Oracle® Communications

Network Exposure Function (NEF) Cloud Native Installation and Upgrade Guide

Release 1.2.0
F24459-02
May 2020

Oracle Communications Network Exposure Function (NEF) Cloud Native Installation and Upgrade Guide, Release 1.2.0

F24459-02

Contents

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

There are no new features/updates performed in the guide for this release.

1 Introduction

The Network Exposure Function (NEF) is a functional element that supports the following functionalities:

- Securely exposes network capabilities and events provided by 3GPP Network Functions to AF.
- Provides a means for the AF to securely provide information to 3GPP network and may authenticate, authorize, and assist in throttling the AF.
- Translates the information received from the AF to the one sent to internal 3GPP NFs, and vice versa.
- Supports to expose information (collected from other 3GPP NFs) to the AF.
- Supports a PFD function that allows the AF to provision PFD(s) and may store and retrieve PFD(s) in the UDR. The NEF further provisions PFD(s) to the SMF.

A specific NEF instance may support one or more of the functionalities described above and consequently an individual NEF may support a subset of the APIs specified for capability exposure.

References

Refer to the following documents for more information about 5G cloud native network exposure function.

- CNE Installation Document

Acronyms and Terminologies

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

Table 1-1 Acronyms and Terminologies

Field	Description
3GPP	3rd Generation Partnership Project
5GC	5G Core Network
5GS	5G System
AF	Application Function
AUSF	Authentication Server Function
BSF	Binding Support Function
CHF	Charging Function
CNE	Cloud Native Environment
EFK stack	Elasticsearch, Fluentd, and Kibana stack
FQDN	Fully Qualified Domain Name

Table 1-1 (Cont.) Acronyms and Terminologies

Field	Description
GPSI	Generic Public Subscription Identifier
IWF	InterWorking and Mediation Function
K8s	Kubernetes
NEF	Network Exposure Function
NF	Network Function
NRF	Network Repository Function
NSSF	Network Slice Selection Function
PCF	Policy Control Function
PFD	Packet Flow Description
QFI	QoS Flow Identifier
QoE	Quality of Experience
SBA	Service Based Architecture
SBI	Service Based Interface
SCP	Service Communication Proxy
SEPP	Security Edge Protection Proxy
SMF	Session Management Function
SUPI	Subscription Permanent Identifier
UDR	Unified Data Repository
UDSF	Unstructured Data Storage Function

2
Installing Network Exposure Function

This section provides instructions for installing Network Exposure Function.

Pre-requisites

Following are pre-requisites user must have before installing Network Exposure Function.

Software	Version
Kubernetes	v1.12.5
HELM	v2.11.0
MySQL	5.7 or later

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 Logging Area
prometheus	8.8.0	Needed for Logging 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 Metrics Server
tracer	0.8.3	Needed for Tracing Area

> **Note:**
>
> In case any of the above services are needed and the respective software is not installed in CNE. Please install software before proceeding.

> **Note:**
>
> If you are using NRF, install it before proceeding with the NEF installation.

Network Access

The Kubernetes cluster hosts must have network access to:

- quay.io/datawire/ambassador docker image repository
- Local helm repository where the Oracle Communications Network Exposure Function helm charts are available. The helm charts are available in tar ball with images.
- Local docker image repository where the Oracle Communications Network Exposure Function images are available. The tar ball has the images which are posted to your repository.

Laptop/Desktop Client Software

Following are the requirements for the laptop/desktop where the deployment commands shall be executed:

- 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 namespace in the Kubernetes cluster.
- Helm client installed with the **push** plugin. The environment should be configured so that the **`helm install`** command deploys the software in the Kubernetes cluster.

Browser Support

It is recommend to use Firefox browser to access Kubernetes dashboard. The Configuration Management GUI page is accessed from different browsers.

Server or space requirements

For server and space requirements, refer to *Oracle Communications Cloud Native Environment Installation Guide*.

Installation Sequence

This section provides the order in which you shall perform the NEF installation.

1. Create a MySQl database account. See Creating Database Account on MySQL Database
2. Download NEF package files and load them to the system. See Installation Preparation.
3. Prepare all variables for Helm install command. See Table 2-1
4. NEF Deployment using Helm command. See Deploying Network Exposure Function
5. Verify NEF Deployment. See Verifying Installation.
6. Configure NEF. See #unique_20.

Creating Database Account on MySQL Database

Create a new database account on MySQL database by executing following command:

```
CREATE USER 'nefusr'@'%'IDENTIFIED BY 'nefpasswd';
GRANT ALL PRIVILEGES ON *.* TO 'nefusr'@'%';
```

Login to MySQL console as a new user created above:

```
mysql -h<MYSQL_HOST> -u <USERNAME> -p <PASSWORD>
```

At first login to MySQL console via new user created above,

```
mysql -h<MYSQL_HOST> -u nefusr -p nefpasswd
```

Execute the below script to initialize Network Exposure Function databases.

```
CREATE DATABASE IF NOT EXISTS `ocpm_nef_me`;
CREATE TABLE IF NOT EXISTS `ocpm_nef_me`.`mesubscription` (
  `id` int(11) NOT NULL,
  `me_subscription` json DEFAULT NULL,
  `tltrid` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE TABLE IF NOT EXISTS `ocpm_nef_me`. `hibernate_sequence` (
  `next_val` bigint(20) DEFAULT NULL
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
insert into `ocpm_nef_me`. `hibernate_sequence` values (0);

CREATE TABLE IF NOT EXISTS `ocpm_nef_me`.`me_charging` (
  `charging_id` varchar(128) NOT NULL,
  `charging_request_data` json DEFAULT NULL,
  PRIMARY KEY (`charging_id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE DATABASE IF NOT EXISTS `ocpm_capif_core`;
CREATE TABLE IF NOT EXISTS `ocpm_capif_core`.`aef_api_profile` (
  `apf_id` varchar(255) NOT NULL,
```

```
  `api_id` varchar(255) NOT NULL,
  `aef_profiles` json DEFAULT NULL,
  `api_name` varchar(255) DEFAULT NULL,
  `description` varchar(255) DEFAULT NULL,
  `supported_features` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`apf_id`,`api_id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE TABLE IF NOT EXISTS `ocpm_capif_core`. `invoker_profile` (
  `invoker_id` varchar(255) NOT NULL,
  `api_invoker_certificate` varchar(255) DEFAULT NULL,
  `api_invoker_public_key` varchar(255) DEFAULT NULL,
  `notification_destination` varchar(255) DEFAULT NULL,
  `onboarding_secret` varchar(255) DEFAULT NULL,
  `request_test_notification` bit(1) DEFAULT NULL,
  `request_websocket_uri` bit(1) DEFAULT NULL,
  `websocket_uri` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`invoker_id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;



CREATE DATABASE IF NOT EXISTS `ocpm_nef_asqos`;
CREATE TABLE IF NOT EXISTS `ocpm_nef_asqos`.`app_session_id_to_sub_id` (
  `app_session_id` varchar(128) NOT NULL,
  `subsription_id` varchar(128) DEFAULT NULL,
  PRIMARY KEY (`app_session_id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

```
CREATE TABLE IF NOT EXISTS `ocpm_nef_asqos`.`as_qos_subscription` (
  `subscription_id` varchar(128) NOT NULL,
  `app_sessionuri` varchar(255) DEFAULT NULL,
  `as_qos_subscription` json DEFAULT NULL,
  `scs_as_id` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`subscription_id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE TABLE IF NOT EXISTS `ocpm_nef_asqos`.`scs_as_id_subscription` (
  `scs_as_id` varchar(128) NOT NULL,
  `sub_id_array` json DEFAULT NULL,
  PRIMARY KEY (`scs_as_id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE TABLE IF NOT EXISTS `ocpm_nef_asqos`.`asqos_charging` (
  `charging_id` varchar(128) NOT NULL,
  `charging_request_data` json DEFAULT NULL,
  PRIMARY KEY (`charging_id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE DATABASE IF NOT EXISTS `ocpm_nef_ti`;
CREATE TABLE IF NOT EXISTS `ocpm_nef_ti`.`app_session_id_to_sub_id` (
  `app_session_id` varchar(128) NOT NULL,
  `subsription_id` varchar(128) DEFAULT NULL,
  PRIMARY KEY (`app_session_id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

```
CREATE TABLE IF NOT EXISTS `ocpm_nef_ti`.`tisubscription` (
  `subscription_id` varchar(128) NOT NULL,
  `app_sessionuri` varchar(255) DEFAULT NULL,
  `scs_as_id` varchar(255) DEFAULT NULL,
  `traffic_influ_sub` json DEFAULT NULL,
  PRIMARY KEY (`subscription_id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE TABLE IF NOT EXISTS `ocpm_nef_ti`.`scs_as_id_subscription` (
  `scs_as_id` varchar(128) NOT NULL,
  `sub_id_array` json DEFAULT NULL,
  PRIMARY KEY (`scs_as_id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE TABLE IF NOT EXISTS `ocpm_nef_ti`.`ti_charging` (
  `charging_id` varchar(128) NOT NULL,
  `charging_request_data` json DEFAULT NULL,
  PRIMARY KEY (`charging_id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE DATABASE IF NOT EXISTS ocpm_nef_ti;
CREATE TABLE IF NOT EXISTS ocpm_nef_ti.ti_data(
scs_as_id varchar(255) NOT NULL,
subscription_id varchar(128) NOT NULL,
traffic_influ_sub json DEFAULT NULL,
af_trans_id varchar(128) UNIQUE,
af_notif_uri varchar(255) DEFAULT NULL,
```

```
influence_id varchar(128) NOT NULL UNIQUE,

supi varchar(128) DEFAULT NULL,

inter_group_id varchar(128) DEFAULT NULL,

PRIMARY KEY (subscription_id)

) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;


CREATE DATABASE IF NOT EXISTS ocpm_nef_bdt;

CREATE TABLE IF NOT EXISTS ocpm_nef_bdt.nef_bdt

(

scs_as_id varchar(150) NOT NULL,

subscription_id varchar(128) NOT NULL,

bdt_reference_id varchar(255) NOT NULL,

subscription_expiry_ts TIMESTAMP NOT NULL,

bdt JSON NOT NULL,

resource_url varchar(255) NOT NULL,

notification_url varchar(255),

PRIMARY KEY (subscription_id)

)DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;



CREATE DATABASE IF NOT EXISTS `ocpm_config_server_nef`;


CREATE TABLE IF NOT EXISTS `ocpm_config_server_nef`.`topic_info` (

  `id` bigint(20) NOT NULL AUTO_INCREMENT,

  `description` varchar(255) COLLATE utf8_unicode_ci DEFAULT 'Default
Topics.',

  `name` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,

  `modify_date` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
```

```
  `version` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `UK_gd6b0a6mdpxc55qbibre2cldc` (`name`)
) AUTO_INCREMENT=3 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

CREATE TABLE IF NOT EXISTS `ocpm_config_server_nef`.`configuration_item` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `cfg_key` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `md5sum` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `cfg_value` mediumtext COLLATE utf8_unicode_ci,
  `version` int(11) NOT NULL,
  `topic_info_id` bigint(20) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `FKdue8drxn6acrdt63iacireky1` (`topic_info_id`)
) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;



insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('policy', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('policySchema', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('policyElement', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('policyParam', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('policygui', 1);


insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.global.cfg', 1);
```

```
insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.smservice.cfg', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.public.sessionrule', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.public.sessionruleprofile', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.public.authorizeddefaultqos', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.public.pccrule', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.public.qosdata', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.public.chargingdata', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.public.pccruleprofile', 1);



insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('amservice.system', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('public.matchlist', 1);



insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pe.serviceTag', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pe.policyTag', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pe.logLevel', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.amservice.app', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('nrfclient.cfg', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('NRF.UDR', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('NRF.BSF', 1);
```

```
insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.userservice.cfg', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('NRF.CHF', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.chfservice', 1);

insert into `ocpm_config_server_nef`.`topic_info` (name, version) values
('pcf.paservice.cfg', 1);
```

Installation Preparation

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

1. Download the NEF package file from Oracle Software Delivery Cloud (OSDC) at: http://edelivery.oracle.com. Package is named as follows:
   `<nfname>-pkg-<marketing-release-number>.tgz`

   For example, `ocnef-pkg-1.2.0.0.0.tgz`
2. Untar the NEF Package File.
   `tar -xvf <<nfname>-pkg-<marketing-releasenumber>>.tgz`

   This command results into `<<nfname>-pkg-<marketingrelease- number>>` directory.

   The directory consists of following:

   - **NEF Docker Images File:**
     `ocnef-images-1.2.0.tar`
   - **Helm File:**
     `ocnef-1.2.0.tgz`
   - **Readme txt File:**
     `Readme.txt` (Contains cksum and md5sum of tarballs)
3. Verify the checksums of tarballs mentioned in `Readme.txt`.

Deploying Network Exposure Function

> **Note:**
>
> The Network Exposure Function requires a MySQL database to store the configuration and run time data.

To deploy the NEF:

1. Download the file, **ocnef-pkg-1.2.0.0.0.tgz**.
2. Untar **ocnef-pkg-1.2.0.0.0.tgz**.

3. Untar displays the following files:

```
ocnef-pkg-1.2.0.0.0.tgz
|_ _ _ _ _ _ ocnef-1.2.0.tgz (helm chart)
|_ _ _ _ _ _ ocnef-images-1.2.0.tar (docker images)
|_ _ _ __ _ Readme.txt (Contains cksum and md5sum of tarballs)
```

4. Check the checksums of tarballs mentioned in the `Readme.txt` file.
5. After you load the tarballs to docker images, if required, re-tag it according to your specific repository.
   - Run the following command to load **`ocnef-images-1.2.0.tar`** to docker and push imported docker images to your's docker image registry.

```
docker load --input /<IMAGE_PATH>/ocnef-images-1.2.0.tar

docker tag ocpcf/nef_asqos:1.2.0 <customer repo>/nef_asqos:1.2.0
docker push <customer repo>/nef_asqos:<IMAGE_TAG>

* Repeat above tag and push commands for ALL imported docker images
as listed in the Table 2-1
```

> **Note:**
>
> User may need to configure docker certificate to access customer docker image repository via HTTPS. Configure the certificate before executing the docker push command or the command may fail to execute.

Table 2-1 provides the details of the docker images file name:

Table 2-1 Docker Images

Service Name	Docker Image Name	Service Category
NEF CM Service	ocnef-nefcm	GUI
NEF Asqos Service	ocnef-nefasqos	NEF
NEF ME Service	ocnef-nefme	NEF
NEF Capif Service	ocnef-nefcapif	NEF
NEF TI Service	ocnef-nefti	NEF
NEF Config Service	ocnef-nefconfigserver	GUI
NEF BDT Service	ocnef-nefbdt	NEF

Table 2-2 provides the information about the modules:

Table 2-2 Module Descriptions

Module Name	Description
GUI	GUI services for NEF

Table 2-2 (Cont.) Module Descriptions

Module Name	Description
NEF	Define NEF services

6. Execute the following command:

> **Note:**
>
> It is mandatory to run the below command under helm chart folder as the last line of the command, **./<HELM_CHART_NAME_WITH_EXTENSION>** specifies that helm chart path is current working path. To run the below command in another server, copy the helm chart file to it first.

```
helm install --namespace=nefsvc --name=ocnef \
set
global.envMysqlPrimaryHost=<MYSQL_PRIMARY_HOST>,global.nvMysqlSecondar
yHost=<MYSQL_SECONDARY_HOST> \
--set global.envMysqlUser=nefusr,global.envMysqlPassword=nefpasswd \
--set
global.envJaegerAgentHost=<JAEGER_SERVICE>.<JAEGER_SERVICE_NAMESPACE>,
global.nvJaegerAgentPort=<JAEGER_SERVICE_PORT> \
--set
global.imageTag=<IMAGE_TAG>,global.dockerRegistry=<CUSTOMER_REPO> \
--set nef-ti.deploymentNefTiService.envUdmBaseUrl=<UDM_BASE_URI>,nef-
ti.deploymentNefTiService.envUdrBaseUrl=<UDR_BASE_URI> \
--set nef-ti.deploymentNefTiService.envBsfBaseUrl=<BSF_BASE_URI>,nef-
ti.deploymentNefTiService.envPcfBaseUrl=<PCF_BASE_URI> \
--set nef-bdt.deploymentNefBdtService.envPcfPort=<PCF_PORT>,nef-
bdt.deploymentNefBdtService.envPcfHost=<PCF_HOST> \
--set nef-bdt.deploymentNefBdtService.envUdmHost=<UDM_HOST>,nef-
bdt.deploymentNefBdtService.envUdmPort=<UDM_PORT> \
./<HELM_CHART_NAME_WITH_EXTENSION>
```

> **Note:**
>
> If Kubernetes cluster has Jaeger service running, set the **global.envJaegerAgentHost** with the value "<JAEGER_SERVICE>.<JAEGER_SERVICE_NAMESPACE>" and **envJaegerAgentPort** with the value "<JAEGER_SERVICE_PORT>". Otherwise, you should not use these variables.

Table 2-3 provides details of each variable:

Table 2-3 Variables Description

Variable	Description	Notes
<MYSQL_PRIMARY_HOST> <MYSQL_SECONDARY_HOST>	MySQL primary host name and secondary host name or IP Address	
<JAEGER_SERVICE>.<JAEGER_SERVICE_NAMESPACE>	Jaeger service and Jaeger service namespace can be found in same Kubernetes via "helm list" command	Follow the below format: **<JAEGER_AGENT_SERVICE_NAME>.<JAEGER_NAMESPACE>** Such as **occne-tracer-jaegeragent.occne-infra**, then the jaeger agent service name under jaeger deployment is **occne-tracer-jaeger-agent**.
<JAEGER_SERVICE_PORT>	Port on which Jaeger service is running.	
<IMAGE_TAG>	The image tag used in customer docker registry, it is recommend to use same image tag when pull docker image to registry. If follow above steps to push docker image to customer docker registry then the *<IMAGE_TAG>* value should be 1.2.0	Each service deployment yaml file would use global.imageTag as image tag to fetch relateddocker image per helm chart design. With the release tar file, the global image tag for all services is 1.2.0
<CUSTOMER_REPO>	The docker registry address in the customer side along with the port number if registry has port attached	If registry has port value, add port. For example, **reg-1:5000**
<UDM_BASE_URI>	Base url for UDM	Follow the below format: " http://{ hostname or ip address}:{ port }/nudm-sdm/v2/"
<UDR_BASE_URI>	Base url for UDR	Follow the below format: " http://{ hostname or ip address}:{ port }/nudr-dr/v1/"
<BSF_BASE_URL>	Base url for BSF	Follow the below format: " http://{ hostname or ip address}:{ port }/nbsf-management/v1/"
<PCF_BASE_URL>	Base url for PCF	Follow the below format: " http://{ hostname or ip address}:{ port }/npcf-policyauthorization /v1/"
<PCF_HOST>, <PCF_PORT>	hostname or ip address and port of PCF.	
<UDM_HOST>,<UDM_PORT>	hostname or ip address and port of UDM.	

Kubernetes provides the following two deployment types.

Table 2-4 Service Deployment Service Type

Service Type	Description
ClusterIP	Exposes the service on a cluster-internal IP. Choosing this value makes the service only reachable from within the cluster. This is the default ServiceType.
NodePort	Exposes the service on each Node's IP at a static port (the NodePort). A ClusterIP service, to which the NodePort service routes is automatically created. Contact the NodePort service from outside the cluster, by requesting, **`<NodeIP>:<NodePort>`** Most of the NEF services use NodePort to deploy.

Verifying Installation

To verify installation, run the following command:

```
kubectl get svc -n nefsvc
```

```
kubectl get pod -n nefsvc
```

If installation is successful, then all pods display in Running/Completed status.

Installation can also be verified via K8S dashboard. The following steps can be used to do that:

1. Under Kubernetes dashboard, click **Namespace** dropdown list and select NEF namespace
2. Click **Pods** link in the side navigation bar, and then navigate to pod list page
3. Check all pods to ensure that all of them are active and running

3
Configuring Network Exposure Function Traffic Influence

You can configure Network Exposure Function (NEF) Traffic Influence (TI) using the graphical user interface (GUI). For this, you need to provide the following mappings in GUI:

- AF-Service-Identifier into DNN and S-NSSAI combination.
- AF-Service-Identifier into list of DNAI(s) and Routing Profile ID(s).
- Geographic Zone identifier into a list of Network Area Info.

To configure mapping through GUI:

1. Configuration Management (CM) service is one which provides GUI. To access CM service, execute the following command:

   ```
   kubectl get svc ocnef-nef-cm -n nefsvc
   ```

2. Open any of the browser and use "**master node ip or hostname**" with NodePort. NodePort can be retrieved by using command mentioned in step 1.
3. Go to **NEF** ->**Configurations**.
4. You can configure the mapping by clicking on the "**AfServiceId Mapping**" and "**GeoZoneId Mapping**".

4
Upgrading Network Exposure Function

User can perform the helm upgrade command in the following scenarios.

- Update an existing parameter setting, such as global.imageTag, global.envMysqlHost.etc.,
- Add more parameters per requirement

To update Network Exposure Function (NEF) services, execute the following command and specify the upgrade parameter (new parameters or update existing parameters):

```
helm upgrade ocnef \

--set
globalenvMysqlPrimaryHost=<MYSQL_PRIMARY_HOST>,global.envMysqlSecondaryHos
t=<MYSQL_SECONDARY_HOST> \

--set global.envMysqlUser=nefusr,global.envMysqlPassword=nefpasswd \

--set
global.envJaegerAgentHost=<JAEGER_SERVICE>.<JAEGER_SERVICE_NAMESPACE>,glob
al.envJaegerAgentPort=<JAEGER_SERVICE_PORT> \

--set
global.imageTag=<IMAGE_TAG>,global.dockerRegistry=<DOCKER_REGISTRY_ADDRESS
> \

--set nef-ti.deploymentNefTiService.envUdmBaseUrl=<UDM_BASE_URI>,nef-
ti.deploymentNefTiService.envUdrBaseUrl=<UDR_BASE_URI> \

--set nef-ti.deploymentNefTiService.envBsfBaseUrl=<BSF_BASE_URI>,nef-
ti.deploymentNefTiService.envPcfBaseUrl=<PCF_BASE_URI> \

--set nef-bdt.deploymentNefBdtService.envPcfPort=<PCF_PORT>,nef-
bdt.deploymentNefBdtService.envPcfHost=<PCF_HOST> \

--set nef-bdt.deploymentNefBdtService.envUdmHost=<UDM_HOST>,nef-
bdt.deploymentNefBdtService.envUdmPort=<UDM_PORT>

./<HELM_CHART_NAME_WITH_EXTENSION>
```

> **Note:**
>
> The upgrade command is similar to install command, because, if user do not specify the same parameters for both upgrade and install, then the settings applied by install command may be lost and use default settings from **values.yaml** file for missing parameters in upgrade command.

Verifying Upgrade

Upgrade should ensure all pods under NEF namespace are updated based on the helm upgrade setting.

Execute below command:

```
kubectl get pod -n nefsvc
```

Verify the following from the output of the above command:

- All pods under NEF namespace should either be in **Running** status or in **Completed** status. If any pod with error status found, check pod's log to view the reason for error.
- For updated service per helm upgrade setting, check its RESTART output and AGE output. If the service is updated, then the restart value should be 0 and age value should be few seconds which would imply that the old pod was deleted and a new pod is brought up.

5
Uninstalling Network Exposure Function

To uninstall or completely delete the Network Exposure Function deployment, execute the following command:

```
helm delete --purge ocnef
```

```
kubectl delete namespace nefsvc
```

Verifying Uninstallation

To verify the NEF uninstallation, run the following command:

```
kubectl get namespace
```

Result: No NEF deployment should be found under the command outputs.