Oracle® Communications

Network Analytics Data Director Troubleshooting Guide

Release 22.0.0
F72955-01
December 2022

Oracle Communications Network Analytics Data Director Troubleshooting Guide, Release 22.0.0

F72955-01

Contents

My Oracle Support (MOS)

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

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

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

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

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

Acronyms

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

Table Acronyms

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

What's New in This Guide

This section lists the documentation updates for Release *22.0.0* in *Oracle Communications Network Analytics Data Director Troubleshooting Guide*.

Release 22.0.0 - F72955-01, December 2022

This is the first release of the document.

1 Introduction

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

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 feeds.

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

Audience

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

Reference

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

- *Oracle Communications Network Analytics Data Director User Guide*
- *Oracle Communications Network Analytics Data Director Installation Guide*
- *Oracle Communications Network Analytics Data Director Disaster Recovery Guide*
- *Oracle Communications Cloud Native Environment Installation Guide*
- *Oracle Communications Cloud Native Core DBTier Disaster Recovery Guide*
- *Oracle Communications Cloud Native Core Data Collector Guide*
- *Oracle Communications Cloud Native Core Console Installation and Upgrade Guide*
- *Oracle Communications Cloud Native Core Console Troubleshooting Guide*

2
Troubleshooting OCNADD

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

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

Generic Checklist

The following sections provide a generic checklists for troubleshooting OCNADD.

Deployment Checklist

Perform the following pre-deployment checks:

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

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

  Verify the output, and check the following columns:

  – READY, STATUS, and RESTARTS

  – PORT(S) of service

  > **Note:**
  >
  > It is normal to observe the Kafka broker restart during deployment.

- Verify if the correct image used and correct environment variables set in the deployment. To check, run the following command:

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

- Check if the microservices can access each other through a REST interface. To check, run the following command:

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

Example:

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

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

The list of URIs for all the microservices:

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

Application Checklist

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

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

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

Example:

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

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

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

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

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

Helm Install and Upgrade Failure

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

Incorrect Image Name in ocnadd/values File

Problem

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

Error Code or Error Message

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

Solution

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

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

Docker Registry is Configured Incorrectly

Problem

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

Error Code or Error Message

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

Solution

Configure the docker registry on all primary and secondary nodes. For information about docker registry configuration, see *Oracle Communications Cloud Native Environment Installation Guide*.

Continuous Restart of Pods

Problem

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

Error Code or Error Message

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

Solution

1. Verify MySQL connectivity.

   MySQL servers(s) may not be configured properly. For more information about the MySQL configuration, see *Oracle Communications Network Analytics Data Director Installation Guide*.
2. Describe the POD to check more details on the error, troubleshoot further based on the reported error.
3. Check the POD log for any error, troubleshoot further based on the reported error.

Adapter Pods Does Not Receive Update Notification

Problem

Update notification from configuration service/GUI is not reaching Adapter pods..

Error Code or Error Message

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

Solution

This can be fixed by running the following command:

- Check the Adapter pod logs with:

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

- If the update notification logs are not present in the Adapter logs execute the below command to re-read the configurations with the following command:

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

Values.yaml File Parse Failure

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

Problem

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

Error Code or Error Message

Error: failed to parse `ocnadd/values.yaml`: error converting YAML to JSON: yaml

Symptom

When parsing the `ocnadd/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 Guide*.

Kafka Brokers Continuously Restart after Reinstallation

Problem

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

Error Code or Error Message

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

Solution

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

For example:

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

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

Kafka Brokers Continuously Restart After the Disk is Full

Problem

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

Error Code or Error Message

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

Solution

1. Delete the STS(Stateful set) deployments of the brokers:
   a. Get the STS's in the namespace with the following command:

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

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

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

      For example, for three broker setup:

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

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

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

   The number of PVCs used will be based on the number of brokers you deploy. So choose the PVCs that have the name kafka-broker or zookeeper and delete them.
   a. To delete PVCs, run:

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

      For example, For a three broker setup in namespace ocnadd-deploy, you will need to delete these PVCs;

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

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

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

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

   To increase the storage edit the fields pvcClaimSize for each broker.
4. 4. Upgrade the helm chart after increasing the PV size

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

5. Create the required topics.

Kafka Brokers Restart on Installation

Problem

Kafka brokers re-start during OCNADD installation.

Error Code or Error Message

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

Solution

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

Database Goes into the Deadlock State

Problem

MySQL locks get struck.

Error Code or Error Message

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

Symptom

Unable to access MySQL.

Solution

Perform the following steps to remove the deadlock:

1. Run the following command on each SQL node:

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

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

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

```
+--------------------------+
3 rows in set (0.00 sec)
```

2. Run the kill command on each SQL node.

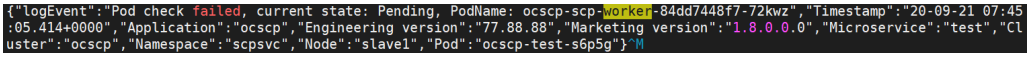
Readiness Probe Failure

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

If the following error appears, check for the readiness probe URL correctness in the particular microservice helm charts under the charts folder:

Low Resources

`Helm install` might fail due to low resources, and the following error may appear:



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

3
Logs

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

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.

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.

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-xxxxxxxxxx-xxxxx -n ocnadd
   ```

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

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

   Example:

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

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

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>:/home/ocnadd-tiger/filename.pcap
   ```

5
Alerts

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

Configuring Alerts

This section describes how to configure alerts in OCNADD.

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

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

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

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

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

List of Alerts

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

System Level Alerts

This section lists the system level alerts for OCNADD.

OCNADD_POD_CPU_USAGE_ALERT

Table 5-1 OCNADD_POD_CPU_USAGE_ALERT

Field	Details
Triggering Condition	POD CPU usage is above set threshold (default 70%)
Severity	Major
Description	OCNADD Pod High CPU usage detected for the continuous period of 5min
Alert Details	**Summary:** 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: CPU usage is {{ "{{" }} $value | printf "%.2f" }} which is above threshold {{ .Values.global.cluster.cpu_threshold }} % ' **Expression**: expr: (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*ocnadd.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*2) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*kafka.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*2) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*zookeeper.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*egw.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*2) or (sum(rate(container_cpu_usage_seconds_total{image!="" , pod=~".*adapter.*"}[5m])) by (pod,namespace) > {{ .Values.global.cluster.cpu_threshold }}*2)
OID	1.3.6.1.4.1.323.5.3.51.29.4002
Metric Used	container_cpu_usage_seconds_total **Note:** This is a Kubernetes metric used for instance availability monitoring. If the metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the CPU utilization is below the critical threshold. **Note:** The threshold is configurable in the `ocnadd/values.yaml` file. If guidance is required, contact #unique_41.

OCNADD_POD_MEMORY_USAGE_ALERT

Table 5-2 OCNADD_POD_MEMORY_USAGE_ALERT

Field	Details
Triggering Condition	POD Memory usage is above set threshold (default 70%)
Severity	Major
Description	OCNADD Pod High Memory usage detected for the continuous period of 5min
Alert Details	**Summary:** 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: Memory usage is {{ "{{" }} $value | printf "%.2f" }} which is above threshold {{ .Values.global.cluster.memory_threshold }} % ' **Expression**: (sum(container_memory_working_set_bytes{image!="" , pod=~".*ocnadd.*"}) by (pod,namespace) > 2*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*kafka.*"}) by (pod,namespace) > 24*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*zookeeper.*"}) by (pod,namespace) > 1*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*egw.*"}) by (pod,namespace) > 2*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100) or (sum(container_memory_working_set_bytes{image!="" , pod=~".*adapter.*"}) by (pod,namespace) > 2*1024*1024*1024*{{ .Values.global.cluster.memory_threshold }}/100)
OID	1.3.6.1.4.1.323.5.3.51.29.4005
Metric Used	container_memory_working_set_bytes Note : This is a Kubernetes metric used for instance availability monitoring. If the metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the memory utilization is below the critical threshold. Note: The threshold is configurable in the `ocnadd/values.yaml` file. If guidance is required, contact #unique_41 .

OCNADD_POD_RESTARTED

Table 5-3 OCNADD_POD_RESTARTED

Field	Details
Triggering Condition	A POD has restarted
Severity	Minor
Description	A POD has restarted in last 2 min
Alert Details	**Summary:** 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: A Pod has restarted' **Expression**: expr: kube_pod_container_status_restarts_total{namespace="{{ .Values.global.cluster.nameSpace.name }}"} > 1
OID	1.3.6.1.4.1.323.5.3.51.29.5006
Metric Used	kube_pod_container_status_restarts_total **Note:** This is a Kubernetes metric used for instance availability monitoring. If the metric is not available, use the similar metric asexposed by the monitoring system.
Resolution	The alert is cleared automatically if the specific pod is up. **Steps**: 1. Check the application logs. Check for database related failures such as connectivity, Kubernetes secrets, and so on. 2. Run the following command to check orchestration logs for liveness or readiness probe failures: 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 #unique_41, If guidance is required.

Application Level Alerts

This section lists the application level alerts for OCNADD.

OCNADD_CONFIG_SVC_DOWN

Table 5-4 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	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: ocnaddconfiguration service is down' **Expression:** expr: up{service="ocnaddconfiguration"} != 1
OID	1.3.6.1.4.1.323.5.3.51.20.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.
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	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: ocnaddconfiguration service is down' **Expression:** expr: up{service="ocnaddconfiguration"} != 1
OID	1.3.6.1.4.1.323.5.3.51.20.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.

Table 5-4 (Cont.) OCNADD_CONFIG_SVC_DOWN

Field	Details
Resolution	The alert is cleared automatically when the OCNADD Configuration service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: 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 #unique_41, If guidance is required..

OCNADD_ALARM_SVC_DOWN

Table 5-5 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
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: ocnaddalarm service is down' **Expression:** expr: up{service="ocnaddalarm"} != 1
OID	1.3.6.1.4.1.323.5.3.51.24.2002

Table 5-5 (Cont.) OCNADD_ALARM_SVC_DOWN

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

Table 5-5 (Cont.) OCNADD_ALARM_SVC_DOWN

Field	Details
Resolution	The alert is cleared automatically when the OCNADD Alarm service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: 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 #unique_41, If guidance is required..

OCNADD_HEALTH_MONITORING_SVC_DOWN

Table 5-6 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 5-6 (Cont.) OCNADD_HEALTH_MONITORING_SVC_DOWN

Field	Details
Alert Details	**Summary**: summary: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: ocnaddhealthmonitoring service is down' **Expression:** expr: up{service="ocnaddhealthmonitoring"} != 1
OID	1.3.6.1.4.1.323.5.3.51.28.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Health monitoring service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: 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 #unique_41, If guidance is required.

OCNADD_SCP_AGGREGATION_SVC_DOWN

Table 5-7 OCNADD_SCP_AGGREGATION_SVC_DOWN

Field	Details
Severity	Critical
Description	OCNADD SCP Aggregation service not available for more than 2 min
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: ocnaddscpaggregation service is down' **Expression:** expr: up{service="ocnaddscpaggregation"} != 1
OID	1.3.6.1.4.1.323.5.3.51.22.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD SCP Aggregation service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: 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 #unique_41, If guidance is required.

Table 5-7 (Cont.) OCNADD_SCP_AGGREGATION_SVC_DOWN

Field	Details
Triggering Condition	The SCP Aggregation service went down or not accessible

OCNADD_NRF_AGGREGATION_SVC_DOWN

Table 5-8 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	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: ocnaddnrfaggregation service is down' **Expression:** expr: up{service="ocnaddnrfaggregation"} != 1
OID	1.3.6.1.4.1.323.5.3.51.22.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.

Table 5-8 (Cont.) OCNADD_NRF_AGGREGATION_SVC_DOWN

Field	Details
Resolution	The alert is cleared automatically when the OCNADD NRF Aggregation service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: kubectl –n <namespace> get pod If it is not in running state, capture the pod logs and events. Run the following command to fetch the events as follows: 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 #unique_41, If guidance is required.

OCNADD_ADMIN_SVC_DOWN

Table 5-9 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
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: ocnaddadminservice service is down' **Expression:** expr: up{service="ocnaddadminservice"} != 1
OID	1.3.6.1.4.1.323.5.3.51.30.2002

Table 5-9 (Cont.) OCNADD_ADMIN_SVC_DOWN

Field	Details
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Admin service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: kubectl –n <namespace> get pod If it is not in running state, capture the pod logs and events. Run the following command to fetch the events as follows: kubectl get events --sortby=.metadata.creationTimestamp -n <namespace> 3. Refer to the application logs and check for database related failures such as connectivity, invalid secrets, and so on. 4. Run the following command to check Helm status and make sure there are no errors: helm status <helm release name of data director> -n<namespace> If it is not in "STATUS: DEPLOYED", then again capture logs and events. 5. If the issue persists, capture all the outputs from the above steps and contact #unique_41, If guidance is required.

OCNADD_EGW_SVC_DOWN

Table 5-10 OCNADD_EGW_SVC_DOWN

Field	Details
Severity	Critical
Description	OCNADD Egress Gateway service not available for more than 2 min

Table 5-10 (Cont.) OCNADD_EGW_SVC_DOWN

Field	Details
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: Egress Gw service is down' **Expression:** expr: up{service=~".*egw.*"} != 1
OID	1.3.6.1.4.1.323.5.3.51.23.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Egress Gateway 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 #unique_41, If guidance is required.
Triggering Condition	The OCNADD Egress Gateway service went down or not accessible

OCNADD_CONSUMER_ADAPTER_SVC_DOWN

Table 5-11 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
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: Consumer Adapter service is down' **Expression:** expr: up{service=~".*adapter.*"} != 1
OID	1.3.6.1.4.1.323.5.3.51.25.2002
Metric Used	'up' Note: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert is cleared automatically when the OCNADD Consumer Adapter service start becoming available. Steps: 1. Check for service specific alerts which may be causing the issues with service exposure. 2. Run the following command to check if the pod's status is in "Running" state: 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 #unique_41, If guidance is required.

OCNADD_MPS_WARNING_INGRESS_THRESHOLD_CROSSED

Table 5-12 OCNADD_MPS_WARNING_INGRESS_THRESHOLD_CROSSED

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

OCNADD_MPS_MINOR_INGRESS_THRESHOLD_CROSSED

Table 5-13 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 5-13 (Cont.) OCNADD_MPS_MINOR_INGRESS_THRESHOLD_CROSSED

Field	Details
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above 90 Percent of Max messages per second: {{ .Values.global.cluster.mps }}' **Expression:** expr: sum(rate(kafka_stream_processor_node_process_total[5m])) by (namespace) > 0.9*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5007
Metric Used	kafka_stream_processor_node_process_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the minor threshold alert level of 90%.

OCNADD_MPS_MAJOR_INGRESS_THRESHOLD_CROSSED

Table 5-14 OCNADD_MPS_MAJOR_INGRESS_THRESHOLD_CROSSED

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

OCNADD_MPS_CRITICAL_INGRESS_THRESHOLD_CROSSED

Table 5-15 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	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above the supported Max messages per second: {{ .Values.global.cluster.mps }}' **Expression:** expr: sum(rate(kafka_stream_processor_node_process_total[5m])) by (namespace) > 1.0*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5007
Metric Used	kafka_stream_processor_node_process_total
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	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above the supported Max messages per second: {{ .Values.global.cluster.mps }}' **Expression:** expr: sum(rate(kafka_stream_processor_node_process_total[5m])) by (namespace) > 1.0*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5007
Metric Used	kafka_stream_processor_node_process_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the critical threshold alert level of 100% of supported MPS

OCNADD_MPS_WARNING_EGRESS_THRESHOLD_CROSSED

Table 5-16 OCNADD_MPS_WARNING_EGRESS_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total egress MPS crossed the warning threshold alert level of 80% of the supported MPS
Severity	Warn
Description	Total Egress Message Rate is above configured warning threshold alert (80%) for the period of 5 min
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above 80% of Max messages per second: {{ .Values.global.cluster.mps }}' **Expression:** expr: sum (rate(ocnadd_egressgateway_http_requests_total[5m])) by (namespace) > 0.80*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5008
Metric Used	ocnadd_egressgateway_http_requests_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the warning threshold alert level of 80% of supported MPS

OCNADD_MPS_MINOR_EGRESS_THRESHOLD_CROSSED

Table 5-17 OCNADD_MPS_MINOR_EGRESS_THRESHOLD_CROSSED

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

Table 5-17 (Cont.) OCNADD_MPS_MINOR_EGRESS_THRESHOLD_CROSSED

Field	Details
OID	1.3.6.1.4.1.323.5.3.51.29.5008
Metric Used	ocnadd_egressgateway_http_requests_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the minor threshold alert level of 90% of supported MPS

OCNADD_MPS_MAJOR_EGRESS_THRESHOLD_CROSSED

Table 5-18 OCNADD_MPS_MAJOR_EGRESS_THRESHOLD_CROSSED

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

OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED

Table 5-19 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

Table 5-19 (Cont.) OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED

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

OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED_FOR_A_CONSUMER

Table 5-20 OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED_FOR_A_CONSUMER

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

Table 5-20 (Cont.) OCNADD_MPS_CRITICAL_EGRESS_THRESHOLD_CROSSED_FOR_A_CONSUMER

Field	Details
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: Message Rate is above supported Max messages per second:{{ .Values.global.cluster.mps }}' **Expression:** expr: sum (rate(ocnadd_egressgateway_http_requests_total[5m])) by (namespace, instance_identifier) > 1.0*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5009
Metric Used	ocnadd_egressgateway_http_requests_total
Resolution	The alert is cleared automatically when the the MPS rate goes below the critical threshold alert level of 100% of supported MPS

OCNADD_E2E_AVG_RECORD_LATENCY_WARNING_THRESHOLD_CROSSED

Table 5-21 OCNADD_E2E_AVG_RECORD_LATENCY_WARNING_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total observed latency is above the configured warning threshold alert level of 80%
Severity	Warn
Description	Average E2E Latency is above configured warning threshold alert level (80%) for the period of 5 min

Table 5-21 (Cont.) OCNADD_E2E_AVG_RECORD_LATENCY_WARNING_THRESHOLD_CROSSED

Field	Details
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: E2E Latency is above 80% of Maximum permissable latency {{ .Values.global.cluster.max_latency }} ms' **Expression:** expr: (sum (irate(ocnadd_egressgateway_e2e_request_processing_latency_seconds_sum[5m])) by (namespace)) /(sum (irate(ocnadd_egressgateway_e2e_request_processing_latency_seconds_count[5m])) by (namespace)) > .80*{{ .Values.global.cluster.max_latency }} <= .90*{{ .Values.global.cluster.max_latency }}
OID	1.3.6.1.4.1.323.5.3.51.29.5010
Metric Used	ocnadd_egressgateway_e2e_request_processing_latency_seconds_sum, ocnadd_egressgateway_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 permissable latency

OCNADD_E2E_AVG_RECORD_LATENCY_MINOR_THRESHOLD_CROSSED

Table 5-22 OCNADD_E2E_AVG_RECORD_LATENCY_MINOR_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total observed latency is above the configured minor threshold alert level of 90%
Severity	Minor
Description	Average E2E Latency is above configured minor threshold alert level (90%) for the period of 5 min

Table 5-22 (Cont.) OCNADD_E2E_AVG_RECORD_LATENCY_MINOR_THRESHOLD_CROSSED

Field	Details
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: E2E Latency is above 90% of Maximum permissable latency {{ .Values.global.cluster.max_latency }} ms' **Expression:** expr: (sum (irate(ocnadd_egressgateway_e2e_request_processing_latency_seconds_sum[5m])) by (namespace)) /(sum (irate(ocnadd_egressgateway_e2e_request_processing_latency_seconds_count[5m])) by (namespace)) > .90*{{ .Values.global.cluster.max_latency }} <= 0.95*{{ .Values.global.cluster.max_latency }}
OID	1.3.6.1.4.1.323.5.3.51.29.5010
Metric Used	ocnadd_egressgateway_e2e_request_processing_latency_seconds_sum, ocnadd_egressgateway_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 permissable latency

OCNADD_E2E_AVG_RECORD_LATENCY_MAJOR_THRESHOLD_CROSSED

Table 5-23 OCNADD_E2E_AVG_RECORD_LATENCY_MAJOR_THRESHOLD_CROSSED

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

Table 5-23 (Cont.) OCNADD_E2E_AVG_RECORD_LATENCY_MAJOR_THRESHOLD_CROSSED

Field	Details
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: E2E Latency is above 95% of Maximum permissable latency {{ .Values.global.cluster.max_latency }} ms' **Expression:** expr: (sum (irate(ocnadd_egressgateway_e2e_request_processing_latency_seconds_sum[5m])) by (namespace)) /(sum (irate(ocnadd_egressgateway_e2e_request_processing_latency_seconds_count[5m])) by (namespace)) > .95*{{ .Values.global.cluster.max_latency }} <= 1.0*{{ .Values.global.cluster.max_latency }}
OID	1.3.6.1.4.1.323.5.3.51.29.5010
Metric Used	ocnadd_egressgateway_e2e_request_processing_latency_seconds_sum, ocnadd_egressgateway_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 permissable latency

OCNADD_E2E_AVG_RECORD_LATENCY_CRITICAL_THRESHOLD_CROSSED

Table 5-24 OCNADD_E2E_AVG_RECORD_LATENCY_CRITICAL_THRESHOLD_CROSSED

Field	Details
Triggering Condition	The total observed latency is above the configured critical threshold alert level of 100%
Severity	Critical
Description	Average E2E Latency is above configured critical threshold alert level (100%) for the period of 5 min

Table 5-24 (Cont.)
OCNADD_E2E_AVG_RECORD_LATENCY_CRITICAL_THRESHOLD_CROSSED

Field	Details
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: E2E Latency is above 100% of Maximum permissable latency {{ .Values.global.cluster.max_latency }} ms' **Expression:** expr: (sum (irate(ocnadd_egressgateway_e2e_request_processing_latency_seconds_sum[5m])) by (namespace)) /(sum (irate(ocnadd_egressgateway_e2e_request_processing_latency_seconds_count[5m])) by (namespace)) > 1.0*{{ .Values.global.cluster.max_latency }}
OID	1.3.6.1.4.1.323.5.3.51.29.5010
Metric Used	ocnadd_egressgateway_e2e_request_processing_latency_seconds_sum, ocnadd_egressgateway_e2e_request_processing_latency_seconds_count
Resolution	The alert is cleared automatically when the average latency goes below the critical threshold alert level of permissable latency

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

Table 5-25 (Cont.)
OCNADD_KAFKA_PACKET_DROP_THRESHOLD_1PERCENT_MPS

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

OCNADD_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 total supported MPS
Severity	Critical
Description	The packet drop rate in Kafka streams is above the configured critical threshold of 10% of total supported MPS in the period of 5 min
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: Packet Drop rate is above 10% thereshold of Max messages per second: {{ .Values.global.cluster.mps }}' **Expression:** expr: sum(rate(kafka_stream_task_dropped_records_total[5m])) by (namespace) > 0.1*{{ .Values.global.cluster.mps }}
OID	1.3.6.1.4.1.323.5.3.51.29.5011
Metric Used	kafka_stream_task_dropped_records_total

Table 5-26 (Cont.)
OCNADD_KAFKA_PACKET_DROP_THRESHOLD_10PERCENT_MPS

Field	Details
Resolution	The alert is cleared automatically when the packet drop rate goes below the critical threshold (10%) alert level of supported MPS

OCNADD_EGW_FAILURE_RATE_THRESHOLD_0.1PERCENT

Table 5-27 OCNADD_EGW_FAILURE_RATE_THRESHOLD_0.1PERCENT

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

OCNADD_EGW_FAILURE_RATE_THRESHOLD_1PERCENT

Table 5-28 OCNADD_EGW_FAILURE_RATE_THRESHOLD_1PERCENT

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

OCNADD_EGW_FAILURE_RATE_THRESHOLD_10PERCENT

Table 5-29 OCNADD_EGW_FAILURE_RATE_THRESHOLD_10PERCENT

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

Table 5-29 (Cont.) OCNADD_EGW_FAILURE_RATE_THRESHOLD_10PERCENT

Field	Details
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: Egress external connection Failure Rate detected above 10 Percent of Total Egress external connections' **Expression:** expr: (sum(rate(ocnadd_egressgateway_third_party_connection_failure_total[5m])) by (namespace))/ (sum(rate(ocnadd_egressgateway_http_requests_total[5m])) by (namespace)) *100 >= 10 < 25
OID	1.3.6.1.4.1.323.5.3.51.29.5012
Metric Used	ocnadd_egressgateway_third_party_connection_failure_total, ocnadd_egressgateway_http_requests_total
Resolution	The alert is cleared automatically when the failure rate towards 3rd party consumer goes below the threshold (10%) alert level of supported MPS

OCNADD_EGW_FAILURE_RATE_THRESHOLD_25PERCENT

Table 5-30 OCNADD_EGW_FAILURE_RATE_THRESHOLD_25PERCENT

Field	Details
Triggering Condition	The Egress gateway failure rate towards the 3rd party application is above the configured threshold of 25% of total supported MPS
Severity	Major
Description	Egress external connection failure rate towards 3rd party application is crossing major threshold of 25% in the period of 5 min
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: Egress external connection Failure Rate detected above 25 Percent of Total Egress external connections' **Expression:** expr: (sum(rate(ocnadd_egressgateway_third_party_connection_failure_total[5m])) by (namespace))/ (sum(rate(ocnadd_egressgateway_http_requests_total[5m])) by (namespace)) *100 >= 25 < 50

Table 5-30 (Cont.) OCNADD_EGW_FAILURE_RATE_THRESHOLD_25PERCENT

Field	Details
OID	1.3.6.1.4.1.323.5.3.51.29.5012
Metric Used	ocnadd_egressgateway_third_party_connection_failure_total, ocnadd_egressgateway_http_requests_total
Resolution	The alert is cleared automatically when the failure rate towards 3rd party consumer goes below the threshold (25%) alert level of supported MPS

OCNADD_EGW_FAILURE_RATE_THRESHOLD_50PERCENT

Table 5-31 OCNADD_EGW_FAILURE_RATE_THRESHOLD_50PERCENT

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

OCNADD_INGRESS_TRAFFIC_RATE_INCREASE_SPIKE_10PERCENT

Table 5-32 OCNADD_INGRESS_TRAFFIC_RATE_INCREASE_SPIKE_10PERCENT

Field	Details
Triggering Condition	The ingress traffic increase is more than 10% of the supported MPS
Severity	Major
Description	The ingress traffic increase is more than 10% of the supported MPS in last 5 min
Alert Details	**Summary**: 'namespace: {{ "{{" }}$labels.namespace}}, podname: {{ "{{" }}$labels.pod}}, timestamp: {{ "{{" }} with query "time()" }}{{ "{{" }} . | first | value | humanizeTimestamp }}{{ "{{" }} end }}: Ingress MPS increase is more than 10 Percent of current supported MPS' **Expression:** expr: sum(irate(kafka_stream_processor_node_process_total[5m])) by (namespace)/ sum(irate(kafka_stream_processor_node_process_total[5m] offset 5m)) by (namespace) >= 1.1
OID	1.3.6.1.4.1.323.5.3.51.29.5013
Metric Used	kafka_stream_processor_node_process_total
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 5-33 OCNADD_INGRESS_TRAFFIC_RATE_DECREASE_SPIKE_10PERCENT

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

Table 5-33 (Cont.)
OCNADD_INGRESS_TRAFFIC_RATE_DECREASE_SPIKE_10PERCENT

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