Oracle® Communications

Network Analytics Data Director Installation, Upgrade, and Fault Recovery Guide

Release 25.2.101
G41328-05
June 2026

ORACLE

Oracle Communications Network Analytics Data Director Installation, Upgrade, and Fault Recovery Guide, Release 25.2.101

G41328-05

Contents

Preface

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.
italic	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 and Terminology

Acronym	Definition
CA	Certificate Authority
CNC Console	Cloud Native Configuration Console
CNLB	Cloud Native Load Balancer
CLI	Command Line Interface
CN	Common Name
CSP	Communication Service Provider
OKE	Container Engine for Kubernetes
KPI	Key Performance Indicator
MPS	Messages Per Second
MOS	My Oracle Support
NDB	Network Data Broker
NF	Network Function
OCI	Oracle Cloud Infrastructure
OCCM	Oracle Communication Certificate Manager
BSF	Oracle Communications Cloud Native Core, Binding Support Function
CNE	Oracle Communications Cloud Native Core, Cloud Native Environment
NRF	Oracle Communications Cloud Native Core, Network Repository Function (NRF)
PCF	Oracle Communications Cloud Native Core, Policy Control Function
SEPP	Oracle Communications Cloud Native Core, Security Edge Protection Proxy
SCP	Oracle Communications Cloud Native Core, Service Communication Proxy
NWDAF	Oracle Communications Networks Data Analytics Function
OHC	Oracle Help Center
OSDC	Oracle Service Delivery Cloud
SVC	Services
SAN	Subject Alternate Name
TLS	Transport Layer Security
URI	Uniform Resource Identifier

What's New in This Guide

This section lists the documentation updates for Release *25.2.1xx* in *Oracle Communications Network Analytics Data Director Installation, Upgrade, and Fault Recovery Guide*.

Release 25.2.101 - G41328-05, June 2026

- Updated the release number to 25.2.101 in the entire document.
- Updated Supported Upgrade Paths.
- Updated Supported Rollback Paths in the Rolling Back OCNADD section.
- Added the details of the following parameter in the Aggregation Service Parameters section:
  - SEPP_DUAL_SITE_ENABLED
- Added the details of the following parameter in the Consumer Aapter Parameters section:
  - ENABLE_TCP_NODELAY
  - NETTYCLIENT_EVENTLOOP_MULTIPLIER
  - EGRESS_REQUEST_COMPLETE_TIMEOUT
  - EGRESS_KEEPALIVE
  - EGRESS_HTTP_MAX_BLOCK_DURATION
  - EGRESS_HTTP_SYNC_ASYNC_RETRY_MAX_ATTEMPT
  - EGRESS_HTTP_SYNC_RETRY_DELAY_MS
  - ADAPTER_METRICS_PER_CONN_ENABLED

Release 25.2.100 - G41328-04, December 2025

- Updated the software details in Software Requirements section.
- Updated the supported cnDBTier versions in Environment Setup Requirements section.

Release 25.2.100 - G41328-03, November 2025

- Updated the stem sentence of step 2 for Configuring ServiceMonitor in OCCNE-INFRA.

Release 25.2.100 - G41328-02, November 2025

- Updated steps for Configuring ServiceMonitor in OCCNE-INFRA.
- Updated Upgrade Tasks – Step 1 by replacing incorrect management group references with worker group instances. For example, occurrences of `mgmt` were replaced with `wg` in the commands.
- Performed the following changes in the Post Upgrade Task section:
  - Added a note specifying that Steps 1, 2, and 3 are required only when OCCM is used to manage certificates in both the source and target releases.
  - Updated Step 1 to indicate that it is optional and required only if certificate updates are needed.
  - Updated Step 2 to indicate that it is conditional and required only if Step 1 was completed.

Release 25.2.100 - G41328-01, October 2025

- **General Updates**:
  - Updated the release number to 25.2.100 in the entire document.
  - Removed all the instances of HNS and hnc-controller as it is not supported in this release.
  - Updated the software details in Software Requirements section.
  - Updated the Environment Setup Requirements details.
  - Removed 'ocnaddadmin' service from management group and added it in the worker group in the Resource Requirements section.
- **Installation Updates**:
  - Updated the docker images and version details in Pushing the Images to Customer and OCI Registry section.
  - Update the notes and multiple commands in the Creating OCNADD Namespace section.
  - Updated the steps for Configuring OCNADD Database.
  - Updated the procedure for Configuring SSL or TLS Certificates.
  - Added the steps for "Configuring ServiceMonitor in OCCNE-INFRA" in the OCCM Prerequisites for Installing OCNADD section.
  - Updated the versions and code snippets in the procedure for Installing OCNADD Package.
  - Updated the procedure for Adding a Worker Group and Deleting a Worker Group.
  - Added the details for Enabling Druid as Extended Storage Feature in "PostInstallation Tasks" section.
  - Updated the default parameters listed for Model 1 and Model 2 in the OCNADD Deployment Models section.
  - Performed the following updates to the parameters listed in the following sections:
    - Added the details of the following parameters in the Global Parameters section:
      - `extendedStorage.druid.enabled`
      - `extendedStorage.druid.druidTLsEnabled`
      - `extendedStorage.druid.namespace`
      - `extendedStorage.druid.service_ip`
      - `extendedStorage.druid.service_port`
      - `String NA 8080 M The port of the Druid router service`
    - Added details of the following parameter in Kafka Configuration Parameters section:
      - `kafkaBroker.kafkaProperties.ramDriveStorage`
- **Upgrade, Rollback, and Uninstall Updates**:
  - Updated the Supported Upgrade Paths.
  - Updated the steps in Preupgrade Tasks.
  - Updated the steps and scenarios in Upgrade Tasks.

- Added the details of "Migration of Kafka to Kraft Mode" in Post Upgrade Task.
- Updated the procedure and scenarios for Rolling Back OCNADD.
- Updated the procedure for Uninstalling OCNADD.
- Updated the fault recovery procedure for Creating OCNADD Restore Job.

1 Introduction

This chapter provides information about installing Oracle Communications Network Analytics Data Director (OCNADD) and its microservices on the supported platforms.

> ⚠ **Caution**
>
> User, computer and applications, and character encoding settings may cause an issue when copy-pasting commands or any content from PDF. PDF reader version also affects the copy-pasting functionality. It is recommended to verify the pasted content especially when hyphens or any special characters are part of copied content.

1.1 Overview

With the emergence of 5G networks, Communication Service Providers (CSPs) now have access to vast amounts of data. Oracle Communications Network Analytics Data Director (OCNADD) serves as a specialized Network Data Broker (NDB) within the 5G Network Architecture. It collects network traffic data from various sources such as 5G network functions (NFs), Non-5G NFs, and third-party producers. OCNADD then performs a range of rule based operations to assist CSPs in maximizing the benefits of this data. These operations include data aggregation, data filtering, data replication, data governance, and secure data transmission for subscribed third-party consumers (third-party consumer applications or platforms).

By efficiently collecting and utilizing data, OCNADD provides the following advantages to CSPs:

- Enhanced service quality
- Ease of scalability
- Simplified monitoring and troubleshooting in the event of failures
- Assistance in identifying new revenue and network monetization streams
- Reduction in network downtime

As an NDB, OCNADD sits in between the 5G infrastructure and third-party tools or consumer applications. Its primary function is to ensure data security, low latency, and redundancy while efficiently collecting and processing data. By correlating and transforming acquired data based on configurable data feed settings, OCNADD allows Communication Service Providers (CSPs) to generate comprehensive dashboards and Key Performance Indicators (KPIs). These insights provide a deep understanding of all functions within the 5G Network Architecture, enabling CSPs to enhance service quality, reduce downtime, facilitate network scalability, and minimize losses.

During network failures, OCNADD data is used for monitoring and troubleshooting the issues. Additionally, OCNADD offers a user-friendly GUI that facilitates the creation, editing, and deletion of data feeds. For more information about OCNADD architecture and features, see *Oracle Communications Network Analytics Data Director User Guide*.

Deployment Overview

OCNADD deployment models help customers to optimize or reduce the Data Director's footprint based on the third party consumer's capability to consume data directly from the Kafka cluster. Users can deploy different Data Director models by configuring the applicable custom values. The following deployment models are supported:

- **Model 1:** All Data Director Services (Default)
- **Model 2:** Kafka, Common Services, and Aggregation Services

For more information on deployment models, see OCNADD Deployment Models section.

Installation Overview

The OCNADD installation comprises of various tasks including prerequisites, preinstallation, and installation. Perform the installation tasks in the same sequence as outlined in the following table:

Table 1-1 Installation Overview

Installation Sequence	Installation Sub-Sections	Applicable to CNE	Applicable to OCI
Prerequisites	Software Requirements	Yes	Yes
	Environment Setup Requirements	Yes	*Oracle Communications Cloud Native Core OCI Adaptor, NF Deployment on OCI Guide*
	Resource Requirements	Yes	Yes
Pre-Installation Tasks	Downloading OCNADD Package	Yes	Yes
	Pushing OCNADD Images to Customer Registry	Yes	No
	Pushing OCNADD Images to OCI Registry	No	Yes
	Creating OCNADD Namespace	Yes	Yes
	Creating Service Account, Role, and Role Binding	Yes	Yes
	Configuring OCNADD Database	Yes	Yes
	Configuring Secrets for Accessing OCNADD Database	Yes	Yes
	Configuring IP Network	Yes	Yes
	Configuring SSL or TLS Certificates	Yes	Yes
Installation Tasks	Installing OCNADD Package	Yes	Yes
	Verifying OCNADD Installation	Yes	Yes
	Creating OCNADD Kafka Topics	Yes	Yes
	Installing OCNADD GUI	Yes	Yes
	Adding a Worker Group	Yes	Yes
	Deleting a Worker Group	Yes	Yes
	Creating Alarms and Dashboard in OCI	No	Yes

Table 1-1 (Cont.) Installation Overview

Installation Sequence	Installation Sub-Sections	Applicable to CNE	Applicable to OCI
Post-Installation Tasks	Enabling Two Site Redundancy	*Oracle Communications Network Analytics Data Director User Guide*	*Oracle Communications Network Analytics Data Director User Guide*

1.2 References

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

- *Oracle Communications Network Analytics Data Director User Guide*
- *Oracle Communications Network Analytics Data Director Troubleshooting Guide*
- *Oracle Communications Cloud Native Environment Installation, Upgrade, and Fault Recovery 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 Network Analytics Suite Security Guide*
- *Oracle Communications Network Analytics Data Director Benchmarking Guide*
- *Oracle Communications Cloud Native Core, OCI Deployment Guide*
- *Oracle Communications Cloud Native Core, OCI Adaptor User Guide*
- *Oracle Communication Certificate Manager Installation, Upgrade and Fault Recovery Guide*
- *Oracle Communication Certificate Manager User Guide*

1.3 Oracle Error Correction Policy

The table below outlines the key details for the current and past releases, their General Availability (GA) dates, and the end dates for the Error Correction Grace Period.

Table 1-2 Oracle Error Correction Policy

Release Number	General Availability (GA) Date	Error Correction Grace Period End Date
25.2.100	September 2025	September 2026
25.1.200	July 2025	July 2026
25.1.100	February 2025	February 2026
3.24.3	November 2024	November 2025

Note

- For the latest patch releases, see their corresponding *Oracle Communications Cloud Native Core Release Notes*.
- For a release, Sev1 and Critical Patch Unit (CPU) patches are supported for 12 months. For more information, see the Oracle Communications Cloud Native Core and Network Analytics Error Correction Policy.

1.4 Oracle Open Source Support Policies

Oracle Communications Cloud Native Core uses open source technology governed by the Oracle Open Source Support Policies. For more information, see Oracle Open Source Support Policies.

2 Installing OCNADD

This chapter provides information about installing Oracle Communications Network Analytics Data Director (OCNADD) on the supported platforms.

The OCNADD installation is supported over the following platforms:

- Oracle Communications Cloud Native Core, Cloud Native Environment (CNE)
- VMware Tanzu Application Platform (TANZU)
- Oracle Cloud Infrastructure (OCI)

> **Note**
>
> This document describes the OCNADD installation on CNE. However, the procedure for installation on OCI and TANZU is similar to the installation on CNE. Any steps unique to OCI or TANZU platform are mentioned explicitly in the document.

2.1 Prerequisites

Before installing and configuring OCNADD, make sure that the following requirements are met:

2.1.1 Software Requirements

This section lists the software that must be installed before installing OCNADD:

Table 2-1 Mandatory Software

Software	Version
Kubernetes	1.33.x, 1.32.x, 1.31.x
Helm	3.15.2
Docker/Podman	4.6.1
OKE (on OCI)	1.27.x

> **Note**
>
> - OCNADD 25.2.101 supports CNE25.2.1xx, 25.1.2xx, and CNE 25.1.1xx.

To check the Oracle Communications Cloud Native Core, Cloud Native Environment (CNE) version, run the following command:

```
echo $OCCNE_VERSION
```

To check the current Helm and Kubernetes versions installed in CNE, run the following commands:

```
kubectl version
```

```
helm version
```

> **Note**
>
> - Starting with CNE 1.8.0, Podman is the preferred container platform instead of docker. For more information on installing and configuring Podman, see the *Oracle Communications Cloud Native Core, Cloud Native Environment Installation, Upgrade, and Fault Recovery Guide*.

If you are installing OCNADD on TANZU, the following software must be installed:

Table 2-2 Mandatory Software

Software	Version
Tanzu	1.4.1

To check the current TANZU version, run the following command:

```
tanzu version
```

> **Note**
>
> Tanzu was supported in release 22.4.0. Release 25.2.101 has not been tested on Tanzu.

Depending on the requirement, you may have to install additional software while deploying OCNADD. The list of additional software items, along with the supported versions and usage, is given in the following table:

Table 2-3 Additional Software

Software	Version	Required For
Prometheus-Operator	2.52.0	Metrics
Metallb	0.14.4	LoadBalancer
cnDBTier	25.1.2xx and 25.1.1xx.	MySQL Database
Druid	33.0.0	It is required for extended storage integration with the Druid database.

> **Note**
>
> - Some of the software are available by default if OCNADD is getting deployed in Oracle Communications Cloud Native Core, Cloud Native Environment (CNE).
> - Install the additional software if any of them are not available by default with CNE.
> - If you are deploying OCNADD in any other environment, for instance, TANZU, then all the above mentioned software must be installed before installing OCNADD.
> - On OCI, the Prometheus-Operator is not required. The metrics and alerts will be managed using OCI monitoring and Alarm services.

To check the installed software items, run the following command:

```
helm ls -A
```

2.1.2 Environment Setup Requirements

This section provides information on environment setup requirements for installing Oracle Communications Network Analytics Data Director (OCNADD).

Network Requirements

The Data Director services, such as Kafka and Redundancy Agent, require external access. These services are created as load balancer services, and the service FQDNs should be used for communication with them. Additionally, the service FQDNs must be configured in the DNS server.

CNLB Network and NADs for Data Director

Egress NADs

Customers must identify or create Egress NADs for their third-party feed endpoint requirements before installing the CNLB CNE cluster. These Egress NADs must be defined in the `cnlb.ini` file of OCCNE to enable CNLB support.

Egress NADs are required for the following traffic segregation scenarios:

- **Separate Egress NAD per third-party destination endpoint per feed**:
  Each destination endpoint of a Consumer Adapter feed will have a dedicated egress network via a separate Egress NAD managed by CNLB.
- **Separate Egress NAD per third-party feed**:
  Each Consumer Adapter feed will have its own dedicated egress network via a separate Egress NAD managed by CNLB.
- **Separate Egress NAD per Data Director**:
  All Consumer Adapter feeds within a Data Director will share a single separate egress network via one Egress NAD managed by CNLB.

Ingress NADs

Customers must identify or create the required CNLB IPs (external IPs) and Ingress NADs for the Data Director Ingress Adapter service.

Based on ingress traffic segregation requirements for non-Oracle NFs, the necessary CNLB IPs (external IPs) and corresponding Ingress NADs must be pre-configured for the Ingress Adapter. These configurations must be defined in the `cnlb.ini` file of OCCNE for CNLB support.

Key considerations:

- Each Ingress Adapter service instance must have a dedicated external IP and a corresponding Ingress NAD managed by CNLB.
- Customers must identify or create an Ingress NAD for the Redundancy Agent's external access and IP.
- Customers must also define the required CNLB IPs (external IPs) and Ingress NADs for the Data Director Kafka service. The number of Ingress NADs and external IPs must match the number of Kafka brokers in the cluster. This configuration is required for each existing or future additional worker group.
- The necessary CNLB external IPs and corresponding Ingress NADs must be defined in the `cnlb.ini` file of OCCNE for CNLB support.

> **Note**
>
> - For more information about the feature see, "Enabling or Disabling Traffic Segregation Through CNLB in OCNADD " in the *Oracle Communications Network Analytics Data Director User Guide.*
> - For more information on CNLB and NADs, see the *Oracle Communications Cloud Native Environment Installation, Upgrade, and Fault Recovery Guide*.

Environment Setup on OCCNE

Network Access

The Kubernetes cluster hosts must have network access to the following repositories:

1. **Local container image repository**: It contains the OCNADD container images. To check if the Kubernetes cluster hosts can access the local container image repository, pull any image with an image-tag using the following command:

   ```
   podman pull docker-repo/image-name:image-tag
   ```

   where,

   - `docker-repo` is the IP address or hostname of the container image repository.
   - `image-name` is the container image name.
   - `image-tag` is the tag assigned to the container image used for the OCNADD pod.

2. **Local Helm repository**: It contains the OCNADD Helm charts. To check if the Kubernetes cluster hosts can access the local Helm repository, run the following command:

   ```
   helm repo update
   ```

3. Service FQDN or IP Addresses of the required OCNADD services, for instance, Kafka Brokers, must be discoverable from outside of the cluster. This information should be publicly exposed so that Ingress messages to OCNADD can come from outside of Kubernetes.

Environment Setup on OCI

OCNADD can be deployed in OCI. While deploying OCNADD on OCI, the user must use the Operator instance/VM instead of Bastion Host.

For OCI infrastructure, see *Oracle Communications Cloud Native Core, OCI Deployment Guide* and *Oracle Communications Cloud Native Core, OCI Adaptor User Guide* documents.

After completing the OCI infrastructure setup requirements, proceed to the next section.

Client Machine Requirements

> **Note**
>
> Run all the `kubectl` and `helm` commands in this guide on a system depending on the infrastructure and deployment. This system could be a client machine, such as a virtual machine, server, local desktop, etc.

This section describes the requirements for client machine, that is, the machine used by the user to run deployment commands.

The client machine must meet the following requirements:

- network access to the helm repository and docker image repository.
- configured Helm repository
- network access to the Kubernetes cluster.
- required environment settings to run the `kubectl`, `podman`, and `docker` commands. The environment should have privileges to create namespace in the Kubernetes cluster.
- The Helm client installed with the **push** plugin. Configure the environment in such a manner that the `helm install` command deploys the software in the Kubernetes cluster.

Server or Space Requirements

For information on the server or space requirements for installing OCNADD, see the following documents:

- *Oracle Communications Cloud Native Core, Cloud Native Environment Installation, Upgrade, and Fault Recovery Guide*
- *Oracle Communications Network Analytics Data Director Benchmarking Guide*
- *Oracle Communications Cloud Native Core, cnDBTier Installation, Upgrade, and Fault Recovery Guide*

cnDBTier Requirement

> **Note**
>
> Obtain the values of the `cnDBTier` parameters listed in the section "cnDBTier Customization Parameters" from the delivered `ocnadd_dbtier_custom_values.yaml` file and use these values in the new `ocnadd_dbtier_custom_values.yaml` file if the parameter values in the new file differ from those in the delivered file.
>
> If you already have an older version of `cnDBTier`, upgrade `cnDBTier` with resources recommended for OCNADD by customizing the `ocnadd_dbtier_custom_values.yaml` file in the `custom-templates` folder of the OCNADD package with the required deployment parameters. Use the same PVC size as in the previous release. For more information, see the section "cnDBTier Customization Parameters."

OCNADD supports cnDBTier 25.2.1xx, 25.1.2xx, and 25.1.1xx in a CNE environment. cnDBTier must be up and running before installing the Data Director. To install cnDBTier 25.2.1xx with resources recommended for OCNADD, customize the `ocnadd_dbtier_custom_values.yaml` file in the custom-templates folder in the OCNADD package with the required deployment parameters.

> **Note**
>
> The `ocnadd_dbtier_custom_values.yaml` file in the DD custom-templates.zip should normally correspond to the same version as the Data Director; however, it may be possible that the cnDBTier custom values belong to a different version than the Data Director. In this case, the `global.version` parameter from the `ocnadd_dbtier_custom_values.yaml` should be checked, and the corresponding GA package of cnDBTier should be used for the installation or upgrade of cnDBTier before installing/upgrading the Data Director package.

cnDBTier parameters for the Data Director may vary. For more information, see section cnDBTier Customization Parameters**.**

For more information about the cnDBTier installation procedure, see *Oracle Communications Cloud Native Core, cnDBTier Installation, Upgrade, and Fault Recovery Guide*.

> **Note**
>
> For OCI Environment, use the `StorageClass` as `oci-bv` in DBTier charts. To find the storage class name, run the below command:
>
> ```
> kubectl get sc -n <namespace>
> ```

2.1.3 Resource Requirements

This section describes the resource requirements to install and run Oracle Communications Network Analytics Data Director (OCNADD).

OCNADD supports centralized deployment, where each data director site has been logically replaced by a worker group. The deployment consists of a management group and multiple worker groups. Traffic processing services are managed within the worker group, while configuration and administration services are managed within the management group.

In the case of centralized deployment, resource planning should consider the following points:

- There will be only one management group consisting of the following services:
  - ocnaddconfiguration
  - ocnaddalarm
  - ocnaddhealthmonitoring
  - ocnaddgui
  - ocnadduirouter
  - ocnaddredundancyagent
  - ocnaddexport
- There can be one or more worker groups managed by the single management group and each worker group logically depicts the standalone data director site w.r.t traffic processing function. This includes the following services:
  - ocnaddkafka
  - kraft-controller
  - ocnaddnrfaggregation
  - ocnaddseppaggregation
  - ocnaddscpaggregation
  - ocnaddcorrelation
  - ocnaddfilter
  - ocnaddadmin
  - ocnaddconsumeradapter
  - ocnaddstorageadapter
  - ocnaddpcfaggregation
  - ocnaddbsfaggregation
- The customer needs to plan for the resources corresponding to the management group and the number of worker groups required.

OCNADD supports various other deployment models. Before finalizing the resource requirements, see the OCNADD Deployment Models section. The resource usage and available features vary based on the deployment model selected. The centralized deployment model is the default model from 23.4.0 onward in the fresh installation with one management group and at least default worker group.

OCNADD Resource Requirements

Table 2-4 OCNADD Resource Requirements(Bases on HTTP2 Data Feed)

OCNADD Services	vCPU Req	vCPU Limit	Memory Req (Gi)	Memory Limit (Gi)	Min Replica	Max Replica	Partitions	Topic Name
ocnaddconfiguration	1	1	1	1	1	1	-	-
ocnaddalarm	1	1	1	1	1	1	-	-
ocnaddadmin	1	1	1	1	1	1	-	-
ocnaddhealthmonitoring	1	1	1	1	1	1	-	-
ocnaddscpaggregation	2	2	2	2	1	3	18	SCP
ocnaddnrfaggregation	2	2	2	2	1	1	6	NRF
ocnaddseppaggregation	2	2	2	2	1	2	12	SEPP
ocnaddadapter	3	3	4	4	2	14	126	MAIN
ocnaddkafka	6	6	64	64	4	4	-	-
zookeeper	1	1	2	2	3	3	-	-
kraftcontroller	1	1	2	2	3	3	-	-
ocnaddgui	1	2	1	1	1	2	-	-
ocnadduirouter	1	2	1	1	1	2	-	-
ocnaddcorrelation	3	3	24	64	1	4	-	-
ocnaddfilter	2	2	3	3	1	4	-	-
ocnaddredundancyagent	1	1	3	3	1	1	-	-
ocnaddstorageadapter	3	3	24	64	1	4	-	-
ocnaddexport	2	4	4	64	1	2	-	-
ocnaddingressadapter	3	3	8	8	1	7	-	-
ocnaddnonoracleaggregation	2	2	2	2	1	1	-	-
ocnaddpcfaggregation	2	2	2	2	1	2	12	PCF
ocnaddbsfaggregation	2	2	2	2	1	2	6	BSF

> **Note**
>
> For detailed information on the OCNADD profiles, see the "Profile Resource Requirements" section in the *Oracle Communications Network Analytics Data Director Benchmarking Guide*.

Ephemeral Storage Requirements

Table 2-5 Ephemeral Storage

Service Name	Ephemeral Storage (min) in Mi	Ephemeral Storage (max) in Mi
<app-name>-adapter	200	800
ocnaddadminservice	100	200
ocnaddalarm	100	500
ocnaddhealthmonitoring	100	500

Table 2-5 (Cont.) Ephemeral Storage

Service Name	Ephemeral Storage (min) in Mi	Ephemeral Storage (max) in Mi
ocnaddscpaggregation	100	500
ocnaddseppaggregation	100	500
ocnaddnrfaggregation	100	500
ocnaddconfiguration	100	500
ocnaddcorrelation	100	500
ocnaddfilter	100	500
ocnaddredundancyagent	100	500
ocnaddstorageadapter	400	800
ocnaddexport	100	2Gi
ocnaddingressadapter	400	800
ocnaddnonoracleaggregation	100	500
ocnaddpcfaggregation	100	500
ocnaddbsfaggregation	100	500

2.2 Installation Sequence

This section provides information on how to install Oracle Communications Network Analytics Data Director (OCNADD).

Note

- It is recommended to follow the steps in the given sequence for preparing and installing OCNADD.
- Make sure you have the required software installed before proceeding with the installation.
- This is the installation procedure for a standard OCNADD deployment. To install a more secure deployment (such as, adding users, changing password, enabling mTLS, and so on) see, *Oracle Communications Network Analytics Suite Security Guide*.

2.2.1 Pre-Installation Tasks

To install OCNADD, perform the preinstallation steps described in this section.

Note

The `kubectl` commands may vary based on the platform used for deploying OCNADD. Users are recommended to replace `kubectl` with environment-specific command line tool to configure Kubernetes resources through kube-api server. The instructions provided in this document are as per the OCCNE's version of kube-api server.

2.2.1.1 Downloading OCNADD Package

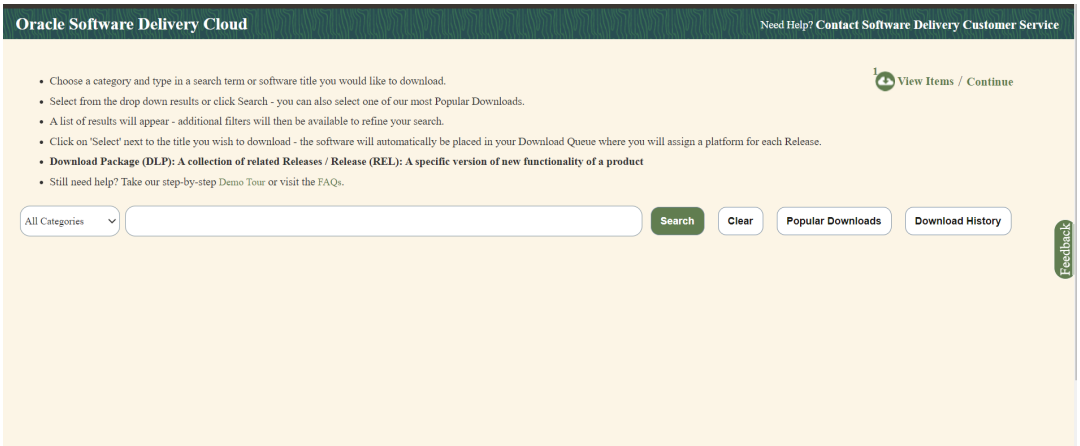
To download the Oracle Communications Network Analytics Data Director (OCNADD) package from MOS, perform the following steps:

1. Log in to My Oracle Support with your credentials.
2. Select the **Patches and Updates** tab to locate the patch.
3. In the **Patch Search** window, click **Product or Family (Advanced)**.
4. Enter "Oracle Communications Network Analytics Data Director" in the **Product** field, select "Oracle Communications Network Analytics Data Director 25.2.101.0.0 from **Release** drop-down list.
5. Click **Search**. The **Patch Advanced Search Results** displays a list of releases.
6. Select the required patch from the search results. The Patch Details window opens.
7. Click **Download**. File Download window appears.
8. Click the **<p********_<release_number>_Tekelec>.zip** file to download the OCNADD package file.
9. Extract the zip file to download the network function patch to the system where the network function must be installed.

To download the Oracle Communications Network Analytics Data Director package from the edelivery portal, perform the following steps:

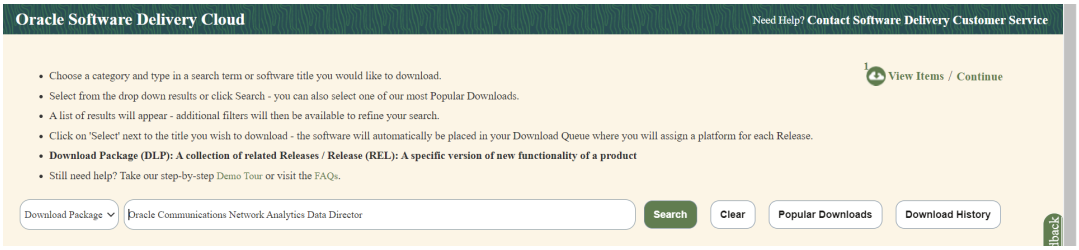
1. Login to the edelivery portal with your credentials. The following screen appears:

Figure 2-1 edelivery portal



2. Select the **Download Package** option, from **All Categories** drop down list.
3. Enter **Oracle Communications Network Analytics Data Director** in the search bar.

Figure 2-2 Search



4. List of release packages available for download are displayed on the screen. Select the release package you want to download, the package automatically gets downloaded.

2.2.1.2 Pushing the Images to Customer and OCI Registry

Container Images

> **⚠ Caution**
>
> 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 Core, Cloud Native Environment (CNE) version of kube-api server.

Oracle Communications Network Analytics Data Director (OCNADD) deployment package includes ready-to-use container images and helm charts to help orchestrate containers in Kubernetes. The communication between Pods of services of OCNADD are preconfigured in the helm charts.

Following table lists the container images of OCNADD:

Table 2-6 Container Images for OCNADD

Service Name	Container Image Name	Image Tag
OCNADD-Configuration	ocnaddconfiguration	25.2.101
OCNADD-ConsumerAdapter	<app-name>-adapter	25.2.101
OCNADD-Aggregation	ocnaddnrfaggregation ocnaddscpaggregation ocnaddseppaggregation ocnaddnonoracleaggregation ocnaddpcfaggregation ocnaddbsfaggregation	25.2.101
OCNADD-Alarm	ocnaddalarm	25.2.101
OCNADD-HealthMonitoring	ocnaddhealthmonitoring	25.2.101
OCNADD-Kafka	kafka-broker-x	4.0.0:25.2.101
OCNADD-Admin	ocnaddadminservice	25.2.101
OCNADD-UIRouter	ocnadduirouter	25.2.101
OCNADD-GUI	ocnaddgui	25.2.101

Table 2-6 (Cont.) Container Images for OCNADD

Service Name	Container Image Name	Image Tag
OCNADD-Backup-Restore	ocnaddbackuprestore	25.2.101
OCNADD-Filter	ocnaddfilter	25.2.101
OCNADD-Correlation	ocnaddcorrelation	25.2.101
OCNADD-Redundancyagent	ocnaddredundancyagent	25.2.101
OCNADD-StorageAdapter	ocnaddstorageadapter	25.2.101
OCNADD-Export	ocnaddexport	25.2.101
OCNADD-IngressAdapter	ocnaddingressadapter	25.2.101

Note

- The service image names are prefixed with the OCNADD release name.
- The above table depicts the default OCNADD microservices and their respective images. However, a few more necessary images are delivered as a part of the OCNADD package, make sure to push all the images delivered with the package.

Pushing OCNADD Images to Customer Registry

To push the images to the registry:

1. Untar the OCNADD package zip file to retrieve the OCNADD docker image tar file:

```
tar -xvzf ocnadd_pkg_25_2_101.tar.gz

cd ocnadd_pkg_25_2_101

tar -xvzf ocnadd-25.2.101.tar.gz
```

   The directory consists of the following:

   - **OCNADD Docker Images File:**

     `ocnadd-images-25.2.101.tar`

   - **Helm File:**

     `ocnadd-25.2.101.tgz`

   - **Readme txt File:**

     `Readme.txt`

   - **Custom Templates:**

     `custom-templates.zip`

- **ssl_certs folder:**

  ```
  ssl_certs
  ```

2. Run one of the following commands to first change the directory and then load the `ocnadd-images-25.2.101.tar` file:

   ```
   cd ocnadd-package-25.2.101
   ```

   ```
   docker load --input /IMAGE_PATH/ocnadd-images-25.2.101.tar
   ```

   ```
   podman load --input /IMAGE_PATH/ocnadd-images-25.2.101.tar
   ```

3. Run one of the following commands to verify if the images are loaded:

   ```
   docker images
   ```

   ```
   podman images
   ```

   Verify the list of images shown in the output with the list of images shown in the table Table 2-6. If the list does not match, reload the image tar file.

4. Run one of the following commands to tag each imported image to the registry:

   ```
   docker tag <image-name>:<image-tag> <docker-repo>/<image-name>:<image-tag>
   ```

   ```
   podman tag <image-name>:<image-tag> <docker-repo>/<image-name>:<image-tag>
   ```

5. Run one of the following commands to push the image to the registry:

   ```
   docker push <docker-repo>/<image-name>:<image-tag>
   ```

   ```
   podman push <podman-repo>/<image-name>:<image-tag>
   ```

   > **Note**
   >
   > It is recommended to configure the docker certificate before running the push command to access customer registry through HTTPS, otherwise, docker push command may fail.

6. Run the following command to push the helm charts to the helm repository:

   ```
   helm push <image_name>.tgz <helm_repo>
   ```

7. Run the following command to extract the helm charts:

   ```
   tar -xvzf ocnadd-25.2.101.tgz
   ```

8. Run the following command to unzip the custom-templates.zip file.

   ```
   unzip custom-templates.zip
   ```

Pushing OCNADD Images to OCI Registry

To push the images to the registry:

1. Untar the OCNADD package zip file to retrieve the OCNADD docker image tar file:

   ```
   tar -xvzf ocnadd_pkg_25_2_101.tar.gz
   cd ocnadd_pkg_25_2_101
   tar -xvzf ocnadd-25.2.101.tar.gz
   ```

   The directory consists of the following:

   - **OCNADD Docker Images File:**

     `ocnadd-images-25.2.101.tar`

   - **Helm File:**

     `ocnadd-25.2.101.tgz`

   - **Readme txt File:**

     `Readme.txt`

   - **Custom Templates:**

     `custom-templates.zip`

   - **ssl_certs folder:**

     `ssl_certs`

2. Run one of the following commands to first change the directory and then load the `ocnadd-images-25.2.101.tar` file:

   ```
   cd ocnadd-package-25.2.101
   ```

   ```
   docker load --input /IMAGE_PATH/ocnadd-images-25.2.101.tar
   ```

   ```
   podman load --input /IMAGE_PATH/ocnadd-images-25.2.101.tar
   ```

3. Run one of the following commands to verify if the images are loaded:

```
docker images
```

```
podman images
```

Verify the list of images shown in the output with the list of images shown in the table Table 2-6. If the list does not match, reload the image tar file.

4. Run the following commands to log in to the OCI registry:

```
docker login -u <REGISTRY_USERNAME> -p <REGISTRY_PASSWORD> <REGISTRY_NAME>
```

```
podman login -u <REGISTRY_USERNAME> -p <REGISTRY_PASSWORD> <REGISTRY_NAME>
```

```
# It will ask for password
# Enter the password generated while creating the auth token.
```

Where,

- `REGISTRY_NAME` is `<Region_Key>.ocir.io`
- `REGISTRY_USERNAME` is `<Object Storage Namespace>/<identity_domain>/email_id`
- `REGISTRY_PASSWORD` is the Authtocken generated by the user.

For the details about the Region Key, refer to Regions and Availability Domains.

Identity Domain will the domain,to which the user is present.

Object Storage Namespace is available at OCI Console> Governanace & Administration> Account Management> Tenenancy Details> Object Storage Namespace.

5. Run one of the following commands to tag each imported image to the registry:

```
docker tag <image-name>:<image-tag> <docker-repo>/<image-name>:<image-tag>
```

```
podman tag <image-name>:<image-tag> <docker-repo>/<image-name>:<image-tag>
```

6. Run one of the following commands to push the image to the registry:

```
docker push <region>/<tenancy_namespace>/<repo-name>/<image-name>:<image-
tag>
```

```
podman push <region>/<tenancy_namespace>/<repo-name>/<image-name>:<image-
tag>
```

> **Note**
>
> It is recommended to configure the docker certificate before running the push command to access OCI registry through HTTPS, otherwise, docker push command may fail.

7. Run the following command to push the helm charts to the helm repository:

   ```
   helm push <image_name>.tgz <helm_repo>
   ```

8. Run the following command to extract the helm charts:

   ```
   tar -xvzf ocnadd-25.2.101.tgz
   ```

9. Run the following command to unzip the custom-templates.zip file.

   ```
   unzip custom-templates.zip
   ```

> **Note**
>
> All the image repositories must be public. Run the following steps to make all image repositories public:
>
> 1. Go to **OCI Console> Developer Services > Containers & Artifacts> Container Registry**.
> 2. Select the root Compartment.
> 3. In the **Repositories and Images Search** option, the images will be listed. Select each image and click **Change to Public**. This step must be performed for all the images sequentially.

2.2.1.3 Creating OCNADD Namespace

This section explains how to verify or create new namespaces in the system. In this section, the namespaces for the management group and worker group should be created.

To verify if the required namespace already exists in the system, run the following command:

```
kubectl get namespaces
```

If the namespace exists, you may continue with the next steps of installation.

If the required namespace is not available, create a namespace using the following command:

> **Note**
>
> The user must create the required namespaces for a centralized deployment with multiple worker groups. If the deployment mode is centralized with the default worker group, a single namespace is sufficient, and all Data Director services can be deployed within it

Run the following command to create a management group namespace where all the management group services will be deployed:

```
kubectl create namespace <managementgroup namespace>
```

> **Note**
>
> - If using centralized mode with only the default worker group, this namespace is sufficient, and there is no need to create child namespaces. You can skip the remaining steps.

Run the following command to create workergroup namespace where all the worker group services will be deployed:

```
kubectl create namespace <workergroup-namespace>
```

For example:

```
kubectl create namespace dd-mgmt-group


kubectl create namespace dd-worker-group1
```

Run the following command to verify the namespaces are created:

```
kubectl get namespaces
```

For example:

```
# kubectl get namepsaces
  dd-mgmt-group
  dd-worker-group1
```

Naming Convention for Namespaces

While choosing the name of the namespace where you wish to deploy OCNADD, make sure the following requirements are met:

- starts and ends with an alphanumeric character
- contains 63 characters or less
- contains only alphanumeric characters or '-'

> **Note**
>
> It is recommended to avoid using prefix `kube-` when creating namespace. This is required as the prefix is reserved for Kubernetes system namespaces.

2.2.1.4 Creating Service Account, Role, and Role Binding

This section is optional and it describes how to manually create a service account, role, and rolebinding. It is required only when customer needs to create a role, rolebinding, and service account manually before installing OCNADD. Skip this if choose to create by default from helm charts.

In the case of centralized deployment, this procedure needs to be repeated for each of the management group and worker group(s).

> **Note**
>
> The secret(s) should exist in the same namespace where OCNADD is getting deployed. This helps to bind the Kubernetes role with the given service account.

Creating Service Account, Role, and RoleBinding for Management Group

To create the service account, role, and rolebinding:

1. **Prepare OCNADD Management Group Resource File:**
   - Run the following command to create an OCNADD resource file specifically for the management group:

     ```
     vi <ocnadd-mgmt-resource-file>.yaml
     ```

     Replace `<ocnadd-mgmt-resource-file>` with the required name for the management group resource file.
   - For example:

     ```
     vi ocnadd-mgmt-resource-template.yaml
     ```

2. **Update OCNADD Management Group Resource Template:**
   - Update the `ocnadd-mgmt-resource-template.yaml` file with release-specific information.

     > **Note**
     >
     > Replace <custom-name> and <namespace> with their respective OCNADD management group namespace. Use a custom name preferably similar to the management namespace name to avoid upgrade issues.

   - A sample template to update the ocnadd-mgmt-resource-template.yaml file with is given below:

     ```
     # # Sample template start #
     apiVersion: v1
     kind: ServiceAccount
     metadata:
         name: < custom - name > -sa - ocnadd
     namespace: < namespace >
         automountServiceAccountToken: false

         -- -
         apiVersion: rbac.authorization.k8s.io / v1
     kind: Role
     metadata:
         name: < custom - name > -cr
     rules:
     ```

```
    -apiGroups: [""]
resources: ["pods", "configmaps", "services", "secrets",
"resourcequotas", "events", "persistentvolumes",
"persistentvolumeclaims"]
verbs: ["*"] -
    apiGroups: ["extensions"]
resources: ["ingresses"]
verbs: ["create", "get", "delete"] -
    apiGroups: [""]
resources: ["nodes"]
verbs: ["get"] -
    apiGroups: ["scheduling.volcano.sh"]
resources: ["podgroups", "queues", "queues/status"]
verbs: ["get", "list", "watch", "create", "delete", "update"]

    -- -
    apiVersion: rbac.authorization.k8s.io / v1
kind: RoleBinding
metadata:
    name: < custom - name > -crb

roleRef:
    apiGroup: ""
kind: Role
name: < custom - name > -cr
subjects:
    -kind: ServiceAccount
name: < custom - name > -sa - ocnadd
namespace: < namespace >

    -- -
    apiVersion: rbac.authorization.k8s.io / v1
kind: RoleBinding
metadata:
    name: < custom - name > -crb - policy
roleRef:
    apiGroup: ""
kind: ClusterRole
name: psp: privileged
subjects:
    -kind: ServiceAccount
name: < custom - name > -sa - ocnadd
namespace: < namespace >
    -- -
    # # Sample template end #
```

3. **Create Service Account, Role, and RoleBinding:**
   - Run the following command to create the service account, role, and rolebinding for the management group:

     ```
     kubectl -n <dd-mgmt-group-namespace> create -f ocnadd-mgmt-resource-
     template.yaml
     ```

     Replace `<dd-mgmt-group-namespace>` with the namespace where the OCNADD management group will be deployed.

- For example:

```
$ kubectl -n dd-mgmt-group create -f ocnadd-mgmt-resource-template.yaml
```

> **Note**
>
> - Update the custom values file `ocnadd-custom-values-25.2.101-mgmt-group.yaml` created/copied from `ocnadd-custom-values-25.2.101.yaml` in the "Custom Templates" folder.
> - Change the following parameters to `false` in `ocnadd-custom-values-25.2.101-mgmt-group.yaml` after adding the global service account to the management group. Failing to do so might result in installation failure due to CRD creation and deletion:
>
> ```
> serviceAccount:
>     create: false
>     name: <custom-name>            ## --> Change this to <custom-
> name> provided in ocnadd-mgmt-resource-template.yaml above ##
>     upgrade: false
> clusterRole:
>     create: false
>     name: <custom-name>            ## --> Change this to <custom-
> name> provided in ocnadd-mgmt-resource-template.yaml above ##
>     clusterRoleBinding:
>     create: false
>     name: <custom-name>            ## --> Change this to <custom-
> name> provided in ocnadd-mgmt-resource-template.yaml above ##
> ```
>
> - Ensure the namespace used in `ocnadd-mgmt-resource-template.yaml` matches the below parameters in `ocnadd-custom-values-25.2.101-mgmt-group.yaml`:
>
> ```
> global.deployment.management_namespace
> global.cluster.nameSpace.name
> ```

Creating Service Account, Role, and RoleBinding for Worker Group

Run the following command to create the service account, role, and rolebinding:

> **Note**
>
> Repeat the below procedure for each of the worker groups that needs to be added to the centralized deployment.

1. **Prepare OCNADD Worker Group Resource File:**
   - Run the following command to create an OCNADD resource file specifically for the worker group:

```
vi <ocnadd-wg1-resource-file>.yaml
```

Replace <ocnadd-wg1-resource-file> with the required name for the worker group resource file.

- For example:

```
vi ocnadd-wg1-resource-template.yaml
```

2. **Update OCNADD Worker Group Resource Template:**
   - Update the ocnadd-wg1-resource-template.yaml file with release-specific information.

> **Note**
>
> Replace <custom-name> and <namespace> with their respective OCNADD worker group namespace. Use a custom name preferably similar to the worker group namespace name to avoid upgrade issues.

   - A sample template to update the ocnadd-wg1-resource-template.yaml file with is given below:

```
## Sample template start#
apiVersion: v1
kind: ServiceAccount
metadata:
  name: <custom-name>-sa-ocnadd
  namespace: <namespace>
automountServiceAccountToken: false

---
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: <custom-name>-cr
rules:
- apiGroups: [""]
  resources: ["pods","configmaps","services",
"secrets","resourcequotas","events","persistentvolumes","persistentvolum
eclaims"]
  verbs: ["*"]
- apiGroups: ["extensions"]
  resources: ["ingresses"]
  verbs: ["create", "get", "delete"]
- apiGroups: [""]
  resources: ["nodes"]
  verbs: ["get"]
- apiGroups: ["scheduling.volcano.sh"]
  resources: ["podgroups", "queues", "queues/status"]
  verbs: ["get", "list", "watch", "create", "delete", "update"]

---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: <custom-name>-crb
```

```
roleRef:
  apiGroup: ""
  kind: Role
  name: <custom-name>-cr
subjects:
- kind: ServiceAccount
  name: <custom-name>-sa-ocnadd
  namespace: <namespace>

---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: <custom-name>-crb-policy
roleRef:
  apiGroup: ""
  kind: ClusterRole
  name: psp:privileged
subjects:
- kind: ServiceAccount
  name: <custom-name>-sa-ocnadd
  namespace: <namespace>
---
## Sample template end#
```

3. **Create Service Account, Role, and RoleBinding:**
   - Run the following command to create the service account, role, and rolebinding for the worker group:

     ```
     kubectl -n <dd-worker-group-namespace> create -f ocnadd-wg1-resource-template.yaml
     ```

     Replace `<dd-worker-group-namespace>` with the namespace where the OCNADD worker group will be deployed.
   - For example:

     ```
     $ kubectl -n dd-worker-group1 create -f ocnadd-wg1-resource-template.yaml
     ```

Note

- Update the custom values file `ocnadd-custom-values-25.2.101-worker-group1.yaml` created/copied from `ocnadd-custom-values-25.2.101.yaml` in the "Custom Templates" folder.
- Change the following parameters to `false` in `ocnadd-custom-values-25.2.101-worker-group1.yaml` after adding the global service account to the worker group. Failing to do so might result in installation failure due to CRD creation and deletion:

```
serviceAccount:
    create: false
    name: <custom-name> ## --> Change this to <custom-name>
provided in ocnadd-wg1-resource-template.yaml above ##
    upgrade: false
clusterRole:
    create: false
    name: <custom-name> ## --> Change this to <custom-name>
provided in ocnadd-wg1-resource-template.yaml above ##
    clusterRoleBinding:
    create: false
    name: <custom-name> ## --> Change this to <custom-name>
provided in ocnadd-wg1-resource-template.yaml above ##
```

- Ensure the namespace used in `ocnadd-wg1-resource-template.yaml` matches the below parameters in `ocnadd-custom-values-25.2.101-worker-group1.yaml`:

  `global.cluster.nameSpace.name`

  The `management_namespace` parameter is set to the namespace used for the management group.

  `global.deployment.management_namespace`

2.2.1.5 Configuring OCNADD Database

OCNADD microservices use MySQL database to store the configuration and run time data.

The database is managed by the helm pre-install hook. However, OCNADD requires the database administrator to create an admin user in MySQL database and provide the necessary permissions to access the databases. Before installing OCNADD it is required to create the MySQL user and databases.

> **Note**
>
> - If the admin user is already available, then update the credentials, such as username and password (base64 encoded) in `ocnadd/templates/ocnadd-secret-hook.yaml`.
> - If the admin user is not available, then create it using the following procedure. Once the user is created, update the credentials for the user in `ocnadd/templates/ocnadd-secret-hook.yaml`.

Creating an Admin User in the Database

To create an admin user in the database:

1. Run the following command to access the MySQL pod:

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

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

2. Run the following command to login to MySQL server using MySQL client:

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

3. To create an admin user, run the following command:

   ```
   CREATE USER IF NOT EXISTS'<ocnadd admin username>'@'%' IDENTIFIED BY
   '<ocnadd admin user password>';
   ```

   Example:

   ```
   CREATE USER IF NOT EXISTS 'ocdd'@'%' IDENTIFIED BY 'ocdd';
   ```

   Where:

   <ocdd> is the admin username and <ocdd> is the password for MySQL admin user

4. Run the following command to grant the necessary permissions to the admin user and run the FLUSH command to reload the grant table:

   ```
   GRANT ALL PRIVILEGES ON *.* TO 'ocdd'@'%' WITH GRANT OPTION;


   FLUSH PRIVILEGES;
   ```

5. Access the `ocnadd-secret-hook.yaml` from the OCNADD helm files using the following path:

   ```
   ocnadd/templates/ocnadd-secret-hook.yaml
   ```

6. Update the following parameters in the `ocnadd-secret-hook.yaml` with the admin user credentials:

   ```
   data:
   MYSQL_USER: b2NkZA==
   MYSQL_ACCESS_KEY: b2NkZA==
   ```

   To generate the base64 encoded user and password from the terminal, run the following command:

   ```
   echo -n <string> | base64 -w 0
   ```

   Where, `<string>` is the admin username or password created in step3.

   For example:

   ```
   echo -n ocdd | base64 -w 0
   b2NkZA==
   ```

Update Database Name

> **Note**
>
> - By default, the database names are configuration_schema, alarm_schema, and healthdb_schema for the respective services.
> - Skip this step if you plan to use the default database names during database creation. If not, change the database names as required.

To update the database names in the Configuration Service, Alarm Service, and Health Monitoring services:

1. Access the `ocdd-db-resource.sql` file from the helm chart using the following path:

   ```
   ocnadd/ocdd-db-resource.sql
   ```

2. Update all occurrences of the database name in `ocdd-db-resource.sql`.

> **Note**
>
> During the OCNADD reinstallation, all three application databases must be removed manually by running the `drop database <dbname>;` command.

2.2.1.6 Configuring Secrets for Accessing OCNADD Database

The secret configuration for OCNADD database is automatically managed during the database creation the helm preinstall procedure.

2.2.1.7 Configuring IP Network

This section defines OCNADD IP configuration for single stack (either only IPv4 or IPv6) or dual stack supported infrastructure.

- For IPv4 network, update the following parameters in `ocnadd-custom-values-25.2.101.yaml`:

```
global:
    ipConfigurations:
        ipFamilyPolicy: SingleStack
        ipFamilies: ["IPv4"]
```

- For IPv6 network, update the following parameters in `ocnadd-custom-values-25.2.101.yaml`:

```
global:
    ipConfigurations:
        ipFamilyPolicy: SingleStack
        ipFamilies: ["IPv6"]
```

> **Note**
>
> - The primary IP family remains fixed once OCNADD is deployed. To change the primary IP family, OCNADD needs to be redeployed.
> - The IPv6 support on OCI is not available in the 25.2.101 Release

2.2.1.8 Configuring SSL or TLS Certificates

Extract the Package

OCNADD supports both TLS and non-TLS communication between its microservices for internal communication. If intra-TLS is enabled, all Data Director microservices must use their own TLS certificates. However, if intra-TLS is disabled, certain microservices or jobs still require the use of a TLS certificate. Detailed information about TLS communication in the Data Director is provided in the 'TLS Configuration' section of the *Oracle Communication Network Analytics Suite Security Guide*.

Before proceeding with the configuration of SSL/TLS certificates for OCNADD, see the 'Certificate and Secret Generation' section in the *Oracle Communication Network Analytics Suite Security Guide*.

If not already done, extract the package `occnadd-package-25.2.101.tgz`.

> **Note**
>
> - Before configuring the SSL/TLS certificates, see "Customizing CSR and Certificate Extensions" section in the *Oracle Communications Network Analytics Suite Security Guide*.
> - This procedure is mandatory, perform it before proceeding with the installation.

Before generating certificates using `cacert` and `cakey`, finalize the Kafka access mode. In step 7 of section, Generate Certificates using CACert and CAKey, provide the script response "y" while running the generate_certs script to create certificates.

The following access modes are available and applicable for worker groups only:

1. When the NF producers and OCNADD are in the same cluster with external access disabled.
2. When the NF producers and OCNADD are in different clusters with LoadBalancer.

> **Note**
>
> - If the NF Producers and OCNADD are deployed in the same cluster, all three ports can be used: 9092 for PLAIN_TEXT, 9093 for SSL, and 9094 for SASL_SSL. However, note that the 9092 port is non-secure and is not recommended for use.
> - If the NF Producers and OCNADD are deployed in different clusters, only the 9094 (SASL_SSL) port is exposed.
> - It is recommended to use individual server IPs in the Kafka bootstrap server list instead of a single service IP like "kafka-broker:9094".

NF producers and OCNADD are in the same cluster with external access disabled

In this mode, the Kafka cluster is not exposed externally. By default, the parameters `externalAccess.enabled` and `externalAccess.autoDiscovery` are set to `false`, therefore no change is needed. The parameters `externalAccess.enabled` and `externalAccess.autoDiscovery` are present in the `ocnadd-custom-values-25.2.101.yaml` file.

The default values of bootstrap-server are given below:

```
kafka-broker-0.kafka-broker-headless:9093
kafka-broker-1.kafka-broker-headless:9093
kafka-broker-2.kafka-broker-headless:9093
```

> **Note**
>
> Use the below FQDN as a bootstrap server if required when NFs deployed in same cluster:
> `kafka-broker-0.kafka-broker-headless.<namespace>.svc.<domain>:9093/9092`
>
> `kafka-broker-1.kafka-broker-headless.<namespace>.svc.<domain>:9093/9092`
>
> `kafka-broker-2.kafka-broker-headless.<namespace>.svc.<domain>:9093/9092`

The NF producers and OCNADD are in different clusters with LoadBalancer

If the NF producers and OCNADD are in different Clusters, then either the LoadBalancer or NodePort Service Type can be used. In both the cases, the IP addresses are required to be updated manually in the ssl_certs/default_values/values of kafka-broker section by using the following steps:

With LoadBalancer

> **Note**
>
> To enable external access in a CNLB-enabled OCCNE cluster, see the 'Enable CNLB External IPs in the Kafka Cluster' section in the *Oracle Communications Network Analytics Data Director User Guide*.

1. Update the following parameters in Kafka section of the `ocnadd-custom-values-25.2.101.yaml` file:
   a. `externalAccess.type` to LoadBalancer
   b. `externalAccess.enabled` to true
   c. `externalAccess.autoDiscovery` to true
   d. If the deployment is on the OCI platform, make sure to update the following parameters:
      i. Set `global.env.oci` to true.
      ii. Update `global.env.subnetOcid` to the specific `<subnet ocid value>`.
2. Update based on LoadBalance IP types as follows:
   a. **When Static LoadBalancer IPs are used**
      i. Update the following parameters in the Kafka section of the `ocnadd-custom-values-25.2.101.yaml` file:
         - `externalAccess.setstaticLoadBalancerIps` to '`true`'. Default is `false`.
         - Static IP list in "externalAccess.LoadBalancerIPList" separated with comma.

         For example:

```
externalAccess:
            setstaticLoadBalancerIps: true
            LoadBalancerIPList:
[10.20.30.40,10.20.30.41,10.20.30.42]
```

ii. While running the script, include all static IPs under the kafka-broker section. To do so, respond with "y" in step 7 of section, Generate Certificates using CACert and CAKey," while running the generate_certs script for certificate creation. Subsequently, add the IPs by selecting the service and entering the required values when prompted.

```
For the following services:
1. kafka-broker
2. ocnaddscpaggregation
3. ocnaddnrfaggregation
4. ocnaddseppaggregation
5. adapter
6. ocnaddfilter
7. ocnaddcorrelation
8. ocnaddstorageadapter
9. ocnaddingressadapter
10. kraft-controller
11. ocnaddnonoracleaggregation
12. ocnaddpcfaggregation
13. ocnaddbsfaggregation
14. ocnaddadminservice

Enter the number corresponding to the service for which you want to
add IP: 1
Please enter IP for the service kafka-broker or enter "n" to exit :
10.20.30.40
Please enter IP for the service kafka-broker or enter "n" to exit :
10.20.30.41
Please enter IP for the service kafka-broker or enter "n" to exit :
10.20.30.42
Please enter IP for the service kafka-broker or enter "n" to exit :
n
Do you want to add IP to any other service (y/n) : n
```

b. **When LoadBalancer IP CIDR block is used**

i. The LoadBalancer IP CIDR block should already be available during the site planning, if not available then contact the CNE infrastructure administrator to get the IP CIDR block for Loadbalancer IPs.

ii. Add all the available IPs under kafka-broker section while running the script. To do so, select "y" in step 7 of Generate Certificates using CACert and CAKey section while running the generate_certs script for creating certificates. Then add the IPs by selecting the service and entering the required IPs.
For example: For the worker-group1, if the available IP CIDR block is "10.x.x.0/26" with IP range is [1-62]

```
For the following services:
1. kafka-broker
2. ocnaddscpaggregation
3. ocnaddnrfaggregation
4. ocnaddseppaggregation
5. adapter
6. ocnaddfilter
7. ocnaddcorrelation
8. ocnaddstorageadapter
9. ocnaddingressadapter
```

```
10. kraft-controller
11. ocnaddnonoracleaggregation
12. ocnaddpcfaggregation
13. ocnaddbsfaggregation
14. ocnaddadminservice

Enter the number corresponding to the service for which you want to
add IP: 1
Please enter IP for the service kafka-broker or enter "n" to exit :
10.x.x.1
Please enter IP for the service kafka-broker or enter "n" to exit :
10.x.x.2
.
.
Please enter IP for the service kafka-broker or enter "n" to exit :
10.x.x.62
Please enter IP for the service kafka-broker or enter "n" to exit :
n
Do you want to add IP to any other service (y/n) : n
```

> **Note**
>
> The Kafka broker individual service FQDNs should be added in the DNS entry and also be used in the bootstrap server configuration for communication with Kafka.

2.2.1.8.1 Generate Certificates using CACert and CAKey

OCNADD allows the users to provide the CACert and CAKey and generate certificates for all the services by running a predefined script.

Use the ssl_cert folder to generate the certificates for Management Group or Worker Group namespaces accordingly.

To generate certificates using CACert and CAKey:

1. Navigate to the `<ssl_certs>/default_values` folder.

Note

<Optional> Users have the flexibility to modify the `service_values_template` file to add or remove specific service blocks for which certificates need to be created or removed.
For example, to generate certificates for the management group, users can edit the "`management_service_values_template`" file.

Similarly, depending on the deployment group type, users can edit the respective template file for that group.

```
Global Params:

[global]
countryName=<country>
stateOrProvinceName=<state>
localityName=<city>
organizationName=<org_name>
organizationalUnitName=<org_bu_name>
defaultDays=<days to expiry>

Root CA common name (e.g. rootca common_name=*.svc.domainName)
##root_ca

commonName=*.svc.domainName

Service common name for client and server and SAN(DNS/IP entries).
(Make sure to follow exact same format and provide an empty line at
the end of each service block)

[service-name-1]
client.commonName=client.cn.name.svc1
server.commonName=server.cn.name.svc1
IP.1=127.0.0.1
DNS.1=localhost

[service-name-2]
client.commonName=client.cn.name.svc2
server.commonName=server.cn.name.svc2
IP.1= 10.20.30.40 t
DNS.1 = *.svc2.namespace.svc.domainName
.
.
.
##end
```

2. Run the `generate_certs.sh` script with the following command:

   ```
   ./generate_certs.sh -cacert <path to>/CAcert.pem -cakey <path to>/CAkey.pem
   ```

   Where, `<path to>` is the folder path where the CACert and CAKey are present.

> **Note**
>
> In case the certificates are being generated for the worker group separately, then make sure the same CA certificate and private keys are used for generating the certificates as used for generating the management group certificates. The similar command as mentioned below can be used for the worker group certificate generation after the management group certificates have been generated:
>
> ```
> ./generate_certs.sh -cacert <path to>/cacert.pem -cakey <path to>/
> private/cakey.pem
> ```

3. Select the mode of deployment:

   ```
   "1" for non-centralized
   "2" for upgrade from non-centralized to centralized
   "3" for centralised
   "4" for simulator


   Select the mode of deployement (1/2/3) : 3
   ```

4. Select the namespace where you want to generate the certificates:

   ```
   Enter kubernetes namespace: <your_working_namespace>
   ```

5. Select the `service_values` file you would like to apply. Below example is for Management Group:

   ```
   Choose the group of services:
   1. management_group_services
   2. worker_group_services

   Choose a file by entering its corresponding number: (1 or 2) 1
   ```

6. Enter the domain name with which the user wants to change the default domain name(occne-ocdd) in chosen service_values file which will be used to create the certificate:

   ```
   Please enter the domain name: <domain_name>
   ```

7. Enter SAN (DNS/IP entries) for any service if required.

   ```
   Do you want to add any IP for adding SAN entries to existing dd services
   (y/n): y
   ```

   If the user selects "y," a list of services will be displayed, and the user can add Subject Alternative Name (SAN) entries for any of the listed services by choosing the corresponding service number.

   In the following example, a list of management services is presented to the user for adding SAN entries. Enter the number corresponding to the service for which the user wants to

input IP addresses. After selecting the service, provide the IP addresses as input. Enter "n" to exit if no further entries are needed.

```
For the following services:
1. ocnadduirouter
2. ocnaddalarm
3. ocnaddconfiguration
4. ocnaddhealthmonitoring
5. ocnaddbackuprestore
6. ocnaddredundancyagent
7. ocnaddexport


Enter the number corresponding to the service for which you want to add
IP: 3
Please enter IP for the service ocnaddalarm or enter "n" to exit :
10.20.30.40
Please enter IP for the service ocnaddalarm or enter "n" to exit :
10.20.30.41
Please enter IP for the service ocnaddalarm or enter "n" to exit : n
Do you want to add IP to any other service (y/n) : n
```

8. Select "y" when prompted to create CA.

   ```
   Do you want to create Certificate Authority (CA)? (y/n) y
   ```

9. Enter the passphrase for CAkey when prompted:

   ```
   Enter passphrase for CA Key file: <passphrase>
   ```

10. Select “y” when prompted to create CSR for each service:

    ```
    Create Certificate Signing Request (CSR) for each service? Y
    ```

11. Select “y” when prompted to sign CSR for each service with CA Key:

    ```
    Would you like to sign CSR for each service with CA key? Y
    ```

12. If the centralized mode of deployment is selected during the creation of management group certificates, once the management group certificate generation is completed, the user will be prompted to continue the certificate generation process for worker groups.

    ```
    Would you like to continue certificate creation for worker group? (y/n) y
    ```

    If "y" is selected, the script will execute to recreate the certificates for the worker group. The script will repeat its execution from step 4 onwards. During the worker group creation flow, choose "worker_group_service_values" in step 5 and proceed. If "n" is selected, the script completes its execution.

    > **Note**
    >
    > The script can be used to create both management certificates and the desired number of supported worker group certificates in a single execution.

13. Run the following command to check if the secrets are created in the specified namespace:

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

14. Run the following command to describe any secret created by script:

    ```
    kubectl describe secret <secret-name> -n <namespace>
    ```

2.2.1.8.2 Generating Certificate Signing Request (CSR)

Users can generate the certificate signing request for each of the services using the OCNADD script, and then can use the generated CSRs to generate the certificates using its own certificate signing mechanism (External CA server, Hashicorp Vault, and Venafi).

Perform the following procedure to generate the CSR:

1. Navigate to the `<ssl_certs>/default_values` folder.
2. Copy the required `service_values_template` (e.g., `management_service_values_template` or `worker_service_values_template` or `simulator_values_template` or `values_template` depending on the deployment mode) to another file named "backup_service_values_template."
3. Edit the corresponding `service_values_template` file and update global parameters, CN, and SAN (DNS/IP entries) for each service based on the provided requirements.
4. Change the default domain (`occne-ocdd`) and default namespace (`ocnadd-deploy`) in the corresponding `service_values_template` file with your cluster domain and namespace. Example:
   - For management group: `namespace = dd-mgmt-group`, `clusterDomain =cluster.local.com`

     ```
     sed -i "s/ocnadd-deploy/dd-mgmt-group/g"
     management_service_values_template
     sed -i "s/occne-ocdd/cluster.local.com/g"
     management_service_values_template
     ```

   - For worker group: `namespace = dd-worker-group`, `clusterDomain = cluster.local.com`

     ```
     sed -i "s/ocnadd-deploy/dd-worker-group/g"
     worker_service_values_template
     sed -i "s/occne-ocdd/cluster.local.com/g" worker_service_values_template
     ```

Note

Edit corresponding `service_values` file for global parameters, RootCA common name and Subject Alternative Name (SAN) keeping the service blocks of all the services for which the certificate needs to be generated..

```
Global Params:

[global]
countryName=<country>
stateOrProvinceName=<state>
localityName=<city>
organizationName=<org_name>
organizationalUnitName=<org_bu_name>
defaultDays=<days to expiry>


Root CA common name (e.g. rootca common_name=*.svc.domainName)
##root_ca

commonName=*.svc.domainName


Service common name for client and server and SAN(DNS/IP entries).
(Make sure to follow exact same format and provide an empty line at
the end of each service block)

[service-name-1]
client.commonName=client.cn.name.svc1
server.commonName=server.cn.name.svc1
IP.1=127.0.0.1
DNS.1=localhost

[service-name-2]
client.commonName=client.cn.name.svc2
server.commonName=server.cn.name.svc2
IP.1= 10.20.30.40
DNS.1 = *.svc2.namespace.svc.domainName
.
.
.
##end
```

5. Run the `generate_certs.sh` script with the `--gencsr` or `-gc` flag.

```
./generate_certs.sh --gencsr
```

6. Select the deployment mode.

```
(1) non-centralized
(2) upgrade from non-centralized to centralized
```

```
(3) centralised
(4) simulator


Select the mode of deployement (1/2/3/4) : 3
```

7. Select the namespace where you would like to generate the certificates:

   ```
   Enter kubernetes namespace: <your_working_namespace>
   ```

8. Select the group of services you would like to apply.

   ```
   Choose the group of services:
   1. management_group_services
   2. worker_group_services
   ```

9. Once the service CSRs are generated the `demoCA` folder will be created. Navigate to CSR and keys in the `demoCA/dd_mgmt_worker_services/<your_namespace>/services` (separate for client and server). The CSR can be signed using your own certificate signing mechanism to generate the certificates.
10. Make sure that the certificates and key names are created in the following format based on the service is acting as a client or server.
    For Client *servicename-clientcert.pem* and *servicename-clientprivatekey.pem*

    For Server *servicename-servercert.pem* and *servicename-serverprivatekey.pem*
11. Once above certificates are generated by signing CSR with the Certificate Authority, copy those certificates in the respective `demoCA/dd_mgmt_worker_services/<your_namespace>/services` folder of each services.

    > **Note**
    >
    > - Make sure to use the same CA key for both management group and worker group(s)
    > - Make sure the certificates are copied in the respective folders for the client and the server based on their generated CSRs

12. Run `generate_certs.sh` with the `cacert` path and `--gensecret` or `-gs` to generate secrets:

    ```
    ./generate_certs.sh -cacert <path to>/cacert.pem --gensecret
    ```

13. Select the namespace where you would like to generate the certificates:

    ```
    Enter kubernetes namespace: <your_working_namespace>
    ```

14. Select “y” when prompted to generate secrets for the services:

    ```
    Would you like to continue to generate secrets? (y/n) y
    ```

15. Run the following command to check if the secrets are created in the specified namespace:

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

16. Run the following command to describe any secret created by the script:

    ```
    kubectl describe secret <secret-name> -n <namespace>
    ```

17. Remove the corresponding service_values_template file once its use is completed after creating new secrets. Rename "`backup_service_values_template`" back to the corresponding service_values_template file.

2.2.1.9 OCCM Prerequisites for Installing OCNADD

Before starting the installation of OCNADD, ensure the following conditions are met regarding the Oracle Communication Certificate Manager (OCCM) installation:

- OCCM must be installed with the necessary permissions to create secrets in the OCNADD Management and Worker Group namespaces. If OCCM is deployed in a separate namespace, it must have at least sufficient privileges to create secrets in both the OCNADD Management and Worker Group namespaces. For more information on deployment models, refer to the OCCM Deployment Models section in the *Oracle Communications Certificate Manager Installation, Upgrade, and Fault Recovery Guide*.
- Issuer (CA) should be configured in OCCM. If multiple OCCMs are used, each should have at least one common issuer (CA) configuration.
- Ensure that OCCM has sufficient capacity to create the required number of certificates. If all features in OCNADD are enabled, 18 certificates are required for management services and 24 certificates are required for each worker group.
- The `cncc-api-access` client should be enabled in the Oracle Communications Cloud Native Configuration Console (CNC Console). For more information, see "Generate Access Tokens" section in *Oracle Communications Cloud Native Configuration Console User Guide*.

OCCM Secrets

Three secrets need to be created in every OCNADD namespace to use OCCM for creating certificates.

- **occm_secret**: This secret should contain the username and password of a CNC Console user with OCCM_READ and OCCM_WRITE roles. This user will be used by OCNADD to communicate with OCCM through CNC Console.

  ```
  $ kubectl create secret generic -n <ocnadd-namespace> --from-
  literal=username=<cncc-user>  --from-literal=password=<cncc-password> occm-
  secret
  ```

  Where:

  - `<ocnadd-namespace>`: OCNADD management or worker group namespace
  - `<cncc-user>`: CNC Console username of the CNC Console user
  - `<cncc-password>`: CNC Console password of the CNC Console user
  - `occm-secret`: Name of the secret storing the credentials of the CNC Console user

For example (for management and one worker group):

```
kubectl create secret generic -n ocnadd-deploy-mgmt --from-
literal=username=occm-cncc  --from-literal=password=occm-cncc-secret occm-
secret
kubectl create secret generic -n ocnadd-deploy-wg1 --from-
literal=username=occm-cncc  --from-literal=password=occm-cncc-secret occm-
secret
```

- **truststore_keystore_secret**: This secret should contain the key used for encrypting the Keystore and Truststore created by OCNADD to store the x509 certificates of each service and the CA.

  ```
  $ kubectl create secret generic -n <ocnadd-namespace> --from-
  literal=keystorekey=<keystore-key> --from-
  literal=truststorekey=<truststore-key> occm-truststore-keystore-secret
  ```

  Where:

  - `<ocnadd-namespace>`: OCNADD management or worker group namespace
  - `<keystore-key>`: Encryption key used for securing the Keystore storing a Service's certificates and private key
  - `<truststore-key>`: Encryption key used for securing the Keystore storing the CA certificate/certificate chain
  - `occm-truststore-keystore-secret`: Name of the secret containing the Keystore and Truststore key

  For example (for management and one worker group):

  ```
  kubectl create secret generic -n ocnadd-deploy-mgmt --from-
  literal=keystorekey=keystorepassword --from-
  literal=truststorekey=truststorepassword occm-truststore-keystore-secret
  kubectl create secret generic -n ocnadd-deploy-wg1 --from-
  literal=keystorekey=keystorepassword --from-
  literal=truststorekey=truststorepassword occm-truststore-keystore-secret
  ```

- **occm_cacert**: This secret stores the CA certificate or CA certificate-chain of the Issuer configured in OCCM.

  ```
  $ kubectl create secret generic -n <ocnadd-namespace> --from-
  file=cacert.pem=<ca-cert-file>.pem occm-ca-secret
  ```

  Where:

  - `<ocnadd-namespace>`: OCNADD management or worker group namespace
  - `<ca-cert-file>`: Name of the PEM file containing the CA certificate or certificate chain
  - `occm-ca-cert`: Name of the secret storing the CA certificate or certificate chain

For example (for management and one worker group):

```
kubectl create secret generic -n ocnadd-deploy-mgmt --from-
file=cacert.pem=<ca-cert-file>.pem occm-ca-secret
kubectl create secret generic -n ocnadd-deploy-wg1 --from-
file=cacert.pem=<ca-cert-file>.pem occm-ca-secret
```

2.2.1.10 Configuring ServiceMonitor in OCCNE-INFRA

This section defines the OCCNE-INFRA `ocnadd` ServiceMonitor configuration used to scrape Kafka Prometheus metrics.

1. Login as the `root` user or any user with administrative privileges.
2. Create the file at `<path>/occne_ocnadd_servicemonitor.yaml` and add the below content to it:

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  labels:
    app: ocnadd
  name: occne-ocnadd-servicemonitor
  namespace: occne-infra
spec:
  endpoints:
  - path: /metrics
    port: kafka-metrics
    relabelings:
    - action: labelmap
      regex: __meta_kubernetes_pod_label_(.+)
  namespaceSelector:
    any: true
  selector:
    matchLabels:
```

3. Apply the configuration using the following command:

```
kubectl -n occne-infra apply -f <path>/occne_ocnadd_servicemonitor.yaml
```

2.2.2 Installation Tasks

This section describes the tasks that the user must follow for installing OCNADD.

> **Note**
>
> Before starting the installation tasks, ensure that the Prerequisites and Pre-Installation Tasks are completed.

2.2.2.1 Installing OCNADD Package

This section describes how to install the Oracle Communications Network Analytics Data Director (OCNADD) package.

To install the OCNADD package, perform the following steps:

Create OCNADD Namespace

Create the OCNADD namespace, if not already created. For more information, see Creating OCNADD Namespace.

Generate Certificates

If OCCM is used to create the certificates, follow the steps defined in OCCM Prerequisites for Installing OCNADD.

Else, perform the steps defined in Configuring SSL or TLS Certificates section to complete the certificate generation.

Update Database Parameters

To update the database parameters, see Configuring OCNADD Database.

Update ocnadd-custom-values-25.2.101.yaml file

Update the `ocnadd-custom-values-25.2.101.yaml` (depending on the type of deployment model) with the required parameters.

For more information on how to access and update the `ocnadd-custom-values-25.2.101.yaml` files, see Customizing OCNADD.

If OCCM is used to create the certificates, update the Mandatory Parameters specified in Helm Parameter Configuration for OCCM.

Install Helm Chart

OCNADD Release 25.2.101 or later release supports fresh deployment in centralized mode only.

OCNADD Installation with Default Worker Group

In 25.2.101, Data Director can be installed with default worker group in centralized mode.

Deploy Centralized Site with Default Group

To set up the centralized site, create copies of the charts and custom values for both the management group and each worker group from the "`ocnadd-package-25.2.101`" folder. The user can create multiple copies of helm charts folder and custom-values file in the following suggested way:

1. **For Management Group:** Create a copy of the following files from extracted folder:

```
# cd ocnadd-package-25.2.101
# cp -rf ocnadd ocnadd_mgmt
# cp custom_templates/ocnadd-custom-values-25.2.101.yaml ocnadd-custom-
values-mgmt-group.yaml
```

2. **For Default Worker Group:** Create a copy of the following files from extracted folder:

```
# cp -rf ocnadd ocnadd_default_wg
# cp custom_templates/ocnadd-custom-values-25.2.101.yaml ocnadd-custom-
values-default-wg-group.yaml
```

Installing Management Group:

1. Create a namespace for the Management Group if it doesn't exist already. See Creating OCNADD Namespace section.
   For example:

```
# kubectl create namespace ocnadd-deploy
```

2. Create certificates if it was not already created for the management group. Use the option "non-centralized" for the certificate generation if "generate_certs.sh". For more information about certificate generation, see Configuring SSL or TLS Certificates.
3. Modify the `ocnadd-custom-values-mgmt-group.yaml` created above and update it as below:

```
    global.deployment.centralized: true
    global.deployment.management: true
    global.deployment.management_namespace:ocnadd-deploy        ##--->
update it with namespace created in Step 1
    global.cluster.namespace.name:ocnadd-deploy                  ##--->
update it with namespace created in Step 1

    global.cluster.serviceAccount.name:ocnadd                   ## -->
update the ocnadd with namespace created in Step 1
    global.cluster.clusterRole.name:ocnadd                        ## -->
update the ocnadd with namespace created in Step 1
    global.cluster.clusterRoleBinding.name:ocnadd                 ## -->
update the ocnadd with namespace created in Step 1
```

4. Install using the "`ocnadd_mgmt`" Helm charts folder created for the management group:

```
helm install <management-release-name> -f ocnadd-custom-values-<mgmt-
group>.yaml --namespace <default-deploy-namespace> <helm_chart>
```

Where,

`<management-release-name>` release name of management group deployment

`<mgmt-group>` management custom values file

`<default-deploy-namespace>` namespace where management group is deployed

`<helm-chart>` helm chart folder of OCNADD

For example:

```
helm install ocnadd-mgmt -f ocnadd-custom-values-mgmt-group.yaml --
namespace ocnadd-deploy ocnadd_mgmt
```

Installing Default Worker Group:

1. Create certificates if it was not already created for the management group. For more information about certificate generation, see Configuring SSL or TLS Certificates and *Oracle Communications Network Analytics Suite Security Guide*.

   > **Note**
   >
   > For worker group certificate creation, select same namespace as selected for management group (For example, dd-default-deploy).

2. Modify the `ocnadd-custom-values-wg-group.yaml` file as follows:

```
    global.deployment.centralized: true
    global.deployment.management: false                              ##--->
default is true
    global.deployment.management_namespace:ocnadd-deploy             ##--->
update it with namespace created in Step 1
    global.cluster.namespace.name:ocnadd-deploy                      ##--->
update it with namespace created in Step 1

    global.cluster.serviceAccount.create: true                      ## -->
update the parameter to false
    global.cluster.clusterRole.create: true                         ## -->
update the parameter to false
    global.cluster.clusterRoleBinding.create: true                     ##
--> update the parameter to false
```

3. Ensure that only the required NF aggregations are enabled on the Data Director. For example, if the customer is only intending to use SCP as the source NF with the Data Director, it is recommended to turn off all other NF-specific aggregation instances. The following modifications should be made in the `ocnadd-custom-values-default-wg-group.yaml` file:

```
global.ocnaddscpaggregation.enabled: true           ##---> default is true
global.ocnaddnrfaggregation.enabled: false          ##---> default is
true, update to false
global.ocnaddseppaggregation.enabled: false         ##---> default is
true, update to false
global.ocnaddbsfaggregation.enabled: false           ##---> default is
false, no change required
global.ocnaddpcfaggregation.enabled: false           ##---> default is
false, no change required
```

4. Install using the "`ocnadd_wg`" Helm charts folder created for the default worker group:

```
helm install <default-worker-group-release-name> -f ocnadd-custom-values-
<default-wg-group>.yaml --namespace <default-deploy-namespace> <helm_chart>
```

   Where,

   `<default-worker-group-release-name>` release name of default worker group deployment

<default-wg-group> default worker group custom values file

<default-deploy-namespace> namespace where default worker group is deployed

<helm-chart> helm chart folder of OCNADD

For example:

```
helm install ocnadd-default-wg -f ocnadd-custom-values-default-wg-
group.yaml --namespace ocnadd-deploy ocnadd_default_wg
```

<Optional> OCNADD Installation with Multiple Worker Groups

To set up the centralized site, create copies of the charts and custom values for both the management group and each worker group from the "ocnadd-package-25.2.101" folder. Follow the below steps:

1. **For Management Group:** Create a copy of the following files from extracted folder:

```
# cd ocnadd-package-25.2.101
# cp -rf ocnadd ocnadd_mgmt
# cp custom_templates/ocnadd-custom-values-25.2.101.yaml ocnadd-custom-
values-mgmt-group.yaml
```

2. **For Worker Group:** Create a copy of the following files from extracted folder :

```
# cp -rf ocnadd ocnadd_wg1
# cp custom_templates/ocnadd-custom-values-25.2.101.yaml ocnadd-custom-
values-wg1-group.yaml
```

> **Note**
>
> For additional worker groups, repeat this process (for example, for Worker Group 2, create "ocnadd_wg2" and "ocnadd-custom-values-wg2-group.yaml").

Installing Management Group:

1. Create a namespace for the Management Group if it doesn't exist already. See Creating OCNADD Namespace section.
   For example:

```
# kubectl create namespace dd-mgmt-group
```

2. Create certificates if it was not already created for the management group. For more information about certificate generation, see Configuring SSL or TLS Certificates and *Oracle Communications Network Analytics Suite Security Guide*.
3. Modify the ocnadd-custom-values-mgmt-group.yaml file as follows:

```
global.deployment.centralized: true
global.deployment.management: true
global.deployment.management_namespace:ocnadd-deploy      ##---> update it
```

```
with management-group namespace for example dd-mgmt-group

global.cluster.namespace.name:ocnadd-deploy              ##---> update it
with management-group namespace for example dd-mgmt-group
global.cluster.serviceAccount.name:ocnadd                ## --> update
the ocnadd with the management-group namespace for example dd-mgmt-group
global.cluster.clusterRole.name:ocnadd                   ## --> update
the ocnadd with the management-group namespace for example dd-mgmt-group
global.cluster.clusterRoleBinding.name:ocnadd            ## --> update
the ocnadd with the management-group namespace for example dd-mgmt-group
```

4. Install using the "`ocnadd_mgmt`" Helm charts folder created for the management group:

   ```
   helm install <management-release-name> -f ocnadd-custom-values-<mgmt-
   group>.yaml --namespace <management-group-namespace> <helm_chart>
   ```

   For example:

   ```
   helm install ocnadd-mgmt -f ocnadd-custom-values-mgmt-group.yaml --
   namespace dd-mgmt-group ocnadd_mgmt
   ```

Installing Worker Group:

1. Create a namespace for Worker Group 1 if it doesn't exist already. See Creating OCNADD Namespace section.
   For example:

   ```
   # kubectl create namespace dd-worker-group1
   ```

2. Create certificates if it was not already created for the management group. For more information about certificate generation, see Configuring SSL or TLS Certificates and *Oracle Communications Network Analytics Suite Security Guide*.
3. Modify the `ocnadd-custom-values-wg1-group.yaml` file as follows:

   ```
       global.deployment.centralized: true
       global.deployment.management: false                          ##--->
   default is true
       global.deployment.management_namespace:ocnadd-deploy         ##--->
   update it with management-group namespace for example dd-mgmt-group

       global.cluster.namespace.name:ocnadd-deploy                  ##--->
   update it with worker-group namespace for example dd-worker-group1
       global.cluster.serviceAccount.name:ocnadd                    ## -->
   update the ocnadd with the worker-group namespace for example dd-worker-
   group1
       global.cluster.clusterRole.name:ocnadd                       ## -->
   update the ocnadd with the worker-group namespace for example dd-worker-
   group1
       global.cluster.clusterRoleBinding.name:ocnadd                ## -->
   update the ocnadd with the worker-group namespace for example dd-worker-
   group1
   ```

4. Ensure that only the required NF aggregations are enabled on the Data Director. For example, if the customer intends to use SCP as the source NF with the Data Director, it is recommended to turn off all other NF-specific aggregation instances. The following modifications should be made in the `ocnadd-custom-values-wg1-group.yaml` file:

```
#The values file should be modified for the particular NF aggregation
instance(s).
# For example customer is only using SCP, so aggregation instance for NRF,
SEPP, BSF, PCF e.t.c should be updated as suggested below:

global.ocnaddscpaggregation.enabled: true            ##---> default is true
global.ocnaddnrfaggregation.enabled: false            ##---> default is
true,   ## --> update the parameter to false
global.ocnaddseppaggregation.enabled: false           ##---> default is
true,   ## --> update the parameter to false
global.ocnaddbsfaggregation.enabled: false            ##---> default is
false   ## --> No change required
global.ocnaddpcfaggregation.enabled: false            ##---> default is
false   ## --> No change required
```

5. Install using the "`ocnadd_wg1`" Helm charts folder created for Worker Group 1:

```
helm install <worker-group1-release-name> -f ocnadd-custom-values-<wg1-
group>.yaml --namespace <worker-group1-namespace> <helm_chart>
```

For example:

```
helm install ocnadd-wg1 -f ocnadd-custom-values-wg1-group.yaml --namespace
dd-worker-group1 ocnadd_wg1
```

> **Note**
>
> For additional worker groups, repeat the "Installing Default Worker Group:" procedure. For instance, for Worker Group 2, replicate the steps accordingly.

> **Caution**
>
> Do not exit from helm install command manually. After running the helm install command, it takes some time to install all the services. In the meantime, you must not press Ctrl+C to come out from the command. It leads to some anomalous behavior.

2.2.2.2 Verifying OCNADD Installation

This section describes how to verify if Oracle Communications Network Analytics Data Director (OCNADD) is installed successfully.

To check the status of OCNADD deployment, perform the following task:

1. In the case of Helm, run one of the following commands:

```
helm status <helm-release> -n <namespace>
```

```
Example:
To check dd-management group
  # helm status ocnadd-mgmt -n dd-mgmt-group

To check dd-worker-group
  # helm status ocnadd-wg1 -n dd-worker-group1
```

The system displays the status as deployed if the deployment is successful.

2. Run the following command to check whether all the services are deployed and active:
   To check management-group:

```
watch kubectl get pod,svc -n dd-mgmt-group
```

   To check worker-group1:

```
watch kubectl get pod,svc  -n dd-worker-group1
```

```
kubectl -n <namespace_name> get services
```

> **Note**
>
> - All microservices status must be Running and Ready.
> - Take a backup of the following files that are required during fault recovery:
>   - Updated Helm charts for both management and worker group(s)
>   - Updated custom-values for both management and worker group(s)
>   - Secrets, certificates, and keys that are used during the installation for both management and worker group(s)
> - If the installation is not successful or you do not see the status as **Running** for all the pods, perform the troubleshooting steps. For more information, refer to *Oracle Communications Network Analytics Data Director Troubleshooting Guide*.

2.2.2.3 Creating OCNADD Kafka Topics

To create OCNADD Kakfa topics, see the "Creating Kafka Topic for OCNADD" section of *Oracle Communications Network Analytics Data Director User Guide*

2.2.2.4 Installing OCNADD GUI

This section describes how to install Oracle Communications Network Analytics Data Director (OCNADD) GUI using the following steps:

- Install OCNADD GUI
- Configure OCNADD GUI in CNC Console
- Access OCNADD GUI

Install OCNADD GUI

The OCNADD GUI gets installed along with the OCNADD services.

Configure OCNADD GUI in CNCC

Prerequisite: To configure OCNADD GUI in CNC Console, you must have the CNC Console installed. For information on how to install CNC Console and configure the OCNADD instance, see *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide*.

Before installing CNC Console, ensure to update the instances parameters with the following details in the `occncc_custom_values.yaml` file:

```
  instances:
    - id: Cluster1-dd-instance1
      type: DD-UI
      owner: Cluster1
      ip: 10.xx.xx.xx    #--> give the cluster/node IP
      port: 31456        #--> give the node port of ocnaddgui
      apiPrefix: /<clustername>/<namespace>/ocnadd
    - id: Cluster1-dd-instance1
      type: DD-API
      owner: Cluster1
      ip: 10.xx.xx.xx   #--> give the cluster/node IP
      port: 32406       #--> give the node port of ocnaddbackendrouter
      apiPrefix: /<clustername>/<namespace>/ocnaddapi

# Applicable only for Manager and Agent core. Used for Multi-Instance-Multi-
Cluster Configuration Validation
  validationHook:
    enabled: false   #--> add this enabled: false to validationHook

#--> do these changes under section :
cncc iam attributes
# If https is disabled, this Port would be HTTPS/1.0 Port (secured SSL)
    publicHttpSignalingPort: 30085  #--> CNC console nodeport

#--> add these lines under cncc-iam attributes
# If Static node port needs to be set, then set staticNodePortEnabled flag to
true and provide value for staticNodePort
    # Else random node port will be assigned by K8
    staticNodePortEnabled: true
    staticHttpNodePort: 30085  #--> CNC console nodeport
    staticHttpsNodePort: 30053

#--> do these changes under section : manager cncc core attributes
#--> add these lines under mcncc-core attributes

# If Static node port needs to be set, then set staticNodePortEnabled flag to
true and provide value for staticNodePort
    # Else random node port will be assigned by K8
    staticNodePortEnabled: true
    staticHttpNodePort: 30075
    staticHttpsNodePort: 30043
```

```
#--> do these changes under section : agent cncc core attributes
#--> add these lines under acncc-core attributes
# If Static node port needs to be set, then set staticNodePortEnabled flag to
true and provide value for staticNodePort
    # Else random node port will be assigned by K8
    staticNodePortEnabled: true
    staticHttpNodePort: 30076
    staticHttpsNodePort: 30044
```

If CNC Console is already installed, ensure to upgrade it with the following parameters updated in the `occncc_custom_values.yaml` file:

```
instances:
  - id: Cluster1-dd-instance1
    type: DD-UI
    owner: Cluster1
    fqdn: ocnaddgui.<dd_mgmt_namespace>.svc.<cluster_domain>     #--> update
the namespace and cluster domain.
    port: 31456        #--> ocnaddgui port
    apiPrefix: /<clustername>/<namespace>/ocnadd
  - id: Cluster1-dd-instance1
    type: DD-API
    owner: Cluster1
    fqdn: ocnadduirouter.<dd_mgmt_namespace>.svc.<cluster_domain> #--> Update
the namespace and cluster domain
    port: 32406       #--> ocnadduirouter port
    apiPrefix: /<clustername>/<namespace>/ocnaddapi
```

Example:

If OCNADD GUI is deployed in the **occne-ocdd** cluster and the **ocnadd-deploy** namespace, then the prefix in CNC Console `occncc_custom_values.yaml` will be as follows:

```
DD-UI apiPrefix:
/occne-ocdd/ocnadd-deploy/ocnadd
DD-API apiPrefix:
/occne-ocdd/ocnadd-deploy/ocnaddapi
```

Access OCNADD GUI

To access OCNADD GUI, follow the procedure mentioned in the "Accessing CNC Console" section of *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide*.

2.2.2.5 Adding a Worker Group

Assumptions:

1. Centralized Site is already deployed with at least one worker group.
2. Management Group deployment is up and running, example namespace "dd-mgmt-group".
3. Worker Group namespace which is being added is created, example namespace "dd-worker-group2".

1. Create the namespace for worker-group2 if not already created. For more infromation, see Creating OCNADD Namespace.
   For example:

   ```
   kubectl create namespace dd-worker-group2
   ```

2. Create a copy of the following files from extracted folder:

   ```
   cp -rf ocnadd ocnadd_wg2
   cp custom_templates/ocnadd-custom-values-25.2.101.yaml ocnadd-custom-
   values-wg2-group.yaml
   ```

3. Generate certificates for the new worker group according to the "Configuring SSL or TLS Certificates" section and the *Oracle Communications Network Analytics Suite Security Guide.*

4. Modify the `ocnadd-custom-values-wg2-group.yaml` file as follows:

   ```
       global.deployment.centralized: true
       global.deployment.management: true                              ##--->
   Update it to 'false'
       global.deployment.management_namespace:ocnadd-deploy            ##--->
   update it with management-group namespace for example dd-mgmt-group

       global.cluster.namespace.name:ocnadd-deploy                     ##--->
   update it with worker-group namespace for example dd-worker-group2
       global.cluster.serviceAccount.name:ocnadd                       ## -->
   update the ocnadd with the worker-group namespace for example dd-worker-
   group2
       global.cluster.clusterRole.name:ocnadd                          ## -->
   update the ocnadd with the worker-group namespace for example dd-worker-
   group2
       global.cluster.clusterRoleBinding.name:ocnadd                   ## -->
   update the ocnadd with the worker-group namespace for example dd-worker-
   group2
   ```

   a. Ensure that only the required NF aggregation is enabled on the Data Director. For example, if the customer intends to use only SCP as the source NF with Data Director, it is recommended to turn off all other NF-specific aggregation instances. The following modifications should be made in `ocnadd-custom-values-wg2-group.yaml` file.

      ```
      #The values file should be modified for the particular NF aggregation
      instance(s).
      # For example customer is only using SCP, so aggregation instance for
      NRF, SEPP, BSF, PCF e.t.c should be updated as suggested below:

      global.ocnaddscpaggregation.enabled: true            ##---> default is
      true
      global.ocnaddnrfaggregation.enabled: false            ##---> default is
      true,   ## --> update the parameter to false
      global.ocnaddseppaggregation.enabled: false           ##---> default is
      true,   ## --> update the parameter to false
      ```

```
global.ocnaddbsfaggregation.enabled: false            ##---> default is
false   ## --> No change required
global.ocnaddpcfaggregation.enabled: false            ##---> default is
false   ## --> No change required
```

5. Install using the ocnadd_wg2 Helm charts folder created for the worker group:

   ```
   helm install <worker-group2-release-name> -f ocnadd-custom-values-<wg2-
   group>.yaml --namespace <worker-group2-namespace> <helm_chart>
   ```

   For example:

   ```
   helm install ocnadd-wg2 -f ocnadd-custom-values-wg2-group.yaml --namespace
   dd-worker-group2 ocnadd_wg2
   ```

6. To verify the installation of the new worker group:

   ```
   # watch kubectl get pod,svc -n dd-worker-group2
   ```

7. Follow the section "Creating OCNADD Kafka Topics" to create topics on newly added worker group.

2.2.2.6 Deleting a Worker Group

Assumptions:

1. Centralized Site is already deployed with at least one worker group.
2. Management Group deployment is up and running, example namespace "dd-mgmt-group".
3. Worker groups "worker-group1" and "worker-group2" deployment are up and running, example namespace 'dd-worker-group1' and 'dd-worker-group2'.
4. Worker group "worker-group2" needs to be deleted.
5. Clean up the configurations corresponding to worker-group which is being deleted. For example, if it is 'worker-group2':
   a. Delete all the adapter feeds corresponding to worker-group2 from the UI.
   b. Delete all the filters applied to worker-group2 from the UI.
   c. Delete all the correlation applied to worker-group2 from the UI.
   d. Delete all the Kafka feeds corresponding to worker-group2 from the UI.
6. Run the follwing command to uninstall the worker group:

   ```
   helm  uninstall <worker-group2-release-name> -n <worker-group1-namespace>
   ```

   For example:

   ```
   helm uninstall ocnadd-wg2 -n dd-worker-group2
   ```

7. Delete the worker group namespace:

   ```
   kubectl delete namespace <worker-group2-namespace>
   ```

For example:

```
kubectl delete namespace dd-worker-group2
```

2.2.2.7 Creating Alarms and Dashboard in OCI

This step is necessary only for the Data Director deployment on the OCI platform. Follow the steps explained in the section 'Creating Alarms and Dashboards in OCI' from the *Oracle Communications Network Analytics Data Director User Guide*.

2.2.2.8 Adding or Updating Load Balancer IPs in SAN When OCCM is Used

The certificates created by OCCM will not contain any IP values in the SAN field, except the values provided in the `global.certificate.occm.san.*.ips` field in `ocnadd-custom-values-25.2.101.yaml` for kafka-broker, ingress adapter and redundancy agent certificates.

For descriptions of the different Helm parameters, see Helm Parameter Configuration for OCCM.

To add or update the Loadbalancer IPs of these services in SAN, see the steps mentioned in the following sections:

Adding Load Balancer IPs for Kafka

1. Update the `global.certificates.occm.san.kafka.ips` in `ocnadd-custom-values-25.2.101.yaml` of the required worker group.

```
global:
  certificates:
    occm:
      san:
        kafka:
          ips: ["10.10.10.10", "10.10.10.11", "10.10.10.12",
"10.10.10.13"]              # Add the loadbalancer ip of each Kafka broker
services
```

2. Run Helm upgrade for the worker group namespace.

```
$ helm upgrade <worker-group-release-name> -f <worker-group-custom-values>
-n <worker-group-ns> <ocnadd-helm-chart-location>
```

3. Update the `global.certificates.occm.san.kafka.update_required`, `global.certificates.occm.san.kafka.uuid.client`, and `global.certificates.occm.san.kafka.uuid.server` in `ocnadd-custom-values-25.2.101.yaml` of the required worker group.

```
global:
  certificates:
    occm:
      san:
        kafka:
          update_required: true
# Set to true, default is false
          uuid:
            client: 9138b974-2c89-4c9d-bc5c-0ca82752d50b
```

```
# Provide the UUID value of the certificate KAFKABROKER-SECRET-CLIENT-
<namespace> from OCCM, where <namespace> is the Worker group namespace
            server: 5e765aeb-ae1b-426b-8481-f8f3dcdd645e
# Provide the UUID value of the certificate KAFKABROKER-SECRET-SERVER-
<namespace> from OCCM, where <namespace> is the Worker group namespace
```

4. Run Helm upgrade for the worker group namespace.

   ```
   $ helm upgrade <worker-group-release-name> -f <worker-group-custom-values>
   -n <worker-group-ns> <ocnadd-helm-chart-location>
   ```

   New certificates will be created. Verify them through the OCCM UI. Kafka brokers will also restart after the Helm upgrade is completed and will start using the newly created certificates.

Adding Load Balancer IP for Redundancy Agent

1. Update the `global.certificates.occm.san.redundancy_agent.ips` in `ocnadd-custom-values-25.2.101.yaml` of the required management group.

   ```
   global:
     certificates:
       occm:
         san:
           redundancy_agent:
             ips: ["10.10.10.10"]  # Add the load balancer IP of the
   redundancy agent service
   ```

2. Run Helm upgrade for the management group namespace.

   ```
   $ helm upgrade <management-group-release-name> -f <management-group-custom-
   values> -n <management-group-ns> <ocnadd-helm-chart-location>
   ```

3. Update the `global.certificates.occm.san.redundancy_agent.update_required`, `global.certificates.occm.san.redundancy_agent.uuid.client`, and `global.certificates.occm.san.redundancy_agent.uuid.server` in `ocnadd-custom-values-25.2.101.yaml` of the required management group.

   ```
   global:
     certificates:
       occm:
         san:
           redundancy_agent:
             update_required: true
             uuid:
               client: 9138b974-2c89-4c9d-bc5c-0ca82752d50b  # Provide the
   UUID value of the certificate REDUNDANCYAGENT-SECRET-CLIENT-ocnadd-mgmt,
   if ocnadd-mgmt is the management group namespace
               server: 5e765aeb-ae1b-426b-8481-f8f3dcdd645e  # Provide the
   UUID value of the certificate REDUNDANCYAGENT-SECRET-SERVER-ocnadd-mgmt,
   if ocnadd-mgmt is the management group namespace
   ```

4. Run Helm upgrade for the worker group namespace.

```
$ helm upgrade <management-group-release-name> -f <management-group-custom-values> -n <management-group-ns> <ocnadd-helm-chart-location>
```

New certificates will be created. Verify them through the OCCM UI. The Redundancy Agent will also restart after the Helm upgrade is completed and will start using the newly created certificates.

Adding Load Balancer IPs for Ingress Adapter

1. Update the `global.certificates.occm.san.ingress_adapter.ips` in `ocnadd-custom-values-25.2.101.yaml` of the required worker group.

```
global:
  certificates:
    occm:
      san:
        ingress_adapter:
          ips: ["10.10.10.10", "10.10.10.11", "10.10.10.12",
"10.10.10.13"]  # Add the load balancer IP of each ingress adapter service
```

2. Run Helm upgrade for the worker group namespace.

```
$ helm upgrade -n <worker-group-ns> <worker-group-chart-name> -f <worker-group-custom-values> <ocnadd-helm-chart-location>
```

3. Update the `global.certificates.occm.san.ingress_adapter.update_required`, `global.certificates.occm.san.ingress_adapter.uuid.client`, and `global.certificates.occm.san.ingress_adapter.uuid.server` in `ocnadd-custom-values-25.2.101.yaml` of the required worker group.

```
global:
  certificates:
    occm:
      san:
        ingress_adapter:
          update_required: true                                  # Set to
true, default is false
          uuid:
            client: 9138b974-2c89-4c9d-bc5c-0ca82752d50b         #
Provide the UUID value of the certificate INGRESSADAPTER-SECRET-CLIENT-
<namespace> from OCCM, where <namespace> is the Worker group namespace
            server: 5e765aeb-ae1b-426b-8481-f8f3dcdd645e         #
Provide the UUID value of the certificate INGRESSADAPTER-SECRET-SERVER-
<namespace> from OCCM, where <namespace> is the Worker group namespace
```

4. Run Helm upgrade for the worker group namespace.

```
$ helm upgrade <worker-group-release-name> -f <worker-group-custom-values>
-n <worker-group-ns> <ocnadd-helm-chart-location>
```

New certificates will be created with the new/updated SAN entries. Verify them through the OCCM UI.

5. Update the `global.env.admin.OCNADD_UPGRADE_WG_NS` in `ocnadd-custom-values-25.2.101.yaml` of the required management group with the worker group namespace.

```
global:
  env:
    admin:
      OCNADD_UPGRADE_WG_NS: ocnadd-wg-1  # Where ocnadd-wg-1 is the
namespace of the ingress adapter service
```

6. Run Helm upgrade for the management group namespace.

```
$ helm upgrade <management-group-release-name> -f <management-group-custom-
values> -n <management-group-ns> <ocnadd-helm-chart-location> --set
global.env.admin.OCNADD_INGRESS_ADAPTER_UPGRADE_ENABLE=true
```

2.2.3 Post-Installation Tasks

2.2.3.1 Enabling Two Site Redundancy

This feature is introduced as part of Georedundancy in OCNADD. To enable it, see 'Two Site Redundancy Enable' section in the *Oracle Communications Network Analytics Data Director User Guide*. It is recommended to enable this feature after completing the deployment of the target release.

2.2.3.2 Enabling Traffic Segregation Using CNLB

This feature is introduced as part of traffic segregation support in OCNADD. To enable it, see 'Enabling or Disabling Traffic Segregation Using CNLB in OCNADD ' section in the *Oracle Communications Network Analytics Data Director User Guide*. It is recommended to enable this feature after completing the deployment of the target release.

2.2.3.3 Enabling Druid as Extended Storage Feature

This feature is introduced as part of extended storage in Data Director. To enable it, see the *"Druid Integration with OCNADD"* section in the *Oracle Communications Network Analytics Data Director User Guide*. The feature is recommended to be enabled after the release installation is completed. Extended storage using the cnDBTier database is available by default.

3
Customizing OCNADD

This chapter describes how to customize the Oracle Communications Network Analytics Data Director (OCNADD) deployment, supported deployment models, and provides a list of configuration parameters in the Helm file that are used for customization. The OCNADD deployment is customized by overriding the default values of various configurable parameters.

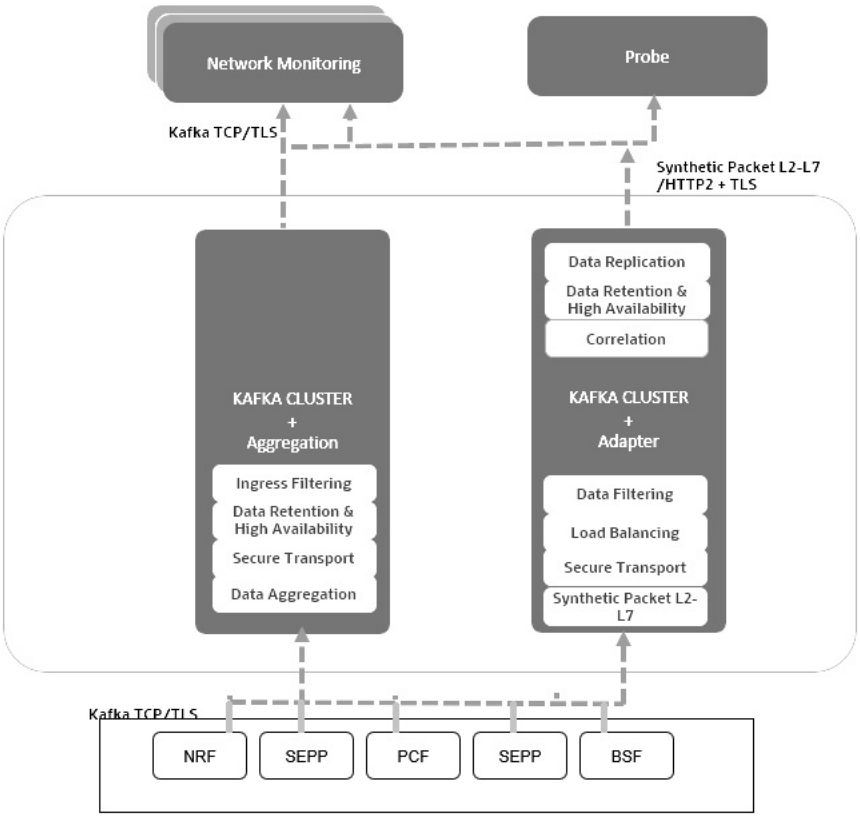
3.1 OCNADD Deployment Models

OCNADD supports the following deployment models:

- **Model 1:** All Data Director Services (Default)
- **Model 2:** Kafka, Common Services, and Aggregation Services

> **Note**
>
> The Data Director supports egress adapters for outbound connections. The egress adapters add value to the message feed by filtering and synthesizing the packets before sending the messages out on the egress connection type 'HTTP/2' or 'Synthetic Feed'. If the customer selects a deployment model that does not include the Egress adapter, additional features such as synthetic packet generation will not be available, although the filtering and correlation features will be available using Kafka feeds only.

The following diagram depicts the various Data Director deployment models:



The following table depicts the resource savings in the various deployment models:

Table 3-1 Resource Saving

Deployment Model	Model 1 (All Services)	Model 2 (Kafka +Aggregation)
Common Services	Available	Available
Aggregation Service	Available	Available
Adapter Service	Available	Not available
Kafka	Available	Available
Resource Saving (approx. %)	0	60
Supported Egress Interfaces	HTTP/2 TCP	Kafka

> **Note**
>
> The resource saving for the default model will be more if many worker groups are going to be managed with a single management group.

Model 1: All Data Director Services (Default: Centralized)

This OCNADD deployment model includes all the services and features. This is the default model and all the services are selected by default in the `ocnadd-custom-values-25.2.101.yaml` file.

This profile can stream NFs (SCP, NRF, SEPP) data up to 15K MPS and can be scaled to handle up to 135K MPS for HTTP2 feed when "weighted_lb" is off.

Replication Factor should be 1 and the incoming message size on OCNADD should be less than or equal to 3500 bytes.

- Use the GUI to configure message feeds on OCNADD.
- The Oracle Producer NFs (SCP, SEPP, NRF, PCF, and BSF) copy the messages to their respective source topics.

For this model, the user need not enable or disable any service in the `ocnadd-custom-values-25.2.101.yaml`. The default parameters are as below.

```
global:
    ocnaddalarm:
        enabled: true
    ocnaddconfiguration:
        enabled: true
    ocnaddhealthmonitoring:
        enabled: true
    ocnaddscpaggregation:
         enabled: true
    ocnaddnrfaggregation:
         enabled: true
    ocnaddseppaggregation:
         enabled: true
    ocnaddbsfaggregation:
         enabled: false
   ocnaddpcfaggregation:
         enabled: false
    ocnaddbackuprestore:
        enabled: true
    ocnaddkafka:
        enabled: true
    ocnaddadmin:
         enabled: true
    ocnadduirouter:
        enabled: true
    ocnaddgui:
        enabled: true
    ocnaddfilter:
        enabled: true
    ocnaddexport:
        enabled: true
   ocnaddnonracleaggregation:
        enabled: true
   ocnaddredundancyagent:
        enabled: true
```

Model 2: Kafka, Common Services, and Aggregation Services

Use this model when the customer does not wish to receive the message feed using HTTP/2 or TCP connection mode. The third-party monitoring application available to the customer can consume data directly from the Kafka cluster. The Egress adapter is not required in this deployment model; however, the OCNADD deployment requires common services (such as UI,

Configuration, Health monitoring, Alarm, and Admin). Features like correlation-id-based load balancing, synthetic feed, and HTTP/2 feeds are unavailable in this deployment model, although the filtering and correlation features will be available using Kafka feeds only. This model saves the egress adapter resource; however, additional resources will be required for Filtering and Correlation services once these features are used in the configurations from UI.

The export feature is also available, however it has to be enabled in the charts by enabling the ocnaddexport service and further export configuration from the UI. For more information, see "Export" section in "OCNADD Features and Feature Specific Limits " chapter and "Export Configuration " section in " Configuring OCNADD" chapter in *Oracle Communications Network Analytics Data Director User Guide*.

This deployment model supports direct Kafka feed. For more information, see "External Kafka Feeds" section in *Oracle Communications Network Analytics Data Director User Guide*.

The default parameters are as below.

```
global:
    ocnaddalarm:
        enabled: true
    ocnaddconfiguration:
        enabled: true
    ocnaddhealthmonitoring:
        enabled: true
    ocnaddscpaggregation:
         enabled: true
    ocnaddnrfaggregation:
         enabled: true
    ocnaddseppaggregation:
         enabled: true
    ocnaddbsfaggregation:
         enabled: false
   ocnaddpcfaggregation:
         enabled: false
    ocnaddbackuprestore:
        enabled: true
    ocnaddkafka:
        enabled: true
    ocnaddadmin:
        enabled: true
    ocnadduirouter:
         enabled: true
    ocnaddgui:
         enabled: true
    ocnaddfilter:
         enabled: true
    ocnaddexport:
        enabled: false
    ocnaddnonracleaggregation:
        enabled: false
    ocnaddredundancyagent:
        enabled: false
```

1. The aggregation service aggregates traffic from the source topics to the Kafka main topic. Choosing any specific combination of NFs for aggregation rules is not possible. The total traffic received is aggregated and available to the consumers.

2. The third-party consumer application must create external Kafka feeds to connect with the Kafka cluster, which will allow them to consume messages directly from the designated topic.

> **Note**
>
> 1. Aggregation: The traffic will be aggregated using the configuration on Oracle Producer NFs to use the Main topic for copying messages on the OCNADD.
> 2. The message feeds must be created from the GUI and aggregation rules determine the source NF combinations for aggregation.
> 3. Metrics related to the feed shall be available on the GUI. The GUI can also be used to view the OCNADD alarms.

The customer can customize the OCNADD deployment based on the identified resources. Plan the resources based on the deployment model and services required for the specific model. For more information of OCNADD resources see *Oracle Communications Network Analytics Data Director Benchmarking Guide*.

In both models, the two-site redundancy feature can be enabled. For more information, see 'Two Site Redundancy Enable' section in the *Oracle Communications Network Analytics Data Director User Guide*.

3.2 Customize Configuration Parameters

Perform the following procedure to customize the `ocnadd-custom-values-25.2.101.yaml` files as per requirements for both parent and sub-charts.

1. Ensure that you have the Data Director charts tgz file, which is available in the extracted release package. For information about how to download the release package from MOS, see Downloading OCNADD Package.
2. Extract the OCNADD package if not already extracted, and unzip the custom-templates.zip
   a. Change the directory to custom-templates to access the parent `ocnadd-custom-values-25.2.101.yaml`. This file is used to customize the deployment parameters during installation.
   Change the following parameters in the `ocnadd-custom-values-25.2.101.yaml` and save the file:
      i. Update the repository path in `global.env.repo.REPO_HOST_PORT: <customer repository path>`
      ii. Update the CLUSTER-INFO parameters:
         i. `cluster.domainName: <customer cluster domain name>`
         ii. `cluster.clusterName: <customer cluster name>`
      iii. Update the database IP and database name:
         - `db_ip: 10.20.30.40` (Update with DB instance IP or with FQDN. For example, `mysql-connectivity-service.<cnDBTier namespace>`)
         - `db_port: 3306` (If using a different port for DB, change it. By default, DB port is `3306`)

- `configuration_db: configuration_schema` (Update the DB name as per the section Update Database Name. No change is needed if default DB names are used.)
- `alarm_db: alarm_schema` (Update the DB name as per the section Update Database Name. No change is needed if default DB names are used.)
- `health_db: healthdb_schema` (Update the DB name as per the section Update Database Name. No change is needed if default DB names are used.)
- `storageadapter_db: storageadapter_schema` (Update the DB name as per the section Update Database Name. No change is needed if default DB names are used.)

iv. Update the cluster and cluster domain name for UI

```
ocnadduirouter:
   ocnadduirouter:
       name: ocnadduirouter
       env:
           PROMETHEUS_API: http://occne-kube-prom-stack-kube-
prometheus.occne-infra.ocnadd:80    ## --> update 'ocnadd' with
<customer cluster domain name>
           DD_PROMETHEUS_PATH: /blurr8/prometheus/api/v1/
query_range        ## --> update 'blurr8' with <customer cluster
name>
```

v. Change the Prometheus Monitoring Details, bases on the desired MPS profile, default threshold MPS is `100000`:

```
 cluster.mps: 100000
```

vi. Update the default cluster.prometheus_url based on the cluster information:
Default prometheus_url: `http://localhost:9000/<cluster-name>/prometheus/api/v1/query_range`

Example URL(For CNE deployment and cluster name as `ocnadd`):

prometheus_url: `http://occne-kube-prom-stack-kube-prometheus.occne-infra.svc.ocnadd:80/ocnadd/prometheus/api/v1/query_range`

vii. (Optional) Updating the OCNADD Backup Cronjob:
Modify the below backup parameters as needed For more information on backup and restore, see "Fault Recovery" section.

```
BACKUP_STORAGE : Represents Backup storage PVC size
     BACKUP_CRONEXPRESSION : Represents the time of execution
     PURGE_DAYS : Represents the backup retention period in days

     ocnaddbackuprestore:
        ocnaddbackuprestore:
            name: ocnaddbackuprestore
            env:
                BACKUP_STORAGE: 20Gi
                BACKUP_CRONEXPRESSION: "0 8 * * *"
                PURGE_DAYS: 7
```

viii. Updating the OCNADD Kafka Deployment Mode:

In the current release deployment by Kraft mode is the only supported mode for the Kafka Cluster, the Zookeeper mode has been obsoleted and removed.

Modify the below global parameter for enabling the Kraft based Kafka deployment. In this case, zookeeper will not be deployed and instead of zookeeper, Kraft controller service will be deployed.

> **Note**
>
> ```
> # Set this parameter to 'true' **only** when KRaft mode is
> selected during a fresh installation.
> # This parameter **must remain 'false'** during a Data
> Director upgrade (Zookeeper mode) or migration from
> Zookeeper to KRaft.
> # During an upgrade, if the source release was already in
> KRaft mode, this parameter should remain 'true'.
>
> global.kafka.kraftEnabled: false  ## --> update this to
> 'true'
> ```

ix. Enable the specific aggregation service as needed. The default values are listed below. To enable a specific aggregation service, set its corresponding value to `true`.

```
global:
    ocnaddscpaggregation:
         enabled: true
    ocnaddnrfaggregation:
         enabled: true
    ocnaddseppaggregation:
         enabled: true
    ocnaddbsfaggregation:
         enabled: false
   ocnaddpcfaggregation:
         enabled: false
```

3. Customize the rules file `<chartpath>/templates/ocnadd-alerting-rules.yaml`:
   - If OCNADD is to be installed in OCI setup, then remove the `<chartpath>/templates/ocnadd-alerting-rules.yaml` and `<chartpath>/templates/ocnadd-mgmt-alerting-rules.yaml` files.
   - If OCNADD is to be installed in CNE Setup, then all the services will be monitored by Prometheus By default. So there will not be any Modifications in the Helm Chart. All the Prometheus Alert Rules Present in Helm Chart will be Updated in Prometheus Server. (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 to be installed in Tanzu setup, then modify the "`metadata.labels`" value in `<chartpath>/templates/ocnadd-alerting-rules.yaml` and `<chartpath>/templates/ocnadd-mgmt-alerting-rules.yaml` files as below:
     For Example "`release: prom-operator`" instead of "`role: cnc-alerting-rules`".

To obtain the labels details use the below command:

```
kubectl get prometheus <Prometheus_Configuration_NAME> -n
<Prometheus_Namespace> -o=jsonpath='{.spec.ruleSelector.matchLabels}'
```

Example:

```
$ kubectl get prometheus prom-operator-kube-prometh-prometheus -n occne-
infra -o=jsonpath='{.spec.ruleSelector.matchLabels}'{"release: prom-
operator"}
```

Sample Alert File:

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

4. <Optional> Creating Registry Credentials:
   If the user image repository is protected and has an authentication mechanism, follow the below steps:
   a. Use the kubectl command to create a secret named "regcred" with the credentials of the image repository.
   b. Update the `imagePullSecret.enable` field to `true` in the `ocnadd-custom-values-25.2.101.yaml` file as follows:

      ```
      imagePullSecret:
          enable: true      ## --> update this to 'true'
      ```

5. <Optional> Kafka preinstall configuration changes:
   a. <Optional> To change the profiles of the brokers, edit the respective values (CPU, memory, storage, external-access, security, jaas-password, replicas, internal replication factor, and so on) in kafka-section of `ocnadd-custom-values-25.2.101.yaml` file. Ensure that this is done for all the applicable worker groups.
   b. <Optional> When the security protocol is SASL and the customer required to add new users, update the `kafka_server_jaas.conf`, `zookeeper_jaas.conf` files in `<chartpath>/charts/ocnaddkafka/config`.
   c. <Optional> Customize `offsetsTopicReplicationFactor` and `transactionStateLogReplicationFactor` under `ocnaddkafka.ocnadd.kafkaBroker.kafkaProperties` in the `ocnadd-custom-values-25.2.101.yaml` file.
      Update internal topics replication factor:

- Update to below values when higher throughput with lower latency is needed. This can have lower message reliability in case of Kafka broker goes down:

```
offsetsTopicReplicationFactor: 1
transactionStateLogReplicationFactor: 1
```

- Update to below values when higher message reliability is required (RF>1). This can potentially have lower throughput and higher latency if the Kafka cluster Disk IOPS & cluster network bandwidth are less performing:

```
offsetsTopicReplicationFactor: 2
transactionStateLogReplicationFactor: 2
```

6. Storage Class:
   - If deploying on Tanzu, update the storageClass in the `ocnadd-custom-values-25.2.101.yaml` with the respective storage class name of the TANZU platform. For example, `zfs-storage-policy`.
   - If deploying on OCI, update the storageClass in the `ocnadd-custom-values-25.2.101.yaml` with the respective storage class name of the OCI platform, it should be "`oci-bv`"

   > **Note**
   >
   > This step is specific to the TANZU and OCI platform. Skip this step if you are installing OCNADD on CNE. For CNE, the default storageClass is standard..

7. <Optional> To enable Egress Annotation, see the "Enabling Egress Annotation" section in the *Oracle Communications Network Analytics Data Director User Guide*. The step can be skipped if the Egress Traffic separation via CNLB is planned to be used.
8. Loadbalancer on OCI: Update the following in `ocnadd-custom-values-25.2.101.yaml`
   a. `global.env.oci: false =====> set it to true`
   b. `global.env.subnetOcid:<subnet_ocid> # Add the OCID of the subnet that user want to use for creating load balancer`
9. <Optional> Enable RAM-based storage for the Kafka cluster. This feature has been introduced to support RAM-based storage in the Kafka cluster. It provides higher throughput in scenarios where lower message retention with lower latency is required. To enable RAM-based storage in the Kafka cluster, see the "Enable RAM Storage in Kafka Cluster" section in the *Oracle Communications Network Analytics Data Director User Guide*.

3.3 Global Parameters

Table 3-2 Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ocnaddalarm.enabled	BOOLEAN	true/ false	true	M	To enable alarm charts.
env.admin.OCNADD_CORR_UPGRADE_ENABLE	BOOLEAN	true/ false	false	M	Upgrade correlation service during Helm upgrade if the flag is set to true
env.admin.OCNADD_INGRESS_ADAPTER_UPGRADE_ENABLE	BOOLEAN	true/ false	false	M	Upgrade ingress adapter during Helm upgrade if the flag is set to true
env.admin.OCNADD_STORAGE_ADAPTER_UPGRADE_ENABLE	BOOLEAN	true/ false	false	M	Upgrade storage adapter during Helm upgrade if the flag is set to true
ocnaddconfiguration.enabled	BOOLEAN	true/ false	true	M	To enable configuration charts.
ocnaddhealthmonitoring.enabled	BOOLEAN	true/ false	true	M	To enable health monitoring charts.
ocnaddfilter.enabled	BOOLEAN	true/ false	false	M	To disable filter charts ('false' for the current release).
ocnaddscpaggregation.enabled	BOOLEAN	true/ false	true	M	To enable SCP aggregation charts
ocnaddnrfaggregation.enabled	BOOLEAN	true/ false	true	M	To enable NRF aggregation charts
ocnaddseppaggregation.enabled	BOOLEAN	true/ false	true	M	To enable SEPP aggregation charts
ocnaddbsfaggregation.enabled	BOOLEAN	true/ false	false	M	To enable BSF aggregation charts
ocnaddpcfaggregation.enabled	BOOLEAN	true/ false	false	M	To enable PCF aggregation charts
ocnaddbackuprestore.enabled	BOOLEAN	true/ false	true	M	To enable backup restore charts.
ocnaddkafka.enabled	BOOLEAN	true/ false	true	M	To enable Kafka charts.
ocnaddadminsvc.enabled	BOOLEAN	true/ false	true	M	To enable adminsvc charts.
ocnaddgui.enabled	BOOLEAN	true/ false	true	M	To enable GUI charts.
ocnadduirouter.enabled	BOOLEAN	true/ false	true	M	To enable UI router charts.
ocnaddredundancyagent.enabled	BOOLEAN	true/ false	false	M	To enable two site redundancy

Table 3-2 (Cont.) Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ocnaddexport	BOOLEAN	true/ false	false	M	To enable export service
ocnaddnonoracleaggregation	BOOLEAN	true/ false	false	M	To enable non-Oracle aggregation feature through aggregation service instance.
ocnaddredundancyagent.egress	BOOLEAN	true/ false	false	C	Required if egress annotation is required to allow traffic outside cluster.
env.oci	BOOLEAN	true/ false	false	M	It should be set to true for OCI platform.
env.subnetOcid	STRING	NA	NA	C	It is required for OCI platform. The OCID of the subnet that you want to use for creating load balancers.
env.topologyKey	STRING	NA	kubernetes.io/ hostname	M	The topology key for setting up the topology constraints on the pod deployment.
env.ocwebclient.OCWEBCLIENT_TIMEOUT	INTEGER	NA	30	O	Webclient timeout in seconds
env.ocwebclient.OCWEBCLIENT_KEEPALIVE_IDLE	INTEGER	NA	90	O	Webclient keep alive idle time in seconds
env.repo.repo.REPO_HOST_PORT	STRING	NA	docker.io	M	Local container registry to pull the images
env.repo.repo.REPO_PATH	STRING	NA	ocdd.repo	M	Additional repo path
env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE	BOOLEAN	NA	false	M	Upgrade consumer adapter during Helm upgrade if the flag is set to true.
scaleDownOnePodAtATime	BOOLEAN	true/ false	false	M	Scale down Pods one at a time.
stabilizationWindowSeconds	INTEGER	NA	60	M	Stabilization period in seconds post which scale down starts.
scaleDownPeriodSeconds	INTEGER	NA	30	M	Period of each scale down operation in seconds.
scaleDownValue	INTEGER	NA	1	M	Number of pods which shall go down in every scaleDownPeriodSeconds.
initContainers.name	STRING	NA	ocnaddinitcontainer	M	Name of initContainer for SSL support
initContainers.REPO_PATH	STRING	NA	utils.repo	M	Repo path where init image is stored
initContainers.volumeMounts.ts_ks_volumeName	STRING	NA	truststore-keystore-volume	M	Volume name for truststore

Table 3-2 (Cont.) Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
initContainers.volumeMounts.ts_ks_volumePath	STRING	NA	/var/securityfiles/keystore	M	Path where keystore files are stored
initContainers.volumeMounts.certificateName	STRING	NA	client-server-certificate	M	Volume name for server certificates
initContainers.volumeMounts.certificatePath	STRING	NA	/var/securityfiles/certs	M	Path where cert files are stored
initContainers.env.cert_file_params.SERVER_CERT_FILE	STRING	NA	servercert.pem	M	Server cert filename
initContainers.env.cert_file_params.CLIENT_CERT_FILE	STRING	NA	clientcert.pem	M	Client cert filename
initContainers.env.cert_file_params.SERVER_KEY_FILE	STRING	NA	serverprivatekey.pem	M	Server Private Key filename
initContainers.env.cert_file_params.CLIENT_KEY_FILE	STRING	NA	clientprivatekey.pem	M	Client Private Key filename
initContainers.env.ks_file_params.SERVER_KEY_STORE	STRING	NA	serverKeyStore.p12	M	Server Keystore file
initContainers.env.ks_file_params.CLIENT_KEY_STORE	STRING	NA	clientKeyStore.p12	M	Client Keystore file
initContainers.env.ks_file_params.TRUST_STORE	STRING	NA	trustStore.p12	M	Truststore file
initContainers.cacert.key	STRING	NA	CA_CERT_FILE	M	Cacert key file
initContainers.cacert.value	STRING	NA	cacert.pem	M	Cacert file
ssl.intraTlsEnabled	BOOLEAN	true/false	false	M	Enable internal service TLS
ssl.mTLS	BOOLEAN	true/false	false	M	Enable MTLS support for internal OCNADD services

Table 3-2 (Cont.) Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ssl.kafkaCipherSuites	STRING	NA	"TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,TLS_AES_256_GCM_SHA384,TLS_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384"	M	Supported Cipher Suites for Kafka Broker service in Data Director

Table 3-2 (Cont.) Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ssl.tlsCipherSuites	STRING	NA	"TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,TLS_AES_256_GCM_SHA384,TLS_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384"	M	Support Cipher Suites for Internal Services of Data Director
ssl.CERT_EXPIRY_CRONEXPRESSION	STRING	NA	0 0 * * *	M	Schedule for Cron Expression that will check certificate expiry at midnight everyday.
ssl.CERT_EXPIRY_CRONJOB	BOOLEAN	true/ false	true	M	Enable cronjob schedule to check certificate expiry.
acl.genericAclAllowed	BOOLEAN	true/ false	false	M	No need to change this flag here, genericAclAllowed=true will be used in upgrade --set command to restrict the generic ACL creation.
acl.kafkaClientAuth	STRING	none/ required	none	M	This Property is to enable or disable MTLS in Kafka.
acl.aclNotAllowed	BOOLEAN	true/ false	true	M	This Property is used to turn on or off the Kafka ACL's.
cluster.domainName	STRING	NA	occne-ocdd	M	Domain name of the setup

Table 3-2 (Cont.) Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
cluster.clusterName	STRING	NA	occne-ocdd	M	Default cluster name of setup
cluster.secret.name	STRING	NA	db-secret	M	Database Secret name where DB credentials are stored
cluster.mysqlNameSpace.name	STRING	NA	occne-cndbtierone	M	cnDBTier namespace
cluster.mysqlPod	STRING	NA	ndbmysqld-0	M	cnDBTier Pod Name
cluster.database.db_ip	STRING	NA	mysql-connectivity-service.occne-cndbtierone	M	Hostname or IP of cnDBTier
cluster.database.db_port	INTEGER	NA	3306	M	DB Port
cluster.database.configuration_db	STRING	NA	configuration_schema	M	Configuration Service Schema Name
cluster.database.alarm_db	STRING	NA	alarm_schema	M	Alarm Service Schema Name
cluster.database.health_db	STRING	NA	healthdb_schema	M	Health Service Schema Name
cluster.storageClass	STRING	NA	standard	M	Storage Class Name
cluster.nameSpace.name	STRING	NA	ocnadd-deploy	M	OCNADD Namespace
cluster.serviceAccount.create	BOOLEAN	true/ false	true	M	To create a ServiceAccount (true/false)
cluster.serviceAccount.name	STRING	NA	ocnadd	M	Name of the service Account
cluster.clusterRole.create	BOOLEAN	true/ false	true	M	To create clusterRole (true/false)
cluster.clusterRole.name	STRING	NA	ocnadd		Name of the clusterRole
cluster.clusterRoleBinding.create	BOOLEAN	true/ false	true	M	To create clusterRoleBinding (true/false)
cluster.clusterRoleBinding.name	STRING	NA	ocnadd		Name of the clusterRoleBinding
cluster.terminationGracePeriodSeconds	INTEGER	NA	5	O	Pod grace termination

Table 3-2 (Cont.) Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
cluster.imagePullSecret.enable	BOOLEAN	true/ false	false	M	Image Pull secret creation
cluster.imagePullSecret.name	STRING	NA	regcred	O	Set to regcred if cluster.imagePullSecret.enable is true
cluster.ALARM_PURGE_DAYS	INTEGER	NA	7	M	Alarm Purge in Days
cluster.kafka.ocnadd_kafka_bootstrap_servers	STRING	NA	kafka-broker:9092	M	Bootstrap server for PLAINTEXT
cluster.kafka.ocnadd_kafka_bootstrap_servers_ssl	STRING	NA	kafka-broker:9093	M	Bootstrap server for SSL
cluster.kafka.ocnadd_kafka_bootstrap_servers_sasl	STRING	NA	kafka-broker:9094	M	Bootstrap server for SASL
cluster.prometheusScrapePort	INTEGER	NA	9000	O	Port to scape metrics required if metrics enabled
cluster.prometheusPortName	STRING	NA	cnc-metrics	O	Role required to define in alert rules yaml
cluster.max_latency	FLOAT	NA	0.05	M	Max latency range of 50ms
cluster.memory_threshold	INTEGER	[0-100]	90	M	Max Threshold limit for memory
cluster.cpu_threshold	INTEGER	[0-100]	85	M	CPU max threshold limit
cluster.mps	INTEGER	NA	100000	M	Default MPS rate
cluster.serviceMonitorLabel	STRING	NA	occne-kube-prom-stack	M	service monitor label to scrape metrics
cluster.prometheus_url	STRING	NA	http://localhost:9000/cluster-name/prometheus/api/v1/query_range	M	Prometheus URL to scrape metrics
network.policy.enable	BOOLEAN	true/ false	false	M	Network Policy enable for intercommunication of OCNADD services
network.ingress.denyall	BOOLEAN	true/ false	false	C	Deny all ingress traffic

Table 3-2 (Cont.) Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
network.ingress.kafka	BOOLEAN	true/ false	true	C	Allow ingress traffic for kafka
network.ingress.aggregation	BOOLEAN	true/ false	true	C	Allow ingress traffic for aggregation service
network.ingress.filter	BOOLEAN	true/ false	true	C	Allow ingress traffic for filter service
network.ingress.adapter	BOOLEAN	true/ false	true	C	Allow ingress traffic for adapter service
network.ingress.egw	BOOLEAN	true/ false	true	C	Allow ingress traffic for egress service
network.ingress.config	BOOLEAN	true/ false	true	C	Allow ingress traffic for configuration service
network.ingress.alarm	BOOLEAN	true/ false	true	C	Allow ingress traffic for alarm service
network.ingress.health	BOOLEAN	true/ false	true	C	Allow ingress traffic for health monitoring service
network.ingress.admin	BOOLEAN	true/ false	true	C	Allow ingress traffic for admin service
network.ingress.namespaces	STRING	NA	- occne-infra - occncc	C	Network communication between allowed namespaces
network.ingress.external.enable	BOOLEAN	true/ false	false	C	Allow kafka LoadBalancer IP to be created
network.ingress.external.cidrs	STRING	NA	- 10.0.0.0/8	C	Cidr for network communication
network.egress.denyall	BOOLEAN	true/ false	false	C	Deny egress traffic
deployment.centralized	BOOLEAN	true/ false	true	M	Whether to use centralized mode of deployment
deployment.management	BOOLEAN	true/ false	true	M	Whether to deploy management group services when centralized deployment model is enabled. Will deploy worker group services if false.

Table 3-2 (Cont.) Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
deployment.management_namespace	STRING	-	Valid namespace name	M	Specify the managment namespace when deploying the worker group services. **Centralized Mode:** For Management Group Deployment: • When `deployment.management = true` – `deployment.management_namespace` should match `cluster.namespace.name`. For Worker Group Deployment: • When `deployment.management = false` – `deployment.management_namespace` should equal the namespace of the management group. This parameter facilitates communication between services in the worker group and the management group of services. **Non-Centralized Mode:** In this mode: • Ensure this parameter is defined and set to the value of `cluster.namespace.name`.
deployment.nonCenToCen_upgrade	BOOLEAN	true/ false	false	M	True if upgrading from non-centralized to centralized. Default is False (direct installation)
deployment.primary_site	BOOLEAN	true/ false	false	C	This parameter is required only in case the redundancy agent service is enabled. It depicts if the configured site is primary or not.
deployment.primary_agent_ip	STRING	NA	-	C	This parameter is required only in case the redundancy agent service is enabled. It is configured in the secondary site and denotes the primary site redundancy agent IP address or service FQDN.

Table 3-2 (Cont.) Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
kafka.kraftEnabled	BOOLEAN	true/ false	true	C	This parameter is required only when Kafka is deployed in KRaft mode. The default value is 'false'. Set this parameter to 'true' only during a fresh install when KRaft mode is selected. It must remain 'false' during a Data Director upgrade (Zookeeper mode) or during migration from Zookeeper to KRaft. If the source release was already using KRaft mode during an upgrade, this parameter must remain 'true'.
kafka.spawnKraftController	BOOLEAN	true/ false	false	C	This parameter indicates whether the KRaft controller should be spawned during the migration process. It should only be enabled when running the migration procedure as defined in the *Oracle Communications Network Analytics Data Director User Guide*.
kafka.migrationBroker	BOOLEAN	true/ false	false	C	Parameter to indicate that the migration of Kafka brokers to Kraft controllers. This should only be enabled while running the migration procedure defined in the User guide
kafka.kraftBroker	BOOLEAN	true/ false	false	C	Parameter to indicate that the migration of Kafka brokers to Kraft. This should only be enabled while running the migration procedure defined in the *Oracle Communications Network Analytics Data Director User Guide*.
kafka.finalizeMigration	BOOLEAN	true/ false	false	C	Parameter to indicate that the migration to Kraft can now be finalized. This should only be enabled while running the migration procedure defined in the *Oracle Communications Network Analytics Data Director User Guide*.
kafka.removeZookeeper	BOOLEAN	true/ false	false	C	Parameter to indicate that zookeeper can now be deprovisioned after the migration to Kraft is finalized. This should only be enabled while running the migration procedure defined in the *Oracle Communications Network Analytics Data Director User Guide*.

Table 3-2 (Cont.) Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
kafka.occmZookeeperClientUUID	STRING	-	false	C	UUID value of Zookeeper client certificate in OCCM. If OCCM is enabled during migration from Zookeeper mode to KRaft Mode then this value is used to delete the CLIENT Certificate of Zookeeper in OCCM
kafka.occmZookeeperServerUUID	STRING	-	false	C	UUID value of Zookeeper server certificate in OCCM. If OCCM is enabled during migration from Zookeeper mode to KRaft Mode then this value is used to delete the SERVER Certificate of Zookeeper in OCCM
admin.OCNADD_UPGRADE_WG_NS	STRING	NA	[ocnadd-deploy-wg1,ocnadd-deploy-wg2]	M	This parameter is a comma-separated list of worker group namespaces to update using Helm.
kafkaReplicas	INTEGER	-	4	M	The parameter to change the replicas for the Kafka broker.
cnlb.consumeradapter.enable	BOOLEAN	true/ false	false	C	If set to `true`, egress traffic segregation will be enabled for consumer feeds. This determines whether an Egress NAD should be attached to consumer adapters. Recommended only for OCCNE with CNLB support.
cnlb.ingressadapter.enable	BOOLEAN	true/ false	false	C	If set to `true`, ingress traffic segregation and external access will be enabled for the ingress adapter. This determines whether an Ingress NAD should be attached. Recommended only for OCCNE with CNLB support.
cnlb.ocnaddredundancyagent.enable	BOOLEAN	true/ false	false	C	If set to `true`, external access will be enabled for the Redundancy Agent. This determines whether CNLB annotations should be applied to redundancy deployments. Recommended only for OCCNE with CNLB support.
cnlb.ocnaddredundancyagent.network	STRING	NA	default/nf-oam-int1@nf-oam-int1	C	This must be the oam network with ingress definition. The entry means single network will be used by another site's Redundancy Agent for ingress communication. Update the network from the generated CNLB annotation, Given value is an example need to be updated as per /var/occne/cluster/$OCCNE_CLUSTER/artifacts/cnlbGenAnnotations.py script.

Table 3-2 (Cont.) Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
cnlb.ocnaddredundancyagent.externalIP	STRING	NA	10.1.1.1	C	Specifies the single external IP required for communication between Redundancy Agents in a two-site setup. Replace with the actual external IP.
cnlb.kafkabroker.enable	BOOLEAN	true / false	false	C	If set to `true`, external access for Kafka brokers will be enabled in a CNLB-enabled cluster.
cnlb.kafkabroker.networks	STRING	NA	"default/nf-sig2-int1@nf-sig2-int1, default/nf-sig2-int2@nf-sig2-int2, default/nf-sig2-int3@nf-sig2-int3, default/nf-sig2-int4@nf-sig2-int4"	C	Used to populate the `k8s.v1.cni.cncf.io/networks` annotation. Update this based on the "Enable CNLB for Kafka Broker" section of the *Oracle Communications Network Analytics Data Director User Guide*.
cnlb.kafkabroker.networks_extip	STRING	NA		C	Used to populate the `oracle.com.cnc/cnlb` annotation. Ensure the IP addresses are also included in the SAN entries for SSL communication.
env.controlPlaneNfList	STRING	NA	BSF,NRF,PCF	M	It enlists all the control plane NFs
env.proxyNfList	STRING	NA	SCP,SEPP	M	It enlist all the proxy NFs
extendedStorage.druid.enabled	BOOLEAN	true / false	false	M	Enable if Druid database as extended storage is available, else cnDBTier as extended storage is used by default.
extendedStorage.druid.druidTLsEnabled	BOOLEAN	true / false	true	O	The parameter depicts if TLS should be used for communication with Druid services. Default is true.
extendedStorage.druid.namespace	STRING	NA	ocnadd-druid	M	The namespace in which Druid cluster is deployed, if deployed in the same cluster as Data Director.
extendedStorage.druid.service_ip	STRING	NA	1.1.1.1	M	The loadbalancer of the Druid router service, this must only be changed if Druid is enabled else leave as is.

Table 3-2 (Cont.) Global Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
extendedStorage.druid.service_port	STRING	NA	8080	M	The port of the Druid router service.
extendedStorage.druid.secret_name	STRING	NA	ocnadd-druid-api-secret	M	The name of the secret containing the Druid API credentials.

3.4 Helm Hook Parameters

Table 3-3 Helm Hook Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ocnaddhelmhook.config.name	STRING	-	helmhook-configmap	M	Name of ConfigMap
ocnaddhelmhook.config.upgrade_name	STRING	-	helmhook-upgrade-configmap	M	Name of Upgrade ConfigMap
ocnaddhelmhook.config.rollback_name	STRING	-	helmhook-rollback-configmap	M	Name of Rollback ConfigMap
ocnaddhelmhook.name	STRING	-	ocnaddhelmhook	M	Helm Hook Name
ocnaddhelmhook.container.name	STRING	-	ocnaddhelmhook	M	Container Name of Helm Hook Job
ocnaddhelmhook.container.image	STRING	-	preinstall-image:2.1.6	M	Image used for preinstall hooks
ocnaddhelmhook.container.imagePullPolicy	STRING	IfNotPresent/ Always/ Never	IfNotPresent	M	Image Pull Policy
ocnaddpostinstallhelmhook.name	STRING	-	ocnaddpostinstallhelmhook	M	Post Install Hook Name
ocnaddpostupgradehelmhook.name	STRING	-	ocnaddpostupgradehelmhook	M	Post Upgrade Hook Name
ocnaddpostrollbackhelmhook.name	STRING	-	ocnaddpostrollbackhelmhook	M	Post Rollback hook name
ocnaddpreupgradehelmhook.name	STRING	-	ocnaddpreupgradehelmhook	M	Pre Upgrade Hook Name
ocnaddprerollbackhelmhook.name	STRING	-	ocnaddprerollbackhelmhook	M	Pre Rollback Hook Name

Table 3-3 (Cont.) Helm Hook Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ocnaddhelmhook.config.auto_backup_restore_cm	STRING	-	ocnadd-configmap-auto-backup-restore	M	Name of the automatic backup restore ConfigMap
ocnaddpreinstallworkergroup helmhook.name	STRING	-	ocnaddpreinstallworkergrouphelmhook	M	Name of the preinstall hook used during the installation of the worker group
ocnaddpreinstallworkergroup helmhook.retry_max_attempt	INTEGER	-	5	M	Maximum number of retries for getting the list of worker group names
ocnaddpreinstallworkergroup helmhook.retry_delay	INTEGER	-	15	M	Delay between retries
ocnaddpostinstallworkergroup helmhook.name	STRING	-	ocnaddpostinstallworkergrouphelmhook	M	Name of the postinstall hook used during the installation of the worker group
ocnaddpostinstallworkergroup helmhook.retry_max_attempt	INTEGER	-	5	M	Maximum number of retries for invoking the create worker group API
ocnaddpostinstallworkergroup helmhook.retry_delay	INTEGER	-	15	M	Delay between retries
ocnaddpostdeleteworkergrouphelmhook.name	STRING	-	ocnaddpostdeleteworkergrouphelmhook	M	Name of the postdelete hook used during the uninstallation of the worker group
ocnaddpostdeleteworkergrouphelmhook.retry_max_attempt	INTEGER	-	5	M	Maximum number of retries for invoking delete API for worker group
ocnaddpostdeleteworkergrouphelmhook.retry_delay	INTEGER	-	15	M	Delay between retries
ocnaddcopybackuppvctemptoorighelmhook.name	STRING	-	ocnaddcopybackuppvctemptoorighelmhook	M	Hook to copy backup from temporary to original PVC
ocnaddcopybackuppvcorigtotemphelmhook.name	STRING	-	ocnaddcopybackuppvcorigtotemphelmhook	M	Hook to copy backup from original to temporary PVC.
ocnaddcreatetemppvchelmhook.name	STRING	-	ocnaddcreatetemppvchelmhook	M	Hook to create temporary backup PVC during upgrade.

3.5 Aggregation Service Parameters

Table 3-4 Aggregation Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
BATCH_SIZE	INTEGER	-	65536	O	The maximum amount of data to be collected before sending the batch.
CONSUMER_POLL_MS	INTEGER	-	50	O	Polling time in ms for consumer
DD_METADATA_MAP_CACHE_EXPIRY_TIME_MS	INTEGER	30ms-30s	30	O	DD metadata cache expiry timer, default is 30ms
DD_METADATA_MAP_CACHE_SCHEDULER_TIME_MS	INTEGER	5ms-2s	5	O	This timer value depends on the attribute METADATA_MAP_CACHE_EXPIRY_TIME_MS. The timer value should be adjusted up or down corresponding to increase or decrease in METADATA_MAP_CACHE_EXPIRY_TIME_MS, default is 5ms.
ENABLE_AGGREGATION_COUNTER_METRICS	BOOLEAN	true,false	false	M	Enable metrics for Aggregation service
FETCH_MAX_WAIT_MS	INTEGER	-	100	O	The maximum amount of time the server will block before answering the fetch request if there isn't sufficient data to immediately satisfy the requirement given by fetch.min.bytes
HEARTBEAT_INTERVAL_MS	INTEGER	-	5000	O	The expected time between heartbeats to the consumer coordinator when using Kafka's group management facilities
KAFKA_FETCH_MAX_BYTES	INTEGER	-	576720	O	The maximum amount of data per-partition the server will return
KAFKA_FETCH_MIN_BYTES	STRING	-	1	O	The minimum amount of data per-partition the server will return
KAFKA_MAX_AGE_CONFIG	INTEGER	-	7500	M	The period of time in milliseconds after which we force a refresh of metadata.
KAFKA_MAX_PARTITIONS_FETCH_BYTES	INTEGER	-	104858	O	The maximum amount of data per-partition the server will return.
KAFKA_PRODUCER_SSL_CLIENT_AUTH	BOOLEAN	true, false	false	M	Kafka SSL client authentication.
KAFKA_SOCKET_BYTES_BUFFER	INTEGER	-	104857	O	Kafka Socket Buffer setting for consumer
LINGER_MS	INTEGER	-	1	O	The time to wait before sending messages out to Kafka

Table 3-4 (Cont.) Aggregation Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
logging.level.com.oracle.cgbu.cne.ocdd	STRING	[INFO,WARN,DEBUG,ERROR]	INFO	O	To set the log level of the OCDD package level
logging.level.com.oracle.cgbu.cne.ocnadd	STRING	[INFO,WARN,DEBUG,ERROR]	INFO	O	To set the log level of the OCNADD package level
MAX_POLL_INTERVAL_MS	INTEGER	-	30000	O	The maximum delay between invocations of poll() when using consumer group management
MAX_POLL_RECORDS	INTEGER	-	300	O	The maximum number of records returned in a single call to poll()
MESSAGE_REORDERING_INCOMPLETE_TRANSACTION_METRICS_ENABLE	BOOLEAN	true/ false	false	O	The parameter enables the Metric to check/count missing/inordered messages of transactions for MESSAGE_SEQUENCING_TYPE=TRANSACTION/REQUEST_RESPONSE
MESSAGE_SEQUENCING_TYPE	INTEGER	[NONE, TIME_WINDOW, TRANSACTION ,REQUEST_RESPONSE]	NONE	M	NONE: No message sequencing. TIME_WINDOW: Messages received within a window time for each partition will be sorted separately based on timestamp and streamed to Kafka topic. TRANSACTION: In-order messages received for each transaction within TRANSACTION_MSG_SEQUENCING_EXPIRY_TIMER will be sorted separately and streamed to Kafka topic. REQUEST_RESPONSE: In-order Request (RxRequest and TxRequest) and/or Response pair (RxResponse and TxResponse) messages received for each transaction within REQUEST_RESPONSE will be sorted separately and streamed to Kafka topic.
OCNADD_AGG_REDUNDANCY_DELAY_MS	INTEGER	-	1000	C	Delay before starting task to check Kafka ingress rate. This parameter is required when the two-site redundancy feature is enabled.
OCNADD_AGG_REDUNDANCY_INTERVAL_MS	INTEGER	-	250	C	Interval between tasks to check Kafka ingress rate. This parameter is required when the two-site redundancy feature is enabled.
OCNADD_AGGREGATION_LOG_NETTY	STRING	-	INFO	O	Default Netty Log level set for the application.

Table 3-4 (Cont.) Aggregation Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_AGGREGATION_LOG_ROOT	STRING	-	INFO	O	Default Log level set for the application.
OCNADD_AGGREGATION_SERVICE_TOPIC_RETRIES_THRESHOLD	INTEGER	-	120000	O	Retry Threshold for TOPIC reachability
OCNADD_KAFKA_BOOTSTRAP_SERVER_SSL	STRING	-	kafka-broker:9093	M	Kafka Boot strap server address for SSL
OCNADD_KAFKA_SASL_MECHANISM	STRING	-	PLAIN	M	Kafka SASL Mechanism.
OCNADD_KAFKA_SECURITY_PROTOCOL_SASL	STRING	-	SASL_SSL	M	Kafka security protocol
OCNADD_KAFKA_SECURITY_PROTOCOL_SSL	STRING	-	SSL	M	Kafka SSL Mechanism.
OCNADD_TRUST_KEYSTORE	BOOLEAN	true, false	true	M	Enable to secure connection via OCWeb Client.
ocnaddbsfaggregation.maxReplicas	INTEGER	-	1	M	The maximum number of replicas required for BSF aggregation service instance
ocnaddbsfaggregation.minReplicas	INTEGER	-	1	M	The minimum number of replicas required for BSF aggregation service instance
ocnaddbsfaggregation.name	STRING	-	ocnaddbsfaggregation	M	Name of the application
ocnaddnonoracleaggregation.resources.limit.cpu	INTEGER	-	3	M	Number of max CPU for non-Oracle aggregation
ocnaddnonoracleaggregation.resources.limit.ephemeralstorage	STRING	-	500Mi	M	Ephemeral Storage for non-Oracle aggregation
ocnaddnonoracleaggregation.resources.limit.memory	STRING	-	2Gi	M	Maximum memory allocation for non-Oracle aggregation
ocnaddnonoracleaggregation.maxReplicas	INTEGER	-	1	M	The maximum number of replicas required for ocnaddnonoracleaggregation aggregation service instance

Table 3-4 (Cont.) Aggregation Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ocnaddnonoracleaggregation.minReplicas	INTEGER	-	1	M	The minimum number of replicas required for ocnaddnonoracleaggregation aggregation service instance
ocnaddnrfaggregation.name	STRING	-	ocnaddnrfaggregation	M	Name of the application
ocnaddnrfaggregation.resources.limit.cpu	INTEGER	-	3	M	Number of maximum CPUs for NRF aggregation
ocnaddnrfaggregation.resources.limit.ephemeralstorage	STRING	-	500Mi	M	Ephemeral Storage for NRF aggregation
ocnaddnrfaggregation.resources.limit.memory	STRING	-	2Gi	M	Max Memory limit for NRF aggregation
ocnaddnrfaggregation.maxReplicas	INTEGER	-	1	M	The maximum number of replicas required for NRF aggregation service instance
ocnaddnrfaggregation.minReplicas	INTEGER	-	1	M	The minimum number of replicas required for NRF aggregation service instance
ocnaddpcfaggregation.name	STRING	-	ocnaddpcfaggregation	M	Name of the application
ocnaddpcfaggregation.resources.limit.cpu	INTEGER	-	2	M	Number of max CPU for PCF Aggregation
ocnaddpcfaggregation.resources.limit.ephemeralstorage	STRING	-	500Mi	M	Ephemeral Storage for PCF Aggregation
ocnaddpcfaggregation.resources.limit.memory	STRING	-	2Gi	M	Max Memory limit for PCF Aggregation
ocnaddscpaggregation.minReplicas	INTEGER	-	1	M	The minimum number of replicas required for SCP aggregation service instance
ocnaddscpaggregation.name	STRING	-	ocnaddscpaggregation	M	Name of the application
ocnaddscpaggregation.resources.limit.cpu	INTEGER	-	3	M	Number of max CPU for SCP aggregation
ocnaddscpaggregation.resources.limit.ephemeralstorage	STRING	-	500Mi	M	Ephemeral Storage for SCP aggregation

Table 3-4 (Cont.) Aggregation Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ocnaddscpaggregation.resources.limit.memory	STRING	-	2Gi	M	Max Memory limit for SCP aggregation
ocnaddseppaggregation.maxReplicas	INTEGER	-	2	M	The maximum number of replicas required for SEPP aggregation service instance
ocnaddseppaggregation.minReplicas	INTEGER	-	1	M	The minimum number of replicas required for SEPP aggregation service instance
ocnaddseppaggregation.name	STRING	-	ocnaddseppaggregation	M	Name of the application
ocnaddseppaggregation.resources.limit.cpu	INTEGER	-	3	M	Number of max CPU for SEPP aggregation
ocnaddseppaggregation.resources.limit.ephemeralstorage	STRING	-	500Mi	M	Ephemeral Storage for SEPP aggregation
ocnaddseppaggregation.resources.limit.memory	STRING	-	2Gi	M	Max Memory limit for SEPP aggregation
REQUEST_RESPONSE_MSG_SEQUENCING_EXPIRY_TIMER	INTEGER	[5-500]ms	10ms	C	When MESSAGE_SEQUENCING_TYPE = REQUEST_RESPONSE
REQUEST_TIMEOUT_MS	INTEGER	-	1000	O	The configuration controls the maximum amount of time the client will wait for the response of a request
SESSION_TIME_OUT	INTEGER	-	15000	O	The timeout used to detect client failures when using Kafka's group management facility.
SRC_FEED_METADATA_CLEAN_DELAY_SEC	INTEGER	-	86400	C	Initial delay in cleaning the metadata cache
SRC_FEED_METADATA_CLEAN_PERIOD_SEC	INTEGER	-	86400	C	Interval after the metadata cache is cleaned
SRC_FEED_METADATA_HISTORY_LENGTH	INTEGER	-	20	C	The number of the metadata maintained in the cache
TOTAL_FORWARDED_MESSAGE_METRICS_ENABLE	BOOLEAN	true/ false	false	O	The parameter enables the Metric to get total count of forwarded messages by aggregation service
TRANSACTION_MSG_SEQUENCING_EXPIRY_TIMER	INTEGER	[20ms-30s]	200ms	C	When MESSAGE_SEQUENCING_TYPE = TRANSACTION

Table 3-4 (Cont.) Aggregation Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
WINDOW_MSG_SEQUENCING_EXPIRY_TIMER	INTEGER	[5-500]ms	10ms	C	When MESSAGE_SEQUENCING_TYPE = TIME_WINDOW
SEPP_DUAL_SITE_ENABLED	BOOLEAN	true/ false	false	O	Set it to true only when SEPP dual-site deployment is enabled • **Role:** It changes DD-metadata transaction handling for SEPP flows. – false (normal/single-site): metadata cache is typically cleared on TX_RESPONSE. – true (dual-site): cache is retained longer and completed on RX_RESPONSE, so metadata is preserved correctly across dual-site SEPP message paths.

3.6 Configuration Service Parameters

Table 3-5 Configuration Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ADAPTER_DEFAULT_INGRESS_ACKS	STRING	-	all	O	The non oracle producer acknowledgement value
ADAPTER_DEFAULT_INGRESS_LIMIT	INTEGER	-	101874	O	The buffer in bytes in the non oracle ingress adapter feed for the http connection
ADAPTER_DEFAULT_INGRESS_PARTITIONS	INTEGER	-	30	O	The number of partitions for the non oracle ingress feed adapter
ADAPTER_DEFAULT_INGRESS_REPLICATION_FACTOR	INTEGER	-	1	O	The non oracle topic replication factor
ADAPTER_DEFAULT_INGRESS_RETENTION_MS	INTEGER	-	600000	O	The non oracle topic retention time in milisec

Table 3-5 (Cont.) Configuration Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ADAPTER_DEFAULT_INGRESS_RETRY	INTEGER	-	3	O	The number of retries for the non oracle ingress adapter feed
ADAPTER_DEFAULT_INGRESS_TOPIC	STRING	-	NON_ORACLE	O	The topic name for the non oracle ingress feed adapter
ADMIN_RESPONSE_TIMEOUT_MS	INTEGER	-	60000	O	The timeout in receiving response from the admin service
ADMIN_RETRY_DELAY_MS	INTEGER	-	5000	O	Delay between successive retries while calling admin service
ADMIN_RETRY_MAXATTEMPT	INTEGER	-	3	O	Maximum number of retries attempted while calling admin service
CONFIG_NOTIFICATION_RETRY_DELAY	INTEGER	-	50000	O	Delay between successive notification retries to be attempted
CONFIG_NOTIFICATION_RETRY_MAXATTEMPT	INTEGER	-	3	O	The number of retries for the notification to the subscribed service instance
CONFIG_NOTIFICATION_SCHEDULER_INITIAL_DELAY_MS	STRING	-	200ms	O	The initial delay for the Notification sender thread in the configuration service
CONFIG_NOTIFICATION_SCHEDULER_PERIOD_MS	STRING	-	200ms	O	The notification sender thread fixed delay upon which it will check for the pending notiifications
CONFIGURATION_FILTER_METHODS	STRING	[GET,POST,PUT,DELETE,PATCH,CONNECT,OPTIONS,TRACE]	GET,POST,PUT,DELETE,PATCH,CONNECT,OPTIONS,TRACE	O	The configuration of methods on which the filter is possible
EXPORT_CONFIG_PURGE_TIMER_HR	INTEGER	[1-48]	24	O	The purge timer for the export configuration
EXPORT_PURGE_SCHEDULER_DELAY_MS	INTEGER	-	30000	O	The delay in milisec after which the purging of the export configuration will be checked.
logging.level.com.oracle.cgbu.cne.ocdd	STRING	-	INFO	O	Logging level for Common OCNADD services
logging.level.com.oracle.cgbu.cne.ocnadd	STRING	-	INFO	O	Logging level for OCNADD services
logging.level.reactor.netty	STRING	-	INFO	O	Netty logging level

Table 3-5 (Cont.) Configuration Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
logging.type	STRING	STDOUT/ LOGJSON	STDOUT	O	Logging type Standard Output or JSON format
MAX_ACTION_TYPE_IN_FILTER	INTEGER	-	2	O	Maximum number of action type in a filter with chaining
MAX_ALLOWED_FILTERS	INTEGER	-	30	O	Maximum number of allowed filters
MAX_CORRELATION_CONFIGURATION_SUPPORTED	INTEGER	-	3	O	Maximum number of correlation feeds allowed
MAX_EXPORT_CONFIGURATION_SUPPORTED	INTEGER	-	3	O	The simultaneous number of export configuration supported on the Data Director
MAX_FILTERS_ASSOC_WITH_APP	INTEGER	-	4	O	Maximum number of filters associated with a single app
MAX_GLOBALL3L4_ROW_SIZE	INTEGER	-	500	O	Maximum size of L3L4 rows
MAX_VALUES_IN_FILTER_ATTRIB	INTEGER	-	20	O	Maximum number of values allowed in filter attributes
OCNADD_CONFIG_FILTER_NOTIF_DELAY_SEC	INTEGER	-	3	O	The delay between the notification to be sent to the filter service instance
OCNADD_CONFIG_TIMEOUT_CONNECT	STRING	-	30s	O	The timeout for receiving no response from the server on a http connection
OCNADD_CONFIG_TIMEOUT_READ_IDLE	STRING	-	30s	O	The setting (which defaults to 30 sec) dictates when to close a connection after it becomes idle
OCNADD_CONFIG_TIMEOUT_SSL_HANDSHAKE	STRING	-	30s	O	The SSL connection handshake timeout in configuration service
OCNADD_CONFIGURATION_LIVENESS_DELAY	INTEGER	-	600	M	Configuration Svc Liveness Param: this field tells the kubelet that it should wait for mentioned seconds before performing the first probe.
OCNADD_CONFIGURATION_LIVENESS_FAILURE	INTEGER	-	50	M	Configuration Svc Liveness Param: For the case of a liveness probe, triggers a restart for that specific container if the container failed to start for given no of failure retries.
OCNADD_CONFIGURATION_LIVENESS_PERIOD_SECONDS	INTEGER	-	15	M	Configuration Svc Liveness Param: this field specifies that the kubelet should perform a liveness probe every given no of seconds.

Table 3-5 (Cont.) Configuration Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_CONFIGURATION_LIVENESS_TIMEOUT	INTEGER	-	20	M	Configuration Svc Liveness Param: Number of seconds after which the probe times out.
OCNADD_MAX_EXTERNAL_KAFKA_FEEDS	INTEGER	-	2	O	Maximum number of allowed external Kafka Feed
OCNADD_MAX_WORKERGROUP_THRESHOLD_PERCENTAGE	INTEGER	-	80	O	The percentage threshold for the maximum worker group supported
OCNADD_REDUNDANCY_NOTIFY_DELAY_SEC	INTEGER	-	120	C	Delay before updating mode of worker group from ACTIVE to STANDBY and vice-versa
OCNADD_TRUST_KEYSTORE	BOOLEAN	-	false	O	Truststore enable for Configuration Service
ocnaddconfiguration.name	STRING	-	ocnaddconfiguration	M	Name of configuration service
resources.limits.cpu	INTEGER	-	1	M	Number of maximum CPUs for each configuration service instance
resources.limits.ephemeralstorage	STRING	-	500Mi	M	Ephemeral Storage for each configuration service
resources.limits.memory	STRING	-	1Gi	M	Max Memory limit for each configuration service instance
resources.requests.cpu	INTEGER	-	1	M	Minimum number of CPUs required for each configuration service instance
resources.requests.ephemeralstorage	STRING	-	100Mi	M	minimum Ephemeral Storage required for each configuration instance
resources.requests.memory	STRING	-	1Gi	M	minimum Memory required for each configuration instance
SITE_REDUNDANCY_RESPONSE_TIMEOUT	INTEGER	-	30000	O	The timeout to get a response from the redundancy agent for the request sent from the configuration service
SITE_REDUNDANCY_RETRY	INTEGER	-	3	O	The number of retries towards the redundancy agent service
SITE_REDUNDANCY_RETRY_DELAY	INTEGER	-	100	O	Delay between successive retries to be attempted

3.7 Health Monitoring and Alarm Service Parameters

Table 3-6 Health Monitoring Service Parameters

Parameter Name	Data Type	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ocnaddhealthmonitoring.name	STRING	ocnaddhealthmonitoring	M	Health monitoring service name
HEALTH_MONITORING_TIMER	INTEGER	5000	O	Timer to check Health of integrated services
HEALTH_METRICS_SCHEDULED	BOOLEAN	true	O	Scheduler for metrics
HEALTH_METRICS_TIMER	INTEGER	120000	O	Timer for health metrics
HEALTH_PURGE_TIME_HR	INTEGER	5	O	Health profile purging timer in hour
HEALTH_MONITORING_CPUTHRESHOLD	INTEGER	75	M	CPU threshold to raise alarm
HEALTH_MONITORING_MEMORYTHRESHOLD	INTEGER	95	M	Memory threshold to raise alarm
Logging Properties				
HEALTH_LOG_HTTPCLIENT	STRING	INFO	O	Set Default Log level for HTTP client
HEALTH_LOG_SPRING_WEB	STRING	INFO	O	Set Default Log level for Spring Web
logging.level.com.oracle.cgbu.cne.ocnadd	STRING	INFO	O	Logging level for Health Monitoring OCNADD Service
logging.level.com.oracle.cgbu.cne.ocdd	STRING	INFO	O	Logging level for Common OCNADD Service
HEALTH_APPLICATION_LOG_LEVEL	STRING	INFO	O	Set application logger level
HEALTH_LOG_REQUEST_DETAILS	BOOLEAN	true	O	If this parameter is set to true, the health request details will be logged
HEALTH_LOG_ROOT	STRING	WARN	O	Root Level Logger
resources.limits.cpu	INTEGER	1	M	Number of maximum CPUs for each health monitoring service instance

Table 3-6 (Cont.) Health Monitoring Service Parameters

Parameter Name	Data Type	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
resources.limits.memory	STRING	1Gi	M	Max Memory limit for each health monitoring service instance
resources.limits.ephemeralstorage	STRING	500Mi	M	Ephemeral Storage for each health monitoring service
resources.requests.cpu	INTEGER	1	M	Minimum number of CPUs required for each health monitoring service instance
resources.requests.memory	STRING	1Gi	M	minimum Memory required for each health monitoring instance
resources.requests.ephemeralstorage	STRING	200Mi	M	minimum Ephemeral Storage required for each health monitoring instance

Table 3-7 Alarm Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ocnaddalarm.name	STRING	-	ocnaddalarm	M	Application name for Alarm Service
logging.type	STRING	-	STDOUT	O	Logging Type Standard Outpuut or JSON Format
OCNADD_TRUST_KEYSTORE	BOOLEAN	true/false	true	O	Trust Keystore Enable
logging.level.com.oracle.cgbu.cne.ocnadd	STRING	-	INFO	O	Logging level for Alarm OCNADD Service
logging.level.com.oracle.cgbu.cne.ocdd	STRING	-	INFO	O	Logging level for Common OCNADD Service
resources.limits.cpu	INTEGER	-	1	M	Number of maximum CPUs for each alarm service instance

Table 3-7 (Cont.) Alarm Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
resources.limits.memory	STRING	-	1Gi	M	Max Memory limit for each alam service instance
resources.limits.ephemeralstorage	STRING	-	200Mi	M	Ephemeral Storage for each alarm service
resources.requests.cpu	INTEGER	-	1	M	Minimum number of CPUs required for each alarm service instance
resources.requests.memory	STRING	-	1Gi	M	minimum Memory required for each alarm instance
resources.requests.ephemeralstorage	STRING	-	200Mi	M	minimum Ephemeral Storage required for each alarm instance
MAX_ALARM_RETRIEVE_COUNT	INTEGER	-	1000	O	Parameter to fetch maximum of 1000 latest alarm for each severity type
OCNADD_ALARM_LIVENESS_DELAY	INTEGER	-	60	M	Alarm Svc Liveness Parameter: this field tells the kubelet that it should wait for mentioned seconds before performing the first probe.
OCNADD_ALARM_LIVENESS_PERIOD_SECONDS	INTEGER	-	15	M	Alarm Svc Liveness Parameter: this field specifies that the kubelet should perform a liveness probe every given no of seconds.

Table 3-7 (Cont.) Alarm Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_ALARM_LIVENESS_FAILURE	INTEGER	-	5	M	Alarm Svc Liveness Parameter: For the case of a liveness probe, triggers a restart for that specific container if the container failed to start for given no of failure retries.
OCNADD_ALARM_LIVENESS_TIMEOUT	INTEGER	-	20	M	Alarm Svc Liveness Parameter: Number of seconds after which the probe times out.

3.8 Admin Service Parameters

Table 3-8 Admin Service Parameters

Parameter Name	Data Type	Range	Default Value	M/O/C	Description
ocnadd.admin.name	STRING	-	ocnaddadminservice	M	Application name for the Admin Service
OCNADD_ADMIN_PORT	INTEGER	-	9181	M	Application port for the Admin Service
resources.limits.cpu	INTEGER	-	1	M	Maximum number of CPUs for each Admin Service instance
resources.limits.memory	STRING	-	1Gi	M	Maximum memory limit for each Admin Service instance
resources.limits.ephemeralstorage	STRING	-	200Mi	M	Ephemeral storage limit for each Admin Service instance

Table 3-8 (Cont.) Admin Service Parameters

Parameter Name	Data Type	Range	Default Value	M/O/C	Description
resources.requests.cpu	INTEGER	-	1	M	Minimum number of CPUs required for each Admin Service instance
resources.requests.memory	STRING	-	1Gi	M	Minimum memory required for each Admin Service instance
resources.requests.ephemeral storage	STRING	-	200Mi	M	Minimum ephemeral storage required for each Admin Service instance
logging.level.com.oracle.cgbu.cne.ocnadd	STRING	-	INFO	O	Logging level for OCNADD services
logging.level.com.oracle.cgbu.cne.ocdd	STRING	-	INFO	O	Logging level for common OCNADD services
ADMIN_SVC_LOGGER_KAFKA	STRING	[ON, OFF]	OFF	O	Whether to enable Kafka logging in the Admin Service
ADMIN_SVC_ROOT_LOGGER	STRING	-	WARN	O	Root logger level for the Admin Service
OCNADD_ADMIN_SVC_LIVENESS_DELAY	INTEGER	-	60	M	Time (in seconds) for the kubelet to wait before performing the first liveness probe
OCNADD_ADMIN_SVC_LIVENESS_PERIOD_SECONDS	INTEGER	-	15	M	Time interval (in seconds) between liveness probes by the kubelet
OCNADD_ADMIN_SVC_LIVENESS_FAILURE	INTEGER	-	5	M	Number of failed liveness probes before restarting the container

Table 3-8 (Cont.) Admin Service Parameters

Parameter Name	Data Type	Range	Default Value	M/O/C	Description
OCNADD_ADMIN_SVC_LIVENESS_TIMEOUT	INTEGER	-	20	M	Timeout (in seconds) for the liveness probe
OCNADD_ADMINSVC_HEALTH_RETRY_DELAY	INTEGER	-	2	M	Retry delay (in seconds) for the Admin Service connection to the health monitoring service
OCNADD_ADMINSVC_HEALTH_HB_TIMER	INTEGER	-	10000	M	Heartbeat interval (in milliseconds) for the Admin Service to the health monitoring service
OCNADD_ADMINSVC_HEALTH_RETRY_COUNT	INTEGER	-	1	M	Number of retry attempts for the Admin Service connection to the health monitoring service

3.8.1 Consumer Aapter Parameters

Table 3-9 Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
consumeradapter.resources.limits.cpu	INTEGER	-	3	M	Number of maximum CPUs for each admin service
consumeradapter.resources.limits.memory	STRING	-	6Gi	M	Max Memory limit for each admin service instance
consumeradapter.resources.limits.ephemeralstorage	STRING	-	1Gi	M	Ephemeral Storage for each admin service

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
consumeradapter.resources.requests.cpu	INTEGER	-	3	M	Minimum number of CPUs required for each admin service instance
consumeradapter.resources.requests.memory	STRING	-	6Gi	M	minimum Memory required for each Correlation instance
consumeradapter.resources.requests.ephemeralstorage	STRING	-	1Gi	M	minimum Ephemeral Storage required for each Correlation instance
OCNADD_CONSUMER_ADAPTER_PORT	INTEGER	-	9182	M	Application port for the consumer adapter
OCNADD_ADAPTER_HEALTH_RETRY_DELAY	INTEGER	-	2	M	Retry delay for the adapter service connection towards healt monitoring service
OCNADD_ADAPTER_HEALTH_HB_TIMER	INTEGER	-	10000	M	Heart beat timer in consumer adapter towards the health monitoring service
OCNADD_ADAPTER_HEALTH_RETRY_COUNT	INTEGER	-	1	M	Retry count for the adapter service connection towards health monitoring service
ALARM_CLIENT_RETRY_COUNT	INTEGER	-	3	M	Retry count for the adapter service connection towards alarm service

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
EGRESS_SSL_HANDSHAKE_TIMEOUT	INTEGER	-	5	O	SSL handshake timeout.
ENABLE_ADAPTER_COUNTER_METRICS	BOOLEAN	true/false	true	O	Enable Adapter Counter Metric.
OCNADD_EGRESS_NETWORK_ENABLE	BOOLEAN	true/false	false	O	Enable this parameter to true if traffic needs to be routed outside the cluster.
OCNADD_EGRESS_NETWORK_NAME_VALUE	STRING	-	oam	O	Name of the egress network configured in the CNE cluster.
OCNADD_CNC_ENABLE	STRING	-	True	O	Enable oracle.com.cnc network.
OCNADD_ADAPTER_MIN_REPLICAS_HTTP2	INTEGER	-	2	M	Minimum Replicas for HTTP2 Adapter
OCNADD_ADAPTER_MAX_REPLICAS_HTTP2	INTEGER	-	13	M	Max Replicas for HTTP2 Adapter
OCNADD_ADAPTER_MIN_REPLICAS_TCP	INTEGER	-	1	M	Minimum Replicas for TCP Adapter
OCNADD_ADAPTER_MAX_REPLICAS_TCP	INTEGER	-	9	M	Max Replicas for TCP Adapter
MAX_TCP_CONNECTION_PER_DEST	INTEGER	-	6	M	Max allowed TCP connection per destination
ADAPTER_KAFKA_FETCH_MIN_BYTES	INTEGER	-	0	O	The minimum amount of data the server should return for a fetch request
ADAPTER_KAFKA_FETCH_MAX_BYTES	STRING	-	57672000	O	The maximum amount of data the server should return for a fetch request

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ADAPTER_KAFKA_MAX_PARTITION_FETCH_BYTES	STRING	-	10485700	O	The maximum amount of data per-partition the server will return
ADAPTER_KAFKA_FETCH_MAX_WAIT_MS	INTEGER	-	40	O	The maximum amount of time the server will block before answering the fetch request
ADAPTER_KAFKA_SESSION_TIME_OUT	INTEGER	-	15000	O	The timeout used to detect client failures when using Kafka's group management facility
ADAPTER_KAFKA_HEARTBEAT_INTERVAL_MS	INTEGER	-	5000	O	The expected time between heartbeats to the consumer coordinator when using Kafka's group management facilities
ADAPTER_KAFKA_MAX_POLL_INTERVAL_MS	INTEGER	-	30000	O	The maximum delay between invocations of poll() when using consumer group management
ADAPTER_KAFKA_MAX_POLL_RECORDS	INTEGER	-	1500	O	The maximum number of records returned in a single call to poll()
ADAPTER_KAFKA_COMMIT_INT_CONFIG	INTEGER	-	30	O	The frequency in milliseconds that the consumer offsets are committed to Kafka

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ADAPTER_KAFKA_NUM_THREADS_CONFIG_HTTP2	INTEGER	-	9	O	The number of threads to run stream processing for http2 connections.
ADAPTER_KAFKA_NUM_THREADS_CONFIG_TCP	INTEGER	-	6	O	The number of threads to run stream processing for tcp connections.
ADAPTER_KAFKA_CONSUMER_POLL_MS	INTEGER	-	30	O	The amount of time in milliseconds to block waiting for input.
ADAPTER_KAFKA_RECEIVE_BUFFER_BYTES	STRING	-	10485700	M	The size of the TCP receive buffer (SO_RCVBUF) to use when reading data.
ADAPTER_LOG_LEVEL_KAFKA	STRING	[ON,OFF]	OFF	O	Whether to ON or OFF Kafka logs in Adapter Service.
OCNADD_ADAPTER_MAX_REPLICAS_TCP	INTEGER	-	2	O	MAX replicas for synthetic Adapter.
OCNADD_ADAPTER_LIVENESS_DELAY	INTEGER	-	60	M	AdapterService Liveness Param: this field tells the kubelet that it should wait for mentioned seconds before performing the first probe.
OCNADD_ADAPTER_LIVENESS_PERIOD_SECONDS	INTEGER	-	15	M	AdapterService Liveness Param: this field specifies that the kubelet should perform a liveness probe every given number of seconds.

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_ADAPTER_LIVENESS_FAILURE	INTEGER	-	5	M	AdapterService Liveness Param: For the case of a liveness probe, triggers a restart for that specific container if the container failed to start for given number of failure retries.
OCNADD_ADAPTER_LIVENESS_TIMEOUT	INTEGER	-	20	M	AdapterService Liveness Param: Number of seconds after which the probe times out.
CONFIG_SVC_DATASTREAM_OFFSET_DELAY_MS	INTEGER	-	3000	O	Delay in milliseconds between Retries to fetch the data stream offset from config service in case of failure.
ADAPTER_KAFKA_LISTCONSUMER_TIMEOUT_MS	INTEGER	-	30000	O	Timeout in milliseconds to list the Consumer Groups.
KAFKA_TOPIC_NO_OF_PARTITIONS	INTEGER	-	42	O	Default number of partitions that will be created for a topic when a new Kafka feed is created.
KAFKA_TOPIC_REPLICATION_FACTOR	INTEGER	-	1	O	Replication Factor for the Kafka Topic of Kafka Feed.
KAFKA_TOPIC_RETENTION_MS	INTEGER	-	300000	O	Retention Time for Kafka Topic.
KAFKA_TOPIC_SEGMENT_MS	INTEGER	-	300000	O	Retention Time for the Kafka topic segment

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
EXTERNAL_CONSUMER_SASL_PORT	INTEGER	-	9094	O	Kafka bootstrap server port number for SASL_SSL protocol.
KAFKA_FUTURE_TIMEOUT_MS	INTEGER	-	25000	O	Timeout to fetch the admin client response details.
VERIFY_KAFKA_CONNECTION_TIMEOUT_MS	INTEGER	-	10000	O	Timeout to verify Kafka connection in milliseconds.
KAFKAFEED_METRICS_SCHEDULED	BOOLEAN	-	true	O	To enable or disable metrics for Kafka Feeds.
KAFKAFEED_METRICS_TIMER	STRING	-	15s	O	Metrics timer for Kafka Feeds.
TCP_STREAM_RESET_ENABLED	BOOLEAN	-	false	O	To enable Kafka feed stream restart. This maybe required only when 3rd Party consumer is not working properly and frequently breaks connections with Synthetic Feed
TCP_STREAM_RESET_ELAPSED_TIME	INTEGER	-	60	O	The time in minutes to check if stream reset is required.
TCP_STREAM_RESET_FIXED_DELAY_MS	INTEGER	-	300000	O	Default 300 sec, Schedular Interval Time
TCP_STREAM_RESET_INI_DELAY_MS	INTEGER	-	150000	O	Default 150 sec, Schedular Initial Delay to Start
TCP_STREAM_RESTART_INTERVAL_MS	INTEGER	-	300000	O	Default 300 sec, interval to check for the TCP stream restart

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
TCP_CONN_INSPECTOR_ENABLED	BOOLEAN	true/false	true	O	The parameter to enable the watcher for the TCP connection in consumer adapter
OCNADD_INTERNAL_CLIENT_SSL_PROTOCOL	STRING	-	TLS	O	The SSL protocol used between the adapter and internal DD services communication
OCNADD_INTERNAL_CLIENT_SSL_PROTOCOLS	STRING	-	TLSv1.2,TLSv1.3	O	The version of TLS supported between adapter and internal DD services communication
OCNADD_INTERNAL_CLIENT_SSL_HANDSHAKE_TIMEOUT	STRING	-	30s	O	SSL handshake timeout between adapter and internal DD services communication
OCNADD_SSL_CIPHERS	STRING	-	TLS_AES_256_GCM_SHA384,TLS_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	O	Oracle Ciphers supported for the TLS communication
OCNADD_ADAPTER_CONFIG_RETRY_DELAY	STRING	-	5s	M	The retry interval between adapter and configuration service communication for subscription

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_ADAPTER_CONFIG_RETRY_DELETE_SUBS_REQ_ATTEMPT	STRING	-	3	M	The number of retries supported for sending the delete subscription request to configuration service by consumer adapter service
OCNADD_ADAPTER_CONFIG_RETRY_DELETE_SUBS_REQ_DELAY	STRING	-	3s	M	The retry interval supported for sending the delete subscription request to configuration service by consumer adapter service
EGRESS_TRUSTSTORE_ENABLED	BOOLEAN	true/false	true	M	The parameter to enable/ disable the truststore between consumer adapter and 3rd party application
EGRESS_SSL_CLIENT_AUTH	STRING	need/want	want	M	SSL Authentication mode to be supported between consumer adapter and 3rd party communication
EGRESS_SSL_HANDSHAKE_TIMEOUT	STRING	-	5s	M	SSL handshake timeout between adapter and 3rd party application communication

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
HTTP_ENABLE_SUBSCRIBE_API_ASYNC_PROCESSING	BOOLEAN	true/false	true	M	The parameter to enable the asynchronous processing of subscription request towards the configuration service
EGRESS_RESPONSE_IDLE_TIMEOUT_SEC	INTEGER	-	120	O	The setting (which defaults to 2 minutes) dictates when to close a connection after it becomes idle
EGRESS_RESPONSE_TIMEOUT_SEC	INTEGER	-	8	O	The amount of time it takes to actually receive the response back from the server, default is 8 sec
EGRESS_CHANNEL_TIMEOUT_SEC	INTEGER	-	6	O	Specifies the amount of time, in seconds, that the HTTP transport channel waits for a read request to complete on a socket after the first read occurs. Default is 6 sec
EGRESS_HTTP_FOLLOW_REDIRECTS	STRING	-	false	O	The parameter to indicate that consumer adapter does not want any redirections from the 3rd party applications
SO_SEND_BUF_BYTES_HTTP	STRING		16777216	O	The send socket buffer size for the HTTP connection towards 3rd party application

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
EGRESS_KEEPALIVE_IDLE	INTEGER	-	60	O	The keepalive will be sent after the connection is idle for the configured seconds
EGRESS_KEEPALIVE_INT	INTEGER	-	60	O	interval between two keepalive messages
EGRESS_KEEPALIVE_COUNT	INTEGER	-	10	O	The maximum number of keepalive packets that will be sent before assuming the connection is dead
EGRESS_HTTP_CLIENT_MAX_CONCURRENT_HTTP_CONN	INTEGER	-	100	O	The maximum number of concurrent HTTP connection that can be made by http client
EGRESS_HTTP_MAX_CONCURR_REQ_PER_HTTP_CONN	INTEGER	-	5	O	The maximum number of concurrent HTTP Requests that can be sent by http client on the single http connection
EGRESS_MAX_CONNECTION_POOL_IDLE	INTEGER	-	30	O	In the http client connection pool, the connections that are not currently in use but are maintained by the pool for reuse
EGRESS_HTTP_CLIENT_SHUTDOWN_QUIET_PERIOD_SEC	INTEGER	-	25	O	Sets the amount of quiet period for shutdown of client thread pools

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
EGRESS_HTTP_CLIENT_SHUTDOWN_TIMEOUT_SEC	INTEGER	-	30	O	Sets the amount of time to wait for shutdown of client thread pools
ADAPTER_TCP_CLIENT_POOL_MAX	INTEGER	-	1000	O	the maximum number of concurrent TCP connections a client can establish with a server
ADAPTER_TCP_CLIENT_CHANNEL_TIMEOUT	INTEGER	-	60	O	A TCP client channel timeout occurs when a TCP client doesn't receive a response from a server within a specific timeframe, leading to the connection being terminated
ADAPTER_TCP_CLIENT_KEEPALIVE_IDLE	INTEGER	-	120	O	The keepalive will be sent after the connection is idle for the configured seconds
ADAPTER_TCP_CLIENT_KEEPALIVE_INT	INTEGER	-	20	O	interval between two keepalive messages
ADAPTER_TCP_CLIENT_KEEPALIVE_COUNT	INTEGER	-	5	O	The maximum number of keepalive packets that will be sent before assuming the connection is dead
ADAPTER_TCP_CLIENT_SSL_HANDSHAKE	INTEGER	-	60	O	The timeout for the SSL handshake to complete for the TCP connection

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
TCP_CONN_INSPECTOR_FIXED_DELAY	STRING	-	30s	O	The delay interval after which the TCP connection watcher will keep on watching the connection
TCP_CONN_INSPECTOR_INI_DELAY	STRING	-	20s	O	The initial delay after which the TCP connection watcher will start watching the connection
ADAPTER_TCP_CONNECT_TIMEOUT	INTEGER	-	60	O	The TCP connection timeout after which there is no response received from the server
ADAPTER_TCP_CONNECT_RETRY_MS	INTEGER	-	1000	O	The TCP connection retry interval
ADAPTER_TCP_CONNECT_MAX_RETRY_DELAY_MS	STRING	-	120000	O	The maximum wait for the TCP connection retry
TCP_SEQ_ACK_CONNECTION_MGMT_TIMER_MILLI	STRING	-	7200000	O	The connection management timer for the sequence acknowledgement feature in consumer adapter. It is configured in milisec
SYNTHETIC_SEQ_ACK_CACHE_CLEAN_DELAY_SEC	INTEGER	-	5	O	The timeout after which the cache cleaning will happen for the sequence ack cache
SYNTHETIC_SEQ_ACK_CACHE_CLEAN_PERIOD_SEC	INTEGER	-	5	O	The time period an entry will remain in sequence ack cache

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
SYNTHETIC_STREAM_ID_TRANSACTION_MGMT_TIMER_MILLI	INTEGER	-	100000	O	The connection management timer for the stream-id feature in consumer adapter. It is configured in milisec
SYNTHETIC_STREAM_ID_CACHE_CLEAN_DELAY_SEC	INTEGER	-	2	O	The timeout after which the cache cleaning will happen for the stream-id cache
SYNTHETIC_STREAM_ID_CACHE_CLEAN_PERIOD_SEC	INTEGER	-	2	O	The time period an entry will remain in stream-id cache
ADAPTER_KAFKA_SOURCE_TOPIC	STRING	-	MAIN	M	The topic name that consumer adapter will start reading from
ADAPTER_KAFKA_TOPIC_CHECK_INITIAL_DELAY	STRING	-	120s	M	The interval after which the consumer adapter will check if the topic to consume has been created.
OCNADD_KAFKA_SSL_PROTOCOL	STRING	-	TLSv1.3	M	SSL protocol configured in consumer adapter with Kafka cluster
ADAPTER_KAFKA_MAX_METADATA_AGE	STRING	-	300000	O	The time after which the consumer adapter will refresh the metadata information from the Kafka cluster

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ENABLE_KAFKA_RECORD_TIMESTAMP_PROCESSING	BOOLEAN	true/false	true	O	The parameter to denote that Kafka record timestamp should be used in the latency calculation
ADAPTER_KAFKA_ENABLE_AUTO_COMMIT	BOOLEAN	true/false	false	O	The parameter to enable the autocommit of Kafka consumer offsets by the consumer adapter
ADAPTER_KAFKA_AUTOCOMMIT_INT_CONFIG	INTEGER	-	0	O	The value of autocommit interval if autocommit is enabled in the consumer adapter
STREAM_THREAD_DELAY_MS	INTEGER	-	10000	O	The initial delay in the stream processing after which the asynchronous retries will be tried.
ADAPTER_ASYNC_ENDPOINT_RETRY_SCH_FXD_DELAY	STRING	-	30s	O	The delay after which the retry is done in an asynchronous communication with the 3rd party application
MAX_CONSECUTIVE_ERRORS_ALLOWED	INTEGER	-	10	O	The number of consecutive failure after which the circuit is broken
MAX_CONSECUTIVE_SUCCESS_REQ_ALLOWED	STRING	-	270000	O	The consecutive successful messages count before the circuit is deemed to be closed again

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ENABLE_TCP_NODELAY	BOOLEAN	true/false	true	O	• **Usage:** It controls Netty socket option TCP_NODELAY for both egress **HTTP** and **TCP** clients. • **Role:** – true: disables Nagle's algorithm, so packets are sent immediately (lower latency, better for real-time/ transaction traffic). – false: enables packet coalescing (can reduce packet count, but may add small send delays). In short, it is a latency-vs-efficiency

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
					tuning knob for outbound OCNADD traffic.
NETTYCLIENT_EVENTLOOP_MULTIPLIER	INTEGER	-	12	M	In CustomNettyHttpClientFactory, OCNADD computes Netty worker threads as: workerCount = availableProcessors * NETTYCLIENT_EVENTLOOP_MULTIPLIER and uses that for the HTTP client event-loop (LoopResources.create(...)). **Effect:** Higher value = more event-loop threads (better concurrency/ throughput potential, higher CPU/context-switch overhead). Lower value = fewer threads (lower overhead, possible bottleneck under heavy load).

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
EGRESS_REQUEST_COMPLETE_TIMEOUT	INTEGER	-	30	M	It is the **overall completion timeout** (in seconds) for outbound egress HTTP requests from OCNADD Consumer Adapter. • If a request does not fully complete within this time, OCNADD raises a TimeoutException (mapped to NettyResponseTimeout), then treats it as a request failure path (metrics/ error handling, retry logic, endpoint health handling).

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
EGRESS_KEEPALIVE	BOOLEAN	true/false	true	O	EGRESS_KEEPALIVE controls TCP keepalive for OCNADD Consumer Adapter's **egress HTTP client**. • **Role:** It sets Netty socket option SO_KEEPALIVE for outbound HTTP connections. – true: keepalive probes are enabled to detect stale/dead peer connections and keep long-lived connections healthier. – false: no TCP keepalive probing. • It works with: – EGRESS_KEEPALIVE_IDLE

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
					– EGRESS_KEEPALIVE_INT – EGRESS_KEEPALIVE_COUNT which tune probe timing/ count (via epoll keepalive options).

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
EGRESS_HTTP_MAX_BLOCK_DURATION	String	-	60000	O	This is a **blocking wait limit (in milliseconds)** for OCNADD Consumer Adapter HTTP egress flows. • **Effect:** It caps how long the calling thread waits for the reactive HTTP pipeline to finish, preventing indefinite blocking. Higher value = waits longer before giving up; lower value = faster unblocking but more chance of premature timeout in slow conditions.

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
EGRESS_HTTP_SYNC_ASYNC_RETRY_MAX_ATTEMPT	INTEGER	-	5	O	This parameter sets the **max immediate retry attempts** for OCNADD Consumer Adapter egress HTTP send operations. Used in HttpService with Retry.fixedDelay(...) for: • – synchronous HTTP send flow – Netty-based send flow (when retry-on-error path is active) • **Effect:** After this retry count is exhausted, OCNADD treats send as failed (error handling/ metrics/ endpoint down handling kicks in).

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
EGRESS_HTTP_SYNC_RETRY_DELAY_MS	INTEGER	-	5000	O	This is the **wait time between immediate egress HTTP retries** in OCNADD Consumer Adapter. • **Effect:** – Higher value: slower retry cadence, less retry pressure. – Lower value: faster retries, quicker recovery but higher burst load. It is separate from scheduler retry delay (ADAPTER_ASYNC_ENDPOINT_RETRY_SCH_DELAY_MS), which is for background endpoint-recovery cycles.

Table 3-9 (Cont.) Consumer Aapter Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ADAPTER_METRICS_PER_CONN_ENABLED	BOOLEAN	-	false	0	ADAPTER_METRICS_PER_CONN_ENABLED controls whether OCNADD publishes **per-TCP-connection message metrics** in Consumer Adapter. In TcpService, when enabled, each successful connection selection pegs: adapterMetrics.pegTotalMessagesPerConnection(endpoint, connectionIndex, workerGroup). • **Metric produced:** ocnadd_synthetic_messages_per_connection (tagged by endpoint + connection index + worker group).

3.8.2 Correlation Service Parameters

All the Correlation parameters are available under `ocnaddadminsvc.correlation` section in ocnadd-custom-values-25.2.101.yaml file.

Table 3-10 Correlation Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
name	STRING	-	ocnaddcorrelation	M	name of the service
resources.limits.cpu	INTEGER	-	3	M	Number of maximum CPUs for each Correlation instance
resources.limits.memory	STRING	-	64Gi	M	Max Memory limit for each Correlation instance
resources.limits.ephemeralstorage	STRING	-	400Mi	M	Ephemeral Storage for each Correlation instance
resources.requests.cpu	INTEGER	-	3	M	Minimum number of CPUs required for each Correlation instance
resources.requests.memory	STRING	-	64Gi	M	minimum Memory required for each Correlation instance
resources.requests.ephemeralstorage	STRING	-	400Mi	M	minimum Ephemeral Storage required for each Correlation instance
Environmental variables for Correlation service are declared under "ocnaddadminsvc.correlation.env" section					
OCNADD_CORRELATION_SVC_LIVENESS_DELAY	INTEGER	-	60	O	This field tells the kubelet that it should wait for mentioned seconds before performing the first probe
OCNADD_CORRELATION_SVC_LIVENESS_PERIOD_SECONDS	INTEGER	-	15	O	This field specifies that the kubelet should perform a liveness probe every given number of seconds.
OCNADD_CORRELATION_SVC_LIVENESS_FAILURE	INTEGER	-	5	O	For the case of a liveness probe, triggers a restart for that specific container if the container failed to start for given number of failure retries.
OCNADD_CORRELATION_SVC_LIVENESS_TIMEOUT	INTEGER	-	20	O	Number of seconds after which the probe times out.

Table 3-10 (Cont.) Correlation Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_CORRELATION_NAME	STRING	-	ocnaddcorrelation	M	correlation service name which will be replaced by the admin service while creating correlation configuration with new name
ADMIN_CORRELATION_RESOURCE_FILE	STRING	-	/tmp/ocnadd/deploy/ocnaddcorrelationservice.yaml	M	Template file for deploying correlation service through admin service.
OCNADD_CORRELATION_ACTIVE_PROFILE	STRING	[prod,dev]	prod	M	profile with dev or production parameters
OCNADD_CORRELATION_SERVER_PORT	INTEGER	-	9664	M	port number for correlation service
OCNADD_CORRELATION_HTTP2_ENABLED	BOOLEAN	[true/false]	TRUE	M	enable or disable http2
OCNADD_TRUST_CLIENT_KEY_TYPE	STRING	-	PKCS12	M	trustore client key Type
KAFKA_STREAM_STATE	STRING	-	/tmp/ocnadd/kafka/state	O	temporary storage for kafka state store
KAFKA_REPLICATION_FACTOR	INTEGER	-	1	O	replication factor for state store
KAFKA_ENABLE_AUTO_COMMIT	BOOLEAN	[true/false]	FALSE	O	enable or disable kafka auto commit
KAFKA_SOCKET_BYTES_BUFFER	INTEGER	-	104857	O	Kafka Socket Buffer setting for consumer
KAFKA_SOCKET_BYTES_BUFFER_PORTION	INTEGER	-	100	O	This parameter is used to multiply with KAFKA_SOCKET_BYTES_BUFFER
KAFKA_FETCH_MIN_BYTES	INTEGER	-	1	O	The minimum amount of data per-partition the server will return
KAFKA_FETCH_MAX_BYTES	INTEGER	-	576720	O	The maximum amount of data per-partition the server will return
KAFKA_FETCH_MAX_BYTES_PORTION	INTEGER	-	100	O	This parameter is used to multiply with KAFKA_FETCH_MAX_BYTES
KAFKA_MAX_PARTITIONS_FETCH_BYTES	INTEGER	-	104858	O	The maximum amount of data per-partition the server will return.

Table 3-10 (Cont.) Correlation Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
KAFKA_MAX_PARTITIONS_FETCH_BYTES_PORTION	INTEGER	-	10	O	This parameter is used to multiply with KAFKA_MAX_PARTITIONS_FETCH_BYTES
FETCH_MAX_WAIT_MS	INTEGER	-	100	O	The maximum amount of time the server will block before answering the fetch request if there isn't sufficient data to immediately satisfy the requirement given by fetch.min.bytes
SESSION_TIME_OUT	INTEGER	-	15000	O	The timeout used to detect client failures when using Kafka's group management facility.
HEARTBEAT_INTERVAL_MS	INTEGER	-	5000	O	The expected time between heartbeats to the consumer coordinator when using Kafka's group management facilities
MAX_POLL_INTERVAL_MS	INTEGER	-	30000	O	The maximum delay between invocations of poll() when using consumer group management
MAX_POLL_RECORDS	INTEGER	-	500	O	The maximum number of records returned in a single call to poll()
KAFKA_OFFSET_CONFIG	STRING	-	latest	O	default kafka data stream offset config
KAFKA_AUTOCOMMIT_INT_CONFIG	INTEGER	-	50	O	It specifies how often the consumer commits its current position, which can be useful for ensuring message processing progress.
KAFKA_COMMIT_INT_CONFIG	INTEGER	-	50	O	this property will configure the interval at which Kafka consumer commits offsets.

Table 3-10 (Cont.) Correlation Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
KAFKA_NUMBER_THREADS_CONFIG	INTEGER	-	6	O	this property is used to configure the number of threads or consumers that Kafka Streams or Kafka consumers will use for processing messages
KAFKA_MAX_AGE_CONFIG	INTEGER	-	7500	O	This property will be used to set a maximum age for Kafka consumer records
KAFKA_CONSUMER_STRATEGY	STRING	-	org.apache.kafka.clients.consumer.RoundRobinAssignor	O	This property will be used to the strategy used for partition assignment when consuming messages from Kafka topics
PRODUCERS_ACKNOWLEDGMENTS	INTEGER	-	0	O	producer acknowledgments
CONSUMER_POLL_MS	INTEGER	-	50	O	Polling time in ms for consumer
BATCH_SIZE	INTEGER	-	65536	O	The maximum amount of data to be collected before sending the batch.
LINGER_MS	INTEGER	-	1	O	The time to wait before sending messages out to Kafka
REQUEST_TIMEOUT_MS	INTEGER	-	1000	O	The configuration controls the maximum amount of time the client will wait for the response of a request
INTERNAL_LEAVE_GROUP_ON_CLOSE	BOOLEAN	[true/false]	TRUE	O	this property controls whether the Kafka Streams application should actively leave the consumer group when it is closed or whether it should rely on the group coordinator to remove it when it becomes unresponsive
OCNADD_CORRELATION_HEALTH_RETRY_COUNT	INTEGER	-	3	O	No of times the correlation service retries for health registration in case of failure

Table 3-10 (Cont.) Correlation Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_CORRELATION_HEALTH_RETRY_DELAY	INTEGER	-	10	O	delay between the each retry for health registration
OCNADD_CORRELATION_HEALTH_HB_TIMER	INTEGER	-	120000	O	heart beat timer for health check
OCNADD_KAFKA_SSL_PROTOCOL	STRING	-	TLSv1.3	O	SSL Protocol version
OCNADD_KAFKA_SECURITY_PROTOCOL_SASL	STRING	-	SASL_SSL	O	describes SASL_SSL kafka security Protocol
OCNADD_KAFKA_SECURITY_PROTOCOL_SSL	STRING	-	SSL	O	describes SSL kafka security Protocol
OCNADD_KAFKA_SASL_MECHANISM	STRING	-	PLAIN	O	describes SASL SCRAM mechanism
OCNADD_KAFKA_SASL_JAAS_USERNAME	STRING	-	ocnadd	O	kafka default jaas username present
OCNADD_KAFKA_SASL_JAAS_MODULE	STRING	-	org.apache.kafka.common.security.plain.PlainLoginModule	O	kafka Login module
OCNADD_KAFKA_JAAS_SECRET_KEY	STRING	-	jaas_password	O	jaas password for kafka authentication taken from the jaas-secret with this key
OCNADD_KAFKA_JASS_SECRET_NAME	STRING	-	jaas-secret	O	jaas-secret name
CORRELATION_LOG_LEVEL_KAFKA	STRING	[ON,OFF]	OFF	O	Kafka Streams Log Level
OCNADD_CORRELATION_LOG_ROOT	STRING	-	WARN	O	root log level
OCNADD_CORRELATION_LOG_NETTY	STRING	-	INFO	O	netty log level
logging.level.com.oracle.cgbu.cne.ocnadd	STRING	-	INFO	O	ocnadd package log level
logging.level.com.oracle.cgbu.cne.ocdd	STRING	-	INFO	O	ocdd package log level
OCNADD_TRUST_KEYSTORE	BOOLEAN	-	TRUE	O	Enable to secure connection via OCWeb Client.
KAFKASTREAMS_PUNCTUATOR_TIMER	INTEGER	-	2000	O	Kafka Stream Scheduler Timer to loop through the Local cache for Timer Expiry Scenario

3.8.3 Storage Adapter Service Parameters

All the Storage Adapter Service parameters are available under `ocnaddadminsvc.storageadapter` section in `ocnadd-custom-values-25.2.101.yaml` file.

Table 3-11 Storage Adapter Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
name	STRING	-	ocnaddstorageadapter	M	Name of the service
resources.limits.cpu	INTEGER	-	3	M	Number of maximum CPUs for each storage adapter instance.
resources.limits.memory	STRING	-	64Gi	M	Maximum memory limit for each storage adapter instance.
resources.limits.ephemeralstorage	STRING	-	400Mi	M	Ephemeral storage for each storage adapter instance.
resources.requests.cpu	INTEGER	-	3	M	Minimum number of CPUs required for each storage adapter instance.
resources.requests.memory	STRING	-	64Gi	M	Minimum memory required for each storage adapter instance.
resources.requests.ephemeralstorage	STRING	-	400Mi	M	Minimum ephemeral storage required for each storage adapter instance.

Environmental variables for Storage Adapter service are declared under "ocnaddadminsvc.storageadapter.env" section

Table 3-12 Environmental variables for Storage Adapter service are declared under "ocnaddadminsvc.storageadapter.env" section

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_STORAGE_ADAPTER_HTTP2_ENABLED	BOOLEAN	[true,false]	true	M	The flag to indicate if HTTP2 should be used or not. Default is true
KAFKA_SOCKET_BYTES_BUFFER	INTEGER	-	104857	O	Kafka Socket Buffer setting for consumer
STORAGE_ADAPTER_KAFKA_FETCH_MIN_BYTES	INTEGER	-	1	O	The minimum amount of data per-partition the server will return

Table 3-12 (Cont.) Environmental variables for Storage Adapter service are declared under "ocnaddadminsvc.storageadapter.env" section

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
STORAGE_ADAPTER_KAFKA_MAX_PARTITION_FETCH_BYTES	INTEGER	-	104858	O	The maximum amount of data per-partition the server will return.
STORAGE_ADAPTER_KAFKA_FETCH_MAX_WAIT_MS	INTEGER	-	100	O	The maximum amount of time the server will block before answering the fetch request if there isn't sufficient data to immediately satisfy the requirement given by fetch.min.bytes
STORAGE_ADAPTER_KAFKA_SESSION_TIME_OUT	INTEGER	-	90000	O	The timeout used to detect client failures when using Kafka's group management facility.
STORAGE_ADAPTER_KAFKA_HEARTBEAT_INTERVAL_MS	INTEGER	-	30000	O	The expected time between heartbeats to the consumer coordinator when using Kafka's group management facilities
STORAGE_ADAPTER_KAFKA_MAX_POLL_INTERVAL_MS	INTEGER	-	240000	O	The maximum delay between invocations of poll() when using consumer group management
STORAGE_ADAPTER_KAFKA_MAX_POLL_RECORDS	INTEGER	-	900	O	The maximum number of records returned in a single call to poll()
STORAGE_ADAPTER_KAFKA_OFFSET_CONFIG	STRING	-	latest	O	Default Kafka data stream offset config

Table 3-12 (Cont.) Environmental variables for Storage Adapter service are declared under "ocnaddadminsvc.storageadapter.env" section

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
STORAGE_ADAPTER_KAFKA_NUM_THREADS_CONFIG	INTEGER	-	6	O	This property is used to configure the number of threads or consumers that Kafka Streams or Kafka consumers will use for processing messages
PRODUCERS_ACKNOWLEDGMENTS	INTEGER	-	0	O	Producer acknowledgments
STORAGE_ADAPTER_KAFKA_CONSUMER_POLL_MS	INTEGER	-	50	O	Polling time in ms for consumer
KAFKA_BATCH_SIZE	INTEGER	-	75000	O	The maximum amount of data to be collected before sending the batch.
STORAGE_ADAPTER_LOG_LEVEL_KAFKA	STRING	-	OFF	O	Kafka log level
OCNADD_STORAGE_ADAPTER_LOG_ROOT	STRING	-	WARN	O	root log level
OCNADD_STORAGE_ADAPTER_LOG_NETTY	STRING	-	INFO	O	Netty log level
logging.level.com.oracle.cgbu.cne.ocnadd	STRING	-	INFO	O	OCNADD package log level
logging.level.com.oracle.cgbu.cne.ocdd	STRING	-	INFO	O	OCNADD package log level
OCNADD_INTERNAL_CLIENT_SSL_PROTOCOL	STRING	-	TLS	O	The secure protocol used between Storage adapter and internal DD services for the HTTP communication

Table 3-12 (Cont.) Environmental variables for Storage Adapter service are declared under "ocnaddadminsvc.storageadapter.env" section

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_INTERNAL_CLIENT_SSL_PROTOCOLS	STRING	-	TLSv1.2,TLSv1.3	O	The TLS version supported by HTTP client in the storage adapter service
OCNADD_INTERNAL_CLIENT_SSL_HANDSHAKE_TIMEOUT	STRING	-	30s	O	The SSL handshake timeout value in the HTTP client used in the storage adapter service
STORAGE_ADAPTER_METRICS_ENABLED	BOOLEAN	true/false	false	O	Parameter to enable the metrics pegging for the storage adapter, defaule is false
STORAGE_ADAPTER_EVENT_ENABLED	BOOLEAN	true/false	true	O	Parameter to enable the events on the storage adapter, default is true
EVENT_DELETE_BATCH_SIZE	INTEGER	-	5000	O	The size of the event list that can be deleted by storage adapter, default is 5000

3.8.4 Ingress Adapter Service Parameters

All the Ingress Adapter Service parameters are available under `ocnaddadminsvc.ingressadapter` section in `ocnadd-custom-values-25.2.101.yaml` file.

Table 3-13 Ingress Adapter Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
name	STRING	-	ocnaddingressadapter	M	Name of the service
resources.limits.cpu	INTEGER	-	3	M	Number of maximum CPUs for each ingress adapter instance.

Table 3-13 (Cont.) Ingress Adapter Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
resources.limits.memory	STRING	-	64Gi	M	Max Memory limit for each ingress adapter instance.
resources.limits.ephemeralstorage	STRING	-	400Mi	M	Ephemeral Storage for each ingress adapter instance.
resources.requests.cpu	INTEGER	-	3	M	Minimum number of CPUs required for each ingress adapter instance.
resources.requests.memory	STRING	-	64Gi	M	Minimum Memory required for each ingress adapter instance.
resources.requests.ephemeralstorage	STRING	-	400Mi	M	Minimum Ephemeral Storage required for each ingress adapter instance.
Environmental variables for Ingress Adapter service are declared under "ocnaddadminsvc.ingressadapter.env" section					
INGRESS_ADAPTER_PORT	INTEGER	-	9188	M	The port of the ingress adapter service.
INGRESS_ADAPTER_SERVICE_NAME	STRING	-	ocnaddingressadapter	M	Name of the service
OCNADD_TRUST_KEYSTORE	BOOLEAN	-	true	O	Enable to secure connection using OCWeb Client.
OCNADD_TRUST_CLIENT_KEY_TYPE	STRING	-	PKCS12	M	Trust store client key Type
OCWEBCLIENT_TIMEOUT	INTEGER	-	60		Webclient timeout
OCWBCLIENT_CHANNEL_TIMEOUT	INTEGER	-	60	O	Webclient channel timeout
OCWEBCLIENT_SSL_HANDSHAKE_TIMEOUT	INTEGER	-	30	O	SSL handshake timeout
OCWEBCLIENT_SSL_FLUSH_TIMEOUT	INTEGER	-	10	O	SSL flush timeout
OCWEBCLIENT_SSL_READ_TIMEOUT	INTEGER	-	10	O	SSL read timeout
logging.level.com.oracle.cgbu.cne.ocnadd	STRING	-	INFO	O	ocnadd package log level
logging.level.com.oracle.cgbu.cne.ocdd	STRING	-	INFO	O	ocdd package log level
INGRESS_ADAPTER_LOG_LEVEL_ROOT	STRING	-	OFF	O	Kafka log level

Table 3-13 (Cont.) Ingress Adapter Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
INGRESS_ADAPTER_NETTY_LOG_LEVEL	STRING	-	WARN	O	Root log level
INGRESS_ADAPTER_LOG_LEVEL_KAFKA	STRING	-	INFO	O	netty log level
INGRESS_ADAPTER_LOG_FILENAME	STRING	-	ingress-adapter.log	O	The log file name
OCNADD_CONSUMER_ADAPTER_HEALTH_RETRY_COUNT	INTEGER	-	3	O	The number of retries with the health service.
OCNADD_CONSUMER_ADAPTER_HEALTH_RETRY_DELAY	INTEGER	-	10	O	The time interval in sec between successive retries with the health service.
INGRESS_ADAPTER_HEALTH_HB_TIMER	INTEGER	-	120000	O	The heartbeat timer with the health service
INGRESS_ADAPTER_HTTP2_ENABLED	BOOLEAN	[true,false]	true	O	The flag to indicate if http2 should be used or not
INGRESS_ADAPTER_SSL_ENABLED	BOOLEAN	[true,false]	true	O	The flag to indicate if TLS should be used or not
INGRESS_ADAPTER_CLIENT_AUTH_CONFIG	STRING	-	want	O	Parameter to check client authentication
INGRESS_ADAPTER_KEYSTORE_TYPE	STRING	-	PKCS12	M	Trust store client key Type
INGRESS_ADAPTER_TRUSTSTORE_TYPE	STRING	-	PKCS12	M	Trust store key Type
INGRESS_HTTPSERVER_ROUTE_PATH	STRING	-	/ocnadd-nonoraclenf/v1/streaming	M	The URL at which the client should stream the data towards ingress adapter.
INGRESS_HTTPSERVER_READ_TIMEOUT_MS	INTEGER	-	30000	O	The ingress adapter read timeout in milliseconds
INGRESS_HTTPSERVER_REQUEST_TIMEOUT_MS	INTEGER	-	30000	O	The ingress adapter request timeout in milliseconds

Table 3-13 (Cont.) Ingress Adapter Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
INGRESS_HTTPSERVER_CONNECT_TIMEOUT_MS	INTEGER	-	60000	O	The ingress adapter connect timeout in milliseconds
INGRESS_HTTPSERVER_IDLE_TIMEOUT_MS	INTEGER	-	120000	O	The ingress adapter idle timeout in milliseconds
INGRESS_HTTPSERVER_SOCKET_RECEIVE_BUF	INTEGER	-	10485	O	The socket receive buffer size
INGRESS_HTTPSERVER_SOCKET_RECEIVE_BUF_PORTION	INTEGER	-	100	O	The socket receive buffer size multiple factor. The actual read buffer bytes will be (INGRESS_HTTPSERVER_SOCKET_RECEIVE_BUF * INGRESS_HTTPSERVER_SOCKET_RECEIVE_BUF_PORTION)
INGRESS_HTTPSERVER_SOCKET_TIMEOUT_MS	INTEGER	-	60000	O	The ingress adapter socket timeout in milliseconds
INGRESS_HTTPSERVER_SOCKET_KEEPALIVE	BOOLEAN	[true,false]	true	O	The flag to indicate if keepalive should be used in the connection
INGRESS_HTTPSERVER_CLOSE_NOTIFY_FLUSH_TIIMEOUT_MS	INTEGER	-	30000	O	Notification flush timeout in milliseconds
INGRESS_HTTPSERVER_CLOSE_NOTIFY_READ_TIIMEOUT_MS	INTEGER	-	30000	O	Notification read timeout in milliseconds
INGRESS_HTTPSERVER_SSL_HANDSHAKE_TIMEOUT_MS	INTEGER	-	30000	O	SSL handshake timeout in milliseconds
KAFKA_SECURITY_PROTOCOL	STRING	-	PLAINTEXT	O	Describes kafka security Protocol
INGRESS_ADAPTER_SECURITY_PROTOCOL	STRING	-	SSL	O	Describes ingress adapter security Protocol
KAFKA_SSL_PROTOCOL	STRING	-	TLSv1.2	O	SSL Protocol version
KAFKA_SASL_ENABLED	BOOLEAN	[true,false]	false	O	The flag to indicate if SASL is used for the authentication

Table 3-13 (Cont.) Ingress Adapter Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
KAFKA_JAAS_CONFIG_MODULE	STRING	-	org.apache.kafka.common.security.plain.PlainLoginModule	O	Kafka Login module
KAFKA_JAAS_CONFIG_USER	STRING	-	username	O	Kafka default jaas username
KAFKA_JAAS_CONFIG_PASS	STRING	-	secret	O	Kafka default jaas password
KAFKA_SASL_MECHANISM	STRING	-	PLAIN	O	Describes SASL SCRAM mechanism
externalAccess.enabled	BOOLEAN	[true,false]	false	O	The flag to indicate if external access is enabled for the ingress adapter
externalAccess.staticLoadBalancerIp	STRING	-	10.10.10.1	O	Default static loadbalancer IP address

3.9 Kafka Configuration Parameters

Table 3-14 Kafka Configuration Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
kafkaBroker.kafkaProperties.logdirs	String	-	/kafka/logdir/kafka-logs	M	The path to store the Kafka logs
kafkaBroker.name	String	-	kafka-broker	-	Name of the Kafka broker
kafkaBroker.replicas	Int	-	4	-	The number of replicas that should be available for the pod.
kafkaBroker.pvcClaimSize	String	-	10Gi	M	Size of Block Volume to attach to Kafka.
target.averageCpuUtilPercentage	Int	-	50	-	The target average CPU utilization percentage.

Table 3-14 (Cont.) Kafka Configuration Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
target.memory UtilPercentage	Int	-	80	-	The target average memory utilization percentage.
kafkaBroker.resource.limits.cpu	Int	-	5	-	The maximum limit for the number of CPUs used for the container.
kafkaBroker.resource.limits.memory	String	-	24Gi	-	The maximum limit for the size of the memory used for the container.
kafkaBroker.kafkaProperties.logRetentionMinutes	Int	-	5	M	Log Retention Time of Topic Data in Minutes.
kafkaBroker.kafkaProperties.kafkaSslProtocol	String	-	TLSv1.2,TLSv1.3	M	TLS version supported.
kafkaBroker.kafkaProperties.socketSendBufferBytes	Int	-	10485760	M	TCP socket buffer sizes for the producer.
kafkaBroker.kafkaProperties.socketReceiveBufferBytes	Int	-	10485760	M	TCP socket buffer sizes for the consumer.
kafkaBroker.kafkaProperties.socketRequestMaxBytes	Int	-	104857600	M	The maximum number of bytes in a socket request.
kafkaBroker.kafkaProperties.queuedMaxRequests	Int	-	4096	M	Number of concurrent connections.
kafkaBroker.kafkaProperties.numIoThreads	Int	-	820	M	Number of threads that pick up requests from the request queue to process them.

Table 3-14 (Cont.) Kafka Configuration Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
kafkaBroker.kafkaProperties.numNetworkThreads	Int	-	820	M	Network threads handle requests to the Kafka cluster, such as produce and fetch requests from client applications.
kafkaBroker.kafkaProperties.numReplicaFetchers	Int	-	640	M	Number of fetcher threads used to replicate records from each source broker.
kafkaBroker.kafkaProperties.backgroundThreads	Int	-	256	M	The number of threads to use for various background processing tasks.
kafkaBroker.kafkaProperties.replicaFetchMinBytes	Int	-	619200	M	Minimum bytes expected for each fetch response.
kafkaBroker.kafkaProperties.replicaFetchMaxBytes	Int	-	37152000	M	The maximum number of bytes we will return for a fetch request.
kafkaBroker.kafkaProperties.replicaFetchWaitMaxMs	Int	-	500	M	The maximum wait time for each fetcher request issued by follower replicas.
kafkaBroker.kafkaProperties.replicaSocketReceiveBufferBytes	Int	-	10485760	M	The socket receive buffer for network requests.

Table 3-14 (Cont.) Kafka Configuration Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
kafkaBroker.kafkaProperties.offsetsTopicReplicationFactor	Int	-	3	M	The replication factor for the offsets' topic (set higher to ensure availability). Internal topic creation will fail until the cluster size meets this replication factor requirement.
kafkaBroker.kafkaProperties.transactionStateLogReplicationFactor	Int	-	3	M	The replication factor for the transaction topic (set higher to ensure availability). Internal topic creation will fail until the cluster size meets this replication factor requirement.
kafkaBroker.externalAccess.enabled	Boolean	-	false	M	Flag to enable External access for Kafka.
kafkaBroker.externalAccess.autoDiscovery	Boolean	-	false	M	Flag to enable auto-discovery of LoadBalancer IPs.
kafkaBroker.externalAccess.type	String	-	LoadBalancer	M	Service Type of Kafka Broker.
kafkaBroker.externalAccess.setstaticLoadBalancerIps	Boolean	-	false	M	Setting Static LoadBalancer IPs.
kafkaBroker.externalAccess.LoadBalancerIPList	List	-	[ ]	C	List of LoadBalancer Static IP available for use.

Table 3-14 (Cont.) Kafka Configuration Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
kafkaBroker.kafkaProperties.ramDriveStorage	Boolean	true/false	false	C	The property is used to enable RAM-based storage for the Kafka cluster in the worker group. When enabled, messages in the Kafka topic will be stored in RAM with very minimal retention. The default value is `false`. By default, the Kafka cluster storage will use CEPH-based persistent storage.

3.10 UI Router Parameters

Listed below are the UI Router Parameters:

Table 3-15 UI Router Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
ocnadduirouter.name	String		ocnadduirouter	M	The express application name for ocnadduirouter.
Router_host	String		http://localhost	M	The Router host contains uirouter service endpoint and value of this variable can be service name, node IP, and Load Balancer IP.
Router_port	String		8080	M	The ocnadduirouter service has exposed this specified port and this port can be used to access specific router service.
DD_UIAPI	String		http://ocnaddgui:80	M	The target endpoint of ocnaddgui service and use to configure the GUI.

Table 3-15 (Cont.) UI Router Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
DD_CONFIG_API	String		http://ocnaddconfiguration:12590	M	Target endpoint of the configuration service used to route the GUI request and forward the response.
DD_ALARM_API	String		http://ocnaddalarm:9099	M	Target endpoint of the Alarm service used to route the GUI request and forward the response.
DD_HEALTH_API	String		http://ocnaddhealthmonitoring:12591	M	Target endpoint of the Health Monitoring service used to route the GUI request and forward the response.
PROMETHEUS_API	String		http://occne-kube-prom-stack-kube-prometheus.occne-infra.svc.ocnadd:80	M	Target endpoint of the Prometheus service used to route the GUI request and forward the response. **Note:** Update this parameter based on the setup.
DD_PROMETHEUS_PATH	String		/blurr8/prometheus/api/v1/query_range	M	The Prometheus endpoint API URL path. **Note:** Update this parameter based on the setup.

3.11 Filter Service Parameters

Filter Service Parameters are present under `ocnaddfilter` section in ocnadd-custom-values-25.2.101.yaml file.

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
name	STRING	-	ocnaddfilter	M	Name of the Service.
minReplicas	INTEGER	-	1	M	Minimum number of Replicas of Filter Service
maxReplicas	INTEGER	-	3	M	Maximum number of Replicas of Filter Service
resources.limits.cpu	INTEGER	-	2	M	Number of maximum CPUs for each Filter service instance

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
resources.limits.memory	STRING	-	3Gi	M	Max Memory limit for each Filter service instance
resources.limits.ephemeralstorage	STRING	-	800Mi	M	Ephemeral Storage for each Filter service instance
resources.requests.cpu	INTEGER	-	2	M	Minimum number of CPUs required for each Filter service instance
resources.requests.memory	STRING	-	3Gi	M	Minimum Memory required for each Filter service instance
resources.requests.ephemeralstorage	STRING	-	500Mi	M	Minimum Ephemeral Storage required for each Filter service instance
Environmental variables are present under section "ocnaddfilter.env"					
OCNADD_TRUST_KEYSTORE	BOOLEAN	[true/false]	true	O	Enable to secure connection via OCWeb Client.
ENABLE_FILTER_METRICS	BOOLEAN	[true/false]	true	O	To enable/ disable filter metrics, default is true
KAFKA_PRODUCER_SSL_PROTOCOL	STRING	-	TLSv1.3	O	Kafka SSL protocol version
KAFKA_PRODUCER_SSL_CLIENT_AUTH	BOOLEAN	[true/false]	false	O	Whether Kafka producer client auth is required or not
KAFKA_MAX_AGE_CONFIG	INTEGER	-	7500	O	The period of time in milliseconds after which we force a refresh of metadata.

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
KAFKA_FETCH_MIN_BYTES	INTEGER	-	1	O	The minimum amount of data per-partition the server will return
KAFKA_FETCH_MAX_BYTES	STRING	-	57672000	O	The maximum amount of data per-partition the server will return
KAFKA_MAX_PARTITIONS_FETCH_BYTES	STRING	-	1048580	O	The maximum amount of data per-partition the server will return.
FETCH_MAX_WAIT_MS	INTEGER	-	100	O	The maximum amount of time the server will block before answering the fetch request if there isn't sufficient data to immediately satisfy the requirement given by fetch.min.bytes
SESSION_TIME_OUT	INTEGER	-	15000	O	The timeout used to detect client failures when using Kafka's group management facility.
HEARTBEAT_INTERVAL_MS	INTEGER	-	5000	O	The expected time between heartbeats to the consumer coordinator when using Kafka's group management facilities
MAX_POLL_INTERVAL_MS	INTEGER	-	240000	O	The maximum delay between invocations of poll() when using consumer group management

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
MAX_POLL_RECORDS	INTEGER	-	1500	O	The maximum number of records returned in a single call to poll()
CONSUMER_POLL_MS	INTEGER	-	50	O	Polling time in ms for consumer
BATCH_SIZE	INTEGER	-	130000	O	The maximum amount of data to be collected before sending the batch.
LINGER_MS	INTEGER	-	2	O	The time to wait before sending messages out to Kafka
REQUEST_TIMEOUT_MS	INTEGER	-	1000	O	The configuration controls the maximum amount of time the client will wait for the response of a request
TRANSACTION_FILTER	BOOLEAN	[true/false]	true	O	To enable or disable transaction filtering
KAFKA_SOCKET_BYTES_BUFFER	STRING	-	1048570	O	Kafka Socket Buffer setting for consumer
FILTER_KAFKA_PARTITIONER_STRATEGY	STRING	[key/custom/ roundrobin]	key	O	Kafka record partitioned strategy
OCNADD_FILTER_KAFKA_TOPIC_INITIAL_DELAY	STRING	-	10s	O	The parameter denotes the initial delay in checking for the Kafka topic existence, default is 10sec
OCNADD_FILTER_KAFKA_TOPIC_RETRY_THRESHOLD	STRING	-	20s	O	The parameter denotes the retry interval for checking the Kafka topic existence, default is 20sec

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_CONFIG_RETRY_COUNT	INTEGER	-	3	O	The number of retries for the communication towards the configuration service in case of failure, default is 3
OCNADD_CONFIG_RETRY_DELAY_MS	INTEGER	-	5000	O	The retry interval for the communication towards the configuration service in case of failure, default is 5ms
KAFKA_DESCRIBE_TOPIC_TIMEOUT_MS	INTEGER	-	10000	O	The timeout for the request to get the Kafka describe topic output from Kafka cluster, default is 10ms
OCNADD_FILTER_HEALTH_SVC_TYPE	STRING	-	FILTER_SERVICE	M	The type with which filter service registers with the health monitoring service
OCNADD_FILTER_HEALTH_HB_TIMER	INTEGER	-	10000	M	The heartbeat timer on the filter service to exchange the heartbeat with the health monitoring service, default is 10sec
OCNADD_FILTER_HEALTH_RETRY_COUNT	INTEGER	-	1	M	The number of retries with the health monitoring service for the registration of filter service health profile, default is 1

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_FILTER_HEALTH_RETRY_DELAY	INTEGER	-	2	M	The retry delay between two consecutive retries for the health profile registration of filter service, default is 2sec
OCNADD_FILTER_MAX_REPLICA	INTEGER	-	1	M	The number of maximum replicas, the health service instance reports to the health service during health profile registration.
OCNADD_FILTER_LIVENESS_FAILURE	INTEGER		60	M	Filter service Liveness Parameter : this field tells the kubelet that it should wait for mentioned seconds before performing the first probe.
OCNADD_FILTER_LIVENESS_DELAY	INTEGER		15	M	Filter service Liveness Parameter : this field specifies that the kubelet should perform a liveness probe every given no of seconds.
OCNADD_FILTER_LIVENESS_PERIOD_SECONDS	INTEGER		5	M	Filter service Liveness Parameter : For the case of a liveness probe, triggers a restart for that specific container if the container failed to start for given no of failure retries.

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_FILTER_LIVENESS_TIMEOUT	INTEGER		20	M	Filter service Liveness Parameter : Number of seconds after which the probe times out.
OCNADD_KAFKA_SSL_PROTOCOL	STRING	-	TLSv1.3	O	SSL Protocol version
OCNADD_KAFKA_SECURITY_PROTOCOL_SASL	STRING	-	SASL_SSL	O	The Kafka security protocol for the filter service for SASL connection with Kafka
OCNADD_KAFKA_SECURITY_PROTOCOL_SSL	STRING	-	SSL	O	The Kafka security protocol for the filter service for SSL connection with Kafka
OCNADD_KAFKA_SASL_JAAS_MODULE	STRING	-	org.apache.kafka.common.security.plain.PlainLoginModule	O	Kafka Login module
OCNADD_KAFKA_JASS_SECRET_NAME	STRING	-	username	O	Kafka default JAAS username present
OCNADD_KAFKA_JAAS_SECRET_KEY	STRING	-	secret	O	Kafka default JASS password present
OCNADD_KAFKA_SASL_MECHANISM	STRING	-	PLAIN	O	Describes SASL SCRAM mechanism

3.12 Redundancy Agent Service Parameters

Table 3-16 Redundancy Agent Service Parameter

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
name	STRING	-	ocnaddredundancyagent	M	Name of service.

Table 3-16 (Cont.) Redundancy Agent Service Parameter

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
autoScaling.enabled	BOOLEAN	true/ false	false	M	Allow `HorizontalAutoScalar` of `ocnaddredundancy` pods.
minReplicas	INTEGER	-	1	C	Number of minimum replicas for HPA.
maxReplicas	INTEGER	-	1	C	Number of maximum replicas for HPA.
resources.limit.cpu	INTEGER	-	2	M	Maximum number of CPU for each pod.
resources.limits.memory	STRING		1Gi	M	Maximum memory limit for each service instance.
resources.limits.ephemeralstorage	STRING		500Mi	M	Ephemeral storage for each service instance.
resources.requests.cpu	INTEGER		2	M	Minimum number of CPUs required for each service instance.
resources.requests.memory	STRING		1Gi	M	Minimum memory required for each service instance.
resources.requests.ephemeralstorage	STRING		500Mi	M	Minimum ephemeral storage required for each service instance.
resources.target.averageCpuUtilsPercentage	INTEGER	-	85	C	Threshold set for Pod AutoScalar.
Environmental variables are present under section **ocnaddredundancyagent.env**					
OCNADD_REDUNDANCY_HB_INTERVAL	INTEGER	-	10	O	Interval of heartbeat requests sent to primary agent by secondary agent.
OCNADD_REDUNDANCY_HB_MISSING	INTEGER		3	O	Max unsuccessful heartbeat in case of secondary agent or max missing heartbeat in case of primary agent.
OCNADD_REDUNDANCY_KAFKA_DELAY_MS	INTEGER		2000	O	Delay before starting periodic Kafka ingress traffic rate.
OCNADD_REDUNDANCY_KAFKA_INTERVAL_MS	INTEGER		500	O	Interval of periodic Kafka ingress traffic check, will switch mode of secondary agent if change is required during the check.
OCNADD_REDUNDANCY_HEALTH_RETRY_COUNT	INTEGER		10	O	Number of retries for Health registration.
OCNADD_REDUNDANCY_HEALTH_RETRY_DELAY	INTEGER		15	O	Delay between each retries for Health Registration.
OCNADD_REDUNDANCY_HEALTH_HB_TIMER	INTEGER		120000	O	Heart Beat Timer interval to health monitoring service.

Table 3-16 (Cont.) Redundancy Agent Service Parameter

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
OCNADD_REDUNDANCY_HEALTH_SVC_TYPE	STRING		REDUNDANCY	O	Health Registration name for REDUNDANCY agent.

3.13 Export Service Parameters

Table 3-17 Export Service Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
name	STRING	-	ocnaddexport	M	Name of service
autoScaling.enabled	BOOLEAN	true/ false	true	M	Allow HorizontalAutoScalar of ocnaddexport service pods.
minReplicas	INTEGER	-	1	C	Number of minimum replicas for HPA.
maxReplicas	INTEGER	-	2	C	Number of maximum replicas for HPA.
resources.limit.cpu	INTEGER	-	6	M	Max number of cpu for each pod.
resources.limits.memory	STRING	-	24Gi	M	Max memory limit for each service instance.
resources.limits.ephemeralstorage	STRING	-	2Gi	M	Ephemeral storage for each service instance.
resources.requests.cpu	INTEGER	-	4	M	Minimum number of CPUs required for each service instance.
resources.requests.memory	STRING	-	4Gi	M	Minimum memory required for each service instance.
resources.requests.ephemeralstorage	STRING	-	100Mi	M	Minimum ephemeral storage required for each service instance.
Environmental variables are present under section **ocnaddexport.env**					
logging.level.com.oracle.cgbu.cne.ocnadd	STRING	-	INFO	O	ocnadd package log level
logging.level.com.oracle.cgbu.cne.ocdd	STRING	-	INFO	O	ocdd package log level
EXPORT_WEB_LOG_LEVEL	STRING	-	INFO	O	Export service application log level
EXPORT_BLOCKINGQUEUE_SIZE	INTEGER	-	10	O	The queue size to store the result set from the database for the export.

Table 3-17 (Cont.) Export Service Parameters

EXPORT_SEQUENCING	BOOLEAN	true,false	true	O	The parameter to decide if the result set needs to be in sequence based on the record timestamp or not.

3.14 Helm Parameter Configuration for OCCM

Table 3-18 Helm Parameter Configuration for OCCM

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
global.certificates.occm.enabled	BOOLEAN	true/ false	false	M	Whether to use OCCM for creating services.
global.certificates.occm.issuer	STRING	-	CA1	M	Name of the Issuer configured in OCCM to use to create certificate
global.certificates.occm.renewBefore	INTEGER	-	14	M	Number of days before expiry, before which OCCM will automatically update the certificates
global.certificates.occm.days	INTEGER	-	90	M	Number of days for which certificates will be valid
global.certificates.occm.cncc.cncc_iam_ingress_gateway.external_ip	STRING	-	-	M	Load balancer IP address of CNCC IAM Ingress Gateway Service
global.certificates.occm.cncc.cncc_iam_ingress_gateway.port	INTEGER	-	80	M	Port of CNCC IAM Ingress Gateway Service
global.certificates.occm.cncc.cncc_mcore_ingress_gateway.external_ip	STRING	-	-	M	Load balancer IP address of CNCC MCORE Ingress Gateway Service
global.certificates.occm.cncc.cncc_mcore_ingress_gateway.port	INTEGER	-	80	M	Port of CNCC MCORE Ingress Gateway Service
global.certificates.occm.cncc.cnccId	STRING	-	Cluster1	M	ID of CNCC owner of OCCM instance
global.certificates.occm.cncc.occm_cncc_instance_id	STRING	-	Cluster1-occm-instance1	M	OCCM instance ID
global.certificates.occm.subject.country	STRING	-	-	M	Specify the country field (C) in DN for each certificate
global.certificates.occm.subject.state	STRING	-	-	M	Specify the state field (S) in DN for each certificate

Table 3-18 (Cont.) Helm Parameter Configuration for OCCM

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
global.certificates.occm.subject.location	STRING	-	-	M	Specify the location field (L) in DN for each certificate
global.certificates.occm.subject.organization	STRING	-	-	M	Specify the organization field (O) in DN for each certificate
global.certificates.occm.subject.country.organizationUnit	STRING	-	-	M	Specify the organization unit field (OU) in DN for each certificate
global.certificates.occm.occm_cacert	STRING	-	occm-ca-secret	O	Name of the Secret storing CA certificate/certificate chain.
global.certificates.occm.truststore_keystore_secret	STRING	-	occm-truststore-keystore-secret	O	Name of the Secret storing truststore and keystore key
global.certificates.occm.occm_secret	STRING	-	occm-secret	O	Name of the Secret storing CNCC user credentials
global.certificates.occm.occm_namespace	STRING	-	-	O	OCCM Namespace
global.certificates.occm.occm_service_name	STRING	-	occm	O	OCCM Kubernetes Service Name
global.certificates.occm.occm_port	INTEGER	-	8989	O	OCCM Service Port
global.certificates.occm.volumes.json	STRING	-	/occm-request	O	Mount path of the JSONs used when sending request to OCCM
global.certificates.occm.volumes.script	STRING	-	/occm-script	O	Mount path of the script used to send request to OCCM
global.certificates.occm.san.kafka.update_required	BOOLEAN	true/ false	false	C	If update of SAN field for Kafka certificates is required. Should be enabled post-installation when external access of Kafka is required.
global.certificates.occm.san.kafka.ips	LIST[STRING]	-	["10.10.10.10", "10.10.10.11", "10.10.10.12"]	C	IPs to add in SAN for Kafka certificate. Provide the Load balancer IPs during installation if static IPs for load balancer are chosen
global.certificates.occm.san.kafka.uuid.client	STRING	-	-	C	UUID of existing Kafka broker certificate with names prefixed by KAFKABROKER-SECRET-CLIENT
global.certificates.occm.san.kafka.uuid.server	STRING	-	-	C	UUID of existing Kafka broker certificate with names prefixed by KAFKABROKER-SECRET-SERVER

Table 3-18 (Cont.) Helm Parameter Configuration for OCCM

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
global.certificates.occm.san.redundancy_agent.update_required	BOOLEAN	true/ false	false	C	If update of SAN field for Redundancy Agent certificates is required. Should be enabled post-installation when two site redundancy is enabled.
global.certificates.occm.san.redundancy_agent.ips	LIST[STRING]	-	["10.10.10.10"]	C	IPs to add in SAN for Redundancy Agent certificate. Provide the Load balancer IP during installation if static IP for load balancer is chosen
global.certificates.occm.san.redundancy_agent.uuid.client	STRING	-	-	C	UUID of existing Redundancy Agent certificate with names prefixed by KAFKABROKER-SECRET-CLIENT
global.certificates.occm.san.redundancy_agent.uuid.server	STRING	-	-	C	UUID of existing Redundancy Agent certificate with names prefixed by KAFKABROKER-SECRET-SERVER
global.certificates.occm.san.ingress_adapter.update_required	BOOLEAN	true/ false	false	C	If update of SAN field for Ingress Adapter certificates is required. Should be enabled post-installation when external access to Ingress Adapter/s is needed.
global.certificates.occm.san.ingress_adapter.ips	LIST[STRING]	-	["10.10.10.10"]	C	IPs to add in SAN for Ingress Adapter certificate
global.certificates.occm.san.ingress_adapter.uuid.client	STRING	-	-	C	UUID of existing Ingress Adapter certificate with names prefixed by KAFKABROKER-SECRET-CLIENT
global.certificates.occm.san.ingress_adapter.uuid.server	STRING	-	-	C	UUID of existing Ingress Adapter certificate with names prefixed by KAFKABROKER-SECRET-SERVER
global.certificates.occm.keyAlgorithm	STRING	RSA/EC	RSA	C	Select OCCM key algorithm, RSA for RSA based key generation and EC for ECDSA based key generation
global.certificates.occm.keySize	STRING	KEYSIZE_2048/ KEYSIZE_4096	KEYSIZE_2048	C	Defines the keySize of RSA based key generation.
global.certificates.occm.ecCurve	STRING	SECP384r1/ SECP256r1	SECP384r1	C	Define the curve parameter when keyAlgorithm select is EC.

3.15 cnDBTier Customization Parameters

The Data Director uses cnDBTier as an independent database for georedundant sites. By default, the `ocnadd_dbtier_custom_values.yaml` file provided with the OCNADD installation is for a single-site deployment of cnDBTier.

Single-site cnDBTier deployment mode: Georeplication is unavailable. Users must continue taking databse backups periodically, preferably on a daily basis, so that the same can be used when fault recovery scenarios arise. See the section "Fault Recovery" for backup options in the OCNADD.

> **Note**
>
> For information about the values of the following parameters, see the `ocnadd_dbtier_custom_values.yaml` file:
>
> - Any change in the cnDBTier `custom_values` file introduced by the cnDBTier patch must be updated in the `custom_values` file provided by OCNADD before deployment.
> - For detailed information on the cnDBTier resources, see the section "DB Profile" in the Oracle Communications Network Analytics Data Director Benchmarking Guide. The resources in the `ocnadd_dbtier_custom_values.yaml` should match with this guide; if not, update them according to this guide.

Table 3-19 cnDBTier Customization Parameters

Parameter	Description	Version
`global.repository`	The value should be updated to point to the actual path of your Docker registry, for example `occne-repo-host:5000/occne`	24.3.0
`global.sitename`	This parameter must be set to the name of the current cluster	24.3.0
`global.domain`	Set it to the name of the Kubernetes cluster on which cnDBTier is installed, for example, `occne1-cgbu-cne-dbtier`	24.3.0
`global.namespace`	The Kubernetes namespace in which the cnDBTier is deployed	24.3.0
`global.storageClassName`	Storage class to be used. By default, `occne-dbtier-sc` will be the storage class. It can be changed to any storage class name currently configured in the cluster.	24.3.0
`global.mgmReplicaCount`	Default value to be used as in the file	24.3.0
`global.ndbReplicaCount`	The default value in the `ocnadd_dbtier_custom_values.yaml` file should be updated as follows: - Should be updated to 4 when cnDBTier is planned to be used as extended storage for xDRs - Default value (2) to be used in the file when cnDBTier is not used as extended storage	24.3.0
`global.ndbappReplicaCount`	Default value (2) to be used as in the file	24.3.0
`global.ndbappReplicaMaxCount`	Default value (4) to be used as in the file `global.ndbappReplicaMaxCount` should always be greater than `global.ndbappReplicaCount`	24.3.0

Table 3-19 (Cont.) cnDBTier Customization Parameters

Parameter	Description	Version
global.apiReplicaCount	The default value in the ocnadd_dbtier_custom_values.yaml file should be updated as follows: - In the case of no replication, the minimum number of SQL nodes required is 0.	24.3.0
global.ndb.datamemory	The default value in the ocnadd_dbtier_custom_values.yaml file should be updated as follows: - Should be updated to 96G when cnDBTier is planned to be used as extended storage for xDRs - Default value (1G) to be used in the file when cnDBTier is not used as extended storage	24.3.0
global.mgm.ndbdisksize	Default value (30Gi) to be used as in the file	24.3.0
global.ndb.ndbdisksize	The default value in the ocnadd_dbtier_custom_values.yaml file should be updated as follows: - Should be updated to ndb.resources.limits.memory + 30Gi when cnDBTier is planned to be used as extended storage for xDRs - Default value (30Gi) to be used in the file when cnDBTier is not used as extended storage	24.3.0
global.ndb.ndbbackupdisksize	Default value (30Gi) to be used as in the file	24.3.0
global.api.ndbdisksize	Default value (30Gi) to be used as in the file	24.3.0
global.ndbapp.ndbdisksize	Default value (20Gi) to be used as in the file	24.3.0
mgm.resources.limits.cpu	Default value (1) to be used as in the file	24.3.0
mgm.resources.limits.memory	Default value (1Gi) to be used as in the file	24.3.0
mgm.resources.requests.cpu	Default value (1) to be used as in the file	24.3.0
mgm.resources.requests.memory	Default value (1Gi) to be used as in the file	24.3.0
ndb.resources.limits.cpu	The default value in the ocnadd_dbtier_custom_values.yaml file should be updated as follows: - Should be updated to 8 when cnDBTier is planned to be used as extended storage for xDRs - Default value (1) to be used in the file when cnDBTier is not used as extended storage	24.3.0
ndb.resources.limits.memory	The default value in the ocnadd_dbtier_custom_values.yaml file should be updated as follows: - Should be updated to 128Gi when cnDBTier is planned to be used as extended storage for xDRs - Default value (4Gi) to be used in the file when cnDBTier is not used as extended storage	24.3.0

Table 3-19 (Cont.) cnDBTier Customization Parameters

Parameter	Description	Version
ndb.resources.requests.cpu	The default value in the ocnadd_dbtier_custom_values.yaml file should be updated as follows: - Should be updated to 8 when cnDBTier is planned to be used as extended storage for xDRs - Default value (1) to be used in the file when cnDBTier is not used as extended storage	24.3.0
ndb.resources.requests.memory	The default value in the ocnadd_dbtier_custom_values.yaml file should be updated as follows: - Should be updated to 128Gi when cnDBTier is planned to be used as extended storage for xDRs - Default value (4Gi) to be used in the file when cnDBTier is not used as extended storage	24.3.0
api.resources.limits.cpu	Default value (1) to be used as in the file	24.3.0
api.resources.limits.memory	Default value (1Gi) to be used as in the file	24.3.0
api.resources.requests.cpu	Default value (1) to be used as in the file	24.3.0
api.resources.requests.memory	Default value (1Gi) to be used as in the file	24.3.0
api.ndbapp.resources.limits.cpu	Default value (1) to be used as in the file	24.3.0
api.ndbapp.resources.limits.memory	Default value (1Gi) to be used as in the file	24.3.0
api.ndbapp.resources.requests.cpu	Default value (1) to be used as in the file	24.3.0
api.ndbapp.resources.requests.memory	Default value (1Gi) to be used as in the file	24.3.0
db-replicationsvc.dbreplsvcdeployments.enabled	Default value (false) to be used as in the file	24.3.0
db-replicationsvc.resources.limits.cpu	Default value (1) to be used as in the file	24.3.0
db-replicationsvc.resources.limits.memory	Default value (2048Mi) to be used as in the file	24.3.0
db-replicationsvc.resources.requests.cpu	Default value (0.6) to be used as in the file	24.3.0
db-replicationsvc.resources.requests.memory	Default value (1024Mi) to be used as in the file	24.3.0
db-monitor-svc.resources.limits.cpu	Default value (200m) to be used as in the file	24.3.0
db-monitor-svc.resources.limits.memory	Default value (500Mi) to be used as in the file	24.3.0
db-monitor-svc.resources.requests.cpu	Default value (200m) to be used as in the file	24.3.0

Table 3-19 (Cont.) cnDBTier Customization Parameters

Parameter	Description	Version
`db-monitorsvc.resources.requests.memory`	Default value (500Mi) to be used as in the file	24.3.0

> **Note**
>
> For more information about these parameters, see *Oracle Communications Cloud Native Core cnDBTier Installation, Upgrade, and Fault Recovery Guide*.

4
Upgrading OCNADD

This section provides information on how to upgrade an existing OCNADD deployment. The section describes the upgrade order for source NFs, CNC Console, cnDBTier, and the upgrade impact on the source NFs.

> **Note**
>
> - The OCNADD can be upgraded from a source release to a target release using CLI procedures as outlined in the following sections. These steps can also be followed for any hotfix upgrade.

4.1 Supported Upgrade Paths

The following table lists the supported upgrade paths for OCNADD:

Table 4-1 Supported Upgrade Path

Source Release	Target Release
25.2.100	25.2.101

4.2 Preupgrade Tasks

> **Note**
>
> - (Optional) Configure Helm parameters for OCCM based Certificate Management, For configuration details, see Helm Parameter Configuration for OCCM.
> - **Kafka PVC Storage Expansion**: In case there is a need to increase the Kafka broker PVC size due to increased throughput support in the target release, then the expansion of the PVC must be done in the source release before initiating the upgrade. To increase the size, follow the section 'Expanding Kafka Storage' from the *Oracle Communications Network Analytics Data Director User Guide*.

Before starting the procedure to upgrade OCNADD, perform the following tasks:

Note

- While performing an upgrade, you must align the `ocnadd-custom-values-25.2.101.yaml` file of the target release as per the `ocnadd/values.yaml` file of the source release or the older release. Do not enable any new feature during the upgrade. The parent or sub-charts values.yaml must not be changed while performing the upgrade, unless it is explicitly specified in this document. For information about enabling any new feature through Helm parameters, see *Oracle Communications Network Analytics Data Director User Guide*.
- The current resource profile in helm chart is for 270K MPS, if throughput is increased and it is required to use 360K MPS, update resources as per 360K MPS resource profile mentioned in *Oracle Communications Network Analytics Data Director Benchmarking Guide*.
- Ensure that if intra-TLS is disabled in the source release, certificates for all the required services are already created. For more details, see the 'Internal TLS Communication' section in the *Oracle Communications Network Analytics Suite Security Guide*.
- Ensure that the Network Policies are disabled before starting the upgrade. They can be re-enabled after the upgrade. For more details, refer to the section "Network Policy" in the *Oracle Communications Network Analytics Suite Security Guide*.
- During the upgrade, creation, update, or deletion of feed configurations is not allowed. Feed configurations can only be managed after a complete and successful upgrade to the target version.

Preupgrade Tasks

1. Fetch the images and charts of the target release as described in Installing OCNADD.
2. Keep a backup of the `ocnadd-custom-values.yaml` file and the extracted chart folder "ocnadd" of the source release as a backup before starting the upgrade procedure.
3. Create the certificates and secrets only for services that are newly added, or if updates are required for existing certificates in the target release, using the CA from the source release. See Create Secrets For Target Release.

   Note

   In the current release, the Admin service has been migrated to the worker group. Each worker group will now have its own dedicated Admin service. Therefore, ensure that an Admin service certificate is created for each worker group.

4. Upgrade is supported only in Centralized mode.
5. If any worker group is deployed in a separate namespace other than the default workergroup, perform the following pre-upgrade steps for each worker group; otherwise, skip this step.
   a. Remove HNC annotations from each worker group

i. Edit the namespace to remove the HNC annotation.

```
kubectl hns tree <management-group-namespace>
```

Example:

```
kubectl hns tree ocnadd-mgmt
```

Sample Output:

```
ocnadd-mgmt
+-- [s] ocnadd-wg1
```

ii. Back up each worker group namespace configuration

```
kubectl get namespace <worker-group-namespace> -o yaml > <worker-
group-namespace>.yaml
```

Example:

```
kubectl get namespace ocnadd-wg1 -o yaml > ocnadd-wg1.yaml
```

iii. Move the child worker group namespace to root

```
kubectl hns set <child-namespace> --root
```

Example:

```
kubectl hns set ocnadd-wg1 --root
```

iv. Verify the changes
Confirm that the management and worker group namespaces are now independent of each other.

```
kubectl hns tree ocnadd-mgmt
```

Expected Output:

```
ocnadd-mgmt
ocnadd-wg1
```

v. Special case – if the management namespace is also a child of another namespace
Run the following command:

```
kubectl annotate namespace <child-namespace> hnc.x-k8s.io/
subnamespace-of=<parent-namespace> --overwrite
```

Example:

```
kubectl hns tree ocnadd-site
```

Sample Output:

```
ocnadd-site
+-- [s] ocnadd-mgmt
    +-- [s] ocnadd-wg1
```

Then run:

```
kubectl annotate namespace ocnadd-wg1 hnc.x-k8s.io/subnamespace-
of=ocnadd-site --overwrite
```

Verify the changes:

```
kubectl hns tree ocnadd-site
```

Expected output:

```
ocnadd-site
+-- [s] ocnadd-mgmt
+-- [s] ocnadd-wg1 (1)
```

b. Update OCCM (if applicable): If OCCM is used to manage certificates, ensure that the worker group namespace is added to the custom-values.yaml file of OCCM. Then, perform the OCCM Helm upgrade to create the required role bindings for the separated worker group namespace.

6. Update the following helm chart files of the target release with the parameter values of the source release files:
   - update `ocnadd/ocdd-db-resource.sql`
   - update `ocnadd/templates/ocnadd-secret-hook.yaml`
   - update `custom-templates/ocnadd-custom-values.x.x.x.yaml`

Note

- Ensure the parameters such as serviceAccount, Role, RoleBinding are retained from the source release. These are the deployment parameters and should not be modified as part of the upgrade.
  Ensure to set the upgrade parameter to true as shown below:

```
serviceAccount:
    create: true
    name: <must be same as previous release>
    upgrade: true  ## ->> Update this to 'true', default is
false
clusterRole:
     create: true
     name: <must be same as previous release>
clusterRoleBinding:
     create: true
     name: <must be same as previous release>
```

7. Update the `pvcClaimSize` in the target release `ocnadd-custom-values-25.2.101.yaml` file.
8. Ensure to update the "`offsetsTopicReplicationFactor`" and "`transactionStateLogReplicationFactor`" in the target release `ocnadd-custom-values-25.2.101.yaml` file.
9. Ensure that the IP Family configuration in the target release matches with the IP configuration in the source release during the upgrade process. Changing the IP Family from IPv4 to IPv6 or vice-versa while performing upgrade is not supported.
10. Alter the mysql users, <admin_user>, ocddAppUsr and ocddPrivilegedUsr.

```
Check user in mysql console:-

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

mysql -h 127.0.0.1 -uroot -p $ Enter password:

Run the below command to check if auth plugin of mysql user:-

SELECT user,plugin FROM mysql.user;

Run below commands if and only if auth plugin is 'mysql_native_password'
for <admin_user>, ocddPrivilegedUsr and ocddAppUsr

For Admin User:-
ALTER USER '<admin_user>'@'%' IDENTIFIED WITH caching_sha2_password BY
'<admin_user_password>';
where, <admin_user> is the user created in Section "Creating an Admin User
in the Database"
Example:
ALTER USER 'ocdd'@'%' IDENTIFIED WITH caching_sha2_password BY 'ocdd';

For ocddPrivilegedUsr:-
ALTER USER 'ocddPrivilegedUsr'@'%' IDENTIFIED WITH caching_sha2_password
BY '<ocddPrivilegedUsr_password>';
```

```
Example:
ALTER USER 'ocddPrivilegedUsr'@'%' IDENTIFIED WITH caching_sha2_password
BY 'ocddPrivilegedPasswd';

For ocddAppUsr:-
ALTER USER 'ocddAppUsr'@'%' IDENTIFIED WITH caching_sha2_password BY
'<ocddAppUsr_password>';
Example:
ALTER USER 'ocddAppUsr'@'%' IDENTIFIED WITH caching_sha2_password BY
'ocddAppPasswd';
```

11. Take the manual backup of the OCNADD before starting the upgrade procedures. See, Performing OCNADD Manual Backup for taking a manual backup of the OCNADD.
12. Make sure to delete all the existing backup, restore, and ocnaddverify jobs before proceeding with the upgrade. Backup related jobs are "`ocnaddbackup`", "`ocnaddrestore`", "`ocnaddverify`" and "`ocnaddmanualbackup`".
    Use the following command to check for the currently running or completed backup jobs:

    ```
    kubectl get job -n <ocnadd-namespace>
    ```

    To delete jobs

    ```
    kubectl delete job -n <ocnadd-namespace> <job-name>
    ```

13. If occne ocnadd service monitor are not present, follow the procedure as described in Configuring ServiceMonitor in OCCNE-INFRA section.
14. If export configurations are present in the source release with `exportType=PCAP`, follow the steps below. Otherwise, this step can be skipped.
    a. Login to MySQL Database Instance with DB user of Data Director
    b. Run the following commands:
        i. `USE <CONFIGURATION-SCHEMA-NAME>;`

            For example:

            ```
            USE configuration_schema_dd;
            ```

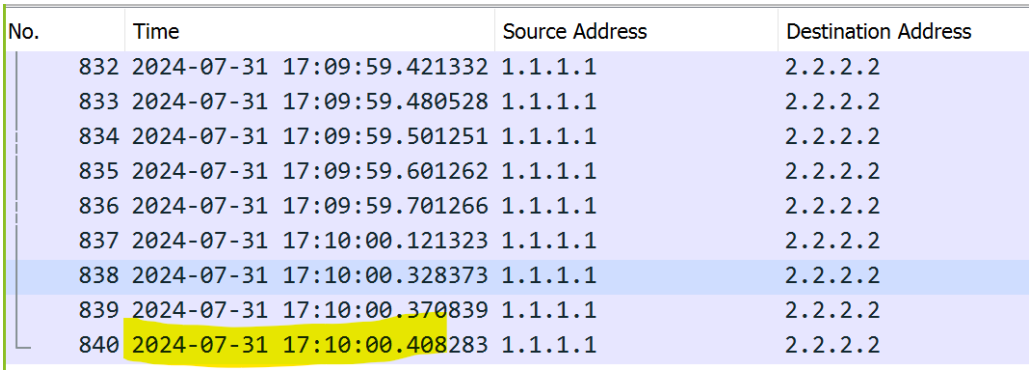
        ii.
            ```
            SELECT S.CONFIGURATION_NAME, S.REMAINING_EXPORT_TIME
            FROM DATA_EXPORT_STATUS S
            JOIN EXPORT_CONFIGURATION C
            ON C.CONFIGURATION_NAME = S.CONFIGURATION_NAME
            WHERE EXPORT_TYPE = 'PCAP';
            ```

            For example:

            ```
            SELECT S.CONFIGURATION_NAME, S.REMAINING_EXPORT_TIME
            FROM DATA_EXPORT_STATUS S
            JOIN EXPORT_CONFIGURATION C
            ON C.CONFIGURATION_NAME = S.CONFIGURATION_NAME
            AND C.EXPORT_TYPE = 'PCAP';
            ```

```
+--------------------+---------------------------+
| CONFIGURATION_NAME | REMAINING_EXPORT_TIME     |
+--------------------+---------------------------+
| pcap_export_test   | +56572-12-29T09:26:48.000Z |
| after-upgrade-pcap | +56573-01-30T05:01:30.000Z |
+--------------------+---------------------------+
```

c. Retrieve the Corresponding PCAP File: Locate the PCAP file present on the SFTP server for the corresponding configuration. Open the file and go to the last frame of the file. Use the timestamp (for example, highlighted below) to update the `REMAINING_EXPORT_TIME` column.



No.	Time	Source Address	Destination Address
832	2024-07-31 17:09:59.421332	1.1.1.1	2.2.2.2
833	2024-07-31 17:09:59.480528	1.1.1.1	2.2.2.2
834	2024-07-31 17:09:59.501251	1.1.1.1	2.2.2.2
835	2024-07-31 17:09:59.601262	1.1.1.1	2.2.2.2
836	2024-07-31 17:09:59.701266	1.1.1.1	2.2.2.2
837	2024-07-31 17:10:00.121323	1.1.1.1	2.2.2.2
838	2024-07-31 17:10:00.328373	1.1.1.1	2.2.2.2
839	2024-07-31 17:10:00.370839	1.1.1.1	2.2.2.2
840	2024-07-31 17:10:00.408283	1.1.1.1	2.2.2.2

For example:

```
Timestamp format in the file: 2024-07-31 17:10:00.408283
Timestamp format to use for updating the column:
2024-07-31T17:10:00.408Z
```

> **Note**
>
> If the export configuration exists for `exportType=PCAP` and the PCAP file is not present on the SFTP server, use the export configuration's "StartTime+1 second" from the UI to update the `REMAINING_EXPORT_TIME` column. In this case, the export will start from "StartTime+1 second."

d. Update `REMAINING_EXPORT_TIME`: Execute the following command on the MySQL terminal to update the `REMAINING_EXPORT_TIME` in the `DATA_EXPORT_STATUS` table and commit the changes:

```
UPDATE DATA_EXPORT_STATUS
SET REMAINING_EXPORT_TIME = '<UTC timestamp from the above step>'
WHERE CONFIGURATION_NAME = '<pcap export configuration name>';

COMMIT;
```

For example:

```
UPDATE DATA_EXPORT_STATUS
```

```
SET REMAINING_EXPORT_TIME = '2024-07-31T17:10:00.408Z'
WHERE CONFIGURATION_NAME = 'pcap_export_test';

COMMIT;
```

e. Repeat steps 'c' and 'd' to update `REMAINING_EXPORT_TIME` for all export configurations with `exportType=PCAP`.

> **Note**
>
> - Post upgrade, if the remaining export time gets updated again(during upgrade) with the incorrect timestamp, then repeat the above steps again after the upgrade.
> - Restart the `ocnaddconfiguration` and `ocnaddexport` services after the timestamp update.

15. Update the SAN entries for the Export service in the certificates. This will be managed by deleting the existing Export service secrets from OCCM and the DD management namespace. The upgrade will then create the required Export service secrets using the updated service name `ocnaddexport`.
    - Using OCCM UI, find and delete the OCNADD Export service `SECRET-CLIENT` and `SECRET-SERVER`.
    - Using kubectl, delete the OCNADD Export service `SECRET-SERVER` and `SECRET-CLIENT` from the management namespace.

4.3 Upgrade Sequence

The upgrade sequence of the procedures to be followed is described in this section.

4.3.1 Upgrade Order for Source NFs

By design, Kafka clients (producers and consumers) and brokers are bidirectionally compatible:

- Clients with a higher version of Kafka API can communicate with brokers of a lower version
- Clients with a lower Kafka API version can communicate with brokers of a higher version

Kafka clients and brokers exchange API version information during a handshake.

Upgrade the source NFs and the OCNADD independently of each other, and no specific upgrade order is required. Upgrade to a new release succeeds if compatibility is maintained. See, "Compatibility Matrix" in the *Oracle Communications Network Analytics Suite Release Notes*.

The OCNADD upgrade requires less time than source NFs (SCP, SEPP, and NRF) upgrade. It is advisable first to upgrade the OCNADD and verify the traffic flow post upgrade for any significant errors or potential roadblocks in the upgrade. If the NFs are upgraded first, rollback of large numbers of source NFs workers and gateway PODs might be required.

Upgrade Order:

1. Upgrade OCNADD
2. Upgrade source NFs (NRF, SCP, and SEPP)

4.3.2 Upgrade Order for CNC Console and cnDBTier

The following upgrade order is recommended for CNC Console and cnDBTier:

1. For CNC Console upgrade, see *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide*.
2. For OCNADD upgrade, see Upgrade Sequence.
3. For cnDBTier upgrade, see *Oracle Communications Cloud Native Core cnDBTier Installation, Upgrade, and Fault Recovery Guide*.

> **Note**
>
> - For more information on customizing database parameters required for the OCNADD, see cnDBTier Customization Parameters section.
> - To perform cnDBTier upgrade, see *Oracle Communications Cloud Native Core, cnDBTier Installation, Upgrade, and Fault Recovery Guide*.

4.4 Upgrade Impact on Source NFs and Third Party Consumers

Listed below are the observed upgrade impacts on the source NFs and the Third Party Consumers:

- In the case of a Kafka upgrade, the Kafka clients (NF producers and consumers) should not impact the Kafka API as the API compatibility is maintained between clients and brokers by Kafka.
- The Kafka binary upgrade is a two step procedure in which the Helm upgrade is performed twice. One upgrade for the Kafka binary upgrade and the second (optional) upgrade if the *InterBrokerProtocolVersion* is changed. During this upgrade, the source NFs (Kafka producers) may face communication disruption with Kafka brokers multiple times as each broker is expected to restart two times. The producer clients should adopt appropriate reconnection and metadata refresh mechanisms. Suppose the producers run with the 'acks=0' and 'retries=0' configurations. There is no guarantee of reliable message delivery between producers and Kafka brokers during the upgrade, as broker instances restart multiple times.
- The NF producers must maintain the list of servers in the *bootstrap-server* parameter instead of a single server in the *bootstrap-server* parameter.
- Assume that the OCNADD upgrade is performed at 20% (approximately) of the supported traffic rate. The upgrade is performed using the rolling upgrade strategy. The traffic flow between the NFs and the OCNADD Kafka may remain degraded for a few minutes. The traffic rate is expected to become normal after the upgrade when the NFs producers reconnect to the Kafka brokers.
- Assume that the OCNADD upgrade is performed at 20% (approximately) of the supported traffic rate. The upgrade is performed using the rolling upgrade strategy. The traffic flow between OCNADD consumer adapters and Third party consumers may remain degraded for a few minutes. The traffic rate is expected to be normal after the upgrade when the Kafka broker pods come up, all the consumer adapter pods have been upgraded, and consumer rebalancing is complete.

- During the OCNADD upgrade, Kafka retention helps prevent data loss for Third party consumers. However, expect some data duplication towards Third party consumers due to multiple Kafka consumer rebalancing.
- Plan the OCNADD upgrade during a maintenance window.

4.5 Upgrade Tasks

This section includes information about upgrading an existing OCNADD deployment.

When you upgrade an existing OCNADD deployment, the running set of containers and pods are replaced with the new set of containers and pods. However, If there is no change in the pod configuration, the running set of containers and pods are not replaced, unless there is a change in the service configuration of a microservice, the service endpoints will remain unchanged (NodePort and so on).

Note

- <Optional> Set a timeout interval of 15 minutes during the upgrade process.
- Ensure no OCNADD pod is in a failed state.
- Ensure that the defined in the Preupgrade Tasks are complete
- During an upgrade that affects all brokers, anticipate approximately a minute of downtime for Kafka brokers. Alternatively, upgrade brokers individually to avoid this downtime if applicable.
- Kafka upgrade along with PVC storage changes isn't supported.
- The creation of Consumer Adapter pods or services occurs when Data feeds are created via OCNADD GUI. Ensure the upgrade of these pods is set to "false" (`global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE` is set to false by default). To perform an upgrade, refer to the "Updating Consumer Adapter Parameters" section in the *Oracle Communications Network Analytics Data Director User Guide*.
- The creation of Correlation pods or services takes place when Correlation configurations are created from OCNADD GUI. Ensure the upgrade of these pods is set to "false" (`global.env.admin.OCNADD_CORR_UPGRADE_ENABLE` is set to false by default). To upgrade, refer to the "Updating Correlation Service Parameters" section in the *Oracle Communications Network Analytics Data Director User Guide*.

Upgrading Source Release(Centralized) to Target Release in Centralized Deployment Mode

Note

This scenario is applicable for 25.2.100 Centralized Deployment Mode upgrade to 25.2.1xx Centralized Deployment Mode.

1. Follow the below steps:
   a. Create a copy of the charts and custom values of the target release for the management group and for the default worker group or if the worker group in separate

namespace, from the ocnadd-package-25.2.101 folder. The user can create copy of helm chart folder and custom-values file in the following suggested way:

i. For Management Group: Create a copy of the following files from extracted folder:

```
# cd ocnadd-package-25.2.101
# cp -rf ocnadd ocnadd_mgmt
# cp custom-templates/ocnadd-custom-values-25.2.101.yaml ocnadd-
custom-values-mgmt-group.yaml
```

ii. For Worker Group: Create a copy of the following files from extracted folder:

```
# cp -rf ocnadd ocnadd_wg
# cp custom-templates/ocnadd-custom-values-25.2.101.yaml ocnadd-
custom-values-wg.yamlNote: Create multiple copies of the worker
group custom values file, depending on the number of worker groups
to be upgraded.
```

b. Upgrade using ocnadd_mgmt helm charts folder created for the management group:

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

For example:

```
helm upgrade ocnadd-mgmt -f ocnadd-custom-values-mgmt-group.yaml --
namespace ocnadd-deploy ocnadd_mgmt
```

c. Verify if the Management group upgrade is successful and the Admin service has been removed.

2. Now upgrade the default Worker Group or the Worker Group in separate namespace, Adapter and Correlation services using the target release.

a. Perform helm upgrade using helm charts folder created for the default worker group:

```
helm upgrade <source-release-name> -f ocnadd-custom-values-<worker-
group>.yaml --namespace <source-release-namespace> <target-release-helm-
chart>
```

Where,

- <source-release-name> is the release name of the source release deployment
- ocnadd-custom-values-<worker-group>.yaml is the custom values file created for default-worker-group or the Worker Group in separate namespace
- <source-release-namespace> is the OCNADD namespace of the source release
- <target-release-helm-chart> is the location of the Helm chart of the target release

For example:

```
helm upgrade ocnadd -f ocnadd-custom-values-wg.yaml --namespace ocnadd-
deploy ocnadd_wg
```

b. Once the above upgrade is successful, perform the Helm upgrade again to update the Adapters, Correlation, and other components using the custom_values file and the `ocnadd_wg` Helm charts folder created for the worker groups:

   i. Perform the upgrade of Adapters and other services using the worker group's custom_values file and charts folder:

      i. If no Correlation configurations are present:

```
helm upgrade <source-release-name> -f ocnadd-custom-values-
<worker-group>.yaml --namespace <source-release-namespace>
<target-release-helm-chart> --set
global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true
```

      For example:

```
helm upgrade ocnadd -f ocnadd-custom-values-wg.yaml --namespace
ocnadd-deploy ocnadd_wg --set
global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true
```

      ii. If correlation configurations are also present:

```
helm upgrade <source-release-name> -f ocnadd-custom-values-
<worker-group>.yaml --namespace <source-release-namespace>
<target-release-helm-chart> --set
global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true,global.env.ad
min.OCNADD_CORR_UPGRADE_ENABLE=true
```

> **Note**
>
> If the correlation configuration was enabled for extended storage (from 24.2.0 onwards), then also use the flag `global.env.admin.OCNADD_STORAGE_ADAPTER_UPGRADE_ENABLE=true` in the above command.

      For example:

```
helm upgrade ocnadd -f ocnadd-custom-values-wg.yaml --namespace
ocnadd-deploy ocnadd_wg --set
global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true,global.env.ad
min.OCNADD_CORR_UPGRADE_ENABLE=true
```

> **Note**
>
> If the correlation configuration was enabled for extended storage (from 24.2.0 onwards), then also use the flag `global.env.admin.OCNADD_STORAGE_ADAPTER_UPGRADE_ENABLE=true` in the above command.
>
> ```
> helm upgrade ocnadd -f ocnadd-custom-values-wg.yaml --
> namespace ocnadd-deploy ocnadd_wg --set
> global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true,global
> .env.admin.OCNADD_CORR_UPGRADE_ENABLE=true,global.env.admi
> n.OCNADD_STORAGE_ADAPTER_UPGRADE_ENABLE=true
> ```

iii. If ingress adapter configurations are also present:

```
helm upgrade <source-release-name> -f ocnadd-custom-values-
<worker-group>.yaml --namespace <source-release-namespace>
<target-release-helm-chart> --set
global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true,global.env.ad
min.OCNADD_CORR_UPGRADE_ENABLE=true,global.env.admin.OCNADD_INGRE
SS_ADAPTER_UPGRADE_ENABLE=true
```

For example:

```
helm upgrade ocnadd -f ocnadd-custom-values-wg.yaml --namespace
ocnadd-deploy ocnadd_wg --set
global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true,global.env.ad
min.OCNADD_CORR_UPGRADE_ENABLE=true,global.env.admin.OCNADD_INGRE
SS_ADAPTER_UPGRADE_ENABLE=true
```

3. Check the status of the upgrade, monitor the pods to come back to the RUNNING state, and wait for the traffic rate to be stabilized to the same rate as before the upgrade. Run the following command to check the upgrade status:

   ```
   helm history <release_name> --namespace <namespace-name>
   ```

   For example:

   For `ocnadd-mgmt`, use:

   ```
   helm history ocnadd-mgmt --namespace ocnadd-deploy
   ```

   For default worker-group, use:

   ```
   helm history ocnadd --namespace ocnadd-deploy
   ```

   Sample output:

   The description should be "`upgrade complete`".

4. Verify if the upgrade is successful using the following steps:

   a. All the pods that have respawned after the upgrade, have the latest age (in secs).

b. The Adapter pods also get respawned for any upgrade. The status can also be verified from GUI for respective Data Feeds. n case of any failure, follow the steps mentioned in the *Oracle Communications Network Analytics Data Director Troubleshooting Guide*.

5. Ensure that the Kafka topics for the required NFs are created post upgrade to current release. To create Kafka topics, see "Creating Kafka Topic for OCNADD" section in *Oracle Communications Network Analytics Data User Guide.*

6. Ensure that only required NFs aggregation is enabled on the OCNADD. For example, if the customer is only intendening to use SCP as source NF with OCNADD, then it is recommended to turn off all the other NF specific aggregation instances. The below modifications should be done in `ocnadd-custom-values-<worker-group>.yaml` file. For example, `ocnadd-custom-values-wg.yaml`:

```
Ensure that only required NFs aggregation is enabled on the Data Director,
for example if the customer is only intending to use SCP as source NF with
Data Director then it is recommended to turn off all the other NF specific
aggregation instances. The below  modifications should be done in ocnadd-
custom-values-default-wg-group.yaml

   global.ocnaddscpaggregation.enabled: true            ##---> default is
true
   global.ocnaddnrfaggregation.enabled: true            ##---> default is
true,   ## --> update the parameter to false
   global.ocnaddseppaggregation.enabled: true           ##---> default is
true,   ## --> update the parameter to false
   global.ocnaddbsfaggregation.enabled: true            ##---> default is
false
   global.ocnaddpcfaggregation.enabled: true            ##---> default is
false
```

7. Perform helm upgrade:

```
helm upgrade <source-release-name> -f ocnadd-custom-values-<worker-
group>.yaml --namespace <source-release-namespace> <target-release-helm-
chart>
```

Where:

- `<source-release-name>`: The release name of the source release deployment.
- `ocnadd-custom-values-<worker-group>.yaml`: The custom values file created for the default worker group or a specific worker group in a separate namespace.
- `<source-release-namespace>`: The OCNADD namespace of the source release.
- `<target-release-helm-chart>`: The location of the Helm chart for the target release.

For example:

```
helm upgrade ocnadd -f ocnadd-custom-values-default-wg.yaml --namespace
ocnadd-deploy ocnadd_default_wg
```

8. <Optional>Repeat the procedure from Step 2 onwards for upgrading the subsequent worker groups of the same deployment.

9. <Optional> To update the SNMP MIBs, follow the instructions in the section "OCNADD MIB FILES" of the *Oracle Communications Network Analytics Data Director User Guide*.

10. Follow the section "Post Upgrade Task" for post upgrade steps.

> **Note**
>
> In case intraTLS is disabled, additional certificates that are not required can be removed once the upgrade is verified and successful. For more details, see "Internal TLS Communication" section in the *Oracle Communications Network Analytics Suite Security Guide*.

4.5.1 Hotfix Upgrade

For any patch upgrade follow the Preupgrade Tasks section along with the ReadMe.txt file provided with the patch.

4.5.2 Create Secrets For Target Release

> **Note**
>
> (Optional): Users can modify the `service_values_template` file and retain only the specific service blocks for which certificates need to be updated. For example, while upgrading from non-centralized to centralized deployment mode, users can edit the `values_template` file. Similarly, to generate certificates for the management group, users can edit the `management_service_values_template` file.

```
Global Params:

[global]
countryName=<country>
stateOrProvinceName=<state>
localityName=<city>
organizationName=<org_name>
organizationalUnitName=<org_bu_name>
defaultDays=<days to expiry>

Root CA common name (e.g. rootca common_name=*.svc.domainName)
##root_ca

commonName=*.svc.domainName

Service common name for client and server and SAN(DNS/IP entries). (Make sure
to follow exact same format and provide an empty line at the end of each
service block)

[service-name-1]
client.commonName=client.cn.name.svc1
server.commonName=server.cn.name.svc1
IP.1=127.0.0.1
DNS.1=localhost

[service-name-2]
client.commonName=client.cn.name.svc2
```

```
server.commonName=server.cn.name.svc2
IP.1= 10.20.30.40
DNS.1 = *.svc2.namespace.svc.domainName
.
.
.
##end
```

Generating Certificates using CACert and CAKey in Target Release

Follow the steps to generate the secrets for the target release.

1. Navigate directory to the target release "ocnadd-package-25.2.101" folder, and then change the directory to the <ssl_certs>.
2. Run the generate_certs.sh script with the following command:

   ```
   ./generate_certs.sh -cacert <path to>/CAcert.pem -cakey <path to>/CAkey.pem
   ```

   Where, <path to> is the folder path where the CACert and CAKey are present.

   > **Note**
   >
   > In case the certificates are being generated for the worker group separately, ensure that the same CA certificate and private keys are used for generating the certificates as those used for generating the management group certificates. The same command, as mentioned below, can be used for the worker group certificate generation after the management group certificates have been generated:
   >
   > ```
   > ./generate_certs.sh -cacert <path to>/cacert.pem -cakey <path to>/
   > private/cakey.pem
   > ```

3. Select the mode of deployment:

   > **Note**
   >
   > - If an upgrade is being performed from Non-centralized to Non-Centralized deployment mode, select option 1.
   > - If an upgrade is being performed from Non-centralized to Centralized deployment mode, select option 2.

   ```
   "1" for non-centralized
   "2" for upgrade from non-centralized to centralized
   "3" for centralised
   "4" for simulator
   Select the mode of deployement (1/2/3) : 3
   ```

4. Select the namespace where the user would like to generate the certificates.

   ```
   Enter Kubernetes namespace: <your_working_namespace>
   ```

5. Select the type of service group the user would like to deploy. The below example is for the Management Group:

   ```
   Choose the group of services:
   1. management_group_services
   2. worker_group_services

   Choose a file by entering its corresponding number: (1 or 2) 1
   ```

6. Enter the domain name with which the user wants to change the default domain name(occne-ocdd) in the chosen service_values file which will be used to create the certificate.

   ```
   Please enter the domain name: <domain_name>
   ```

7. Enter SAN (DNS/IP entries) for any service if required.

   ```
   Do you want to add any IP for adding SAN entries to existing dd services
   (y/n): y
   ```

   If the user selects 'y,' a list of services will be displayed, and the user can add SAN entries for any of the listed services by selecting the corresponding service number.

   In the following example, the list of management services is presented for the user to add SAN entries. Enter the number corresponding to the service for which the user wants to input IP addresses. After choosing the service, provide IP addresses as input; otherwise, enter 'n' to exit."

   ```
   For the following services:
   1. ocnadduirouter
   2. ocnaddadminservice
   3. ocnaddalarm
   4. ocnaddconfiguration
   5. ocnaddhealthmonitoring
   6. ocnaddbackuprestore
   7. ocnaddredundancyagent
   8. ocnaddexport

   Enter the number corresponding to the service for which you want to add
   IP: 7
   Please enter IP for the service ocnaddredundancyagent or enter "n" to
   exit : 10.20.30.41
   Please enter IP for the service ocnaddredundancyagent or enter "n" to
   exit : n
   Do you want to add IP to any other service (y/n) : n
   ```

8. Select "y" when prompted to create CA.

   ```
   Do you want to create Certificate Authority (CA)? (y/n) y
   ```

9. Enter the passphrase for the CA key when prompted.

   ```
   Enter passphrase for CA Key file: <passphrase>
   ```

10. Select “y” when prompted to create CSR for each service.

    ```
    Create Certificate Signing Request (CSR) for each service?: Y
    ```

11. Select “y” when prompted to sign CSR for each service with CA Key.

    ```
    Would you like to sign CSR for each service with CA key? Y
    ```

12. If the centralized mode of deployment is selected while creating management group certificates, once the generation of management group certificates is completed, the user will be prompted to continue the certificate generation process for worker groups.

    ```
    Would you like to continue certificate creation for worker group? (y/n) y
    ```

    If "y" is selected, the script will execute to recreate the certificates for worker group. The script will repeat its execution from step 4 onwards. During worker group creation flow, select "worker_group_service_values" in step 5 and proceed.If "n" is selected, script will complete its execution

13. Run the following command to check if the secrets are created in the specified namespace:

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

14. Run the following command to describe any secret created by the script:

    ```
    kubectl describe secret <secret-name> -n <namespace>
    ```

Generate Certificate Signing Request (CSR) in Target Release

In this mode, users can generate Certificate Signing Requests (CSRs) for services listed in the corresponding service_values_template file and subsequently obtain certificates through their preferred methods, such as an external Certificate Authority (CA) server, Hashicorp Vault, or Venafi.

To generate CSRs for the services, follow the procedure explained in Generating Certificate Signing Request (CSR) section. Perform these steps in the <ssl_certs> folder of the target release.

> **Note**
>
> After completing Step 4 of the aforementioned procedure, it is crucial to append the existing Subject Alternative Name (SAN) entries from the source release to the "service_values_file" that will be utilized in the target release for the respective services.

4.6 Post Upgrade Task

> **Note**
>
> Steps 1, 2, and 3 are required only when OCCM is used to manage certificates in both the source and target releases.

1. **<Optional> Update OCCM-Managed Certificates (only if certificates need updating):** Update the following parameters in the `ocnadd-custom-values-25.2.101.yaml` file for the required worker group:
   - `global.certificates.occm.san.kafka.update_required`
   - `global.certificates.occm.san.kafka.uuid.client`
   - `global.certificates.occm.san.kafka.uuid.server`

   Kafka SAN Upgrade Example:

   ```
   global:
     certificates:
       occm:
         san:
           kafka:
             update_required: true  # Set to true, default is false
             uuid:
               client: 9138b974-2c89-4c9d-bc5c-0ca82752d50b  # Provide the
   UUID value of the certificate KAFKABROKER-SECRET-CLIENT-<namespace> from
   OCCM, where <namespace> is the Worker group namespace
               server: 5e765aeb-ae1b-426b-8481-f8f3dcdd645e  # Provide the
   UUID value of the certificate KAFKABROKER-SECRET-SERVER-<namespace> from
   OCCM, where <namespace> is the Worker group namespace
   ```

2. **<Conditional> Run Helm Upgrade for the Worker Group Namespace (only if step 1 was completed):**

   ```
   helm upgrade <worker-group-release-name> -f <worker-group-custom-values> -
   n <worker-group-ns> <ocnadd-helm-chart-location>
   ```

3. After running the Helm upgrade, new certificates will be created. Verify them through the OCCM UI. Kafka Brokers will restart after the Helm upgrade is completed and will begin using the newly created certificates.
4. **Migration of Kafka to Kraft Mode (Mandatory):**
   To migrate the Kafka cluster to Kraft mode from Zookeeper mode:
   a. Create the certificates for kraft-controller before migration. See Create Secrets For Target Release.
   b. Run the following `sed` command to update the `ocnadd/charts/ocnaddkafka/templates/scripts-controller-config.yaml` file:
      i. `sed -i "s/CONTROLLER:PLAINTEXT/CONTROLLER:SSL/g" ocnadd/charts/ocnaddkafka/templates/scripts-controller-config.yaml`

ii. Confirm that the sed command has successfully modified line 317 in the file, as shown below:

```
if [[ $ACLVALUE == false || $KAFKACLIENTAUTH == "required" ]]; then
   echo "Kraft ACL Configuration"
   export
AUTHORIZER="org.apache.kafka.metadata.authorizer.StandardAuthorizer"
   export
PROTOCOLMAP="INTERNAL_SSL:SSL,CONTROLLER:SSL,INTERNAL_PLAINTEXT:PLAI
NTEXT"
else
   echo "Kraft Default Configuration"
   export
PROTOCOLMAP="INTERNAL_SSL:SSL,CONTROLLER:SSL,INTERNAL_PLAINTEXT:PLAI
NTEXT" ## Sed command updated this line from CONTROLLER:PLAINTEXT
to CONTROLLER:SSL
fi
```

c. Edit *ocnadd/charts/ocnaddkafka/templates/configmap-kraftcontroller-occm.yaml* and replace the contents with below content:

```
# Pseudo Code
# if ( occm-enabled and ( ( centralised and !management ) or !
centralised ) and ( spawnKraftController or kraftEnabled ) )
{{ if and .Values.global.certificates.occm.enabled
        (and
                (or
                        ( and .Values.global.deployment.centralized
( not .Values.global.deployment.management ) )
( not .Values.global.deployment.centralized )
                )

( or .Values.global.kafka.spawnKraftController .Values.global.kafka.kraf
tEnabled )
        )
}}
apiVersion: v1
kind: ConfigMap
metadata:
  name: config-kraftcontroller-scripts
  namespace: {{.Values.global.cluster.nameSpace.name }}
  annotations:
    # This is what defines this resource as a hook. Without this line,
the
    # job is considered part of the release.
    "helm.sh/hook": pre-install,pre-upgrade
    "helm.sh/hook-weight": "-5"
data:
  server.json: |
    {{- include "ocnadd.occm.script.kraftcontroller.server" . | indent
4 }}
  client.json: |
    {{- include "ocnadd.occm.script.kraftcontroller.client" . | indent
4 }}
{{- end }}
```

Run helm lint command to verify if changes are done correctly or not.

```
helm lint ocnadd
```

d. Follow the procedure "Migration of Kafka Cluster to KRaft Mode", as defined in the *Oracle Communications Network Analytics Data Director User Guide*.

5. **Druid Cluster Integration with OCNADD Site:**

> **Note**
>
> In the previous release(s), extended storage was provided using the cnDBTier database. Migration from cnDBTier-based extended storage to Druid-based extended storage is not supported. If a user wants to move to Druid-based extended storage from cnDBTier-based extended storage, they must first remove the correlation, export, and trace configurations before integrating Druid-based extended storage. After Druid storage has been integrated with the OCNADD site, the user can recreate the correlation, export, and trace configurations.

This feature is introduced as part of extended storage in Data Director. To enable it, refer to the *"Druid Cluster Integration with OCNADD"* section in the *Oracle Communications Network Analytics Data Director User Guide*. It is recommended to enable this feature only after the release upgrade is completed. If Druid cluster integration is not enabled, extended storage using the cnDBTier database will continue to be available by default.

6. **Enabling RAM based Storage (Optional)**

> **Note**
>
> Enabling RAM-based storage from existing CEPH-based storage involves migration and may result in data loss from the existing Kafka cluster CEPH storage. It is recommended to perform this procedure during a maintenance window to minimize data loss.

This feature has been introduced to support RAM-based storage in the Kafka cluster. It provides higher throughput in scenarios where lower message retention with lower latency is required. To enable RAM-based storage in the Kafka cluster, refer to the "Enable RAM Storage in Kafka Cluster" section in the *Oracle Communications Network Analytics Data Director User Guide*. It is recommended to enable this feature after the release upgrade is completed.

5
Rolling Back OCNADD

This chapter describes the OCNADD rollback procedure from a target release to a previously supported version. In the current release, centralized deployment is supported, and OCNADD microservices will follow the centralized deployment mode with the management group and the worker group separation concerning microservices functions.

The rollback will be supported for the following cases:

- Upgrade was done from source releases to the target release in the Centralized deployment mode.

Table 5-1 Supported Rollback Paths

Source Release	Target Release
25.2.101	25.2.100

Rollback Steps

These steps are common for all the rollback cases. To roll back to a previous version, follow the steps as mentioned:

> **Note**
>
> - (Optional) A timeout interval of 15 minutes can be set while performing an upgrade, as only one pod of the OCNADD services is upgraded at a time.
> - Ensure that the status of the target version in the Helm history is not in a failed or error state. If Data Director has been integrated with a Druid cluster after upgrading to the current release and a rollback is required, it is recommended to remove all correlation and export configurations before performing the rollback.

Rollback Usecase : Upgrade Was Done from Source Releases (Centralized) to Target Release in the Centralized Deployment Mode

> **Note**
>
> This scenario is applicable for rollback from 25.2.101 Centralized Deployment Mode to 25.2.100 Centralized Deployment Mode.

To roll back to a previous working version in the target rollback release, follow these steps:

If any worker group deployed in a separate namespace (other than the default workergroup) was upgraded to 25.2.101, and you want to restore the HNS support as it was before the upgrade, perform the following pre-rollback steps for each worker group belonging to a separate namespace

1. Add back the HNC annotations

Use the following command:

```
kubectl hns set <child-namespace> --parent <parent-namespace>
```

Example:

```
kubectl hns tree ocnadd-mgmt
```

Sample Output:

```
ocnadd-mgmt
ocnadd-wg1
```

Then run:

```
kubectl hns set ocnadd-wg1 --parent ocnadd-mgmt
```

Verify the changes:

```
kubectl hns tree ocnadd-mgmt
```

Sample Output:

```
ocnadd-mgmt
+-- [s] ocnadd-wg1
```

2. Special case – if the management namespace was also a child of another namespace
Run the following command:

```
kubectl annotate namespace <child-namespace> hnc.x-k8s.io/subnamespace-
of=<parent-namespace> --overwrite
```

Example:

```
kubectl hns tree ocnadd-site
```

Sample Output:

```
ocnadd-site
+-- [s] ocnadd-mgmt
+-- [s] ocnadd-wg1 (1)
```

Then run:

```
kubectl annotate namespace ocnadd-wg1 hnc.x-k8s.io/subnamespace-of=ocnadd-
mgmt --overwrite
```

Verify the changes:

```
kubectl hns tree ocnadd-site
```

Sample Output:

```
ocnadd-site
└──── [s] ocnadd-mgmt
    └──── [s] ocnadd-wg1
```

3. Repeat the above steps for each worker group namespace.

Rollback of Management Group:

1. In case of default worker group, first disable the Admin service in default worker group custom_value file as below:

   ```
   global.ocnaddadmin.enabled: false
   ```

   Upgrade the default worker group in the current release

   ```
   helm upgrade <ocandd-default-workergroup-release-name>
   <helm_chart_current_release>  -n <ocndd-mgmt-namespace> -f ocnadd-custom-
   values-<current_release>.yaml
   ```

2. Check Revision of the Management Group Release to Rollback:

   ```
   helm history <ocandd-release-name> --namespace <ocndd-mgmt-namespace>
   ```

   Where,

   `<ocandd-release-name>` is the release name used for management group deployment.

   For example:

   ```
   helm history ocnadd --namespace ocnadd-deploy-mgmt
   ```

   Sample Helm history output:

   ```
   REVISION   UPDATED                    STATUS
   CHART                         APP VERSION         DESCRIPTION
   1          Fri May 08 04:57:43 2025    superseded
   ocnadd-25.2.100               25.2.100.0.0        Install complete
   (revision required for rollback)
   2          Fri May 08 05:07:43 2025    deployed
   ocnadd-25.2.101               25.2.101.0.0        Upgrade complete
   ```

3. Rollback to Required Revision of the management release:

   ```
   helm rollback <ocandd-mgmt-release-name> <REVISION> --namespace <ocndd-
   mgmt-namespace>
   ```

   Where,

<REVISION> is the revision number obtained in the previous step to which the services need to be rolled back.

For example:

```
helm rollback ocnadd 2 --namespace ocnadd-deploy
```

4. After the successful completion of the management rollback (i.e., no service is in the Init stage), execute the following command:

```
kubectl rollout restart deployment -n <ocndd-mgmt-namespace>
ocnaddconfiguration
```

Example:

```
kubectl rollout restart deployment -n ocnadd-deploy-mgmt
ocnaddconfiguration
```

5. After restarting the configuration service, verify that all required configurations have been initialized.

Rollback of Worker Groups

1. Check Revision for Rollback:

```
helm history <ocandd-release-name> --namespace <ocnadd-workergroup-
namespace>
```

Where,

<ocandd-release-name> is the release name used for default worker group deployment.

For example:

```
helm history ocnadd --namespace ocnadd-deploy
```

Sample Helm history output:

```
REVISION   UPDATED                    STATUS
CHART                         APP VERSION       DESCRIPTION
1          Fri May 08 04:57:43 2026    superseded
ocnadd-25.2.100               25.2.100.0.0        Install complete
(revision required for rollback)
2          Fri May 08 05:07:43 2026    superseded
 ocnadd-25.2.101               25.2.101.0.0        Upgrade complete
3          Fri May 08 05:15:43 2026    superseded
 ocnadd-25.2.101               25.2.101.0.0        Upgrade complete
4          Fri May 08 07:07:43 2026    deployed
ocnadd-25.2.101               25.2.101.0.0        Upgrade complete
```

2. Rollback to Required Revision or the worker group release:

```
helm rollback <ocandd-release-name> <REVISION> --namespace <ocndd-
workergroup-namespace>
```

Where,

<REVISION> is the revision number obtained in the previous step to which the services need to be rolled back.

For example:

```
helm rollback ocnadd 2 --namespace ocnadd-deploy
```

3. Rollback Adapter Services and other services to the previous version. Follow the below steps:
   a. Perform helm upgrade using the management group charts folder:
      i. If no correlation configurations are present:

```
helm upgrade <management-release-name> -f ocnadd-custom-values-
<mgmt-group>.yaml --namespace <ocnadd-namespace> <mgmt_helm_chart>
--set global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true
```

For example:

```
helm upgrade ocnadd-mgmt -f ocnadd-custom-values-mgmt-group.yaml --
namespace ocnadd-deploy ocnadd_mgmt --set
global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true
```

      ii. If correlation configurations are also present:

```
helm upgrade <management-release-name> -f ocnadd-custom-values-
<mgmt-group>.yaml --namespace <ocnadd-namespace> <mgmt_helm_chart>
--set
global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true,global.env.admin
.OCNADD_CORR_UPGRADE_ENABLE=true
```

> **Note**
>
> If the correlation configuration was enabled for extended storage (from 24.2.0 onwards) then also use the flag "`global.env.admin.OCNADD_STORAGE_ADAPTER_UPGRADE_ENABLE=true`" in the above command.

For example:

```
helm upgrade ocnadd-mgmt -f ocnadd-custom-values-mgmt-group.yaml --
namespace ocnadd-deploy ocnadd_mgmt --set
global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true,global.env.admin
.OCNADD_CORR_UPGRADE_ENABLE=true
```

> **Note**
>
> If the correlation configuration was enabled for extended storage (from 24.2.0 onwards), then also use the flag `global.env.admin.OCNADD_STORAGE_ADAPTER_UPGRADE_ENABLE=true` in the above command.
>
> ```
> helm upgrade ocnadd-mgmt -f ocnadd-custom-values-mgmt-
> group.yaml --namespace ocnadd-deploy ocnadd_mgmt --set
> global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true,global.en
> v.admin.OCNADD_CORR_UPGRADE_ENABLE=true,global.env.admin.OCNA
> DD_STORAGE_ADAPTER_UPGRADE_ENABLE=true
> ```

iii. If ingress adapter configurations are also present:

```
helm upgrade <management-release-name> -f ocnadd-custom-values-
<mgmt-group>.yaml --namespace <ocnadd-namespace> <mgmt_helm_chart>
--set
global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true,global.env.admin
.OCNADD_CORR_UPGRADE_ENABLE=true,global.env.admin.OCNADD_INGRESS_ADA
PTER_UPGRADE_ENABLE=true
```

For example:

```
helm upgrade ocnadd-mgmt -f ocnadd-custom-values-mgmt-group.yaml --
namespace ocnadd-deploy ocnadd_mgmt --set
global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true,global.env.admin
.OCNADD_CORR_UPGRADE_ENABLE=true,global.env.admin.OCNADD_INGRESS_ADA
PTER_UPGRADE_ENABLE=true
```

4. Check the status of the upgrade, and monitor the pods to come back to the RUNNING state.

> **Note**
>
> - If the rollback is unsuccessful, see troubleshooting steps outlined in the *Oracle Communications Network Analytics Data Director Troubleshooting Guide*.
> - On rollback completion, if the export, correlation, aggregation, ingress-adapter and/or consumer adapter services may not receive the configuration notification in timely manner and start processing data, then see the section "Invalid Subscription entry in the subscription table" in the *Oracle Communication Network Analytics Troubleshooting Guide*.
> - If the rollback is successful and intraTLS is disabled, then create the certificates of all the OCNADD services. For required certificates, see rollbacked release *Oracle Communication Network Analytics Security Guide*.

6
Uninstalling OCNADD

This chapter provides information on how to uninstall Oracle Communications Network Analytics Data Director (OCNADD).

When you uninstall a helm chart from the OCNADD deployment, it removes only the Kubernetes objects created during the installation.

> **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 Core, Cloud Native Environment (CNE) version of kube-api server.

> **Caution**
>
> - While deleting any OCNADD resources make sure to provide the corresponding namespace used in the deployment.
> - Based on requirement, make sure to retain the OCNADD backup before the uninstallation procedure. For more information, see Performing OCNADD Backup Procedures.
> - Ensure any configured datafeeds are deleted using the OCNADD GUI prior to performing the OCNADD uninstallation steps. For deletion of the datafeeds, refer to *Oracle Communications Network Analytics Data Director User Guide*.

Uninstalling OCNADD in Centralized Deployment Mode

To uninstall OCNADD in Centralized deployment mode, run the following steps:

1. Uninstall the worker groups one after another using the following command:

   ```
   helm uninstall <worker-group-release-name> --namespace <worker-group-namespace>
   ```

   For example:

   ```
   helm  uninstall ocnadd-wg1 -namespace dd-worker-group1
   ```

2. Clean up Kafka Configuration for all the worker groups.
   To clean up the Kafka configuration, perform the following steps for all the worker groups:
   a. To list the secrets in the namespace, run the following command:

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

b. To delete all the secrets related to Kafka, run the following command:

```
kubectl delete secret --all -n <worker-group-namespace>
```

c. To delete configmap used for Kafka, run the following command:

```
kubectl delete configmap --all -n <worker-group-namespace>
```

d. To delete PVCs used for Kafka,

i. run the following command, and list the PVCs used in the namespace:

```
kubectl get pvc -n <worker-group-namespace>
```

ii. run the following command, and delete the PVCs used by the brokers and zookeepers:

```
kubectl delete pvc --all -n <worker-group-namespace>
```

3. Run the following command to delete all the objects:

a. To delete all the Kubernetes objects:

```
kubectl delete all --all -n <wg-group-namespace>
```

> ⚠ **Caution**
>
> The command deletes all the Kubernetes objects of the specified namespace. In case, you have created the RBAC resources and service accounts before the helm installation in the same namespace, and these resources are required, then do not delete them.

b. Run the following command to delete the specific resources:

```
kubectl delete <resource-type> <resource-name> -n <wg-group-namespace>
```

4. Delete all the worker group's namespaces using the below command (This step is only needed if there are more than one worker group):

```
kubectl delete namesapce <worker-group-namespace>
```

> ⚠ **Caution**
>
> The command removes all the resources or objects created in the namespace. Therefore, ensure that you run the command only when you want to delete the namespace completely.

5. Uninstall the management group using the following command:

```
helm uninstall <management-release-name> --namespace <management-group-namespace>
```

For example:

```
helm  uninstall ocnadd-mgmt --namespace dd-mgmt-group
```

6. Clean up the Database.
   To clean up the database, perform the following steps:
   a. Log in to the MySQL client on SQL Node with the OCNADD user and password:

      ```
      mysql -h <IP_address of SQL Node> -u <ocnadduser> -p
      ```

   b. To clean up the configuration, alarm, and health database, run the following command: pass the database names in

      ```
      mysql> drop database <dbname>;
      ```

   c. To remove MySQL users while uninstalling OCNADD, run the following commands:

      ```
      SELECT user FROM mysql.user;
      DROP USER 'ocnaddappuser@'%';
      ```

Verifying Uninstallation

To verify the OCNADD centralized deployment uninstallation, run the following command:

1. Check if any worker group namespaces exists

   ```
   kubectl get namespaces
   ```

   a. If the output list have any worker group namespaces then perform the following:

      ```
      kubectl delete namespace <worker-group-namespace>
      ```

      > ⚠ **Caution**
      >
      > The command removes all the resources or objects created in the namespace. Therefore, ensure that you run the command only when you want to delete the namespace completely.

2. Check the management namespace

   ```
   kubectl get all -n <management-group-namespace>
   ```

   In case of successful uninstallation, no OCNADD resource is displayed in the command output.

   If the command output displays the OCNADD resources or objects, then perform the following procedure:

   a. Run the following command to delete all the objects:

i. To delete all the Kubernetes objects:

```
kubectl delete all --all -n <mgmt-group-namespace>
```

> **⚠ Caution**
>
> The command deletes all the Kubernetes objects of the specified namespace. In case, you have created the RBAC resources and service accounts before the helm installation in the same namespace, and these resources are required, then do not delete them.

ii. Run the following command to delete the specific resources:

```
kubectl delete <resource-type> <resource-name> -n <management-group-namespace>
```

iii. Run the following command to delete the management group namespace:

```
kubectl delete namespace <management-group-namespace>
```

For example:

```
kubectl delete namespace ocnadd-deploy
```

> **⚠ Caution**
>
> The command removes all the resources or objects created in the namespace. Therefore, ensure that you run the command only when you want to delete the namespace completely.

7

Migrating to OCCM Managed Certificates

> **⚠ Caution**
>
> - It is expected that there will be downtime when the services are migrated to use the new certificates generated by the OCCM. The amount of downtime will depend on the method of migration performed as described below.
> - This procedure is applicable when certificates are being migrated within the same release.

This section provides information on how to migrate the certificates initially created by following the section "Configuring SSL or TLS Certificates" during OCNADD installation.

The below steps can be followed to use certificates created by OCCM:

1. **Upgrading the Helm Charts**: No configuration or the existing data will be lost. Expected downtime will be equal to time taken to upgrade worker group + time taken to upgrade consumer adapter and correlation + time taken for kafka-broker and zookeeper to stabilize.

7.1 Upgrading the Helm Charts

> **⚠ Caution**
>
> Migration is supported only for current release version.

To manually create certificates for OCNADD, follow these steps:

1. Follow the steps to create secrets for OCCM for each management and worker group namespace as specified in the OCCM Prerequisites for Installing OCNADD section.
2. Enable the OCCM based certificate management in the Management and Worker group custom-values. For descriptions of the Helm parameters required for enabling OCCM, see Helm Parameter Configuration for OCCM.
3. Upgrade the Management group helm chart:

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

   For example:

   ```
   helm upgrade ocnadd-mgmt -f ocnadd-custom-values-mgmt-group.yaml --
   namespace dd-mgmt-group ocnadd_mgmt
   ```

> **Note**
>
> The Admin service restart is expected until the Worker group certificate migrations are completed.

4. Upgrade the Worker group helm chart:

   ```
   helm upgrade <worker-group-release-name> -f ocnadd-custom-values-<wg-group>.yaml --namespace <worker-group-namespace> <helm_chart>
   ```

   For example:

   ```
   helm upgrade ocnadd-wg1 -f ocnadd-custom-values-wg1-group.yaml --namespace dd-worker-group1 ocnadd_wg1
   ```

5. Update the Worker group namespace in `global.env.admin.OCNADD_UPGRADE_WG_NS` of the Manamgment group custom-values.yaml file:

   ```
   global:
     env:
       admin:
         OCNADD_UPGRADE_WG_NS: dd-worker-group1                    # Where dd-worker-group1 is the namespace of the worker group service
   ```

6. Perform helm upgrade using the Management group charts:

   ```
   helm upgrade <management-group-release-name> -f <management-group-custom-values> -n <management-group-ns> <ocnadd-helm-chart-location> --set global.env.admin.OCNADD_INGRESS_ADAPTER_UPGRADE_ENABLE=true,global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true,global.env.admin.OCNADD_CORR_UPGRADE_ENABLE=true,global.env.admin.OCNADD_STORAGE_ADAPTER_UPGRADE_ENABLE=true
   ```

   For example:

   ```
   helm upgrade ocnadd-mgmt -f ocnadd-custom-values-mgmt-group.yaml -n dd-mgmt-group ocnadd_mgmt --set global.env.admin.OCNADD_INGRESS_ADAPTER_UPGRADE_ENABLE=true,global.env.admin.OCNADD_ADAPTER_UPGRADE_ENABLE=true,global.env.admin.OCNADD_CORR_UPGRADE_ENABLE=true,global.env.admin.OCNADD_STORAGE_ADAPTER_UPGRADE_ENABLE=true
   ```

7. Now, delete and re-create all the Data Feeds with the same name and select "Resume from point of failure" in the "Handle Failure" page.
8. If multiple Worker groups are present, repeat steps 5 to 7 for each Worker group.

8
Fault Recovery

This chapter provides information about fault recovery for OCNADD deployment.

8.1 Overview

This section describes procedures to perform the backup and restore for the Oracle Communications Network Analytics Data Director (OCNADD) deployment. The backup and restore procedures will be used in the fault recovery of the OCNADD. The OCNADD operators can take only the OCNADD instance specific database and required OCNADD Kafka metadata backup and restore them either on the same or a different Kubernetes cluster.

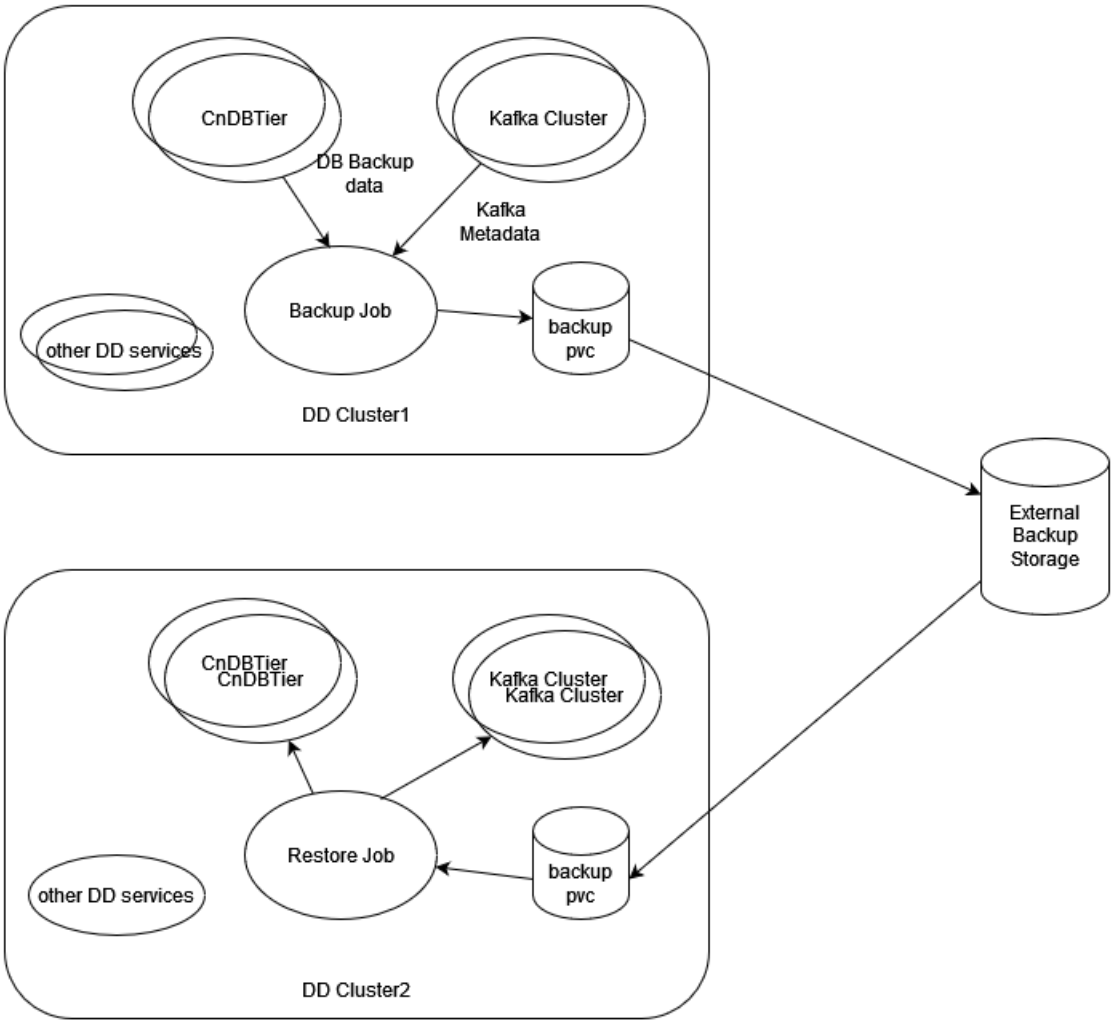
The backup and restore procedures are helpful in the following scenarios:

- OCNADD fault recovery
- OCNADD cluster migration
- OCNADD setup replication from production to development or staging
- OCNADD cluster upgrade to new CNE version or K8s version

The OCNADD backup contains the following data:

- OCNADD database(s) backup
- OCNADD Kafka metadata backup including the topics and partitions information

> **Note**
>
> If the deployed helm charts and the customized `ocnadd-custom-values-25.2.101.yaml` for the current deployment is stored in the customer helm or artifact repository, then the helm chart and `ocnadd-custom-values-25.2.101.yaml` backup is not required.



OCNADD Database(s) Backup

The OCNADD database consists of the following:

- **Configuration data**: This data is exclusive for the given OCNADD instance. Therefore, an exclusive logical database is created and used by an OCNADD instance to store its configuration data and operator driven configuration. Operators can configure the OCNADD instance specific configurations using the Configuration UI service through the Cloud Native Configuration Console.
- **Alarm configuration data**: This data is also exclusive to the given OCNADD instance. Therefore, an exclusive logical database is created and used by an OCNADD Alarm service instance to store its alarm configuration and alarms.
- **Health monitoring data**: This data is also exclusive to the given OCNADD instance. Therefore, an exclusive logical database is created and used by an OCNADD Health monitoring service instance to store the health profile of various other services.

The database backup job uses the mysqldump utility.

The Scheduled regular backups helps in:

- Restoring the stable version of the data directory databases
- Minimize significant loss of data due to upgrade or rollback failure

- Minimize loss of data due to system failure
- Minimize loss of data due to data corruption or deletion due to external input
- Migration of the database information from one site to another site

OCNADD Kafka Metadata Backup

The OCNADD Kafka metadata backup contains the following information:

- Created topics information
- Created partitions per topic information

8.1.1 Fault Recovery Impact Areas

The following table shares information about impact of OCNADD fault recovery scenarios:

Table 8-1 OCNADD Fault Recovery Scenarios Impact Information

Scenario	Requires Fault Recovery or Reinstallation of CNE?	Requires Fault Recovery or Reinstallation of cnDBTier?	Requires Fault Recovery or Reinstallation of Data Director?
Scenario 1: Deployment Failure Recovering OCNADD when its deployment is corrupted	No	No	Yes
Scenario 2: cnDBTier Corruption	No	Yes	No However, it requires to restore the databases from backup and Helm upgrade of the same OCNADD version to update the OCNADD configuration. For example, change in cnDBTier service information, such as cnDB endpoints, DB credentials, and so on.
Scenario 3: Database Corruption Recovering from corrupted OCNADD configuration database	No	No	No However, it requires to restore the databases from old backup.
Scenario 4: Site Failure Complete site failure due to infrastructure failure, for example, hardware, CNE, and so on.	Yes	Yes	Yes

8.1.2 Prerequisites

Before you run any fault recovery procedure, ensure that the following prerequisites are met:

- cnDBTier must be in a healthy state and available on a new or newly installed site where the restore needs to be performed
- Automatic backup should be enabled for OCNADD.
- Docker images used during the last installation or upgrade must be retained in the external data storage or repository
- The `ocnadd-custom-values-25.2.101.yaml` used at the time of OCNADD deployment must be retained. If the `ocnadd-custom-values-25.2.101.yaml` file is not retained, it is required to be recreated manually. This task increases the overall fault recovery time.

Important

Do not change DB Secret or cnDBTier MySQL FQDN or IP or PORT configurations during backup and restore.

8.2 Backup and Restore Flow

Important

- It is recommended to keep the backup storage in the external storage that can be shared between different clusters. This is required, so that in an event of a fault, the backup is accessible on the other clusters. The backup job should create a PV or PVC from the external storage provided for the backup.
- In case the external storage is not made available for the backup storage, the customer should take care to copy the backups from the associated backup PV in the cluster to the external storage. The security and connectivity to the external storage should be managed by the customer. To copy the backup from the backup PV to the external server, follow Verifying OCNADD Backup.
- The restore job should have access to the external storage so that the backup from the external storage can be used for the restoration of the OCNADD services. In case the external storage is not available, the backup should be copied from the external storage to the backup PV in the new cluster. For information on the procedure, see Verifying OCNADD Backup.
- In case of two site redundancy feature is enabled then respective site backup should be used to restore the site during failure recovery.

> **Note**
>
> At a time, only one among the three backup jobs (ocnaddmanualbackup, ocnaddverify or ocnaddrestore) can be running. If any existing backup job is running, that job needs to be deleted to spawn the new job.
>
> ```
> kubectl delete job.batch/<ocnadd*> -n <namespace>
>
> where namespace = Namespace of OCNADD deployment
>       ocnadd* = Running jobs in the namespace (ocnaddmanualbackup,
> ocnaddverify or ocnaddrestore)
>
> Example:
>       kubectl delete job.batch/ocnaddverify -n ocnadd-deploy
> ```

Backup

1. The OCNADD backup is managed using the backup job created at the time of installation. The backup job runs as a cron job and takes the daily backup of the following:
   - OCNADD databases for configuration, alarms, and health monitoring
   - OCNADD Kafka metadata including topics and partitions, which are previously created
2. The automated backup job spawns as a container and takes the backup at the scheduled time. The backup file `OCNADD_Backup_DD-MM-YYYY_hh-mm-ss.tar.bz2` is created and stored in the PV mounted on the path `/work-dir/backup` by the backup container.
3. On-demand backup can also be created by creating the backup container. For more information, see Performing OCNADD Manual Backup.
4. The backup can be stored on external storage.

Restore

1. The OCNADD restore job must have access to the backups from the backup PV/PVC.
2. The restore uses the latest backup file available in the backup storage if the BACKUP_FILE argument is not given.
3. The restore job performs the restore in the following order:
   a. Restore the OCNADD database(s) on the cnDBTier.
   b. Restore the Kafka metadata.

8.3 OCNADD Backup

The OCNADD backup is of two types:

- Automated backup
- Manual backup

Automated Backup

- This is managed by the automated K8s job configured during the installation of the OCNADD. For more information, see Updating the OCNADD Backup Cronjob step.

- It is a scheduled job and runs daily at the configured time to collect the OCNADD backup and creates the backup file `OCNADD_Backup_DD-MM-YYYY_hh-mm-ss.tar.bz2`.

Manual Backup

- This is managed by an on-demand job.
- A new K8s job will be created on executing the Performing OCNADD Manual Backup procedure.
- The job completes after taking the backup. Follow Verifying OCNADD Backup procedure to verify the generated backup.

8.4 Performing OCNADD Backup Procedures

8.4.1 Performing OCNADD Manual Backup

Note

If you have used OCCM to create certificates, then use the `ocnadd_manualBackup_occm.yaml` file. Here is the file template:

```
# ocnadd_manualBackup_occm.yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: ocnaddmanualbackup
  namespace: ocnadd-deploy
spec:
  template:
    metadata:
      name: ocnaddmanualbackup
      labels:
        role: backup
    spec:
      automountServiceAccountToken: false
      volumes:
      - name: backup-vol
        persistentVolumeClaim:
          claimName: backup-mysql-pvc
      - name: config-vol
        configMap:
          name: config-backuprestore-scripts
      - name: client-server-certificate-client
        secret:
          secretName: ocnaddbackuprestore-secret-client
      - name: client-server-certificate-server
        secret:
          secretName: ocnaddbackuprestore-secret-server
      - name: client-server-certificate-ca
        secret:
          secretName: occm-ca-secret
      - name: truststore-keystore-volume
        emptyDir: {}
      serviceAccountName: ocnadd-deploy-sa-ocnadd
      securityContext:
        runAsUser: 1000
        runAsGroup: 1000
        fsGroup: 1000
        runAsNonRoot: true
        seccompProfile:
          type: RuntimeDefault
      containers:
      - name: ocnaddmanualbackup
        image: ocdd-docker.dockerhub-phx.oci.oraclecorp.com/ocdd.repo/
```

```
ocnaddbackuprestore:2.0.10
        securityContext:
          allowPrivilegeEscalation: false
          capabilities:
            drop: ["ALL"]
        volumeMounts:
          - mountPath: "work-dir"
            name: backup-vol
          - mountPath: "config-backuprestore-scripts"
            name: config-vol
          - name: client-server-certificate-client
            mountPath: /var/securityfiles/certs
            readOnly: true
          - name: truststore-keystore-volume
            mountPath: /var/securityfiles/keystore
        env:
          - name: HOME
            value: /home/ocnadd
          - name: KS_PASS
            valueFrom:
              secretKeyRef:
                name: occm-truststore-keystore-secret
                key: keystorekey
          - name: TS_PASS
            valueFrom:
              secretKeyRef:
                name: occm-truststore-keystore-secret
                key: truststorekey
          - name: DB_USER
            valueFrom:
              secretKeyRef:
                name: db-secret
                key: MYSQL_USER
          - name: DB_PASSWORD
            valueFrom:
              secretKeyRef:
                name: db-secret
                key: MYSQL_PASSWORD
          - name: BACKUP_DATABASES
            value: ALL
          - name: BACKUP_ARG
            value: ALL
        command:
        - /bin/sh
        - -c
        - |
          cp /config-backuprestore-scripts/*.sh /home/ocnadd
          chmod +x /home/ocnadd/*.sh
          cp /config-backuprestore-scripts/command.properties /home/
ocnadd
          chmod 660 /home/ocnadd/command.properties
          sed -i "s*\$KS_PASS*$KS_PASS*" /home/ocnadd/
command.properties
          sed -i "s*\$TS_PASS*$TS_PASS*" /home/ocnadd/
command.properties
          mkdir /work-dir/backup
```

```
            echo "Executing manual backup script"
            bash /home/ocnadd/backup.sh $BACKUP_DATABASES $BACKUP_ARG
            ls -lh /work-dir/backup
      initContainers:
      - name: ocnaddinitcontainer
        image: ocdd-docker.dockerhub-phx.oci.oraclecorp.com/utils.repo/
jdk21-openssl:1.1.0
        securityContext:
          allowPrivilegeEscalation: false
          readOnlyRootFilesystem: true
          capabilities:
            drop: ["ALL"]
        env:
             - name: SERVER_CERT_FILE
               value: /var/securityfiles/certs/server/
ocnaddbackuprestore-servercert.pem
             - name: CLIENT_CERT_FILE
               value: /var/securityfiles/certs/client/
ocnaddbackuprestore-clientcert.pem
             - name: SERVER_KEY_FILE
               value: /var/securityfiles/certs/server/
ocnaddbackuprestore-serverprivatekey.pem
             - name: CLIENT_KEY_FILE
               value: /var/securityfiles/certs/client/
ocnaddbackuprestore-clientprivatekey.pem
             - name: SERVER_KEY_STORE
               value: /var/securityfiles/keystore/serverKeyStore.p12
             - name: CLIENT_KEY_STORE
               value: /var/securityfiles/keystore/clientKeyStore.p12
             - name: TRUST_STORE
               value: /var/securityfiles/keystore/trustStore.p12
             - name: CA_CERT_FILE
               value: /var/securityfiles/certs/ca/cacert.pem
             - name: KS_PASS
               valueFrom:
                 secretKeyRef:
                   name: occm-truststore-keystore-secret
                   key: keystorekey
             - name: TS_PASS
               valueFrom:
                 secretKeyRef:
                   name: occm-truststore-keystore-secret
                   key: truststorekey
        command: ['/bin/sh']
        args: ['-c', "openssl pkcs12 -export -inkey $CLIENT_KEY_FILE -
in $CLIENT_CERT_FILE -out $CLIENT_KEY_STORE -password pass:$KS_PASS &&
keytool -importcert -file $CA_CERT_FILE -alias ocnaddcacert -
keystore $TRUST_STORE -storetype PKCS12 -storepass $TS_PASS -
noprompt;"]
        volumeMounts:
          - name: truststore-keystore-volume
            mountPath: /var/securityfiles/keystore
          - name: client-server-certificate-client
            mountPath: /var/securityfiles/certs/client
          - name: client-server-certificate-server
            mountPath: /var/securityfiles/certs/server
```

```
            - name: client-server-certificate-ca
              mountPath: /var/securityfiles/certs/ca
      restartPolicy: Never
```

Perform the following steps to take the manual backup:

1. Go to `custom-templates` folder in the extracted ocnadd-release package and update the `ocnadd_manualBackup.yaml` or the `ocnadd_manualBackup_occm.yaml` file with the following information:
   a. Value for BACKUP_DATABASES can be set to ALL (that is, healthdb_schema, configuration_schema, and alarm_schema) or the individual DB names can also be passed. By default, the value is 'ALL'.
   b. Value of BACKUP_ARG can be set to ALL, DB, or KAFKA. By default, the value is ALL.
   c. Update other values as follows:

```
apiVersion: batch/v1
kind: Job
metadata:
  name: ocnaddmanualbackup
  namespace: ocnadd-deploy           #---> update the namespace
------------
spec:
      serviceAccountName: ocnadd-deploy-sa-ocnadd
#---> update the service account name. Format:<serviceAccount>-sa-ocnadd
------------
      containers:
      - name: ocnaddmanualbackup
        image: <repo-path>/ocdd.repo/ocnaddbackuprestore:2.0.10
#---> update repository path
------------
      initContainers:
      - name: ocnaddinitcontainer
        image: <repo-path>/utils.repo/jdk21-openssl:1.1.0
#---> update repository path
        env:
          - name: BACKUP_DATABASES
            value: ALL
          - name: BACKUP_ARG
            value: ALL
```

2. Run the below command to run the job:

```
kubectl create -f ocnadd_manualBackup.yaml
```

Or, use the following command if OCCM is used:

```
kubectl create -f ocnadd_manualBackup_occm.yaml
```

8.4.2 Verifying OCNADD Backup

> ⚠ **Caution**
>
> The connectivity between the external storage through either PV/PVC or network connectivity must be ensured.

Note

If you have used OCCM to create certificates, then use the `ocnadd_verify_backup_occm.yaml` file. Here is the file template:

```
# ocnadd_verify_backup_occm.yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: ocnaddverify
  namespace: ocnadd-deploy
spec:
  template:
    metadata:
      name: ocnaddverify
    spec:
      automountServiceAccountToken: false
      volumes:
      - name: backup-vol
        persistentVolumeClaim:
            claimName: backup-mysql-pvc
      - name: config-vol
        configMap:
          name: config-backuprestore-scripts
      - name: client-server-certificate-client
        secret:
          secretName: ocnaddbackuprestore-secret-client
      - name: client-server-certificate-server
        secret:
          secretName: ocnaddbackuprestore-secret-server
      - name: client-server-certificate-ca
        secret:
          secretName: occm-ca-secret
      - name: truststore-keystore-volume
        emptyDir: {}
      serviceAccountName: ocnadd-deploy-sa-ocnadd
      securityContext:
        runAsUser: 1000
        runAsGroup: 1000
        fsGroup: 1000
        runAsNonRoot: true
        seccompProfile:
          type: RuntimeDefault
      containers:
      - name: ocnaddverify
        image: ocdd-docker.dockerhub-phx.oci.oraclecorp.com/ocdd.repo/
ocnaddbackuprestore:2.0.10
        securityContext:
          allowPrivilegeEscalation: false
          capabilities:
            drop: ["ALL"]
        volumeMounts:
          - mountPath: "work-dir"
            name: backup-vol
          - mountPath: "config-backuprestore-scripts"
```

```
              name: config-vol
            - name: client-server-certificate-client
              mountPath: /var/securityfiles/certs
              readOnly: true
            - name: truststore-keystore-volume
              mountPath: /var/securityfiles/keystore
          env:
            - name: HOME
              value: /home/ocnadd/
            - name: KS_PASS
              valueFrom:
                secretKeyRef:
                  name: occm-truststore-keystore-secret
                  key: keystorekey
            - name: TS_PASS
              valueFrom:
                secretKeyRef:
                  name: occm-truststore-keystore-secret
                  key: truststorekey
            - name: DB_USER
              valueFrom:
                secretKeyRef:
                  name: db-secret
                  key: MYSQL_USER
            - name: DB_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: db-secret
                  key: MYSQL_PASSWORD
          command:
          - /bin/sh
          - -c
          - |
            cp /config-backuprestore-scripts/*.sh /home/ocnadd/
            chmod +x /home/ocnadd/*.sh
            cp /config-backuprestore-scripts/command.properties /home/
ocnadd
            chmod 660 /home/ocnadd/command.properties
            sed -i "s*\$KS_PASS*$KS_PASS*" /home/ocnadd/
command.properties
            sed -i "s*\$TS_PASS*$TS_PASS*" /home/ocnadd/
command.properties
            mkdir -p /work-dir/backup
            echo "Checking backup path"
            ls -lh /work-dir/backup
            sleep 20m
        initContainers:
        - name: ocnaddinitcontainer
          image: ocdd-docker.dockerhub-phx.oci.oraclecorp.com/utils.repo/
jdk21-openssl:1.1.0
          securityContext:
            allowPrivilegeEscalation: false
            readOnlyRootFilesystem: true
            capabilities:
              drop: ["ALL"]
          env:
```

```
              - name: SERVER_CERT_FILE
                value: /var/securityfiles/certs/server/
ocnaddbackuprestore-servercert.pem
              - name: CLIENT_CERT_FILE
                value: /var/securityfiles/certs/client/
ocnaddbackuprestore-clientcert.pem
              - name: SERVER_KEY_FILE
                value: /var/securityfiles/certs/server/
ocnaddbackuprestore-serverprivatekey.pem
              - name: CLIENT_KEY_FILE
                value: /var/securityfiles/certs/client/
ocnaddbackuprestore-clientprivatekey.pem
              - name: SERVER_KEY_STORE
                value: /var/securityfiles/keystore/serverKeyStore.p12
              - name: CLIENT_KEY_STORE
                value: /var/securityfiles/keystore/clientKeyStore.p12
              - name: TRUST_STORE
                value: /var/securityfiles/keystore/trustStore.p12
              - name: CA_CERT_FILE
                value: /var/securityfiles/certs/ca/cacert.pem
              - name: KS_PASS
                valueFrom:
                  secretKeyRef:
                    name: occm-truststore-keystore-secret
                    key: keystorekey
              - name: TS_PASS
                valueFrom:
                  secretKeyRef:
                    name: occm-truststore-keystore-secret
                    key: truststorekey
          command: ['/bin/sh']
          args: ['-c', "openssl pkcs12 -export -inkey $CLIENT_KEY_FILE -
in $CLIENT_CERT_FILE -out $CLIENT_KEY_STORE -password pass:$KS_PASS &&
keytool -importcert -file $CA_CERT_FILE -alias ocnaddcacert -
keystore $TRUST_STORE -storetype PKCS12 -storepass $TS_PASS -
noprompt;"]
          volumeMounts:
            - name: truststore-keystore-volume
              mountPath: /var/securityfiles/keystore
            - name: client-server-certificate-client
              mountPath: /var/securityfiles/certs/client
            - name: client-server-certificate-server
              mountPath: /var/securityfiles/certs/server
            - name: client-server-certificate-ca
              mountPath: /var/securityfiles/certs/ca
        restartPolicy: Never
```

To verify the backup, perform the following steps:

1. Go to the `custom-templates` folder in the extracted ocnadd-release package and update the `ocnadd_verify_backup.yaml` or the `ocnadd_verify_backup_occm.yaml` file with the following information:
   a. Sleep time is configurable, update it if required (the default value is set to 10m).

b. Update other values as follows:

```
apiVersion: batch/v1
kind: Job
metadata:
  name: ocnaddverify
  namespace: ocnadd-deploy         #---> update the namespace
------------
spec:
      serviceAccountName: ocnadd-sa-ocnadd                          #---
> update the service account name. Format:<serviceAccount>-sa-ocnadd
------------
      containers:
      - name: ocnaddverify
        image: <repo-path>/ocdd.repo/ocnaddbackuprestore:2.0.10
#---> update repository path
------------
      initContainers:
      - name: ocnaddinitcontainer
        image: <repo-path>/utils.repo/jdk21-openssl:1.1.0           #---
> update repository path
```

2. Run the below command to create the job:

   ```
   kubectl create -f ocnadd_verify_backup.yaml
   ```

   Or, use the following command if OCCM is used:

   ```
   kubectl create -f ocnadd_verify_backup_occm.yaml
   ```

3. If the external storage is used as PV/PVC, then enter the ocnaddverify-xxxx container using the following commands:
   a. `kubectl exec -it <ocnaddverify-xxxx> -n <ocnadd namespace> -- bash`
   b. Change the directory to `/work-dir/backup` and inside the latest backup file `OCNADD_BACKUP_DD-MM-YYYY_hh-mm-ss.tar.bz2`, verify the DB backup and Kafka metadata backup files.

8.4.3 Retrieving the OCNADD Backup Files

1. Run the Verifying OCNADD Backup procedure to spawn the `ocnaddverify-xxxx`.
2. Go to the running `ocnaddverify` pod to identify and retrieve the desired backup folder using the following commands:
   a. Run the following command to access the pod:

      ```
      kubectl exec -it <ocnaddverify-xxxx> -n <ocandd-namespace> -- bash
      ```

      where,

      `<ocnadd-namespace>` is the namespace where the ocnadd management group services are running.

      `<ocnaddverify-xxxx>` is the backup verification pod in the same namespace.

b. Change the directory to `/work-dir/backup` and identify the backup file "`OCNADD_BACKUP_DD-MM-YYYY_hh-mm-ss.tar.bz2`".

c. Exit the `ocnaddverify` pod.

3. Copy the backup file from the pod to the local bastion server by copying the file `OCNADD_BACKUP_DD-MM-YYYY_hh-mm-ss.tar.bz2`, and run the following command:

```
kubectl cp -n <ocnadd-namespace> <ocnaddverify-xxxx>:/work-dir/backup/
<OCNADD_Backup_DD-MM-YYYY_hh-mm-ss.tar.bz2> <OCNADD_Backup_DD-MM-YYYY_hh-
mm-ss.tar.bz2>
```

where,

`<ocnadd-namespace>` is the namespace where the ocnadd management group services are running.

`<ocnaddverify-xxxx>` is the backup verification pod in the same namespace.

For example:

```
kubectl cp -n ocnadd ocnaddverify-drwzq:/work-dir/backup/
OCNADD_BACKUP_10-05-2023_08-00-05.tar.bz2
OCNADD_BACKUP_10-05-2023_08-00-05.tar.bz2
```

8.4.4 Copying and Restoring the OCNADD backup

1. Retrieve the OCNADD backup file.
2. Perform the Verifying OCNADD Backup procedure to spawn the `ocnaddverify-xxxx`.
3. Copy the backup file from the local bastion server to the running ocnaddverify pod, run the following command:

```
kubectl cp <OCNADD_Backup_DD-MM-YYYY_hh-mm-ss.tar.bz2> <ocnaddverify-
xxxx>:/work-dir/backup/<OCNADD_Backup_DD-MM-YYYY_hh-mm-ss.tar.bz2> -n
<ocnadd-namespace>
```

For example:

```
kubectl cp OCNADD_BACKUP_10-05-2023_08-00-05.tar.bz2 ocnaddverify-mrdxn:/
work-dir/backup/OCNADD_BACKUP_10-05-2023_08-00-05.tar.bz2 -n ocnadd
```

4. Go to `ocnaddverify` pod and path, `/workdir/backup/OCNADD_BACKUP_DD-MM-YYYY_hh-mm-ss.tar.bz2` to verify if the backup has been copied.
5. Restore OCNADD using the procedure defined in Creating OCNADD Restore Job.
6. Restart the `ocnaddalarm`, `ocnaddhealthmonitoring`, and `ocnaddadminsvc` pods.
7. Restart the `ocnaddconfiguration` after the ocnaddadmin service has been restarted completely.

8.5 Disaster Recovery Scenarios

This chapter describes the disaster recovery procedures for different recovery scenarios.

8.5.1 Scenario 1: Deployment Failure

This section describes how to recover OCNADD when the OCNADD deployment corrupts.

For more information, see Restoring OCNADD.

8.5.2 Scenario 2: cnDBTier Corruption

This section describes how to recover the cnDBTier corruption. For more information, see *Oracle Communications Cloud Native Core cnDBTier Installation, Upgrade, and Fault Recovery Guide*. After the cnDBTier recovery, restore the OCNADD database from the previous backup.

To restore the OCNADD database, execute the procedure Creating OCNADD Restore Job by setting BACKUP_ARG to DB.

8.5.3 Scenario 3: Database Corruption

This section describes how to recover from the corrupted OCNADD database.

Perform the following steps to recover the OCNADD configuration database (DB) from the corrupted database:

1. Retain the working ocnadd backup by following Retrieving the OCNADD Backup Files procedure.
2. Drop the existing Databases by accessing the MySql DB.
3. Perform the Copying and Restoring the OCNADD backup procedure to restore the backup.

8.5.4 Scenario 4: Site Failure

This section describes how to perform fault recovery when the OCNADD site has software failure.

Perform the following steps in case of a complete site failure:

1. Run the Cloud Native Environment (CNE) installation procedure to install a new Kubernetes cluster. For more information, see *Oracle Communications Cloud Native Environment Installation, Upgrade, and Fault Recovery Guide*.
2. Run the cnDBTier installation procedure. For more information, see *Oracle Communications Cloud Native Core cnDBTier Installation, Upgrade, and Fault Recovery Guide*.
3. For cnDBTier fault recovery, take a data backup from an older site and restore it to a new site. For more information about cnDBTier backup, see "Create On-demand Database Backup" and to restore the database to a new site, see "Restore DB with Backup" in *Oracle Communications Cloud Native Core cnDBTier Installation, Upgrade, and Fault Recovery Guide*.
4. Restore OCNADD. For more information, see Restoring OCNADD.

8.6 Restoring OCNADD

Perform this procedure to restore OCNADD when a fault event has occurred or deployment is corrupted.

> **Note**
>
> This procedure expects the OCNADD backup folder is retained.

1. Get the retained backup file "`OCNADD_BACKUP_DD-MM-YYYY_hh-mm-ss.tar.bz2`".
2. Get the Helm charts that was used in the earlier deployment.
3. Run the following command to uninstall the corrupted OCNADD deployment: Management Group or any Worker Group:

   ```
   helm uninstall <release_name> --namespace <namespace>
   ```

   Where,

   `<release_name>` is the release name of the ocnadd deployment which is being uninstalled.

   `<namespace>` is the namespace of OCNADD deployment which is being uninstalled.

   For example: To uninstall the Management Group

   ```
   helm  uninstall ocnadd-mgmt --namespace dd-mgmt-group
   ```

4. Install the Management Group or any Worker Group that was corrupted and uninstalled in the previous step using the helm charts that were used in the earlier deployment. For the installation procedure see, Installing OCNADD.
5. To verify whether OCNADD installation is complete, see Verifying OCNADD Installation.
6. Follow procedure Copying and Restoring the OCNADD backup

8.7 Creating OCNADD Restore Job

Note

If you have used OCCM to create certificates, then use the `ocnadd_restore_occm.yaml` file. Here is the file template:

```
# ocnadd_restore_occm.yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: ocnaddrestore
  namespace: ocnadd-deploy
spec:
  template:
    metadata:
      name: ocnaddrestore
      labels:
        role: backup
    spec:
      automountServiceAccountToken: false
      volumes:
      - name: backup-vol
        persistentVolumeClaim:
            claimName: backup-mysql-pvc
      - name: config-vol
        configMap:
          name: config-backuprestore-scripts
      - name: client-server-certificate-client
        secret:
          secretName: ocnaddbackuprestore-secret-client
      - name: client-server-certificate-server
        secret:
          secretName: ocnaddbackuprestore-secret-server
      - name: client-server-certificate-ca
        secret:
          secretName: occm-ca-secret
      - name: truststore-keystore-volume
        emptyDir: {}
      serviceAccountName: ocnadd-deploy-sa-ocnadd
      securityContext:
        runAsUser: 1000
        runAsGroup: 1000
        fsGroup: 1000
        runAsNonRoot: true
        seccompProfile:
          type: RuntimeDefault
      containers:
      - name: ocnaddrestore
        image: ocdd-docker.dockerhub-phx.oci.oraclecorp.com/ocdd.repo/
ocnaddbackuprestore:2.0.10
        securityContext:
          allowPrivilegeEscalation: false
```

```
            capabilities:
              drop: ["ALL"]
          volumeMounts:
            - mountPath: "work-dir"
              name: backup-vol
            - mountPath: "config-backuprestore-scripts"
              name: config-vol
            - name: client-server-certificate-client
              mountPath: /var/securityfiles/certs
              readOnly: true
            - name: truststore-keystore-volume
              mountPath: /var/securityfiles/keystore
          env:
            - name: HOME
              value: /home/ocnadd/
            - name: KS_PASS
              valueFrom:
                secretKeyRef:
                  name: occm-truststore-keystore-secret
                  key: keystorekey
            - name: TS_PASS
              valueFrom:
                secretKeyRef:
                  name: occm-truststore-keystore-secret
                  key: truststorekey
            - name: DB_USER
              valueFrom:
                secretKeyRef:
                  name: db-secret
                  key: MYSQL_USER
            - name: DB_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: db-secret
                  key: MYSQL_PASSWORD
            - name: BACKUP_ARG
              value: ALL
            - name: BACKUP_FILE
              value: ""
          command:
          - /bin/sh
          - -c
          - |
            cp /config-backuprestore-scripts/*.sh /home/ocnadd/
            chmod +x /home/ocnadd/*.sh
            cp /config-backuprestore-scripts/command.properties /home/
ocnadd
            chmod 660 /home/ocnadd/command.properties
            sed -i "s*\$KS_PASS*$KS_PASS*" /home/ocnadd/
command.properties
            sed -i "s*\$TS_PASS*$TS_PASS*" /home/ocnadd/
command.properties
            echo "Executing restore script"
            ls -lh /work-dir/backup
            bash /home/ocnadd/restore.sh $BACKUP_ARG $BACKUP_FILE
            sleep 5m
```

```
      initContainers:
      - name: ocnaddinitcontainer
        image: ocdd-docker.dockerhub-phx.oci.oraclecorp.com/utils.repo/
jdk21-openssl:1.1.0
        securityContext:
          allowPrivilegeEscalation: false
          readOnlyRootFilesystem: true
          capabilities:
            drop: ["ALL"]
        env:
             - name: SERVER_CERT_FILE
               value: /var/securityfiles/certs/server/
ocnaddbackuprestore-servercert.pem
             - name: CLIENT_CERT_FILE
               value: /var/securityfiles/certs/client/
ocnaddbackuprestore-clientcert.pem
             - name: SERVER_KEY_FILE
               value: /var/securityfiles/certs/server/
ocnaddbackuprestore-serverprivatekey.pem
             - name: CLIENT_KEY_FILE
               value: /var/securityfiles/certs/client/
ocnaddbackuprestore-clientprivatekey.pem
             - name: SERVER_KEY_STORE
               value: /var/securityfiles/keystore/serverKeyStore.p12
             - name: CLIENT_KEY_STORE
               value: /var/securityfiles/keystore/clientKeyStore.p12
             - name: TRUST_STORE
               value: /var/securityfiles/keystore/trustStore.p12
             - name: CA_CERT_FILE
               value: /var/securityfiles/certs/ca/cacert.pem
             - name: KS_PASS
               valueFrom:
                 secretKeyRef:
                   name: occm-truststore-keystore-secret
                   key: keystorekey
             - name: TS_PASS
               valueFrom:
                 secretKeyRef:
                   name: occm-truststore-keystore-secret
                   key: truststorekey
        command: ['/bin/sh']
        args: ['-c', "openssl pkcs12 -export -inkey $CLIENT_KEY_FILE -
in $CLIENT_CERT_FILE -out $CLIENT_KEY_STORE -password pass:$KS_PASS &&
keytool -importcert -file $CA_CERT_FILE -alias ocnaddcacert -
keystore $TRUST_STORE -storetype PKCS12 -storepass $TS_PASS -
noprompt;"]
        volumeMounts:
          - name: truststore-keystore-volume
            mountPath: /var/securityfiles/keystore
          - name: client-server-certificate-client
            mountPath: /var/securityfiles/certs/client
          - name: client-server-certificate-server
            mountPath: /var/securityfiles/certs/server
          - name: client-server-certificate-ca
```

```
            mountPath: /var/securityfiles/certs/ca
      restartPolicy: Never
```

Follow the below steps to create and run OCNADD restore job:

1. Restore the OCNADD database by following below steps:
   a. Go to the `custom-templates` folder inside the extracted `ocnadd-release` package and update the `ocnadd_restore.yaml` or the `ocnadd_restore_occm.yaml` file based on the restore requirements:
      i. The value of BACKUP_ARG can be set to DB, KAFKA, and ALL. By default, the value is 'ALL'.
      ii. The value of BACKUP_FILE can be set to folder name which needs to be restored, if not mentioned the latest backup will be used.
      iii. Update other values as below:

```
apiVersion: batch/v1
kind: Job
metadata:
  name: ocnaddrestore
  namespace: ocnadd-deploy            #---> update the namespace
------------
spec:
      serviceAccountName: ocnadd-deploy-sa-ocnadd           #--->
update the service account name. Format:<serviceAccount>-sa-ocnadd
------------
      containers:
      - name: ocnaddrestore
        image: <repo-path>/ocdd.repo/ocnaddbackuprestore:2.0.10
#---> update repository path
------------
      initContainers:
      - name: ocnaddinitcontainer
        image: <repo-path>/utils.repo/jdk21-openssl:1.1.0    #--->
update repository path
        env:
          - name: BACKUP_ARG
            value: ALL
          - name: BACKUP_FILE
            value: ""                         #---> update the backup
file name which needs to be restored, if not mentioned the latest
backup will be used for example "OCNADD_Backup_DD-MM-YYYY_hh-mm-
ss.tar.bz2"
```

2. Run the following command to run the restore job:

```
kubectl create -f ocnadd_restore.yaml
```

Or, use the following command if OCCM is used:

```
kubectl create -f ocnadd_restore_occm.yaml
```

> **Note**
>
> Make sure to delete all the backup, restore, and verify jobs before creating the restore job. Related jobs are `ocnaddbackup`, `ocnaddrestore`, `ocnaddverify`, and `ocnaddmanualbackup`.

3. Wait for the restore job to be completed. It usually takes 10 to 15 minutes or more depending upon the size of the backup.
4. Restart the below services in the provided order:
   a. `ocnaddhealthmonitoring`
   b. `ocnaddalarm`
   c. `ocnaddconfiguration`
5. Restart the `ocnaddadmin` in all the available worker groups, after the step 4 has been completed.
6. Restart `ocnaddfilter` service for all the worker group after the restore job is completed.
7. To restart the Redundancy Agent pods post OCNADD Restore, see Two-Site Redundancy Fault Recovery.
8. Perform the rollout restart for the deployments in management group and all the available worker groups:

```
kubectl rollout restart deployment -n <mgmt-grp-namespace>

kubectl rollout restart deployment -n <workergroup1-ns, workergroup2-ns>
```

> **Note**
>
> If the backup is not available for the mentioned date, the pod will be in an error state, notifying the backup is not available for the given date: $DATE. In such case, provide the correct backup dates and repeat the procedure.

8.8 Configuring Backup and Restore Parameters

To configure backup and restore parameters, configure the parameters listed in the following table:

> **Note**
>
> For information about backup and restore procedure, see Backup and Restore Flow section.

Table 8-2 Backup and Restore Parameters

Parameter Name	Data Type	Range	Default Value	Mandatory(M)/ Optional(O)/ Conditional(C)	Description
BACKUP_STORAGE	STRING	-	20Gi	M	Persistent Volume storage to keep the OCNADD backups
MYSQLDB_NAMESPACE	STRING	-	occne-cndbtierone	M	Mysql Cluster Namespace
BACKUP_CRONEXPRESSION	STRING	-	0 8 * * *	M	Cron expression to schedule backup cronjob
BACKUP_ARG	STRING	-	ALL	M	KAFKA, DB, or ALL backup
BACKUP_FILE	STRING	-	-	O	Backup folder name which needs to be restored
BACKUP_DATABASES	STRING	-	ALL	M	Individual databases or all databases backup that need to be taken

8.9 Two-Site Redundancy Fault Recovery

This section describes how to perform fault recovery of the OCNADD sites with Two-Site Redundancy enabled.

Scenario 1: When DB backup is available for both sites

1. Follow the generic recovery procedure based on the failure scenarios described in the section "Fault Recovery."
2. Use the respective site's backup during the restore procedure.
3. Once the recovery is completed, restart the Redundancy Agent pods of the Primary site and the Secondary site.

Scenario 2: When DB backup is not available on one of the mated sites

1. Access any one of the pods of the working site and run the below curl command to delete Redundancy Configuration:
   - `kubectl exec -it n <namespace> <pod> -- bash`
     For example:

     ```
     kubectl exec -it -n ocnadd-deploy ocnaddadmin-xxxxxx -- bash
     ```

   - ```
     curl -k --cert-type P12 --cert /var/securityfiles/keystore/
     serverKeyStore.p12:$OCNADD_SERVER_KS_PASSWORD --location -X DELETE
     ```
     ```

```
'https://ocnaddconfiguration:12590/ocnadd-configuration/v1/tsr-configure/
<workergroup name>?sync=false'
```

Where,

`<worker-group-name>` is the namespace of the worker group with the cluster. For example, `ocnadd-wg1:cluster-name`

2. Follow the generic recovery procedure based on the failure scenarios described in the section Disaster Recovery Scenarios.
3. Once the recovery is completed, restart the Redundancy Agent pods first on the Primary site, then on the Secondary site.
4. Re-create the Redundancy Configuration from the Primary UI.

> **Note**
>
> If the DB was lost on the Primary site and the user wants the Secondary site configuration to be restored on the Primary site, then set the **Way** to **Bidirectional** while creating the **Redundancy Configuration**.