Oracle® Communications

Policy Control Function Cloud Native Installation and Upgrade Guide

Release 1.0
F17001-01
May 2019

Oracle Communications Policy Control Function Cloud Native Installation and Upgrade Guide, Release 1.0

F17001-01

Contents

1 Introduction

References 1-1
Acronyms 1-1
Locate Product Documentation on the Oracle Help Center Site 1-2
Customer Training 1-3
My Oracle Support 1-3
Emergency Response 1-3

2 Installing Policy Control Function

Installation Sequence 2-1
Pre-requisites 2-1
    Creating Database Account on MySQL Database 2-2
Installation Preparation 2-5
Deploying Policy Control Function 2-6
    Verifying PCF Installation 2-11

3 Configuring Policy Control Function

Disabling BSF Binding 3-1
Disabling AM Service HTTP2 Flag 3-2
Enabling LoadBalancer with MetalLB 3-2
    Updating API-Gateway Service 3-2
Updating nrf Client Service configmap 3-5
    Get nrf Service URL 3-6
    Get PCF Profile 3-6
Verify kong API Gateway service 3-6

4 Upgrading Policy Control Function

5 Uninstalling Policy Control Function

6 Troubleshooting Policy Control Function

Cannot Access MySQL 6-1
Cassandra Data Lost 6-2
Kubernetes DNS Error 6-4
Service Status CrashLoopBackOff 6-5
Service Status ErrImagePull Error 6-6
Performance Service Fails to Start 6-6

List of Figures

3-1 Disabling BSF Binding 3-1

3-2 Disabling AM Service 3-2

6-1 DNS Error - Waiting 6-5

6-2 DNS Error - Others 6-5

List of Tables

1-1	Acronyms	1-1
2-1	Docker Image Name	2-7
2-2	Variable Details	2-8
2-3	Service Deployment Service Type	2-10
3-1	Variables	3-3
4-1	Parameters	4-2
6-1	Variables	6-4
6-2	Service Parameters	6-7

1 Introduction

The Oracle Communications Policy Management solution is enhanced to add Policy Control Function (PCF) that extends the functionality of PCRF as part of the 5G core network. The Policy Control Function is a functional element for policy control decision and flow-based charging control functionalities.

The PCF provides the following functions:

- Policy rules for application and service data flow detection, gating, QoS, and flow based charging to the SMF.
- Access and Mobility Management related policies to the AMF

References

Refer to the following documents for more information about 5G cloud native policy control function.

- Cloud Native Environment Installation Document
- Policy Control Function Cloud Native User's Guide

Acronyms

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

Table 1-1 Acronyms

Field	Description
5G-AN	5G Access Network
5GC	5G Core Network
5G-GUTI	5G Globally Unique Temporary Identifier
5QI	5G QoS Identifier
5G-S-TMSI	5G S-Temporary Mobile Subscription Identifier
5GS	5G System
5G-EIR	5G-Equipment Identity Register
(R)AN	(Radio) Access Network
AMF	Access and Mobility Management Function
AUSF	Authentication Server Function
CAPIF	Common API Framework for 3GPP northbound APIs
IWF	Inter-Working Function
NEF	Network Exposure Function
NF	Network Function
NRF	Network Repository Function
NSI ID	Network Slice Instance Identifier

Table 1-1 (Cont.) Acronyms

Field	Description
NSSAI	Network Slice Selection Assistance Information
NSSF	Network Slice Selection Function
NSSP	Network Slice Selection Policy
PEI	Permanent Equipment Identifier
PCF	Policy Control Function
QFI	QoS Flow Identifier
QoE	Quality of Experience
SEPP	Security Edge Protection Proxy
SBA	Service Based Architecture
SBI	Service Based Interface
SSC	Session and Service Continuity
SSCMSP	Session and Service Continuity Mode Selection Policy
SMF	Session Management Function
SMSF	Short Message Service Function
S-NSSAI	Single Network Slice Selection Assistance Information
UDM	Unified Data Management
UDR	Unified Data Repository
UDSF	Unstructured Data Storage Function

Locate Product Documentation on the Oracle Help Center Site

Oracle Communications customer documentation is available on the web at the Oracle Help Center (OHC) 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 the `Oracle Communications documentation` link.

   The Communications Documentation page appears. Most products covered by these documentation sets will appear under the headings "Network Session Delivery and Control Infrastructure" or "Platforms."
4. Click on your Product and then the Release Number.

   A list of the entire documentation set for the selected product and release appears.
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
Installing Policy Control Function

This section provides instruction for installing Policy Control Function.

Installation Sequence

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

1. Create MySQl database. See Creating Database Account on MySQL Database.
2. Download PCF package files and load them to the system. See Installation Preparation.
3. Prepare all variables for Helm install command. See Table 2-2.
4. PCF Deployment using Helm command. See Deploying Policy Control Function.
5. Verify PCF Deployment. See Verifying PCF Installation.
6. Configure PCF. See Configuring Policy Control Function.
7. Verify PCF Configuration.

Pre-requisites

Following are the pre-requisites required for installing Policy Control Function.

PCF Software

PCF software package includes:

- PCF Helm Charts
- PCF Docker Images

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 Metrics Area
prometheus	8.8.0	Needed for Metrics Area

Software	Chart Version	Notes
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:**
>
> 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 PCF installation.

Network access

The Kubernetes cluster hosts must have network access to:

- quay.io/datawire/ambassador docker image repository
- Local helm repository where the PCF helm charts are available
- Local docker image repository where the PCF images are available

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
- 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*.

Creating Database Account on MySQL Database

To create an database account on MySQL database,

1. Navigate from SSH to MySQL database.
2. Login to database using the command,

   mysql - h<mysqlhost> -u<user> -p<password>

3. Execute the following command. The username and password is only provided as an example:

   CREATE USER 'pcfusr'@'%' IDENTIFIED BY 'pcfpasswd';

   GRANT ALL PRIVILEGES ON *.* TO 'pcfusr'@'%';

Execute the following script to the initial PCF databases with above created database user. At first login to MySQL console via new user created above, `mysql -h<mysqlhost> -u<pcfuser> -p<pcfpassword>`

```
CREATE DATABASE IF NOT EXISTS `ocpm_config_server`;

CREATE TABLE IF NOT EXISTS `ocpm_config_server`.`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`.`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`.`topic_info` (name, version) values ('policy',
1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('policySchema', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('policyElement', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('policyParam', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('policygui', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pcf.global.cfg', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pcf.smservice.cfg', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pcf.public.sessionrule', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pcf.public.sessionruleprofile', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pcf.public.authorizeddefaultqos', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pcf.public.pccrule', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pcf.public.qosdata', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
```

```
('pcf.public.chargingdata', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pcf.public.pccruleprofile', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('amservice.system', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('public.matchlist', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pe.serviceTag', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pe.policyTag', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pe.logLevel', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pcf.amservice.app', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('nrfclient.cfg', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values ('NRF.UDR',
1);
insert into `ocpm_config_server`.`topic_info` (name, version) values ('NRF.BSF',
1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pcf.userservice.cfg', 1);
insert into `ocpm_config_server`.`topic_info` (name, version) values ('NRF.CHF',
1);
insert into `ocpm_config_server`.`topic_info` (name, version) values
('pcf.chfservice', 1);
CREATE DATABASE IF NOT EXISTS `pcf`;
create table if not exists  pcf.SmPolicyAssociation (
k Varchar(1024) primary key not null ,
v Text not null,
SUPI Varchar(128),
GPSI Varchar(128),
IPV4 Varchar(15),
IPV6 Varchar(45),
DNN Varchar(128),
SNSSAI Varchar(16),
IPD Varchar(128)
);
create table if not exists pcf.userservice_user (
  k varchar(128) primary key not null ,
  v blob not null,
  msisdn varchar(128),
  imsi varchar(128),
  nai varchar(128),
  extid varchar(128),
  other varchar(128)
);
create index idx_msisdn on pcf.userservice_user (msisdn);
create index idx_imsi on pcf.userservice_user (imsi);
create index idx_nai on pcf.userservice_user (nai);
create index idx_extid on pcf.userservice_user (extid);
create index idx_other on pcf.userservice_user (other);
create table if not exists pcf.chf_user (
  k varchar(128) primary key not null,
  v blob not null,
  supi varchar(128),
  gpsi varchar(128)
);
create index idx_supi on pcf.chf_user (supi);
create index idx_gpsi on pcf.chf_user (gpsi);
```

```
CREATE DATABASE IF NOT EXISTS `pcf_amservice`;
CREATE TABLE IF NOT EXISTS `pcf_amservice`.`documents` (
  `id` varchar(45) NOT NULL,
  `projection` varchar(45) DEFAULT NULL,
  `name` varchar(100) DEFAULT NULL,
  `data` blob,
  PRIMARY KEY (`id`)
)  DEFAULT CHARSET=utf8;
CREATE TABLE IF NOT EXISTS `pcf_amservice`.`events` (
  `id` varchar(45) NOT NULL,
  `streamId` varchar(45) DEFAULT NULL,
  `data` blob,
  `type` varchar(100) DEFAULT NULL,
  `version` int(11) DEFAULT NULL,
  `createdDate` datetime(6) DEFAULT NULL,
  `sequence` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`)
)  DEFAULT CHARSET=utf8;

CREATE TABLE IF NOT EXISTS `pcf_amservice`.`projections` (
  `name` varchar(100) NOT NULL,
  `lastEventSequence` int(11) DEFAULT NULL,
  PRIMARY KEY (`name`)
)   DEFAULT CHARSET=utf8;
CREATE TABLE IF NOT EXISTS `pcf_amservice`.`snapshots` (
  `id` varchar(45) NOT NULL,
  `stream` varchar(45) DEFAULT NULL,
  `data` blob,
  `lastEventSequence` int(11) DEFAULT NULL,
  `createDate` datetime DEFAULT NULL,
  PRIMARY KEY (`id`)
)   DEFAULT CHARSET=utf8;

CREATE TABLE IF NOT EXISTS `pcf_amservice`.`streams` (
  `id` varchar(45) NOT NULL,
  `type` varchar(100) DEFAULT NULL,
  `version` int(11) DEFAULT NULL,
  `createdDate` datetime(6) DEFAULT NULL,
  `deletedDate` datetime(6) DEFAULT NULL,
  PRIMARY KEY (`id`)
)  DEFAULT CHARSET=utf8;
```

Result: The database account is created. Execute the command, `show databases` to view database.

Installation Preparation

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

1. Download the PCF package file from Oracle Software Delivery Cloud (OSDC). Package is named as follows:
   `<nfname>-pkg-<marketing-release-number>.tgz`

   For example, `ocpcf-pkg-1.0.0.0.0.tgz`

2. Untar the PCF Package File.
   `tar -xvf <<nfname>-pkg-<marketing-releasenumber>>.tgz`

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

The directory consists of following:

- **PCF Docker Images File:**
  `ocpcf-images-1.0.0.tar`
- **Helm File:**
  `ocpcf-1.0.0.tgz`
- **Readme txt File:**
  `Readme.txt` (Contains cksum and md5sum of tarballs)

3. Verify the checksums of tarballs mentioned in `Readme.txt`.

Deploying Policy Control Function

The Policy Control Function requires a MySQL database to store the configuration and run time data.

The PCF Software components as mentioned in pre-requisites section, can be extracted following the below steps.

1. Download the file, **ocpcf-pkg-1.0.0.0.0.tgz**.
2. Untar **ocpcf-pkg-1.0.0.0.0.tgz**.
3. Untar displays the following files:

```
ocpcf-pkg-1.0.0.0.0.tgz
          |_ _ _ _ _ _ ocpcf-1.0.0.tgz (helm chart to be used in step 3)
          |_ _ _ _ _ _ ocpcf-images-1.0.0.tar (docker images)
          |_ _ _ _ _ _ Readme.txt (Contains cksum and md5sum of tarballs)
```

4. Check the checksums of tarballs mentioned in the Readme.txt file.
5. Run the following command to load **ocpcf-images-1.0.0.tar** to docker and push imported docker images to user docker registry.

```
docker load --input /<IMAGE_PATH>/ocpcf-images-1.0.0.tar

docker tag ocpcf/pcf_smservice:1.0.0 <customer repo>/pcf_smservice:1.0.0
docker push <customer repo>/pcf_smservice:1.0.0

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

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

> **Note:**
>
> Each service deployment yaml file uses global.imageTag as image tag to fetch related docker image per helm chart design.
>
> With release tar file, the global image tag for all services is 1.0.0

Table 2-1 provides details of docker images file names.

Table 2-1 Docker Image Name

S.No	Service Name	Docker Image Name	Service Category
1	User Service	pcf_userservice	PCF
2	AM Service	pcf-amservice	PCF
3	SM Service	pcf_smservice	PCF
4	DB Service	db-service	Common
5	Nrf Client Service	nrf_clientservice	Common
6	CM Service	ocpm_cm_service	Common
7	Performance Monitoring Service	perf_info	Platform
8	Config Server Service	ocpm_config_server	Common
9	Policy Runtime Service	ocpm_pre	PCF
10	Diameter Gateway	diam-gateway	PCF
11	Diameter Connector	diam-connector	PCF
12	Application Info Service	app_info	Platform
13	Readiness check	readiness-detector	Common
14	Oracle Linux 7 with JDK11	ocpm_ol7_jdk11	Common

> **Note:**
>
> Before proceeding with the installation, setup the variables. See Table 2-2.

6. Navigate to helm chart and execute the following command to install PCF:

> **Note:**
>
> It is mandatory to run following command under helm chart folder as the last line, `./<HELM_CHART_NAME_WITH_EXTENSION>` specifies that helm chart path is the current working path. If you want to run the command in another server, copy the helm chart file to that server folder, before executing the command from that server.

```
helm install --namespace=<NAMESPACE>-pcf --name=<NAME>-pcf \
--set
global.envMysqlHost=<MYSQL_HOST>,global.envMysqlUser=pcfusr,global.envMysqlPas
sword=pcfpasswd \
--set global.envJaegerAgentHost=<JAEGER_SERVICE>.<JAEGER_SERVICE_NAMESPACE> \
--set
global.envManageNF=PCF,global.envSystemName=PCF,common.configmapApplicationCon
fig.nrfClientType=PCF \
--set
common.configmapApplicationConfig.nrfDeployName=<NAME>,common.configmapApplica
tionConfig.nrfDeployNamespace=<NAMESPACE> \
--set
global.imageTag=<IMAGE_TAG>,global.dockerRegistry=<DOCKER_REGISTRY_ADDRESS> \
--set
platform.enabled=true,pcf.enabled=true,bsf.enabled=false,common.enabled=true,n
ef.enabled=false \
--set global.envDefaultBsfDeployName=<NAME>-
pcf,global.envDefaultBsfDeployNS=<NAMESPACE>-
pcf ,common.deploymentNrfClientservice.envNamespace=<NAMESPACE>-pcf \
./<HELM_CHART_NAME_WITH_EXTENSION>
```

Table 2-2 Variable Details

S.No	Variable	Description	Notes
1	<NAMESPACE>	Indicates deployment NF namespace used by helm command	Variable name can include uppercase and lower case alphabets, numbers, and special characters _, and -. Maximum allowed character length is 10.
2	<NAME>	Indicates deployment NF name used by helm command	
3	<MYSQL_HOST>	MySQL host name or IP address	global.envMysqlUser and global.envMysqlPassword variables in above command from database section configured in previous step

Table 2-2 (Cont.) Variable Details

S.No	Variable	Description	Notes
4	<JAEGER_SERVICE> <JAEGER_SERVICE_NAMESPACE>	Both parameters could be found in same Kubernetes cluster.	Use the following format "<JAEGER_AGENT_SERVICE_NAME>.<JAEGER_NAMESPACE>" Such as "occne-tracer-jaeger-agent.occne-infra", occne-tracer-jaeger-agent is jaeger agent service name under jaeger deployment
5	<IMAGE_TAG>	The image tag used in customer docker registry. It is recommend to use the same image tag when pulling docker image to registry. If followed above steps to push docker image to customer docker registry then the <IMAGE_TAG> value should be 1.0.0	Each service deployment yaml file would use global.imageTag as image tag to fetch related docker image per helm chart design, if one service cannot use global image tag then please edit that service part under values.yaml, such as below: Before update: image: bsf_management_service imageTag: " After update: image: bsf_management_service imageTag: <IMAGE_TAG>
6	<DOCKER_REGISTRY_ADDRESS>	Indicates user docker registry address	If registry has port value, add port as well, such as "reg-1:5000"

Table 2-2 (Cont.) Variable Details

S.No	Variable	Description	Notes
7	global.envDefaultBsfDeployName global.envDefaultBsfDeployNS	PCF need to interact with BSF service for some functionalities, these two parameters are used for interaction.	• If installing PCF without BSF, then these two parameters should be set as above helm install command: **global.envDefaultBsfDeployName=<NAME>-pcf** **global.envDefaultBsfDeployNS=<NAMESPACE>-pcf** • If installing both PCF and BSF, then these two parameters could be configured to indicate BSF deploy name and namespace as following to enable the interaction of PCF and BSF: **global.envDefaultBsfDeployName=<NAME>-bsf,** **global.envDefaultBsfDeployNS=<NAMESPACE>-bsf** For installing BSF, refer to *Oracle Communications Binding Support Function Cloud Native Installation and Upgrade Guide*.

Kubernetes provides the following three deployment types:

Table 2-3 Service Deployment Service Type

S.No	Service Type	Description
1	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

Table 2-3 (Cont.) Service Deployment Service Type

S.No	Service Type	Description
2	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. User will be able to contact the NodePort service, from outside the cluster, by requesting <NodeIP>:<NodePort>. Most PCF service use NodePort to deploy.
3	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 GUI page and API gateway service, It is mandatory to use loadBalancer type. Given latest OCCNE already integrated METALLB, configure IP address to METALLB on OCCNE.

Verifying PCF Installation

Run the following command to verify the PCF installation:

```
kubectl get svc -n <PCF-Namespace>
kubectl get pod -n <PCF-Namespace>
```

If the installation is successful, all the pods should be in Running/Completed status except for some error pods named kong-migration. See the section in Known behavior about failed kong-migration pod in Verify kong API Gateway service.

3
Configuring Policy Control Function

> **Note:**
>
> Before configuring, verify the installation.

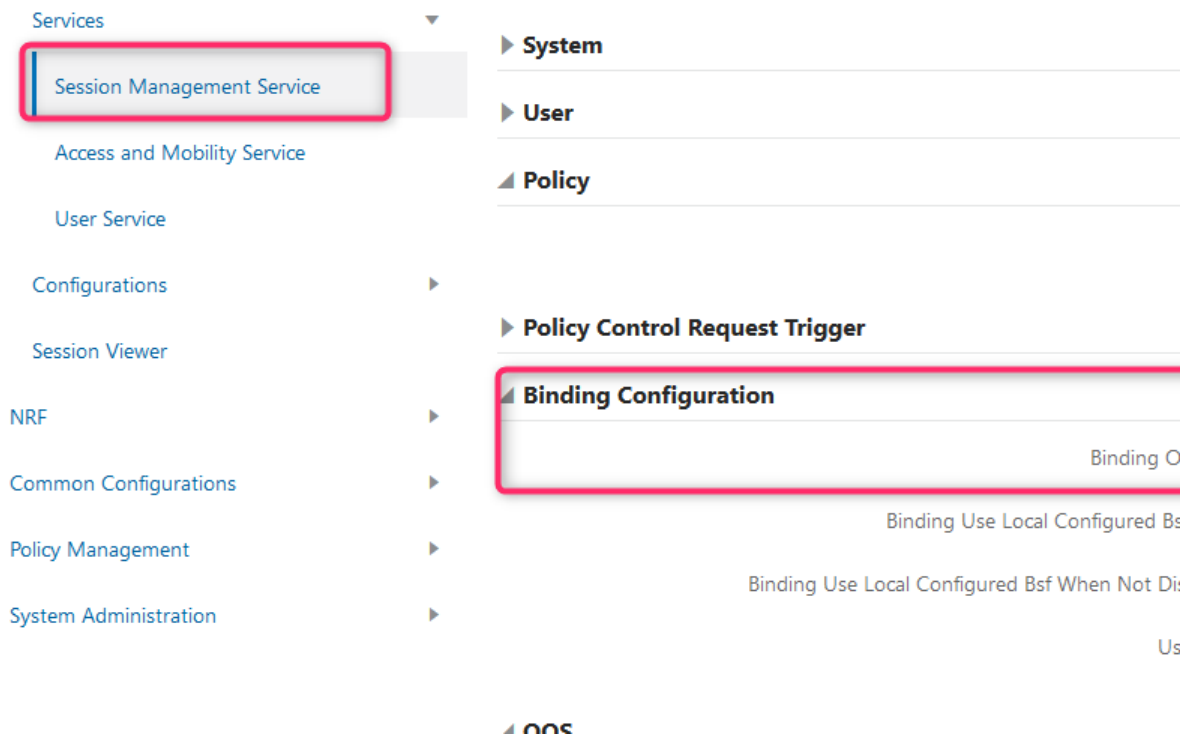
Click the PODS to ensure all ports are active and running.

The following subsections provides the information for configuring PCF.

Disabling BSF Binding

If you are installing only PCF, it is mandatory to disable BSF binding on the user interface page. By default, the binding setting is enabled as otherwise, the Session Management service application throws an exception. Disable the BSF binding as shown in the Figure 3-1:

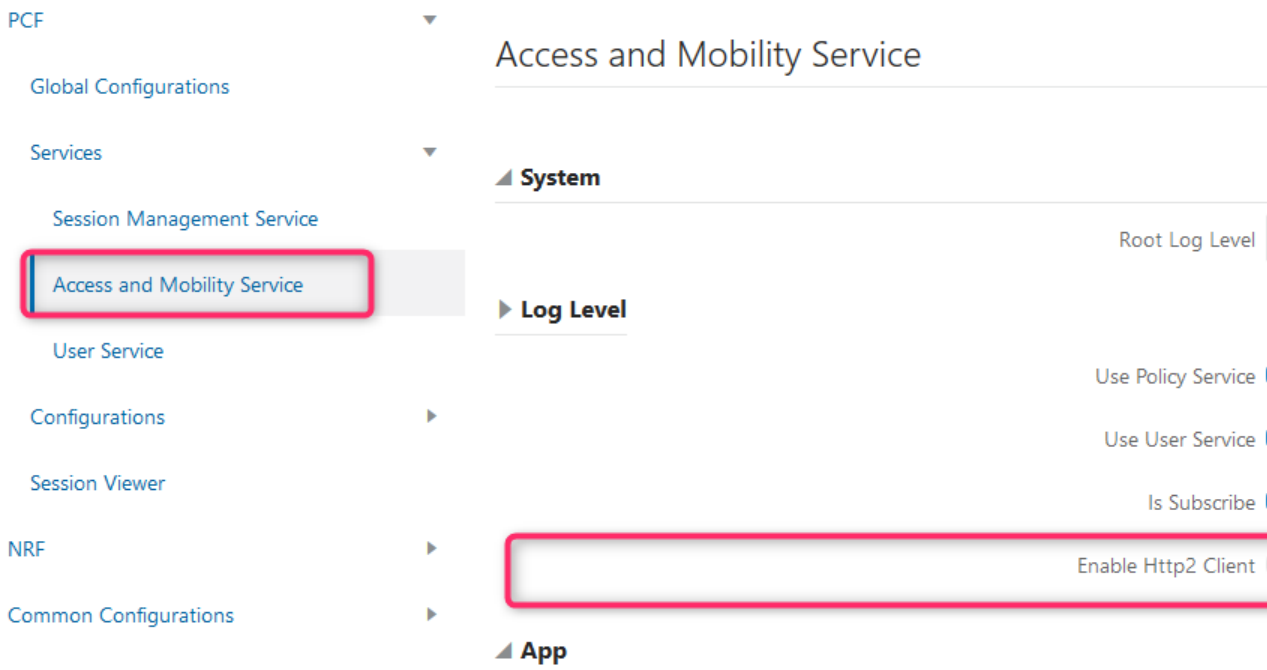
Figure 3-1 Disabling BSF Binding



Disabling AM Service HTTP2 Flag

By default the AM (Access and Mobility Service) uses HTTP2 to interact with other services, however, few services do not support HTTP2. User need to disable HTTP2 for AM service and use HTTP1 instead. Disable HTTP2 and enable HTTP1 as shown in the Disabling AM Service HTTP2 Flag:

Figure 3-2 Disabling AM Service



Enabling LoadBalancer with MetalLB

Cloud Native Network have MetalLB installed, and free external IPs are already configured under MetalLB.

Perform the following steps to enable LoadBalancer to specific services.

> **Note:**
>
> In PCF namespace, only API-Gateway service and cm service with GUI page requires load-balancer setting with accessible external IP. Other services are accessible by API-Gateway service.

Updating API-Gateway Service

To update API-Gateway service: Logon kubernetes cluster master node via ssh command,

1. Login to Kubernetes cluster master node using ssh command.
2. Run the following command to edit svc yaml file for API-Gateway:

```
kubectl edit svc <PCF_NAME>-pcf-api-gateway-service -n <PCF_NAME_SPACE>
```

Table 3-1 Variables

Variable Name	Description
PCF_NAME	The --name value used in helm install command
PCF_NAME_SPACE	The --namespace value used in helm install command

Following is an sample content that displays in API-Gateway edit window.

```
1 # Please edit the object below. Lines beginning with a '#' will be ignored,
 2 # and an empty file will abort the edit. If an error occurs while saving
this file will be
 3 # reopened with the relevant failures.
 4 #
 5 apiVersion: v1
 6 kind: Service
 7 metadata:
 8   creationTimestamp: 2019-04-02T08:17:51Z
 9   labels:
10     category: common
11     io.kompose.service: <PCF_NAME>-pcf-api-gateway-service
12   name: <PCF_NAME>-pcf-api-gateway-service
13   namespace: <PCF_NAME_SPACE>
14   resourceVersion: "25282719"
15   selfLink: /api/v1/namespaces/<PCF_NAME_SPACE>/services/<PCF_NAME>-pcf-
api-gateway-service
16   uid: cec8f019-551f-11e9-acc3-a0369f714f30
17 spec:
18   clusterIP: 10.233.63.101
19   externalTrafficPolicy: Cluster
20   ports:
21   - name: http
22     nodePort: 32314
23     port: 8080
24     protocol: TCP
25     targetPort: 8080
26   selector:
27     io.kompose.service: <PCF_NAME>-pcf-api-gateway-service
28   sessionAffinity: None
29   type: NodePort
30 status:
31   loadBalancer: {}
```

3. Add two new lines after line 7, after metadata:

```
annotations:
```

```
metallb.universe.tf/address-pool: <ADDRESS_POOL_NAME>
```

> **Note:**
>
> - As per user MetalLB setting, select an appropriate pool name to replace the variable, *<ADDRESS_POOL_NAME>*
> - *annotation:* line must be kept vertical align with line 16, while following line, *metallb.universe.tf/address-pool: <ADDRESS_POOL_NAME>* must be kept vertical align with line 10. If vertical align restriction failed to follow this rule, the svc yaml file update may fail.

4. Replace line 29 text, **type: NodePort** with **type: LoadBalancer**.
Following is the sample content after replacing the line 29:

```
1 # Please edit the object below. Lines beginning with a '#' will be ignored,
 2 # and an empty file will abort the edit. If an error occurs while saving
this file will be
 3 # reopened with the relevant failures.
 4 #
 5 apiVersion: v1
 6 kind: Service
 7 metadata:
 8   creationTimestamp: 2019-04-02T08:17:51Z
 9   labels:
10     category: common
11     io.kompose.service: <PCF_NAME>-pcf-api-gateway-service
12   name: <PCF_NAME>-pcf-api-gateway-service
13   namespace: <PCF_NAME_SPACE>
14   resourceVersion: "25282719"
15   selfLink: /api/v1/namespaces/<PCF_NAME_SPACE>/services<PCF_NAME>-pcf-
api-gateway-service
16   uid: cec8f019-551f-11e9-acc3-a0369f714f30
     annotations:
       metallb.universe.tf/address-pool: <ADDRESS_POOL_NAME>
17 spec:
18   clusterIP: 10.233.63.101
19   externalTrafficPolicy: Cluster
20   ports:
21   - name: http
22     nodePort: 32314
23     port: 8080
24     protocol: TCP
25     targetPort: 8080
26   selector:
27     io.kompose.service: <PCF_NAME>-pcf-api-gateway-service
28   sessionAffinity: None
29   type: LoadBalancer
30 status:
31   loadBalancer: {}
```

5. Quit vim editor and save changes. A new API-Gateway pod starts up.

   a. In the new pod, following sample content displays. Note that if the EXTERNAL-IP is available then the load balancer setting for API-Gateway service works.

```
NAME                        TYPE           CLUSTER-IP      EXTERNAL-IP
PORT(S)                                                      AGE
<PCF_NAME>-pcf-api-gateway       LoadBalancer   10.xxx.xx.xx
10.xxx.xxx.xx   80:30000/TCP,443:30001/TCP,8001:30373/TCP,8444:31448/
TCP    4d
```

Updating cm-service

Follow the same process to update svc yaml for **<PCF_NAME> -pcf-cm-service**.

Updating nrf Client Service configmap

Following are few use cases which involves updating NRF/PCF:

- Currently deployed PCF uses updated configmap to register PCF itself to NRF service.
- Client gets PCF related FQDN, IPv4, or IPv6 address from NRF service and send request to specific service under PCF with that FQDN, IPv4 or IPv6 address. The request reaches to API Gateway service under PCF deployment.
- The API Gateway service under PCF would parse coming request, then distributes the request to specific service per request URL and forward response from upstream service to client.

The nrf client configmap establish connection between NRF and PCF, in order to register PCF to NRF services and hence it is mandatory to update configmap to use correct nrf service URL and apply extra settings. Before updating nrf Client:

- Get nrf Service URL. See Get nrf Service URL.
- Get PCF Profile. See Get PCF Profile.

To update the nrf client service:

1. Run the following command and get config map item, **<PCF-NAME>-application-config**.

   ```
   kubectl get configmap -n <PCF-Namespace>
   ```

2. Run the following and then navigate to vim editor with config map content displayed.

   ```
   kubectl edit configmap <PCF-NAME>-application-config -n <PCF-Namespace>
   ```

3. Update **nrfApiRoot** to actual deployed nrf service URL. For example, **nrfApiRoot=http://ocnrf-endpoint.nrf1-1:80**. See Get nrf Service URL to get actual nrf service URL.
   At least one of the address parameters such as fqdn, ipv4address, or ipv6adress which marked bold should populate in the NF Profile under **configmap** definition. NF Profile here indicates the entrance of currently deployed PCF, which means it is the API Gateway service URL (FQDN), API Gateway service IP V4 address, or IP V6 address.

   ```
   appProfiles=[{"nfInstanceId": "fe7d882b-0541-4c7d-ab84-
   c6d70b1b0123","nfType": "PCF","nfStatus": "REGISTERED","plmn":
   null,"nsiList": null, "fqdn": null,"interPlmnFqdn": null,"ipv4Addresses":
   null,"ipv6Addresses": null
   ```

   It is recommended to update fqdn value while keeping IPv4Addresses and IPv6Addresses as it is.

   For example, consider **ocpcf-api-gateway.pcf1:80** for fqdn value, then the updated result is below:

   ```
   appProfiles=[{"nfInstanceId": "fe7d882b-0541-4c7d-ab84-
   c6d70b1b0123","nfType": "PCF","nfStatus": "REGISTERED","plmn":
   null,"nsiList": null, "fqdn": "ocpcf-api-gateway.pcf1:80","interPlmnFqdn":
   null, "ipv4Addresses": null,"ipv6Addresses": null.......
   ```

Refer to Get PCF Profile to get actual fqdn value.

4. Quit vim edit mode and save the changes. Wait for a while to verify whether PCF had been registered to NRF or not.

Get nrf Service URL

- Run the following command and get nrf service name (if you have installed Oracle NRF, then service name is **ocnrf-endpoint**) and service port (default port value is 80).

  ```
  kubectl get svc -n <NRF-Namespace>
  ```

- For example, *<NRF-Namespace>* variable is **nrf1-1** and *nrf service name* as **ocnrf-endpoint**, then the *nrf service value* is **ocnrf-endpoint.nrf1-1:80**

Get PCF Profile

The PCF Profile here indicates the entrance of current deployed PCF service sets, which means it is the API Gateway service URL (FQDN), API Gateway service IP V4 address or IP V6 address.

Get API gateway service FQDN via kubectl command

- Run the following command and get API gateway service name, **<PCF-Name>-pcf-api-gateway** with default application port value 80.
  ```
  kubectl get svc -n <PCF-Namespace>
  ```

  For example, consider <PCF-Namespace> variable, as **pcf1** and API gateway service name as **ocpcf-api-gateway**, the API gateway service value is **ocpcf-api-gateway.pcf1:80**

Verify kong API Gateway service

As the entrance of PCF/BSF, all request goes through API gateway to specific service. It is necessary to verify if it works.

Refer to Enabling LoadBalancer with MetalLB for detailed description to update API gateway service to use loadbalancer.

> **Note:**
>
> If any of the following steps fails, refer to the section, Troubleshooting Policy Control Function.

1. Login to Kubernetes Dashboard.
2. Select **pcf** namespace.
3. Click **Services** and navigate to service page to find how to access API gateway service.
   - **If API gateway is deployed with Loadbalancer type:**
     - Get external IP of API gateway service, and open the link, http://<API_GATEWAY_EXTERNAL_IP>:8001/services in a web browser., user can see API gateway services data similar to the following:

       ```
       {
          "next":null,
       ```

```
"data":[
   {
      "host":"<NAME>-pcf-nrf-clientservice",
      "created_at":1553667178,
      "connect_timeout":60000,
      "id":"897af1e0-8562-4843-b66a-e7453d29eeb9",
      "protocol":"http",
      "name":"nrf-client",
      "read_timeout":60000,
      "port":5910,
      "path":null,
      "updated_at":1553667178,
      "retries":5,
      "write_timeout":60000
   },
   {
      "host":"istio-ingressgateway.istio-system.svc",
      "created_at":1553667178,
      "connect_timeout":60000,
      "id":"8c2b6ade-d876-4edd-ad28-c0fd28807a5c",
      "protocol":"http",
      "name":"sm-ingress",
      "read_timeout":60000,
      "port":80,
      "path":null,
      "updated_at":1553667178,
      "retries":5,
      "write_timeout":60000
   },
   {
      "host":"<NAME>-pcf-bsf-management-service",
      "created_at":1553667178,
      "connect_timeout":60000,
      "id":"a3fa1ace-7016-4a2b-aabf-0e7e7a2618f0",
      "protocol":"http",
      "name":"bsf-service",
      "read_timeout":60000,
      "port":5903,
      "path":null,
      "updated_at":1553667178,
      "retries":5,
      "write_timeout":60000
   },
   {
      "host":"<NAME>-pcf-pcf-amservice",
      "created_at":1553667178,
      "connect_timeout":60000,
      "id":"aa5ac6a2-162a-45ea-9a3c-c48f501b827e",
      "protocol":"http",
      "name":"am-service",
      "read_timeout":60000,
      "port":5904,
      "path":null,
      "updated_at":1553667178,
      "retries":5,
      "write_timeout":60000
   },
   {
      "host":"<NAME>-pcf-pcf-smservice",
      "created_at":1553667178,
      "connect_timeout":60000,
```

```
            "id":"d020415d-6e50-4e35-87b5-050eedb1ee6c",
            "protocol":"http",
            "name":"sm-service",
            "read_timeout":60000,
            "port":5809,
            "path":null,
            "updated_at":1553667178,
            "retries":5,
            "write_timeout":60000
         },
         {
            "host":"<NAME>-pcf-pcf-userservice",
            "created_at":1553667178,
            "connect_timeout":60000,
            "id":"fe364cb8-2ff9-447f-88e6-f4e21b8d9022",
            "protocol":"http",
            "name":"user-service",
            "read_timeout":60000,
            "port":5808,
            "path":null,
            "updated_at":1553667178,
            "retries":5,
            "write_timeout":60000
         }
      ]
   }
```


– Use new URL, http://<API_GATEWAY_EXTERNAL_IP>:8001/routes to check API gateway routes data:


```
{
   "next":null,
   "data":[
      {
         "created_at":1553667178,
         "updated_at":1553667178,
         "strip_path":true,
         "service":{
            "id":"a3fa1ace-7016-4a2b-aabf-0e7e7a2618f0"
         },
         "name":null,
         "hosts":null,
         "id":"0bfb813d-64cb-465c-b8ab-fba851c24c94",
         "preserve_host":false,
         "regex_priority":0,
         "paths":[
            "\/bsf-service"
         ],
         "sources":null,
         "destinations":null,
         "snis":null,
         "protocols":[
            "http",
            "https"
         ],
         "methods":null
      },
      {
         "created_at":1553667178,
         "updated_at":1553667178,
         "strip_path":true,
         "service":{
```

```
            "id":"aa5ac6a2-162a-45ea-9a3c-c48f501b827e"
         },
         "name":null,
         "hosts":null,
         "id":"52265a1f-9fcb-405f-ba10-1a1b3042a539",
         "preserve_host":false,
         "regex_priority":0,
         "paths":[
            "\/am-service"
         ],
         "sources":null,
         "destinations":null,
         "snis":null,
         "protocols":[
            "http",
            "https"
         ],
         "methods":null
      },
      {
         "created_at":1553667178,
         "id":"5eec4c17-4b03-4bb1-a30d-1812121b8136",
         "strip_path":true,
         "service":{
            "id":"d020415d-6e50-4e35-87b5-050eedb1ee6c"
         },
         "name":"sm-route",
         "hosts":null,
         "updated_at":1553667178,
         "preserve_host":false,
         "regex_priority":0,
         "paths":[
            "\/sm-service"
         ],
         "sources":null,
         "destinations":null,
         "snis":null,
         "protocols":[
            "http",
            "https"
         ],
         "methods":null
      },
      {
         "created_at":1553667178,
         "updated_at":1553667178,
         "strip_path":true,
         "service":{
            "id":"897af1e0-8562-4843-b66a-e7453d29eeb9"
         },
         "name":null,
         "hosts":null,
         "id":"83c923a5-1c61-46af-b1b9-a7fa16ce3ce8",
         "preserve_host":false,
         "regex_priority":0,
         "paths":[
            "\/nrf-client"
         ],
         "sources":null,
         "destinations":null,
         "snis":null,
```

```
            "protocols":[
               "http",
               "https"
            ],
            "methods":null
         },
         {
            "created_at":1553667178,
            "updated_at":1553667178,
            "strip_path":true,
            "service":{
               "id":"8c2b6ade-d876-4edd-ad28-c0fd28807a5c"
            },
            "name":null,
            "hosts":null,
            "id":"b42ee6c2-adc6-4ca5-b1ea-d90de1a78529",
            "preserve_host":false,
            "regex_priority":0,
            "paths":[
               "\/sm-ingress"
            ],
            "sources":null,
            "destinations":null,
            "snis":null,
            "protocols":[
               "http",
               "https"
            ],
            "methods":null
         },
         {
            "created_at":1553667178,
            "updated_at":1553667178,
            "strip_path":true,
            "service":{
               "id":"fe364cb8-2ff9-447f-88e6-f4e21b8d9022"
            },
            "name":null,
            "hosts":null,
            "id":"b5134b99-b506-4e35-aa94-f4133daf3c33",
            "preserve_host":false,
            "regex_priority":0,
            "paths":[
               "\/user-service"
            ],
            "sources":null,
            "destinations":null,
            "snis":null,
            "protocols":[
               "http",
               "https"
            ],
            "methods":null
         }
      ]
   }
```


- **Known behavior about failed kong-migration pod**
  For PCF deployment, **kong-migration** job startup quickly for first time deployment. However, it starts so quickly before the kong-database pod is ready. Few kong-

migration pod may fail until kong database pod available. Check pod log and to troubleshoot this issue, refer to Cassandra Data Lost.

4
Upgrading Policy Control Function

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

- Update an existing parameter settings.
- Add more parameters as per the requirement

To upgrade PCF:

```
helm upgrade <NAME>-pcf \
--set
global.envMysqlHost=<MYSQL_HOST>,global.envMysqlUser=pcfusr,global.envMysqlPasswo
rd=pcfpasswd \
--set global.envJaegerAgentHost=<JAEGER_SERVICE>.<JAEGER_SERVICE_NAMESPACE> \
--set
global.envManageNF=PCF,global.envSystemName=PCF,common.configmapApplicationConfig
.nrfClientType=PCF \
--set
common.configmapApplicationConfig.nrfDeployName=<NAME>,common.configmapApplicatio
nConfig.nrfDeployNamespace=<NAMESPACE> \
--set
global.imageTag=<IMAGE_TAG>,global.dockerRegistry=<DOCKER_REGISTRY_ADDRESS> \
--set
platform.enabled=true,pcf.enabled=true,bsf.enabled=false,common.enabled=true,nef.
enabled=false \
--set global.envDefaultBsfDeployName=<NAME>-
pcf,global.envDefaultBsfDeployNS=<NAMESPACE>-
pcf,common.deploymentNrfClientservice.envNamespace=<NAMESPACE>-pcf \
./<HELM_CHART_NAME_WITH_EXTENSION>
```

> **Note:**
>
> <NAME> and <NAMESPACE> must be same as helm install command

> **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 lost and use default settings from **values.yaml** file for missing parameters in upgrade command.

For specific deployment, few parameters cannot be updated. Table 4-1 provides details of the parameters which cannot be updated.

Table 4-1 Parameters

Deploymnet	Parameter	Description
PCF	platform.enabled=true,pcf.enabled=true,bsf.enabled=false,common.enabled=true,nef.enabled=false	Module enable/disable setting flag indicates which module should be startup, if one module is enabled for deployment, then all the services under the module are picked up and startup by kubernetes. For PCF deployment, following modules must be enabled: • platform • pcf • common
PCF	global.envManageNF=PCF,global.envSystemName=PCF,common.configmapApplicationConfig.nrfClientType=PCF	Since GUI service is an common service which defined under common module, it requires the startup parameters to show which NF should be displayed under specific deployment. Same process logic applies to **nrfclien**t service.

MetalLB Settings for Upgrade

After executing the helm upgrade command, the configured MetalLB settings may lost. User is required to update the settings manually by following the procedure in the Enabling LoadBalancer with MetalLB.

5
Uninstalling Policy Control Function

To uninstall or completely delete the Policy Control Function deployment, execute the following command:

```
helm delete --purge <helm_release_name_for_ocpcf>
```

6
Troubleshooting Policy Control Function

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

Cannot Access MySQL

Issue:

Bunch pods are under Init status. Following is an example for reference. In this scenario, database cannot access with user specified in helm install command.

```
<DEPLOY_NAME>-pcf-api-gateway-775cbd588f-db4dd                1/1
Running           5         15h
<DEPLOY_NAME>-pcf-kong-database-58f4477579-9rmhq              1/1
Running           0         15h
<DEPLOY_NAME>-pcf-nrf-clientservice-64d55499f8-s78h8          0/1    Init:
0/1          0         15h
<DEPLOY_NAME>-pcf-pcf-cm-service-6df4649ddf-f8bh8             0/1    Init:
0/1          0         15h
<DEPLOY_NAME>-pcf-pcf-config-7f6f7cddd5-m9b9w                 0/1    Init:
0/1          112       15h
<DEPLOY_NAME>-pcf-pcf-dbservice-7976fbdff4-gtrxk              0/1    Init:
0/1          112       15h
<DEPLOY_NAME>-pcf-ocpm-pre-756b8f5568-h487r                   0/1    Init:
0/1          0         15h
<DEPLOY_NAME>-pcf-pcf-amservice-6b886b9777-gjvv7              0/1    Init:
0/1          0         15h
<DEPLOY_NAME>-pcf-pcf-diam-connector-service-7c96b547df-lt4fd  0/1
ImagePullBackOff  0         15h
<DEPLOY_NAME>-pcf-pcf-diam-gateway-service-bfb9dcdd9-kxd48     0/1
ImagePullBackOff  0         15h
<DEPLOY_NAME>-pcf-pcf-smservice-b89d8f654-fz9l2               0/1    Init:
0/1          0         15h
<DEPLOY_NAME>-pcf-pcf-userservice-7cf7b49578-x7bkc            0/1    Init:
0/1          0         15h
<DEPLOY_NAME>-pcf-appinfo-5c7468f676-7zlv9                    1/1
Running           0         15h
<DEPLOY_NAME>-pcf-performance-847889fb64-2wxm9                0/1
ImagePullBackOff  0         15h
kong-migration-6q7zr                                          0/1
Completed         0         15h
kong-migration-gqb6n                                          0/1
Init:Error        0         15h
kong-migration-rvjk7                                          0/1
Init:Error        0         15h
```

If user clicks, pod logs icon on kubernetes dashboard to view log information, following log appears:

```
ocbsf-pcf-config-75475d5db-h99bz Waiting: PodInitializing
```

```
container "pcf-config" in pod "ocbsf-pcf-config-75475d5db-h99bz" is waiting to
start: PodInitializing
```

Solution:

Check helm install command, get values for variables **global.envMysqlUser** and **global.envMysqlPassword** (such as pcfusr/pcfpasswd), then use these values to login MySQL to check the following:

- Whether current user can login MySQL or not.
- Check database exists or not after login MySQL with that user.

Cassandra Data Lost

Issue:

API-gateway service database may lost on cassandra due to power outage, pod killing. Given that situation, kong database pod may re-startup but related kong database would lost, which leads to api-gateway service failed to start.

Phenomenon: API gateway service failed to start with status, **CrashLoopBackOff**

Reason: To start API gateway service, following components are mandatory:

- An kong-database pod to host cassandra database available
- Initial kong database to cassandra
- Import PCF/BSF/NEF services/routes data to kong to send request with specific route, then kong identifies the path to distribute the request per route name

Solution: Define the following in kubernetes:

- InitContainers part: Execute the command, `kong migrations bootstrap` to initialize kong database to cassandra.
- Containers part: Import PCF/BSF/NEF services/routes data via kong API to cassandra database.

Following is a sample file.

```
apiVersion: batch/v1
kind: Job
metadata:
  name: kong-migration
spec:
  template:
    metadata:
      annotations:
        sidecar.istio.io/inject: 'false'
    spec:
      initContainers:
        - name: kong-migration
          image: kong:latest
          command: ["kong"]
          args: ["migrations","bootstrap"]
          envFrom:
            - configMapRef:
                name: kong-config
      containers:
```

```
        - name: kong-data-initial
          image: <DOCKER_REGISTRY>/ocpm_ol7_jdk11:<IMAGE_TAG>
          command: ["/bin/sh"]
          args:
          - -c
          - >
            bash -c
            "
             curl -s -X POST http://<DEPLOY_NAME>-api-gateway:8001/routes -d
"paths[]=/sm-service" -d "name=sm-route" \
             -d "service.id=$(curl -s -X POST http://<DEPLOY_NAME>-api-gateway:
8001/services -d "name=sm-service" \
             -d "url=http://<DEPLOY_NAME>-pcf-smservice:5809" | python -c
'import sys, json; print json.load(sys.stdin)["id"]')"

             curl -s -X POST http://<DEPLOY_NAME>-api-gateway:8001/routes -d
"paths[]=/user-service" \
             -d "service.id=$(curl -s -X POST http://<DEPLOY_NAME>-api-gateway:
8001/services -d "name=user-service" \
             -d "url=http://<DEPLOY_NAME>-pcf-userservice:5808" | python -c
'import sys, json; print json.load(sys.stdin)["id"]')"

             curl -s -X POST http://<DEPLOY_NAME>-api-gateway:8001/routes -d
"paths[]=/bsf-service" \
             -d "service.id=$(curl -s -X POST http://<DEPLOY_NAME>-api-gateway:
8001/services -d "name=bsf-service" \
             -d "url=http://<DEPLOY_NAME>-bsf-management-service:5903" | python -
c 'import sys, json; print json.load(sys.stdin)["id"]')"

             curl -s -X POST http://<DEPLOY_NAME>-api-gateway:8001/routes -d
"paths[]=/nrf-client" \
             -d "service.id=$(curl -s -X POST http://<DEPLOY_NAME>-api-gateway:
8001/services -d "name=nrf-client" \
             -d "url=http://<DEPLOY_NAME>-nrf-clientservice:5910" | python -c
'import sys, json; print json.load(sys.stdin)["id"]')"

             curl -s -X POST http://<DEPLOY_NAME>-api-gateway:8001/routes -d
"paths[]=/sm-ingress" \
             -d "service.id=$(curl -s -X POST http://<DEPLOY_NAME>-api-gateway:
8001/services -d "name=sm-ingress" \
             -d "url=http://istio-ingressgateway.istio-system.svc:80" | python -
c 'import sys, json; print json.load(sys.stdin)["id"]')"

             curl -s -X POST http://<DEPLOY_NAME>-api-gateway:8001/routes -d
"paths[]=/sm-service" -d "name=sm-route" \
             -d "service.id=$(curl -s -X POST http://<DEPLOY_NAME>-api-gateway:
8001/services -d "name=sm-service" \
             -d "url=http://<DEPLOY_NAME>-pcf-smservice:5809" | python -c
'import sys, json; print json.load(sys.stdin)["id"]')"

             curl -s -X POST http://<DEPLOY_NAME>-api-gateway:8001/routes -d
"paths[]=/am-service" \
             -d "service.id=$(curl -s -X POST http://<DEPLOY_NAME>-api-gateway:
8001/services -d "name=am-service" \
             -d "url=http://<DEPLOY_NAME>-pcf-amservice:5904" | python -c
'import sys, json; print json.load(sys.stdin)["id"]')"
            "
      restartPolicy: Never
```

> **Note:**
>
> Above job is one time job which would be executed only once with helm install command. If encounter power outage or kong database pod deletion, then kong database would lost.

Perform the following:

1. Click **Jobs** in kubernetes dashboard.
2. Delete existing kong-migration job.
3. Click **Create** to create a new job.
4. Paste above job definition content to create new job dialog, pay attention to replace the following variables.

Table 6-1 Variables

Variables Name	Variable Value
DOCKER_REGISTRY	Registry value
DEPLOY_NAME	<NAME> value specified in helm install command, for example: `helm install --namespace=<NAMESPACE> --name= <NAME>`
IMAGE_TAG	The image tag value for docker image ocpm_ol7_jdk11

5. Click Pods in kubernetes dashboard, an kong-migration pod startsup.
6. Delete existing api-gateway pod with status **CrashLoopBackOff**, an new api gateway pod startsup.
7. Wait for a while then new launched api gateway pod status runs.

Kubernetes DNS Error

Issue:

API gateway service cannot access kong database and following is the error log:

```
nginx: [error] [lua] log.lua:68: log(): could not resolve Cassandra contact point
'ocbsf-kong-database': dns server error: 3 name error
```

Solution: In general, one pod can access another pod under same namespace in kubernetes environment.

Perform the following workaround:

1. Click **Service** and open service page to get cluster IP of **<*NAME*>-kong-database**, such as **ocbsf-kong-database**.
2. Click **configmap** and replace **<NAME>-kong-database** with **<CLUSTER_IP>**.
3. Save changes to kong-config map.
4. Wait for a while until api-gateway service startup, keep an eye to output log,

5. If API gateway still fails to start, check output log of api-gateway service, if the following log appears.

```
2019/04/09 06:52:04 [error] 1#0: init_by_lua error: /usr/local/share/lua/5.1/
kong/init.lua:337: database needs bootstrap; run 'kong migrations bootstrap'
```

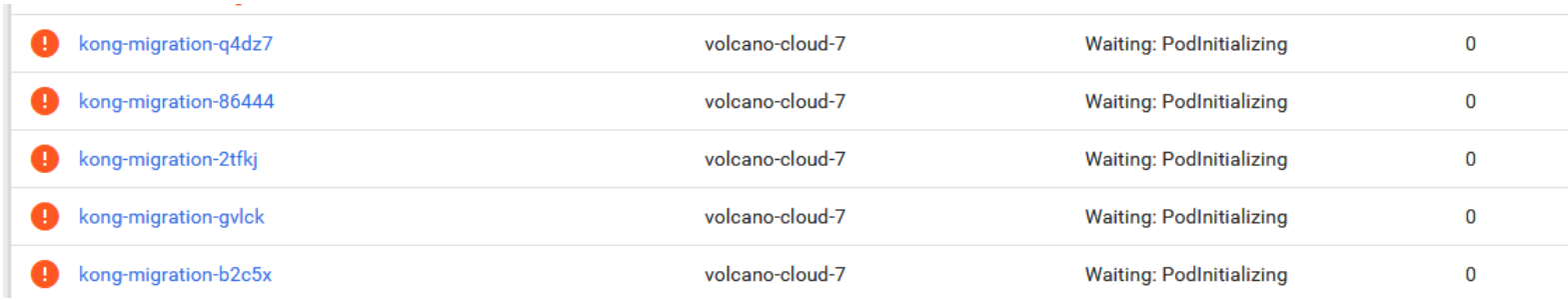
Perform the steps mentioned in the solution section of Cassandra Data Lost.

It is possible that the kong-migration job fails with this DNS error. You can view this from pods list by following status:

- No kong-migration pod execute successful with status, **Terminated: Completed**
- All kong-migration related pods are under status, **Waiting: PodInitializing**

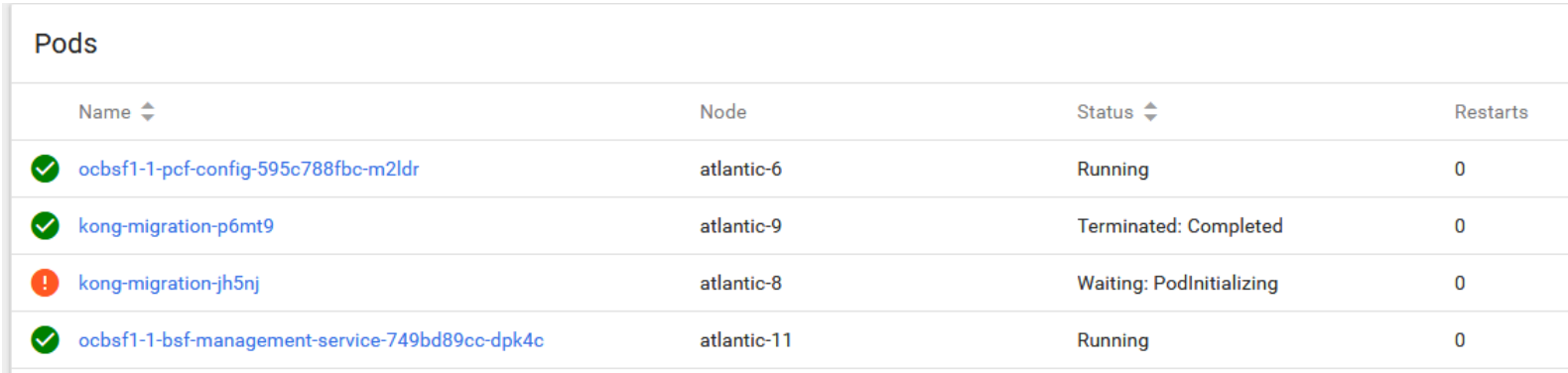
Following are example screenshots

Figure 6-1 DNS Error - Waiting



kong-migration-q4dz7	volcano-cloud-7	Waiting: PodInitializing	0
kong-migration-86444	volcano-cloud-7	Waiting: PodInitializing	0
kong-migration-2tfkj	volcano-cloud-7	Waiting: PodInitializing	0
kong-migration-gvlck	volcano-cloud-7	Waiting: PodInitializing	0
kong-migration-b2c5x	volcano-cloud-7	Waiting: PodInitializing	0

Figure 6-2 DNS Error - Others



Pods

Name	Node	Status	Restarts
ocbsf1-1-pcf-config-595c788fbc-m2ldr	atlantic-6	Running	0
kong-migration-p6mt9	atlantic-9	Terminated: Completed	0
kong-migration-jh5nj	atlantic-8	Waiting: PodInitializing	0
ocbsf1-1-bsf-management-service-749bd89cc-dpk4c	atlantic-11	Running	0

Service Status CrashLoopBackOff

Issue: Few service pods such as **pcf-amservice**, **pcf-smservice**, or **pcf-config** cannot startup with status **CrashLoopBackOff**. For example,

```
ocbsf-api-gateway-678cbb84dc-mmhzk Waiting: CrashLoopBackOff
```

Click pod log icon on kubernetes dashboard to view log information. Following are some of the known reasons for this error.

MySql Error: Too many connections

Caused by **com.mysql.cj.exceptioens.CJExcption**. Data source rejected establishment of connection, message from server displays as **Too many connections**.

Solution: Perform the following steps to apply changes to MySQL.

```
Logon mysql console via "mysql -h<MYSQL_HOST> -u<MYSQL_USER> -p<MYSQL_PWD>" then
execute below command:

--> show variables like "max_connections";

This will return you something like this.
+-----------------+-------+
| Variable_name   | Value |
+-----------------+-------+
| max_connections | 151   |
+-----------------+-------+

You can change the setting to e.g. 200 by issuing the following command without
having to restart the MySQL server.
--> set global max_connections = 200;

Restart MySQL the next time it will use this setting instead of the default.
```

For more information, refer to https://dev.mysql.com/doc/refman/5.5/en/too-many-connections.html

Application Program Error

Application cannot startup as exception error occurs.

Solution:

Open pod log and check detailed exception stack trace. User need to contact Administrator to resolve the issue.

Service Status ErrImagePull Error

Issue: Few service pods such as **pcf-diam-connector-service**, or **pcf-smservice, pcf-config** cannot startup with status ErrImagePull. For example, `ocbsf-bsf-diam-connector-service-5944544cd7-8dvzq 0/1 ErrImagePull`

Solution: cannot pull service docker image from docker registry, either docker image does not exist or docker registry cannot access.

Run the following command to verify if image exists.

```
docker pull <DOKCER_REGISTRY>(:<REGISTRY_PORT>)/<SERVICE_NAME>:<SERVICE_TAG>
```

If image is not found, contact your system administrator or Oracle Support.

Performance Service Fails to Start

Issue: Performance service failed to start with status CrashLoopBackOff.

Solution: As an performance monitoring service to monitor current deployment, the target is to collect performance data such as CPU, and memory usage. If not configured properly, it cannot

startup successfully. Following parameters are mandatory for it to work, and user can find them in **configmap** named **perinfo-config-<NAME>**.

Table 6-2 Service Parameters

Variable Name	Varaible Value
prometheus	prometheus servie URL. For example, http://prometheus-server.prometheus:5802