Oracle® Communications

Network Repository Function (NRF) Cloud Native Installation and Upgrade Guide

Release 1.7.2

F32782-02

August 2020

ORACLE®

Oracle Communications Network Repository Function (NRF) Cloud Native Installation and Upgrade Guide, Release 1.7.2

F32782-02

Contents

6 Troubleshooting OCNRF

List of Figures

List of Tables

My Oracle Support

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:

1. Select **2** for New Service Request.

2. Select **3** for Hardware, Networking and Solaris Operating System Support.

3. Select one of the following options:

   - 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**.

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.

What's New in This Guide

This section introduces the documentation updates for Release 1.7.x in Oracle Communications Cloud Native Network Repository Function (NRF) Installation Guide.

Release 1.7.2

For Release 1.7.2, the following changes are performed in this document:

- `ingress-gateway.serviceMeshCheck` parameter is added under OCNRF Configuration Parameters.

- Aspen Service Mesh parameters are updated for all microservices in OCNRF Configuration Parameters.

- HookJob parameters are updated under global section in OCNRF Configuration Parameters.

- Aspen Service Mesh specific configuration procedure is updated in OCNRF Configuration.

Release 1.7.0

For Release 1.7.0, the following changes are performed in this document:

- Updated the pre-configuration procedures OCNRF pre-deployment configuration.

- Added custom labels and annotation to global level and per microservice level parameters. Refer to OCNRF Configuration Parameters for more information.

- Support for adding Prefix/Suffix to container names

- Moved `endpoint` and `endpointport` configuration from helm to rest based parameter.

1
OCNRF Overview

This section includes information about the role of Oracle Communications Network Repository Function (OCNRF) in 5G Service Based Architecture.

The OCNRF is one of the main components of the 5G Service Based Architecture. The OCNRF maintains an updated repository of all the Network Functions (NFs) available in the operator's network along with the services provided by each of the NFs in the 5G core that are expected to be instantiated, scaled and terminated with minimal or no manual intervention.

The OCNRF supports discovery mechanisms that allow NFs to discover each other and get updated status of the desired NFs.

The OCNRF supports the following functions:

- Maintains the profiles of the available NF instances and their supported services in the 5G core network.
- Allows consumer NF instances to discover other provider's NF instances in the 5G core network.
- Allows NF instances to track the status of other NF instances.
- Provides Oauth2 based Access Token service for consumer NF authorization.
- Provides specific NF Type selection based on subscriber identity.
- Supports forwarding of messages from one NRF to another NRF.
- Supports geo-redundancy to ensure service availability.

The OCNRF interacts with every other Network Function in the 5G core network and it supports the above functions through the following services:

- Management Services
- Discovery Services
- AccessToken Service

References

- Cloud Native Environment 1.5 Installation Document
- Network Repository Function (NRF) User's Guide
- CNC Console User's Guide
- ATS User Manual

Acronyms

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

Table 1-1   Acronyms

Field	Description
5G System	3GPP system consisting of 5G Access Network (AN), 5G Core Network and UE
5G-AN	5G Access Network
5GC	5G Core Network
5G-NF	5G Network Function
AMF	Access and Mobility Management Function
API-Gateway	Application Program Interface Gateway
CNE	Cloud Native Environment
FQDN	Fully Qualified Domain Name
K8s	Kubernetes
MMI	Machine Machine Interface
MPS	Messages Per Second
NDB	Network Database
NF	Network Function
Network Function	A functional building block within a network infrastructure, which has well defined external interfaces and well defined functional behavior. In practical terms, a network function is often a network node or physical appliance.
Network Slice	A logical network that provides specific network capabilities and network characteristics.
Network Slice instance	A set of Network Function instances and the required resources (e.g. compute, storage and networking resources) which form a deployed Network Slice.
NF Consumer	A generic way to refer to an NF which consumes services provided by another NF. Ex: An AMF is referred to as a Consumer when it consumes AMPolicy services provided by the PCF.
NF Instance	A specific instance of a network function type.
NF Producer or NF Provider	A generic way to refer to an NF which provides services that can be consumed by another NF. Ex: A PCF is a provider NF and provides AMPolicy Services
NRF	Network Repository Function
OCNRF	Oracle Communications Network Repository Function
OHC	Oracle Help Center
OSDC	Oracle Software Download Center
PLMN	Public Land Mobile Network
Resiliency	The ability of the NFV framework to limit disruption and return to normal or at a minimum acceptable service delivery level in the fame of a fault, failure, or an event that disrupts normal operation.
Scaling	Ability to dynamically extend/reduce resources granted to the Virtual Network Function (VNF) as needed. This includes scaling out/in or scaling up/down.
Scaling Out/In/ Horizontally	The ability to scale by add/remove resource instances (e.g. VMs). Also called scaling Horizontally.
Scaling Up/Down/ Vertically	The ability to scale by changing allocated resources, e.g. increase/decrease memory, CPU capacity or storage size.
PCF	Policy Control Function

Table 1-1   (Cont.) Acronyms

Field	Description
SEPP	Security Edge Protection Proxy
SCP	Service Communication Proxy
SLF	Subscriber Location Function
URI	Universal Resource Identifier

2
Installing OCNRF

This section describes the prerequisites and installation procedure for OCNRF.

Prerequisites

Following are the prerequisites to install and configure OCNRF:

OCNRF Software

The OCNRF software includes:

- OCNRF Helm charts
- OCNRF docker images

The following software must be installed before installing OCNRF:

Software	Version
Kubernetes	v1.15.3
HELM	v2.14.3 and v3.2

Additional software that needs to be deployed as per the requirement of the services:

Software	Chart Version	Required For
elasticsearch	5.5.4	Logging Area
elastic-curator	5.5.4	Logging Area
elastic-exporter	1.0.2	Logging Area
logs	2.0.7	Logging Area
kibana	6.7.0	Logging Area
grafana	6.1.6	Metrics Area
prometheus	9.1.2	Metrics Area
prometheus-node-exporter	0.17.0	Metrics Area
metallb	0.7.3	External IP
metrics-server	0.3.1	Metric Server
tracer	0.8.3	Tracing Area

> **Note:**
>
> Install the specified software items before proceeding, if any of the above services are needed and the respective software is not already installed in CNE.
> To check the installed software items, execute:
>
> ```
> helm ls
> ```
>
> Some of the systems may need to use helm command with `admin.conf` file, such as:
>
> ```
> helm --kubeconfig admin.conf
> ```

Network access

The Kubernetes cluster hosts must have network access to:

- Local docker image repository where the OCNRF images are available.
  To check if the Kubernetes cluster hosts has network access to the local docker image repository, try to pull any image with tag name to check connectivity by executing:

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

  > **Note:**
  >
  > Some of the systems may need to use helm command with `admin.conf` file, such as:
  >
  > ```
  > helm --kubeconfig admin.conf
  > ```

- Local helm repository where the OCNRF helm charts are available.
  To check if the Kubernetes cluster hosts has network access to the local helm repository, execute:

  ```
  helm repo update
  ```

  > **Note:**
  >
  > Some of the systems may need to use helm command with `admin.conf` file, such as:
  >
  > ```
  > helm --kubeconfig admin.conf
  > ```

> **Note:**
>
> All the kubectl and helm related commands that are used in this document must be executed on a system depending on the infrastructure of the deployment. It could be a client machine such as a VM, server, local desktop, and so on.

Client machine requirement

Client machine needs to have the following minimum requirements:

- Network access to the helm repository and docker image repository.
- Helm repository must be configured on the client.
- Network access to the Kubernetes cluster.
- Necessary environment settings to run the `kubectl` commands. The environment should have privileges to create a namespace in the Kubernetes cluster.
- Helm client must be installed. The environment should be configured so that the `helm install` command deploys the software in the Kubernetes cluster.

Server or Space Requirements

For information on the server or space requirements, see the Oracle Communications Cloud Native Environment (OCCNE) Installation Guide.

Secret file requirement

For HTTPs and Access token, the following certs and pem files has to be created before creating secret files for Keys and MySql.

Note: The following files must be created before creating secret files.

1. ECDSA private Key and CA signed ECDSA Certificate (if initialAlgorithm: ES256)
2. RSA private key and CA signed RSA Certificate (if initialAlgorithm: RS256)
3. TrustStore password file
4. KeyStore password file
5. CA signed ECDSA certificate

ServiceAccount requirement

Operator must create a service account, bind it with a Role for resource with permissions for atleast get, watch and list.

serviceAccountName is a mandatory parameter. Kubernetes Secret resource is used for providing the following:

- MYSQL DB Details to micro-services.
- NRF's Private Key, NRF's Certificate and CA Certificate Details to Ingress/Egress Gateway for TLS.
- NRF's Private and NRF's Public Keys to nfAccessToken micro-service for Digitally Signing AccessTokenClaims.

- Producer/Consumer NF's Service/Endpoint details for routing messages from/to Egress/Ingress Gateway.

The Secret(s) can be under same namespace where OCNRF is getting deployed (recommended) or # Operator can choose to use different namespaces for different secret(s). If all the Secret(s) are under same namespace as OCNRF, then Kubernetes Role can be binded with the given ServiceAccount. Otherwise ClusterRole needs to be binded with the given ServiceAccount. The Role/ClusterRole needs to be created with resources: (services, configmaps, pods, secrets, endpoints) and (verbs: get, watch, list). Refer to Creating Service Account, Role and Role bindings for more details.

DB Tier Requirement

DB Tier must be up and running. In case of geo-redundant deployments, replication between geo-redundant DB Tier must be configured. Refer to DB Tier section in OCCNE installation guide.

Installation Sequence

This section explains the tasks to be performed for installing OCNRF.

OCNRF pre-deployment configuration

Following are the pre-deployment configuration procedures:

1. Creating OCNRF namespace

> **Note:**
>
> This is a mandatory procedure, execute this before proceeding any further. The namespace created/verified in this procedure is an input for next procedures.

2. Creating Service Account, Role and Role bindings

> **Note:**
>
> Following are sample steps, in case of already configured service account with role and role-bindings or the user has previously prepared procedure to create service account, skip this procedure.

3. Configuring MySql database and user
4. Configuring Kubernetes Secret for Accessing OCNRF Database
5. Configuring secrets for enabling HTTPS
6. Configuring Secret for Enabling AccessToken Service

Creating OCNRF namespace

This section explains how the user can verify if the required namespace is available in the system or not.

Procedure

1. Verify required namespace already exists in system:

```
$ kubectl get namespaces
```

2. In the output of the above command, check if required namespace is available. If not available, create the namespace using following command:

> **Note:**
>
> This is an optional step. In case required namespace already exists, skip this procedure.

```
$ kubectl create namespace <required namespace>
```

For example:-

```
$ kubectl create namespace ocnrf
```

Creating Service Account, Role and Role bindings

This section explains how user can create service account, required role and role bindings resources. The Secret(s) can be under same namespace where OCNRF is getting deployed (recommended) or operator can choose to use different namespaces for different secret(s). If all the Secret(s) are under same namespace as OCNRF, then Kubernetes Role can be binded with the given ServiceAccount. Otherwise ClusterRole needs to be binded with the given ServiceAccount.

Sample template for the resources is as follows and add sample template content to resource input yaml file.

Example file name: *ocnrf-resource-template.yaml*

Example command for creating the resources

```
kubectl -n <ocnrf-namespace> create -f ocnrf-resource-template.yaml
```

Sample template to create the resources

> **Note:**
>
> Update **<helm-release>** and **<namespace>** with respective OCNRF namespace and planned OCNRF helm release name in the place holders.

```
## Sample template start#
apiVersion: v1
kind: ServiceAccount
metadata:
```

```
    name: <helm-release>-ocnrf-serviceaccount
    namespace: <namespace>
---

apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: <helm-release>-ocnrf-role
  namespace: <namespace>
rules:
- apiGroups:
  - "" # "" indicates the core API group
  resources:
  - services
  - configmaps
  - pods
  - secrets
  - endpoints
  verbs:
  - get
  - watch
  - list
---

apiVersion: rbac.authorization.k8s.io/v1beta1
kind: RoleBinding
metadata:
  name: <helm-release>-ocnrf-rolebinding
  namespace: <namespace>
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: <helm-release>-ocnrf-role
subjects:
- kind: ServiceAccount
  name:  <helm-release>-ocnrf-serviceaccount
  namespace: <namespace>
## Sample template end#
```

Configuring MySql database and user

This section explains how database administrator can create the databases and users for OCNRF network function.

> **Note:**
>
> 1. Procedure can be different for geo-redundant OCNRF sites and standalone OCNRF site.
>
> 2. Before executing the below procedure for Geo-Redundant sites, ensure that the DB-Tier for Geo-Redundant sites are already up and replication channels are enabled.
>
> 3. While performing Fresh Installation, in case OCNRF release is already deployed, purge the deployment, remove databases, users used for previous deployment. Refer to Uninstalling OCNRF for uninstallation procedure.

Procedure for Geo-Redundant OCNRF sites

1. Login to the machine where ssh keys are stored and which has permission to access the SQL nodes of NDB cluster.

2. Connect to the SQL nodes.

3. Login to the MySQL prompt using root permission or user, which has permission to create users with conditions as mentioned below. For example: `mysql -h 127.0.0.1 -uroot -p`

   > **Note:**
   >
   > This command may vary from system to system, path for MySQL binary, root user and root password. After executing this command, user need to enter the password specific to the user mentioned in the command.

4. Check OCNRF network function database user already exists. If the user does not exists, create an OCNRF network function database user.
   Below steps covers the creation of two type of OCNRF database users. Different users has different set of permissions.

   a. OCNRF privileged user: This user has complete set of permissions. This user can perform create, alter, drop operations on tables to perform install/upgrade/rollback or delete operations.

   b. OCNRF application user: This user has less set of permissions and will be used by OCNRF application during service operations handling. This user can insert, update, get, remove the records. This user can't create, alter and drop the database as wells as tables

   ```
   $ SELECT User FROM mysql.user;
   ```

   In case, user already exists, move to next step. Else, create new following ocnrf user:

   - Create new ocnrf privileged user:
     ```
     $ CREATE USER '<OCNRF Privileged-User Name>'@'%' IDENTIFIED BY
     '<OCNRF Privileged-User Password>';
     ```

Example:

```
$ CREATE USER 'nrfPrivilegedUsr'@'%' IDENTIFIED BY
'nrfPrivilegedPasswd'
```

- Create new ocnrf application user:

```
$ CREATE USER '<OCNRF APPLICATION User Name>'@'%' IDENTIFIED BY
'<OCNRF APPLICATION User Password>';
```

Example:

```
$ CREATE USER 'nrfApplicationUsr'@'%' IDENTIFIED BY
'nrfApplicationPasswd'
```

> **Note:**
>
> Both users must be created on all the SQL Nodes on all the sites.

5. Check OCNRF network function database already exists. If the database does not exists, create databases for OCNRF network function:
Execute the following command to check if database exists:

```
$ show databases;
```

In case database already exists, then move to next step. Else, perform the following steps.

For OCNRF application, two type of databases are required:

a. OCNRF application database: This database consists of tables used by application to perform functionality of NRF network function.

b. OCNRF network database: This database consists of tables used by OCNRF to store per the network details like system details and database backups.

a. Create database for OCNRF application:

```
$ CREATE DATABASE IF NOT EXISTS <OCNRF Application Database>
CHARACTER SET utf8;
```

Example:

```
$ CREATE DATABASE IF NOT EXISTS nrfApplicationDB CHARACTER SET
utf8;
```

b. Create database for OCNRF network database:

```
$ CREATE DATABASE IF NOT EXISTS <OCNRF network database>
CHARACTER SET utf8;
```

Example:

```
$ CREATE DATABASE IF NOT EXISTS nrfNetworkDB CHARACTER SET utf8;
```

> **Note:**
>
> OCNRF application and network database must be created on any one of SQL node on any one of the OCNRF site.

c. Grant permission to users on the OCNRF database created:

> **Note:**
>
> This step must be executed on all the SQL nodes on all the OCNRF Geo-Redundant sites.

```
$ GRANT SELECT, INSERT, CREATE, ALTER, DROP, LOCK TABLES, CREATE
TEMPORARY TABLES, DELETE, UPDATE, EXECUTE ON <OCNRF Application
Database>.* TO '<OCNRF Privileged-User Name>'@'%';
```

Example:

```
$ GRANT SELECT, INSERT, CREATE, ALTER, DROP, LOCK TABLES, CREATE
TEMPORARY TABLES, DELETE, UPDATE, EXECUTE ON nrfApplicationDB.*
TO 'nrfPrivilegedUsr'@'%';
```

d. Grant permission to OCNRF privileged user on OCNRF network database:

```
$ GRANT SELECT, INSERT, CREATE, ALTER, DROP, LOCK TABLES, CREATE
TEMPORARY TABLES, DELETE, UPDATE, EXECUTE ON <OCNRF network
database>.* TO '<OCNRF Privileged-User Name>'@'%';
```

Example:

```
$ GRANT SELECT, INSERT, CREATE, ALTER, DROP, LOCK TABLES, CREATE
TEMPORARY TABLES, DELETE, UPDATE, EXECUTE ON nrfNetworkDB.* TO
'nrfPrivilegedUsr'@'%';
```

e. Grant permission to OCNRF application user on OCNRF application database:

```
$ GRANT SELECT, INSERT, LOCK TABLES, DELETE, UPDATE, EXECUTE
ON <OCNRF Application Database>.* TO '<OCNRF APPLICATION User
Name>'@'%';
```

Example:

```
$ GRANT SELECT, INSERT, LOCK TABLES, DELETE, UPDATE, EXECUTE ON
nrfApplicationDB.* TO 'nrfApplicationUsr'@'%';
```

ORACLE

    f.   Grant read permission to OCNRF application user for replication_info:

```
$ GRANT SELECT ON replication_info.* TO '<OCNRF APPLICATION User
Name>'@'%';
```

        Example:

```
$ GRANT SELECT ON replication_info.* TO 'nrfApplicationusr'@'%';
```

    g. Apply the grants using following command:

```
FLUSH PRIVILEGES;
```

6. Execute the command, `show grants for <username>`, to confirm that users has all of the required permissions

7. Exit from database and logout from MYSQL nodes.

Procedure for standalone OCNRF site

1. Login to the machine where ssh keys are stored and which has permission to access the SQL nodes of NDB cluster.

2. Connect to the SQL nodes.

3. Login to the MySQL prompt using root permission or user, which has permission to create users with conditions as mentioned below. For example: `mysql -h 127.0.0.1 -uroot -p`

> **✎ Note:**
>
> This command may vary from system to system, path for mysql binary, root user and root password. After executing this command, user need to enter the password specific to the user mentioned in the command.

4. Check OCNRF network function user already exists. If the NF does not exists, create an OCNRF network function user.
Below steps covers the creation of two type of OCNRF users. Different users has different set of permissions.

    a. OCNRF privileged user: This user has complete set of permissions. This user can perform create, alter, drop operations on tables to perform install/upgrade/rollback or delete operations.

    b. OCNRF application user: This user has less set of permissions and will be used by OCNRF application during service operations handling. This user can insert, update, get, remove the records. This user can't create, alter and drop the database as wells as tables.

```
$ SELECT User FROM mysql.user;
```

In case, user already exists, move to next step. Else, create new following OCNRF user:

- Create new OCNRF application user:

```
$ CREATE USER '<OCNRF APPLICATION User Name>'@'%' IDENTIFIED BY
'<OCNRF APPLICATION Password>';
```

Example:

```
$ CREATE USER 'nrfApplicationUsr'@'%' IDENTIFIED BY
'nrfApplicationPasswd'
```

- Create new OCNRF privileged user:

```
$ CREATE USER '<OCNRF Privileged-User Name>'@'%' IDENTIFIED BY
'<OCNRF Privileged-User Password>';
```

Example:

```
$ CREATE USER 'nrfPrivilegedUsr'@'%' IDENTIFIED BY
'nrfPrivilegedPasswd'
```

> **Note:**
>
> Both users must be created on all the SQL Nodes for all the sites.

5. Check OCNRF network function databases already exists. If not exists, create databases for OCNRF network function:
   Execute the following command to check if database exists:

```
$ show databases;
```

Check if required database is already in list. In case the database already exists, then move to next step. Else, perform the following steps.

For OCNRF application, two type of databases are required:

a. OCNRF application database: This database consists of tables used by application to perform functionality of NRF network function.

b. OCNRF network database: This database consists of tables used by OCNRF to store per OCNRF network details like system details and database backups.

a. Create database for OCNRF application:

```
$ CREATE DATABASE IF NOT EXISTS <OCNRF Application Database>
CHARACTER SET utf8;
```

Example:

```
$ CREATE DATABASE IF NOT EXISTS nrfApplicationDB CHARACTER SET
utf8;
```

   b. Create database for OCNRF network:

```
$ CREATE DATABASE IF NOT EXISTS <OCNRF network database>
CHARACTER SET utf8;
```

Example:

```
$ CREATE DATABASE IF NOT EXISTS nrfNetworkDB CHARACTER SET utf8;
```

6. Grant permissions to users on the databases:

> **✎ Note:**
>
> This step must be executed on all the SQL nodes on each OCNRF standalone site.

   a. Grant permission to OCNRF privileged user on OCNRF application database:

```
$ GRANT SELECT, INSERT, CREATE, ALTER, DROP, LOCK TABLES, CREATE
TEMPORARY TABLES, DELETE, UPDATE, EXECUTE ON <OCNRF Application
Database>.* TO '<OCNRF Privileged-User Name>'@'%';
```

Example:

```
$ GRANT SELECT, INSERT, CREATE, ALTER, DROP, LOCK TABLES, CREATE
TEMPORARY TABLES, DELETE, UPDATE, EXECUTE ON nrfApplicationDB.*
TO 'nrfPrivilegedUsr'@'%';
```

   b. Grant permission to OCNRF privileged user on OCNRF network database:

```
$ GRANT SELECT, INSERT, CREATE, ALTER, DROP, LOCK TABLES, CREATE
TEMPORARY TABLES, DELETE, UPDATE, EXECUTE ON <OCNRF network
database>.* TO '<OCNRF Privileged-User Name>'@'%';
```

Example:

```
$ GRANT SELECT, INSERT, CREATE, ALTER, DROP, LOCK TABLES, CREATE
TEMPORARY TABLES, DELETE, UPDATE, EXECUTE ON nrfNetworkDB.* TO
'nrfPrivilegedUsr'@'%';
```

   c. Grant permission to OCNRF application user on OCNRF application database:

```
$ GRANT SELECT, INSERT, LOCK TABLES, DELETE, UPDATE, EXECUTE
ON <OCNRF Application Database>.* TO '<OCNRF APPLICATION User
Name>'@'%';
```

Example:

```
$ GRANT SELECT, INSERT, LOCK TABLES, DELETE, UPDATE, EXECUTE ON
nrfApplicationDB.* TO 'nrfApplicationUsr'@'%';
```

d. Grant read permission to OCNRF application user for replication_info:

```
$ GRANT SELECT ON replication_info.* TO '<OCNRF APPLICATION User
Name>'@'%';
```

Example:

```
$ GRANT SELECT ON replication_info.* TO 'nrfApplicationusr'@'%';
```

7. Apply the grants using following command:

```
FLUSH PRIVILEGES;
```

8. Exit from MySQL prompt and SQL nodes.

Configuring Kubernetes Secret for Accessing OCNRF Database

This section explains the steps to configure kubernetes secrets for accessing the OCNRF database created in the above section. This procedure must be executed before deploying OCNRF.

Kubernetes Secret Creation for OCNRF Privileged Database User

This section explains the steps to create kubernetes secrets for accessing OCNRF database and privileged user details created by database administrator in above section. This section must be execute before deploying OCNRF.

Create kubernetes secret for privileged user as follows:

1. Create kubernetes secret for MySQL:

```
$ kubectl create secret generic <privileged user secret name> --
from-literal=dbUsername=<OCNRF Privileged Mysql database username>
--from-literal=dbPassword=<OCNRF Privileged Mysql User database
passsword> --from-literal=appDbName=<OCNRF Mysql database name>
--from-literal=networkScopedDbName=<OCNRF Mysql Network database
name> -n <Namespace of OCNRF deployment>
```

> **Note:**
>
> Note down the command used during the creation of kubernetes secret, this command is used for updates in future.

Example:

```
$ kubectl create secret generic
privilegeduser-secret --from-literal=dbUsername=nrfPrivilegedUsr
```

```
--from-literal=dbPassword=nrfPrivilegedPasswd --
from-literal=appDbName=nrfApplicationDb --from-
literal=networkScopedDbName=nrfNetworkDB -n ocnrf
```

2. Verify the secret created using above command:

```
$ kubectl describe secret <database secret name> -n <Namespace of
OCNRF deployment>
```

Example:

```
$ kubectl describe secret privilegeduser-secret -n ocnrf
```

Kubernetes Secret Update for OCNRF Privileged Database User

This section describes the steps to update the secrets. Update Kubernetes secret for privileged user as follows:

1. Copy the exact command used in Kubernetes Secret Creation for OCNRF Privileged Database User section during creation of secret:

```
$ kubectl create secret generic <privileged user secret
name> --from-literal=dbUsername=<OCNRF Privileged Mysql database
username> --from-literal=dbPassword=<OCNRF Privileged Mysql
database password> --from-literal=appDbName=<OCNRF Mysql database
name> --from-literal=networkScopedDbName=<OCNRF Mysql Network
database name> -n <Namespace of OCNRF deployment>
```

2. Update the same command with string "--dry-run -o yaml" and "kubectl replace -f -
   -n <Namespace of MYSQL secret>". After update, the command will be as follows:

```
$ kubectl create secret generic <privileged user secret
name> --from-literal=dbUsername=<OCNRF Privileged Mysql database
username> --from-literal=dbPassword=<OCNRF Privileged Mysql
database password> --from-literal=appDbName=<OCNRF Mysql database
name> --from-literal=networkScopedDbName=<OCNRF Mysql Network
database name> --dry-run -o yaml -n <Namespace of OCNRF deployment>
| kubectl replace -f - -n <Namespace of OCNRF deployment>
```

3. Execute the updated command. The following message is displayed:

```
secret/<database secret name> replaced
```

Kubernetes Secret Creation for OCNRF Application Database User

This section explains the steps to create secrets for accessing and configuring application database user created in above section. This section must be execute before deploying OCNRF.

Create kubernetes secret for OCNRF application database user for configuring records is as follows:

1. Create kubernetes secret for OCNRF application database user:

```
$ kubectl create secret generic <appuser-secret name> --
from-literal=dbUsername=<OCNRF APPLICATION User Name> --from-
literal=dbPassword=<Password for OCNRF APPLICATION User> --from-
literal=appDbName=<OCNRF Application Database> -n <Namespace of
OCNRF deployment>
```

> **Note:**
>
> Note down the command used during the creation of kubernetes secret,
> this command will be used for updates in future.

Example:

```
$ kubectl create secret generic
appuser-secret --from-literal=dbUsername=nrfApplicationUsr
--from-literal=dbPassword=nrfApplicationPasswd --from-
literal=appDbName=nrfApplicationDB -n ocnrf
```

2. Verify the secret creation:

```
$ kubectl describe secret <appuser-secret name> -n <Namespace of
OCNRF deployment>
```

Example:

```
$ kubectl describe secret appuser-secret -n ocnrf
```

Kubernetes Secret Update for OCNRF Application Database User

This section explains how to update the kubernetes secret.

1. Copy the exact command used in above section during creation of secret:

```
$ kubectl create secret generic <appuser-secret name> --
from-literal=dbUsername=<OCNRF APPLICATION User Name> --from-
literal=dbPassword=<Password for OCNRF APPLICATION User> --from-
literal=appDbName=<OCNRF Application Database> -n <Namespace of
OCNRF deployment>
```

2. Update the same command with string "--dry-run -o yaml" and "kubectl replace -f -
-n <Namespace of MYSQL secret>". After update, the command will be as follows:

```
$ kubectl create secret generic <database secret name>
--from-literal=dbUsername=<OCNRF APPLICATION User Name> --from-
literal=dbPassword=<Password for OCNRF APPLICATION User> --from-
literal=appDbName=<OCNRF Application Database> --dry-run -o yaml
-n <Namespace of OCNRF deployment> | kubectl replace -f - -n
<Namespace of OCNRF deployment>
```

3. Execute the updated command. The following message is displayed:

```
secret/<database secret name> replaced
```

Configuring secrets for enabling HTTPS

Creation of secrets for enabling HTTPS in OCNRF Ingress gateway

This section explains the steps to configure secrets for enabling HTTPS in ingress and egress gateways. This section must be executed before enabling HTTPS in OCNRF Ingress/Egress gateway.

> **Note:**
>
> The passwords for TrustStore and KeyStore are stored in respective password files mentioned below.

To create kubernetes secret for HTTPS, following files are required:

- ECDSA private key and CA signed certificate of OCNRF (if initialAlgorithm is ES256)
- RSA private key and CA signed certificate of OCNRF (if initialAlgorithm is RS256)
- TrustStore password file
- KeyStore password file
- CA certificate

> **Note:**
>
> Creation process for private keys, certificates and passwords is on discretion of user/operator.

1. Execute the following command to create secret:

```
$ kubectl create secret generic
<ocingress-secret-name> --from-file=<ssl_ecdsa_private_key.pem>
--from-file=<rsa_private_key_pkcs1.pem> --from-
file=<ssl_truststore.txt> --from-file=<ssl_keystore.txt> --from-
file=<caroot.cer> --from-file=<ssl_rsa_certificate.crt> --from-
file=<ssl_ecdsa_certificate.crt> -n <Namespace of OCNRF deployment>
```

> **Note:**
>
> Note down the command used during the creation of kubernetes secret, this command will be used for updates in future.

Example: The names used below are same as provided in custom_values.yaml in OCNRF deployment.

```
$ kubectl create secret generic
ocingress-secret --from-file=ssl_ecdsa_private_key.pem --from-
file=rsa_private_key_pkcs1.pem --from-file=ssl_truststore.txt
--from-file=ssl_keystore.txt --from-file=caroot.cer --from-
file=ssl_rsa_certificate.crt --from-file=ssl_ecdsa_certificate.crt -
n ocnrf
```

2. Verify the secret created using the following command:

```
$ kubectl describe secret <ocingress-secret-name> -n <Namespace of
OCNRF deployment>
```

Example:

```
$ kubectl describe secret ocingress-secret -n ocnrf
```

Update the secrets for enabling HTTPS in OCNRF Ingress gateway

This section explains how to update the secret with updated details.

1. Copy the exact command used in above section during creation of secret.
2. Update the same command with string "--dry-run -o yaml" and "kubectl replace -f -
    -n <Namespace of OCNRF deployment>".
3. Create secret command will look like:

```
$ kubectl create secret generic
<ocingress-secret-name> --from-file=<ssl_ecdsa_private_key.pem>
--from-file=<rsa_private_key_pkcs1.pem> --from-
file=<ssl_truststore.txt> --from-file=<ssl_keystore.txt> --from-
file=<caroot.cer> --from-file=<ssl_rsa_certificate.crt> --from-
file=<ssl_ecdsa_certificate.crt> --dry-run -o yaml -n <Namespace of
OCNRF deployment> | kubectl replace -f - -n <Namespace of OCNRF
deployment>
```

Example:-

The names used below are same as provided in custom_values.yaml in OCNRF deployment:

```
$ kubectl create secret generic
ocingress-secret --from-file=ssl_ecdsa_private_key.pem --from-
file=rsa_private_key_pkcs1.pem --from-file=ssl_truststore.txt
--from-file=ssl_keystore.txt --from-file=caroot.cer --from-
file=ssl_rsa_certificate.crt --from-file=ssl_ecdsa_certificate.crt
--dry-run -o yaml -n ocnrf | kubectl replace -f - -n ocnrf
```

4. Execute the updated command.

5. After successful secret update, the following message is displayed:

```
secret/<ocingress-secret> replaced
```

Creation of secrets for enabling HTTPS in OCNRF Egress gateway

This section explains the steps to create secret for HTTPS related details. This section must be executed before enabling HTTPS in OCNRF Egress gateway.

> **Note:**
>
> The passwords for TrustStore and KeyStore are stored in respective password files mentioned below.

To create kubernetes secret for HTTPS, following files are required:

- ECDSA private key and CA signed certificate of OCNRF (if initialAlgorithm is ES256)
- RSA private key and CA signed certificate of OCNRF (if initialAlgorithm is RS256)
- TrustStore password file
- KeyStore password file
- CA certificate

> **Note:**
>
> Creation process for private keys, certificates and passwords is on discretion of user/operator.

1. Execute the following command to create secret.

```
$ kubectl create secret generic <ocegress-
secret-name> --from-file=<ssl_ecdsa_private_key.pem>  --from-
file=<ssl_rsa_private_key.pem> --from-file=<ssl_truststore.txt>
--from-file=<ssl_keystore.txt> --from-file=<ssl_cabundle.crt>
--from-file=<ssl_rsa_certificate.crt> --from-
file=<ssl_ecdsa_certificate.crt> -n <Namespace of OCNRF deployment>
```

> **Note:**
>
> Note down the command used during the creation of kubernetes secret, this command will be used for updates in future.

Example: The names used below are same as provided in custom_values.yaml in OCNRF deployment.

```
$ kubectl create secret generic ocegress-secret --from-
file=ssl_ecdsa_private_key.pem --from-file=ssl_rsa_private_key.pem
--from-file=ssl_truststore.txt --from-file=ssl_keystore.txt --from-
file=ssl_cabundle.crt --from-file=ssl_rsa_certificate.crt --from-
file=ssl_ecdsa_certificate.crt -n ocnrf
```

2. Command to verify secret created:

```
$ kubectl describe secret <ocegress-secret-name> -n <Namespace of
OCNRF deployment>
```

Example:

```
$ kubectl describe secret ocegress-secret -n ocnrf
```

Update the secrets for enabling HTTPS in OCNRF Egress gateway

This section explains how to update the secret with updated details.

1. Copy the exact command used in above section during creation of secret:

2. Update the same command with string "--dry-run -o yaml" and "kubectl replace -f -
   -n <Namespace of OCNRF deployment>".

3. Create secret command will look like:

```
kubectl create secret generic <ocegress-
secret-name> --from-file=<ssl_ecdsa_private_key.pem> --from-
file=<ssl_rsa_private_key.pem> --from-file=<ssl_truststore.txt>
--from-file=<ssl_keystore.txt> --from-file=<ssl_cabundle.crt>
--from-file=<ssl_rsa_certificate.crt> --from-
file=<ssl_ecdsa_certificate.crt> --dry-run -o yaml -n <Namespace of
OCNRF Egress Gateway secret> | kubectl replace -f - -n <Namespace
of OCNRF deployment>
```

Example:

The names used below are same as provided in custom_values.yaml in OCNRF deployment:

```
$ kubectl create secret generic
egress-secret --from-file=ssl_ecdsa_private_key.pem --from-
file=rsa_private_key_pkcs1.pem --from-file=ssl_truststore.txt
--from-file=ssl_keystore.txt --from-file=caroot.cer --from-
file=ssl_rsa_certificate.crt --from-file=ssl_ecdsa_certificate.crt
--dry-run -o yaml -n ocnrf | kubectl replace -f - -n ocnrf
```

4. Execute the updated command.

5. After successful secret update, the following message is displayed:

```
secret/<ocegress-secret> replaced
```

Configuring Secret for Enabling AccessToken Service

Access Token secret creation

This section explains the steps to create secret for AccessToken service of OCNRF. This section must be executed before enabling Access Token in OCNRF.

> **Note:**
>
> The password for KeyStore is stored in respective password file mentioned below.

To create kubernetes secret for AccessToken, following files are required:

- ECDSA private key and CA signed certificate of OCNRF (if initialAlgorithm is ES256)
- RSA private key and CA signed certificate of OCNRF (if initialAlgorithm is RS256)
- KeyStore password file: This file contains a password which is used to protect the PrivateKeys/Certificates that will get loaded into the application in-memory (KeyStore).
  For example:`echo qwerpoiu > keystore_password.txt`

  where `qwerpoiu` is the password and `keystore_password.txt` is the target file which is provided as input to the AccessToken secret.

> **Note:**
>
> Creation process for private keys, certificates and passwords is on discretion of user/operator.

1. Execute the following command to create secret. The names used below are same as provided in custom values.yaml in OCNRF deployment:

```
kubectl create secret generic <ocnrfaccesstoken-secret> --from-
file=<ecdsa_private_key.pem> --from-file=<rsa_private_key.pem> --
from-file=<keystore_password.txt> --from-file=<rsa_certificate.crt>
--from-file=<ecdsa_certificate.crt> -n <Namespace of OCNRF
deployment>
```

> **Note:**
>
> Note down the command used during the creation of kubernetes secret, this command will be used for updates in future.

Example:

```
$ kubectl create secret generic ocnrfaccesstoken-secret --from-
file=ecdsa_private_key.pem --from-file=rsa_private_key.pem --from-
file=keystore_password.txt --from-file=rsa_certificate.crt --from-
file=ecdsa_certificate.crt -n ocnrf
```

2. Execute the following command to verify secret created:

```
$ kubectl describe secret <ocnrfaccesstoken-secret-name> -n
<Namespace of OCNRF deployment>
```

Example:

```
 $ kubectl describe secret ocnrfaccesstoken-secret -n ocnrf
```

Access Token secret update

This section explains how to update the access token secret with updated details.

1. Copy the exact command used in above section during creation of secret.

2. Update the same command with string "--dry-run -o yaml" and "kubectl replace -f -
   -n <Namespace of OCNRF deployment>".

3. Create secret command will look like:

```
kubectl create secret generic <ocnrfaccesstoken-secret> --from-
file=<ecdsa_private_key.pem> --from-file=<rsa_private_key.pem> --
from-file=<keystore_password.txt> --from-file=<rsa_certificate.crt>
--from-file=<ecdsa_certificate.crt> --dry-run -o yaml -n <Namespace
of OCNRF deployment> | kubectl replace -f - -n <Namespace of OCNRF
deployment>
```

Example:-

The names used below are same as provided in custom_values.yaml in OCNRF
deployment:

```
$ kubectl create secret generic ocnrfaccesstoken-secret --from-
file=ecdsa_private_key.pem --from-file=rsa_private_key.pem --from-
file=keystore_password.txt --from-file=rsa_certificate.crt --from-
file=ecdsa_certificate.crt --dry-run -o yaml -n ocnrf | kubectl
replace -f - -n ocnrf
```

4. Execute the updated command.

5. After successful secret update, the following message is displayed:

```
secret/<ocnrfaccesstoken-secret> replaced
```

Installation Tasks

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

Downloading OCNRF package

Following is the procedure to download the release package from MOS:

1. Login to MOS using the appropriate login credentials.

2. Select **Product & Updates** tab.

3. In **Patch Search** console select **Product or Family (Advanced)** tab.

4. Enter *Oracle Communications Cloud Native Core - 5G* in **Product** field and select the product from the Product drop-down.

5. Select *Oracle Communications Cloud Native Core Network Repository Function <release_number>* in **Release** field.

6. Click **Search**. The **Patch Advanced Search Results** list appears.

7. Select the required patch from the list. The Patch Details window appears.

8. Click on **Download**. File Download window appears.

9. Click on the **<p********_<release_number>_Tekelec>.zip** file.

10. Click on the release package zip file to download the network function patch to the system where network function must be installed.

Installing OCNRF

1. Unzip the release package file to the system where you want to install the network function. You can find the OCNRF package as follows:
   ```
   ReleaseName-pkg-Releasenumber.tgz
   ```

   where:
   ReleaseName is a name which is used to track this installation instance.

   Releasenumber is the release number.

   For example, ocnrf-pkg-1.7.2.0.0.tgz

2. Untar the OCNRF package file to get OCNRF docker image tar file:

   ```
   tar -xvzf ReleaseName-pkg-Releasenumber.tgz
   ```

3. Load the *ocnrf-images-<release_number>.tar* file into the Docker system:

   ```
   docker load --input /IMAGE_PATH/ocnrf-images-<release_number>.tar
   ```

4. Verify that the image is loaded correctly by entering this command:

   ```
   docker images
   ```

5. Execute the following commands to push the docker images to docker registry:

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

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

6. Untar the helm files:

```
tar -xvzf ocnrf-<release_number>.tgz
```

7. Create the customize ocnrf-custom-values-1.7.2.yaml file with the required input parameters. To customize the file, refer to Customizing OCNRF chapter.

8. Go to the extracted OCNRF package as explained in:

```
cd ocnrf-<release_number>
```

9. Install OCNRF by executing the following command:

```
helm install ocnrf/ --name <helm-release> --namespace <k8s
namespace> -f <ocnrf_customized_values.yaml>
```

Example: `helm install ocnrf/ --name ocnrf --namespace ocnrf -f ocnrf-
custom-values-1.7.2.yaml`

> ⚠ **Caution:**
>
> This command will appear hung for a while. Because from OCNRF 1.7.2 release onwards, Kubernetes jobs will get execute by Install/Upgrade/ Rollback helm hooks. Helm Deployment will be shown as DONE after all the applicable hooks are executed.
> timeout duration (optional): If not specified, default value will be 300 (300 seconds) in Helm2 and 5m (5 minutes) in Helm3. Specifies the time to wait for any individual kubernetes operation (like Jobs for hooks). Default value is 5m0s. If the helm install command fails at any point to create a kubernetes object, it will internally call the purge to delete after timeout value (default: 300s). Here timeout value is not for overall install, but it is for automatic purge on installation failure.

To verify the deployment status, open a new terminal and execute the following command:

Command: `$ watch kubectl get pods -n <k8s namespace>`

The pod status gets updated on a regular interval. When helm install command exits with the status, you may stop watching the status of kubernetes pods.

> **✎ Note:**
>
> In case helm purge do not clean the deployment and kubernetes objects completely then follow Cleaning OCNRF deployment section.

10. Execute the following command to check the status:
    `helm status <helm-release>`

    For example: `helm status ocnrf`

11. Execute the following command to check status of the services:
    `kubectl -n <k8s namespace> get services`

    For example:
    `kubectl -n ocnrf get services`

    Note: If `external load balancer` is used, `EXTERNAL-IP` is assigned to `<helm release name>-ingressgateway.ocnrf` is the release name. `ocnrf` is the helm release name.

    ```
    NAME                    TYPE          CLUSTER-IP     EXTERNAL-IP
    PORT(S)            AGE
    ocnrf-egressgateway     ClusterIP     10.233.1.61    <none>
    8080/TCP,5701/TCP   30h
    ocnrf-ingressgateway    LoadBalancer  10.233.52.194  <pending>
    80:31776/TCP        30h
    ocnrf-nfaccesstoken     ClusterIP     10.233.53.115  <none>
    8080/TCP            30h
    ocnrf-nfdiscovery       ClusterIP     10.233.21.28   <none>
    8080/TCP            30h
    ocnrf-nfregistration    ClusterIP     10.233.4.140   <none>
    8080/TCP            30h
    ocnrf-nfsubscription    ClusterIP     10.233.44.98   <none>
    8080/TCP            30h
    ocnrf-nrfauditor        ClusterIP     10.233.1.71    <none>
    8080/TCP            30h
    ocnrf-nrfconfiguration  LoadBalancer  10.233.40.230  <pending>
    8080:30076/TCP      30h
    ocnrf-ocnrf-app-info    ClusterIP     10.104.113.86  <none>
    5906/TCP            30h
    ```

12. Execute the following command to check status of the pods:
    `kubectl get pods -n <k8s namespace>`

    Status column of all the pods should be 'Running'.

    Ready column of all the pods should be n/n, where n is number of containers in the pod.

    For example:

    `kubectl get pods -n ocnrf`

    ```
    NAME                                   READY  STATUS   RESTARTS
    AGE
    ocnrf-egressgateway-d6567bbdb-9jrsx     2/2    Running  0
    30h
    ```

```
ocnrf-egressgateway-d6567bbdb-ntn2v        2/2    Running  0
30h
ocnrf-ingressgateway-754d645984-h9vzq      2/2    Running  0
30h
ocnrf-ingressgateway-754d645984-njz4w      2/2    Running  0
30h
ocnrf-nfaccesstoken-59fb96494c-k8w9p       2/2    Running  0
30h
ocnrf-nfaccesstoken-49fb96494c-k8w9q       2/2    Running  0
30h
ocnrf-nfdiscovery-84965d4fb9-rjxg2         1/1    Running  0
30h
ocnrf-nfdiscovery-94965d4fb9-rjxg3         1/1    Running  0
30h
ocnrf-nfregistration-64f4d8f5d5-6q92j      1/1    Running  0
30h
ocnrf-nfregistration-44f4d8f5d5-6q92i      1/1    Running  0
30h
ocnrf-nfsubscription-5b6db965b9-gcvpf      1/1    Running  0
30h
ocnrf-nfsubscription-4b6db965b9-gcvpe      1/1    Running  0
30h
ocnrf-nrfauditor-67b676dd87-xktbm          1/1    Running  0
30h
ocnrf-nrfconfiguration-678fddc5f5-c5htj    1/1    Running  0
30h
ocnrf-appinfo-8b7879cdb-jds4r              1/1    Running  0
30h
```

3

Customizing OCNRF

This section includes information about OCNRF customization.

- OCNRF Configuration
- OCNRF Configurable Parameters

OCNRF Configuration

This section describes about the OCNRF customization.

The OCNRF deployment is customized by overriding the default values of various configurable parameters.

Follow the below steps to customize the `ocnrf-custom-values-1.7.2.yaml` file as per the required parameters:

1. Go to the Oracle Help Center (OHC) Web site.
2. Navigate to **Industries->Communications->Cloud Native Core->Release 2.2.1**.
3. Click the **NRF Custom Template** link to download the zip file.
4. Unzip the file to get ocnrf-custom-configTemplates-1.7.2.0.0 file that contains the ocnrf-custom-configTemplates-1.7.2.0.0. This file is used during installation.

   - `ocnrf-custom-values-1.7.2.yaml`: This file is used during installation.
   - `NrfDashboard-1.7.2.json`: This file is used by `grafana`.
   - `NrfAlertrules-1.7.2.yaml`: This file is used for `prometheus`.
   - `OCNRF-MIB-TC-1.7.2.mib`: This is considered as OCNRF top level mib file, where the Objects and their data types are defined.
   - `OCNRF-MIB-1.7.2.mib`: This file fetches the Objects from the top level mib file and based on the Alert notification, these objects can be selected for display.
   - `OCNRF-Configuration-OpenAPI-1.7.2.yaml`: This file is OPEN API specification for OCNRF configuration.

5. Customize the ocnrf-custom-values-1.7.2.yaml file.
6. Save the updated ocnrf-custom-values-1.7.2.yaml file in the helm chart directory.

> ✏️ **Note:**
>
> Refer section OCNRF Configuration Parameters to know more about the configurable parameters.

OCNRF Images

Following are the OCNRF images:

Table 3-1   OCNRF Images

Services	Image	Tag
<helm-release-name>-nfregistration	`ocnrf-nfregistration`	1.7.2
<helm-release-name>-nfsubscription	`ocnrf-nfsubscription`	1.7.2
<helm-release-name>-nfdiscovery	`ocnrf-nfdiscovery`	1.7.2
<helm-release-name>-nrfauditor	`ocnrf-nrfauditor`	1.7.2
<helm-release-name>-nrfconfiguration	`ocnrf-nrfconfiguration`	1.7.2
<helm-release-name>-appinfo	`ocnrf-appinfo`	1.7.3
<helm-release-name>-nfaccesstoken	`configurationinit`	1.2.0
	`configurationupdate`	1.2.0
	`ocnrf-nfaccesstoken`	1.7.2
<helm-release-name>-egressgateway	`configurationinit`	1.2.0
	`configurationupdate`	1.2.0
	`ocegress_gateway`	1.7.7
<helm-release-name>-ingressgateway	`configurationinit`	1.2.0
	`configurationupdate`	1.2.0
	`ocingress_gateway`	1.7.7

> **Note:**
>
> IngressGateway, EgressGateway and NFAccessToken uses same configurationinit and configurationupdates docker images.

ASM Specific Configuration

Perform the following configuration changes when the OCNRF is deployed with Aspen Service Mesh (ASM):

1. Add the following annotation to `ocnrf-custom-values-1.7.2.yaml` file:

```
  # ********  Sub-Section Start: Custom Extension Global Parameters
********

#*****************************************************************
*******
global:
  customExtension:
    allResources:
      labels: {}
      annotations:
        sidecar.istio.io/inject: "false"

    lbServices:
      labels: {}
      annotations: {}
```

```
      lbDeployments:
        labels: {}
        annotations:
          sidecar.istio.io/inject: "true"
          oracle.com/cnc: "true"
      nonlbServices:
        labels: {}
        annotations: {}

      nonlbDeployments:
        labels: {}
        annotations:
          sidecar.istio.io/inject: "true"
          oracle.com/cnc: "true"

  # ********  Sub-Section End: Custom Extensions Global Parameters
********

#*****************************************************************
*******
```

2. Enable Service Mesh Flag under ingress-gateway section in `ocnrf-custom-values-1.7.2.yaml` file:

```
ingress-gateway:
  # Mandatory: This flag needs to set it "true" is Service Mesh
would be present where NRF will be deloyed
  serviceMeshCheck: true
```

3. Change Ingress-Gateway and OCNRF configuration service types to **ClusterIP**:
   Under ingress-gateway Section

```
ingress-gateway:
  global:
    # Service Type
    type: ClusterIP
```

   Under nrfconfiguration Section

```
nrfconfiguration:
  service:
    # Service Type
    type: ClusterIP
```

4. Enable HTTP rewrite Probe for appinfo service:

```
appinfo:
  deployment:
    customExtension:
      labels: {}
      annotations:
        sidecar.istio.io/rewriteAppHTTPProbers: "true"
```

5. Add Kubernetes API Server IP to `excludeOutbondIP` List under nfaccesstoken section:

```
nfaccesstoken:
  deployment:
    customExtension:
      labels: {}
      annotations:
        traffic.sidecar.istio.io/excludeOutboundIPRanges:
<Kubernetes API Server IP Address>
```

OCNRF Configuration Parameters

This section includes information about the configuration parameters of OCNRF.

OCNRF allows customization of parameters for the following services and related settings.

Mandatory Configurations

Following is the mandatory parameter, which must be configured before installing OCNRF:

• `nrfInstanceId`: NFInstanceID of OCNRF.

Global Parameters

Table 3-2   Global Parameters

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`mysql.primary.host`	Primary DB Connection Service IP or Hostname	mysql-connectivity-service.occne-infra	M	Primary DB Connection Service HostName or IP	OCNRF connects to Primary DB Connection Service if not available then it connects to Secondary DB Connection Service. For NDB Cluster, use Host/IP of the DB Connection Service.
`mysql.primary.port`	Primary DB Connection Service	3306	M	Primary DB Connection Service Port	Port that is used while connecting to Primary DB Connection Service.
`mysql.secondary.host`	Secondary DB Connection Service IP or Hostname		O	Secondary DB Connection Service HostName or IP	OCNRF connects to Secondary DB Connection Service only if the Primary DB Connection Service is unavailable. It again switch pack to Primary DB Connection Service one it is available. For NDB Cluster, use Host/IP of the Remote DB Connection Service (if available).

Table 3-2   (Cont.) Global Parameters

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`mysql.seco ndary.port`	Secondary DB Connection Service Port		O	Secondary DB Connection Service Port	Port that is used while connecting to Secondary DB Connection Service.
`nrfInstanc eId`	OCNRF's NF Instance ID		M		This is the NfInstanceId of OCNRF that will get deployed. Format of NfInstanceId: Universally Unique Identifier (UUID) version 4, as described in IETF RFC 4122 e.g.: 6faf1bbc-6e4a-4454-a507-a14ef8e1bc5c This ID is used to uniquely identify this OCNRF instance in a Geo-Redundant Deployment. Hence it is very important that the Instance ID MUST be unique across all OCNRF deployments.
`dockerRegi stry`	Registry for docker		M		Docker Registry's FQDN/ Port where OCNRF's docker images are available.
`database.n ameSpace`	Namespace for database connection	ocnrf	M		The Namespace where the Kubernetes Secret is created which contains MYSQL details. **Note**: See database.name configuration for more details.
`database.n ame`	Secret name for OCNRF Application user used for APP-INFO	appuse r-secret	M		The Kubernetes Secret which contains the Database name, Database User name and the Password for OCNRF Application user. **Note**: Refer OCNRF Pre-requisites section for the file format.

Table 3-2   (Cont.) Global Parameters

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`database.appUserSecretName`	Secret name for OCNRF Application user	appuser-secret	M		The Kubernetes Secret which contains the Database name, Database User name and the Password for OCNRF Application user. **Note**: Refer OCNRF Pre-requisites section for the file format.
`database.privilegedUserSecretName`	Secret name for OCNRF Privileged user	privilegeduser-secret	M		The Kubernetes Secret which contains the Database name, Database User name and the Password for OCNRF Privileged user. **Note**: Refer OCNRF Pre-requisites section for the file format.
`hookJobResources.limits.cpu`	Maximum amount of CPU that **K8s** will allow the hook job resource to use	2	O		It is the maximum CPU resource allocated to hook job.
`hookJobResources.limits.memory`	Maximum memory that **K8s** will allow the hook job resource to use	2Gi	O		It is the maximum Memory allocated to hook job.
`hookJobResources.requests.cpu`	The amount of CPU that the system will guarantee for the hook job resource, **and K8s** will use this value to decide on which node to place the pod	1	O		It is the maximum CPU resource for requests allocated to hook job.
`hookJobResources.requests.memory`	The memory that the system will guarantee for the hook job resource, **and K8s** will use this value to decide on which node to place the pod	1Gi	O		It is the maximum memory for requests allocated to hook job.

ORACLE®

Table 3-2   (Cont.) Global Parameters

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`serviceAcc ountName`	ServiceAccount which is having permission for get, watch and list operation for following kubernetes resources; services, configmaps, pods, secrets and endpoints		M		This SeviceAccount is used for: • fetching MYSQL DB Details from configured kubernetes secret • fetching OCNRF's Private Key, OCNRF's Certificate and CA Certificate from configured kubernetes secret • fetching OCNRF's Private and OCNRF's Public Keys for Digitally Signing AccessTokenClaims. • fetching Producer/ Consumer NF's Service/Endpoint details for routing messages from/to Egress/Ingress Gateways. Refer to prerequisites for command details.
`customExte nsion.allR esources.l abels`	Custom Labels that needs to be added to all the OCNRF k8s resources		O		This can be used to add custom label(s) to all k8s resources that will be created by OCNRF helm chart.
`customExte nsion.allR esources.a nnotations`	Custom Annotations that needs to be added to all the OCNRF k8s resources		O		This can be used to add custom annotation(s) to all k8s resources that will be created by OCNRF helm chart.
`customExte nsion.lbSe rvices.lab els`	Custom Labels that needs to be added to OCNRF Services that are considered as Load Balancer type		O		This can be used to add custom label(s) to all Load Balancer Type Services that will be created by OCNRF helm chart.

Table 3-2   (Cont.) Global Parameters

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
customExte nsion.lbSe rvices.ann otations	Custom Annotations that needs to be added to OCNRF Services that are considered as Load Balancer type		O		This can be used to add custom annotation(s) to all Load Balancer Type Services that will be created by OCNRF helm chart.
customExte nsion.lbDe ployments. labels	Custom Labels that needs to be added to OCNRF Deployments that are associated to a Service which is of Load Balancer type		O		This can be used to add custom label(s) to all Deployments that will be created by OCNRF helm chart which are associated to a Service which if of Load Balancer Type.
customExte nsion.lbDe ployments. annotation s	Custom Annotations that needs to be added to OCNRF Deployments that are associated to a Service which is of Load Balancer type		O		This can be used to add custom annotation(s) to all Deployments that will be created by OCNRF helm chart which are associated to a Service which if of Load Balancer Type.
customExte nsion.nonl bServices. labels	Custom Labels that needs to be added to OCNRF Services that are considered as not Load Balancer type		O		This can be used to add custom label(s) to all non-Load Balancer Type Services that will be created by OCNRF helm chart.
customExte nsion.nonl bServices. annotation s	Custom Annotations that needs to be added to OCNRF Services that are considered as not Load Balancer type		O		This can be used to add custom annotation(s) to all non-Load Balancer Type Services that will be created by OCNRF helm chart.
customExte nsion.nonl bDeploymen ts.labels	Custom Labels that needs to be added to OCNRF Deployments that are associated to a Service which is not of Load Balancer type		O		This can be used to add custom label(s) to all Deployments that will be created by OCNRF helm chart which are associated to a Service which if not of Load Balancer Type.

Table 3-2   (Cont.) Global Parameters

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`customExte nsion.nonl bDeploymen ts.annotat ions`	Custom Annotations that needs to be added to OCNRF Deployments that are associated to a Service which is not of Load Balancer type		O		This can be used to add custom annotation(s) to all Deployments that will be created by OCNRF helm chart which are associated to a Service which if not of Load Balancer Type.
`k8sResourc e.containe r.prefix`	Value that will be prefixed to all the container names of OCNRF.		O		This value will be used to prefix to all the container names of OCNRF.
`k8sResourc e.containe r.suffix`	Value that will be suffixed to all the container names of OCNRF.		O		This value will be used to suffix to all the container names of OCNRF.

Ingress Gateway Global Parameters

Table 3-3   Ingress Gateway Global Parameters

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`staticIpAd dressEnabl ed`	Static load balancer IP enabled flag	false	O	true/false	
`staticIpAd dress`	Static IP address assigned to the Load Balancer from the metalLB IP pool.	`<ipaddr ess>`	M, when `static IpAddr essEna bledis` true		If Static load balancer IP needs to be set, then set staticIpAddressEnabled flag to true and provide value for staticIpAddress. Else random IP will be assigned by the metalLB from its IP Pool.
`staticNode PortEnable d`	Static Node Port enabled flag	false	O	true/false	If Static node port needs to be set, then set staticNodePortEnabled flag to true and provide value for staticHttpNodePort or staticHttpsNodePort. Else random node port will be assigned by K8.

Table 3-3   (Cont.) Ingress Gateway Global Parameters

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`staticHttp NodePort`	HTTP node port	30080	M, when `static NodePo rtEnab led` is true and `ingres s- gatewa y.enab leInco mingHt tp` is true		
`staticHttp sNodePort`	HTTPs node port	30443	M, when `static NodePo rtEnab led` is true and `ingres s- gatewa y.enab leInco mingHt tps` is true		
`publicHttp SignalingP ort`	Service Port on which OCNRF's Ingress Gateway is exposed	80	O		If enableIncomingHttp is true, publicHttpSignalingPort will be used as HTTP/2.0 Port (unsecured)
`publicHttp sSignallin gPort`	Service Port on which OCNRF's Ingress Gateway is exposed	443	O		If enableIncomingHttps is true, publicHttpsSignallingPort Port will be used as HTTPS/2.0 Port (secured TLS)

Ingress Gateway

Table 3-4   Ingress Gateway

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`ingress-gateway.enableIncomingHttp`	This flag is for enabling/ disabling HTTP/2.0 (insecure) in Ingress Gateway.	true	O	true/false	If the value is set to false, OCNRF will not accept any HTTP/2.0 (unsecured) Traffic. If the value is set to true, OCNRF will accept HTTP/2.0 (unsecured) Traffic
`ingress-gateway.enableIncomingHttps`	This flag is for enabling/ disabling HTTPS/2.0 (secure) in Ingress Gateway.	false	O	true/false	If the value is set to false, OCNRF will not accept any HTTPS/2.0 (unsecured) Traffic. If the value is set to true, OCNRF will accept HTTPS/2.0 (unsecured) Traffic
`ingress-gateway.serviceMeshCheck`	This flag needs to be set to "true" if Service Mesh exists where OCNRF is deployed.	false	O	true/false	If the value is set to false, OCNRF's ingress-gateway will try to create connection directly with the backend micro-services's PODs. If the value is set to true, OCNRF's ingress-gateway will try to create connection using Service FQDN of the backend micro-services.
`ingress-gateway.image.name`	Ingress Gateway image name.	ocingress_gateway	O		
`ingress-gateway.image.tag`	Tag name of Ingress Gateway image	OCNRF images	O		
`ingress-gateway.image.pullPolicy`	This setting will tell if image need to be pulled or not	IfNotPresent	O	Always, IfNotPresent, Never	
`ingress-gateway.initContainersImage.name`	Image Name for Ingress Gateway init container	configurationinit	O		
`ingress-gateway.initContainersImage.tag`	Tag name of Ingress Gateway init container	OCNRF images	O		

Table 3-4   (Cont.) Ingress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
ingress-gateway.initContainersImage.pullPolicy	This setting will tell if image need to be pulled or not	IfNotPresent	O	Always, IfNotPresent, Never	
ingress-gateway.updateContainersImage.name	Image Name for Ingress Gateway update container	configurationupdate	O		
ingress-gateway.updateContainersImage.tag	Tag name of Ingress Gateway update container	OCNRF images	O		
ingress-gateway.updateContainersImage.pullPolicy	This setting will tell if image need to be pulled or not	IfNotPresent	O	Always, IfNotPresent, Never	
ingress-gateway.jaegerTracingEnabled	Flag to enable or disable the Jaeger Tracing at ingress-gateway	false	O	true / false	While making this flag as true, update the below attributes with correct values.
ingress-gateway.opentracing.jaeger.udpsender.host	Host name of Jaeger Agent Service	jaeger-agent.cne-infra	M, if ingress-gateway.jaegerTracingEnabled is true		
ingress-gateway.opentracing.jaeger.udpsender.port	Port of Jaeger Agent Service	6831	M, if ingress-gateway.jaegerTracingEnabled is true		

Table 3-4   (Cont.) Ingress Gateway

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`ingress-gateway.op entracing. jaeger.pro babilistic Sampler`	Jaeger message sampler	0.5	O	0 to 1	# Jaeger message sampler. Value range: 0 to 1 # e.g. Value 0: No Trace will be sent to Jaeger collector # e.g. Value 0.3: 30% of message will be sampled and will be sent to Jaeger collector # e.g. Value 1: 100% of message (i.e. all the messages) will be sampled and will be sent to Jaeger collector
`ingress-gateway.ci pherSuites`	Allowed CipherSuites for TLS1.2		M, if `ingres s-gatewa y.enab leInco mingHt tps` is true	- TLS_ECDH E_ECDSA_ WITH_AES_ 256_GCM_ SHA384 - TLS_ECDH E_RSA_WIT H_AES_256 _GCM_SHA 384 - TLS_ECDH E_RSA_WIT H_CHACHA 20_POLY13 05_SHA256 - TLS_DHE_ RSA_WITH_ AES_256_G CM_SHA38 4 - TLS_ECDH E_ECDSA_ WITH_AES_ 128_GCM_ SHA256 - TLS_ECDH E_RSA_WIT H_AES_128 _GCM_SHA 256	

Table 3-4   (Cont.) Ingress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`ingress-gateway.service.ssl.privateKey.k8SecretName`	Secret name that contains OCNRF Ingress gateway Private Key	ocingress-secret	M, if `ingress-gateway.enableIncomingHttps` is true		
`ingress-gateway.service.ssl.privateKey.k8NameSpace`	Namespace in which k8SecretName is present	ocnrf	M, if `ingress-gateway.enableIncomingHttps` is true		
`ingress-gateway.service.ssl.privateKey.rsa.filename`	OCNRF's Private Key (RSA type) file name	rsa_private_key_pkcs1.pem	M, if `ingress-gateway.enableIncomingHttps` is true and `ingress-gateway.service.ssl.initialAlgorithm` is RS256		If initialAlgorithm is configured as RSA, then rsa file name must be configured. Otherwise OCNRF's ingress gateway will not comeup.

Table 3-4   (Cont.) Ingress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`ingress-gateway.service.ssl.privateKey.ecdsa.filename`	OCNRF's Private Key (ECDSA type) file name	ssl_ecdsa_private_key.pem	M, if `ingress-gateway.enableIncomingHttps` is true and `ingress-gateway.service.ssl.initialAlgorithm` is ES256		If initialAlgorithm is configured as ECDSA, then rsa file name must be configured. Otherwise OCNRF's ingress gateway will not comeup.
`ingress-gateway.service.ssl.certificate.k8SecretName`	Secret name that contains OCNRF's Certificate for HTTPS	ocingress-secret	M, if `ingress-gateway.enableIncomingHttps` is true		This is a Secret object for OCNRFcertificate details for HTTPS.
`ingress-gateway.service.ssl.certificate.k8NameSpace`	Namespace in which OCNRF's Certificate is present	ocnrf	M, if `ingress-gateway.enableIncomingHttps` is true		

Table 3-4   (Cont.) Ingress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`ingress-gateway.service.ssl.certificate.rsa.filename`	OCNRF's Certificate (RSA type) file name	ssl_rsa _certificate.crt	M, if `ingress-gateway.enableIncomingHttps` is true and `ingress-gateway.service.ssl.initialAlgorithm` is RS256		If initialAlgorithm is configured as RSA, then rsa file name must be configured. Otherwise OCNRF's ingress gateway will not comeup.
`ingress-gateway.service.ssl.certificate.ecdsa.filename`	OCNRF's Certificate (ECDSA type) file name	ssl_ecdsa_certificate.crt	M, if `ingress-gateway.enableIncomingHttps` is true and `ingress-gateway.service.ssl.initialAlgorithm` is ES256		If initialAlgorithm is configured as ECDSA, then rsa file name must be configured. Otherwise OCNRF's ingress gateway will not comeup.

Table 3-4   (Cont.) Ingress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`ingress-gateway.service.ssl.caBundle.k8SecretName`	Secret name that contains OCNRF's CA details for HTTPS	ocingress-secret	M, if `ingress-gateway.enableIncomingHttps` is true		
`ingress-gateway.service.ssl.caBundle.k8NameSpace`	Namespace in which OCNRF's CA details is present	ocnrf	M, if `ingress-gateway.enableIncomingHttps` is true		
`ingress-gateway.service.ssl.caBundle.filename`	OCNRF's CA bundle filename	caroot.cer	M, if `ingress-gateway.enableIncomingHttps` is true		
`ingress-gateway.service.ssl.keyStorePassword.k8SecretName`	Secret name that contains keyStorePassword	ocingress-secret	M, if `ingress-gateway.enableIncomingHttps` is true		
`ingress-gateway.service.ssl.keyStorePassword.k8NameSpace`	Namespace in which OCNRF's keystore password is present	ocnrf	M, if `ingress-gateway.enableIncomingHttps` is true		

Table 3-4   (Cont.) Ingress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`ingress-gateway.service.ssl.keyStorePassword.fileName`	OCNRF's Key Store password Filename	ssl_key store.txt	M, if `ingress-gateway.enableIncomingHttps` is true		
`ingress-gateway.service.ssl.trustStorePassword.k8SecretName`	Secret name that contains trustStorePassword	ocingre ss-secret	M, if `ingress-gateway.enableIncomingHttps` is true		
`ingress-gateway.service.ssl.trustStorePassword.k8NameSpace`	Namespace in which trustStorePassword is present	ocnrf	M, if `ingress-gateway.enableIncomingHttps` is true		
`ingress-gateway.service.ssl.trustStorePassword.fileName`	OCNRF's trustStorePassword Filename	ssl_trus tstore.tx t	M, if `ingress-gateway.enableIncomingHttps` is true		
`ingress-gateway.service.ssl.initialAlgorithm`	Initial Algorithm for HTTPS	RS256	O	ES256, RS256	Algorithm that will be used in TLS handshake
`ingress-gateway.service.log.level.root`	setting logging level	WARN	O	OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE, ALL	

Table 3-4   (Cont.) Ingress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`ingress-gateway.service.log.level.ingress`	setting logging level	WARN	O	OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE, ALL	
`ingress-gateway.service.log.level.oauth`	setting logging level	WARN	O	OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE, ALL	
`ingress-gateway.service.customExtension.labels`	Custom Labels that needs to be added to ingress-gateway specific Service.		O		This can be used to add custom label(s) to ingress-gateway Service.
`ingress-gateway.service.customExtension.annotations`	Custom Annotations that needs to be added to ingress-gateway specific Services.		O		This can be used to add custom annotation(s) to ingress-gateway Service.
`ingress-gateway.global.type`	Kind of Service that will be used for this deployment	LoadBalancer	O	ClusterIP, NodePort, LoadBalancer and ExternalName	It is not recommended to change the Service Type.
`ingress-gateway.deployment.customExtension.labels`	Custom Labels that needs to be added to ingress-gateway specific Deployment.		O		This can be used to add custom label(s) to ingress-gateway Deployment.
`ingress-gateway.deployment.customExtension.annotations`	Custom Annotations that needs to be added to ingress-gateway specific Deployment.		O		This can be used to add custom annotation(s) to ingress-gateway Deployment.
`ingress-gateway.resources.limits.cpu`	Maximum amount of CPU that **K8s** will allow the ingress-gateway service container to use	4	O		It is the maximum CPU resource allocated to ingress-gateway.

Table 3-4   (Cont.) Ingress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`ingress-gateway.resources.limits.initServiceCpu`	Maximum amount of CPU that **K8s** will allow the ingress-gateway init container to use	1	O		It is the CPU resource allocated to ingress-gateway init container.
`ingress-gateway.resources.limits.updateServiceCpu`	Maximum amount of CPU that **K8s** will allow the ingress-gateway update container to use	1	O		It is the CPU resource allocated to ingress-gateway update container.
`ingress-gateway.resources.limits.memory`	Maximum memory that **K8s** will allow the ingress-gateway service container to use	4Gi	O		It is the maximum Memory allocated to ingress-gateway.
`ingress-gateway.resources.limits.initServiceMemory`	Memory Limit for ingress-gateway init container	1Gi	O		It is the memory allocated to ingress-gateway init container.
`ingress-gateway.resources.limits.updateServiceMemory`	Memory Limit for ingress-gateway update container	1Gi	O		It is the memory allocated to ingress-gateway update container.
`ingress-gateway.resources.requests.cpu`	The amount of CPU that the system will guarantee for the ingress-gateway service container, **and K8s** will use this value to decide on which node to place the pod	4	O		It is the maximum CPU resource allocated to ingress-gateway.

Table 3-4   (Cont.) Ingress Gateway

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`ingress-gateway.re sources.re quests.ini tServiceCp u`	The amount of CPU that the system will guarantee for the ingress-gateway init container, **and K8s** will use this value to decide on which node to place the pod	1	O		It is the CPU resource allocated to ingress-gateway init container.
`ingress-gateway.re sources.re quests.upd ateService Cpu`	The amount of CPU that the system will guarantee for the ingress-gateway update container, **and K8s** will use this value to decide on which node to place the pod	1	O		It is the CPU resource allocated to ingress-gateway update container.
`ingress-gateway.re sources.re quests.mem ory`	The memory that the system will guarantee for the ingress-gateway service container, **and K8s** will use this value to decide on which node to place the pod	4Gi	O		It is the maximum memory for requests allocated to ingress-gateway.
`ingress-gateway.re sources.re quests.ini tServiceMe mory`	Memory Limit for ingress-gateway init container	1Gi	O		It is the memory allocated to ingress-gateway init container.
`ingress-gateway.re sources.re quests.upd ateService Memory`	Memory Limit for ingress-gateway update container	1Gi	O		It is the memory allocated to ingress-gateway update container.
`ingress-gateway.re sources.ta rget.avera geCpuUtil`	Target CPU utilization after which Horizontal Pod Autoscaler will be triggered.	80	O		

Table 3-4   (Cont.) Ingress Gateway

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`ingress-gateway.minReplicas`	Minimum number of pod that will be deployed	2	O		
`ingress-gateway.maxReplicas`	Maximum number of pod that will be scaled up	5	O		

Egress Gateway

Table 3-5   Egress Gateway

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`egress-gateway.enableOutgoingHttps`	This flag is for enabling/ disabling HTTPS/2.0 (secured TLS) in Egress Gateway.	false	O	true/false	If the value is set to false, OCNRF will not accept any HTTPS/2.0 (unsecured) Traffic. If the value is set to true, OCNRF will accept HTTPS/2.0 (unsecured) Traffic
`egress-gateway.deploymentegressgateway.image`	Egress Gateway image name	ocegress_gateway	O		
`egress-gateway.deploymentegressgateway.imageTag`	tag name of image	OCNRF images	O		
`egress-gateway.deploymentegressgateway.pullPolicy`	This setting will tell if image need to be pulled or not	IfNotPresent	O	Always, IfNotPresent, Never	
`egress-gateway.initContainersImage.name`	Image Name for Egress Gateway init container	configurationinit	O		

Table 3-5   (Cont.) Egress Gateway

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`egress-gateway.in itContaine rsImage.ta g`	Tag name of Egress Gateway init container	OCNRF images	O		
`egress-gateway.in itContaine rsImage.pu llPolicy`	This setting will tell if image need to be pulled or not	IfNotPr esent	O	Always, IfNotPresent , Never	
`egress-gateway.up dateContai nersImage. name`	Image Name for Egress Gateway update container	configur ationup date	O		
`egress-gateway.up dateContai nersImage. tag`	Tag name of Egress Gateway update container	OCNRF images	O		
`egress-gateway.up dateContai nersImage. pullPolicy`	This setting will tell if image need to be pulled or not	IfNotPr esent	O	Always, IfNotPresent , Never	
`egress-gateway.ja egerTracin gEnabled`	Flag to enable or disable the Jaeger Tracing at egress gateway	false	O	true / false	While making this flag as true, update the below attributes with correct values.
`egress-gateway.op entracing. jaeger.udp sender.hos t`	Host name of Jaeger Agent Service	jaeger-agent.c ne-infra	M, if `egress – gateway.jaeg erTrac ingEna bled` is enabled		

Table 3-5   (Cont.) Egress Gateway

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`egress-gateway.op entracing. jaeger.udp sender.por t`	Port of Jaeger Agent Service	6831	M, if `egress - gatewa y.jaeg erTrac ingEna bled` is enabled		
`egress-gateway.op entracing. jaeger.pro babilistic Sampler`	Jaeger message sampler	0.5	O	0 to 1	# Jaeger message sampler. Value range: 0 to 1 # e.g. Value 0: No Trace will be sent to Jaeger collector # e.g. Value 0.3: 30% of message will be sampled and will be sent to Jaeger collector # e.g. Value 1: 100% of message (i.e. all the messages) will be sampled and will be sent to Jaeger collector
`egress-gateway.sc pIntegrati onEnabled`	Using SCP as an Proxy in Egress Gateway	false	O	true/false	If it is configured as false, SCP will not be used as an proxy. Messages will be directly sent to the Producers/HTTP Servers. If it is configured as true, SCP will be used as an Proxy for delivering messages to the Producers/HTTP Servers.
`egress-gateway.sc pHttpHost`	SCP Configuration For Egress Gateway	localho st	M, if `egress - gatewa y.scpI ntegra tionEn abled` is true		All the SCP related configuration will be used only if scpIntegrationEnabled is set to true. SCP's HTTP Host/IP and Port Combination. This will be while sending HTTP/2.0 (unsecured) traffic.

Table 3-5   (Cont.) Egress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`egress-gateway.scpHttpPort`	SCP's HTTP Port	80	M, if `egress-gateway.scpIntegrationEnabled` is true		
`egress-gateway.scpHttpsHost`	SCP Configuration For Egress Gateway	localho st	M, if `egress-gateway.scpIntegrationEnabled` is true		All the SCP related configuration will be used only if scpIntegrationEnabled is set to true. SCP's HTTP Host/IP and Port Combination. This will be while sending HTTP/2.0 (unsecured) traffic.
`egress-gateway.scpHttpsPort`	SCP's HTTPS Port	443	M, if `egress-gateway.scpIntegrationEnabled` is true		This will be while sending HTTPS/2.0 (unsecured) traffic.
`egress-gateway.scpApiPrefix`	SCP's API Prefix. (Applicable only for SCP with TLS enabled)	/	O		This will be used for constructing the Egress messgage's APIROOT while proxying message to SCP. Change this value to SCP's apiprefix. "/" is not expected to be provided along.
`egress-gateway.scpDefaultScheme`	SCP's default scheme when 3gpp-sbi-target-apiroot header is missing	https	O		

Table 3-5　(Cont.) Egress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`egress-gateway.cipherSuites`	Allowed CipherSuites for TLS1.2		M, if `egress-gateway.enableOutgoingHttps` is true	- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 - TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 - TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 - TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 - TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 - TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	
`egress-gateway.service.ssl.privateKey.k8SecretName`	Secret name that contains OCNRF Egress gateway Private Key	ocegress-secret	M, if `egress-gateway.enableOutgoingHttps` is true		

Table 3-5   (Cont.) Egress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`egress-gateway.service.ssl.privateKey.k8NameSpace`	Namespace in which k8SecretName is present	ocnrf	M, if `egress-gateway.enableOutgoingHttps` is true		
`egress-gateway.service.ssl.privateKey.rsa.filename`	OCNRF's Private Key (RSA type) file name	ssl_rsa_private_key.pem	M, if `egress-gateway.enableOutgoingHttps` is true and `egress-gateway.service.ssl.initialAlgorithm` is RS256		If initialAlgorithm is configured as RSA, then rsa file name must be configured. Otherwise OCNRF's egress gateway will not comeup.
`egress-gateway.service.ssl.privateKey.ecdsa.filename`	OCNRF's Private Key (ECDSA type) file name	ssl_ecdsa_private_key.pem	M, if `egress-gateway.enableOutgoingHttps` is true and `egress-gateway.service.ssl.initialAlgorithm` is ES256		If initialAlgorithm is configured as ECDSA, then rsa file name must be configured. Otherwise OCNRF's egress gateway will not comeup.

Table 3-5   (Cont.) Egress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
egress-gateway.service.ssl.certificate.k8SecretName	Secret name that contains OCNRF's Certificate for HTTPS	ocegress-secret	M, if `egress-gateway.enableOutgoingHttps` is true		This is a Secret object for OCNRFcertificate details for HTTPS.
egress-gateway.service.ssl.certificate.k8NameSpace	Namespace in which OCNRF's Certificate is present	ocnrf	M, if `egress-gateway.enableOutgoingHttps` is true		
egress-gateway.service.ssl.certificate.rsa.filename	OCNRF's Certificate (RSA type) file name	ssl_rsa_certificate.crt	M, if `egress-gateway.enableOutgoingHttps` is true and `egress-gateway.service.ssl.initialAlgorithm` is RS256		If initialAlgorithm is configured as RSA, then rsa file name must be configured. Otherwise OCNRF's egress gateway will not comeup.

Table 3-5   (Cont.) Egress Gateway

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`egress-gateway.se rvice.ssl. certificat e.ecdsa.fi lename`	OCNRF's Certificate (ECDSA type) file name	ssl_ecd sa_certi ficate.cr t	M, if `egress – gatewa y.enab leOutg oingHt tps` is true and `egress – gatewa y.serv ice.ss l.init ialAlg orithm` is ES256		If initialAlgorithm is configured as ECDSA, then rsa file name must be configured. Otherwise OCNRF's egress gateway will not comeup.
`egress-gateway.se rvice.ssl. caBundle.k 8SecretNam e`	Secret name that contains OCNRF's CA details for HTTPS	ocegres s-secret	M, if `egress – gatewa y.enab leOutg oingHt tps` is true		
`egress-gateway.se rvice.ssl. caBundle.k 8NameSpace`	Namespace in which OCNRF's CA details is present	ocnrf	M, if `egress – gatewa y.enab leOutg oingHt tps` is true		
`egress-gateway.se rvice.ssl. caBundle.f ilename`	OCNRF's CA bundle filename	ssl_cab undle.cr t	M, if `egress – gatewa y.enab leOutg oingHt tps` is true		

Table 3-5   (Cont.) Egress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`egress-gateway.service.ssl.keyStorePassword.k8SecretName`	Secret name that contains keyStorePassword	ocegress-secret	M, if `egress-gateway.enableOutgoingHttps` is true		
`egress-gateway.service.ssl.keyStorePassword.k8NameSpace`	Namespace in which OCNRF's keystore password is present	ocnrf	M, if `egress-gateway.enableOutgoingHttps` is true		
`egress-gateway.service.ssl.keyStorePassword.fileName`	OCNRF's Key Store password Filename	ssl_keystore.txt	M, if `egress-gateway.enableOutgoingHttps` is true		
`egress-gateway.service.ssl.trustStorePassword.k8SecretName`	Secret name that contains trustStorePassword	ocegress-secret	M, if `egress-gateway.enableOutgoingHttps` is true		
`egress-gateway.service.ssl.trustStorePassword.k8NameSpace`	Namespace in which trustStorePassword is present	ocnrf	M, if `egress-gateway.enableOutgoingHttps` is true		

Table 3-5   (Cont.) Egress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`egress-gateway.service.ssl.trustStorePassword.fileName`	OCNRF's trustStorePassword Filename	ssl_truststore.txt	M, if `egress-gateway.enableOutgoingHttps` is true		
`egress-gateway.service.ssl.initialAlgorithm`	Initial Algorithm for HTTPS	RS256	O	ES256, RS256	Algorithm that will be used in TLS handshake
`egress-gateway.service.log.level.root`	setting logging level	WARN	O	OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE, ALL	
`egress-gateway.service.log.level.egress`	setting logging level	WARN	O	OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE, ALL	
`egress-gateway.service.log.level.oauth`	setting logging level	WARN	O	OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE, ALL	
`egress-gateway.service.customExtension.labels`	Custom Labels that needs to be added to egress-gateway specific Service		O		This can be used to add custom label(s) to egress-gateway Service
`egress-gateway.service.customExtension.annotations`	Custom Annotations that needs to be added to egress-gateway specific Services		O		This can be used to add custom annotation(s) to egress-gateway Service

Table 3-5   (Cont.) Egress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
egress-gateway.service.type	Kind of Service that will be used for this Deployment	LoadBalancer	O	ClusterIP, NodePort, LoadBalancer and ExternalName	It is not recommended to change the Service Type.
egress-gateway.deployment.customExtension.labels	Custom Labels that needs to be added to egress-gateway specific Deployment		O		This can be used to add custom label(s) to egress-gateway Deployment.
egress-gateway.deployment.customExtension.annotations	Custom Annotations that needs to be added to egress-gateway specific Deployment		O		This can be used to add custom annotation(s) to egress-gateway Deployment.
egress-gateway.resources.limits.cpu	Maximum amount of CPU that **K8s** will allow the egress-gateway service container to use	4	O		It is the maximum CPU resource allocated to egress-gateway.
egress-gateway.resources.limits.initServiceCpu	Maximum amount of CPU that **K8s** will allow the egress-gateway init container to use	1	O		It is the CPU resource allocated to egress-gateway init container.
egress-gateway.resources.limits.updateServiceCpu	Maximum amount of CPU that **K8s** will allow the egress-gateway update container to use	1	O		It is the CPU resource allocated to egress-gateway update container.
egress-gateway.resources.limits.memory	Maximum memory that **K8s** will allow the egress-gateway update container to use	4Gi	O		It is the maximum Memory allocated to egress-gateway.
egress-gateway.resources.limits.initServiceMemory	Memory Limit for egress-gateway init container	1Gi	O		It is the memory allocated to egress-gateway init container.

Table 3-5   (Cont.) Egress Gateway

Parameter	Description	Default value	Manda tory (M)/ Option al (O)	Range or Possible Values (If applicable)	Notes
`egress-gateway.re sources.li mits.updat eServiceMe mory`	Memory Limit for egress-gateway update container	1Gi	O		It is the memory allocated to egress-gateway update container.
`egress-gateway.re sources.re quests.cpu`	The amount of CPU that the system will guarantee for the egress-gateway service container, **and K8s** will use this value to decide on which node to place the pod	4	O		It is the maximum CPU resource allocated to egress-gateway.
`egress-gateway.re sources.re quests.ini tServiceCp u`	The amount of CPU that the system will guarantee for the egress-gateway init container, **and K8s** will use this value to decide on which node to place the pod	1	O		It is the CPU resource allocated to egress-gateway init container.
`egress-gateway.re sources.re quests.upd ateService Cpu`	The amount of CPU that the system will guarantee for the egress-gateway update container, **and K8s** will use this value to decide on which node to place the pod	1	O		It is the CPU resource allocated to egress-gateway update container.
`egress-gateway.re sources.re quests.mem ory`	The memory that the system will guarantee for the egress-gateway service container, **and K8s** will use this value to decide on which node to place the pod	4Gi	O		It is the maximum memory for requests allocated to egress-gateway.

Table 3-5   (Cont.) Egress Gateway

Parameter	Description	Default value	Mandatory (M)/ Optional (O)	Range or Possible Values (If applicable)	Notes
`egress-gateway.resources.requests.initServiceMemory`	Memory Limit for egress-gateway init container	1Gi	O		It is the memory allocated to egress-gateway init container.
`egress-gateway.resources.requests.updateServiceMemory`	Memory Limit for egress-gateway update container	1Gi	O		It is the memory allocated to egress-gateway update container.
`egress-gateway.resources.target.averageCpuUtil`	Target CPU utilization after which Horizontal Pod Autoscaler will be triggered.	80	O		
`egress-gateway.minReplicas`	Minimum number of pod that will be deployed	2	O		
`egress-gateway.maxReplicas`	Maximum number of pod that will be scaled up	5	O		

NF Registration Micro service (`nfregistration`)

Table 3-6   NF Registration

Parameter	Description	Default value	Mandatory (M) / Optional (O)	Range or Possible Values (If applicable)	Notes
`nfregistration.image.name`	Full Image Path	ocnrf-nfregistration	O	Full image path of image	
`nfregistration.image.tag`	Tag of Image	OCNRF images	O	Tag of image in docker repository	
`nfregistration.image.pullPolicy`	This setting will tell if image need to be pulled or not	IfNotPresent	O	Always, IfNotPresent, Never	
`nfregistration.log.level`	Logging level	WARN	O	OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE, ALL	Logging level

Table 3-6   (Cont.) NF Registration

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfregistratio n.service.cus tomExtension. labels`	Custom Labels that needs to be added to nfregistration specific Service		O		This can be used to add custom label(s) to nfregist ration Service
`nfregistratio n.service.cus tomExtension. annotations`	Custom Annotations that needs to be added to nfregistration specific Services		O		This can be used to add custom annotati on(s) to nfregist ration Service
`nfregistratio n.service.typ e`	Service that will be used for this Deployment	ClusterI P	O	ClusterIP, NodePort, LoadBalancer and ExternalName	It is not recom mende d to change the Service Type.
`nfregistratio n.deployment. customExtensi on.labels`	Custom Labels that needs to be added to nfregistration specific Deployment		O		This can be used to add custom label(s) to nfregist ration Deploy ment
`nfregistratio n.deployment. customExtensi on.annotation s`	Custom Annotations that needs to be added to nfregistration specific Deployment		O		This can be used to add custom annotati on(s) to nfregist ration Deploy ment

Table 3-6   (Cont.) NF Registration

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfregistratio n.resources.l imits.cpu`	Maximum amount of CPU that **K8s** will allow the nfregistration service container to use	4	O		It is the maximu m CPU resourc e allocate d to nfregist ration Deploy ment.
`nfregistratio n.resources.l imits.memory`	Maximum memory that **K8s** will allow the nfregistration service container to use	2Gi	O		It is the maximu m Memor y allocate d to nfregist ration Deploy ment.
`nfregistratio n.resources.r equests.cpu`	The amount of CPU that the system will guarantee for the nfregistration service container, **and K8s** will use this value to decide on which node to place the pod	4	O		It is the maximu m CPU resourc e allocate d to nfregist ration Deploy ment.
`nfregistratio n.resources.r equests.memor y`	The memory that the system will guarantee for the nfregistration, **and K8s** will use this value to decide on which node to place the pod	2Gi	O		It is the maximu m memor y for request s allocate d to nfregist ration Deploy ment.
`nfregistratio n.resources.t arget.average CpuUtil`	Target CPU utilization after which Horizontal Pod Autoscaler will be triggered.	80	O		

Table 3-6   (Cont.) NF Registration

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfregistratio n.minReplicas`	Minimum number of pod that will be deployed	2	O		
`nfregistratio n.maxReplicas`	Maximum number of pod that will be scaled up	7	O		
`nfregistratio n.responseCom pressionGzip`	To enable/disable gzip compression	true	O	true/false	

NF Subscription Micro service (`nfsubscription`)

Table 3-7   NF Subscription

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfsubscriptio n.image.name`	Full Image Path	ocnrf-nfsubsc ription	O	Full image path of image	
`nfsubscriptio n.image.tag`	Tag of Image	OCNRF images	O	Tag of image in docker repository	
`nfsubscriptio n.image.pullP olicy`	This setting will tell if image need to be pulled or not	IfNotPr esent	O	Always, IfNotPresent , Never	
`nfsubscriptio n.log.level`	Logging level	WARN	O	OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE, ALL	
`nfsubscriptio n.service.cus tomExtension. labels`	Custom Labels that needs to be added to nfsubscription specific Service		O		This can be used to add custom label(s) to nfsubsc ription Service

Table 3-7   (Cont.) NF Subscription

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfsubscriptio n.service.cus tomExtension. annotations`	Custom Annotations that needs to be added to nfsubscription specific Services		O		This can be used to add custom annotati on(s) to nfsubsc ription Service
`nfsubscriptio n.service.typ e`	Kind of Service that will be used for this Deployment	ClusterI P	O	ClusterIP, NodePort, LoadBalanc er and ExternalNa me	It is not recom mende d to change the Service Type.
`nfsubscriptio n.deployment. customExtensi on.labels`	Custom Labels that needs to be added to nfsubscription specific Deployment		O		This can be used to add custom label(s) to nfsubsc ription Deploy ment.
`nfsubscriptio n.deployment. customExtensi on.annotation s`	Custom Annotations that needs to be added to nfsubscription specific Deployment		O		This can be used to add custom annotati on(s) to nfsubsc ription Deploy ment.

Table 3-7   (Cont.) NF Subscription

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfsubscriptio n.resources.l imits.cpu`	Maximum amount of CPU that **K8s** will allow the nfsubscription service container to use	2	O		It is the maximu m CPU resourc e allocate d to nfsubsc ription Deploy ment.
`nfsubscriptio n.resources.l imits.memory`	Maximum memory that **K8s** will allow the nfsubscription service container to use	2Gi	O		It is the maximu m Memor y allocate d to nfsubsc ription Deploy ment.
`nfsubscriptio n.resources.r equests.cpu`	The amount of CPU that the system will guarantee for the nfsubscription service container, **and K8s** will use this value to decide on which node to place the pod	2	O		It is the maximu m CPU resourc e allocate d to nfsubsc ription Deploy ment.
`nfsubscriptio n.resources.r equests.memor y`	The memory that the system will guarantee for the nfsubscription, **and K8s** will use this value to decide on which node to place the pod	2Gi	O		It is the maximu m memor y for request s allocate d to nfsubsc ription Deploy ment.
`nfsubscriptio n.resources.t arget.average CpuUtil`	Target CPU utilization after which Horizontal Pod Autoscaler will be triggered.	80	O		

Table 3-7   (Cont.) NF Subscription

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
nfsubscriptio n.minReplicas	Minimum number of pod that will be deployed	2	O		
nfsubscriptio n.maxReplicas	Maximum number of pod that will be scaled up	7	O		

OCNRF Auditor Micro service (`nrfauditor`)

Table 3-8   OCNRF Auditor

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
nrfauditor.im age.name	Full Image Path	ocnrf- nrfaudit or	O	Full image path of image	
nrfauditor.im age.tag	Tag of Image	OCNRF images	O	Tag of image in docker repository	
nrfauditor.im age.pullPolic y	This setting indicates if the image needs to be pulled or not	IfNotPr esent	O	Always, IfNotPresent , Never	
nrfauditor.lo g.level	Logging level	WARN	O	OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE, ALL	
nrfauditor.se rvice.customE xtension.labe ls	Custom Labels that needs to be added to nrfauditor specific Service		O		This can be used to add custom label(s) to nrfaudit or Service

Table 3-8 (Cont.) OCNRF Auditor

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nrfauditor.se rvice.customE xtension.anno tations`	Custom Annotations that needs to be added to nrfauditor specific Services		O		This can be used to add custom annotati on(s) to nrfaudit or Service
`nrfauditor.se rvice.type`	Kind of Service that will be used for this Deployment	ClusterI P	O	ClusterIP, NodePort, LoadBalanc er and ExternalNa me	It is not recom mende d to change the Service Type
`nrfauditor.de ployment.cust omExtension.l abels`	Custom Labels that needs to be added to nrfauditor specific Deployment		O		This can be used to add custom label(s) to nrfaudit or Deploy ment
`nrfauditor.de ployment.cust omExtension.a nnotations`	Custom Annotations that needs to be added to nrfauditor specific Deployment		O		This can be used to add custom annotati on(s) to nrfaudit or Deploy ment

Table 3-8   (Cont.) OCNRF Auditor

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nrfauditor.re sources.limit s.cpu`	Maximum amount of CPU that **K8s** will allow the nrfauditor service container to use	6	O		It is the maximu m CPU resourc e allocate d to nrfaudit or Deploy ment.
`nrfauditor.re sources.limit s.memory`	Maximum memory that **K8s** will allow the nrfauditor service container to use	3Gi	O		It is the maximu m Memor y allocate d to nrfaudit or Deploy ment.
`nrfauditor.re sources.reque sts.cpu`	The amount of CPU that the system will guarantee for the nrfauditor service container, **and K8s** will use this value to decide on which node to place the pod	6	O		It is the maximu m CPU resourc e allocate d to nrfaudit or Deploy ment.
`nrfauditor.re sources.reque sts.memory`	The memory that the system will guarantee for the nrfauditor, **and K8s** will use this value to decide on which node to place the pod	3Gi	O		It is the maximu m memor y for request s allocate d to nrfaudit or Deploy ment.

NF Discovery Micro service (`nfdiscovery`)

Table 3-9   NF Discovery

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfdiscovery.image.name`	Full Image Path	ocnrf-nfdisco very	O	Full image path of image	
`nfdiscovery.image.tag`	Tag of Image	OCNRF images	O	Tag of image in docker repository	
`nfdiscovery.image.pullPolicy`	This setting determines if image needs to be pulled or not	IfNotPr esent	O	Always, IfNotPresent , Never	
`nfdiscovery.log.level`	Logging level	WARN	O	OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE, ALL	
`nfdiscovery.service.customExtension.labels`	Custom Labels that needs to be added to nfdiscovery specific Service		O		This can be used to add custom label(s) to nfdisco very Service
`nfdiscovery.service.customExtension.annotations`	Custom Annotations that needs to be added to nfdiscovery specific Services		O		This can be used to add custom annotati on(s) to nfdisco very Service
`nfdiscovery.service.type`	Kind of Service that will be used for this Deployment	ClusterI P	O	ClusterIP, NodePort, LoadBalanc er and ExternalNa me	

Table 3-9   (Cont.) NF Discovery

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfdiscovery.deployment.customExtension.labels`	Custom Labels that needs to be added to nfdiscovery specific Deployment		O		This can be used to add custom label(s) to nfdisco very Deploy ment
`nfdiscovery.deployment.customExtension.annotations`	Custom Annotations that needs to be added to nfdiscovery specific Deployment		O		This can be used to add custom annotati on(s) to nfdisco very Deploy ment
`nfdiscovery.resources.limits.cpu`	Maximum amount of CPU that **K8s** will allow the nfdiscovery service container to use	4	O		It is the maximu m CPU resourc e allocate d to nfdisco very Deploy ment.
`nfdiscovery.resources.limits.memory`	Maximum memory that **K8s** will allow the nfdiscovery service container to use	2Gi	O		It is the maximu m Memor y allocate d to nfdisco very Deploy ment.

Table 3-9   (Cont.) NF Discovery

Parameter	Description	Default value	Mandatory (M) / Optional (O)	Range or Possible Values (If applicable)	Notes
`nfdiscovery.resources.requests.cpu`	The amount of CPU that the system will guarantee for the nfdiscovery service container, **and K8s** will use this value to decide on which node to place the pod	4	O		It is the maximum CPU resource allocated to nfdiscovery Deployment.
`nfdiscovery.resources.requests.memory`	The memory that the system will guarantee for the nfdiscovery, **and K8s** will use this value to decide on which node to place the pod	2Gi	O		It is the maximum memory for requests allocated to nfdiscovery Deployment.
`nfdiscovery.resources.target.averageCpuUtil`	Target CPU utilization after which Horizontal Pod Autoscaler will be triggered.	80	O		
`nfdiscovery.minReplicas`	Minimum number of pod that will be deployed	2	O		
`nfdiscovery.maxReplicas`	Maximum number of pod that will be scaled up	7	O		

OCNRF Configuration

Table 3-10   OCNRF Configuration

Parameter	Description	Default value	Mandatory (M) / Optional (O)	Range or Possible Values (If applicable)	Notes
`image.name`	Full Image Path	nrfconfiguration	O	Full image path of image	

Table 3-10   (Cont.) OCNRF Configuration

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
image.tag	Tag of Image	OCNRF images	O	Tag of image in docker repository	
image.pull Policy	This setting determines if image needs to be pulled or not	IfNotPr esent	O	Always, IfNotPresent , Never	
log.level	Logging level	WARN	O	OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE	
service.st aticIpAddr essEnabled	Static load balancer IP enabled flag	false	O		If Static load balancer IP needs to be set, then set staticIpAddressEnabled flag to true and provide value for staticIpAddress. Else random IP will be assigned by the metalLB from its IP Pool
service.st aticIpAddr ess	Static load balancer IP	<ipaddr ess>	M, if nrfcon figura tion.s ervice .metal LbIpAl locati onEnab led is true		Static IP address assigned to the Load Balancer from the metalLB IP pool.
service.st aticNodePo rtEnabled	Static Node Port enabled flag	false	O		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

Table 3-10   (Cont.) OCNRF Configuration

Parameter	Description	Default value	Mandatory (M) / Optional (O)	Range or Possible Values (If applicable)	Notes
`service.staticNodePort`	Static Node Port	30076	M, if `nrfconfiguration.service.staticIpAddressEnabled` is enabled.		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
`nrfconfiguration.service.customExtension.labels`	Custom Labels that needs to be added to nrfconfiguration specific Service		O		This can be used to add custom label(s) to nrfconfiguration Service
`nrfconfiguration.service.customExtension.annotations`	Custom Annotations that needs to be added to nrfconfiguration specific Services		O		This can be used to add custom annotation(s) to nrfconfiguration Service
`nrfconfiguration.service.type`	Kind of Service that will be used for this Deployment	LoadBalancer	O	ClusterIP, NodePort, LoadBalancer and ExternalName	It is not recommended to change the Service Type.
`nrfconfiguration.deployment.customExtension.labels`	Custom Labels that needs to be added to nrfconfiguration specific Deployment		O		This can be used to add custom label(s) to nrfconfiguration Deployment
`nrfconfiguration.deployment.customExtension.annotations`	Custom Annotations that needs to be added to nrfconfiguration specific Deployment		O		This can be used to add custom annotation(s) to nrfconfiguration Deployment.

Table 3-10   (Cont.) OCNRF Configuration

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nrfconfigu ration.res ources.lim its.cpu`	Maximum amount of CPU that **K8s** will allow the nrfconfiguration service container to use	2	O		It is the maximum CPU resource allocated to nrfconfiguration Deployment.
`nrfconfigu ration.res ources.lim its.memory`	Maximum memory that **K8s** will allow the nrfconfiguration service container to use	2Gi	O		It is the maximum Memory allocated to nrfconfiguration Deployment.
`nrfconfigu ration.res ources.req uests.cpu`	The amount of CPU that the system will guarantee for the nrfconfiguration service container, **and K8s** will use this value to decide on which node to place the pod	2	O		It is the maximum CPU resource allocated to nrfconfiguration Deployment.
`nrfconfigu ration.res ources.req uests.memo ry`	The memory that the system will guarantee for the nrfconfiguration, **and K8s** will use this value to decide on which node to place the pod	2Gi	O		It is the maximum memory for requests allocated to nrfconfiguration Deployment.
`nrfconfigu ration.res ources.tar get.averag eCpuUtil`	Target CPU utilization after which Horizontal Pod Autoscaler will be triggered.	80	O		

NF Access Token (`nfaccesstoken`)

Table 3-11   NF Access Token

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfaccessto ken.enable d`	Flag to disable Oauth functionality	true	O	true / false	If AccessToken service is not required, operator can choose to set it as false so that nfAccessToken micro-service will not be deployed.
`nfaccessto ken.image. name`	Full Image Path for access token service container	ocnrf-nfacces stoken	O	Full image path of image	
`nfaccessto ken.image. tag`	Tag of Image	OCNRF images	O	Tag of image in docker repository	
`nfaccessto ken.image. pullPolicy`	This setting will tell if image need to be pulled or not	IfNotPr esent	O	Always IfNotPresent Never	
`nfaccessto ken.initCo ntainersIm age.name`	Full Image Path for init container	configur ationinit	O	Image Name for Access token Key certificate infrastructur e	This image is used by OCNRF gateway for Key/ Certificate infrastructure.
`nfaccessto ken.initCo ntainersIm age.tag`	Tag of Image	OCNRF images	O	Tag of image in docker repository	
`nfaccessto ken.initCo ntainersIm age.pullPo licy`	This setting will tell if image need to be pulled or not	IfNotPr esent	O	Always IfNotPresent Never	
`nfaccessto ken.update Containers Image.name`	Full Image Path for update container	configur ationup date	O	Image Name for Access token Key certificate infrastructur e	
`nfaccessto ken.update Containers Image.tag`	Tag of Image	OCNRF images	O	Tag of image in docker repository	

Table 3-11 (Cont.) NF Access Token

Parameter	Description	Default value	Mandatory (M) / Optional (O)	Range or Possible Values (If applicable)	Notes
`nfaccessto ken.update Containers Image.pull Policy`	This setting will tell if image need to be pulled or not	IfNotPr esent	O	Always IfNotPresent Never	
`nfaccessto ken.oauth. nrfInstanc eId`	OCNRF's NF Instance ID that is used for signing AccessTokenClaim	6faf1bb c-6e4a- 4454- a507- a14ef8 e1bc5c	M		This is NRF Instance ID that will be used for signing AccessTokenClaim (is IE of AccessTokenClaim). If NRF needs to issue AccessTokenClaim using its own NF instance ID then the nrfInstanceId configured in the global section (global.nrfInstanceId) needs to configured here again. If NRF needs to issue AccessTokenClaim using a common/virtual then a common/virtual NF instance ID needs to be configured here (along with the common/virtual PrivateKey and Certificate Pair). The same NF instance id and PrivateKey and Certificate Pair needs to be configured in all other NRFs as well so that tokens issues by all the NRF can be validated using a Single NfInstanceId and KeyPair.
`nfaccessto ken.oauth. privateKey .k8SecretN ame`	Secret name that contains OCNRF Private key	ocnrfac cesstok en- secret	M, if `nfacce sstoke n.enab led` is true		This is a Secret object for OCNRFPrivate Key.
`nfaccessto ken.oauth. privateKey .k8NameSpa ce`	Namespace in which OCNRF Private key is present	ocnrf	M, if `nfacce sstoke n.enab led` is true		

Table 3-11   (Cont.) NF Access Token

Parameter	Description	Default value	Mandatory (M) / Optional (O)	Range or Possible Values (If applicable)	Notes
`nfaccessto ken.oauth. privateKey .rsa.filen ame`	OCNRF's Private Key (RSA type) file name	rsa_priv ate_key .pem	M, if `nfacce sstoke n.enab led` is true and `nfacce sstoke n.oaut h.init ialAlg orithm` is RS256		If initialAlgorithm is configured as RSA, then rsa file name must be configured. Otherwise OCNRF gateway will not comeup.
`nfaccessto ken.oauth. privateKey .ecdsa.fil ename`	ECDSA key file names	ecdsa_ private_ key.pe m	M, if `nfacce sstoke n.enab led` is true and `nfacce sstoke n.oaut h.init ialAlg orithm` is ES256		If initialAlgorithm is configured as ECDSA, then rsa file name must be configured. Otherwise OCNRF's NFAccessToken microservice will not comeup.
`nfaccessto ken.oauth. certificat e.k8Secret Name`	Secret name that contains OCNRF's certificate	ocnrfac cesstok en- secret	M, if `nfacce sstoke n.enab led` is true		This is a Secret object for OCNRFcertificate details for HTTPS.
`nfaccessto ken.oauth. certificat e.k8NameSp ace`	Namespace in which k8SecretName is present	ocnrf	M, if `nfacce sstoke n.enab led` is true		

Table 3-11   (Cont.) NF Access Token

Parameter	Description	Default value	Mandatory (M) / Optional (O)	Range or Possible Values (If applicable)	Notes
nfaccessto ken.oauth. certificat e.rsa.file name	OCNRF's certificate (RSA type) file name	rsa_cer tificate. crt	M, if nfacce sstoke n.enab led is true and nfacce sstoke n.oaut h.init ialAlg orithm is RS256		If initialAlgorithm is configured as RSA, then rsa file name must be configured. Otherwise OCNRF's NFAccessToken microservice will not comeup.
nfaccessto ken.oauth. certificat e.ecdsa.fi lename	OCNRF's certificate (ECDSA type) file name	ecdsa_ certifica te.crt	M, if nfacce sstoke n.enab led is true and nfacce sstoke n.oaut h.init ialAlg orithm is ES256		If initialAlgorithm is configured as ECDSA, then rsa file name must be configured. Otherwise OCNRF's NFAccessToken microservice will not comeup.
nfaccessto ken.oauth. keyStorePa ssword.k8S ecretName	Secret name that contains OCNRF's keystore password	ocnrfac cesstok en- secret	M, if nfacce sstoke n.enab led is true		
nfaccessto ken.oauth. keyStorePa ssword.k8N ameSpace	Namespace in which OCNRF's keystore password is present	ocnrf	M, if nfacce sstoke n.enab led is true		Password that is used for creating in-memory Java Key Store (JKS)
nfaccessto ken.oauth. keyStorePa ssword.fil ename	KeyStore password file	keystor e_pass word.txt	M, if nfacce sstoke n.enab led is true		

Table 3-11   (Cont.) NF Access Token

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfaccessto ken.oauth. initialAlg orithm`	Initial Algorithm for Access Token key certificate infrastructure	ES256	O	ES256, RS256	
`nfaccessto ken.log.le vel`	Logging level	WARN	O	OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE	
`nfaccessto ken.servic e.customEx tension.la bels`	Custom Labels that needs to be added to nfaccesstoken specific Service		O		This can be used to add custom label(s) to nfaccesstoken Service
`nfaccessto ken.servic e.customEx tension.an notations`	Custom Annotations that needs to be added to nfaccesstoken specific Services		O		This can be used to add custom annotation(s) to nfaccesstoken Service
`nfaccessto ken.servic e.type`	Kind of Service that will be used for this Deployment	ClusterI P	O	ClusterIP, NodePort, LoadBalanc er and ExternalNa me	It is not recommended to change the Service Type.
`nfaccessto ken.deploy ment.custo mExtension .labels`	Custom Labels that needs to be added to nfaccesstoken specific Deployment		O		This can be used to add custom label(s) to nfaccesstoken Deployment
`nfaccessto ken.deploy ment.custo mExtension .annotatio ns`	Custom Annotations that needs to be added to nfaccesstoken specific Deployment		O		This can be used to add custom annotation(s) to nfaccesstoken Deployment
`nfaccessto ken.resour ces.limits .cpu`	Maximum amount of CPU that **K8s** will allow the nfaccesstoken service container to use	4	O		It is the maximum CPU resource allocated to nfaccesstoken.

Table 3-11   (Cont.) NF Access Token

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfaccessto ken.resour ces.limits .initServi ceCpu`	Maximum amount of CPU that **K8s** will allow the nfaccesstoken initi container to use	1	O		It is the CPU resource allocated to nfaccesstoken init container.
`nfaccessto ken.resour ces.limits .updateSer viceCpu`	Maximum amount of CPU that **K8s** will allow the nfaccesstoken update container to use	1	O		It is the CPU resource allocated to nfaccesstoken update container.
`nfaccessto ken.resour ces.limits .memory`	Maximum memory that **K8s** will allow the nfaccesstoken service container to use	2Gi	O		It is the maximum Memory allocated to nfaccesstoken.
`nfaccessto ken.resour ces.limits .initServi ceMemory`	Memory Limit for nfaccesstoken init container	1Gi	O		It is the memory allocated to nfaccesstoken init container.
`nfaccessto ken.resour ces.limits .updateSer viceMemory`	Memory Limit for nfaccesstoken update container	1Gi	O		It is the memory allocated to nfaccesstoken update container.
`nfaccessto ken.resour ces.reques ts.cpu`	The amount of CPU that the system will guarantee for the nfaccesstoken service container, **and K8s** will use this value to decide on which node to place the pod	4	O		It is the maximum CPU resource allocated to nfaccesstoken.

Table 3-11   (Cont.) NF Access Token

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfaccessto ken.resour ces.reques ts.initSer viceCpu`	The amount of CPU that the system will guarantee for the nfaccesstoken initicontainer, **and K8s** will use this value to decide on which node to place the pod	1	O		It is the CPU resource allocated to nfaccesstoken init container.
`nfaccessto ken.resour ces.reques ts.updateS erviceCpu`	The amount of CPU that the system will guarantee for the nfaccesstoken update container, **and K8s** will use this value to decide on which node to place the pod	1	O		It is the CPU resource allocated to nfaccesstoken update container.
`nfaccessto ken.resour ces.reques ts.memory`	The memory that the system will guarantee for the nfaccesstoken, **and K8s** will use this value to decide on which node to place the pod	2Gi	O		It is the maximum memory for requests allocated to nfaccesstoken.
`nfaccessto ken.resour ces.reques ts.initSer viceMemory`	Memory Limit for nfaccesstoken init container	1Gi	O		It is the memory allocated to nfaccesstoken init container.
`nfaccessto ken.resour ces.reques ts.updateS erviceMemo ry`	Memory Limit for nfaccesstoken update container	1Gi	O		It is the memory allocated to nfaccesstoken update container.
`nfaccessto ken.resour ces.target .averageCp uUtil`	Target CPU utilization after which Horizontal Pod Autoscaler will be triggered.	80	O		

Table 3-11 (Cont.) NF Access Token

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`nfaccessto ken.minRep licas`	Minimum number of pod that will be deployed	2	O		
`nfaccessto ken.maxRep licas`	Maximum number of pod that will be scaled up	7	O		

Application Info

Table 3-12 Application Info (`appinfo`)

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`appinfo.image .name`	Full Image Path	app_inf o	O	Full image path of image	
`appinfo.image .tag`	Tag of Image	OCNRF images	O	Tag of image in docker repository	
`appinfo.pullP olicy`	This setting will tell if image need to be pulled or not	IfNotPr esent	O	Always IfNotPresent Never	
`appinfo.resou rces.limits.c pu`	Maximum amount of CPU that **K8s** will allow the appinfo service container to use	200m	O		It is the maximu m CPU resourc e allocate d to appinfo Deploy ment.
`appinfo.resou rces.limits.m emory`	Maximum memory that **K8s** will allow the appinfo service container to use	1Gi	O		It is the maximu m Memor y allocate d to appinfo Deploy ment.

Table 3-12   (Cont.) Application Info (`appinfo`)

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`appinfo.resou rces.requests .cpu`	The amount of CPU that the system will guarantee for the appinfo service container, **and K8s** will use this value to decide on which node to place the pod	200m	O		It is the maximu m CPU resourc e allocate d to appinfo Deploy ment.
`appinfo.resou rces.requests .memory`	The memory that the system will guarantee for the appinfo serv, **and K8s** will use this value to decide on which node to place the pod	1Gi	O		It is the maximu m memor y for request s allocate d to appinfo Deploy ment.
`appinfo.servi ce.type`	Kind of Service that will be used for this Deployment	ClusterI P	O	ClusterIP, NodePort, LoadBalanc er and ExternalNa me	It is not recom mende d to change the Service Type
`appinfo.servi ce.customExte nsion.labels`	Custom Labels that needs to be added to appinfo specific Service		O		This can be used to add custom label(s) to nfacces stoken Service
`appinfo.servi ce.customExte nsion.annotat ions`	Custom Annotations that needs to be added to appinfo specific Services		O		This can be used to add custom annotati on(s) to nfacces stoken Service

Table 3-12   (Cont.) Application Info (`appinfo`)

Parameter	Description	Default value	Manda tory (M) / Option al (O)	Range or Possible Values (If applicable)	Notes
`appinfo.deplo yment.customE xtension.labe ls`	Custom Labels that needs to be added to appinfo specific Deployment		O		This can be used to add custom label(s) to nfacces stoken Deploy ment
`appinfo.deplo yment.customE xtension.anno tations`	Custom Annotations that needs to be added to appinfo specific Deployment		O		This can be used to add custom annotati on(s) to nfacces stoken Deploy ment

4

Upgrading OCNRF

This section includes information about upgrading an existing OCNRF deployment.

> **Note:**
>
> IF YOU HAVE ENABLED SERVICE MESH THEN YOU HAVE TO
> PERFORM FRESH INSTALLATION OF OCNRF. YOU CANNOT UPGRADE
> FROM OCNRF 1.7.0 TO OCNRF 1.7.2.

When you attempt to upgrade an existing OCNRF 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.

If you need to change any configuration then change the `ocnrf-custom-values-1.7.2.yaml` file with new values.

> **Note:**
>
> It is advisable to create a backup of the file before changing any
> configuration.

To configure the parameters, see section OCNRF Configuration.

> **Caution:**
>
> OCNRF 1.7.2 upgrade supports changing HELM configurable parameters
> only.

Execute the following command to upgrade an existing OCNRF deployment:

```
$ helm upgrade <release> <helm chart> [--version <OCNRF version>] -f
<ocnrf_customized_values.yaml>
```

For example:

```
$ helm upgrade <release> <helm chart> [--version <OCNRF version>] -f
ocnrf-custom-values-1.7.2.yaml
```

To check the status of the upgrade, execute:

```
helm status <helm-release>
```

For example: `helm status ocnrf`

Table 4-1   Parameters and Definitions during OCNRF Upgrade

Parameters	Definitions
`<helm chart>`	It is the name of the chart that is of the form `<repository/ocnrf>`. For example: `reg-1/ocnrf or cne-repo/ocnrf`
`<release>`	It can be found in the output of `helm list` command

In case of backout:

1. Check the history of helm deployment:
   `helm history <helm_release>`

2. Rollback to the required revision:
   `helm rollback <release name> <revision number>`

5
Uninstalling OCNRF

This section explains uninstallation procedure of OCNRF and its details in MySQL.

Deleting the OCNRF deployment

This procedure explains how to delete the OCNRF deployment:

Execute the following command to completely delete or remove the OCNRF deployment:

```
$ helm del --purge <helm-release>
```

Example:

```
$ helm del --purge ocnrf
```

> **Note:**
>
> In case helm purge do not clean the deployment and kubernetes objects completely then follow Cleaning OCNRF deployment section.

To check if helm purge has not deleted all of the kubernetes objects, execute the following command:

```
$ kubectl get all -n <release-namespace>
```

This will give a detailed overview of the current objects of <release-namespace> which were not cleaned during helm purge.

> ⚠ **Caution:**
>
> rbac and service account details may have been created by user itself prior to helm install in same namespace and not using helm install. In case same service account and rbac resource are needed, then don't delete them.

Cleaning OCNRF deployment

This procedure explains how to cleanup the OCNRF deployment.

1. Remove failed helm release:

Run command to get all of the helm release.

```
$ helm ls --all
```

If OCNRF helm release is in a failed state, please purge the namespace using the command:

```
$ helm delete --purge <release-namespace>
```

> **✎ Note:**
>
> If this is taking more time as it will run the delete hook jobs. In this case, run below script parallel in another session to clear all the delete jobs.

Cleanup hook-jobs

```
$ while true; do kubectl delete jobs --all -n <release-namespace>; sleep 5;done
```

Monitor the "helm delete --purge <release-namespace>" command. Once that is succeeded, press "ctrl+c" to stop the above command execution.

2. Cleanup all of the kubernetes objects.

```
$ kubectl get all -n <release-namespace>
This will give a detailed overview of the current objects of
<release-namespace>. Delete all those objects.
Deleting all the kubernetes objects: "kubectl delete all --all -n
<release-namespace>"
Deleting all the current configmaps: "kubectl delete cm --all -n
<release-namespace>"
```

> **⚠ Caution:**
>
> This command will delete all objects of kubernetes in specified namespace. Please note, rbac and service account details may have been created by user itself prior to helm install in same namespace and not using helm install. In case same service account and rbac resource are needed, then don't delete them. In case, custom service account was not provided by the user and safe to remove it, then below commands may help.

```
Deleting all the kubernetes objects: "kubectl delete all --all -n
<release-namespace>"
Deleting all the current configmaps: "kubectl delete cm --all -n
<release-namespace>"
```

3. Execute the following command to delete kubernetes namespace:

> **Caution:**
>
> Removing the namespace will delete all secrets too created in the namespace.

```
$ kubectl delete namespace <ocnrf kubernetes namespace>
```

Example:

```
$ kubectl delete namespace ocnrf
```

Deleting the OCNRF MySQL details

This procedure explains how to delete the OCNRF MySQL database after deletion of OCNRF deployment.

> **Note:**
>
> Procedure can be different for Geo-Redundant OCNRF sites and standalone OCNRF site.

Procedure for Geo-Redundant OCNRF sites

1. Login to the machine which has permission to access the SQL nodes of NDB cluster.

2. Connect to the SQL node of NDB cluster successively. MySQL commands must be run on all the SQL nodes.

3. Login to the MySQL prompt using root permission or user, which has permission to delete the table records. For example: `mysql -h 127.0.0.1 -uroot -p`

   > **Note:**
   >
   > This command may vary from system to system, path for mysql binary, root user and root password. After executing this command, user need to enter the password specific to the user mentioned in the command.

4. Execute the following command to delete data specific to purged site:

```
$ DELETE FROM NfScreening WHERE nrfInstanceId = '<OCNRF's NF
Instance ID of Site under deletion>';
$ DELETE FROM NrfSystemOptions WHERE nrfInstanceId = '<OCNRF's NF
Instance ID of Site under deletion>';
$ DELETE FROM NfInstances WHERE nrfInstanceId = '<OCNRF's NF
Instance ID of Site under deletion>';
$ DELETE FROM NfStatusMonitor WHERE nrfInstanceId = '<OCNRF's NF
Instance ID of Site under deletion>';
$ DELETE FROM NfSubscriptions WHERE nrfInstanceId = '<OCNRF's NF
```

```
Instance ID of Site under deletion>';
$ DELETE FROM NrfEventTransactions WHERE currentOwner = '<OCNRF's
NF Instance ID of Site under deletion>';
```

> ⚠️ **Caution:**
>
> Since these tables are shared by each geo-redundant site, tables shall not be deleted.

5. Exit from MySQL prompt and SQL nodes:

> ✏️ **Note:**
>
> Execute the commands on any one SQL node on one geo-redundant site. Other Geo-redundant sites will get the data records removed automatically.

Procedure for standalone OCNRF site

1. Login to the machine which has permission to access the SQL nodes of NDB cluster

2. Connect to the SQL node of NDB cluster successively. MySQL commands must be run on all the SQL nodes.

3. Login to the MySQL prompt using root permission or user, which has permission to drop the tables. For example: `mysql -h 127.0.0.1 -uroot -p`

> ✏️ **Note:**
>
> This command may vary from system to system, path for mysql binary, root user and root password. After executing this command, user need to enter the password specific to the user mentioned in the command.

4. Execute the following commands to drop the tables:

```
$ DROP TABLE IF EXISTS 'NfInstances';
$ DROP TABLE IF EXISTS 'NfStatusMonitor';
$ DROP TABLE IF EXISTS 'NfSubscriptions';
$ DROP TABLE IF EXISTS 'NfScreening';
$ DROP TABLE IF EXISTS 'NrfSystemOptions';
$ DROP TABLE IF EXISTS 'NrfEventTransactions';
```

5. Exit from MySQL prompt and SQL node

> ✏️ **Note:**
>
> Execute the commands on any SQL node of standalone site.

Procedure for complete removal of MySql database and username

This procedure explains the steps to complete removal of MySql database and username in below cases:

1. OCNRF is not going to be install on that cluster.

2. Change the MySql database name or MySql user name.

Procedure

1. Login to the machine which has permission to access the SQL nodes of NDB cluster.

2. Connect to the SQL node of NDB cluster successively. MySQL commands must be run on all the SQL nodes.

3. Login to the MySQL prompt using root permission or user, which has permission to drop the tables. For example: `mysql -h 127.0.0.1 -uroot -p`

> ✏ **Note:**
>
> This command may vary from system to system, path for mysql binary, root user and root password. After executing this command, user need to enter the password specific to the user mentioned in the command.

4. Execute the following command to remove OCNRF database:

> ⚠ **Caution:**
>
> Removal of database from any one of the SQL node from any one of the cluster will remove the database from all Geo-redundant site.

Remove OCNRF application database

```
$ DROP DATABASE if exists <OCNRF application database>;
```

Example

```
$ DROP DATABASE if exists nrfApplicationDB;
```

Remove OCNRF network scoped database:

```
$ DROP DATABASE if exists <OCNRF network scoped database>;
```

Example

```
$ DROP DATABASE if exists nrfNetworkDB;
```

5. Execute the following command to remove the OCNRF MySql Users:

Remove OCNRF privileged user:

```
$ DROP USER IF EXISTS <OCNRF Privileged-User Name>;
```

Example

```
 $ DROP USER IF EXISTS nrfPrivilegedUsr;
```

Remove OCNRF application user:

```
$ DROP USER IF EXISTS <OCNRF Application User Name>;
```

Example

```
$ DROP USER IF EXISTS nrfApplicationUsr;
```

> ⚠️ **Caution:**
>
> Removal of Mysql Users must be done on all the SQL nodes from all the OCNRF sites.

6. Exit from MySQL prompt and SQL node.

6

Troubleshooting OCNRF

This section provides information to troubleshoot the common error which can be encountered during the installation and upgrade of Oracle Communications Network Repository Function (OCNRF).

Following are the troubleshooting procedures:

- Helm Install Failure
- Custom Value File Parse Failure
- Kubernetes Node Failure
- Tiller Pod Failure

Generic Checklist

The following sections provide generic checklist for troubleshooting tips.

Deployment related tips

Perform the following checks before the deployment:

- Are OCNRF deployment, pods and services created, running and available? Execute following the command:

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

  Inspect the output, check the following columns:

  – AVAILABLE of deployment

  – READY, STATUS and RESTARTS of pod

  – PORT(S) of service

- Is the correct image used and the correct environment variables set in the deployment?
  Execute following the command:

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

  Inspect the output, check the environment and image.

```
# kubectl -n nrf-svc get deployment ocnrf-nfregistration -o yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
    kubectl.kubernetes.io/last-applied-configuration: |
```

{"apiVersion":"apps/v1","kind":"Deployment","metadata":
{"annotations":{},"name":"ocnrf-nfregistration","namespace":"nrf-
svc"},"spec":{"replicas":1,"selector":{"matchLabels":
{"app":"ocnrf-nfregistration"}},"template":{"metadata":
{"labels":{"app":"ocnrf-nfregistration"}},"spec":
{"containers":[{"env":[{"name":"MYSQL_HOST","value":"mysql"},
{"name":"MYSQL_PORT","value":"3306"},
{"name":"MYSQL_DATABASE","value":"nrfdb"},
{"name":"NRF_REGISTRATION_ENDPOINT","value":"ocnrf-
nfregistration"},{"name":"NRF_SUBSCRIPTION_ENDPOINT","value":"ocnrf-
nfsubscription"},{"name":"NF_HEARTBEAT","value":"120"},
{"name":"DISC_VALIDITY_PERIOD","value":"3600"}],"image":"dsr-
master0:5000/ocnrf-
nfregistration:latest","imagePullPolicy":"Always","name":"ocnrf-
nfregistration","ports":
[{"containerPort":8080,"name":"server"}]}]}}}}
      creationTimestamp: 2018-08-27T15:45:59Z
      generation: 1
      name: ocnrf-nfregistration
      namespace: nrf-svc
      resourceVersion: "2336498"
      selfLink: /apis/extensions/v1beta1/namespaces/
nrf-svc/deployments/ocnrf-nfregistration
      uid: 4b82fe89-aa10-11e8-95fd-fa163f20f9e2
spec:
      progressDeadlineSeconds: 600
      replicas: 1
      revisionHistoryLimit: 10
      selector:
        matchLabels:
          app: ocnrf-nfregistration
      strategy:
        rollingUpdate:
          maxSurge: 25%
          maxUnavailable: 25%
        type: RollingUpdate
      template:
        metadata:
          creationTimestamp: null
          labels:
            app: ocnrf-nfregistration
        spec:
          containers:
          - env:
            - name: MYSQL_HOST
              value: mysql
            - name: MYSQL_PORT
              value: "3306"
            - name: MYSQL_DATABASE
              value: nrfdb
            - name: NRF_REGISTRATION_ENDPOINT
              value: ocnrf-nfregistration
            - name: NRF_SUBSCRIPTION_ENDPOINT
              value: ocnrf-nfsubscription
            - name: NF_HEARTBEAT

```
          value: "120"
        - name: DISC_VALIDITY_PERIOD
          value: "3600"
        image: dsr-master0:5000/ocnrf-nfregistration:latest
        imagePullPolicy: Always
        name: ocnrf-nfregistration
        ports:
        - containerPort: 8080
          name: server
          protocol: TCP
        resources: {}
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext: {}
      terminationGracePeriodSeconds: 30
status:
  availableReplicas: 1
  conditions:
  - lastTransitionTime: 2018-08-27T15:46:01Z
    lastUpdateTime: 2018-08-27T15:46:01Z
    message: Deployment has minimum availability.
    reason: MinimumReplicasAvailable
    status: "True"
    type: Available
  - lastTransitionTime: 2018-08-27T15:45:59Z
    lastUpdateTime: 2018-08-27T15:46:01Z
    message: ReplicaSet
"ocnrf-nfregistration-7898d657d9" has successfully progressed.
    reason: NewReplicaSetAvailable
    status: "True"
    type: Progressing
  observedGeneration: 1
  readyReplicas: 1
  replicas: 1
  updatedReplicas: 1
```

- Check if the micro-services can access each other via REST interface.
  Execute following command:

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

Example:

```
# kubectl -n nrf-svc exec $(kubectl -n nrf-svc get pods -o name|cut
-d'/' -f2|grep nfs) --
        curl http://ocnrf-nfregistration:8080/nnrf-nfm/v1/nf-
instances


# kubectl -n nrf-svc exec $(kubectl -n nrf-svc get pods -o name|cut
-d'/' -f2|grep nfr) --
```

```
        curl http://ocnrf-nfsubscription:8080/nnrf-nfm/v1/nf-
instances
```

> **Note:**
>
> These commands are in their simple form and display the logs only if
> there is 1 nrf<registration> and nf<subscription> pod deployed.

Application related tips

Check the application logs and look for exceptions, by executing the following
command:

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

You can use '-f' to follow the logs or 'grep' for specific pattern in the log output.

Example:

```
# kubectl -n nrf-svc logs -f $(kubectl -n nrf-svc get pods -o name|cut
-d'/' -f2|grep nfr)
 # kubectl -n nrf-svc logs -f $(kubectl -n nrf-svc get pods -o name|cut
-d'/' -f2|grep nfs)
```

> **Note:**
>
> These commands are in their simple form and display the logs only if there is
> 1 nrf<registration> and nf<subscription> pod deployed.

Helm Install Failure

This section describes the various scenarios in which `helm install` might fail.
Following are some of the scenarios:

- Incorrect image name in ocnrf-custom-values files
- Docker registry is configured incorrectly
- Continuous Restart of Pods

Incorrect image name in ocnrf-custom-values files

Problem

`helm install` might fail if incorrect image name is provided in the *ocnrf-custom-values*
file.

Error Code/Error Message

When `kubectl get pods -n <ocnrf_namespace>` is executed, the status of the pods
might be ImagePullBackOff or ErrImagePull.

Solution

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

1. Edit *ocnrf-custom-values* file and provide release specific image name and tags. Refer to Customizing OCNRF for OCNRF images details.

2. Execute `helm install` command.

3. Execute `kubectl get pods -n <ocnrf_namespace>` to verify if the status of all the pods is **Running**.

Docker registry is configured incorrectly

Problem

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

Error Code/Error Message

When `kubectl get pods -n <ocnrf_namespace>` is executed, the status of the pods might be ImagePullBackOff or ErrImagePull.

Solution

Configure docker registry on all primary and secondary nodes.

Continuous Restart of Pods

Problem

`helm install` might fail if MySQL primary and secondary hosts may not be configured properly in ocnrf-custom-values.yaml.

Error Code/Error Message

When `kubectl get pods -n <ocnrf_namespace>` is executed, the pods restart count increases continuously.

Solution

MySQL servers(s) may not be configured properly according to the pre-installation steps as mentioned in Configuring MySql database and user.

Custom Value File Parse Failure

This section explains troubleshooting procedure in case of failure during parsing custom values file.

Problem

Not able to parse *ocnrf-custom-values-x.x.x.yaml*, while running helm install.

Error Code/Error Message

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

Symptom

While creating the *ocnrf-custom-values-x.x.x.yaml* file, if the above mentioned error is received, it means that the file is not created properly. The tree structure may not have been followed and/or there may also be tab spaces in the file.

Solution

Following the procedure as mentioned:

1. Download the latest NRF templates zip file from OHC. Refer to Installation Tasks for more information.

2. Follow the steps mentioned in the Installation Tasks section.

Kubernetes Node Failure

Problem

Kubernetes nodes goes down.

Error Code/Error Message

"NotReady" status is displayed against the Kubernetes node.

Symptom

On running the command **kubectl get nodes**, "NotReady" status is displayed, as shown below:

Figure 6-1   Kubernetes Nodes Output

Solution

Following is the procedure to identify the kubernetes nodes failure:

1. Execute the following command to describe the node:
    ```
    kubectl describe node <kubernete_node_name>
    ```

    Example: ```kubectl describe node k8s-1.odyssey.morrisville.us.lab.oracle.com```

2. Check Nodes utilization by running the command:
    ```
    kubectl top nodes
    ```

Tiller Pod Failure

Problem

Tiller Pod is not ready to run helm install.

Error Code/Error Message

The error '*could not find a ready tiller pod*' message is received.

Symptom

When `helm ls` is executed, '*could not find a ready tiller pod*' message is received.

Solution

Following is the procedure to install helm and tiller using the below commands:

1. Delete the pre-installed helm:

```
kubectl delete svc tiller-deploy -n kube-system
kubectl delete deploy tiller-deploy -n kube-system
```

2. Install helm and tiller using this commands:

```
helm init --client-only
helm plugin install https://github.com/rimusz/helm-tiller
helm tiller install
helm tiller start kube-system
```