Oracle® Communications

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

Release 1.2
F21618-01
September 2019

Oracle Communications Network Repository Function (NRF) Cloud Native Installation and Upgrade Guide, Release 1.2

F21618-01

Contents

List of Tables

1

OCNRF Overview

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

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**

OCNRF Supported Services

This section includes information about the services supported by OCNRF.

OCNRF supports the following services:

OCNRF Management Services

The OCNRF Management service is identified by the service operation name `Nnrf_NFManagement`.

OCNRF supports the following management services:

> **Note:**
>
> The respective service operation name is mentioned next to each service.

- **Register NF instance** (`NFRegister`): Allows an NF instance to register its NF profile in the OCNRF along with the list of services provided by the NF instance.
- **Update NF instance** (`NFUpdate`): Enables an NF instance to partially update or replace the parameters of its NF profile in the OCNRF. It also allows to add or delete services provided by the NF instance.
  This operation supports the following:
  - Complete Replacement of NF profile
  - Partial Replacement of NF profile
  - Add, Remove, or Update attributes of NF Profile
  - Heart beat & Load info of NF
- **De-register NF instance** (`NFDeregister`): Enables an NF instance to de-register its NF profile and the services provided by the NF instance from the 5G network.
- **Subscribe to Status** (`NFStatusSubscribe`): Enables an NF instance to subscribe the status changes of other NF instances registered in the OCNRF.
- **Unsubscribe to Status** (`NFStatusUnsubscribe`): Enables an NF instance to unsubscribe the status changes of other NF instances.
- **Receive Notifications of Status** (`NFStatusNotify`): Enables the OCNRF to notify changes in status of NF instances to any subscriber of NF status. Changes also include information regarding newly registered and de-registered NFs.

OCNRF Discovery Services

The OCNRF Discovery service is identified by the service operation name `Nnrf_NFDiscovery Service`.

OCNRF supports the following Discovery service:

> **Note:**
>
> The respective service operation name is mentioned next to the supported Discovery service.

Discover NF instance (`NFDiscover`): OCNRF supports discovery of the IP address/FQDN of the NF instances, or NF Services that match certain input criteria.

References

- CNE Installation Document 1.0
- OCNRF User's Guide

Acronyms and Terminology

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

Table 1-1 Acronyms and Terminology

Field	Description
5G-AN	5G Access Network
5GC	5G Core Network
AMF	Access and Mobility Management Function
CNE	Cloud Native Environment
MMI	Machine Machine Interface
MPS	Messages Per Second
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
OCNRF	Oracle Communications Network Repository Function
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
SEPP	Security Edge Protection Proxy
SCP	Service Controller Proxy
5G System	3GPP system consisting of 5G Access Network (AN), 5G Core Network and UE

Documentation Admonishments

Admonishments are icons and text throughout this manual that alert the reader to assure personal safety, to minimize possible service interruptions, and to warn of the potential for equipment damage.

Table 1-2 Admonishments

Icon	Description
	Danger: (This icon and text indicate the possibility of personal injury.)
	Warning: (This icon and text indicate the possibility of equipment damage.)
	Caution: (This icon and text indicate the possibility of service interruption.)

Locate Product Documentation on the Oracle Help Center Site

Oracle Communications customer documentation is available on the web at the Oracle Help Center site, http://docs.oracle.com. You do not have to register to access these documents. Viewing these files requires Adobe Acrobat Reader, which can be downloaded at http://www.adobe.com.

1. Access the Oracle Help Center site at http://docs.oracle.com.
2. Click **Industries**.
3. Under the Oracle Communications subheading, click **Oracle Communications documentation** link.

   The Communications Documentation page displays.
4. Click on your product and then the release number.

   A list of the documentation set for the selected product and release displays.
5. To download a file to your location, right-click the **PDF** link, select **Save target as** (or similar command based on your browser), and save to a local folder.

Customer Training

Oracle University offers training for service providers and enterprises. Visit our web site to view, and register for, Oracle Communications training:

http://education.oracle.com/communication

To obtain contact phone numbers for countries or regions, visit the Oracle University Education web site:

www.oracle.com/education/contacts

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.

Emergency Response

In the event of a critical service situation, emergency response is offered by the Customer Access Support (CAS) main number at 1-800-223-1711 (toll-free in the US), or by calling the Oracle Support hotline for your local country from the list at http://www.oracle.com/us/support/contact/index.html. The emergency response provides immediate coverage, automatic escalation, and other features to ensure that the critical situation is resolved as rapidly as possible.

A critical situation is defined as a problem with the installed equipment that severely affects service, traffic, or maintenance capabilities, and requires immediate corrective action. Critical situations affect service and/or system operation resulting in one or several of these situations:

- A total system failure that results in loss of all transaction processing capability
- Significant reduction in system capacity or traffic handling capability
- Loss of the system's ability to perform automatic system reconfiguration
- Inability to restart a processor or the system
- Corruption of system databases that requires service affecting corrective actions
- Loss of access for maintenance or recovery operations
- Loss of the system ability to provide any required critical or major trouble notification

Any other problem severely affecting service, capacity/traffic, billing, and maintenance capabilities may be defined as critical by prior discussion and agreement with Oracle.

2
OCNRF Customization

This section includes information about OCNRF customization.

OCNRF Configuration

This section describes about the OCNRF customization.

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

Customize the yaml file `ocnrf-custom-values-1.2.0.yaml` as per the required parameters.

The `ocnrf-custom-values-1.2.0.yaml` template can be downloaded from OHC.

Download the package `Network Repository Function (NRF) Custom Template` and Unzip to get `ocnrf-custom-values-1.2.0.yaml` file.

Sample content of ocnrf-custom-values-1.2.0.yaml

> **Note:**
>
> - To download the `ocnrf-custom-values-1.2.0.yaml` file, refer section, OCNRF Deployment.
> - To know more about the configurable parameters, refer section OCNRF Configurable Parameters.

```
# Copyright 2019 (C), Oracle and/or its affiliates. All rights reserved.


# This yaml file could be supplied in helm install command when deploying OCNRF
v1.2.0
#
# helm install ocnrf/ --name ocnrf --namespace <namespace> -f <this file>
#
# Compatible with OCNRF CHART VERSION 1.2.0
#
# I.
# Update the parameters in global and microservices sections
# 1.) Configure nrfName. This is unique deployment name for OC-NRF
#     global.nrfName
#
# 2.) Set the hostname and port number of the primary and secondary sql nodes in
```

```
#     global.mysql.primary.host
#     global.mysql.secondary.host
#     global.mysql.port
#     global.mysql.database
#     global.mysql.username
#     global.mysql.password
#
# 3.) Set ambassador Id. Ambassador Id to uniquely identify the Ambassador
instance in cluster.
#     All 3 values below should be same.
#
#     global.ambassadorId
#     ambassador.ambassador.id
#     ambassador.service.annotations.getambassador.io/config.ambassador_id
#
#
# 4.) Set ocnrf enpoint. Service Name for OC-NRF API-GW. Format:- NAME-endpoint
#     Here NAME is helm release name.
#
# I.
# - To configure SCP as outbound proxy in NRF, if needed.
# set the SCP worker's k8s service name (<servicename>.<namespace> or <complete
fqdn> and port in
# nfsubscription.http.proxy.host
# nfsubscription.http.proxy.port
#
#############

# section:- global
global:
  nrfName: "default-ocnrf"
  jaeger:
   service:
     name: "cne-tracer-jaeger-collector.cne-infra.svc.cluster.local"
     port: 9411
  mysql:
    primary:
      host: "ocnrf-mysql"
    secondary:
      host: "ocnrf-mysql"
    port: 3306
    database: "nrfdb"
    username: "xxxxxx"
    password: "xxxxxx
  endpoint: "ocnrf-endpoint"
  endpointPort: 80
  ambassadorId: "ocnrf-endpoint"

# section:- nfregistration
nfregistration:
  image:
    repository: ocnrf/ocnrf-nfregistration
    tag: 1.2.0
    pullPolicy: IfNotPresent

# section:- nfsubscription
nfsubscription:
  image:
    repository: ocnrf/ocnrf-nfsubscription
    tag: 1.2.0
    pullPolicy: IfNotPresent
```

```
  http:
    proxy:
      host:
      port: 80

# section:- nrfauditor
nrfauditor:
  image:
    repository: ocnrf/ocnrf-nrfauditor
    tag: 1.2.0
    pullPolicy: IfNotPresent

# section:- nfdiscovery
nfdiscovery:
  image:
    repository: ocnrf/ocnrf-nfdiscovery
    tag: 1.2.0
    pullPolicy: IfNotPresent
  nrfSupportedFeature: false

# section:- nrfconfiguration
nrfconfiguration:
  image:
    repository: ocnrf/ocnrf-nrfconfiguration
    tag: 1.2.0
    pullPolicy: IfNotPresent
  service:
    configServiceNetworkNameEnabled: true
    configServiceNetworkName: "metallb.universe.tf/address-pool: oam"

# section:- ambassador
ambassador:
  fullnameOverride: ocnrf-endpoint
  ambassador:
    id: ocnrf-endpoint
  image:
    repository: ocnrf/ocnrf-endpoint
    tag: 0.50.3
    pullPolicy: IfNotPresent
  podAnnotations:
     prometheus.io/scrape: "true"
     prometheus.io/port: "9102"
  prometheusExporter:
    repository: ocnrf/prom/statsd-exporter
    tag: v0.8.1
  service:
    annotations :
      metallb.universe.tf/address-pool: signaling
      getambassador.io/config: |
           ---
           apiVersion: ambassador/v1
           kind: Module
           name: ambassador
           ambassador_id: ocnrf-endpoint
           config:
             service_port: 8080
```

OCNRF Configurable Parameters

This section includes information about the allowed configurable options on OCNRF micro services that needs to be updated only during the deployment using helm chart.

Note:

- **NAME**: is the release name used in helm install command.
- **NAMESPACE**: is the namespace used in helm install command.
- **K8S_DOMAIN**: is the default kubernetes domain (`svc.cluster.local`).
- Default Helm Release Name: ocnrf

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

Global Parameters

Table 2-1 Global Parameters

Parameter	Description	Default value	Range or Possible Values (If applicable)	Notes
nrfName	OCNRF Deployment Name	default-ocnrf	OCNRF Deployment Name	OCNRF Deployment Name by which OCNRF can be identified
mysql.primary.host	Primary MYSQL Host IP or Hostname	ocnrf-mysql	Primary Mysql HostName or IP	OCNRF will connect Primary MYSQL if not available then it will connect secondary host. For MYSQL Cluster use respective IP Address or Mysql Host or Service
mysql.secondary.host	Secondary MYSQL Host IP or Hostname	ocnrf-mysql	Secondary Mysql HostName or IP	For MYSQL Cluster use respective Secondary IP Address or Mysql Host or Service
mysql.port	Port of MYSQL Database	3306	Port of MySQL Database	
mysql.database	MSQL Database for OCNRF	nrfdb	Name of OCNRF Database	
mysql.username	OCNRF MYSQL UserName		User Name of OCNRF Database	User Name of OCNRF Database decided by DBA while creating User in Mysql Cluster

Table 2-1 (Cont.) Global Parameters

Parameter	Description	Default value	Range or Possible Values (If applicable)	Notes
mysql.password	OCNRF MYSQL Password		Password of OCNRF Database	Password of OCNRF Database decided by DBA while creating User in Mysql Cluster
endpoint	OCNRF END Point Name	ocnrf-endpoint	Service Name for OCNRF API-GW	
endpointPort	OCNRF END Point Port	80	Port for OCNRF API-GW	This parameter will be used as OCNRF end point port.
ambassadorId	Ambassador Id to uniquely identify the Ambassador instance in cluster	ocnrf-endpoint	NAME-endpoint	Format:- NAME-endpoint
jaeger.service.name	Jaegar Service Name installed in CNE	occne-tracer-jaeger-collector.occne-infra	Service Name of Jaegar for Tracing from OCNRF API-GW	
jaeger.service.port	Jaegar Service Port installed in CNE	9411	Port of Jaegar for Tracing from OCNRF API-GW	

NF Registration Micro service (`nfregistration`)

Table 2-2 NF Registration

Parameter	Description	Default value	Range or Possible Values (If applicable)	Notes
image.repository	Full Image Path	ocnrf/ocnrf-nfregistration	Full image path of image	
image.tag	Tag of Image	1.2.0	Tag of image in docker repository	
image.pullPolicy	This setting will tell if image need to be pulled or not	IfNotPresent	Possible Values - Always IfNotPresent Never	
log.level	Logging level	INFO	INFO, DEBUG, FATAL, ERROR, WARN	Logging level

NF Subscription Micro service (`nfsubscription`)

Table 2-3 NF Subscription

Parameter	Description	Default value	Range or Possible Values (If applicable)	Notes
`image.repository`	Full Image Path	ocnrf/ ocnrf-nfsubscription	Full image path of image	
`image.tag`	Tag of Image	1.2.0	Tag of image in docker repository	
`image.pullPolicy`	This setting will tell if image need to be pulled or not	IfNotPresent	Possible Values: `Always`, `IfNotPresent`, `Never`	
`http.proxy.host`	Hostname of Proxy for HTTP		Not applicable	Proxy host for Notification. It is a valid FQDN or IP address of a host with http proxy function ability for routing the notification messages originating from OCNRF.
`http.proxy.port`	Port of Proxy for HTTP	80	Not applicable	Proxy port for Notification. It is a valid port of a host with http proxy function ability for routing the notification messages originating from OCNRF.
`log.level`	Logging level	WARN	INFO, DEBUG, FATAL, ERROR, WARN	

OCNRF Auditor Micro service (`nrfauditor`)

Table 2-4 OCNRF Auditor

Parameter	Description	Default value	Range or Possible Values (If applicable)
`image.repository`	Full Image Path	ocnrf/ocnrf-nrfauditor	Full image path of image
`image.tag`	Tag of Image	1.2.0	Tag of image in docker repository
`image.pullPolicy`	This setting indicates if the image needs to be pulled or not	IfNotPresent	Possible Values: `Always`, `IfNotPresent`, `Never`
`log.level`	Logging level	WARN	INFO, DEBUG, FATAL, ERROR, WARN

OCNRF APIGW Ambassador Micro service (`ambassador`)

Table 2-5 Ambassador

Parameter	Description	Default value	Range or Possible Values (If applicable)	Notes
`fullnameOverride`	Full Name of Microservice with Helm Release name	ocnrf-endpoint	Name identification for Ambassador service	Format:- NAME-endpoint
`ambassador.id`	Ambassador Id to uniquely identify the Ambassador instance in cluster	ocnrf-endpoint	NAME-endpoint	There can be multiple Ambassador installed in same cluster for different NFs. So, this is unique identification for it.
`image.repository`	Full Image Path	ocnrf/ocnrf-endpoint	Full image path of image	
`image.tag`	Tag of Image	0.50.3	Image Tag	
`image.pullPolicy`	This setting will tell if image need to be pulled or not	IfNotPresent	Possible Values: `Always`, `IfNotPresent`, `Never`	
`service.annotations.getambassador.io/config.ambassador_id`	Ambassador Id to uniquely identify the Ambassador instance in cluster	ocnrf-endpoint	NAME-endpoint	There can be multiple Ambassador installed in same cluster for different NFs. So, this is unique identification for it. Format: `NAME-endpoint`
`service.annotations.metallb.universe.tf/address-pool`	Address Pool for Metallb	signaling	As defined by operator	
`podAnnotations.prometheus.io/scrape`	Prometheus related setting	true	Possible Values - true false	
`podAnnotations.prometheus.io/port`	Prometheus related setting	9102		
`prometheusExporter.enabled`	Prometheus related setting	true	Possible Values - true false	

NF Discovery Micro service (`nfdiscovery`)

Table 2-6 NF Discovery

Parameter	Parameter	Description	Range or Possible Values (If applicable)
`image.repository`	Full Image Path	ocnrf/ocnrf-nfdiscovery	Full image path of image

Table 2-6 (Cont.) NF Discovery

Parameter	Parameter	Description	Range or Possible Values (If applicable)
`image.tag`	Tag of Image	1.2.0	Tag of image in docker repository
`image.pullPolicy`	This setting indicates if the image needs to be pulled or not	IfNotPresent	Possible Values: `Always, IfNotPresent, Never`
`log.level`	Logging level	WARN	INFO, DEBUG, FATAL, ERROR, WARN

Table 2-7 OCNRF Configuration

Parameter	Parameter	Description	Range or Possible Values (If applicable)
`image.pullPolicy`	This setting will tell if image need to be pulled or not	IfNotPresent	Possible Values: `Always, IfNotPresent, Never`
`image.repository`	Full Image Path	ocnrf/nrfconfiguration	Full image path of image
`image.tag`	Tag of Image	1.2.0	Tag of image in docker repository
`log.level`	Logging level	WARN	INFO, DEBUG, FATAL, ERROR, WARN
`service.configServiceNetworkName`	Address Pool for Metallb	`metallb.universe.tf/address-pool : oam`	As defined by operator

3
OCNRF Installation

OCNRF Pre-requisites

This section includes information about the required pre-requisites before initiating OCNRF Installation.

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:

Software	Version
Kubernetes	v1.12.5
HELM	v2.11.0

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

Software	Chart Version	Notes
elasticsearch	1.21.1	Needed for Logging Area
elastic-curator	1.2.1	Needed for Logging Area
elastic-exporter	1.1.2	Needed for Logging Area
logs	2.0.7	Needed for Logging Area
kibana	1.5.2	Needed for Logging Area
grafana	2.2.0	Needed for Metrics Area
prometheus	8.8.0	Needed for Metrics Area
prometheus-node-exporter	1.3.0	Needed for Metrics Area
metallb	0.8.4	Needed for External IP
metrics-server	2.4.0	Needed for Metric Server
tracer	0.8.3	Needed for 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

There are some requirements for the client machine where the deployment commands need to be executed:

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

OCNRF Installation Preparation

The following procedure describes the steps to download the OCNRF Images and Helm files from OSDC.

For more information about configuring docker image and registry, see chapter *OCCNE Docker Image Registry Configuration* in *OCCNE Installation Guide 1.0*.

Table 3-1 OCNRF Installation Preparation

Step	Procedure	Description

1
☐
	Download the OCNRF package file	Customers are required to download the OCNRF package file from Oracle Software Delivery Cloud (OSDC). Package is named as follows:
		`<nfname>-pkg-<marketing-release-number>.tgz`
		`For example: ocnrf-pkg-1.2.0.0.0.tgz`

2 ☐	Untar the OCNRF Package File	Untar the OCNRF package: `tar -xvf <<nfname>-pkg-<marketing-release-number>>.tgz` This command results into `<<nfname>-pkg-<marketing-release-number>>` directory. The directory consists of following: 1. OCNRF Docker Images File: `ocnrf-images-1.2.0.tar` 2. OCNRF Helm File `ocnrf-1.2.0.tgz` 3. Readme txt file `Readme.txt` (Contains cksum and md5sum of tarballs)
3 ☐	Verify the checksums	Verify the checksums of tarballs mentioned in `Readme.txt`.

Table 3-1 (Cont.) OCNRF Installation Preparation

Step	Procedure	Description
4 ☐	Load the tarball to system	Execute the following command to load the images to the customer's local registry: `docker load --input ocnrf-images-1.2.0.tar`
5 ☐	Check if all the images are loaded	Execute the following command to check: `docker images` Refer the below table **OCNRF Images** for the list of images.

6 ☐	Push docker images to docker registry	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>`

7 ☐	Untar Helm Files	Untar the helm files: `tar -xvzf ocnrf-1.2.0.tgz`
8 ☐	Download the Network Repository Function (NRF) Custom Template ZIP file	Download the **Network Repository Function (NRF) Custom Template** ZIP file from OHC: • Go to the URL, docs.oracle.com • Navigate to **Industries->Communications->Diameter Signaling Router->Cloud Native Network Elements** • Click the **Network Repository Function (NRF) Custom Template** link to download the zip file. • Unzip the template to get `ocnrf-custom-configTemplates-1.2.0.0.0` file that contains the following: – `NrfAlertrules.yaml`: This file is used for `prometheus`. – `NrfDashboard.json`: This file is used by `grafana`. – `ocnrf-custom-values-1.2.0.yaml`: This file is used during installation.

OCNRF Images

Following are the OCNRF images:

Table 3-2 OCNRF Images

Services	Image	Tag
NFRegistration	`ocnrf-nfregistration`	1.2.0
NFSubscription	`ocnrf-nfsubscription`	1.2.0
NFDiscovery	`ocnrf-nfdiscovery`	1.2.0
API-GW - ocnrf-endpoint	`ocnrf-endpoint`	1.2.0
NRF Auditor	`ocnrf-nrfauditor`	1.2.0
NRF Configuration	`ocnrf-nrfconfiguration`	1.2.0

OCNRF Deployment

This chapter contains information about the OCNRF Deployment in Cloud Native Environment.

Pre-installation Procedure

Create Database User/Group

The OCNRF uses a MySQL database to store the configuration and run time data.

The OCNRF deployment using MySQL NDB cluster requires the database administrator to create a user in the MYSQL DB, and to provide the user with the necessary permissions to access the tables in the NDB cluster.

> **Note:**
>
> This procedure must be performed only once before initiating the OCNRF deployment.

1. Login to the server where the ssh keys are stored and SQL nodes are accessible.
2. Connect to the SQL nodes.
3. Login to the Database as a root user.
4. Create a user and assign it to a group having necessary permission to access the tables on all the SQL nodes:
   Create User with permission to access the tables on all the SQL nodes present in the NDB cluster, by executing:

   ```
   CREATE USER '****'@'%' IDENTIFIED BY '****';
   DROP DATABASE if exists nrfdb;
   CREATE DATABASE nrfdb CHARACTER SET utf8;
   GRANT SELECT, INSERT, CREATE, ALTER, DROP, LOCK TABLES, CREATE TEMPORARY
   TABLES, DELETE, UPDATE,
   EXECUTE ON nrfdb.* TO '****'@'%';
   USE nrfdb;
   ```

   > **Note:**
   >
   > - Check if the user already exists, if yes then first drop the existing user and create a new one.
   > - The <username> and <password> is created by the Database Administrator.

5. In case OCNRF 1.2 is installed on a system having MYSQL cluster that was used in OCNRF 1.0 or OCNRF 1.1, the existing tables must be dropped before creating again for OCNRF 1.2, by executing the following command on one of the NDB SQL Nodes:

   ```
   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`;
   ```

6. Create OCNRF tables on one of the SQL Nodes by executing:

```
CREATE TABLE IF NOT EXISTS `NfInstances` (
  `nfInstanceId` VARCHAR(36) NOT NULL,
  `nfType` VARCHAR(30) NOT NULL,
  `nfStatus` ENUM('REGISTERED', 'SUSPENDED', 'UNDISCOVERABLE') NOT NULL,
  `doc` JSON NOT NULL,
  `creationTimestamp` DATETIME NOT NULL,
  `lastUpdateTimestamp` DATETIME NOT NULL,
  primary key (`nfInstanceId`)
)ENGINE=NDBCLUSTER DEFAULT CHARSET=utf8;

CREATE TABLE IF NOT EXISTS `NfStatusMonitor` (
  `nfInstanceId` VARCHAR(36) NOT NULL,
  `lastHbTimestamp` DATETIME,
  `lastNotifLoad` INTEGER,
  `nfLoad` INTEGER,
  `heartBeatTimer` INTEGER,
  `nfStatus` ENUM('REGISTERED', 'SUSPENDED','UNDISCOVERABLE') NOT NULL,
  `numHbMissed` INTEGER NOT NULL,
  `numHbMissedBeforeAudit` INTEGER NOT NULL,
  `suspendedTimestamp` DATETIME,
  `creationTimestamp` DATETIME NOT NULL,
  `lastUpdateTimestamp` DATETIME NOT NULL,
  primary key (`nfInstanceId`)
)ENGINE=NDBCLUSTER DEFAULT CHARSET=utf8;

CREATE TABLE IF NOT EXISTS `NfSubscriptions`(
  `subscriptionId` VARCHAR(36) NOT NULL,
  `nfStatusNotificationUri` VARCHAR(256) NOT NULL,
  `doc` JSON NOT NULL,
  `validityTime` DATETIME NOT NULL,
  `creationTimestamp` DATETIME NOT NULL,
  `lastUpdateTimestamp` DATETIME NOT NULL,
  primary key (`subscriptionId`)
)ENGINE=NDBCLUSTER DEFAULT CHARSET=utf8;

CREATE TABLE IF NOT EXISTS `NfScreening` (
  `nfScreeningRulesListType` ENUM('NF_FQDN', 'NF_IP_ENDPOINT',
'CALLBACK_URI', 'PLMN_ID', 'NF_TYPE_REGISTER') NOT NULL,
  `nfScreeningType` ENUM('BLACKLIST', 'WHITELIST') NOT NULL,
  `nfScreeningRulesListStatus` ENUM('ENABLED', 'DISABLED') NOT NULL DEFAULT
'DISABLED',
  `nfScreeningData` JSON NOT NULL,
  `creationTimestamp` DATETIME NOT NULL,
  `lastUpdateTimestamp` DATETIME NOT NULL,
  primary key (`nfScreeningRulesListType`)
)ENGINE=NDBCLUSTER DEFAULT CHARSET=utf8;

CREATE TABLE IF NOT EXISTS `NrfSystemOptions` (
  `id` VARCHAR(36) UNIQUE NOT NULL,
  `nfScreeningFeatureStatus` ENUM('ENABLED','DISABLED') NOT NULL,
  `nfScreeningFailureHttpCode` SMALLINT NOT NULL,
  `nfHeartBeatTimer` SMALLINT NOT NULL,
  `nrfPlmnList` JSON NOT NULL,
  `nfNotifyLoadThreshold` SMALLINT NOT NULL,
  `nrfSupportForProfileChangesInResponse` BOOLEAN NOT NULL,
  `subscriptionValidityDuration` INTEGER NOT NULL,
  `nrfSupportForProfileChangesInNotification` BOOLEAN NOT NULL,
  `nfProfileSuspendDuration` INTEGER NOT NULL,
  `nfHearbeatMissAllowed` SMALLINT NOT NULL,
```

```
  `discoveryValidityPeriod` INTEGER NOT NULL,
  `profilesCountInDiscoveryResponse` SMALLINT,
  `discoveryResultLoadThreshold` SMALLINT,
  `creationTimestamp` DATETIME NOT NULL,
  `lastUpdateTimestamp` DATETIME NOT NULL,
  primary key (`id`)
)ENGINE=NDBCLUSTER DEFAULT CHARSET=utf8;
```

7. Exit from database and logout from SQL node.

OCNRF Deployment on Kubernetes

> **Note:**
>
> By default, the namespace and helm release for OCNRF is `ocnrf`.

Table 3-3 OCNRF Deployment

Step #	Procedure	Description
1 ☐	Create customized `ocnrf-custom-values-1.2.0.yaml` file	Create the customized `ocnrf-custom-values-1.2.0.yaml` with the required input parameters. To configure the parameters, see section OCNRF Configuration. or, The `ocnrf-custom-values-1.2.0.yaml` template can be downloaded from OHC. Download the Network Repository Function (NRF) Custom Template. Unzip the `ocnrf-custom-configTemplates-1.2.0.0.0.zip` to get `ocnrf-custom-values-1.2.0.yaml` file.
2 ☐	Go to the unzipped OCNRF package	Go to the unzipped OCNRF package in the following directory: `cd ocnrf-pkg-1.2.0.0.0`
3 ☐	Deploy OCNRF	Execute the following command: `helm install ocnrf/ --name <helm-release> --namespace <k8s namespace> -f <ocnrf_customized_values.yaml>` For example: `helm install ocnrf/ --name ocnrf --namespace ocnrf -f ocnrf-custom-values-1.2.0.yaml`
4 ☐	Check status of the deployment	Execute the following command: `helm status <helm-release>` For example: `helm status ocnrf`

5 ☐	Check status of the services	Execute the following command:
		`kubectl get services -n <k8s namespace>`
		For example:
		`kubectl get services -n ocnrf`
		Note: If `metallb` is used, `EXTERNAL-IP` is assigned to `ocnrf-endpoint`.

Table 3-3 (Cont.) OCNRF Deployment

Step #	Procedure	Description

6
☐
	Check status of the pods	Execute the following command:
		`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-endpoint-6cb48bdc57-s6vws 2/2 Running 0 18h`
		`ocnrf-mysql-5d695b599f-5gwhh 1/1 Running 0 18h`
		`ocnrf-nfdiscovery-6c4b69c97f-sc84n 1/1 Running 0 18h`
		`ocnrf-nfregistration-6bcf5d84b7-qfrjm 1/1 Running 0 18h`
		`ocnrf-nfsubscription-67c6cc8fff-gtldj 1/1 Running 0 18h`
		`ocnrf-nrfauditor-77d57949c7-gx2p2 1/1 Running 0 18h`

Table 3-4 Parameters and Definitions for OCNRF Installation

Parameters	Definitions
`<helm-release>`	It is a name provided by the user to identify the helm deployment
`<k8s namespace>`	It is a name provided by the user to identify the kubernetes namespace of the OCNRF. All the OCNRF micro services are deployed in this kubernetes namespace.

4
OCNRF Upgrade

This section includes information about upgrading an existing NRF deployment.

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-1.2.0.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**

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-1.2.0.yaml
```

To check the status of the upgrade, execute:

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

For example: `helm status ocnrf`

> **Caution:**
>
> Upgrading OCNRF within same release supports only configuration changes.

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
OCNRF Uninstallation

Deleting the OCNRF deployment

To completely delete or remove the OCNRF deployment, execute:

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

For example:

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

Delete kubernetes namespace

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

For example:

```
kubectl delete namespace ocnrf
```