cnc/
cnlb"}'
```

b. Identify the CNLB App Assigned to Service Set 0

- i. Run the following command to identify which active CNLB app is assigned to service set 0

```
kubectl exec -ti -n occne-infra $(kubectl -n <occne-infra
namespace> get po --no-headers -l app=cnlb-manager -o custom-
columns=:.metadata.name) -- curl http://localhost:5001/node-pod-role
```

Sample output:

```
[root@k8s-bastion-1 ~]# kubectl exec -ti -n occne-infra $(kubectl -n occne-infra get po --no-headers -l app=cnlb-manager -o custom-columns=
:.metadata.name) -- curl http://localhost:5001/node-pod-role
{"serviceIpSet0":[{"nodeName":"k8s-node-2.dv-m1.lab.in.oracle.com","podIp":"10.233.105.155","podRole":"standBy"},{"nodeName":"k8s-node-6.dv
-m1.lab.in.oracle.com","podIp":"10.233.71.213","podRole":"active"}], "serviceIpSet1":[{"nodeName":"k8s-node-1.dv-m1.lab.in.oracle.com","podI
p":"10.233.123.52","podRole":"standBy"}, {"nodeName":"k8s-node-5.dv-m1.lab.in.oracle.com","podIp":"10.233.81.239","podRole":"active"}]}
[root@k8s-bastion-1 ~]#
```

- ii. Determine which CNLB app has the pod IP 10.233.71.213:

```
$ kubectl get po -n occne-infra -o wide | grep 10.233.71.213
```

Sample output:

```
[root@k8s-bastion-1 ~]# kubectl get po -n occne-infra -o wide | grep 10.233.71.213
cnlb-app-58cb4fd4bc-8wgp9      1/1      Running    0      2d      10.233.71.213      k8s-node-6.dv-m1.la
b.in.oracle.com      <none>      <none>
```

- iii. Check CNLB POD Status and Logs and Investigate identified CNLB pod:

```
kubectl logs -n <occne-infra namespace> cnlb-app-58cb4fd4bc-xpbwm
```

Analyze the logs to determine if the issue is related to the CNLB pod.

Ingress/Egress NAD information missing in Dropdown for Ingress Feed and Adapter Feed in GUI

1. First, investigate the CNLB job logs for any errors. If error logs are found, verify that there is a sufficient number of external IPs or that the network is correctly defined in the cnlb.ini file.

```
$ kubectl logs -n <dd-med-namespaces> <cnlb-job-pod-name>
```

For example:

```
$ kubectl logs -n ocnadd-med ocnaddcnlbannotationgen-h5fsb
```

2. Re-run the CNLB job to update the ConfigMap with the latest available NAD information. Refer to the *Oracle Communication Network Analytics Data Director User Guide* for instructions in the chapter "Create CNLB Annotation Generation Job."
3. Verify the content of the ingress/egress ConfigMap to ensure it has been created with the available ingress and egress NAD information.c

```
$ kubectl get cm -n <dd-med-namespace> ocnadd-configmap-cnlb-egress -o yaml
$ kubectl get cm -n <dd-med-namespace> ocnadd-configmap-cnlb-ingress -o
yaml
```

For example:

```
apiVersion: v1
data:
  egressNadInfo: '{"egressNadResponseDto":[{"egressNadName":"nf-oam-egr1","egressNadDstInfo":{"Route
    [dst=10.121.45.0/26, gw=172.16.110.193]}],{"egressNadName":"nf-oam-egr2","egressNadDstInfo":{"Route
    [dst=10.121.45.0/26, gw=172.16.110.194]}],{"egressNadName":"nf-sig1-egr1","egressNadDstInfo":{"Route
    [dst=10.121.45.0/26, gw=172.16.111.193]}, {"egressNadName":"nf-sig1-egr2","egressNadDstInfo":{"Route
    [dst=10.121.45.0/26, gw=172.16.111.194]}, {"egressNadName":"nf-sig2-egr1","egressNadDstInfo":{"Route
    [dst=10.121.45.0/26, gw=172.16.112.193]}, {"egressNadName":"nf-sig2-egr2","egressNadDstInfo":{"Route
    [dst=10.121.45.0/26, gw=172.16.112.194]}, {"egressNadName":"nf-sig3-egr1","egressNadDstInfo":{"Route
    [dst=10.148.204.16/29, gw=172.16.113.193]}, {"egressNadName":"nf-sig3-egr2","egressNadDstInfo":{"Route
    [dst=10.148.204.16/29, gw=172.16.113.194]}, {"egressNadName":"nf-sig4-egr1","egressNadDstInfo":{"Route
    [dst=10.148.204.24/29, gw=172.16.114.193]}, {"egressNadName":"nf-sig4-egr2","egressNadDstInfo":{"Route
    [dst=10.148.204.24/29, gw=172.16.114.194]}, {"egressNadName":"nf-sig5-egr1","egressNadDstInfo":{"Route
    [dst=10.148.204.32/29, gw=172.16.115.193]}, {"egressNadName":"nf-sig5-egr2","egressNadDstInfo":{"Route
    [dst=10.148.204.32/29, gw=172.16.115.194]}, {"egressNadName":"nf-sig6-egr1","egressNadDstInfo":{"Route
    [dst=10.121.45.0/26, gw=172.16.116.193]}, {"egressNadName":"nf-sig6-egr2","egressNadDstInfo":{"Route
    [dst=10.121.45.0/26, gw=172.16.116.194]}]}}'
kind: ConfigMap
metadata:
  creationTimestamp: "2024-07-28T11:31:51Z"
  name: ocnadd-configmap-cnlb-egress

apiVersion: v1
data:
  ingressCnlbNetworks: '{"ingressCnlbNetworks":["nf-sig6-ut1/10.121.44.173","nf-sig5-ut1/10.121.44.167","nf-sig2-ut1/10.121.44.149","nf-sig1-ut2/10.121.44.144","nf-sig4-ut2/10.121.44.161","nf-sig3-ut3/10.121.44.157"]}'
kind: ConfigMap
metadata:
  creationTimestamp: "2024-07-28T11:31:54Z"
  name: ocnadd-configmap-cnlb-ingress
```

Ensure that at least one NAD entry is present in the ConfigMaps "ocnadd-configmap-cnlb-egress" and "ocnadd-configmap-cnlb-ingress" to enable traffic segregation.

2.5 Extended Storage with Druid Troubleshooting Scenarios

This section should be referred to for troubleshooting issues encountered after OCNADD integration with the DRUID database cluster. The issues mentioned below are generic and can be investigated using the Druid unified console or APIs.

Getting the list of active Supervisor Spec from Druid

A Supervisor Spec is created for every correlation configuration that supports extended storage when the extended storage type is set to Druid database. The API below provides a way to fetch the active Supervisor Specs running for Kafka ingestion.

URL: `http://<druid_router_ip>:<druid_router_port>/druid/indexer/v1/supervisor`

Endpoint: `/druid/indexer/v1/supervisor`

Method: GET

Request Body: None

Response: 200 OK – Returns a List<String>

Description: Returns an array of strings representing the names of active supervisors. If there are no active supervisors, it returns an empty array.

Example Response:

```
[
  "corr1_tdr",
  "corr2_sudr"
]
```

2. List Active Supervisors (Full Supervisor Objects)

Endpoint: /druid/indexer/v1/supervisor?full=null

Method: GET

Request Body: None

Response: 200 OK – Returns a List<Supervisor>

Description: Returns an array of Supervisor objects. If there are no active supervisors, it returns an empty array.

Example:

```
[
  {
    "id": "joe_corr_feed_2XDR_dvm1",
    "state": "RUNNING",
    "detailedState": "RUNNING",
    "healthy": true,
    "spec": {
      "type": "kafka",
      "spec": {
        "dataSchema": {
          "dataSource": "joe_corr_feed_2XDR_dvm1",
          "timestampSpec": {
            "column": "beginTime",
            "format": "iso",
            "missingValue": null
          },
          "dimensionsSpec": {
            "dimensions": [
              { "type": "string", "name": "version", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
              { "type": "string", "name": "beginTime", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
              { "type": "string", "name": "endTime", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
              { "type": "string", "name": "configurationName",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
              { "type": "string", "name": "xdrStatus", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
              { "type": "long", "name": "totalPduCount",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": false },
              { "type": "long", "name": "totalLength", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": false },
              { "type": "string", "name": "transactionTime",
```

```

"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "path", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "methodType", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "egressAuthority",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "feedSourceNfFqdn",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "feedSourceNfId",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "producerFqdn",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "contentType",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "supi", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "ingressAuthority",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "consumerId", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "consumerFqdn",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "ueId", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "consumerNfType",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "xdrId", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "gpsi", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "pei", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "statusCode", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "userAgent", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "transactionId",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "feedSourceNfType",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "producerId", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "registrationTime",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "smfInstanceId",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "smfSetId", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "snssai", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "dnn", "multiValueHandling":
"SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "pcfInstanceId",
"multiValueHandling": "SORTEd_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "consumerVia",

```

```

"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "producerVia",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "consumerPlmn",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "producerPlmn",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "long", "name": "pduSessionId", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": false },
  { "type": "json", "name": "messages", "formatVersion": 5,
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true }
],
"dimensionExclusions": ["__time"],
"includeAllDimensions": false,
"useSchemaDiscovery": false
},
"metricsSpec": [],
"granularitySpec": {
  "type": "uniform",
  "segmentGranularity": "HOUR",
  "queryGranularity": { "type": "none" },
  "rollup": false,
  "intervals": []
},
"transformSpec": {
  "filter": null,
  "transforms": [
    { "type": "expression", "name": "beginTime", "expression":
"timestamp_format(__time)" },
    { "type": "expression", "name": "xdrId", "expression":
"timestamp(endTime)+1" }
  ]
}
},
"ioConfig": {
  "topic": "ADC-CORR-FEED-2-CORRELATED",
  "topicPattern": null,
  "inputFormat": { "type": "json", "keepNullColumns": false },
  "replicas": 1,
  "taskCount": 1,
  "taskDuration": "PT900S",
  "consumerProperties": {
    "auto.offset.reset": "earliest",
    "bootstrap.servers": "kafka-broker-0.kafka-broker-headless.iq-s-
mgmt.svc.dv-m1.lab.in.oracle.com:9094,kafka-broker-1.kafka-broker-headless.iq-
s-mgmt.svc.dv-m1.lab.in.oracle.com:9094,kafka-broker-2.kafka-broker-
headless.iq-s-mgmt.svc.dv-m1.lab.in.oracle.com:9094,kafka-broker-3.kafka-
broker-headless.iq-s-mgmt.svc.dv-m1.lab.in.oracle.com:9094",
    "ssl.protocol": "TLSv1.3",
    "security.protocol": "SASL_SSL",
    "ssl.keystore.location": "/var/securityfiles/keystore/
serverKeyStore.p12",
    "ssl.truststore.location": "/var/securityfiles/keystore/
trustStore.p12",
    "ssl.truststore.type": "PKCS12",
    "ssl.keystore.type": "PKCS12",

```

```

    "druid.dynamic.config.provider": {
      "type": "environment",
      "variables": {
        "ssl.keystore.password": "CLIENT_KS_PASSWORD",
        "ssl.truststore.password": "CLIENT_TS_PASSWORD"
      }
    },
    "sasl.mechanism": "SCRAM-SHA-256",
    "sasl.jaas.config":
      "org.apache.kafka.common.security.scram.ScramLoginModule required
      username=\"extuser1\" password=\"extuser1\";"
  },
  "autoScalerConfig": null,
  "pollTimeout": 100,
  "startDelay": "PT5S",
  "period": "PT30S",
  "useEarliestOffset": true,
  "completionTimeout": "PT1800S",
  "lateMessageRejectionPeriod": null,
  "earlyMessageRejectionPeriod": null,
  "lateMessageRejectionStartDateTime": null,
  "configOverrides": null,
  "idleConfig": null,
  "stopTaskCount": null,
  "emitTimeLagMetrics": false,
  "stream": "JOE-CORR-FEED-2-CORRELATED",
  "useEarliestSequenceNumber": true
},
"tuningConfig": {
  "type": "kafka",
  "appendableIndexSpec": { "type": "onheap",
"preserveExistingMetrics": false },
  "maxRowsInMemory": 150000,
  "maxBytesInMemory": 0,
  "skipBytesInMemoryOverheadCheck": false,
  "maxRowsPerSegment": 50000,
  "maxTotalRows": null,
  "intermediatePersistPeriod": "PT1M",
  "maxPendingPersists": 0,
  "indexSpec": {
    "bitmap": { "type": "roaring" },
    "dimensionCompression": "lz4",
    "stringDictionaryEncoding": { "type": "utf8" },
    "metricCompression": "lz4",
    "longEncoding": "longs"
  },
  "indexSpecForIntermediatePersists": {
    "bitmap": { "type": "roaring" },
    "dimensionCompression": "lz4",
    "stringDictionaryEncoding": { "type": "utf8" },
    "metricCompression": "lz4",
    "longEncoding": "longs"
  },
  "reportParseExceptions": false,
  "handoffConditionTimeout": 900000,
  "resetOffsetAutomatically": true,

```

```

    "segmentWriteOutMediumFactory": null,
    "workerThreads": null,
    "chatRetries": 8,
    "httpTimeout": "PT10S",
    "shutdownTimeout": "PT80S",
    "offsetFetchPeriod": "PT30S",
    "intermediateHandoffPeriod": "P2147483647D",
    "logParseExceptions": false,
    "maxParseExceptions": 2147483647,
    "maxSavedParseExceptions": 0,
    "numPersistThreads": 1,
    "maxColumnsToMerge": -1,
    "skipSequenceNumberAvailabilityCheck": false,
    "repartitionTransitionDuration": "PT120S"
  }
},
"dataSchema": {
  "dataSource": "joe_corr_feed_2XDR_dvm1",
  "timestampSpec": {
    "column": "beginTime",
    "format": "iso",
    "missingValue": null
  },
  "dimensionsSpec": {
    "dimensions": [
      { "type": "string", "name": "version", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "string", "name": "beginTime", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "string", "name": "endTime", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "string", "name": "configurationName",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "string", "name": "xdrStatus", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "long", "name": "totalPduCount", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": false },
      { "type": "long", "name": "totalLength", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": false },
      { "type": "string", "name": "transactionTime",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "string", "name": "path", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "string", "name": "methodType", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "string", "name": "egressAuthority",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "string", "name": "feedSourceNfFqdn",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "string", "name": "feedSourceNfId",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "string", "name": "producerFqdn", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "string", "name": "contentType", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
      { "type": "string", "name": "supi", "multiValueHandling":

```

```

"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "ingressAuthority",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "consumerId", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "consumerFqdn", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "ueId", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "consumerNfType",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "xdrId", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "gpsi", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "pei", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "statusCode", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "userAgent", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "transactionId",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "feedSourceNfType",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "producerId", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "registrationTime",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "smfInstanceId",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "smfSetId", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "snssai", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "dnn", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "pcfInstanceId",
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "consumerVia", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "producerVia", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "consumerPlmn", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "string", "name": "producerPlmn", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": true },
  { "type": "long", "name": "pduSessionId", "multiValueHandling":
"SORTED_ARRAY", "createBitmapIndex": false },
  { "type": "json", "name": "messages", "formatVersion": 5,
"multiValueHandling": "SORTED_ARRAY", "createBitmapIndex": true }
},
"dimensionExclusions": ["__time"],
"includeAllDimensions": false,
"useSchemaDiscovery": false
},

```

```

    "metricsSpec": [],
    "granularitySpec": {
      "type": "uniform",
      "segmentGranularity": "HOUR",
      "queryGranularity": { "type": "none" },
      "rollup": false,
      "intervals": []
    },
    "transformSpec": {
      "filter": null,
      "transforms": [
        { "type": "expression", "name": "beginTime", "expression":
"timestamp_format(__time)" },
        { "type": "expression", "name": "xdrId", "expression":
"timestamp(endTime)+1" }
      ]
    },
    "tuningConfig": {
      "type": "kafka",
      "appendableIndexSpec": { "type": "onheap", "preserveExistingMetrics":
false },
      "maxRowsInMemory": 150000,
      "maxBytesInMemory": 0,
      "skipBytesInMemoryOverheadCheck": false,
      "maxRowsPerSegment": 50000,
      "maxTotalRows": null,
      "intermediatePersistPeriod": "PT1M",
      "maxPendingPersists": 0,
      "indexSpec": {
        "bitmap": { "type": "roaring" },
        "dimensionCompression": "lz4",
        "stringDictionaryEncoding": { "type": "utf8" },
        "metricCompression": "lz4",
        "longEncoding": "longs"
      },
      "indexSpecForIntermediatePersists": {
        "bitmap": { "type": "roaring" },
        "dimensionCompression": "lz4",
        "stringDictionaryEncoding": { "type": "utf8" },
        "metricCompression": "lz4",
        "longEncoding": "longs"
      },
      "reportParseExceptions": false,
      "handoffConditionTimeout": 900000,
      "resetOffsetAutomatically": true,
      "segmentWriteOutMediumFactory": null,
      "workerThreads": null,
      "chatRetries": 8,
      "httpTimeout": "PT10S",
      "shutdownTimeout": "PT80S",
      "offsetFetchPeriod": "PT30S",
      "intermediateHandoffPeriod": "P2147483647D",
      "logParseExceptions": false,
      "maxParseExceptions": 2147483647,
      "maxSavedParseExceptions": 0,

```

```

        "numPersistThreads": 1,
        "maxColumnsToMerge": -1,
        "skipSequenceNumberAvailabilityCheck": false,
        "repartitionTransitionDuration": "PT120S"
    },
    "ioConfig": {
        "topic": "JOE-CORR-FEED-2-CORRELATED",
        "topicPattern": null,
        "inputFormat": { "type": "json", "keepNullColumns": false },
        "replicas": 1,
        "taskCount": 1,
        "taskDuration": "PT900S",
        "consumerProperties": {
            "auto.offset.reset": "earliest",
            "bootstrap.servers": "kafka-broker-0.kafka-broker-headless.iq-s-
mgmt.svc.dv-m1.lab.in.oracle.com:9094,kafka-broker-1.kafka-broker-headless.iq-
s-mgmt.svc.dv-m1.lab.in.oracle.com:9094,kafka-broker-2.kafka-broker-
headless.iq-s-mgmt.svc.dv-m1.lab.in.oracle.com:9094,kafka-broker-3.kafka-
broker-headless.iq-s-mgmt.svc.dv-m1.lab.in.oracle.com:9094",
            "ssl.protocol": "TLSv1.3",
            "security.protocol": "SASL_SSL",
            "ssl.keystore.location": "/var/securityfiles/keystore/
serverKeyStore.p12",
            "ssl.truststore.location": "/var/securityfiles/keystore/
trustStore.p12",
            "ssl.truststore.type": "PKCS12",
            "ssl.keystore.type": "PKCS12",
            "druid.dynamic.config.provider": {
                "type": "environment",
                "variables": {
                    "ssl.keystore.password": "CLIENT_KS_PASSWORD",
                    "ssl.truststore.password": "CLIENT_TS_PASSWORD"
                }
            }
        },
        "sasl.mechanism": "SCRAM-SHA-256",
        "sasl.jaas.config":
"org.apache.kafka.common.security.scram.ScramLoginModule required
username=\"extuser1\" password=\"extuser1\";"
    },
    "autoScalerConfig": null,
    "pollTimeout": 100,
    "startDelay": "PT5S",
    "period": "PT30S",
    "useEarliestOffset": true,
    "completionTimeout": "PT1800S",
    "lateMessageRejectionPeriod": null,
    "earlyMessageRejectionPeriod": null,
    "lateMessageRejectionStartDateTime": null,
    "configOverrides": null,
    "idleConfig": null,
    "stopTaskCount": null,
    "emitTimeLagMetrics": false,
    "stream": "JOE-CORR-FEED-2-CORRELATED",
    "useEarliestSequenceNumber": true
},
"context": null,

```

```

        "suspended": false
      },
      "suspended": false
    }
  ]
}

```

3. Get Supervisor Specification

Endpoint: /druid/indexer/v1/supervisor/{supervisorId}

Method: GET

Request Body: None

Response:

- 200 OK – Returns IngestionSpec
- 404 NotFound – None

Description: Retrieves the specification for a single supervisor. The returned specification includes:

- dataSchema
- ioConfig
- tuningConfig

Example:

```

{
  "type": "kafka",
  "spec": {
    "dataSchema": {
      "dataSource": "DD_FEED2XDR",
      "timestampSpec": {
        "column": "beginTime",
        "format": "iso",
        "missingValue": null
      },
      "dimensionsSpec": {
        "dimensions": [
          {
            "type": "string",
            "name": "version",
            "multiValueHandling": "SORTED_ARRAY",
            "createBitmapIndex": true
          },
          {
            "type": "string",
            "name": "beginTime",
            "multiValueHandling": "SORTED_ARRAY",
            "createBitmapIndex": true
          },
          {
            "type": "string",
            "name": "endTime",
            "multiValueHandling": "SORTED_ARRAY",
            "createBitmapIndex": true
          }
        ]
      }
    }
  }
}

```

```

    },
    {
      "type": "string",
      "name": "configurationName",
      "multiValueHandling": "SORTED_ARRAY",
      "createBitmapIndex": true
    },
    {
      "type": "string",
      "name": "xdrStatus",
      "multiValueHandling": "SORTED_ARRAY",
      "createBitmapIndex": true
    },
    {
      "type": "long",
      "name": "totalPduCount",
      "multiValueHandling": "SORTED_ARRAY",
      "createBitmapIndex": false
    },
    {
      "type": "long",
      "name": "totalLength",
      "multiValueHandling": "SORTED_ARRAY",
      "createBitmapIndex": false
    },
    {
      "type": "string",
      "name": "transactionTime",
      "multiValueHandling": "SORTED_ARRAY",
      "createBitmapIndex": true
    },
    {
      "type": "string",
      "name": "path",
      "multiValueHandling": "SORTED_ARRAY",
      "createBitmapIndex": true
    },
    {
      "type": "string",
      "name": "methodType",
      "multiValueHandling": "SORTED_ARRAY",
      "createBitmapIndex": true
    },
    {
      "type": "string",
      "name": "egressAuthority",
      "multiValueHandling": "SORTED_ARRAY",
      "createBitmapIndex": true
    },
    {
      "type": "string",
      "name": "feedSourceNffqdn",
      "multiValueHandling": "SORTED_ARRAY",
      "createBitmapIndex": true
    },
  },
  {

```

```

        "type": "string",
        "name": "feedSourceNfId",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "producerFqdn",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "contentType",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "supi",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "ingressAuthority",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "consumerId",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "consumerFqdn",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "ueId",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "consumerNfType",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "xdrId",

```

```

        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "gpsi",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "pei",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "statusCode",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "userAgent",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "transactionId",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "feedSourceNfType",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "producerId",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "registrationTime",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    },
    {
        "type": "string",
        "name": "smfInstanceId",
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    }

```

```
    },  
    {  
      "type": "string",  
      "name": "smfSetId",  
      "multiValueHandling": "SORTED_ARRAY",  
      "createBitmapIndex": true  
    },  
    {  
      "type": "string",  
      "name": "snssai",  
      "multiValueHandling": "SORTED_ARRAY",  
      "createBitmapIndex": true  
    },  
    {  
      "type": "string",  
      "name": "dnn",  
      "multiValueHandling": "SORTED_ARRAY",  
      "createBitmapIndex": true  
    },  
    {  
      "type": "string",  
      "name": "pcfInstanceId",  
      "multiValueHandling": "SORTED_ARRAY",  
      "createBitmapIndex": true  
    },  
    {  
      "type": "string",  
      "name": "consumerVia",  
      "multiValueHandling": "SORTED_ARRAY",  
      "createBitmapIndex": true  
    },  
    {  
      "type": "string",  
      "name": "producerVia",  
      "multiValueHandling": "SORTED_ARRAY",  
      "createBitmapIndex": true  
    },  
    {  
      "type": "string",  
      "name": "consumerPlmn",  
      "multiValueHandling": "SORTED_ARRAY",  
      "createBitmapIndex": true  
    },  
    {  
      "type": "string",  
      "name": "producerPlmn",  
      "multiValueHandling": "SORTED_ARRAY",  
      "createBitmapIndex": true  
    },  
    {  
      "type": "long",  
      "name": "pduSessionId",  
      "multiValueHandling": "SORTED_ARRAY",  
      "createBitmapIndex": false  
    },  
  ],  
  {
```

```

        "type": "json",
        "name": "messages",
        "formatVersion": 5,
        "multiValueHandling": "SORTED_ARRAY",
        "createBitmapIndex": true
    }
],
"dimensionExclusions": [
    "__time"
],
"includeAllDimensions": false,
"useSchemaDiscovery": false
},
"metricsSpec": [],
"granularitySpec": {
    "type": "uniform",
    "segmentGranularity": "HOUR",
    "queryGranularity": {
        "type": "none"
    },
    "rollup": false,
    "intervals": []
},
"transformSpec": {
    "filter": null,
    "transforms": [
        {
            "type": "expression",
            "name": "beginTime",
            "expression": "timestamp_format(__time)"
        },
        {
            "type": "expression",
            "name": "xdrId",
            "expression": "timestamp(endTime)+1"
        }
    ]
}
},
"ioConfig": {
    "topic": "FEED2-CORRELATED",
    "topicPattern": null,
    "inputFormat": {
        "type": "json",
        "keepNullColumns": false,
        "assumeNewlineDelimited": false,
        "useJsonNodeReader": false
    },
    "replicas": 1,
    "taskCount": 1,
    "taskDuration": "PT3600S",
    "consumerProperties": {
        "bootstrap.servers": "192.168.56.108:9092"
    },
    "autoScalerConfig": null,
    "pollTimeout": 100,

```

```

    "startDelay": "PT5S",
    "period": "PT30S",
    "useEarliestOffset": false,
    "completionTimeout": "PT1800S",
    "lateMessageRejectionPeriod": null,
    "earlyMessageRejectionPeriod": null,
    "lateMessageRejectionStartDateTime": null,
    "configOverrides": null,
    "idleConfig": null,
    "stopTaskCount": null,
    "stream": "FEED2-CORRELATED",
    "useEarliestSequenceNumber": false
  },
  "tuningConfig": {
    "type": "kafka",
    "appendableIndexSpec": {
      "type": "onheap",
      "preserveExistingMetrics": false
    },
    "maxRowsInMemory": 150000,
    "maxBytesInMemory": 0,
    "skipBytesInMemoryOverheadCheck": false,
    "maxRowsPerSegment": 50000,
    "maxTotalRows": null,
    "intermediatePersistPeriod": "PT10M",
    "maxPendingPersists": 0,
    "indexSpec": {
      "bitmap": {
        "type": "roaring"
      },
      "dimensionCompression": "lz4",
      "stringDictionaryEncoding": {
        "type": "utf8"
      },
      "metricCompression": "lz4",
      "longEncoding": "longs"
    },
    "indexSpecForIntermediatePersists": {
      "bitmap": {
        "type": "roaring"
      },
      "dimensionCompression": "lz4",
      "stringDictionaryEncoding": {
        "type": "utf8"
      },
      "metricCompression": "lz4",
      "longEncoding": "longs"
    },
    "reportParseExceptions": false,
    "handoffConditionTimeout": 900000,
    "resetOffsetAutomatically": false,
    "segmentWriteOutMediumFactory": null,
    "workerThreads": null,
    "chatRetries": 8,
    "httpTimeout": "PT10S",
    "shutdownTimeout": "PT80S",

```

```

        "offsetFetchPeriod": "PT30S",
        "intermediateHandoffPeriod": "P2147483647D",
        "logParseExceptions": false,
        "maxParseExceptions": 2147483647,
        "maxSavedParseExceptions": 0,
        "numPersistThreads": 1,
        "skipSequenceNumberAvailabilityCheck": false,
        "repartitionTransitionDuration": "PT120S"
    }
},
"context": null,
"suspended": false
}

```

Getting the Health, Status and Stats of the supervisor

URL: `http://<druid_router_ip>:<druid_router_port>/druid/indexer/v1/supervisor`

1. Get Supervisor Health

Endpoint: `/druid/indexer/v1/supervisor/{supervisorId}/health`

Method: GET

Request Body: None

Response:

- 200 OK – Returns a JSON Object
- 404 Not Found – "Invalid SupervisorID"
- 503 Service Unavailable – "Supervisor is unhealthy"

Description: Retrieves the current health report for a single supervisor. The health of a supervisor is determined by the supervisor's state (as returned by the `/status` endpoint).

Example Response:

```

{
  "healthy": false
}

```

2. Get Supervisor Stats

Endpoint: `/druid/indexer/v1/supervisor/{supervisorId}/stats`

Method: GET

Request Body: None

Response:

- 200 OK – Returns a JSON Object
- 404 Not Found – "Cannot find any supervisor with id: [\"XXX\"]"

Description: Returns a snapshot of the current ingestion row counters for each task being managed by the supervisor, along with moving averages for the row counters.

Example:

```
{
  "0": {
    "index_kafka_adc_corr_feed_2XDR_dvm1_a971c56a6a3aa33_kioblpa": {
      "movingAverages": {
        "buildSegments": {
          "5m": {
            "processed": 11.075906329418917,
            "processedBytes": 13420.280089272523,
            "unparseable": 0.0,
            "thrownAway": 0.0,
            "processedWithError": 0.0
          },
          "15m": {
            "processed": 16.880517867690518,
            "processedBytes": 20190.08795040925,
            "unparseable": 0.0,
            "thrownAway": 0.0,
            "processedWithError": 0.0
          },
          "1m": {
            "processed": 10.000016421654086,
            "processedBytes": 12169.78312232924,
            "unparseable": 0.0,
            "thrownAway": 0.0,
            "processedWithError": 0.0
          }
        }
      }
    }
  },
  "totals": {
    "buildSegments": {
      "processed": 8497,
      "processedBytes": 10334893,
      "processedWithError": 0,
      "thrownAway": 0,
      "unparseable": 0
    }
  }
}
```

3. Get Supervisor Status**Endpoint:** /druid/indexer/v1/supervisor/{supervisorId}/status**Method:** GET**Request Body:** None**Response:**

- 200 OK – Returns a JSON Object
- 404 NotFound – "Cannot find any supervisor with id: [\"xxx\"]"

Description: Retrieves the current status report for a single supervisor. The report contains:

- The state of the supervisor tasks
- An array of recently thrown exceptions

Example:

```
{
  "id": "adc_corr_feed_2XDR_dvm1",
  "generationTime": "2025-08-11T04:44:09.346Z",
  "payload": {
    "dataSource": "adc_corr_feed_2XDR_dvm1",
    "stream": "ADC-CORR-FEED-2-CORRELATED",
    "partitions": 6,
    "replicas": 1,
    "durationSeconds": 900,
    "activeTasks": [
      {
        "id": "index_kafka_adc_corr_feed_2XDR_dvm1_1951ed0543fa752_hggimlaf",
        "startingOffsets": {
          "1": 390784,
          "0": 389009,
          "3": 389412,
          "2": 389161,
          "5": 389612,
          "4": 390184
        },
        "startTime": "2025-08-11T04:41:11.149Z",
        "remainingSeconds": 721,
        "type": "ACTIVE",
        "currentOffsets": {
          "1": 391078,
          "0": 389256,
          "3": 389701,
          "2": 389366,
          "5": 389901,
          "4": 390473
        },
        "lag": {
          "1": 0,
          "0": 0,
          "3": 0,
          "2": 0,
          "5": 0,
          "4": 0
        }
      }
    ],
    "publishingTasks": [],
    "latestOffsets": {
      "1": 391078,
      "0": 389256,
      "3": 389701,
      "2": 389366,
      "5": 389901,
      "4": 390473
    },
    "minimumLag": {
```

```

    "1": 0,
    "0": 0,
    "3": 0,
    "2": 0,
    "5": 0,
    "4": 0
  },
  "aggregateLag": 0,
  "offsetsLastUpdated": "2025-08-11T04:43:43.314Z",
  "suspended": false,
  "healthy": true,
  "state": "RUNNING",
  "detailedState": "RUNNING",
  "recentErrors": []
}
}

```

Indexing Jobs not processing due to missing offsets

Problem: The Kafka ingestion job is not processing data due to missing offsets in Druid. A **"Druid Kafka offset error"**, specifically an `OffsetOutOfRangeException`, occurs when a Druid Kafka indexing task tries to consume data from a Kafka topic offset that no longer exists. This usually happens when messages in Kafka are deleted according to the retention policy before Druid can ingest them.

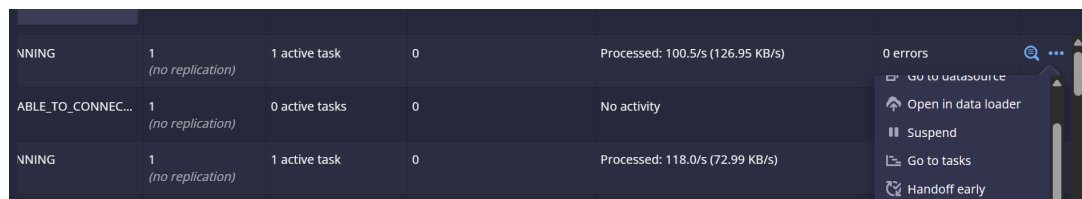
Solution:

1. Login to the Druid web console using the following URL:

```
http://<druid_router_ip>:<druid_router_port>/unified-console.html#supervisors
```

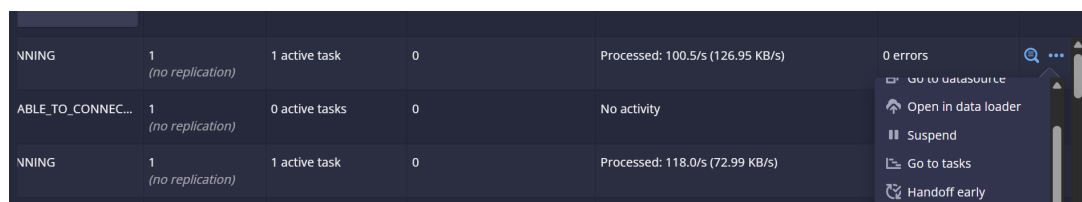
2. Access the affected supervisor:

Scroll to the right side of the supervisor configuration and click the ... menu. You will see the **Suspend** option.



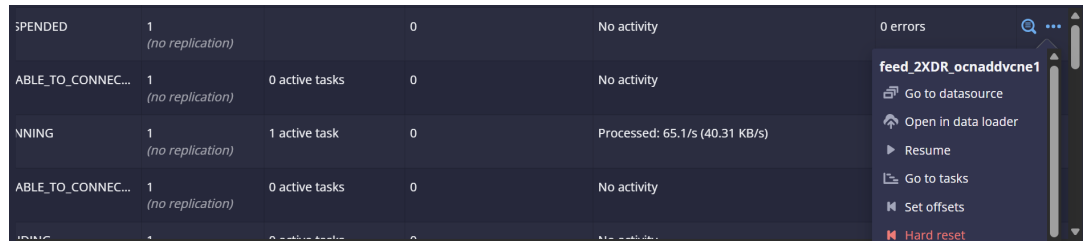
3. Suspend the supervisor:

Click **Suspend** to pause the supervisor.



4. Hard reset the offsets:

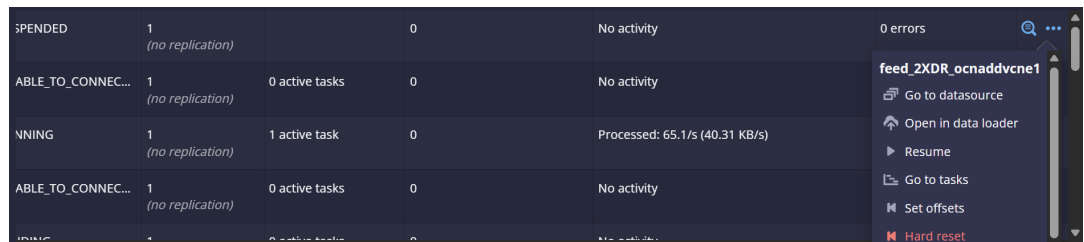
Click the ... menu again and select **Hard Reset**.



Task ID	Replication	Active Tasks	Errors	Activity
PENDING	1 (no replication)	0	0	No activity
ABLE_TO_CONNEC...	1 (no replication)	0 active tasks	0	No activity
UNING	1 (no replication)	1 active task	0	Processed: 65.1/s (40.31 KB/s)
ABLE_TO_CONNEC...	1 (no replication)	0 active tasks	0	No activity

Task ID	Replication	Active Tasks	Errors	Activity
PENDING	1 (no replication)	0	0	No activity
ABLE_TO_CONNEC...	1 (no replication)	0 active tasks	0	No activity
UNING	1 (no replication)	1 active task	0	Processed: 65.1/s (40.31 KB/s)
ABLE_TO_CONNEC...	1 (no replication)	0 active tasks	0	No activity

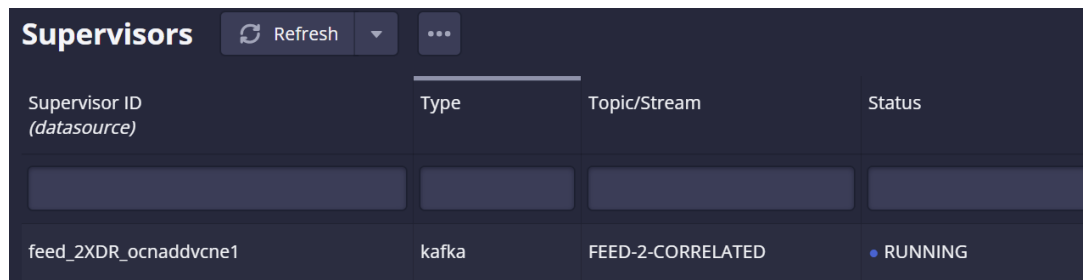
- Resume the supervisor:**
Click **Resume** to restart the supervisor configuration.



Task ID	Replication	Active Tasks	Errors	Activity
PENDING	1 (no replication)	0	0	No activity
ABLE_TO_CONNEC...	1 (no replication)	0 active tasks	0	No activity
UNING	1 (no replication)	1 active task	0	Processed: 65.1/s (40.31 KB/s)
ABLE_TO_CONNEC...	1 (no replication)	0 active tasks	0	No activity

Task ID	Replication	Active Tasks	Errors	Activity
PENDING	1 (no replication)	0	0	No activity
ABLE_TO_CONNEC...	1 (no replication)	0 active tasks	0	No activity
UNING	1 (no replication)	1 active task	0	Processed: 65.1/s (40.31 KB/s)
ABLE_TO_CONNEC...	1 (no replication)	0 active tasks	0	No activity

- Verify ingestion:**
The supervisor should enter the **Running** state, and data ingestion should start successfully.



Supervisor ID (datasource)	Type	Topic/Stream	Status
feed_2XDR_ocnaddvcne1	kafka	FEED-2-CORRELATED	● RUNNING

Running Sql native query on Druid console

- Login to the Druid web console using the following URL:**
`http://<druid_router_ip>:<druid_router_port>/unified-console.html`
- Select the Query tab:**
From the top menu, click on **Query**.
- Run the SQL query:**
Run the SQL query for the corresponding datasource directly on the web console, as shown below.

druid

Search

feed_2XDR_ocnaddvcne1

feed_3XDR_ocnaddvcne1

Tab 1

```
1 SELECT *
2 FROM "feed_2XDR_ocnaddvcne1"
3 WHERE _time >= CURRENT_TIMESTAMP - INTERVAL '1' DAY
```

Run Engine: Auto [SQL (native)] 1,000+ results in 0.29s

_time	A version	A beginTime	A endTime	A configurationName	A xdrStatus
2025-08-11T05:00:00.124Z	2.0.0	2025-08-11T05:00:00.12	2025-08-11T05:00:00.12	feed-2	SUDR
2025-08-11T05:00:00.224Z	2.0.0	2025-08-11T05:00:00.22	2025-08-11T05:00:00.22	feed-2	SUDR

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 Helm 3.
- When collecting logs, capture them from all OCNADD groups: Management, Relay Agent, and Mediation.

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

List of Alerts

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

5.1 System Level Alerts

This section lists the system level alerts for OCNADD.

Management Group Alerts

OCNADD_POD_CPU_USAGE_ALERT

Table 5-1 OCNADD_POD_CPU_USAGE_ALERT

Field	Details
Triggering Condition	POD CPU usage is above the set threshold (default 85%)
Severity	Major
Description	OCNADD Pod High CPU usage detected for a continuous period of 5min

Table 5-1 (Cont.) OCNADD_POD_CPU_USAGE_ALERT

Field	Details
Alert Details CNE	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 }} % ' PromQL Expression: expr: <pre>(sum(rate(container_cpu_usage_seconds_total{image!="", pod=~".*ocnaddalarm.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmanagement.ocnaddalarm.ocnaddalarm.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="", pod=~".*ocnaddconfiguration.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmanagement.ocnaddconfiguration.ocnaddconfiguration.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="", pod=~".*ocnaddgui.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmanagement.ocnaddgui.ocnaddgui.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="", pod=~".*ocnaddhealthmonitoring.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmanagement.ocnaddhealthmonitoring.ocnaddhealthmonitoring.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="", pod=~".*ocnadduirouter.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmanagement.ocnadduirouter.ocnadduirouter.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="", pod=~".*ocnaddexport.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmanagement.ocnaddexport.ocnaddexport.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="", pod=~".*ocnaddmanagementgateway.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmanagement.ocnaddmanagementgateway.ocnaddmanagementgateway.resources.limits.cpu }})</pre>

Table 5-1 (Cont.) OCNADD_POD_CPU_USAGE_ALERT

Field	Details
Alert Details OCI	Summary: Alarm "OCNADD_POD_CPU_USAGE_ALERT" is in a "X" state; because n metrics meet the trigger rule: "pod_cpu_usage_seconds_total[10m]{pod=~"*alarm"*admin"*health"*config"*management*".rate().groupby(namespace,pod).sum()*100>=85 pod_cpu_usage_seconds_total[10m]{pod=~"*ui*".rate().groupby(namespace,pod).sum()*100>=2*85 pod_cpu_usage_seconds_total[10m]{pod=~"*export*".rate().groupby(namespace,pod).sum()*100>=4*85", with a trigger delay of 1 minute where, X = FIRING/OK, n = Different services that violated the rule. MQL Expression: pod_cpu_usage_seconds_total[10m]{pod=~"*alarm"*admin"*health"*config"*management*".rate().groupby(namespace,pod).sum()*100>={{ CPU Threshold }} pod_cpu_usage_seconds_total[10m]{pod=~"*ui*".rate().groupby(namespace,pod).sum()*100>=2*{{ CPU Threshold }} pod_cpu_usage_seconds_total[10m]{pod=~"*export*".rate().groupby(namespace,pod).sum()*100>=4*{{ CPU Threshold }} Note: CPU Threshold will be assigned will executing the terraform script
OID	1.3.6.1.4.1.323.5.3.53.1.29.4002
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 a similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the CPU utilization is below the critical threshold. Note: The thresholds are configurable in the <i>ocnadd-common-custom-values.yaml</i> & <i>ocnadd-management-custom-values.yaml</i> files. If guidance is required, contact My Oracle Support .

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 90%)
Severity	Major
Description	OCNADD Pod High Memory usage detected for a continuous period of 5min

Table 5-2 (Cont.) OCNADD_POD_MEMORY_USAGE_ALERT

Field	Details
Alert Details CNE	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 }} % ' PromQL Expression: expr: (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddalarm.*"}) by (pod,namespace) > {{ .Values.ocnaddmanagement.ocnaddalarm.ocnaddalarm.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddconfiguration.*"}) by (pod,namespace) > {{ .Values.ocnaddmanagement.ocnaddconfiguration.ocnaddconfiguration.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddgui.*"}) by (pod,namespace) > {{ .Values.ocnaddmanagement.ocnaddgui.ocnaddgui.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddhealthmonitoring.*"}) by (pod,namespace) > {{ .Values.ocnaddmanagement.ocnaddhealthmonitoring.ocnaddhealthmonitoring.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnadduirouter.*"}) by (pod,namespace) > {{ .Values.ocnaddmanagement.ocnadduirouter.ocnadduirouter.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddexport.*"}) by (pod,namespace) > {{ .Values.ocnaddmanagement.ocnaddexport.ocnaddexport.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddmanagementgateway.*"}) by (pod,namespace) > {{ .Values.ocnaddmanagement.ocnaddmanagementgateway.ocnaddmanagementgateway.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100)

Table 5-2 (Cont.) OCNADD_POD_MEMORY_USAGE_ALERT

Field	Details
Alert Details OCI	Summary: Alarm "OCNADD_POD_MEMORY_USAGE_ALERT" is in a "X" state; because n metrics meet the trigger rule: "container_memory_usage_bytes[5m]{pod=~"*alarm*"*admin*"*health*"*config*"*management*"*export*"*ui*"}.groupby(namespace,pod).sum()/container_spec_memory_limit_bytes[5m]{pod=~"*alarm*"*admin*"*health*"*config*"*management*"*export*"*ui*"}.groupby(namespace,pod).sum()*100>=90", with a trigger delay of 1 minute where, X = FIRING/OK, n = Different services that violated the rule. MQL Expression: container_memory_usage_bytes[10m]{pod=~"*alarm*"*admin*"*health*"*config*"*management*"*export*"*ui*"}.groupby(namespace,pod).sum()/container_spec_memory_limit_bytes[10m]{pod=~"*alarm*"*admin*"*health*"*config*"*management*"*export*"*ui*"}.groupby(namespace,pod).sum()*100>={{ Memory Threshold }}
OID	1.3.6.1.4.1.323.5.3.53.1.29.4005
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 a similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the memory utilization is below the critical threshold. Note: The thresholds are configurable in the <i>ocnadd-common-custom-values.yaml</i> & <i>ocnadd-management-custom-values.yaml</i> files. If guidance is required, contact My Oracle Support.

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 the last 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} } . first value humanizeTimestamp }}{{ "{{" }} end }}: A Pod has restarted' PromQL Expression: expr: kube_pod_container_status_restarts_total{namespace="{{ .Values.globals.ocnaddmanagement.cluster.nameSpace.name }}" } > 1
Alert Details OCI	MQL Expression: No MQL equivalent is available
OID	1.3.6.1.4.1.323.5.3.53.1.29.5006

Table 5-3 (Cont.) OCNADD_POD_RESTARTED

Field	Details
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 a similar metric as exposed 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: <code>kubectl get po -n <namespace></code> Note the full name of the pod that is not running, and use it in the following command: <code>kubectl describe pod <desired full pod name> -n <namespace></code> 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. If guidance is required.

RelayAgent Group Alerts

OCNADD_POD_CPU_USAGE_ALERT

Table 5-4 OCNADD_POD_CPU_USAGE_ALERT

Field	Details
Triggering Condition	POD CPU usage is above the set threshold (default 85%)
Severity	Major
Description	OCNADD Pod High CPU usage detected for a continuous period of 5min

Table 5-4 (Cont.) OCNADD_POD_CPU_USAGE_ALERT

Field	Details
Alert Details CNE	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 }} % ' PromQL Expression: expr: <pre>(sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*ocnaddbsfaggregation.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddrelayagent.oc naddaggregation.ocnaddbsfaggregation.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*ocnaddnrfaggregation.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddrelayagent.oc naddaggregation.ocnaddnrfaggregation.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*ocnaddpcfaggregation.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddrelayagent.oc naddaggregation.ocnaddpcfaggregation.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*ocnaddscpaggregation.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddrelayagent.oc naddaggregation.ocnaddscpaggregation.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*ocnaddseppaggregation.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddrelayagent.oc naddaggregation.ocnaddseppaggregation.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*kafka-broker.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddrelayagent.oc naddkafka.ocnadd.kafkaBroker.resource.limit.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*kraft-controller.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddrelayagent.oc naddkafka.ocnadd.kraftController.resource.limit.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*ocnaddrelayagentgateway.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddrelayagent.oc naddrelayagentgateway.ocnaddrelayagentgateway.resources.limits.cpu }}))</pre>

Table 5-4 (Cont.) OCNADD_POD_CPU_USAGE_ALERT

Field	Details
Alert Details OCI	Summary: Alarm "OCNADD_POD_CPU_USAGE_ALERT" is in a "X" state; because n metrics meet the trigger rule: "pod_cpu_usage_seconds_total[10m]{pod=~"*kraft* *relay*"}.rate().groupby(namespace,pod).sum()*100>=85 pod_cpu_usage_seconds_total[10m]{pod=~"*aggregation*"}.rate().groupby(namespace,pod).sum()*100>=2*85 pod_cpu_usage_seconds_total[10m]{pod=~"*kafka*"}.rate().groupby(namespace,pod).sum()*100>=6*85", with a trigger delay of 1 minute where, X = FIRING/OK, n = Different services that violated the rule. MQL Expression: pod_cpu_usage_seconds_total[10m]{pod=~"*relay* *kraft*"}.rate().groupby(namespace,pod).sum()*100>={{ CPU Threshold }} pod_cpu_usage_seconds_total[10m]{pod=~"*aggregation*"}.rate().groupby(namespace,pod).sum()*100>=2*{ CPU Threshold } pod_cpu_usage_seconds_total[10m]{pod=~"*kafka*"}.rate().groupby(namespace,pod).sum()*100>=6*{ CPU Threshold }} Note: CPU Threshold will be assigned will executing the terraform script
OID	1.3.6.1.4.1.323.5.3.53.1.29.4002
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 a similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the CPU utilization is below the critical threshold. Note: The thresholds are configurable in the <i>ocnadd-common-custom-values.yaml</i> and <i>ocnadd-relayagent-custom-values.yaml</i> files. If guidance is required, contact My Oracle Support .

OCNADD_POD_MEMORY_USAGE_ALERT

Table 5-5 OCNADD_POD_MEMORY_USAGE_ALERT

Field	Details
Triggering Condition	POD Memory usage is above set threshold (default 90%)
Severity	Major
Description	OCNADD Pod High Memory usage detected for a continuous period of 5min

Table 5-5 (Cont.) OCNADD_POD_MEMORY_USAGE_ALERT

Field	Details
Alert Details CNE	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 }} % ' PromQL Expression: expr: <pre>(sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddbsfaggregation.*"}) by (pod,namespace) > {{ .Values.ocnaddrelayagent.ocnaddaggregation.ocnaddbsfaggregation.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddnrfaggregation.*"}) by (pod,namespace) > {{ .Values.ocnaddrelayagent.ocnaddaggregation.ocnaddnrfaggregation.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddpcfaggregation.*"}) by (pod,namespace) > {{ .Values.ocnaddrelayagent.ocnaddaggregation.ocnaddpcfaggregation.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddscpaggregation.*"}) by (pod,namespace) > {{ .Values.ocnaddrelayagent.ocnaddaggregation.ocnaddscpaggregation.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddseppaggregation.*"}) by (pod,namespace) > {{ .Values.ocnaddrelayagent.ocnaddaggregation.ocnaddseppaggregation.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*kafka-broker.*"}) by (pod,namespace) > {{ .Values.ocnaddrelayagent.ocnaddkafka.ocnadd.kafkaBroker.resource.limit.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*kraft-controller.*"}) by (pod,namespace) > {{ .Values.ocnaddrelayagent.ocnaddkafka.ocnadd.kraftController.resource.limit.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddrelayagentgateway.*"}) by (pod,namespace) > {{ .Values.ocnaddrelayagent.ocnaddrelayagentgateway.ocnaddrelayagentgateway.resources.limits.memory regexFind "[0-9]+" }}</pre>

Table 5-5 (Cont.) OCNADD_POD_MEMORY_USAGE_ALERT

Field	Details
	"[0-9]+" } } *1024*1024*1024*{ .Values.global.cluster.memory_threshold } /100)
Alert Details OCI	Summary: Alarm "OCNADD_POD_MEMORY_USAGE_ALERT" is in a "X" state; because n metrics meet the trigger rule: "container_memory_usage_bytes[5m]{pod=~"*aggregation"*kafka* *kraft* *relay*"},groupby(namespace,pod).sum()/container_spec_memory_limit_bytes[5m]{pod=~"*aggregation"*kafka* *kraft* *relay*"},groupby(namespace,pod).sum()*100>=90", with a trigger delay of 1 minute where, X = FIRING/OK, n = Different services that violated the rule. MQL Expression: container_memory_usage_bytes[10m]{pod=~"*aggregation"*kafka* *kraft* *relay*"},groupby(namespace,pod).sum()/container_spec_memory_limit_bytes[10m]{pod=~"*aggregation"*kafka* *kraft* *relay*"},groupby(namespace,pod).sum()*100>={{ Memory Threshold }}
OID	1.3.6.1.4.1.323.5.3.53.1.29.4005
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 a similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the memory utilization is below the critical threshold. Note: The thresholds are configurable in the <i>ocnadd-common-custom-values.yaml</i> and <i>ocnadd-relayagent-custom-values.yaml</i> files. If guidance is required, contact My Oracle Support .

OCNADD_POD_RESTARTED

Table 5-6 OCNADD_POD_RESTARTED

Field	Details
Triggering Condition	A POD has restarted
Severity	Minor
Description	A POD has restarted in the last 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: A Pod has restarted' PromQL Expression: expr: kube_pod_container_status_restarts_total{namespace="{{ .Values.global.ocnaddrelayagent.cluster.nameSpace.name }}" } > 1
Alert Details OCI	MQL Expression: No MQL equivalent is available
OID	1.3.6.1.4.1.323.5.3.53.1.29.5006

Table 5-6 (Cont.) OCNADD_POD_RESTARTED

Field	Details
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 a similar metric as exposed 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: <code>kubectl get po -n <namespace></code> Note the full name of the pod that is not running, and use it in the following command: <code>kubectl describe pod <desired full pod name> -n <namespace></code> 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. If guidance is required.

Mediation Group Alerts

OCNADD_POD_CPU_USAGE_ALERT

Table 5-7 OCNADD_POD_CPU_USAGE_ALERT

Field	Details
Triggering Condition	POD CPU usage is above the set threshold (default 85%)
Severity	Major
Description	OCNADD Pod High CPU usage detected for a continuous period of 5min

Table 5-7 (Cont.) OCNADD_POD_CPU_USAGE_ALERT

Field	Details
Alert Details CNE	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 }} % ' PromQL Expression: expr: <pre>(sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*ocnaddadminservice.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmediation.oc naddadmin.ocnadd.admin.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*ocnaddfilter.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmediation.oc naddfilter.ocnaddfilter.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*kafka-broker.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmediation.oc naddkafka.ocnadd.kafkaBroker.resource.limit.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*kraft-controller.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmediation.oc naddkafka.ocnadd.kraftController.resource.limit.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image=~".*consumerad apter.*", pod=~".*adapter.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmediation.oc naddadmin.ocnadd.consumeradapter.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*correlation.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmediation.oc naddadmin.ocnadd.correlation.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*storage-adapter.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmediation.oc naddadmin.ocnadd.storageAdapter.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*ingress-adapter.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmediation.oc naddadmin.ocnadd.ingressadapter.resources.limits.cpu }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*ocnaddmediationgateway.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*{{ .Values.ocnaddmediation.oc naddmediationgateway.ocnaddmediationgateway.resources.limits.cpu }})</pre>

Table 5-7 (Cont.) OCNADD_POD_CPU_USAGE_ALERT

Field	Details
Alert Details OCI	Summary: Alarm "OCNADD_POD_CPU_USAGE_ALERT" is in a "X" state; because n metrics meet the trigger rule: "pod_cpu_usage_seconds_total[10m]{pod=~"*admin*"*kraft*"*mediation*".rate().groupby(namespace,pod).sum()*100>=85 pod_cpu_usage_seconds_total[10m]{pod=~"*corr*".rate().groupby(namespace,pod).sum()*100>=3*85 pod_cpu_usage_seconds_total[10m]{pod=~"*kafka*".rate().groupby(namespace,pod).sum()*100>=6*85 pod_cpu_usage_seconds_total[10m]{pod=~"*export*".rate().groupby(namespace,pod).sum()*100>=4*85 pod_cpu_usage_seconds_total[10m]{pod=~"*storageadapter*".rate().groupby(namespace,pod).sum()*100>=3*85 pod_cpu_usage_seconds_total[10m]{pod=~"*ingressadapter*".rate().groupby(namespace,pod).sum()*100>=3*85 pod_cpu_usage_seconds_total[10m]{pod=~"*filter*".rate().groupby(namespace,pod).sum()*100>=2*85", with a trigger delay of 1 minute where, X = FIRING/OK, n = Different services that violated the rule. MQL Expression: pod_cpu_usage_seconds_total[10m]{pod=~"*admin*"*kraft*"*mediation*".rate().groupby(namespace,pod).sum()*100>={{ CPU Threshold }} pod_cpu_usage_seconds_total[10m]{pod=~"*corr*".rate().groupby(namespace,pod).sum()*100>=3*{{ CPU Threshold }} pod_cpu_usage_seconds_total[10m]{pod=~"*kafka*".rate().groupby(namespace,pod).sum()*100>=6*{{ CPU Threshold }} pod_cpu_usage_seconds_total[10m]{pod=~"*export*".rate().groupby(namespace,pod).sum()*100>=4*{{ CPU Threshold }} pod_cpu_usage_seconds_total[10m]{pod=~"*storageadapter*".rate().groupby(namespace,pod).sum()*100>=3*{{ CPU Threshold }} pod_cpu_usage_seconds_total[10m]{pod=~"*ingressadapter*".rate().groupby(namespace,pod).sum()*100>=3*{{ CPU Threshold }} pod_cpu_usage_seconds_total[10m]{pod=~"*filter*".rate().groupby(namespace,pod).sum()*100>=2*{{ CPU Threshold }} Note: CPU Threshold will be assigned will executing the terraform script
OID	1.3.6.1.4.1.323.5.3.53.1.29.4002
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 a similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the CPU utilization is below the critical threshold. Note: The thresholds are configurable in the <i>ocnadd-common-custom-values.yaml</i> and <i>ocnadd-mediation-custom-values.yaml</i> files. If guidance is required, contact My Oracle Support .

OCNADD_POD_MEMORY_USAGE_ALERT

Table 5-8 OCNADD_POD_MEMORY_USAGE_ALERT

Field	Details
Triggering Condition	POD Memory usage is above set threshold (default 90%)
Severity	Major
Description	OCNADD Pod High Memory usage detected for a continuous period of 5min

Table 5-8 (Cont.) OCNADD_POD_MEMORY_USAGE_ALERT

Field	Details
Alert Details CNE	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 }} % ' PromQL Expression: expr: (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddfilter.*"}) by (pod,namespace) > {{ .Values.ocnaddmediation.ocnaddfilter.ocnaddfilter.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*kafka-broker.*"}) by (pod,namespace) > {{ .Values.ocnaddmediation.ocnaddkafka.ocnadd.kafkaBroker.resource.limit.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*kraft-controller.*"}) by (pod,namespace) > {{ .Values.ocnaddmediation.ocnaddkafka.ocnadd.kraftController.resource.limit.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image=~".*consumeradapter.*", pod=~".*adapter.*"}) by (pod,namespace) > {{ .Values.ocnaddmediation.ocnaddadmin.ocnadd.consumeradapter.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*correlation.*"}) by (pod,namespace) > {{ .Values.ocnaddmediation.ocnaddadmin.ocnadd.correlation.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*storage-adapter.*"}) by (pod,namespace) > {{ .Values.ocnaddmediation.ocnaddadmin.ocnadd.storageAdapter.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ingress-adapter.*"}) by (pod,namespace) > {{ .Values.ocnaddmediation.ocnaddadmin.ocnadd.ingressadapter.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnaddmediationgateway.*"}) by (pod,namespace) > {{ .Values.ocnaddmediation.ocnaddmediationgateway.ocnaddmediationgateway.resources.limits.memory regexFind "[0-9]+" }}*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or

Table 5-8 (Cont.) OCNADD_POD_MEMORY_USAGE_ALERT

Field	Details
	"[0-9]+" } } *1024*1024*1024*{{ .Values.global.cluster.memory_threshold } }/100)
Alert Details OCI	Summary: Alarm "OCNADD_POD_MEMORY_USAGE_ALERT" is in a "X" state; because n metrics meet the trigger rule: "container_memory_usage_bytes[5m]{pod=~"*adapter"*kafka"*kraft"*mediation"*corr"*export"*storageadapter"*ingressadapter"*filter*".groupby(namespace,pod).sum()/container_spec_memory_limit_bytes[5m]{pod=~"*adapter"*kafka"*kraft"*mediation"*corr"*export"*storageadapter"*ingressadapter"*filter*".groupby(namespace,pod).sum()*100>=90", with a trigger delay of 1 minute where, X = FIRING/OK, n = Different services that violated the rule. MQL Expression: container_memory_usage_bytes[10m]{pod=~"*adapter"*kafka"*kraft"*mediation"*corr"*export"*storageadapter"*ingressadapter"*filter*".groupby(namespace,pod).sum()/container_spec_memory_limit_bytes[10m]{pod=~"*adapter"*kafka"*kraft"*mediation"*corr"*export"*storageadapter"*ingressadapter"*filter*".groupby(namespace,pod).sum()*100>={{ Memory Threshold }}
OID	1.3.6.1.4.1.323.5.3.53.1.29.4005
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 a similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the memory utilization is below the critical threshold. Note: The thresholds are configurable in the <i>ocnadd-common-custom-values.yaml</i> and <i>ocnadd-mediation-custom-values.yaml</i> files. If guidance is required, contact My Oracle Support .

OCNADD_POD_RESTARTED

Table 5-9 OCNADD_POD_RESTARTED

Field	Details
Triggering Condition	A POD has restarted
Severity	Minor
Description	A POD has restarted in the last 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }} {{ "{{" }} . first value humanizeTimestamp }} {{ "{{" }} end }}: A Pod has restarted' PromQL Expression: expr: kube_pod_container_status_restarts_total{namespace="{{ .Values.global.ocnaddmediation.cluster.nameSpace.name }}" } > 1

Table 5-9 (Cont.) OCNADD_POD_RESTARTED

Field	Details
Alert Details OCI	MQL Expression: No MQL equivalent is available
OID	1.3.6.1.4.1.323.5.3.53.1.29.5006
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 a similar metric as exposed 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: kubectl get po -n <namespace> Note the full name of the pod that is not running, and use it in the following command: kubectl describe pod <desired full pod name> -n <namespace> 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 , If guidance is required.

5.2 Application Level Alerts

This section lists the application level alerts for OCNADD.

Management Group Alerts

OCNADD_CONFIG_SVC_DOWN

Table 5-10 OCNADD_CONFIG_SVC_DOWN

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

Table 5-10 (Cont.) OCNADD_CONFIG_SVC_DOWN

Field	Details
Alert Details OCI	Summary: Alarm "OCNADD_CONFIG_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddconfiguration"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddconfiguration"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.20.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Configuration service starts 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 the "Running" state: kubectl -n <namespace> get pod If it is not in a 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, If guidance is required..

OCNADD_ALARM_SVC_DOWN

Table 5-11 OCNADD_ALARM_SVC_DOWN

Field	Details
Triggering Condition	The alarm service went down or not accessible
Severity	Critical
Description	OCNADD Alarm service not available for more than 2 min

Table 5-11 (Cont.) OCNADD_ALARM_SVC_DOWN

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddalarm service is down' PromQL Expression: expr: up{service="ocnaddalarm"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_ALARM_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddalarm"}.mean()=1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddalarm"}.mean()=1
OID	1.3.6.1.4.1.323.5.3.53.1.24.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Alarm service starts 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 the "Running" state: kubectl -n <namespace> get pod If it is not in a 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, If guidance is required..

OCNADD_HEALTH_MONITORING_SVC_DOWN

Table 5-12 OCNADD_HEALTH_MONITORING_SVC_DOWN

Field	Details
Triggering Condition	The health monitoring service went down or not accessible
Severity	Critical
Description	OCNADD Health monitoring service not available for more than 2 min

Table 5-12 (Cont.) OCNADD_HEALTH_MONITORING_SVC_DOWN

Field	Details
Alert Details CNE	Summary: summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}': ocnaddhealthmonitoring service is down' PromQL Expression: expr: up{service="ocnaddhealthmonitoring"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_HEALTH_MONITORING_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddhealthmonitoring"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddhealthmonitoring"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.28.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Health monitoring service starts 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 the "Running" state: kubect! -n <namespace> get pod If it is not in a running state, capture the pod logs and events. Run the following command to fetch the events as follows: kubect! 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. If guidance is required.

OCNADD_EXPORT_SVC_DOWN

Table 5-13 OCNADD_EXPORT_SVC_DOWN

Field	Details
Triggering Condition	The OCNADD Export service went down or not accessible
Severity	Critical

Table 5-13 (Cont.) OCNADD_EXPORT_SVC_DOWN

Field	Details
Description	OCNADD Export service not available for more than 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Export service is down' PromQL Expression: expr: up{service=~".*export.*"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_EXPORT_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="export"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="export"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.39.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD export service starts 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 the "Running" state: kubectl -n <namespace> get pod If it is not in a 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 . If guidance is required.

OCNADD_MANGEMENT_GW_SVC_DOWN

Table 5-14 OCNADD_MANGEMENT_GW_SVC_DOWN

Field	Details
Triggering Condition	The Management Gateway service went down or not accessible
Severity	Critical

Table 5-14 (Cont.) OCNADD_MANGEMENT_GW_SVC_DOWN

Field	Details
Description	OCNADD Mangement Gateway service not available for more than 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}': ocnaddmanagementgateway service is down' PromQL Expression: expr: up{service="ocnaddmanagementgateway"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_MANGEMENT_GW_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddmanagementgateway"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddmanagementgateway"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.41.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Management Gateway service starts 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 the "Running" state: kubectl -n <namespace> get pod If it is not in a 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, If guidance is required.

RelayAgent Group Alerts

OCNADD_SCP_AGGREGATION_SVC_DOWN

Table 5-15 OCNADD_SCP_AGGREGATION_SVC_DOWN

Field	Details
Triggering Condition	The SCP Aggregation service went down or not accessible
Severity	Critical
Description	OCNADD SCP Aggregation service not available for more than 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddscpaggregation service is down' PromQL Expression: expr: up{service="ocnaddscpaggregation"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_SCP_AGGREGATION_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddscpaggregation"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddscpaggregation"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.22.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD SCP Aggregation service starts 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 the "Running" state: kubectl -n <namespace> get pod If it is not in a 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, If guidance is required.

OCNADD_NRF_AGGREGATION_SVC_DOWN

Table 5-16 OCNADD_NRF_AGGREGATION_SVC_DOWN

Field	Details
Triggering Condition	The NRF Aggregation service went down or not accessible
Severity	Critical
Description	OCNADD NRF Aggregation service not available for more than 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddnrfaggregation service is down' PromQL Expression: expr: up{service="ocnaddnrfaggregation"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_NRF_AGGREGATION_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddnrfaggregation"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddnrfaggregation"}.mean() != 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 a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD NRF Aggregation service starts 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 the "Running" state: kubectrl -n <namespace> get pod If it is not in a running state, capture the pod logs and events. Run the following command to fetch the events as follows: kubectrl 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, If guidance is required.

OCNADD_SEPP_AGGREGATION_SVC_DOWN

Table 5-17 OCNADD_SEPP_AGGREGATION_SVC_DOWN

Field	Details
Triggering Condition	The SEPP Aggregation service went down or not accessible
Severity	Critical
Description	OCNADD SEPP Aggregation service not available for more than 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddseppaggregation service is down' PromQL Expression: expr: up{service="ocnaddseppaggregation"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_SEPP_AGGREGATION_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddseppaggregation"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddseppaggregation"}.mean() != 1
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 a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD SEPP Aggregation service starts 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 the "Running" state: kubectl -n <namespace> get pod If it is not in a 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, If guidance is required.

OCNADD_BSF_AGGREGATION_SVC_DOWN

Table 5-18 OCNADD_BSF_AGGREGATION_SVC_DOWN

Field	Details
Triggering Condition	The BSF Aggregation service went down or not accessible
Severity	Critical
Description	OCNADD BSF Aggregation service not available for more than 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddbsfaggregation service is down' PromQL Expression: expr: up{service="ocnaddbsfaggregation"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_BSF_AGGREGATION_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddbsfaggregation"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddbsfaggregation"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.40.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD BSF Aggregation service starts 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 the "Running" state: kubectl -n <namespace> get pod If it is not in a 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, If guidance is required.

OCNADD_PCF_AGGREGATION_SVC_DOWN

Table 5-19 OCNADD_PCF_AGGREGATION_SVC_DOWN

Field	Details
Triggering Condition	The PCF Aggregation service went down or not accessible
Severity	Critical
Description	OCNADD PCF Aggregation service not available for more than 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddpcfaggregation service is down' PromQL Expression: expr: up{service="ocnaddpcfaggregation"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_PCF_AGGREGATION_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddpcfaggregation"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddpcfaggregation"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.41.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD PCF Aggregation service starts 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 the "Running" state: kubectl -n <namespace> get pod If it is not in a 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, If guidance is required.

OCNADD_RELAYAGENT_GW_SVC_DOWN

Table 5-20 OCNADD_RELAYAGENT_GW_SVC_DOWN

Field	Details
Triggering Condition	The RelayAgent Gateway service went down or not accessible
Severity	Critical
Description	OCNADD RelayAgent Gateway service not available for more than 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddrelayagentgateway service is down' PromQL Expression: expr: up{service="ocnaddrelayagentgateway"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_RELAYAGENT_GW_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddrelayagentgateway"}.mean()=1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddrelayagentgateway"}.mean()=1
OID	1.3.6.1.4.1.323.5.3.53.1.41.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD RelayAgent Gateway service starts 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 the "Running" state: kubect! -n <namespace> get pod If it is not in a running state, capture the pod logs and events. Run the following command to fetch the events as follows: kubect! 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. If guidance is required.

OCNADD_MPS_WARNING_INGRESS_THRESHOLD_CROSSED

Table 5-21 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 the configured warning threshold (80%) for the period of 5 min
Alert Details CNE	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 }}' PromQL Expression: expr: sum(irate(kafka_stream_processor_node_process_total{service=~".*aggregation.*"}[5m])) by (namespace) > 0.8*{{ .Values.global.cluster.mps }}
Alert Details OCI	Summary: Alarm "OCNADD_MPS_WARNING_INGRESS_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "kafka_stream_processor_node_process_total[10m]{microservice=~"*aggregation*"}.rate().groupBy(k8Namespace).sum()>.8*{{ MPS Threshold }}" , with a trigger delay of 1 minute MQL Expression: kafka_stream_processor_node_process_total[10m]{microservice=~"*aggregation*"}.rate().groupBy(k8Namespace).sum()>.8*{{ MPS Threshold }}
OID	1.3.6.1.4.1.323.5.3.53.1.29.5007
Metric Used	kafka_stream_processor_node_process_total
Resolution	The alert is cleared automatically when the MPS rate goes below the warning threshold level of 80%.

OCNADD_MPS_MINOR_INGRESS_THRESHOLD_CROSSED**Table 5-22 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

Table 5-22 (Cont.) OCNADD_MPS_MINOR_INGRESS_THRESHOLD_CROSSED

Field	Details
Alert Details CNE	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 }}' PromQL Expression: expr: sum(irate(kafka_stream_processor_node_process_total{service=~".*aggregation.*"}[5m])) by (namespace) > 0.9*{{ .Values.global.cluster.mps }}
Alert Details OCI	Summary: Alarm "OCNADD_MPS_MINOR_INGRESS_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "kafka_stream_processor_node_process_total[10m]{microservice=~"*aggregation*"}.rate().groupBy(k8Namespace).sum()>.9*{{ MPS Threshold }}" , with a trigger delay of 1 minute MQL Expression: kafka_stream_processor_node_process_total[10m]{microservice=~"*aggregation*"}.rate().groupBy(k8Namespace).sum()>.9*{{ MPS Threshold }}
OID	1.3.6.1.4.1.323.5.3.53.1.29.5008
Metric Used	kafka_stream_processor_node_process_total
Resolution	The alert is cleared automatically when the MPS rate goes below the minor threshold alert level of 90%.

OCNADD_MPS_MAJOR_INGRESS_THRESHOLD_CROSSED

Table 5-23 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 the configured major threshold alert (95%) for the period of 5 min
Alert Details CNE	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 }}' PromQL Expression: expr: sum(irate(kafka_stream_processor_node_process_total{service=~".*aggregation.*"}[5m])) by (namespace) > 0.95*{{ .Values.global.cluster.mps }}

Table 5-23 (Cont.) OCNADD_MPS_MAJOR_INGRESS_THRESHOLD_CROSSED

Field	Details
Alert Details OCI	Summary: Alarm "OCNADD_MPS_MAJOR_INGRESS_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "kafka_stream_processor_node_process_total[10m]{microservice=~"*aggregation*".rate().groupBy(k8Namespace).sum()}>0.95*{{ MPS Threshold }}" , with a trigger delay of 1 minute MQL Expression: kafka_stream_processor_node_process_total[10m]{microservice=~"*aggregation*".rate().groupBy(k8Namespace).sum()}>0.95*{{ MPS Threshold }}
OID	1.3.6.1.4.1.323.5.3.53.1.29.5009
Metric Used	kafka_stream_processor_node_process_total
Resolution	The alert is cleared automatically when the MPS rate goes below the major threshold alert level of 95%.

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
Alert Details CNE	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 }}' PromQL Expression: expr: sum(irate(kafka_stream_processor_node_process_total{service=~".*aggregation.*"}[5m])) by (namespace) > 1.0*{{ .Values.global.cluster.mps }}
Alert Details OCI	Summary: Alarm "OCNADD_MPS_CRITICAL_INGRESS_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "kafka_stream_processor_node_process_total[10m]{microservice=~"*aggregation*".rate().groupBy(k8Namespace).sum()}>1.0*{{ MPS Threshold }}" , with a trigger delay of 1 minute MQL Expression: kafka_stream_processor_node_process_total[10m]{microservice=~"*aggregation*".rate().groupBy(k8Namespace).sum()}>1.0*{{ MPS Threshold }}
OID	1.3.6.1.4.1.323.5.3.53.1.29.5010
Metric Used	kafka_stream_processor_node_process_total

Table 5-24 (Cont.) OCNADD_MPS_CRITICAL_INGRESS_THRESHOLD_CROSSED

Field	Details
Resolution	The alert is cleared automatically when the MPS rate goes below the critical threshold alert level of 100% of supported MPS

OCNADD_KAFKA_PACKET_DROP_THRESHOLD_1PERCENT_MPS

Table 5-25 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 the 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
Alert Details CNE	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 }}' PromQL Expression: expr: sum(rate(kafka_stream_task_dropped_records_total{service=~".*aggregation.*"}[5m])) by (namespace) > 0.01*{{ .Values.global.cluster.mps }}
Alert Details OCI	Summary: Alarm "OCNADD_KAFKA_PACKET_DROP_THRESHOLD_1PERCENT_MPS" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "kafka_stream_task_dropped_records_total[10m]{microservice=~"*aggregation"}.rate().groupBy(k8Namespace).sum()*100 > {{ MPS Threshold }}", with a trigger delay of 1 minute MQL Expression: kafka_stream_task_dropped_records_total[10m]{microservice=~"*aggregation"}.rate().groupBy(k8Namespace).sum()*100 > {{ MPS Threshold }}
OID	1.3.6.1.4.1.323.5.3.53.1.29.5020
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_KAFKA_PACKET_DROP_THRESHOLD_10PERCENT_MPS

Table 5-26 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 the total supported MPS
Severity	Critical

Table 5-26 (Cont.) OCNADD_KAFKA_PACKET_DROP_THRESHOLD_10PERCENT_MPS

Field	Details
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 CNE	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 }}' PromQL Expression: expr: sum(rate(kafka_stream_task_dropped_records_total{service=~".*aggregation.*"}[5m])) by (namespace) > 0.1*{{ .Values.global.cluster.mps }}
Alert Details OCI	Summary: Alarm "OCNADD_KAFKA_PACKET_DROP_THRESHOLD_10PERCENT_MPS" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "kafka_stream_task_dropped_records_total[10m]{microservice=~"*aggregation"}.rate().groupBy(k8Namespace).sum()*100>10*{{MPS Threshold }}", with a trigger delay of 1 minute MQL Expression: kafka_stream_task_dropped_records_total[10m]{microservice=~"*aggregation"}.rate().groupBy(k8Namespace).sum()*100>10*{{MPS Threshold }}
OID	1.3.6.1.4.1.323.5.3.53.1.29.5021
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_INGRESS_TRAFFIC_RATE_INCREASE_SPIKE_10PERCENT

Table 5-27 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 the last 5 min
Alert Details CNE	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' PromQL 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
Alert Details OCI	Not Available

Table 5-27 (Cont.)

OCNADD_INGRESS_TRAFFIC_RATE_INCREASE_SPIKE_10PERCENT

Field	Details
OID	1.3.6.1.4.1.323.5.3.53.1.29.5027
Metric Used	kafka_stream_processor_node_process_total
Resolution	The alert is cleared automatically when the increase in MPS comes back to lower than 10% of the supported MPS

OCNADD_INGRESS_TRAFFIC_RATE_DECREASE_SPIKE_10PERCENT

Table 5-28 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 the last 5 min
Alert Details CNE	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' PromQL 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
Alert Details OCI	Not Available
OID	1.3.6.1.4.1.323.5.3.53.1.29.5028
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

Mediation Group Alerts

OCNADD_ADMIN_SVC_DOWN

Table 5-29 OCNADD_ADMIN_SVC_DOWN

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

Table 5-29 (Cont.) OCNADD_ADMIN_SVC_DOWN

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: ocnaddadminservice service is down' PromQL Expression: expr: up{service="ocnaddadminservice"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_ADMIN_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddadminservice"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddadminservice"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.30.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Admin service starts 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 the "Running" state: kubect! -n <namespace> get pod If it is not in a running state, capture the pod logs and events. Run the following command to fetch the events as follows: kubect! 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 , If guidance is required.

OCNADD_CONSUMER_ADAPTER_SVC_DOWN

Table 5-30 OCNADD_CONSUMER_ADAPTER_SVC_DOWN

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

Table 5-30 (Cont.) OCNADD_CONSUMER_ADAPTER_SVC_DOWN

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Consumer Adapter service is down' PromQL Expression: expr: up{service=~".*adapter.*"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_CONSUMER_ADAPTER_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="correlation"}.mean()=1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner=~"adapter*"} .mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.25.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a 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: 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, If guidance is required.

OCNADD_FILTER_SVC_DOWN

Table 5-31 OCNADD_FILTER_SVC_DOWN

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

Table 5-31 (Cont.) OCNADD_FILTER_SVC_DOWN

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Filter service is down' PromQL Expression: expr: up{service=~".*filter.*"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_FILTER_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddfilter"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddfilter"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.34.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Filter service starts 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 the "Running" state: <pre>kubectl -n <namespace> get pod</pre> If it is not in a 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. If guidance is required.

OCNADD_CORRELATION_SVC_DOWN

Table 5-32 OCNADD_CORRELATION_SVC_DOWN

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

Table 5-32 (Cont.) OCNADD_CORRELATION_SVC_DOWN

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Correlation service is down' PromQL Expression: expr: up{service=~".*correlation.*"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_CORRELATION_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="correlation"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="correlation"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.33.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Correlation service starts 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 the "Running" state: kubect! -n <namespace> get pod If it is not in a running state, capture the pod logs and events. Run the following command to fetch the events as follows: kubect! 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 . If guidance is required.

OCNADD_MEDIATION_GW_SVC_DOWN

Table 5-33 OCNADD_MEDIATION_GW_SVC_DOWN

Field	Details
Triggering Condition	The Mediation Gateway service went down or not accessible
Severity	Critical
Description	OCNADD Mediation Gateway service not available for more than 2 min

Table 5-33 (Cont.) OCNADD_MEDIATION_GW_SVC_DOWN

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}': ocnaddmediationgateway service is down' PromQL Expression: expr: up{service="ocnaddmediationgateway"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_MEDIATION_GW_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ocnaddmediationgateway"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ocnaddmediationgateway"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.41.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Mediation Gateway service starts 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 the "Running" state: kubect! -n <namespace> get pod If it is not in a running state, capture the pod logs and events. Run the following command to fetch the events as follows: kubect! 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. If guidance is required.

OCNADD_STORAGE_ADAPTER_SVC_DOWN

Table 5-34 OCNADD_STORAGE_ADAPTER_SVC_DOWN

Field	Details
Triggering Condition	The OCNADD Storage adapter service went down or not accessible
Severity	Critical

Table 5-34 (Cont.) OCNADD_STORAGE_ADAPTER_SVC_DOWN

Field	Details
Description	OCNADD Storage adapter service not available for more than 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Storage adapter service is down' PromQL Expression: expr: up{service=~".*storageadapter.*"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_STORAGE_ADAPTER_SVC_DOWN" is in a "OK/ FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="storageadapter"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="storageadapter"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.38.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Storage adapter service starts 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 the "Running" state: kubect! -n <namespace> get pod If it is not in a running state, capture the pod logs and events. Run the following command to fetch the events as follows: kubect! 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 , If guidance is required.

OCNADD_INGRESS_ADAPTER_SVC_DOWN

Table 5-35 OCNADD_INGRESS_ADAPTER_SVC_DOWN

Field	Details
Triggering Condition	The OCNADD Ingress Adapter service went down or not accessible
Severity	Critical

Table 5-35 (Cont.) OCNADD_INGRESS_ADAPTER_SVC_DOWN

Field	Details
Description	OCNADD Ingress Adapter service not available for more than 2 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Ingress Adapter service is down' PromQL Expression: expr: up{service=~".*ingressadapter.*"} != 1
Alert Details OCI	Summary: Alarm "OCNADD_INGRESS_ADAPTER_SVC_DOWN" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "podStatus[10m]{podOwner="ingressadapter"}.mean() != 1", with a trigger delay of 1 minute MQL Expression: podStatus[10m]{podOwner="ingressadapter"}.mean() != 1
OID	1.3.6.1.4.1.323.5.3.53.1.36.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Ingress Adapter service starts 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 the "Running" state: kubect! -n <namespace> get pod If it is not in a running state, capture the pod logs and events. Run the following command to fetch the events as follows: kubect! 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 , If guidance is required.

OCNADD_MPS_CRITICAL_INGRESS_ADAPTER_THRESHOLD_CROSSED

Table 5-36 OCNADD_MPS_CRITICAL_INGRESS_ADAPTER_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total ingress MPS crossed for ingress adapter the critical threshold alert level of 100% of the supported MPS

Table 5-36 (Cont.)

OCNADD_MPS_CRITICAL_INGRESS_ADAPTER_THRESHOLD_CROSSED

Field	Details
Severity	Critical
Description	Total Ingress Message Rate for ingress adapter is above configured critical threshold alert (100%) for the period of 5 min
Alert Details CNE	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 }}' PromQL Expression: expr: sum(irate(kafka_stream_processor_node_process_total{service=~".*ingress-adapter.*"}[5m]) by (namespace) > 1.0*{{ .Values.global.cluster.mps }}
Alert Details OCI	Summary: Alarm "OCNADD_MPS_CRITICAL_INGRESS_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "kafka_stream_processor_node_process_total[10m]{microservice=~"*aggregation*"}.rate().groupBy(k8Namespace).sum()>1.0*{{ MPS Threshold }}", with a trigger delay of 1 minute MQL Expression: kafka_stream_processor_node_process_total[10m]{microservice=~"*ingress-adapter*"}.rate().groupBy(k8Namespace).sum()>1.0*{{ MPS Threshold }}
OID	1.3.6.1.4.1.323.5.3.53.1.36.5010
Metric Used	kafka_stream_processor_node_process_total
Resolution	The alert is cleared automatically when the MPS rate goes below the critical threshold alert level of 100% of supported MPS

OCNADD_MPS_WARNING_EGRESS_THRESHOLD_CROSSED

Table 5-37 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	The total Egress Message Rate is above the configured warning threshold alert (80%) for the period of 5 min

Table 5-37 (Cont.) OCNADD_MPS_WARNING_EGRESS_THRESHOLD_CROSSED

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }} \$labels.worker_group }}, 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 }}' PromQL Expression: expr: sum (irate(ocnadd_egress_requests_total[5m])) by (namespace) > 0.80*{{ .Values.global.cluster.mps }}
Alert Details OCI	Summary: Alarm "OCNADD_MPS_WARNING_EGRESS_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_requests_total[10m] {app=~"*adapter"}.rate().groupBy(worker_group,app).sum()>0.80*{{ MP S Threshold }}"", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_requests_total[10m] {app=~"*adapter"}.rate().groupBy(worker_group,app).sum()>0.80*{{ MP S Threshold }}"
OID	1.3.6.1.4.1.323.5.3.53.1.29.5011
Metric Used	ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the MPS rate goes below the warning threshold alert level of 80% of supported MPS

OCNADD_MPS_MINOR_EGRESS_THRESHOLD_CROSSED

Table 5-38 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	The total Egress Message Rate is above the configured minor threshold alert (90%) for the period of 5 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }} \$labels.worker_group }}, 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 }}' PromQL Expression: expr: sum (irate(ocnadd_egress_requests_total[5m])) by (namespace) > 0.90*{{ .Values.global.cluster.mps }}

Table 5-38 (Cont.) OCNADD_MPS_MINOR_EGRESS_THRESHOLD_CROSSED

Field	Details
Alert Details OCI	Summary: Alarm "OCNADD_MPS_MINOR_EGRESS_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_requests_total[10m] {app=~"*adapter"}.rate().groupBy(worker_group,app).sum()>0.90*{{ MPS Threshold }}"", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_requests_total[10m] {app=~"*adapter"}.rate().groupBy(worker_group,app).sum()>0.90*{{ MPS Threshold }}
OID	1.3.6.1.4.1.323.5.3.53.1.29.5012
Metric Used	ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the MPS rate goes below the minor threshold alert level of 90% of supported MPS

OCNADD_MPS_MAJOR_EGRESS_THRESHOLD_CROSSED

Table 5-39 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	The total Egress Message Rate is above the configured major threshold alert (95%) for the period of 5 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, worker_group: {{ "{{" }}\$labels.worker_group}}, 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 }}
Alert Details OCI	Summary: Alarm "OCNADD_MPS_MAJOR_EGRESS_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_requests_total[10m] {app=~"*adapter"}.rate().groupBy(worker_group,app).sum()>0.95*{{ MPS Threshold }}"", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_requests_total[10m] {app=~"*adapter"}.rate().groupBy(worker_group,app).sum()>0.95*{{ MPS Threshold }}"
OID	1.3.6.1.4.1.323.5.3.53.1.29.5013
Metric Used	ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the MPS rate goes below the major threshold alert level of 95% of supported MPS

OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED**Table 5-40 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	The total Egress Message Rate is above the configured critical threshold alert (100%) for the period of 5 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }} \$labels.worker_group }}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above supported Max messages per second:{{ .Values.global.cluster.mps }}' PromQL Expression: expr: sum (irate(ocnadd_egress_requests_total[5m])) by (namespace) > 1.0*{{ .Values.global.cluster.mps }}
Alert Details OCI	Summary: Alarm "OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_requests_total[10m] {app=~"*adapter"}.rate().groupBy(worker_group,app).sum()>1.0*{{ MPS Threshold }}"", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_requests_total[10m] {app=~"*adapter"}.rate().groupBy(worker_group,app).sum()>1.0*{{ MPS Threshold }}"
OID	1.3.6.1.4.1.323.5.3.53.1.29.5014
Metric Used	ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when 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 5-41 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	The total Egress Message Rate is above the configured critical threshold alert (100%) for the period of 5 min

Table 5-41 (Cont.)

OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED_FOR_A_CONSUMER

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }} \$labels.worker_group }}, 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 }}
Alert Details OCI	Summary: Alarm "OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED_FOR_A_CONSUMER" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_requests_total[10m] {app=~"*adapter"}.rate().groupBy(worker_group,app).sum()>1.0 {{ MPS Threshold }}", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_requests_total[10m] {app=~"*adapter"}.rate().groupBy(worker_group,app).sum()>1.0 {{ MPS Threshold }}
OID	1.3.6.1.4.1.323.5.3.53.1.29.5015
Metric Used	ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the MPS rate goes below the critical threshold alert level of 100% of supported MPS

OCNADD_E2E_AVG_RECORD_LATENCY_WARNING_THRESHOLD_CROSSED

Table 5-42 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 the configured warning threshold alert level (80%) for the period of 5 min

Table 5-42 (Cont.)

OCNADD_E2E_AVG_RECORD_LATENCY_WARNING_THRESHOLD_CROSSED

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }} \$labels.worker_group}}, 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 }}
Alert Details OCI	Summary: Alarm "OCNADD_E2E_AVG_RECORD_LATENCY_WARNING_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum())>0.040&&ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum())<=0.045", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum())>0.040&&ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum())<=0.045 Note: {{ .Values.global.cluster.max_latency }} is considered to be 0.05
OID	1.3.6.1.4.1.323.5.3.53.1.29.5016
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_E2E_AVG_RECORD_LATENCY_MINOR_THRESHOLD_CROSSED

Table 5-43 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 the configured minor threshold alert level (90%) for the period of 5 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }}\$labels.worker_group }, 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' PromQL 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)) > .90*{{ .Values.global.cluster.max_latency }} <= 0.95*{{ .Values.global.cluster.max_latency }}
Alert Details OCI	Summary: Alarm "OCNADD_E2E_AVG_RECORD_LATENCY_MINOR_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum())>0.045&&ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum())<=0.0475", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum())>0.045&&ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum())<=0.0475 Note: {{ .Values.global.cluster.max_latency }} is considered to be 0.05
OID	1.3.6.1.4.1.323.5.3.53.1.29.5017
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_MAJOR_THRESHOLD_CROSSED

Table 5-44 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 the configured minor threshold alert level (95%) for the period of 5 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }}\$labels.worker_group }, 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' PromQL 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)) > .95*{{ .Values.global.cluster.max_latency }} <= 1.0*{{ .Values.global.cluster.max_latency }}
Alert Details OCI	Summary: Alarm "OCNADD_E2E_AVG_RECORD_LATENCY_MAJOR_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum())>0.0475&&ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum())<=0.05", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum())>0.0475&&ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum())<=0.05 Note: {{ .Values.global.cluster.max_latency }} is considered to be 0.05
OID	1.3.6.1.4.1.323.5.3.53.1.29.5018
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_CRITICAL_THRESHOLD_CROSSED

Table 5-45 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 the configured critical threshold alert level (100%) for the period of 5 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }} \$labels.worker_group }}, 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' PromQL 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)) > 1.0*{{ .Values.global.cluster.max_latency }}
Alert Details OCI	Summary: Alarm "OCNADD_E2E_AVG_RECORD_LATENCY_CRITICAL_THRESHOLD_CROSSED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum()/>0.05", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_e2e_request_processing_latency_seconds_sum[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_e2e_request_processing_latency_seconds_count[10m].rate().groupBy(worker_group,app).sum()/>0.05 Note: {{ .Values.global.cluster.max_latency }} is considered to be 0.05
OID	1.3.6.1.4.1.323.5.3.53.1.29.5019
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_EGRESS_FAILURE_RATE_THRESHOLD_0.1PERCENT**Table 5-46 OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_0.1PERCENT**

Field	Details
Triggering Condition	The Egress adapter failure rate towards the 3rd party application is above the configured threshold of 0.1% of the total supported MPS
Severity	Info

Table 5-46 (Cont.) OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_0.1PERCENT

Field	Details
Description	Egress external connection failure rate towards 3rd party application is crossing the info threshold of 0.1% in the period of 5 min
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }} \$labels.worker_group }}, 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' PromQL 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
Alert Details OCI	Summary: Alarm "OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_01PERCENT" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100>=0.1&&ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100<1", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100>=0.1&&ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100<1
OID	1.3.6.1.4.1.323.5.3.53.1.29.5022
Metric Used	ocnadd_egress_failed_request_total, ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the failure rate towards 3rd party consumers goes below the threshold (0.1%) alert level of supported MPS

OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_1PERCENT

Table 5-47 OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_1PERCENT

Field	Details
Triggering Condition	The Egress adapter failure rate towards the 3rd party application is above the configured threshold of 1% of the total supported MPS
Severity	Warn
Description	Egress external connection failure rate towards 3rd party application is crossing the warning threshold of 1% in the period of 5 min

Table 5-47 (Cont.) OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_1PERCENT

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }} \$labels.worker_group }}, 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' PromQL Expression: expr: (sum(rate(ocnadd_egress_failed_request_total[5m])) by (namespace))/(sum(rate(ocnadd_egress_requests_total[5m])) by (namespace)) *100 >= 1 < 10
Alert Details OCI	Summary: Alarm "OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_1PERCENT" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100>=1&&ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100<10", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100>=1&&ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100<10
OID	1.3.6.1.4.1.323.5.3.53.1.29.5023
Metric Used	ocnadd_egress_failed_request_total, ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the failure rate towards 3rd party consumers goes below the threshold (1%) alert level of supported MPS

OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_10PERCENT

Table 5-48 OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_10PERCENT

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

Table 5-48 (Cont.) OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_10PERCENT

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }} \$labels.worker_group }}, 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' PromQL Expression: expr: (sum(rate(ocnadd_egress_failed_request_total[5m])) by (namespace))/(sum(rate(ocnadd_egress_requests_total[5m])) by (namespace)) *100 >= 10 < 25
Alert Details OCI	Summary: Alarm "OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_10PERCENT" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100>=10&&ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100<25", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100>=10&&ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100<25
OID	1.3.6.1.4.1.323.5.3.53.1.29.5024
Metric Used	ocnadd_egress_failed_request_total, ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the failure rate towards 3rd party consumers goes below the threshold (10%) alert level of supported MPS

OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_25PERCENT

Table 5-49 OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_25PERCENT

Field	Details
Triggering Condition	The Egress adapter failure rate towards the 3rd party application is above the configured threshold of 25% of the total supported MPS
Severity	Major
Description	Egress external connection failure rate towards 3rd party application is crossing the major threshold of 25% in the period of 5 min

Table 5-49 (Cont.) OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_25PERCENT

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }} \$labels.worker_group }}, 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' PromQL Expression: expr: (sum(rate(ocnadd_egress_failed_request_total[5m])) by (namespace))/(sum(rate(ocnadd_egress_requests_total[5m])) by (namespace)) *100 >= 25 < 50
Alert Details OCI	Summary: Alarm "OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_25PERCENT" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100>=25&&ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100<50", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100>=25&&ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100<50
OID	1.3.6.1.4.1.323.5.3.53.1.29.5025
Metric Used	ocnadd_egress_failed_request_total, ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the failure rate towards 3rd party consumers goes below the threshold (25%) alert level of supported MPS

OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_50PERCENT

Table 5-50 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 the total supported MPS
Severity	Critical
Description	Egress external connection failure rate towards 3rd party application is crossing the critical threshold of 50% in the period of 5 min

Table 5-50 (Cont.) OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_50PERCENT

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }} \$labels.worker_group }, 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' PromQL Expression: expr:(sum(rate(ocnadd_egress_failed_request_total[5m])) by (namespace))/(sum(rate(ocnadd_egress_requests_total[5m])) by (namespace)) *100 >= 50
Alert Details OCI	Summary: Alarm "OCNADD_EGRESS_FAILURE_RATE_THRESHOLD_50PERCENT" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100>=50", with a trigger delay of 1 minute MQL Expression: ocnadd_egress_failed_request_total[10m].rate().groupBy(worker_group,app).sum()/ocnadd_egress_requests_total[10m].rate().groupBy(worker_group,app).sum()*100>=50
OID	1.3.6.1.4.1.323.5.3.53.1.29.5026
Metric Used	ocnadd_egress_failed_request_total, ocnadd_egress_requests_total
Resolution	The alert is cleared automatically when the failure rate towards 3rd party consumers goes below the threshold (50%) alert level of supported MPS

OCNADD_TRANSACTION_SUCCESS_CRITICAL_THRESHOLD_DROPPED

Table 5-51 OCNADD_TRANSACTION_SUCCESS_CRITICAL_THRESHOLD_DROPPED

Field	Details
Triggering Condition	The total transaction success xDRs rate has dropped the critical threshold alert level of 90%
Severity	Critical
Description	The total transaction success xDRs rate has dropped the critical threshold alert level of 90% for the period of 5min

Table 5-51 (Cont.)
OCNADD_TRANSACTION_SUCCESS_CRITICAL_THRESHOLD_DROPPED

Field	Details
Alert Details CNE	Summary: 'namespace: {{ "{{" }}\$labels.namespace}}, workergroup: {{ "{{" }} \$labels.worker_group }}, podname: {{ "{{" }}\$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . first value humanizeTimestamp }}{{ "{{" }} end }}: Transaction Success Rate is below 90% per hour:{{ .Values.global.cluster.mps }}' Expression: expr: sum(irate(ocnadd_total_transactions_total{service=~".*correlation.*",status="SUCCESS"}[5m]))by (namespace,service) / sum(irate(ocnadd_total_transactions_total{service=~".*correlation.*"}[5m]))by (namespace,service) *100 < 90
Alert Details OCI	Summary: Alarm "OCNADD_TRANSACTION_SUCCESS_CRITICAL_THRESHOLD_DROPPED" is in a "OK/FIRING" state; because 0/1 metrics meet the trigger rule: "ocnadd_total_transactions_total[10m]{status="SUCCESS",app=~"*corr*"}.rate().groupBy(workername,app).sum()/ocnadd_total_transactions_total[10m]{app=~"*corr*"}.rate().groupBy(workername,app).sum()*100<90", with a trigger delay of 1 minute MQL Expression: ocnadd_total_transactions_total[10m]{status="SUCCESS",app=~"*corr*"}.rate().groupBy(workername,app).sum()/ocnadd_total_transactions_total[10m]{app=~"*corr*"}.rate().groupBy(workername,app).sum()*100<90
OID	1.3.6.1.4.1.323.5.3.53.1.33.5029
Metric Used	ocnadd_total_transactions_total
Resolution	The alert is cleared automatically when the transaction success rate goes above the critical threshold alert level of 90%