Oracle® Communications

Cloud Native Policy Control Function Installation and Upgrade Guide

Release 1.6.1
F30861-02
May 2020

ORACLE

Oracle Communications Cloud Native Policy Control Function Installation and Upgrade Guide, Release 1.6.1

F30861-02

Contents

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:

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

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

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

What's New in This Guide

This section introduces the documentation updates for Release 1.6.x in Oracle Communications Cloud Native Policy Control Function (PCF) Installation Guide.

Significant Documentation Updates for Release 1.6.x

For Release 1.6.x, the following sections in this guide have been updated:

- Updated Installation Procedure to support hooks.
- Added Upgrade Procedure
- Added Downgrading procedure
- Added Troubleshooting section
- Updated Software Compatibility
- Updated Docker Images table
- Updated Customization Parameters

1 Introduction

The Policy Control Function (PCF) is one of the main key component of the 5G Service Based Architecture. The PCF provides the following session management related functionality:

- Policy and charging control for a service data flows
- Protocol Data Unit (PDU) Session related policy control
- PDU Session event reporting to Application Function (AF)

The PCF interacts with Access and Mobility Management Function (AMF), Session Management Function (SMF), and AF to provide policy control rules to the Network Functions (NFs) and also interacts with User Data Repository (UDR) to get the subscriber related information for creating the rules.

The PCF supports the above functions through the following services:

- Session Management Service
- Access and Mobility Service
- Policy Authorization Service
- User Equipment (UE) Policy Service

For more information about the PCF supported services, see *Oracle Communications Cloud Native Policy Control Function User's Guide*.

References

Refer the following documents for more information about Cloud Native Policy Control Function (PCF):

- Oracle Communications Cloud Native Environment Installation Document
- Oracle Communications Cloud Native Policy Control Function (PCF) 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

Acronym	Definition
AMF	Access and Mobility Management Function
BSF	Binding Support Function
CHF	Charging Function
CM	Configuration Management

Table 1-1 (Cont.) Acronyms and Terminology

Acronym	Definition
CUSTOMER_REPO	Docker registry address including the port number, if the docker registry has an associated port.
IMAGE_TAG	Image tag from release tar file. You can use any tag number. However, make sure that you use that specific tag number while pushing docker image to the docker registry.
MCC	Mobile Country code
METALLB_ADDRESS_POOL	Address pool which configured on metallb to provide external IPs .
MNC	Mobile Network code
NRF	Network Repository Function
PCF	Policy Control Function
SAN	Storage Area Network
SMF	Session Management Function
UDR	Unified Data Repository

2
Installing Cloud Native Policy Control Function

This chapter describes how to install Cloud Native Policy Control Function (PCF) on a cloud native environment.

This chapter contains the following:

Pre-Installation Tasks

Prior to installing the PCF, perform the following tasks:

Checking the Software Requirements

The following softwares must be installed before installing Policy Control Function (PCF):

Software	Version
Kubernetes	v1.15.3
HELM	v2.14.3

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

Software	App Version	Notes
alertmanager	0.18.0	Required for Tracing
elasticsearch	7.4.0	Required for Logging
elastic-curator	2.0.2	Required for Logging
elastic-exporter	1.1.2	Required for Logging
logs	2..7.0	Required for Logging
kibana	7.4.0	Required for Logging
grafana	3.8.4	Required for Metrics
prometheus	9.2.0	Required for Metrics
prometheus-node-exporter	1.6.0	Required for Metrics
metallb	0.8.4	Required for External
metrics-server	2.5.1	Required for Metric Server
occne-snmp-notifier	0.3.0	Required for Metric Server
tracer	0.13.3	Required for Tracing

> **Note:**
>
> The above softwares are available if the PCF is deployed in the Oracle Communications Cloud Native Environment (OCCNE). If you are deploying PCF in any other environment, the above softwares must be installed before installing the PCF. To check the installed software items,
>
> ```
> helm ls
> ```
>
> Some of the systems may need to use helm command with **admin.conf** file as follows:
>
> ```
> helm --kubeconfig admin.conf
> ```

> **Note:**
>
> If you are using Network Repository Function (NRF), install it before proceeding with the PCF installation.

Checking the Environment Setup

> **Note:**
>
> This section is applicable only when the Policy Control Function (PCF) is deployed in the environment, other than OCCNE.

Network access

The Kubernetes cluster hosts must have network access to:

- Local helm repository, where the PCF helm charts are available.
  To check if the Kubernetes cluster hosts have network access to the local helm repository, execute the following command:

  ```
  helm repo update
  ```

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

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

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

  where:

  docker-repo is the IP address or host name of the repository.

  image-name is the docker image name.

  image-tag is the tag the image used for the PCF pod.

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

Client Machine Requirements

Following are the client machine requirements where the deployment commands executed:

- It should have network access to the helm repository and docker image repository.
- 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 namespace in the Kubernetes cluster.
- It should have 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.

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 enabling HTTPs on Ingress/Egress gateway the following certificates and pem files has to be created before creating secret files for keys:

- ECDSA private Key and CA signed ECDSA Certificate (if initialAlgorithm: ES256)
- RSA private key and CA signed RSA Certificate (if initialAlgorithm: RSA256)
- TrustStore password file
- KeyStore password file
- CA signed ECDSA certificate

Installation Tasks

Downloading PCF package

To download the PCF package from MOS:

1. Login to My Oracle Support with your credentials.
2. Select **Patches and Updates** tab to locate the patch.
3. In **Patch Search** window, click **Product or Family (Advanced)**.
4. Enter *Oracle Communications Cloud Native Core - 5G* in **Product** field, select *Oracle Communications Cloud Native Core Policy Control Function 1.6.1.0.0* from **Release** drop-down.
5. Click **Search**. The **Patch Advanced Search Results** displays a list of releases.
6. Click the required patch from the search results. A window opens and click **Download**.
7. Click the zip file to download the package. Package is named 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, `ocpcf-pkg-1.6.1.0.0.tgz`

Pushing the Images to Customer Docker Registry

To Push the images to customer docker resgistry:

1. Untar the PCF package file to get PCF docker image tar file.
   `tar -xvzf ReleaseName-pkg-Releasenumber.tgz`

   The directory consists of the following:

   - **PCF Docker Images File:**
     `ocpcf-images-1.6.1.tar`
   - **Helm File:**
     `ocpcf-1.6.1.tgz`
   - **Readme txt File:**
     `Readme.txt` (Contains cksum and md5sum of tarballs)
2. Load the **ocpcf-images-1.6.1.tar** file into the Docker system

   ```
   docker load --input /IMAGE_PATH/ocpcf-images-1.6.1.tar
   ```

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

   ```
   docker images
   ```

   Refer Appendix A for more information on docker images available in PCF.

4. Create a new tag for each imported image and push the image to the customer docker registry by entering this command:

```
docker tag ocpcf/pcf_smservice:1.6.1 CUSTOMER_REPO/pcf_smservice:1.6.1
docker push CUSTOMER_REPO/pcf_smservice:1.6.1

docker tag ocpcf/pcf_ueservice:1.6.1 CUSTOMER_REPO/pcf_ueservice:1.6.1
docker push CUSTOMER_REPO/pcf_ueservice:1.6.1

docker tag ocpcf/pcf-amservice:1.6.1 CUSTOMER_REPO/pcf-amservice:1.6.1
docker push CUSTOMER_REPO/pcf-amservice:1.6.1

docker tag ocpcf/pcf_userservice:1.6.1 CUSTOMER_REPO/pcf_userservice:
1.6.1
docker push CUSTOMER_REPO/pcf_userservice:1.6.1

docker tag ocpcf/ocpm_pre:1.6.1 CUSTOMER_REPO/ocpm_pre:1.6.1
docker push CUSTOMER_REPO/ocpm_pre:1.6.1

docker tag ocpcf/diam-connector:1.6.1 CUSTOMER_REPO/diam-connector:1.6.1
docker push CUSTOMER_REPO/diam-connector:1.6.1

docker tag ocpcf/diam-gateway:1.6.1 CUSTOMER_REPO/diam-gateway:1.6.1
docker push CUSTOMER_REPO/diam-gateway:1.6.1

docker tag ocpcf/ocpm_config_server:1.6.1 CUSTOMER_REPO/
ocpm_config_server:1.6.1
docker push CUSTOMER_REPO/ocpm_config_server:1.6.1

docker tag ocpcf/ocpm_cm_service:1.6.1 CUSTOMER_REPO/ocpm_cm_service:
1.6.1
docker push CUSTOMER_REPO/ocpm_cm_service:1.6.1

docker tag ocpcf/nrf-client:1.2.0 CUSTOMER_REPO/nrf-client:1.2.0
docker push CUSTOMER_REPO/nrf-client:1.2.0

docker tag ocpcf/ocpm_queryservice:1.6.1 CUSTOMER_REPO/
ocpm_queryservice:1.6.1
docker push CUSTOMER_REPO/ocpm_queryservice:1.6.1

docker tag ocpcf/audit_service:1.6.1 CUSTOMER_REPO/audit_service:1.6.1
docker push CUSTOMER_REPO/audit_service:1.6.1

docker tag ocpcf/readiness-detector:1.6.1 CUSTOMER_REPO/readiness-
detector:1.6.1
docker push CUSTOMER_REPO/readiness-detector:1.6.1

docker tag ocpcf/perf_info:1.6.1 CUSTOMER_REPO/perf_info:1.6.1
docker push CUSTOMER_REPO/perf_info:1.6.1

docker tag ocpcf/app_info:1.6.1 CUSTOMER_REPO/app_info:1.6.1
docker push CUSTOMER_REPO/app_info:1.6.1

docker tag ocpcf/policyds:1.6.1 CUSTOMER_REPO/policyds:1.6.1
docker push CUSTOMER_REPO/policyds:1.6.1
```

```
docker tag ocpcf/ldap-gateway:1.6.1 CUSTOMER_REPO/ldap-gateway:1.6.1
docker push CUSTOMER_REPO/ldap-gateway:1.6.1

docker tag ocpcf/ocingress_gateway:1.6.3 CUSTOMER_REPO/
ocingress_gateway:1.6.3
docker push CUSTOMER_REPO/ocingress_gateway:1.6.3

docker tag ocpcf/ocegress_gateway:1.6.3 CUSTOMER_REPO/ocegress_gateway:
1.6.3
docker push CUSTOMER_REPO/ocegress_gateway:1.6.3

docker tag ocpcf/configurationinit:1.1.1 CUSTOMER_REPO/
configurationinit:1.1.1
docker push CUSTOMER_REPO/configurationinit:1.1.1

docker tag ocpcf/configurationupdate:1.1.1 CUSTOMER_REPO/
configurationupdate:1.1.1
docker push CUSTOMER_REPO/configurationupdate:1.1.1
```

where:
CUSTOMER_REPO is the docker registry address having Port Number, if registry has port attached.

> **Note:**
>
> For OCCNE, copy the package to bastion server and use **localhost: 5000** as CUSTOMER_REPO to tag the images and push to bastion docker registry.

> **Note:**
>
> You may need to configure the Docker certificate before the push command to access customer registry via HTTPS, otherwise, docker push command may fail.

Installing the PCF Package

To install the PCF package:

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 an admin user and grant all the necessary permissions to the user by executing the following command:

```
CREATE USER 'username'@'%' IDENTIFIED BY 'password';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_smservice.* TO 'username'@'%';
```

```
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_amservice.* TO 'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_userservice.* TO 'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON ocpm_config_server.* TO 'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON oc5g_audit_service.* TO 'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_release.* TO 'username'@'%';
FLUSH PRIVILEGES;
```

where:
username is the username and *password* is the password for MYSQL admin user.

> **Note:**
>
> Admin user can use helm hooks to perform DDL and DML operations to perform install/upgrade/rollback or delete operations.

For Example: In the below example "pcfadminusr" is used as username, "pcfadminpasswd" is used as password and granting the necessary permissions to "pcfadminusr". In this example, default database names of micro services are used.

```
CREATE USER 'pcfadminusr'@'%' IDENTIFIED BY 'pcfadminpasswd';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_smservice.* TO 'pcfadminusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_amservice.* TO 'pcfadminusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_userservice.* TO 'pcfadminusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON ocpm_config_server.* TO 'pcfadminusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON oc5g_audit_service.* TO 'pcfadminusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_release.* TO 'pcfadminusr'@'%';
FLUSH PRIVILEGES;
```

5. Create an application user and grant all the necessary permissions to the user by executing the following command:

```
CREATE USER 'username'@'%' IDENTIFIED BY 'password';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE ON pcf_smservice.* TO
'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE ON pcf_amservice.* TO
'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE ON pcf_userservice.* TO
'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE ON ocpm_config_server.* TO
'username'@'%';
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON oc5g_audit_service.* TO
'username'@'%';
```

where:

username is the username and *password* is the password for MYSQL database user.

For Example: In the below example "pcfusr" is used as username, "pcfpasswd" is used as password and granting the necessary permissions to "pcfusr". In this example, default database names of micro services are used.

```
CREATE USER 'pcfusr'@'%' IDENTIFIED BY 'pcfpasswd';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE ON pcf_smservice.* TO
'pcfusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE ON pcf_amservice.* TO 'pcfusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE ON pcf_userservice.* TO
'pcfusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE ON ocpm_config_server.* TO
'pcfusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE ON oc5g_audit_service.* TO
'pcfusr'@'%';
```

> **Note:**
>
> The database name can be specified in the **envMysqlDatabase** parameter for respective services in the custom-value.yaml file.

It is recommended to use unique database name when there are multiple instances of PCF deployed in the network and they share the same data tier (MySQL cluster).

6. Execute the command, `show grants for` *username*, to confirm that admin user has all the permission.
7. Exit from database and logout from MYSQL node.
8. Create namespace if already does not exists by entering the command:

   ```
   kubectl create namespace release_namespace
   ```

   where:
   release_namespace is the deployment CNPCF namespace used by helm command.
9. Create a kubernetes secret for an admin user and an application user.
   To create a kubernetes secret for storing database username and password for these users:
   a. Create a yaml file with the application user's username and password with the syntax shown below:

      ```
      apiVersion: v1
      kind: Secret
      ```

```
metadata:
  name: pcf-db-pass
type: Opaque
data:
  mysql-username: <base64 encoded mysql username>
  mysql-password: <base64 encoded mysql password>
```

b. Create a yaml file with the admin user's username and password with the syntax shown below:

```
apiVersion: v1
kind: Secret
metadata:
  name: pcf-admin-db-pass
type: Opaque
data:
  mysql-username: <base64 encoded mysql username>
  mysql-password: <base64 encoded mysql password>
```

> **Note:**
>
> The values for **mysql-username** and **mysql-password** should be base64 encoded.

c. Execute the following commands to add the kubernetes secrets in a namespace:

```
kubectl create -f yaml_file_name1 -n release_namespace
kubectl create -f yaml_file_name2 -n release_namespace
```

where:
release_namespace is the deployment namespace used by the helm command.

yaml_file_name1 is a name of the yaml file that is created in step 1.

yaml_file_name2 is a name of the yaml file that is created in step 2.

10. Create the customize *ocpcf-custom-values-1.6.1.yaml* file with the required input parameters. To customize the file, see Customizing Policy Control Function.

> **Note:**
>
> The values of the parameters mentioned in the custom values yaml file overrides the defaults values specified in the helm chart. If the **envMysqlDatabase** parameter is modified, then you should modify the **configDbName** parameter with the same value.

11. **Caution:**

    Do not exit from `helm install` command manually. After running the `helm install` command, it takes some time to install all the services. In the meantime, you must not press "ctrl+c" to come out from `helm install` command. It leads to some anomalous behavior.

    a. Install PCF by using Helm2:

    ```
    helm install <helm-chart> --name <release_name> --namespace
    <release_namespace> -f <custom_file> --atomic
          --timeout 600
    ```

    b. Install PCF by using Helm3:

    ```
    helm install -f <custom_file> <release_name> <helm-chart> --
    namespace <release_namespace> --atomic --timeout
          10m
    ```

    where:
    *HELM_CHART*is the location of the helm chart extracted from ocpcf-pkg-1.6.1.tgz file

    release_name is the release name used by helm command.

    release_namespace is the deployment namespace used by helm command.

    custom_file - is the name of the custom values yaml file (including location).

    For example:

    ```
    helm install /home/cloud-user/pcf-1.6.1.tgz --name ocpcf --namespace
    ocpcf -f ocpcf-custom-values-1.6.1.yaml --atomic
    ```

    Refer Customizing Policy Control Function for the sample yaml file.

    Parameters in `helm install` command:

    - **atomic**: If this parameter is set, installation process purges chart on failure. The --wait flag will be set automatically.
    - **wait**: If this parameter is set, installation process will wait until all pods, PVCs, Services, and minimum number of pods of a deployment, StatefulSet, or ReplicaSet are in a ready state before marking the release as successful. It will wait for as long as --timeout.
    - **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.

12. You can verify the installation while running the install command by entering this commands:

    ```
    watch kubectl get jobs,pods -n release_namespace
    ```

    Press "Ctrl+C" to exit watch mode.

    ```
    helm status release_name -n release_namespace
    ```

13. Check the installation status by entering this command

    ```
    helm ls release_name
    ```

    For example:

    ```
    helm ls ocpcf
    ```

    You will see the status as **DEPLOYED** if the deployment has been done successfully.
    Execute the following command to get status of jobs and pods:

    ```
    kubectl get jobs,pods -n release_namespace
    ```

    For example:

    ```
    kubectl get pod -n ocpcf
    ```

    You will see the status as **Running** for all the nodes if the deployment has been done successfully.

3
Customizing Policy Control Function

This chapter describes how to customize the Policy Control Function (PCF) deployment in a cloud native environment.

The PCF deployment is customized by overriding the default values of various configurable parameters in the **ocpcf-custom-values-1.6.1.yaml** file.

To customize the **ocpcf-custom-values-1.6.1.yaml** file as per the required parameters:

1. Go to the Oracle Help Center (OHC) Web site: https://docs.oracle.com
2. Navigate to **Industries**->**Communications**->**Cloud Native Core->Release 2.2.0**
3. Click the **Policy Control Function (PCF) Custom Template** link to download the zip file.
4. Unzip the file to get `ocpcf-custom-configTemplates-1.6.1.0.0.0` file that contains the `ocpcf-custom-values-1.6.1.yaml`. This file is used during installation.
5. Customize the **ocpcf-custom-values-1.6.1.yaml** file.
6. Save the updated **ocpcf-custom-values-1.6.1.yaml** file in the helm chart directory.

Following is a sample **ocpcf-custom-values-1.6.1.yaml** file created based on all the parameters described in the Configurable Parameters section .

```
# Copyright 2019 (C), Oracle and/or its affiliates. All rights reserved.
# section:- global
global:
  # Docker registry name
  dockerRegistry: 'occne-bastion:5000'
  #Jaeger Hostname
  envJaegerAgentHost: 'occne-tracer-jaeger-agent.occne-infra'
  # Primary MYSQL Host IP or Hostname
  envMysqlHost: ''
  # K8s secret object name containing OCPCF MYSQL UserName and Password
  dbCredSecretName: 'pcf-db-pass'
  privilegedDbCredSecretName: 'pcf-privileged-db-pass'
  #Release DB name containing release version details
  releaseDbName: 'pcf_release'
  # -----Ingress Gateway Settings - BEGIN-----
  # If httpsEnabled is false, this Port would be HTTP/2.0 Port (unsecured)
  publicHttpSignalingPort: 80
  # If httpsEnabled is true, this Port would be HTTPS/2.0 Port (secured
SSL)
  publicHttpsSignallingPort: 443

  # Enable or disable IP Address allocation from Metallb Pool
```

```
  metalLbIpAllocationEnabled: false

  # Address Pool Annotation for Metallb
  metalLbIpAllocationAnnotation: "metallb.universe.tf/address-pool:
signaling"
  # -----Ingress Gateway Settings - END-----

  # API root of PCF that will be used in notification URLs generated by
PCF's when sending request to other producer NFs
  #If not configured then the ingress gateway service name and port will
be used as default value. ex:"https://<k8s namespace>-pcf-egress-gateway:
443"
  pcfApiRoot: ''

  nrfClientEnable: true
  deploymentNrfClientService:
    #K8s namespace of PCF
    envNfNamespace: ''
    #same as pcfApiRoot
    nfApiRoot: ''

  # Global flag to enable/disable Policy DS service
  policydsEnable: false
  # Global flag to enable/disable LDAP Gateway service
  ldapGatewayEnable: false

  diamConnectorEnable: true

am-service:
  enabled: true
  envMysqlDatabase: pcf_amservice

sm-service:
  enabled: true
  envMysqlDatabase: pcf_smservice
  envMysqlDatabaseUserService: pcf_userservice
  defaultBsfApiRoot: 'https://bsf.apigateway:8001'
  auditSmSessionTtl: 86400
  auditSmSessionMaxTtl: 172800

user-service:
  envMysqlDatabase: pcf_userservice

config-server:
  envMysqlDatabase: ocpm_config_server

queryservice:
  envMysqlDatabaseSmService: pcf_smservice
  envMysqlDatabaseUserService: pcf_userservice

audit-service:
  envMysqlDatabase: oc5g_audit_service

nrf-client:
  # This config map is for providing inputs to NRF-Client
```

```
  configmapApplicationConfig:
    # primaryNrfApiRoot - Primary NRF Hostname and Port
    # SecondaryNrfApiRoot - Secondary NRF Hostname and Port
    # retryAfterTime - Default downtime(in ISO 8601 duration format) of an
NRF detected to be unavailable.
    # nrfClientType - The NfType of the NF registering
    # nrfClientSubscribeTypes - the NFType for which the NF wants to
subscribe to the NRF.
    # appProfiles - The NfProfile of the NF to be registered with NRF.
    # enableF3 - Support for 29.510 Release 15.3
    # enableF5 - Support for 29.510 Release 15.5
    # renewalTimeBeforeExpiry - Time Period(seconds) before the
Subscription Validity time expires.
    # validityTime - The default validity time(days) for subscriptions.
    # enableSubscriptionAutoRenewal - Enable Renewal of Subscriptions
automatically.
    # acceptAdditionalAttributes - Enable additionalAttributes as part of
29.510 Release 15.5
    profile: |-
      [appcfg]
      primaryNrfApiRoot=http://nrf1-api-gateway.svc:80
      secondaryNrfApiRoot=http://nrf2-api-gateway.svc:80
      retryAfterTime=PT120S
      nrfClientType=PCF
      nrfClientSubscribeTypes=CHF,UDR,BSF
      appProfiles=[{ "nfInstanceId": "fe7d992b-0541-4c7d-ab84-
c6d70b1b0123", "nfType": "PCF", "nfStatus": "REGISTERED", "plmnList":
null, "nsiList": null, "fqdn": "ocpcf-api-gateway.ocpcf.svc",
"interPlmnFqdn": null, "ipv4Addresses": null, "ipv6Addresses": null,
"priority": null, "capacity": null, "load": 80, "locality": null,
"pcfInfo": { "dnnList": [ "internet", "volte" ], "supiRanges":
[ { "start": "12123444444", "end": "232332323323232", "pattern":
null } ] }, "customInfo": null, "recoveryTime": null, "nfServices":
[ { "serviceInstanceId": "03063893-cf9e-4f7a-9827-067f6fa9dd01",
"serviceName": "npcf-am-policy-control", "versions":
[ { "apiVersionInUri": "v1", "apiFullVersion": "1.0.0", "expiry":
null } ], "scheme": "http", "nfServiceStatus": "REGISTERED", "fqdn":
"ocpcf-api-gateway.ocpcf.svc", "interPlmnFqdn": null, "ipEndPoints": null,
"apiPrefix": null, "defaultNotificationSubscriptions": null,
"allowedPlmns": null, "allowedNfTypes": [ "AMF", "NEF" ],
"allowedNfDomains": null, "allowedNssais": null, "priority": null,
"capacity": null, "load": null, "recoveryTime": null, "supportedFeatures":
null }, { "serviceInstanceId": "03063893-cf9e-4f7a-9827-067f6fa9dd02",
"serviceName": "npcf-smpolicycontrol", "versions": [ { "apiVersionInUri":
"v1", "apiFullVersion": "1.0.0", "expiry": null } ], "scheme": "http",
"nfServiceStatus": "REGISTERED", "fqdn": "ocpcf-api-gateway.ocpcf.svc",
"interPlmnFqdn": null, "ipEndPoints": null, "apiPrefix": null,
"defaultNotificationSubscriptions": null, "allowedPlmns": null,
"allowedNfTypes": [ "SMF", "NEF", "AF" ], "allowedNfDomains": null,
"allowedNssais": null, "priority": null, "capacity": null, "load": null,
"recoveryTime": null, "supportedFeatures": null }, { "serviceInstanceId":
"03063893-cf9e-4f7a-9827-067f6fa9dd03", "serviceName": "npcf-ue-policy-
control", "versions": [ { "apiVersionInUri": "v1", "apiFullVersion":
"1.0.0", "expiry": null } ], "scheme": "http", "nfServiceStatus":
"REGISTERED", "fqdn": "ocpcf-api-gateway.ocpcf.svc", "interPlmnFqdn":
```

```
null, "ipEndPoints": null, "apiPrefix": null,
"defaultNotificationSubscriptions": null, "allowedPlmns": null,
"allowedNfTypes": [ "AMF" ], "allowedNfDomains": null, "allowedNssais":
null, "priority": null, "capacity": null, "load": null, "recoveryTime":
null, "supportedFeatures": null } ]}]
      enableF3=true
      enableF5=true
      renewalTimeBeforeExpiry=3600
      validityTime=30
      enableSubscriptionAutoRenewal=true
      acceptAdditionalAttributes=false

  nrf-client-nfdiscovery:
    cacheDiscoveryResults: false

appinfo:
  serviceAccountName: ''

perf-info:
  configmapPerformance:
    prometheus: http://prometheus-server.prometheus:5802

diam-connector:
  envDiameterRealm: 'oracle.com'
  envDiameterIdentity: 'ocpcf'

diam-gateway:
  envDiameterRealm: 'oracle.com'
  envDiameterIdentity: 'ocpcf-gateway'


ingress-gateway:
  #Service Mesh (Istio) to take care of load-balancing
  serviceMeshCheck: false

  # ----OAUTH CONFIGURATION - BEGIN ----
  oauthValidatorEnabled: false
  nfInstanceId: 6faf1bbc-6e4a-4454-a507-a14ef8e1bc11
  allowedClockSkewSeconds: 0
  nrfPublicKeyKubeSecret: ''
  nrfPublicKeyKubeNamespace: ''
  validationType: relaxed
  producerPlmnMNC: 123
  producerPlmnMCC: 456
  # ----OAUTH CONFIGURATION - END ----

  # Enable it to accept incoming http requests
  enableIncomingHttp: true

  # ---- HTTPS Configuration - BEGIN ----
  enableIncomingHttps: false

  service:
    ssl:
      tlsVersion: TLSv1.2
```

```
      initialAlgorithm: RSA256
      privateKey:
        k8SecretName: ocpcf-gateway-secret
        k8NameSpace: ocpcf
        rsa:
          fileName: rsa_private_key_pkcs1.pem
      certificate:
        k8SecretName: ocpcf-gateway-secret
        k8NameSpace: ocpcf
        rsa:
          fileName: ocegress.cer
      caBundle:
        k8SecretName: ocpcf-gateway-secret
        k8NameSpace: ocpcf
        fileName: caroot.cer
      keyStorePassword:
        k8SecretName: ocpcf-gateway-secret
        k8NameSpace: ocpcf
        fileName: key.txt
      trustStorePassword:
        k8SecretName: ocpcf-gateway-secret
        k8NameSpace: ocpcf
        fileName: trust.txt


egress-gateway:
  # ---- Oauth Configuration - BEGIN ----
  oauthClientEnabled: false
  nrfAuthority: 10.75.224.7:8085
  nfInstanceId: fe7d992b-0541-4c7d-ab84-c6d70b1b01b1
  consumerPlmnMNC: 345
  consumerPlmnMCC: 567
  # ---- Oauth Configuration - END ----

  # ---- HTTPS Configuration - BEGIN ----

  #Enabling it for egress https requests
  enableOutgoingHttps: false

  egressGwCertReloadEnabled: false
  egressGwCertReloadPath: /egress-gw/store/reload

  service:
    ssl:
      tlsVersion: TLSv1.2
      initialAlgorithm: RSA256
      privateKey:
        k8SecretName: ocpcf-gateway-secret
        k8NameSpace: ocpcf
        rsa:
          fileName: rsa_private_key_pkcs1.pem
        ecdsa:
          fileName: ssl_ecdsa_private_key.pem
      certificate:
        k8SecretName: ocpcf-gateway-secret
```

```
        k8NameSpace: ocpcf
        rsa:
          fileName: ocegress.cer
        ecdsa:
          fileName: ssl_ecdsa_certificate.crt
      caBundle:
        k8SecretName: ocpcf-gateway-secret
        k8NameSpace: ocpcf
        fileName: caroot.cer
      keyStorePassword:
        k8SecretName: ocpcf-gateway-secret
        k8NameSpace: ocpcf
        fileName: key.txt
      trustStorePassword:
        k8SecretName: ocpcf-gateway-secret
        k8NameSpace: ocpcf
        fileName: trust.txt
  # ---- HTTPS Configuration - END ----

  # ---- SCP Configuration - BEGIN ----
  # Change this to false when scp integration is not required
  scpIntegrationEnabled: false
  scpHttpHost: localhost
  scpHttpPort: 80
  scpHttpsHost: localhost
  scpHttpsPort: 443
  #Change this value to corresponding prefix "/" is not expected to be
provided along. Applicable for SCP with TLS enabled. Example: nef , pcf
etc.,
  scpApiPrefix: /
  # Default scheme applicable when 3gpp-sbi-target-apiroot header is
missing
  scpDefaultScheme: https
  # ---- SCP Configuration - END ----

  #Enable this if loadbalancing is to be done by egress instead of K8s
  K8ServiceCheck: false
```

Configurable Parameters

> **Note:**
>
> - All parameters mentioned as mandatory must be present in custom values file.
> - All fixed value parameters mentioned must be present in the custom values file with the exact values as specified here.

Global Configurations

These configuration parameters are common for all micro services.

Table 3-1 Customizable Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
dockerRegistry	Name of the Docker registry which hosts Policy Control Function (PCF) docker images	Yes	Not applicable	This is a docker registry running in OCCNE bastion server where all PCF docker images will be loaded. For example, 'occne-bastion:5000'
envMysqlHost	IP address or host name of the MySql server which hosts PCF's databases	Yes	Not applicable	
cmServiceNodePort	Custom node port for CM service	No	0	When not specified (default 0), kubernetes assigns a random port.
pcfDiamGatewayNodePort	Custom node port for Diameter Gateway service	No	0	When not specified, kubernetes assigns a random port.
envJaegerAgentHost	Hostname or IP address for the jaeger agent	Yes	Not applicable	This parameter is the fqdn of Jaeger Agent service running in OCCNE cluster under namespace occne-infra.
dbCredSecretName	Name of the Kubernetes secret object containing Database username and password	Yes	Not applicable	
privilegedDbCredSecretName	Name of the Kubernetes secret object containing Database username and password for an admin user	Yes	Not applicable	
pcfApiRoot	API root of PCF that is used in notification URLs generated by PCF's when sending request to other producer NFs (like NRF, UDR, CHF, etc..)	No	Ingress gateway service name and port	If not configured then the ingress gateway service name and port will be used as default value. Example: If the PCF is deployed in namespace "site1" with https enabled in port 443, then the default value will be "https://site1-pcf-egress-gateway:443"

Core Services

Table 3-2 Customizable Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
am-service.envMysqlDatabase	Name of the database for AM-Service	No	pcf_amservice	
sm-service.envMysqlDatabase	Name of the database for SM-Service	No	pcf_smservice	
sm-service.envMysqlDatabaseUserService	Name of the database of User Service	No	pcf_userservice	Same value as "user-service.envMysqlDatabase"
sm-service.auditSmSessionTtl	SM Policy Association normal age	No	86400	Specifies age of a SM policy association after which a record is considered to be stale on PCF and the SMF is queried for presence of such associations.
sm-service.auditSmSessionMaxTtl	SM Policy Association maximaum age	No	172800	Specifies maximum age of a SM Policy Association after which a record is purged from PCF SM database without sending further queries to SMF.
sm-service.defaultBsfApiRoot	Api root of pre-configured BSF	No	Not applicable	Required, if PCF uses pre-configured BSF. For Example: "https://bsf.apigateway:8001/"
user-service.envMysqlDatabase	Name of the database for User-Service	No	pcf_userservice	

Common Services

Table 3-3 Customizable Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
cm-service.enableHttps	Flag to enable/disable HTTPS for cm-service GUI/API	Optional	false	
config-server.envMysqlDatabase	Name of the database for Config Server service	No	ocpm_config_server	

Table 3-3 (Cont.) Customizable Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
queryservice.envMysqlDatabaseSmService	Specify the database name of SM service	Conditional	pcf_smservice	
queryservice.envMysqlDatabaseUserService	Specify the database name of User service	Conditional	pcf_userservice	Same value as "user-service.envMysqlDatabase"
perf-info.configmapPerformance.prometheus	Specifies Prometheus server URL	Conditional	http://prometheus-server.prometheus:5802	If no value is specified, PCFs load reported to NRF is always 0.
appinfo.serviceAccountName	K8s Service Account to access (RBAC) the K8s API server to retrieve status of PCF services and pods. The account should have read access ("get" , "watch" , "list") to pods, services and nodes	Conditional	Not applicable	If no value is specified, PCF creates a service account at the time of deployment.

NRF Client

Table 3-4 Customizable Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
global.deploymentNrfClientService.envNfNamespace	K8s namespace of PCF	Mandatory	Not Applicable	
global.deploymentNrfClientService.nfApiRoot	Api root of PCF	Mandatory	Not Applicable	same value as global.pcfApiRoot
nrf-client.configmapApplicationConfig.profile	Contains configuration parameters that goes into nrf-client's config map	Mandatory	Not Applicable	Refer below table for config parameters in config-map

Config parameters in Config-map

Parameter	Description	Allowed Values	Notes
primaryNrfApiRoot	Primary NRF API root <http scheme>://<Hostname/IP>:<Port>	valid api root	For Example: http://nrf1-api-gateway.svc:80
SecondaryNrfApiRoot	secondary NRF API root <http scheme>://<Hostname/IP>:<Port>	valid api root	For Example: http://nrf2-api-gateway.svc:80
retryAfterTime	When primary NRF is down, this will be the wait Time (in ISO 8601 duration format) after which request to primary NRF will be retried to detect primary NRF's availability.	valid ISO 8601 duration format	For Example: PT120S
nrfClientType	This should be set to PCF	PCF	
nrfClientSubscribeTypes	NF Type(s) for which the NF wants to discover and subscribe to the NRF	BSF,UDR,CHF	Leave blank if PCF does not require.
appProfiles	NfProfile of PCF to be registered with NRF	Valid NF Profile	
enableF3	Support for 29.510 Release 15.3	true/false	
enableF5	Support for 29.510 Release 15.5	true/false	
renewalTimeBeforeExpiry	Time Period(seconds) before the Subscription Validity time expires	Time in seconds	For Example: 3600 (1hr)
validityTime	The default validity time(days) for subscriptions	Time in days	For Example: 30 (30 days)
enableSubscriptionAutoRenewal	Enable Renewal of Subscriptions automatically	true/false	
acceptAdditionalAttributes	Enable additionalAttributes as part of 29.510 Release 15.5	true/false	

Diameter

Table 3-5 Customizable Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
diam-connector.envDiameterRealm	Diameter Realm of PCF	Yes	Not applicable	example: oracle.com
diam-connector.envDiameterIdentity	Diameter Host of PCF	Yes	Not applicable	example: ocpcf

Table 3-5 (Cont.) Customizable Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
diam-gateway.envDiameterRealm	Diameter Realm of PCF diameter gateway	Yes	Not applicable	example: oracle.com
diam-gateway.envDiameterIdentity	Diameter Host of PCF diameter gateway	Yes	Not applicable	example: ocpcf-gateway

Ingress Gateway Service

Table 3-6 Customizable Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
global.publicHttpSignalingPort	HTTP/2.0 Port of ingress gateway	No	80	
global.publicHttpsSignallingPort	HTTPS/2.0 Port of ingress gateway	No	443	
global.metalLbIpAllocationEnabled	Enable or disable IP Address allocation from Metallb Pool	No	false	
global.metalLbIpAllocationAnnotation	Address Pool Annotation for Metallb	No	"metallb.universe.tf/address-pool: signaling"	
ingress-gateway.serviceMeshCheck	Enable this parameter if load balancing is handled by Service Mesh	No	False	
ingress-gateway.oauthValidatorEnabled	Enable or disable Oauth Validator	Yes	False	
ingress-gateway.nfInstanceId	NF Instance Id of service producer	No	6faf1bbc-6e4a-4454-a507-a14ef8e1bc11	
ingress-gateway.allowedClockSkewSeconds	set this value if clock on the parsing NF (producer) is not perfectly in sync with the clock on the NF (consumer) that created by JWT	No	0	

Table 3-6 (Cont.) Customizable Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
ingress-gateway.nrfPublicKeyKubeSecret	Name of the secret which stores the public key(s) of NRF	No		
ingress-gateway.nrfPublicKeyKubeNamespace	Namespace of the NRF public key secret	No		
ingress-gateway.validationType	Possible values are: • strict • relaxed strict- If incoming request does not contain "Authorization" (Access Token) header, the request is rejected. relaxed- relaxed means that if Incoming request contains "Authorization" header, it is validated. If Incoming request does not contain "Authorization" header, validation is ignored.	No		
ingress-gateway.producerPlmnMNC	MNC of the service producer	No		
ingress-gateway.producerPlmnMCC	MCC of the service producer	No		
ingress-gateway.enableIncomingHttp	To enable http (INSECURE) for ingress traffic	No	False	
ingress-gateway.enableIncomingHttps	To enable https for ingress traffic	No	False	
ingress-gateway.service.ssl.privateKey.k8SecretName	Name of the privatekey secret	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.privateKey.k8NameSpace	Namespace of privatekey	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.privateKey.rsa.fileName	rsa private key file name	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.privateKey.ecdsa.fileName	ecdsa private key file name	No	Not Applicable	required if enableIncomingHttps is true

Table 3-6 (Cont.) Customizable Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
ingress-gateway.service.ssl.certificate.k8SecretName	Name of the privatekey secret	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.certificate.k8NameSpace	Namespace of privatekey	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.certificate.rsa.fileName	rsa private key file name	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.certificate.ecdsa.fileName	ecdsa private key file name	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.caBundle.k8SecretName	Name of the privatekey secret	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.caBundle.k8NameSpace	Namespace of privatekey	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.caBundle.rsa.fileName	rsa private key file name	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.keyStorePassword.k8SecretName	Name of the privatekey secret	No	Not Applicable	required if enableIncomingHttp is true
ingress-gateway.service.ssl.keyStorePassword.k8NameSpace	Namespace of privatekey	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.keyStorePassword.fileName	File name that has password for keyStore	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.trustStorePassword.k8SecretName	Name of the privatekey secret	No	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.trustStorePassword.k8NameSpace	Namespace of privatekey	No	Not Applicable	required if enableIncomingHttps is true

Table 3-6 (Cont.) Customizable Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
ingress-gateway.service.ssl.trustStorePassword.fileName	File name that has password for trustStore	No	Not Applicable	required if enableIncomingHttps is true

Egress Gateway Service

Table 3-7 Customization Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
egress-gateway.oauthClientEnabled	Oauth Validator Enabled	No	false	
egress-gateway.nrfAuthority	NRF's ${HOSTNAME}:{PORT}	No	Not Applicable	Modify the parameter with actual value, if oAuth is enabled.
egress-gateway.nfInstanceId	NF InstanceId of Producer	No	Not Applicable	Modify the parameter with actual value, if oAuth is enabled.
egress-gateway.consumerPlmnMNC	MNC of service Consumer	No		Modify the parameter with actual value, if oAuth is enabled.
egress-gateway.consumerPlmnMCC	MCC of service Consumer	No		Modify the parameter with actual value, if oAuth is enabled.
egress-gateway.enableOutgoingHttps	Enabling it for outgoing https request	No		
egress-gateway.egressGwCertReloadEnabled		No		
egress-gateway.egressGwCertReloadPath		No		
egress-gateway.service.ssl.privateKey.k8SecretName	Name of the privatekey secret	No	Not Applicable	
egress-gateway.service.ssl.privateKey.k8NameSpace	Namespace of privatekey	No	Not Applicable	

Table 3-7 (Cont.) Customization Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
egress-gateway.service.ssl.privateKey.rsa.fileName	rsa private key file name	No	Not Applicable	
egress-gateway.service.ssl.privateKey.ecdsa.fileName	ecdsa private key file name	No	Not Applicable	
egress-gateway.service.ssl.certificate.k8SecretName	Name of the privatekey secret	No	Not Applicable	
egress-gateway.service.ssl.certificate.k8NameSpace	Namespace of privatekey	No	Not Applicable	
egress-gateway.service.ssl.certificate.rsa.fileName	rsa private key file name	No	Not Applicable	
egress-gateway.service.ssl.certificate.ecdsa.fileName	ecdsa private key file name	No	Not Applicable	
egress-gateway.service.ssl.caBundle.k8SecretName	Name of the privatekey secret	No	Not Applicable	
egress-gateway.service.ssl.caBundle.k8NameSpace	Namespace of privatekey	No	Not Applicable	
egress-gateway.service.ssl.caBundle.rsa.fileName	rsa private key file name	No	Not Applicable	
egress-gateway.service.ssl.keyStorePassword.k8SecretName	Name of the privatekey secret	No	Not Applicable	
egress-gateway.service.ssl.keyStorePassword.k8NameSpace	Namespace of privatekey	No	Not Applicable	
egress-gateway.service.ssl.keyStorePassword.fileName	File name that has password for keyStore	No	Not Applicable	

Table 3-7 (Cont.) Customization Parameters

Parameter	Description	Mandatory Parameter	Default Value	Notes
egress-gateway.service.ssl.trustStorePassword.k8SecretName	Name of the privatekey secret	No	Not Applicable	
egress-gateway.service.ssl.trustStorePassword.k8NameSpace	Namespace of privatekey	No	Not Applicable	
egress-gateway.service.ssl.trustStorePassword.fileName	File name that has password for trustStore	No	Not Applicable	
egress-gateway.scpIntegrationEnabled	Change this to false when scp integration is not required	No	false	
egress-gateway.scpHttpHost	SCP HTTP IP/FQDN	No	Not Applicable	
egress-gateway.scpHttpPort	SCP HTTP PORT	No	80	
egress-gateway.scpHttpsHost	SCP HTTPS IP/FQDN	No	n/a	
egress-gateway.scpHttpsPort	SCP HTTPS PORT	No	443	
egress-gateway.scpApiPrefix	Change this value to corresponding prefix "/" is not expected to be provided along. Applicable only for SCP with TLS enabled.	No	/	
egress-gateway.scpDefaultScheme	Default scheme applicable when 3gpp-sbi-target-apiroot header is missing	No	https	
egress-gateway.K8ServiceCheck	Enable this if loadbalancing is to be done by egress instead of K8s	No	false	

4
Upgrading Policy Control Function

This chapter describes the Policy Control Function (PCF) upgrade strategies and procedures to upgrade from PCF 1.5.x to 1.6.x.

Upgrading from PCF 1.5.x to 1.6.x

To upgrade from PCF 1.5.x to 1.6.x:

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 an admin user and grant all the permissions to the admin user by executing the following command:

```
CREATE USER 'username'@'%' IDENTIFIED BY 'password';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_smservice.* TO 'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_amservice.* TO 'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_userservice.* TO 'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON ocpm_config_server.* TO 'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON oc5g_audit_service.* TO 'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_release.* TO 'username'@'%';
FLUSH PRIVILEGES;
```

where:
username is the username and *password* is the password for MYSQL admin user.

For Example: In the below example "pcfadminusr" is used as username, "pcfadminpasswd" is used as password and granting all the permissions to "pcfadminusr".

```
CREATE USER 'pcfadminusr'@'%' IDENTIFIED BY 'pcfadminpasswd';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_smservice.* TO 'pcfadminusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_amservice.* TO 'pcfadminusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_userservice.* TO 'pcfadminusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON ocpm_config_server.* TO 'pcfadminusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON oc5g_audit_service.* TO 'pcfadminusr'@'%';
```

```
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER, REFERENCES,
INDEX ON pcf_release.* TO 'pcfadminusr'@'%';
FLUSH PRIVILEGES;
```

5. Execute the command, `show grants for pcfadminusr`, to confirm that admin user has all the permission.
6. Run below SQL commands to create service-specific release entries for upgrade:

```
CREATE DATABASE IF NOT EXISTS `pcf_release`;
CREATE TABLE IF NOT EXISTS `pcf_release`.`release_config` ( `cfg_key`
varchar(255) COLLATE utf8_unicode_ci NOT NULL, `cfg_value` mediumtext
COLLATE utf8_unicode_ci, PRIMARY KEY (`cfg_key`))  DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci;
INSERT INTO `pcf_release`.`release_config`
values('public.hook.configserver','{"currentVersion" :
100500,"rollbackVersion" : -1}');
INSERT INTO `pcf_release`.`release_config`
values('public.hook.smservice','{"currentVersion" :
100500,"rollbackVersion" : -1}');
INSERT INTO `pcf_release`.`release_config`
values('public.hook.amservice','{"currentVersion" :
100500,"rollbackVersion" : -1}');
INSERT INTO `pcf_release`.`release_config`
values('public.hook.auditservice','{"currentVersion" :
100500,"rollbackVersion" : -1}');
INSERT INTO `pcf_release`.`release_config`
values('public.hook.userservice','{"currentVersion" :
100500,"rollbackVersion" : -1}');
```

7. Execute the command, `select * from pcf_release.release_config;`, to verify the release informtaion.
8. Exit from database and logout from MYSQL node.
9. Create a yaml file with the username and password with the syntax shown below:

```
apiVersion: v1
kind: Secret
metadata:
  name: pcf-admin-db-pass
type: Opaque
data:
  mysql-username: <base64 encoded mysql username>
  mysql-password: <base64 encoded mysql password>
```

> **Note:**
>
> The values for **mysql-username** and **mysql-password** should be base64 encoded.

10. Add the created secret in an existing namespace which needs an upgradation

```
kubectl create -f yaml_file_name -n release_namespace
```

where:
release_namespace is the deployment namespace used by the helm command.

yaml_file_name is a name of the yaml file that is created in step 9.

11. Update the customize *ocpcf-custom-values.yaml* file with the required input parameters. To customize the file, see Customizing Policy Control Function.

> **Note:**
>
> Before running the upgrade command, you should have pushed all the required Docker images for PCF release 1.5.x to the bastion server. To push the required docker images, see Installation Tasks

12. a. Upgrade PCF by using helm 2 upgrade command:

```
helm upgrade <release-namespace> <helm-chart> -f <custom-file>
```

    b. Upgrade PCF by using helm 3 upgrade command:

```
helm upgrade <release-name> <helm-chart> -f <custom-file> -n
<release-namespace>
```

    where:
    *helm-chart*is the location of the helm chart extracted from ocpcf-pkg-1.5.0.tgz file

    custom-file - is the name of the custom values yaml file (including location).

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

13. Execute the following command to get status of jobs and pods:

```
kubectl get jobs,pods -n <release-namespace>
```

    where:
    release-namespace is the name space in which to create PCF Kubernetes objects. All the PCF microservices are deployed in this kubernetes namespace.

    You will see the status as **Running** for all the nodes if the deployment has been done successfully.

5
Downgrading Policy Control Function

This chapter describes the Policy Control Function (PCF) roll back procedure from PCF 1.6.x to previous version.

To roll back from PCF 1.6.x to previous version:

1. Check which revision you need to roll back by executing the below command:

   ```
   helm history <release_namespace>
   ```

2. Execute the roll back command to roll back to that revision:
   a. Below is a command to roll back using Helm2:

      ```
      helm rollback <release_namespace> <revision number>
      ```

   b. Below is a command to roll back using Helm3:

      ```
      helm rollback <release_name> <revision number> -n
      <release_namespace>
      ```

3. Clean the tables and databases created as part of PCF 1.6.x by executing the below command:

   ```
   DROP DATABASE IF EXISTS pcf_release;
   DROP TABLE IF EXISTS pcf_userservice.SubscriberContext;
   ```

6
Uninstalling Policy Control Function

When you uninstall a Helm chart from your PCF deployment, it removes only the Kubernetes objects that it created during installation.

To uninstall, enter this command:

```
helm delete release_name
```

where *release_name* is the release name used by helm command.

Helm keeps a record of its releases, so you can still re-activate the release after you uninstall it.

To completely remove the release from the cluster, add the --purge option to the command:

```
helm delete --purge release_name
```

For example, to completely remove a release named "ocpcf",enter this command:

```
helm delete --purge ocpcf
```

Deleting Kubernetes Namespace

To delete kubernetes namespace, enter this command:

```
kubectl delete namespace release_namespace
```

where *release_namespace* is the deployment namespace used by helm command.

For example, to delete a kubernetes namespace named "ocpcf", enter this command:

```
kubectl delete namespace ocpcf
```

Cleaning Up Database

To clean up database, enter this command:

```
DROP DATABASE IF EXISTS database_name;
```

where *database_name* is the database created for this release.

7
Troubleshooting Policy Control Function

This section provides information to troubleshoot the common error which can be encountered during the installation and upgrade of Policy Control Function.

If `helm install` command Fails

This section covers the reasons and troubleshooting procedures if the `helm install` command fails.

Reasons for `helm install` failure:

- **Chart syntax issue [This issue could be shown in the few seconds]**
  Please resolve the chart specific things and rerun the `helm install` command, because in this case, no hooks should have begun.
- **Most possible reason [TIMEOUT]**
  If any job stuck in a pending/error state and not able to execute, it will result in the timeout after 5 minutes. As default timeout for helm command is "5 minutes". In this case, we have to follow the below steps to troubleshoot.
- **`helm install` command failed in case of duplicated chart**

  ```
  helm install /home/cloud-user/pcf_1.6.0/sprint3.1/ocpcf-1.6.0-sprint.
  3.1.tgz --name ocpcf2 --namespace ocpcf2 -f cust-ashish.yaml
  Error: release ocpcf2 failed: configmaps "perfinfo-config-ocpcf2"
  already exists
  ```

  Here, configmap 'perfinfo-config-ocpcf2' exists multiple times, while creating Kubernetes objects after pre-upgrade hooks, this will be failed. In this case also please go through the below troubleshooting steps.
  Troubleshooting steps:

  1. Execute the below command to cleanup the databases created by the `helm install` command :

     ```
     DROP DATABASE IF EXISTS `pcf_smservice`;
     DROP DATABASE IF EXISTS `pcf_amservice`;
     DROP DATABASE IF EXISTS `pcf_userservice`;
     DROP DATABASE IF EXISTS `ocpm_config_server`;
     DROP DATABASE IF EXISTS `oc5g_audit_service`;
     DROP DATABASE IF EXISTS `pcf_release`;
     ```

  2. Execute the below command to get kubernetes objects:

     ```
     kubectl get all -n <release_namespace>
     ```

     This gives a detailed overview of which objects are stuck or in a failed state.

3. Execute the below command to delete all kubernetes objects:

   ```
   kubectl delete all --all -n <release_namespace>
   ```

4. Execute the below command :

   ```
   helm ls --all
   ```

   If this is in a failed state, please purge the namespace using the command

   ```
   helm delete --purge <release_namespace>
   ```

   > **Note:**
   >
   > If the execution of this command is taking more time, run the below command parallelly in another session to clear all the delete jobs.
   >
   > ```
   > while true; do kubectl delete jobs --all -n
   > <release_namespace>; sleep 5;done
   > ```

   Monitor the below command:

   ```
   helm delete --purge <release_namespace>
   ```

   Once that is succeeded, press "ctrl+c" to stop the above script.

5. After the database cleanup, run the `helm install` command.

If `helm upgrade` command Fails

This section covers the reasons and troubleshooting procedures if the `helm upgrade` command fails.

Reasons for `helm upgrade` failure:

- **Chart syntax issue [This issue could be shown in the few seconds]**
  Please resolve the chart specific things and rerun the `helm upgrade` command, because in this case, no hooks should have begun.
- **Most possible reason [TIMEOUT]**
  If any job stuck in a pending/error state and not able to execute, it will result in the timeout after 5 minutes. As default timeout for helm command is "5 minutes". In this case, we have to follow the below steps to troubleshoot.
- **Helm upgrade failed in case of duplicated chart**

  ```
  helm upgrade upgradetestpcf <helm-chart> -f custom-value.yaml
  Error: release ocpcf2 failed: configmaps "perfinfo-config-ocpcf2"
  already exists
  ```

Here, configmap 'perfinfo-config-ocpcf2' exists multiple times, while creating Kubernetes objects after pre-upgrade hooks, this will be failed. In this case also please go through the below troubleshooting steps.

Troubleshooting steps:

1. Execute the below command to cleanup upgrade jobs:

   ```
   kubectl get jobs -n <release_namespace>
   ```

   This gives a detailed overview of all the objects. Delete all the kubernetes objects.

2. Execute the below command to cleanup kubernetes objects:

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

   Execute the above script until all jobs are killed from given release namespace.

3. Execute the below command to refill the version configurations for a clean upgrade:

   ```
   TRUNCATE TABLE `pcf_release`.`release_config`;
   INSERT INTO `pcf_release`.`release_config`
   values('public.hook.configserver','{"currentVersion" :
   100500,"rollbackVersion" : -1}');
   INSERT INTO `pcf_release`.`release_config`
   values('public.hook.smservice','{"currentVersion" :
   100500,"rollbackVersion" : -1}');
   INSERT INTO `pcf_release`.`release_config`
   values('public.hook.amservice','{"currentVersion" :
   100500,"rollbackVersion" : -1}');
   INSERT INTO `pcf_release`.`release_config`
   values('public.hook.auditservice','{"currentVersion" :
   100500,"rollbackVersion" : -1}');
   INSERT INTO `pcf_release`.`release_config`
   values('public.hook.userservice','{"currentVersion" :
   100500,"rollbackVersion" : -1}');
   ```

4. Fix the issues that happened in the helm upgrade. And run the `helm upgrade` command again.

A
Docker Images

Policy Control Function (PCF) cloud native deployment package includes ready-to-use docker images and Helm charts to help you orchestrate containers in Kubernetes.

You can use the Docker images and Helm chart to help you deploy and manage Pods of PCF product services in Kubernetes. Communication between Pods of services of PCF products are preconfigured in the Helm charts.

Table A-1 lists the docker images for PCF.

Table A-1 Docker Images for PCF

Service Name	Docker Image Name
AM Service	pcf-amservice
Application Info Service	app_info
CM Service	ocpm_cm_service
Config Server Service	ocpm_config_server
Diameter Connector	diam-connector
Diameter Gateway	diam-gateway
Egress Gateway	ocegress_gateway
Ingress Gateway	ocingress_gateway
Ingress/Egress Gateway init configuration	configurationinit
Ingress/Egress Gateway update configuration	configurationupdate
LDAP Gateway Service	ldap-gateway
Nrf Client Service	nrf_clientservice
Performance Monitoring Service	perf_info
PolicyDS Service	policyds
Policy Runtime Service	ocpm_pre
Query Service	ocpm_queryservice
Readiness check	readiness-detector
Session State Audit	audit_service
SM Service	pcf_smservice
UE Service	pcf_ueservice
User Service	pcf_userservice

B

Deployment Service Type Selection

Service Type	Description
ClusterIP	Exposes the service on a cluster-internal IP. Specifying 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 will route, is automatically created. You'll be able to contact the NodePort service, from outside the cluster, by requesting *NodeIP:NodePort* Most PCF service use NodePort to deploy in this release.
LoadBalancer	Exposes the service externally using a cloud provider's load balancer. NodePort and ClusterIP services, to which the external load balancer will route, are automatically created. For CM Service, API gateway, Diameter Gateway service, it's recommended to use LoadBalancer type. Given that the CNE already integrated with a load balancer (METALLB, for OCCNE deployed on baremetal).