

Oracle® Communications

Cloud Native Binding Support Function

Installation Guide



Release 1.5.1

F35326-01

September 2020

The Oracle logo, consisting of a solid red square with the word "ORACLE" in white, uppercase, sans-serif font centered within it.

ORACLE®

F35326-01

Copyright © 2019, 2020, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

1 Introduction

References	1-1
Acronyms and Terminology	1-1

2 Installing Cloud Native Binding Support Function

Pre-Installation Tasks	2-1
Checking the Software Requirements	2-1
Checking the Environment Setup	2-2
Installation Procedure	2-4
Downloading BSF Package	2-4
Pushing the Images to Customer Docker Registry	2-5
Configuring Database, Creating Users, and Granting Permissions	2-7
Installing BSF Package	2-11
Verifying BSF Installation	2-13

3 Customizing Binding Support Function

Configuring Mandatory Parameters	3-1
Enabling/Disabling Services Configurations	3-2
Configuring Tracing Parameters	3-4
Configuring Database Names	3-6
Configuring NRF client	3-6
Configuring Diameter Gateway/Connector	3-9
API Root Configuration for Notification URI	3-10
Configuring Ingress Gateway	3-11
Configuring Service and Container Ports	3-13
OAUTH Configuration	3-19
Configuring Ingress/Egress Gateway HTTPS	3-22
Configuring SCP	3-27
Logging Configuration	3-33
Additional Configurations	3-34

4 Uninstalling Binding Support Function

5 Troubleshooting Binding Support Function

A Docker Images

B Deployment Service Type Selection

What's New in This Guide

This section introduces the documentation updates in Oracle Communications Cloud Native Binding Support Function Installation Guide.

Documentation Updates for Release 1.5.1

For Release 1.5.1, the following sections in this guide have been added or updated:

- Updated [Installation Procedure](#) to include updated installation procedure for BSF.
- Added [Configuring Service and Container Ports](#) section to include the new configurable parameters in Helm.
- Updated [Customizing Binding Support Function](#) section to include configurable parameters in Helm.
- Updated [Uninstalling Binding Support Function](#) section to describe how to uninstall Binding Support function.

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.

1

Introduction

The Binding Support Function (BSF) allows Policy Control Function (PCF) to register, update, and remove the binding information from it, and allows Network Function (NF) consumers to discover the selected Policy Control Function.

The BSF stores the binding information for certain PDU sessions and discovers the selected Policy Control Function according to the binding information. It also acts as diameter proxy agent or diameter redirect agent to Rx requests targeting an IP address of a UE to the selected Policy Control Function.

For any AF using Rx, such as P-CSCF, the Binding Support Function determines the selected Policy Control Function address according to the information carried by the incoming Rx requests.

The BSF provides a PDU session binding functionality, which ensures that an Application Function (AF) request for a certain PDU Session reaches the relevant PCF holding the PDU Session information. This service:

- Allows Policy Control Function users to register, update, and remove the binding information
- Allows NF consumers to retrieve the binding information

For more information, see Oracle Communications Cloud Native Binding Support Function User's Guide.

References

Refer to the following documents for more information about Cloud Native Binding Support Function:

- Oracle Communications Cloud Native Environment Installation Guide
- Oracle Communications Cloud Native Binding Support Function 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
CNPCRF	Cloud Native Policy and Charging Rules Function
SAN	Storage Area Network
SMF	Session Management Function
UDR	Unified Data Repository

2

Installing Cloud Native Binding Support Function

This chapter describes how to install Cloud Native Binding Support Function (BSF) on a cloud native environment.

This chapter contains the following:

- [Pre-Installation Tasks](#)
- [Installation Procedure](#)

Pre-Installation Tasks

Prior to installing the BSF, perform the following tasks:

- [Checking the Software Requirements](#)
- [Checking the Environment Setup](#)

Checking the Software Requirements

The following software must be installed before installing Binding Support Function (BSF):

 **Note:**

In this release, BSF supports Oracle Communications Cloud Native Environment (OCCNE) 1.5.

To check the OCCNE version, run the following command:

```
echo ${OCCNE_VERSION}
```

Software	Version
Kubernetes	v1.16.7
HELM	v3.0

To check the current helms and Kubernetes version installed in the CNE, run the following commands:

```
kubectl version
```

```
helm3 version
```

The following table summarizes additional software that may be needed depending on the requirement of the services:

Software	App Version	Notes
elasticsearch	7.6.1	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	7.6.1	Required for Logging
grafana	5.0.5	Required for Metrics
prometheus	11.0.2	Required for Metrics
prometheus-node-exporter	1.9.0	Required for Metrics
metallb	0.12.0	Required for External IP
metrics-server	2.10.0	Required for Metric Server
tracer	0.13.3	Required for Tracing


 **Note:**

Software, listed in the table given above, are available if the BSF is deployed in the Oracle Communications Cloud Native Environment (OCCNE). If you are deploying BSF in any other environment, the required software must be installed before installing BSF. To check the installed software items, run:

```
helm ls
```

Some of the systems may need to use helm command with **admin.conf** file. For example:

```
helm --kubeconfig admin.conf
```

 **Note:**

If you are using Network Repository Function (NRF), install it before proceeding with the BSF installation. BSF 1.5.1 supports NRF 1.8.

Checking the Environment Setup

 **Note:**

This section is applicable only when the Binding Support Function (BSF) is deployed in the environment, other than OCCNE.

Network access

The Kubernetes cluster hosts must have network access to:

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

```
helm repo update
```

Note:

Some of the systems may need to use helm command with **admin.conf** file. For example:

```
helm --kubeconfig admin.conf
```

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

```
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 BSF 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

This section lists the client machine requirements where the deployment commands are 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` and `docker` 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 Procedure

This section describes the tasks that you need to perform, in the given sequence, to be able to install Binding Support Function (BSF).

The installation procedure consists of following steps:

1. [Downloading BSF Package](#)
2. [Pushing the Images to Customer Docker Registry](#)
3. [Configuring Database, Creating Users, and Granting Permissions](#)
4. [Installing BSF Package](#)
5. [Verifying BSF Installation](#)

Downloading BSF Package

This section provides information on how to download Binding Support Function (BSF) package.

To download the BSF 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 Binding Support Function 1.5.0.0.0* from **Release** drop-down.
5. Click on **Search**. The **Patch Advanced Search Results** displays a list of releases.
6. Select the required patch from the search results. The Patch Details window opens.

7. Click on **Download**. File Download window appears.
8. Click on the zip file to download the package. Package is named as follows:
ReleaseName-pkg-ReleaseNumber.tgz

where:

ReleaseName is used to track this installation instance.

ReleaseNumber is the release number.

For example, *ocbsf-pkg-1.5.1.0.0.tgz*

Pushing the Images to Customer Docker Registry

This section describes how to push the images to customer docker registry.

To Push the images to customer docker resgistry, perform the following steps:

1. Untar the Binding Support Function (BSF) package file to get BSF docker image tar file.

```
tar -xvzf ocbsf-pkg-1.5.1.0.0.tgz
```

The directory consists of the following:

- **BSF Docker Images File:**
ocbsf-images-1.5.1.tar
- **Helm File:**
ocbsf-1.5.1.tgz
- **Readme txt File:**
Readme.txt
- **Checksum for Helm chart tgz file:**
ocbsf-1.5.1.tgz.sha256
- **Checksum for images' tgz file:**
ocbsf-images-1.5.1.tar.sha256

2. Load the **ocbsf-images-1.5.1.tar** file into the Docker system

```
docker load --input /IMAGE_PATH/ocbsf-images-1.5.1.tar
```

3. To verify that the image is loaded correctly, enter the following command:

```
docker images
```

For more information on docker images available in BSF, see [Docker Images](#).

4. Run the following set of commands to create a new tag for each imported image and push the image to the customer docker registry:

```
docker tag ocbsf/oc-config-mgmt:1.8.0 CUSTOMER_REPO/oc-config-mgmt:1.8.0  
docker push CUSTOMER_REPO/oc-config-mgmt:1.8.0
```

```
docker tag ocbsf/oc-config-server:1.8.0 CUSTOMER_REPO/oc-config-server:1.8.0  
docker push CUSTOMER_REPO/oc-config-server:1.8.0
```

```
docker tag ocbf/oc-diam-connector:1.8.1 CUSTOMER_REPO/oc-diam-connector:1.8.1
docker push CUSTOMER_REPO/oc-diam-connector:1.8.1

docker tag ocbf/configurationupdate:1.4.0 CUSTOMER_REPO/configurationupdate:1.4.0
docker push CUSTOMER_REPO/configurationupdate:1.4.0

docker tag ocbf/oc-app-info:1.8.0 CUSTOMER_REPO/oc-app-info:1.8.0
docker push CUSTOMER_REPO/oc-app-info:1.8.0

docker tag ocbf/ocingress_gateway:1.8.2 CUSTOMER_REPO/ocingress_gateway:1.8.2
docker push CUSTOMER_REPO/ocingress_gateway:1.8.2

docker tag ocbf/ocegress_gateway:1.8.2 CUSTOMER_REPO/ocegress_gateway:1.8.2
docker push CUSTOMER_REPO/ocegress_gateway:1.8.2

docker tag ocbf/oc-diam-gateway:1.8.1 CUSTOMER_REPO/oc-diam-gateway:1.8.1
docker push CUSTOMER_REPO/oc-diam-gateway:1.8.1

docker tag ocbf/oc-bsf-management:1.5.1 CUSTOMER_REPO/oc-bsf-management:1.5.1
docker push CUSTOMER_REPO/oc-bsf-management:1.5.1

docker tag ocbf/oc-query:1.8.1 CUSTOMER_REPO/oc-query:1.8.1
docker push CUSTOMER_REPO/oc-query:1.8.1

docker tag ocbf/oc-perf-info:1.8.0 CUSTOMER_REPO/oc-perf-info:1.8.0
docker push CUSTOMER_REPO/oc-perf-info:1.8.0

docker tag ocbf/nrf-client:1.3.0 CUSTOMER_REPO/nrf-client:1.3.0
docker push CUSTOMER_REPO/nrf-client:1.3.0

docker tag ocbf/oc-readiness-detector:1.5.1 CUSTOMER_REPO/oc-readiness-detector:1.5.1
docker push CUSTOMER_REPO/oc-readiness-detector:1.5.1

docker tag ocbf/configurationinit:1.4.0 CUSTOMER_REPO/configurationinit:1.4.0
docker push CUSTOMER_REPO/configurationinit:1.4.0
```

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.

Configuring Database, Creating Users, and Granting Permissions

This section describes how to configure database, create users, and grant permissions to the users.

Configure Database, Create Users, and Grant Permissions

Perform the following steps to configure MySQL database for different microservices:

1. Login to the server where the ssh keys are stored and SQL nodes are accessible.
2. Connect to the SQL nodes.
3. Log into the database as a root user.
4. Create database for different microservices by running the following commands:

```
CREATE DATABASE IF NOT EXISTS ocbsf_config_server_<identifier>;
CREATE DATABASE IF NOT EXISTS ocbsf_release_<identifier>;
```

5. Create database for BSF management service:

```
CREATE DATABASE IF NOT EXISTS ocpm_bsf_<Identifier>;

CREATE TABLE IF NOT EXISTS ocpm_bsf_<Identifier>.pcf_binding (
  binding_id binary(16) not null,
  ipv4_addr varchar(64),
  ip_domain varchar(128),
  ipv6_prefix varchar(64),
  mac_addr_48 varchar(64),
  dnn varchar(128),
  supi varchar(64),
  gpsi varchar(64),
  snssai_sd varchar(64),
  snssai_sst integer,
  created_date_time datetime(6) not null,
  json_content longblob not null,
  primary key (binding_id),
  key idx_created_date_time (created_date_time),
  key idx_ipv4Addr (ipv4_addr, created_date_time),
  key idx_ipv6Prefix (ipv6_prefix, created_date_time),
  key idx_macAddr48 (mac_addr_48, created_date_time),
  key idx_supi (supi, created_date_time),
  key idx_gpsi (gpsi, created_date_time)
);
```

 **Note:**

The script given above uses the default database name "ocpm_bsf". You may replace the default name with custom database name wherever applicable.

6. Create an admin user and grant all the necessary permissions by running the following set of commands:

```
CREATE USER 'username'@'%' IDENTIFIED BY 'password';

GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER,
REFERENCES, INDEX ON bsf_config_server_<Identifier>.* TO
'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER,
REFERENCES, INDEX ON bsf_release_<Identifier>.* TO 'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER,
REFERENCES, INDEX ON ocpm_bsf_<Identifier>.* TO 'username'@'%';
FLUSH PRIVILEGES;
```

where

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

In the below example, "bsfadminusr" is used as username, "bsfadminpasswd" is used as password. Here, all permissions are being granted to "bsfadminusr".

```
CREATE USER 'bsfadminusr'@'%' IDENTIFIED BY 'bsfadminpasswd';

GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER,
REFERENCES, INDEX ON bsf_config_server_<Identifier>.* TO
'bsfadminusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP,
ALTER, REFERENCES, INDEX ON bsf_release_<Identifier>.* TO
'bsfadminusr'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, ALTER,
REFERENCES, INDEX ON ocpm_bsf_<Identifier>.* TO 'bsfadminusr'@'%';
FLUSH PRIVILEGES;
```

7. Create an application user and grant all the necessary permissions by running the following set of commands:

```
CREATE USER 'username'@'%' IDENTIFIED BY 'password';

GRANT SELECT, INSERT, UPDATE, DELETE ON
bsf_config_server_<Identifier>.* TO 'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE ON bsf_release_<Identifier>.*
TO 'username'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE ON ocpm_bsf_<Identifier>.* TO
```



```
'username'@'%' ;  
FLUSH PRIVILEGES;
```

where

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

In the below example, "bsfusr" is used as username, "bsfpasswd" is used as password. Here, all permissions are being granted to "bsfusr".

```
CREATE USER 'bsfusr'@'%' IDENTIFIED BY 'bsfpasswd';  
  
GRANT SELECT, INSERT, UPDATE, DELETE ON  
bsf_config_server_<Identifier>.* TO 'bsfusr'@'%' ;  
GRANT SELECT, INSERT, UPDATE, DELETE ON bsf_release_<Identifier>.*  
TO 'bsfusr'@'%' ;  
GRANT SELECT, INSERT, UPDATE, DELETE ON ocpm_bsf_<Identifier>.* TO  
'bsfusr'@'%' ;  
FLUSH PRIVILEGES;
```

 **Note:**

The database name is specified in the **envMySQLDatabase** parameter for respective services in the custom-value.yaml file.

 **Note:**

It is recommended to use unique database name when multiple instances of Binding Support Function (BSF) are deployed in the network as they share the same data tier (MySQL cluster).

Verify Permissions

To verify that admin user and application user have all the permissions, you can run the following command:

```
show grants for username
```

where *username* is the name of the admin or application user.

For example:

```
show grants for bsfadminusr ;  
show grants for bsfusr ;
```

Then, exit from database and logout from MYSQL node.

Create Namespace

If a namespace does not exist, create namespace by running the following command:

```
kubectl create namespace release_namespace
```

where:

release_namespace is the deployment BSF namespace used by helm command.

For example, running the following command creates `ocbsf` namespace:

```
kubectl create namespace ocbsf
```

Create Kubernetes secret

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:

1. Create a yaml file (*yaml_file_name1*) with the application user's username and password with the syntax shown below:

```
apiVersion: v1
kind: Secret
metadata:
  name: bsf-db-pass
type: Opaque
data:
  mysql-username: YnNmdXNy
  mysql-password: YnNmcGFzc3dk
```

 **Note:**

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

2. Create a yaml file (*yaml_file_name2*) with the admin user's username and password with the syntax shown below:

```
apiVersion: v1
kind: Secret
metadata:
  name: bsf-admin-db-pass
type: Opaque
data:
  mysql-username: YnNmcHJpdmlsZWdlZHVzcg==
  mysql-password: YnNmcHJpdmlsZWdlZHBhc3N3ZA==
```

 **Note:**

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

3. Run 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 the name of the yaml file that is created in step 1.

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

For example: In the below example "application.yaml" is used as yaml file name created in step 1 and "admin.yaml" is used as a file name created in step 2, and "ocbsf" is used as a namespace:

```
kubectl create -f application.yaml -n ocbsf
```

```
kubectl create -f admin.yaml -n ocbsf
```

Installing BSF Package

This section describes how to install Binding Support Function (BSF) package.

1. Customize the custom-values.yaml file. For more information, see [Customizing Binding Support Function](#).

 **Note:**

The values of the parameters mentioned in the custom values yaml file overrides the default values specified in the helm chart. If the **envMySQLDatabase** parameter is modified, then you must modify the **configDbName** parameter with the same value.

 **Note:**

The URL syntax for **perf-info** must be in the correct syntax, otherwise it keeps on restarting. The following is an example of URL for bastion server:

```
perf-info:
  configmapPerformance:
    prometheus: http://ocne-prometheus-server.ocne-
infra.svc
```

2.

 **Caution:**

When you run the install command, make sure that you 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 as it leads to some anomalous behavior.

Install BSF using Helm2:

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

Install BSF 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 `ocbsf-pkg-1.5.1.tgz` file

release_name is the release name used by helm command. The maximum allowed length is 63 characters.

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/bsf-1.5.1.tgz --name ocbsf --
namespace ocbsf -f ocbsf-custom-values-1.5.1.yaml --atomic
```

Parameters in `helm install` command:

- **atomic:** If this parameter is set, installation process purges chart on failure. The `--wait` flag will be set automatically.
- **timeout duration** (optional): If not specified default value will be 300 (300 seconds) in Helm2 and 5m (5 minutes) in Helm3. It specifies the time to

wait for any individual kubernetes operation (like Jobs for hooks). The 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).

 **Note:**

Timeout value is not for the overall install, but for automatic purge on installation failure.

Verifying BSF Installation

This section describes how to verify if Binding Support function (BSF) is installed successfully.

To verify installation while running the install command, run the following command:

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

Press "Ctrl+C" to exit watch mode. Make sure to run the `watch` command on a different terminal.

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

Check Installation Status

To check the installation status, run any of the following commands:

For Helm2:

```
helm ls release_name
```

For example:

```
helm ls ocbsf
```

For Helm3:

```
helm3 ls release_name -n <release-namespace>
```

You should see the status as **DEPLOYED** if the deployment is successful.

To get status of jobs and pods, run the following command:

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

For example:

```
kubectl get pod -n ocbsf
```

You should see the status as Running for all the pods if the deployment is successful.



Note:

If the installation is not successful or you do not see the status as Running for all the pods, perform the troubleshooting steps mentioned under [Troubleshooting Binding Support Function](#).

3

Customizing Binding Support Function

This chapter describes how to customize the Binding Support Function (BSF) deployment in a cloud native environment.

The BSF deployment is customized by overriding the default values of various configurable parameters in the **ocbsf-custom-values-1.5.1.yaml** file.

To customize the **ocbsf-custom-values-1.5.1.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 2.3.0**.
3. Click the **Binding Support Function (BSF) Custom Template** link to download the zip file.
4. Unzip the file to get `ocbsf-custom-configTemplates-1.5.1.0.0` file that contains the **ocbsf-custom-values-1.5.1.yaml** . This file is used during installation.
5. Customize the **ocbsf-custom-values-1.5.1.yaml** file.
6. Save the updated **ocbsf-custom-values-1.5.1.yaml** file in the helm chart directory.

Note:

- All parameters mentioned as mandatory must be present in **ocbsf-custom-values-1.5.1.yaml** file.
- All fixed value parameters listed must be present in the **ocbsf-custom-values-1.5.1.yaml** file with the exact values as specified here.

Configuring Mandatory Parameters

This section describes the mandatory configurable parameters that you must customize in the **ocbsf-custom-values-1.5.1.yaml** file for successful installation of Binding Support Function (BSF).

Table 3-1 Configurable Parameters for Mandatory Configurations

Parameter	Description
global.dockerRegistry	This mandatory parameter specifies the name of the Docker registry that hosts Binding Support Function docker images. Note: The Docker registry runs in OCCNE bastion server where all OAuth docker images are loaded.

Table 3-1 (Cont.) Configurable Parameters for Mandatory Configurations

Parameter	Description
global.envMysqlHost	This mandatory parameter specifies the IP address or host name of the MySQL server where BSF databases are hosted.
global.envMysqlPort	This mandatory parameter specifies the port number of the MySQL server where BSF databases are hosted.
global.dbCredSecretName	This mandatory parameter specifies the name of the Kubernetes secret object that contains Database username and password. Default Value: ocbsf-db-pass
global.privilegedDbCredSecretName	This mandatory parameter specifies the name of the Kubernetes secret object containing Database username and password for an admin user. Default Value: ocbsf-privileged-db-pass
global.releaseDbName	This mandatory parameter specifies the name of the release database that contains details of release version. Default Value: ocbsf_release

Here is a sample configuration for mandatory parameters in custom-values.yaml file:

```
global:
# Docker registry name
  dockerRegistry: ''
# Primary MYSQL Host IP or Hostname
  envMysqlHost: ''
  envMysqlPort: ''
# K8s secret object name containing OCBSF MYSQL UserName and Password
  dbCredSecretName: 'ocbsf-db-pass'
  privilegedDbCredSecretName: 'ocbsf-privileged-db-pass'
#Release DB name containing release version details
  releaseDbName: 'ocbsf_release'
```

Enabling/Disabling Services Configurations

This section describes the configuration parameters that can be used to select the services that you want to enable/disable for your deployment.

To configure these parameters, you should configure the following configurable parameters in the custom-values.yaml file:

Table 3-2 Configurable Parameters for Enabling/Disabling the BSF Core Service

Parameter	Description
global.bsfManagementEnable	This parameter determines if the BSF core service is enabled or not. Default Value: true
global.bsfManagementVersion1Enable	
global.bsfManagementVersion2Enable	

Table 3-3 Configurable Parameters for Enabling/Disabling the NRF Client Services

Parameter	Description
global.nrfClientNfManagementEnable	This is an optional parameter. Default Value: true
global.appinfoServiceEnable	This optional parameter determines if the app info service is enabled or not. Default Value: true
global.performanceServiceEnable	This optional parameter determines if the performance service is enabled or not. Default Value: true

Table 3-4 Configurable Parameters for Enabling/Disabling the Diameter Gateway and Diameter Connector

Parameter	Description
global.diamConnectorEnable	This optional parameter determines if the diameter connector is enabled or not. Default Value: true
global.diamGatewayEnable	This optional parameter determines if the diameter gateway is enabled or not. Default Value: true

Here is a sample configuration for configurable parameters in custom-values.yaml file:

```
global:
# BSF Core Services Enable/Disable option
  bsfManagementEnable: true
  bsfManagementVersion1Enable: false
  bsfManagementVersion2Enable: false

  nrfClientNfManagementEnable: true
  appinfoServiceEnable: true
  performanceServiceEnable: true

  diamConnectorEnable: true
  diamGatewayEnable: true
```

Configuring Tracing Parameters

This section describes the configurable tracing parameters that you may customize in the `ocbsf-custom-values-1.5.1.yaml` file.

Table 3-5 Configurable Parameters for Tracing Configuration in Ingress Gateway

Parameter	Description
<code>global.envJaegerAgentHost</code>	This mandatory parameter specifies the Hostname or IP address for the jaeger agent. It is the FQDN of Jaeger Agent service running in OCCNE cluster under namespace <code>occne-infra</code> . It is written in the following format: <JAEGER_SVC_NAME>.<JAEGER_NAMESPACE>
<code>ingress-gateway.jaegerTracingEnabled</code>	Optional Parameter Default Value: true
<code>ingress-gateway.openTracing.jaeger.udpSender.host</code>	Optional Parameter Default Value: <code>occne-tracer-jaeger-agent.occne-infra</code>
<code>ingress-gateway.openTracing.jaeger.udpSender.port</code>	Optional Parameter Default Value: 6831
<code>ingress-gateway.openTracing.jaeger.probabilisticSampler</code>	Optional Parameter Default Value: 0.5

Here is a sample configuration for tracing in ingress-gateway in custom-values.yaml file:

```
jaegerTracingEnabled: true
  openTracing :
    jaeger:
      udpSender:
        # udp sender host
        host: "occne-tracer-jaeger-agent.occne-infra"
        # udp sender port
        port: 6831
      probabilisticSampler: 0.5
```

Table 3-6 Configurable Parameters for Tracing Configuration in Egress Gateway

Parameter	Description
<code>egress-gateway.jaegerTracingEnabled</code>	Optional Parameter Default Value: true

Table 3-6 (Cont.) Configurable Parameters for Tracing Configuration in Egress Gateway

Parameter	Description
egress-gateway.openTracing.jaeger.udpSender.host	Optional Parameter Default Value: occne-tracer-jaeger-agent.occne-infra
egress-gateway.openTracing.jaeger.udpSender.port	Optional Parameter Default Value: 6831
egress-gateway.openTracing.jaeger.probabilisticSampler	Optional Parameter Default Value: 0.5

Here is a sample configuration for tracing in egress-gateway in custom-values.yaml file:

```
egress-gateway:
  jaegerTracingEnabled: true
  openTracing :
    jaeger:
      udpSender:
        # udp sender host
        host: "occne-tracer-jaeger-agent.occne-infra"
        # udp sender port
        port: 6831
      probabilisticSampler: 0.5
```

To configure tracing in nrf-client-nfmanagement, you may configure the following configurable parameters in custom-value.yaml file:

Table 3-7 Configurable Parameters for Tracing Configuration in nrf-client-nfmanagement

Parameter	Description
nrf-client.nrf-client-nfmanagement.envJaegerSamplerParam	Note: You must customize this parameter only when NRF client services are enabled. Default Value: 1
nrf-client.nrf-client-nfmanagement.envJaegerSamplerType	Note: You must customize this parameter only when NRF client services are enabled. Default Value: ratelimiting
nrf-client.nrf-client-nfmanagement.envJaegerServiceName	Note: You must customize this parameter only when NRF client services are enabled. Default Value: pcf-nrf-client-nfmanagement

Here is a sample configuration for tracing under nrf-client-nfmanagement in custom-values.yaml file:

```
nrf-client-nfmanagement:
  envJaegerSamplerParam: '1'
```

```
envJaegerSamplerType: ratelimiting
envJaegerServiceName: pcf-nrf-client-nfmanagement
```

Configuring Database Names

This section describes the configuration parameters that can be used to customize the database names.

Note:

Database name specified in the custom.yaml file should be used while creating the database during installation. See [Configuring Database, Creating Users, and Granting Permissions](#).

Table 3-8 Customizable Parameters for Database Name Configuration for BSF Services

Parameter	Description
bsf-management-service.envMySQLDatabase	This parameter specifies the name of the database of BSF Management Service. Default Value: ocpm_bsf
config-server.envMySQLDatabase	This optional parameter specifies the name of the database for Config Server service. Default Value: bsf_config_server

Here is a sample configuration for configurable parameters in custom-values.yaml file:

```
bsf-management-service:
  envMySQLDatabase: 'ocpm_bsf'

config-server:
  envMySQLDatabase: bsf_config_server
```

Configuring NRF client

This section describes the configurable parameters that you may customize in the **ocbsf-custom-values-1.5.1.yaml** file for configuring NRF client

Important:

These configurations are required when NF is required to register with NRF. Before configuring NRF client configuration, make sure that NRF Client services are enabled.

Table 3-9 Configurable Parameters for NRF Client Configuration

Parameter	Description
global.deploymentNrfClientService.envNfNamespace	This mandatory parameter specifies the K8s namespace of PCF.
nrf-client.configmapApplicationConfig.profile	This mandatory parameter contains configuration parameters that goes into nrf-client's config map. See config-map table for more details.
appinfo.infraServices	Set this conditional parameter to an empty array if any one of below condition is met: <ul style="list-style-type: none"> Deploying on OCCNE 1.4 or lesser version Not deploying on OCCNE Do not wish to monitor infra services such as db-monitor service
perf-info.configmapPerformance.prometheus	This conditional parameter specifies the Prometheus server URL. Default Value: <code>http://prometheus-server.prometheus:5802</code> Note: If you do not specify any value for this parameter, PCF reported 0 loads to NRF.

Configurable parameters for NRF Client Configuration in Config-map

Parameter	Description
primaryNrfApiRoot	Primary NRF hostname and port in the following format: <http scheme>://<Hostname/IP>:<Port> This parameter can only contain valid API root. For example: <code>http://nrf1-api-gateway.svc:80</code>
SecondaryNrfApiRoot	Secondary NRF hostname and port in the following format: <http scheme>://<Hostname/IP>:<Port> This parameter can only contain valid API root. For example: <code>http://nrf2-api-gateway.svc:80</code>
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. This parameter can only contain valid ISO 8601 duration format. For example: <code>PT120S</code> .
nrfClientType	The NfType of the NF registering. The value for this parameter must be set to BSF.
nrfClientSubscribeTypes	NF Type(s) for which the NF wants to discover and subscribe to the NRF.
appProfiles	NfProfile of BSF to be registered with NRF. This parameter can only contain valid NF profile.
enableF3	Support for 29.510 Release 15.3 This parameter can only have true (default) or false as values.
enableF5	Support for 29.510 Release 15.5 This parameter can only have true (default) or false as values.

Parameter	Description
renewalTimeBeforeExpiry	Time Period (in seconds) before the Subscription Validity time expires. For example: 3600
validityTime	The default validity time (in days) for subscriptions. For example: 30
enableSubscriptionAutoRenewal	This parameter can be used to enable renewal of subscriptions automatically. This parameter can only have true (default) or false as values.
acceptAdditionalAttributes	This parameter can be used to enable additional Attributes as part of 29.510 Release 15.5. This parameter can only have true or false (default) as values.

Here is a sample configuration for NRF client in custom-values.yaml file:

```
deploymentNrfClientService:
  #K8s namespace of BSF
  envNfNamespace: ''

appinfo:
  serviceAccountName: ''
  # Set Infrastructure services to empty array if any one of below
  # condition is met
  # 1. Deploying on occne 1.4 or lesser version
  # 2. Not deploying on OCCNE
  # 3. Do not wish to monitor infra services such as db-monitor service
  # then the below mentioned attribute 'infra_services' should be
  # uncommneted and epmtly array should be passed as already mentioned.
  #infraServices: []

perf-info:
  configmapPerformance:
    prometheus: ''

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=BSF

appProfiles=[{"nfInstanceId":"25a59926-3049-479c-8954-16ce0xyz","nfType":
:"BSF","nfStatus":"REGISTERED","fqdn":"ocbsf1-2-api-
gateway.bsfl-2.svc.atlantic.morrisville.us.lab.oracle.com","priority":1,
"capacity":1,"load":2,"bsfInfo":{"ipv4AddressRanges":
[{"start":"10.0.0.1","end":"10.113.255.255"}],"ipv6PrefixRanges":
[{"start":"2800:a00:cc03::/64","end":"2800:a00:cc04::/64"}]},"nfServices
":{"serviceInstanceId":"03063893-
cf9e-4f7a-9827-111111111111","serviceName":"nbsf-management","versions":
[{"apiVersionInUri":"v1","apiFullVersion":"1.R15.1.0","expiry":"2019-08-
03T18:66:08.871+0000"}],"scheme":"http","nfServiceStatus":"REGISTERED","
fqdn":"ocbsf1-2-api-
gateway.bsfl-2.svc.atlantic.morrisville.us.lab.oracle.com","interPlmnFqdn":
null,"ipEndpoints":
[{"ipv4Address":"10.233.22.149","transport":"TCP","port":80}],"apiPrefix
":null,"allowedNfTypes":
["PCF","AF","NEF"],"priority":1,"capacity":1,"load":2}}]}]
enableF3=true
enableF5=true
renewalTimeBeforeExpiry=3600
validityTime=30
enableSubscriptionAutoRenewal=true
acceptAdditionalAttributes=false

```

Configuring Diameter Gateway/Connector

This section describes the configurable parameters that you may customize in the **ocbsf-custom-values-1.5.1.yaml** file for configuring diameter gateway and diameter connector.

Note:

You must configure the parameters listed in the following table only when diameter connector is enabled.

Table 3-10 Configurable Parameters for Diameter Connector

Parameter	Description
diam-connector.envDiameterRealm	This mandatory parameter specifies the Diameter Realm of BSF. For example: oracle.com
diam-connector.envDiameterIdentity	This mandatory parameter specifies the Diameter Host of BSF. For example: ocbsf

**Note:**

You must configure the parameters listed in the following table only when diameter gateway is enabled.

Table 3-11 Configurable Parameters for Diameter Gateway

Parameter	Description
diam-gateway.envGatewayMode	This mandatory parameter specifies the Diameter Gateway mode. For BSF, the value must be set to bsf.
diam-gateway.envGatewayDeploymentType	This mandatory parameter specifies the Diameter Gateway deployment type. For BSF, the value must be set to PCF.
diam-gateway.envDiameterRealm	This mandatory parameter specifies the Diameter Realm of BSF diameter gateway. For example, oracle.com.
diam-gateway.envDiameterIdentity	This mandatory parameter specifies the Diameter host of BSF diameter gateway. For example, oc-diam-gateway.

Here is a sample configuration in custom-values.yaml file:

```
diam-connector:
  envDiameterRealm: 'oracle.com'
  envDiameterIdentity: 'ocbsf'

diam-gateway:
  #The diam-gateway mode i.e. converged, bsf, pcf and pcrf
  envGatewayMode: bsf
  #The diam-gateway deployment type (applicable only when mode is
  converged) i.e. CONVERGED, PCF and PCRF
  envGatewayDeploymentType: PCF
  envDiameterRealm: 'oracle.com'
  envDiameterIdentity: 'oc-diam-gateway'
```

API Root Configuration for Notification URI

This section describes the configuration parameters that can be used to API Root configuration.

To configure these parameters, you should configure the following configurable parameters in the custom-values.yaml file:

Table 3-12 Configurable Parameters for Api Root Configuration for Notification URI

Parameter	Description
global.bsApiRoot	This optional parameter specifies the API root of BSF that is used in notification URLs generated by BSF's when sending request to other producer NFs. If the value is not configured for this parameter, the ingress gateway service name and port is used as default value. For example: <code>https://<Helm namespace>-ocbsf-ingress-gateway:443</code> .
global.deploymentNrfClientService.nfApiRoot	This mandatory parameter specifies Api root of BSF. Note: This parameter must be configured only when when NRF Client services are enabled. Its value should be same as the value of "global.bsApiRoot" parameter.

```
# API root of BSF that will be used in notification URLs generated by
BSF'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://<helm name>-ocbsf-ingress-
gateway:443"
global:
  bsApiRoot: ''
  deploymentNrfClientService:
    #same as bsApiRoot
    nfApiRoot: ''
```

Configuring Ingress Gateway

This section describes the configuration parameters that are required for basic configurations in Ingress Gateway.

 **Note:**

Following configurations are applicable only when ingress-gateway is enabled.

Table 3-13 Configurable Parameters for Basic Configurations in Ingress Gateway

Parameter	Description	Mandatory/Optional Parameter	Default Value
global.metallbIpAllocationEnabled	Enable or disable IP Address allocation from Metallb Pool	Optional	false

Table 3-13 (Cont.) Configurable Parameters for Basic Configurations in Ingress Gateway

Parameter	Description	Mandatory/Optional Parameter	Default Value
global.metallbIpAllocationAnnotation	Address Pool Annotation for Metallb	Optional	metallb.universe.tf/address-pool:signaling
ingress-gateway.enableIncomingHttp	Enable it to accept incoming http requests	Optional	true
ingress-gateway.ingressServer.keepAlive.enabled		Optional	false
ingress-gateway.ingressServer.keepAlive.idealTime		Optional	180 (in seconds)
ingress-gateway.ingressServer.keepAlive.count		Optional	9
ingress-gateway.ingressServer.keepAlive.interval		Optional	60 (in seconds)

Here is a sample configuration for configurable parameters in custom-values.yaml.file:

```
ingress-gateway:

  # 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-----

ingress-gateway:
#keep alive settings
  ingressServer:
    keepAlive:
      enabled: false
      idealTime: 180 #in seconds
      count: 9
      interval: 60 #in seconds

ingress-gateway:
# Enable it to accept incoming http requests
  enableIncomingHttp: true
```

Configuring Service and Container Ports

This section describes the customizations that you can make in `custom-values.yaml` file to configure service and container ports.

 **Note:**

For upgrade scenario, changing port will cause temporary service disruption.

To override the default port numbers, used by service and container ports, and customize them as per your requirements, you can configure the following configurable parameters in `custom-values.yaml` file:

Table 3-14 Customizable Parameters for Service Ports Configuration

Parameter	Description	Mandatory/ Optional Parameter	Default Value
<code>global.servicePorts.bsfManagementServiceHttp</code>	HTTP signaling port for BSF management service.	Optional	5903
<code>global.servicePorts.bsfManagementServiceHttps</code>	HTTPS signaling port for BSF management service.	Optional	8443
<code>global.servicePorts.appInfoHttp</code>	HTTP signaling port for app info. Note: The value for this port must be same as <code>svcAppInfoHttp</code>	Optional	5906
<code>global.servicePorts.cmServiceHttp</code>	HTTP signaling port for CM service.	Optional	5808
<code>global.servicePorts.configServerHttp</code>	HTTP signaling port for config server. Note: The value for this port must be same as <code>svcConfigServerHttp</code>	Optional	5807
<code>global.servicePorts.diamConnectorHttp</code>	HTTP signaling port for Diameter connector.	Optional	8080
<code>global.servicePorts.diamConnectorDiameter</code>	Port for Diameter connector.	Optional	3868
<code>global.servicePorts.diamGatewayHttp</code>	HTTP signaling port for Diameter gateway.	Optional	8080
<code>global.servicePorts.diamGatewayDiameter</code>	Port for Diameter gateway.	Optional	3868
<code>global.servicePorts.perfInfoHttp</code>	HTTP signaling port for perf info. The value for this port must be same as <code>svcPerfInfoHttp</code> .	Optional	5905
<code>global.servicePorts.queryServiceHttp</code>	HTTP signaling port for queryservice.	Optional	5805

Table 3-14 (Cont.) Customizable Parameters for Service Ports Configuration

Parameter	Description	Mandatory/ Optional Parameter	Default Value
global.servicePorts.egressGatewayHttp	HTTP signaling port for Egress Gateway. The value for this port must be same as svcEgressGatewayHttp.	Optional	8080
global.servicePorts.nrfClientNfManagementHttp	HTTP signaling port for NRF client management service. The value for this port must be same as svcNrfClientNfManagementHttp.	Optional	5910
global.servicePorts.nrfClientNfManagementHttps	HTTPS signaling port for NRF client management service. The value for this port must be same as svcNrfClientNfManagementHttps.	Optional	5805

Here is a sample of service ports configurable parameters in custom-values.yaml file:

```

servicePorts:
  bsfManagementServiceHttp: 5903
  bsfManagementServiceHttps: 8443
  # app info
  appInfoHttp: &svcAppInfoHttp 5906
  # cm service
  cmServiceHttp: 5808
  # config server
  configServerHttp: &svcConfigServerHttp 5807
  # diam connector
  diamConnectorHttp: 8080
  diamConnectorDiameter: 3868
  # diameter gateway
  diamGatewayHttp: 8080
  diamGatewayDiameter: 3868
  # perf info
  perfInfoHttp: &svcPerfInfoHttp 5905
  # query service
  queryServiceHttp: 5805
  # egress gateway
  egressGatewayHttp: &svcEgressGatewayHttp 8080
  # nrf client
  nrfClientNfManagementHttp: &svcNrfClientNfManagementHttp 5910
  nrfClientNfManagementHttps: &svcNrfClientNfManagementHttps 5805

```

Table 3-15 Customizable Parameters for Container Ports Configuration

Parameter	Description	Mandatory/ Optional Parameter	Default Value
global.containerPorts.monitoringHttp	HTTP signaling port for monitoring. Note: The value for this port must be same as containerMonitoringHttp.	Optional	9000
global.containerPorts.bsfManagementServiceHttp	HTTP signaling port for BSF Management service.	Optional	8080
global.containerPorts.bsfManagementServiceHttps	HTTPS signaling port for BSF Management service.	Optional	8443
global.containerPorts.appInfoHttp	HTTP signaling port for app info.	Optional	5906
global.containerPorts.cmServiceHttp	HTTP signaling port for CMservice.	Optional	5807
global.containerPorts.configServerHttp	HTTP signaling port for config server.	Optional	8001
global.containerPorts.diamConnectorHttp	HTTP signaling port for Diameter Connector.	Optional	8080
global.containerPorts.diamConnectorDiameter	Diameter connector.	Optional	3868
global.containerPorts.diamGatewayHttp	HTTP signaling port for Diameter Gateway.	Optional	8080
global.containerPorts.diamGatewayDiameter	Diameter gateway.	Optional	3868
global.containerPorts.perfInfoHttp	HTTP signaling port for perf-info.	Optional	5905
global.containerPorts.queryServiceHttp	HTTP signaling port for queryservice.	Optional	8081
global.containerPorts.nrfClientNfManagementHttp	HTTP signaling port for NRF client management. Note: The value for this port must be same as containerNrfClientNfManagementHttp.	Optional	8000
global.containerPorts.nrfClientNfManagementHttps	HTTPS signaling port for NRF client management. Note: The value for this port must be same as containerNrfClientNfManagementHttps.	Optional	9443
global.containerPorts.ingressGatewayHttp	HTTP signaling port for Ingress Gateway. Note: The value for this port must be same as containerIngressGatewayHttp.	Optional	8081

Table 3-15 (Cont.) Customizable Parameters for Container Ports Configuration

Parameter	Description	Mandatory/ Optional Parameter	Default Value
global.containerPorts.ingressGatewayHttps	HTTPS signaling port for Ingress Gateway. Note: The value for this port must be same as containerIngressGatewayHttps.	Optional	9443

Here is a sample of service ports configurable parameters in custom-values.yaml file:

```

containerPorts:
  bsfManagementServiceHttp: 8000
  bsfManagementServiceHttps: 8443
  monitoringHttp: &containerMonitoringHttp 9000
  # app info
  appInfoHttp: 5906
  # cm service
  cmServiceHttp: 5807
  # config server
  configServerHttp: 8001
  # diam connector
  diamConnectorHttp: 8080
  diamConnectorDiameter: 3868
  # diameter gateway
  diamGatewayHttp: 8080
  diamGatewayDiameter: 3868
  # perf info
  perfInfoHttp: 5905
  # query service
  queryServiceHttp: 8081
  # nrf client
  nrfClientNfManagementHttp: &containerNrfClientNfManagementHttp 8080
  nrfClientNfManagementHttps: &containerNrfClientNfManagementHttps
9443
  # ingress gateway
  ingressGatewayHttp: &containerIngressGatewayHttp 8081
  ingressGatewayHttps: &containerIngressGatewayHttps 9443

```

Table 3-16 Customizable Parameters for Ports Configuration in Ingress Gateway

Parameter	Description	Mandatory/ Optional Parameter	Default Value
global.publicHttpSignalingPort	HTTP/2.0 Port of ingress gateway	Optional	80

Table 3-16 (Cont.) Customizable Parameters for Ports Configuration in Ingress Gateway

Parameter	Description	Mandatory/Optional Parameter	Default Value
global.publicHttpsSignallingPort	HTTPS/2.0 Port of ingress gateway The value for this port must be set to 0 if HTTPS is disabled.	Optional	443
global.configServerPort	HTTP signaling port for config server.	Optional	Note: The value for this port must be same as svcConfigServerHttp.
ingress-gateway.ports.actuatorPort		Optional	Same value as containerMonitoringHttp
ingress-gateway.ports.containerPort		Optional	Same value as containerIngressGatewayHttp
ingress-gateway.ports.containersslPort		Optional	Same value as containerIngressGatewayHttps

Here is a sample of configurable parameters for ingress-gateway's ports in custom-values.yaml file:

```
# -----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
configServerPort: *svcConfigServerHttp

ingress-gateway:
  ports:
    actuatorPort: *containerMonitoringHttp
    containerPort: *containerIngressGatewayHttp
    containersslPort: *containerIngressGatewayHttps
```

Table 3-17 Customizable Parameters for Ports Configuration in Egress Gateway

Parameter	Description	Mandatory/ Optional Parameter	Default Value
egress-gateway.serviceEgressGateway.actuatorPort		Optional	Same value as containerMonitoringHttp
egress-gateway.serviceEgressGateway.Port		Optional	Same value as svcEgressGatewayHttp

Here is a sample of configurable parameters for egress-gateway's ports in custom-values.yaml file:

```
egress-gateway:
  serviceEgressGateway:
    actuatorPort: *containerMonitoringHttp
    port: *svcEgressGatewayHttp
```

Table 3-18 Customizable Parameters for Ports Configuration in nrf-client-nfmanagement

Parameter	Description	Mandatory/ Optional Parameter	Default Value	Value
global.nrf-client-nfmanagement.envPlatformServicePort	HTTP signaling port for app info.	Optional	5906	Same value as svcAppInfoHttp
global.nrf-client-nfmanagement.envPerformanceServicePort	HTTP signaling port for perf info.	Optional	5905	Same value as svcPerfInfoHttp
global.nrf-client-nfmanagement.envConfigServerPort	HTTP signaling port for config server.	Optional	5807	same vale as svcConfigServerHttp
global.nrf-client-nfmanagement.containerHttpPort	HTTP signaling port for NRF client discovery.	Optional	8000	Same value as containerNrfClientNfManagementHttp

Table 3-18 (Cont.) Customizable Parameters for Ports Configuration in nrf-client-nfmanagement

Parameter	Description	Mandatory/Optional Parameter	Default Value	Value
global.nrf-client-nfmanagement.containerHttpsPort	HTTPS signaling port for NRF client discovery.	Optional	9443	Same value as containerNrfClientNfManagementHttps
global.nrf-client-nfmanagement.serviceHttpPort	HTTP signaling port for NRF client discovery service.	Optional	5910	Same value as svcNrfClientNfManagementHttp
global.nrf-client-nfmanagement.serviceHttpsPort	HTTPS signaling port for NRF client discovery service.	Optional	8443	Same value as svcNrfClientNfManagementHttps

Here is a sample of configurable parameters for nrf-client-nfmanagement's ports in custom-values.yaml file:

```
nrf-client-nfmanagement:
  envPlatformServicePort: *svcAppInfoHttp
  envPerformanceServicePort: *svcPerfInfoHttp
  envCfgServerPort: *svcConfigServerHttp
  containerHttpPort: *containerNrfClientNfManagementHttp
  containerHttpsPort: *containerNrfClientNfManagementHttps
  serviceHttpPort: *svcNrfClientNfManagementHttp
  serviceHttpsPort: *svcNrfClientNfManagementHttps
```

OAUTH Configuration

This section describes the customizations that you should make in custom-value.yaml files to configure OAUTH in ingress/egress gateway.

Note:

These configurations are applicable when the Ingress/Egress Gateway is enabled and the NRF Client services are enabled.

To configure OAUTH in ingress-gateway, you should configure the following configurable parameters in custom-value.yaml file:

Table 3-19 Configurable Parameters for OAUTH Configuration in Ingress Gateway

Parameter	Description	Mandatory/Optional Parameter	Default Value
ingress-gateway.oauthValidatorEnabled	Enable or disable OAuth Validator.	Mandatory	False
ingress-gateway.nfInstanceId	NF Instance Id of service producer	Optional	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	Optional	0
ingress-gateway.nrfPublicKeyKubeSecret	Name of the secret which stores the public key(s) of NRF	Optional	
ingress-gateway.nrfPublicKeyKubeNamespace	Namespace of the NRF public key secret	Optional	
ingress-gateway.validationType	Possible values are: <ul style="list-style-type: none"> 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.	Optional	relaxed
ingress-gateway.producerPlmnMNC	MNC of the service producer	Optional	123
ingress-gateway.producerPlmnMCC	MCC of the service producer	Optional	456

Here is a sample OAUTH configurations in ingress-gateway in custom-values.yaml.file:

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

```
producerPlmnMCC: 456
# ----OAUTH CONFIGURATION - END ----
```

Table 3-20 Configurable Parameters for OAUTH Configuration in Egress Gateway

Parameter	Description	Mandatory/ Optional Parameter	Default Value
egress-gateway.oauthClient.enabled	OAuth Validator Enabled	Optional	false
egress-gateway.oauthClient.dnsSrvEnabled	Enable/Disable the DNS-SRV query to coreDNS Server	Optional	false
egress-gateway.oauthClient.httpsEnabled	Determine if https support is enabled or not which is a deciding factor for oauth request scheme and search query parameter in dns-srv request.	Optional	false
egress-gateway.oauthClient.virtualFqdn	virtualFqdn value which needs to be populated and sent in the dns-srv query.	Conditional (If dnsSrvEnabled is set to true.)	-1
egress-gateway.oauthClient.staticNrfList	List of Static NRF's	Conditional (If oAuth is enabled.)	
egress-gateway.oauthClient.nfInstanceId	NF InstanceId of Producer	Optional	fe7d992b-0541-4c7d-ab84-c6d70b1b01b1 Note: Update the parameter with actual value, if OAuth is enabled.
egress-gateway.oauthClient.consumerPlmnMNC	MNC of service Consumer	Optional	345 Note: Update the parameter with actual value, if OAuth is enabled.
egress-gateway.oauthClient.consumerPlmnMCC	MCC of service Consumer	Optional	567 Note: Update the parameter with actual value, if OAuth is enabled.

Table 3-20 (Cont.) Configurable Parameters for OAUTH Configuration in Egress Gateway

Parameter	Description	Mandatory/Optional Parameter	Default Value
egress-gateway.oauthClient.maxRetry	Maximum number of retry that need to be performed to other NRF Fqdn's in case of failure response from first contacted NRF based on the errorCodeSeries configured.	Conditional (If oAuth is enabled.)	2
egress-gateway.oauthClient.apiPrefix	apiPrefix that needs to be appended in the Oauth request flow.	Conditional (If oAuth is enabled.)	
egress-gateway.oauthClient.errorCodeSeries	Determines the fallback condition to other NRF in case of failure response from currently contacted NRF.	Conditional (If oAuth is enabled and required a different error code series.)	4XX
egress-gateway.oauthClient.retryAfter	RetryAfter value in milliseconds that needs to be set for a particular NRF Fqdn, if the error matched the configured errorCodeSeries.	Conditional (If oAuth is enabled.)	5000

Here is a sample OAUTH configurations in egress-gateway in custom-values.yaml.file:

```
# ---- Oauth Configuration - BEGIN ----
  oauthClient:
    enabled: false
    dnsSrvEnabled: false
    httpsEnabled: false
    virtualFqdn: nrf.oracle.com:80
    staticNrfList:
      - nrf1.oracle.com:80
    nfInstanceId: fe7d992b-0541-4c7d-ab84-c6d70b1b01b1
    consumerPlmnMNC: 345
    consumerPlmnMCC: 567
    maxRetry: 2
    apiPrefix: ""
    errorCodeSeries: 4XX
    retryAfter: 5000
# ---- Oauth Configuration - END ----
```

Configuring Ingress/Egress Gateway HTTPS

This section describes the customizations that you should make in custom-value.yaml files to configure HTTPS in ingress/egress gateway.

 **Note:**

These configurations are applicable only when ingress/egress gateway is enabled and the following parameters are set to true in custom-yaml file:

- `ingress-gateway.enableIncomingHttps`
- `egress-gateway.enableOutgoingHttps`

To configure HTTPS in ingress-gateway, you should configure the following configurable parameters in custom-value.yaml file:

Table 3-21 Configurable Parameters for HTTPS Configurations in Ingress Gateway

Parameter	Description	Mandatory /Optional Parameter	Default Value	Notes
<code>ingress-gateway.enableIncomingHttps</code>	To enable https for ingress traffic	Optional	False	
<code>ingress-gateway.service.ssl.privateKey.k8SecretName</code>	Name of the private key secret.	Optional	Not Applicable	required if <code>enableIncomingHttps</code> is true
<code>ingress-gateway.service.ssl.privateKey.k8NameSpace</code>	Namespace of private key.	Optional	Not Applicable	required if <code>enableIncomingHttps</code> is true
<code>ingress-gateway.service.ssl.privateKey.rsa.fileName</code>	rsa private key file name.	Optional	Not Applicable	required if <code>enableIncomingHttps</code> is true
<code>ingress-gateway.service.ssl.certificate.k8SecretName</code>	Name of the private key secret	Optional	Not Applicable	required if <code>enableIncomingHttps</code> is true
<code>ingress-gateway.service.ssl.certificate.k8NameSpace</code>	Namespace of private key	Optional	Not Applicable	required if <code>enableIncomingHttps</code> is true
<code>ingress-gateway.service.ssl.certificate.rsa.fileName</code>	rsa private key file name	Optional	Not Applicable	required if <code>enableIncomingHttps</code> is true
<code>ingress-gateway.service.ssl.caBundle.k8SecretName</code>	Name of the private key secret	Optional	Not Applicable	required if <code>enableIncomingHttps</code> is true
<code>ingress-gateway.service.ssl.caBundle.k8NameSpace</code>	Namespace of private key	Optional	Not Applicable	required if <code>enableIncomingHttps</code> is true

Table 3-21 (Cont.) Configurable Parameters for HTTPS Configurations in Ingress Gateway

Parameter	Description	Mandatory /Optional Parameter	Default Value	Notes
ingress-gateway.service.ssl.caBundle.fileName	private key file name	Optional	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.keyStorePassword.k8SecretName	Name of the privatekey secret	Optional	Not Applicable	required if enableIncomingHttp is true
ingress-gateway.service.ssl.keyStorePassword.k8Namespace	Namespace of privatekey	Optional	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.keyStorePassword.fileName	File name that has password for keyStore	Optional	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.trustStorePassword.k8SecretName	Name of the privatekey secret	Optional	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.trustStorePassword.k8Namespace	Namespace of privatekey	Optional	Not Applicable	required if enableIncomingHttps is true
ingress-gateway.service.ssl.trustStorePassword.fileName	File name that has password for trustStore	Optional	Not Applicable	required if enableIncomingHttps is true

Here is a sample HTTPS configurations in ingress-gateway in custom-values.yaml file:

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

service:
  ssl:
    privateKey:
      k8SecretName: ocbsf-gateway-secret
      k8Namespace: ocbsf
      rsa:
        fileName: rsa_private_key_pkcs1.pem
    certificate:
      k8SecretName: ocbsf-gateway-secret
      k8Namespace: ocbsf
      rsa:
        fileName: ocegress.cer
    caBundle:
```

```

k8SecretName: ocbsf-gateway-secret
k8Namespace: ocbsf
fileName: caroot.cer
keyStorePassword:
k8SecretName: ocbsf-gateway-secret
k8Namespace: ocbsf
fileName: key.txt
trustStorePassword:
k8SecretName: ocbsf-gateway-secret
k8Namespace: ocbsf
fileName: trust.txt

```

Table 3-22 Configurable Parameters for HTTPS Configurations in Egress Gateway

Parameter	Description	Mandatory /Optional Parameter	Default Value	Notes
egress-gateway.enableOutgoingHttps	Enabling it for outgoing https request	No	false	
egress-gateway.egressGwCertReloadEnabled		No	false	
egress-gateway.egressGwCertReloadPath		No	/egress-gw/store/reload	
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	
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	

Table 3-22 (Cont.) Configurable Parameters for HTTPS Configurations in Egress Gateway

Parameter	Description	Mandatory /Optional Parameter	Default Value	Notes
egress-gateway.service.ssl.caBundle.k8NameSpace	Namespace of privatekey	No	Not Applicable	
egress-gateway.service.ssl.caBundle.fileName	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	
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	

Here is a sample HTTPS configurations in egress-gateway in custom-values.yaml.file:

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

#Enabling it for egress https requests
enableOutgoingHttps: false

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

service:
  ssl:
    privateKey:
      k8SecretName: ocbsf-gateway-secret
      k8NameSpace: ocbsf
    rsa:
      fileName: rsa_private_key_pkcs1.pem
    ecdsa:
      fileName: ssl_ecdsa_private_key.pem
```



```

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

```

Configuring SCP

This section describes the customizations that you can make in custom-value.yaml files to support SCP integration.

To configure SCP integration support, you should configure the following configurable parameters in custom-value.yaml file:

Table 3-23 Configurable Parameters for SCP Configuration

Parameter	Description	Mandatory /Optional Parameter	Default Value	Notes
egress-gateway.scp.scpIntegrationEnabled	Change this to false when scp integration is not required	Mandatory	false	
egress-gateway.scp.scpRerouteEnabled	Set this flag to true if re-routing to multiple SCP instances is to be enabled. globalretry can be enabled only when scpRerouteEnabled flag is set to true.	Optional	false	
egress-gateway.globalretry.enabled	globalretry can be enabled only when scpRerouteEnabled flag is set to true. And, it is applied only when no "retries" is specified under routesConfig.	Optional	false	
egress-gateway.globalretry.retries		Optional	2	

Table 3-23 (Cont.) Configurable Parameters for SCP Configuration

Parameter	Description	Mandatory /Optional Parameter	Default Value	Notes
egress-gateway.scp.instances.http.host	SCP HTTP IP/FQDN	Optional	Not Applicable	
egress-gateway.scp.instances.http.Port	SCP HTTP PORT	Optional	80	
egress-gateway.scp.instances.http.ApiPrefix	Change this value to corresponding prefix "/" is not expected to be provided along. Applicable only for SCP with TLS enabled.	Optional	/	
egress-gateway.scp.scpDefaultScheme	Default scheme applicable when 3gpp-sbi-target-apiroot header is missing	Optional	https	
egress-gateway.K8ServiceCheck	Enable this if loadbalancing is to be done by egress instead of K8s	Optional	false	
httpsScpOnly	This is global parameter which will be taken into consideration if route (under routeConfig section) based httpsScpOnly parameter is not available. If set to true, select SCP instances for https list only. If set to false, run existing logic as per provided scheme.	Optional	false	Please note double quotes to be enclosed for values of httpsScpOnly.
httpRuriOnly	This is global parameter which will be taken into consideration if route (under routeConfig section) based httpRuriOnly parameter is not available. If set to true, change scheme of RURI to http. If set to false, don't change the scheme.	Optional	false	Please notedouble quotes to be enclosed for values of httpsScpOnly.

Table 3-23 (Cont.) Configurable Parameters for SCP Configuration

Parameter	Description	Mandatory /Optional Parameter	Default Value	Notes
routesConfig.httpRuriOnly	If set to true, change Scheme of RURI to http. If set to false, don't change the scheme.	Optional	false	Please note double quotes to be enclosed for values of httpsRuriOnly. If httpsRuriOnly under route is not present globally available value will be considered.
routesConfig.httpsScpOnly	If set to true, select SCP instances for https list only. If set to false, run existing logic as per provided scheme.	Optional	false	Please note double quotes to be enclosed for values of httpsScpOnly. If httpsScpOnly under route is not present globally available value will be considered.
egress-gateway.scp.instances.scpSets[0]	SetId for the SCP instances. Only one set of Static configuration of SCP instances are allowed to be configured. Dynamic configuration sets can be any number. Refer Custom-values file for more details.	Mandatory	false	
egress-gateway.scp.instances.scpSets[0].httpConfigs[0].host	First Scp instance HTTP IP/FQDN	Mandatory (If scp.scpIntegrationEnabled is set to true.)		More SCP instances can be configured in a similar way if required.

Table 3-23 (Cont.) Configurable Parameters for SCP Configuration

Parameter	Description	Mandatory /Optional Parameter	Default Value	Notes
egress-gateway.scp.instances.scpSets[0].httpConfigs[0].port	First Scp instance Port	Mandatory (If scp.scpIntegrationEnabled is set to true.)		
egress-gateway.scp.instances.scpSets[0].httpConfigs[0].apiPrefix	First Scp instance apiPrefix. Change this value to corresponding prefix if "/" is not expected to be provided along. Applicable only for SCP with TLS enabled.	Optional	/	Examples : XXX, Point to be noted here is that "/" is not required to be included when providing some data.
egress-gateway.scp.instances.scpSets[0].httpConfigs[0].virtualHost	This will have Http VirtualFQDN and is applicable from SetId 1 and later.	Mandatory (If DnsSrv integration is required)	Not Applicable	
egress-gateway.scp.instances.scpSets[0].httpsConfigs[0].host	First SCP instance HTTPS IP/FQDN	Mandatory (If scp.scpIntegrationEnabled is set to true.)	Not Applicable	More SCP instances can be configured in a similar way if required.
egress-gateway.scp.instances.scpSets[0].httpsConfigs[0].port	First SCP instance HTTPS Port	Mandatory (If scp.scpIntegrationEnabled is set to true.)	Not Applicable	
egress-gateway.scp.instances.scpSets[0].httpsConfigs[0].apiPrefix	First Scp instance apiPrefix. Change this value to corresponding prefix if "/" is not expected to be provided along. Applicable only for SCP with TLS enabled.	Optional	/	Examples : XXX, Point to be noted here is that "/" is not required to be included when providing some data.
egress-gateway.scp.instances.scpSets[0].httpsConfigs[0].virtualHost	This will have Http VirtualFQDN and is applicable from SetId 1 and later.	Mandatory (If DnsSrv integration is required)	Not Applicable	

Here is a sample configurations for SCP integration in custom-values.yaml file:

```
# ---- SCP Configuration - BEGIN ----
# globalretry can be enabled only when scpRerouteEnabled flag is set
to true. This is an OPTIONAL configuration. And
# it is applied only when no "retries" specified under routesConfig
globalretry:
  enabled: false
  retries: 2

#true: Select SCP instances for https list only
#false: Run existing logic as per provided scheme.
#Change the flag's accordingly. Please note double quotes to be
enclosed for values of httpsScpOnly
httpsScpOnly: "false"

#true: Means change Scheme of RURI to http
#false: Keep scheme as is.
#Change the flag's accordingly. Please note double quotes to be
enclosed for values of httpRuriOnly
httpRuriOnly: "false"

# Below is a basic route configuration for SCP. This configuration
routes all egress traffic towards SCP.
# filterName1 - (fixed value)should be set to ScpFilter
# The retry section (filterName2) is required only when there is a
need to retry the requests. Retry will be sent to secondary SCP, if no
secondary configured then retry will happen on primary.
# filterName2.name - (fixed value) should have the value ScpRetry.
# filterName2.retries - (Customizable value) number of retries can be
done for a request
# filterName2.methods - (Customizable value) HTTP request methods for
which retries should be done.
# filterName2.statuses - (Customizable value) HTTP status received on
response for which request should be retried.
# httpsScpOnly - "true" Select SCP instances for https list only,
"false" Run existing logic as per provided scheme.
# httpRuriOnly - "true" Means change Scheme of RURI to http, "false"
Keep scheme as is.

#routesConfig:
#- id: scp_route
#  uri: https://dummy.dontchange
#  path: /**
#  order: 1
#  httpsScpOnly: "false"
#  httpRuriOnly: "false"
#  filterName1: ScpFilter
#  filterName2:
#    name: ScpRetry
#    retries: 1
#    methods: GET, POST, PUT, DELETE, PATCH
#    statuses: INTERNAL_SERVER_ERROR, BAD_GATEWAY

dnsSrv:
```

```
host: 10.75.225.67
port: 32081
scheme: http
errorCodeOnDNSResolutionFailure: 425

scp:
  # Change this to true when scp integration is required. Below SCP
  configurations will take effect only when this is 'true'.
  scpIntegrationEnabled: false

  # Default scheme applicable when 3gpp-sbi-target-apiroot header is
  missing
  scpDefaultScheme: http

  # Set this flag to true if re-routing to multiple SCP instances is
  to be enabled.
  scpRerouteEnabled: false
  #globalretry can be enabled only when scpRerouteEnabled flag is set
  to true.

  # Configure the SCP instance(s) host/IP and port.
  # At least one SCP host details (under http or https) is required
  when scpIntegrationEnabled
  # In this example scp-host-1 is primary SCP and scp-host-1 is
  secondary SCP.
  instances:
    scpSets:
      - setId: 0
        httpConfigs:
          - host: scp-host-1
            port: 80
            apiPrefix: "/" # Change this value to corresponding
prefix "/" is not expected to be provided along.
          - host: scp-host-2
            port: 80
            apiPrefix: "/"
          - host: scp-host-3
            port: 80
            apiPrefix: "/"
        httpsConfigs:
          - host: scp-host-1
            port: 443
            apiPrefix: "/"
          - host: scp-host-2
            port: 443
            apiPrefix: "/"
          - host: scp-host-3
            port: 443
            apiPrefix: "/"
      - setId: 1
        httpConfigs:
          - virtualHost: xyz.test.com
            apiPrefix: "/"
        httpsConfigs:
          - virtualHost: abc.test.com
```

```

    apiPrefix: "/"
# ---- SCP Configuration - END ----

```

Logging Configuration

This section describes the customizations that you should make in custom-value.yaml files to configure logging.

To configure logging in ingress-gateway, you should configure the following configurable parameters in custom-value.yaml file:

Table 3-24 Configurable Parameters for Logging Configuration in Ingress Gateway

Parameter	Description	Mandatory /Optional Parameter	Default Value	Notes
ingress-gateway.log.level.root	Log level for root logs	No	WARN	Applicable only when ingress-gateway is enabled.
ingress-gateway.log.level.ingress	Log level for ingress logs	No	WARN	Applicable only when ingress-gateway is enabled.
ingress-gateway.log.level.oauth	Log level for oauth logs	No	WARN	Applicable only when ingress-gateway is enabled.

Here is a sample configurations for logging in ingress-gateway in custom-values.yaml file:

```

ingress-gateway:
  log:
    level:
      root: WARN
      ingress: WARN
      oauth: WARN

```

Table 3-25 Configurable Parameters for Logging Configuration in Egress Gateway

Parameter	Description	Mandatory /Optional Parameter	Default Value	Notes
egress-gateway.log.level.root	Log level for root logs	No	WARN	Applicable only when egress-gateway is enabled.
egress-gateway.log.level.egress	Log level for egress logs	No	WARN	Applicable only when egress-gateway is enabled.
egress-gateway.log.level.oauth	Log level for oauth logs	No	WARN	Applicable only when egress-gateway is enabled.

Here is a sample configurations for logging in egress-gateway in custom-values.yaml file:

```
egress-gateway:
  log:
    level:
      root: WARN
      egress: WARN
      oauth: WARN
```

Additional Configurations

This section describes the additional customizations that you can make in custom-values.yaml files of Binding Support Function.

- **Annotation to support custom extension global parameters:** To support custom extension global parameters, update the following parameters in `custom extension` under `global` section of custom values file:

```
global:
  customExtension:
    allResources:
      labels: {}
      annotations: {}

  lbServices:
    labels: {}
    annotations: {}

  lbDeployments:
```



```
labels: {}
annotations: {}

nonlbServices:
  labels: {}
  annotations: {}

nonlbDeployments:
  labels: {}
  annotations: {}
```

- **Custom container name:** You can customize the name of containers of a pod with a prefix and suffix. To do so, add the prefix and suffix to the `k8sResource` under global section of custom values file:

```
global:
  k8sResource:
    container:
      prefix: ABC
      suffix: XYZ
```

Then, after installing BSF, you will see the container names as shown below:

```
Containers:
  abcd-am-service-xyz:
```

- **Kubernetes service account name:** You can use a custom service account for all services by adding it to `appinfo` section in the custom values file:

```
appinfo:
  serviceAccountName: ocbsfsaccount
```

 **Note:**

You can create the service account and roles before the installation as well.

4

Uninstalling Binding Support Function

When you uninstall a Helm chart from your Binding Support Function (BSF) deployment, it removes only the Kubernetes objects that it created during installation.

Run any of the following commands to uninstall BSF:

For Helm2:

```
helm delete --purge <release_name>
```

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

For Helm3:

```
helm3 delete <release_name> -n <release_namespace>
```

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

For example, to uninstall a release named "ocbsf", enter this command:

```
helm delete --purge ocbsf
```

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.

For example:

```
DROP DATABASE IF EXISTS 'bsf_release_1_5';
```

5

Troubleshooting Binding Support Function

This section provides information to troubleshoot the common errors that may occur during the installation and upgrade of Binding Support 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 is 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, follow the below steps to troubleshoot.
- **`helm install` command failed in case of duplicated chart**

```
helm install /home/cloud-user/bsf_1.5.0/sprint3.1/ocbsf-1.5.0-  
sprint.3.1.tgz --name ocbsf2 --namespace ocbsf2 -f cust-ashish.yaml  
Error: release ocbsf2 failed: configmaps "perfinfo-config-ocbsf2"  
already exists
```

Here, configmap 'perfinfo-config-ocbsf2' exists multiple times, while creating Kubernetes objects after pre-upgrade hooks, this will be failed. In this case also follow the below given troubleshooting steps.

Troubleshooting steps:

1. Run the following command to cleanup the databases created by the `helm install` command :

```
DROP DATABASE IF EXISTS `ocpm_config_server_1_5`;  
DROP DATABASE IF EXISTS `ocpm_bsf_1_5`;  
DROP DATABASE IF EXISTS `bsf_release_1_5`;
```

2. Run the following 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. Run the following command to delete all kubernetes objects:

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

A

Docker Images

Cloud Native Binding Support 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 BSF product services in Kubernetes. Communication between Pods of services of BSF products are preconfigured in the Helm charts.

The following table lists the docker images for BSF:

Table A-1 Docker Images for BSF

Service Name	Docker Image Name
BSF Service	oc-bsf-management
Diameter Gateway	oc-diam-gateway
Diameter Connector	oc-diam-connector
Query Service	oc-query
NRF Client Service	nrf-client
CM Service	oc-config-mgmt
Config Server Service	oc-config-server
Readiness Check	oc-readiness-detector
Performance Monitoring Service	oc-perf-info
Application info service	app_info
Ingress Gateway	ocingress_gateway
Egress Gateway	ocegress_gateway
Ingress/Egress Gateway update configuration	configurationupdate
Ingress/Egress Gateway init configuration	configurationinit

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 worker node's IP (public IP address) 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 <i>NodeIP:NodePort</i> . Most BSF 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).