# Contents

## 1 Introduction

<table>
<thead>
<tr>
<th>Overview</th>
<th>1-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture</td>
<td>1-2</td>
</tr>
<tr>
<td>References</td>
<td>1-2</td>
</tr>
<tr>
<td>Locate Product Documentation on the Oracle Help Center Site</td>
<td>1-2</td>
</tr>
<tr>
<td>Documentation Admonishments</td>
<td>1-3</td>
</tr>
<tr>
<td>Customer Training</td>
<td>1-3</td>
</tr>
<tr>
<td>My Oracle Support</td>
<td>1-3</td>
</tr>
<tr>
<td>Emergency Response</td>
<td>1-4</td>
</tr>
</tbody>
</table>

## 2 Installing User Data Repository

<table>
<thead>
<tr>
<th>Pre-requisites</th>
<th>2-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installation Preparation</td>
<td>2-3</td>
</tr>
<tr>
<td>Deploying Cloud Native User Data Repository</td>
<td>2-3</td>
</tr>
<tr>
<td>Modifying UDR DB Name</td>
<td>2-5</td>
</tr>
</tbody>
</table>

## 3 Customizing User Data Repository

## 4 Upgrading an Existing User Data Repository Deployment

## 5 Uninstall User Data Repository

## 6 Configuring User Parameters for User Data Repository
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>Admonishments</td>
<td>1-3</td>
</tr>
<tr>
<td>6-1</td>
<td>nudr-drsservice Micro service</td>
<td>6-1</td>
</tr>
<tr>
<td>6-2</td>
<td>nudr-nrclient-service micro service</td>
<td>6-4</td>
</tr>
<tr>
<td>6-3</td>
<td>ocudr-ambassador Micro service</td>
<td>6-6</td>
</tr>
</tbody>
</table>
1

Introduction

This document provides information for installing Cloud Native User Data Repository product.

Overview

The 5G User Data Repository is one of the main key components of the 5G Service Based Architecture. User Data Repository is a converged repository which is used by other 5G Network Functions to store the data.

Oracle 5G UDR, implemented as cloud native function, offers a unified database for storing application, subscription, authentication, service authorization, policy data, session binding and Application state information. UDR is one of the main key components of the 5G Service Based Architecture. UDR provides a HTTP2 based RESTful interface for other NFs and provisioning clients to access the data stored. As per the 3GPP, the Unified Data Repository (UDR) supports the following functionality:

- Storage and retrieval of subscription data by the UDM.
- Storage and retrieval of policy data by the PCF.
- Storage and retrieval of structured data for exposure.
- Application data (including Packet Flow Descriptions (PFDs) for application detection, AF request information for multiple UEs), by the NEF.

Oracle's 5G UDR:

- Leverages a common Oracle Communications Cloud Native Framework
- Compliant to 3GPP Release 15 specification
- Has tiered architecture providing separation between the connectivity, business logic and data layers
- Uses MySQL NDB Cluster as the backend database in the Data Tier
- Registers with NRF in the 5G network, so the other NFs in the network can discover UDR through NRF.

Note:

In this release, User Data Repository supports only Policy Control Function data.

UDSF (Unstructured Data Storage Function) is the functionality which supports storage and retrieval of unstructured data by any 5G NF. The specifications of UDSF is not defined by 3GPP. This functionality is part of Oracle's 5G UDR solution.
Architecture

The Cloud Native User Data Repository architecture is based on the following three tiers:

**Connectivity Tier**
- Ambassador is used as API gateway which will receive all requests and forwards them to Nudr-drservice service in Business Tier.
- It also load balances the traffic and provides required authentication.
- It runs on Kubernetes as a microservice.

**Business Tier**
- Provides the business logic of 5G User Data Repository.
- It runs on Kubernetes and has two micro services:
  - **Nudr-drservice**: The core service which will handles flexible URI support, runtime schema validation and connects to Data Tier for DB operations.
  - **Nrf-client-service**: Handles Network Repository Function registration, heartbeat, update, and deregistration with Network Repository Function.

**Data Tier**
- Uses MySQL NDB Cluster as backend DB which provides HA and geo-redundancy capabilities.
- Runs on VMs.

References

Refer to the following documents for more information about 5G cloud native user data repository.
- CNE Installation Document
- User Data Repository User's Guide

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](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](http://www.adobe.com).

1. Access the Oracle Help Center site at [http://docs.oracle.com](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.

Documentation Admonishments

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

Table 1-1 Admonishments

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>![DANGER]</td>
<td>Danger: (This icon and text indicate the possibility of personal injury.)</td>
</tr>
<tr>
<td>![WARNING]</td>
<td>Warning: (This icon and text indicate the possibility of equipment damage.)</td>
</tr>
<tr>
<td>![CAUTION]</td>
<td>Caution: (This icon and text indicate the possibility of service interruption.)</td>
</tr>
</tbody>
</table>

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/
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.
Installing User Data Repository

This section provides instructions for installing User Data Repository.

Pre-requisites

Following are the pre-requisites you must have before proceeding with the Cloud Native User Data Repository installation.

Oracle Communications User Data Repository Software

The Oracle Communications User Data Repository software consists of:

- **Oracle Communications User Data Repository Helm Chart** - It reflects the Oracle Communications User Data Repository software version. It comes in form of a zipped tar file.
- Software images of the micro-services. They come in form of docker images and/or tar file.

The following software must be installed.

<table>
<thead>
<tr>
<th>Software</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes</td>
<td>v1.12.5</td>
</tr>
<tr>
<td>HELM</td>
<td>v2.11.0</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Software</th>
<th>Version</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>elasticsearch</td>
<td>1.21.1</td>
<td>Needed for Logging Area</td>
</tr>
<tr>
<td>elastic-curator</td>
<td>1.2.1</td>
<td>Needed for Logging Area</td>
</tr>
<tr>
<td>elastic-exporter</td>
<td>1.1.2</td>
<td>Needed for Logging Area</td>
</tr>
<tr>
<td>logs</td>
<td>2.0.7</td>
<td>Needed for Logging Area</td>
</tr>
<tr>
<td>kibana</td>
<td>1.5.2</td>
<td>Needed for Logging Area</td>
</tr>
<tr>
<td>grafana</td>
<td>2.2.0</td>
<td>Needed for Metrics Area</td>
</tr>
<tr>
<td>prometheus</td>
<td>8.8.0</td>
<td>Needed for Metrics Area</td>
</tr>
<tr>
<td>prometheus-node-exporter</td>
<td>1.3.0</td>
<td>Needed for Metrics Area</td>
</tr>
<tr>
<td>metallb</td>
<td>0.8.4</td>
<td>Needed for External IP</td>
</tr>
<tr>
<td>metrics-server</td>
<td>2.4.0</td>
<td>Needed for Metric Server</td>
</tr>
<tr>
<td>tracer</td>
<td>0.8.3</td>
<td>Needed for Tracing Area</td>
</tr>
</tbody>
</table>

**Note:**

In case any of the above services are needed and the respective software is not installed in CNE, install software before proceeding.
Create Database User/Group

The UDR uses a NDB MySQL database to store the subscriber information.

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

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

```sql
CREATE USER '<username>' identified by '<password>'; 
DROP DATABASE if exists udrdb; 
CREATE DATABASE udrdb CHARACTER SET utf8; 
GRANT SELECT, INSERT, CREATE, ALTER, DROP, LOCK TABLES, CREATE TEMPORARY TABLES, DELETE, UPDATE, EXECUTE ON udrdb.* TO '<username>' '@%'; 
USE udrdb;
```

Network access

The Kubernetes cluster hosts must have network access to:

- quay.io/datawire/ambassador docker image repository
- Local docker image repository where the Oracle Communications User Data Repository images are available
- Local helm repository where the Oracle Communications User Data Repository helm charts 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
- Helm repository must be configured on the client
- Network access to the Kubernetes cluster
- Necessary environment settings to run the `kubectl` commands. The environment should have privileges to create namespace in the Kubernetes cluster.
- Helm client installed with the `push` plugin. The environment should be configured so that the `helm install` command deploys the software in the Kubernetes cluster.

Note:

All the kubectl and helm related commands that are used in this document must be executed on a system depending on the infrastructure of the deployment. It could be user machine such as VM, server, local desktop and so on.
Installation Preparation

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

1. Download the UDR package file from Oracle Software Delivery Cloud (OSDC). Package is named as follows:
   \( <\text{nfname}>-\text{pkg-}<\text{marketing-release-number}\.tgz \)
   For example: \( \text{ocudr-pkg-1.0.0.0.0.tgz} \).

2. Untar the UDR Package File.
   \( \text{tar -xvf } <\text{nfname}>-\text{pkg-}<\text{marketing-release-number}\.tgz \)
   This command results into \( <\text{nfname}>-\text{pkg-}<\text{marketing-release-number} \) directory.
   The directory consists of following:
   - UDR Docker Images File:
     \( \text{ocudr-images-1.0.0.tar} \)
   - Helm File
     \( \text{ocudr-1.0.0.tgz} \)
   - Readme txt file
     \( \text{Readme.txt} \) (Contains cksum and md5sum of tarballs)

3. Verify the checksums of tarballs mentioned in \( \text{Readme.txt} \).

Deploying Cloud Native User Data Repository

The User Data Repository requires a MySQL database to store the configuration and run time data.

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

1. Download \( \text{ocudr-pkg-1.0.0.0.0.tgz} \).

2. Untar \( \text{ocudr-pkg-1.0.0.0.0.tgz} \).

3. Untar displays the following files:
   \( \text{ocudr-pkg-1.0.0.0.0.tgz} \)
   \( \text{| __ __ __ __ ocudr-1.0.0.tgz (helm chart)} \)
   \( \text{| __ __ __ __ ocudr-images-1.0.0.tar (docker images)} \)
   \( \text{| __ __ __ __ Readme.txt (Contains cksum and md5sum of tarballs)} \)

4. Check the checksums of tarballs mentioned in the \( \text{Readme.txt} \) file.

5. After you load the tarballs to docker images, if required, re-tag it according to your specific repository.
   - Run the following command to load the tarball to system.
     \( \# \text{ docker load --input } /\text{root}/\text{ocudr-images-1.0.0.tar} \)
6. If required, user can re-tag the images and push them according to their repo by executing the following command:

```
# docker tag ocudr/nudr_datarepository_service:1.0.0 <customer repo>/nudr_datarepository_service:1.0.0
# docker push <customer repo>/nudr_datarepository_service:1.0.0
# docker tag ocudr/nrf_client_service:1.0.0 <customer repo>/nrf_client_service:1.0.0
# docker push <customer repo>/nrf_client_service:1.0.0
```

**Deploying Cloud Native User Data Repository in Kubernetes Cluster**

**Note:**

Before proceeding to helm installation, note that the dbname parameter under MySQL section in `ocudr_values.yaml` has the default value, `udrdb` and it is recommended to use the same as the UDR deployment takes the UDR db creation and related table creation and values by connecting to NDBCLUSTER and uses the dbname as udrdb to do the same.

If user wants to configure a different DB name for UDR DB other than the default value, `udrdb`, follow the steps mentioned in the section, Modifying UDR DB Name.

To deploy in Kubernetes Cluster:

1. Execute the following command to deploy cloud native User Data repository in Kubernetes Cluster using `ocudr_values.yaml` file modified as per site configurations.

   ```
   helm install <helm chart> [--version <OCUDR version>] --name <release> --namespace <k8s namespace> -f <ocudr_values.yaml>
   ```

   In the above command:
   - `<helm chart>` - is the name of the chart which is of the form `<helm repo>/ocudr`
   - `<OCUDR version>` - is the software version (helm chart version) of the User Data Repository. This is optional. If omitted, the default is `latest` version available in helm repository.
   - `<release>` - is a name of user's choice to identify the helm deployment
   - `<k8s namespace>` - is a name of user's choice to identifying the kubernetes namespace of the User Data Repository. All the User Data Repository micro services are deployed in this kubernetes namespace.
   - `<ocudr_values.yaml>` - is the customized User Data Repository `values.yaml` file. The OCUDR `values.yaml` file is a part of customer documentation. Users needs to download the file and modify as per the user site. See Customizing User Data Repository for more details.

2. (Optional) Customize the User Data Repository by overriding the default values of various configurable parameters. See Customizing User Data Repository.

3. Update the following parameters in `ocudr_values.yaml` file:

   - `mysql`:
     - `primaryhost`: "<Primary MYSQL Node of NDB Cluster>"
     - `primaryport`: "<Port>"
- secondaryhost: "<Secondary MYSQL Node of NDB Cluster>"
- secondaryport: "<Port>"
- dbname: "<DB_name>"
- dsusername: "<Dbusername>"
- dspassword: "<Dbpassword>"
- nrfClient:
  - host:
    * baseurl: "<To connect to NRF for registration>"
    * proxy: "<Proxy setting if any to connect to NRF>"
- capacityMultiplier: "<Capacity Multiplier>"
- supirange: "<supi range for UDR>"
- priority: "<priority>"
- livenessProbeUrl: "liveness probe url of both nudr-drservice and ocudr ambassador"
- fqdn: "FQDN of nudr-drservice for NRF to use while sending request. Will be carried in registration request to NRF"

Modifying UDR DB Name

The default UDR DB name is udrdb. For using a different DB name for UDR, perform the following steps before proceeding to the helm installation of UDR.

1. Under mysql section, modify the parameter, dbname to the desired value in ocudr_values.yaml file.
2. Login to the NDBCLUSTER sql node and copy the udrSchema.sql file to the same.

   ![Note:]

   The udrSchema.sql is available in the UDR Customer Documentation.

3. Change the DB name to desired value in udrSchema.sql file in the below highlighted places.

   ```sql
   CREATE DATABASE IF NOT EXISTS udrdb CHARACTER SET utf8;
   ``

4. Login to MySQL console, create an empty database with the desired database name on the sql node, and then exit then MySQL console.
5. Run the following command from the folder where the schema is present.

   ```
   mysql -h 127.0.0.1 -username -password <UDR DB_NAME TO BE USED> < udrSchema.sql
   
   For example,
   
   mysql -h 127.0.0.1 -username -password udrdb_demo < udrSchema.sql
   ```
Customizing User Data Repository

The User Data Repository deployment could be customized by overriding the default values of various configurable parameters.

In Deploying Cloud Native User Data Repository, the MySQL host is customized.

A `ocudr_values.yaml` file could be prepared by hand to customize the parameters.

Following is an example of User Data Repository customization file.

```yaml
# Copyright 2018 (C), Oracle and/or its affiliates. All rights reserved.
ocudr:
  image:
    repository: reg-1:5000/ocudr/nudr_datarepository_service
    tag: 1.0.0
    pullPolicy: Always

  service:
    http2enabled: "true"
    type: ClusterIP   #Note: Suggested to be set as ClusterIP (default value)
    always
    port:
      http: 80
      https: 443
      management: 9000

  deployment:
    replicaCount: 2

mysql:
  primaryhost: "ocudr-mysql"
  primaryport: "3306"
  secondaryhost: "ocudr-mysql"
  secondaryport: "3306"
  dbname: "udrdb"
  dsusername: "udruser"
  dspassword: "udrDbPasswd"

hikari:
  poolsize: "25"

logging:
  level:
    root: "WARN"

nrfClient:
  host:
    baseurl: "http://ocnrf-endpoint.mynrf.svc.cluster.local/nnrf-nfm/v1/nf-instances"
    proxy:
    ssl: "false"

logging:
  level:
```

---

ORACLE

3-1
root: "WARN"

image:
  repository: reg-1:5000/ocudr/nrf_client_service
  tag: 1.0.0
  pullPolicy: Always
  capacityMultiplier: "500"
  supirange: "[\"start\": \"10000000000\", \"end\": \"20000000000\"]"
  priority: "10"
  fqdn: "ocudr-ambassador.myudr.svc.cluster.local"

ambassador:
  id: ambassador-ocudr

jaeger:
  service:
    name: "udr-jaeger-collector.default.svc.cluster.local"
    port: 9411

resources:
  requests:
    cpu: 3
  target:
    averageCpuUtil: 80

minReplicas: 1
maxReplicas: 4

ambassador:
  ambassador:
    id: ambassador-ocudr
  service:
    type: LoadBalancer
    annotations:
      metallb.universe.tf/address-pool: signaling
  adminService:
    type: LoadBalancer
    annotations:
      metallb.universe.tf/address-pool: signaling
  image:
    repository: quay.io/datawire/ambassador
    tag: 0.40.2
    pullPolicy: IfNotPresent
  resources:
    requests:
      cpu: 3
Upgrading an Existing User Data Repository Deployment

Upgrading an existing deployment replaces the running containers and pods with new ones. If there is no change in the pod configuration, it will not get replaced. Unless there is a change in the service configuration of a micro service, the service endpoints remains unchanged. For example, ClusterIP.

For the parameters that can be configurable, see Customizing User Data Repository.

Run the following command to upgrade an existing User Data Repository deployment.

```
$ helm upgrade <release> <helm chart> [--version <OCUDR version>] -f <ocudr_customized_values.yaml>
```

<release> could be found in the output of `helm list` command
<chart> is the name of the chart in the form of <repository/ocudr> e.g. reg-1/ocudr or cne-repo/ocudr
Uninstall User Data Repository

To uninstall or completely delete the User Data Repository deployment, execute the following command:

```
helm del --purge <helm_release_name_for_ocudr>
```
Configuring User Parameters for User Data Repository

The UDR micro services have configuration options. The user should be able to configure them via deployment values.yaml.

**Note:**
Some of settings default value may change.

**Note:**
- **NAME**: is the release name used in helm install command
- **NAMESPACE**: is the namespace used in helm install command
- **K8S_DOMAIN**: is the default kubernetes domain (svc.cluster.local)

**Default Helm Release Name**: ocudr

Table 6-1 table provides the parameters for nudr-drservice Micro service

<table>
<thead>
<tr>
<th>Service</th>
<th>Parameter</th>
<th>Description</th>
<th>Default Value</th>
<th>Range of Possible Values (if applicable)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>ambassador.id</td>
<td>Ambassador Identification</td>
<td>ambassador-ocudr</td>
<td>Not applicable</td>
<td>There can be multiple Ambassador installed in same cluster for different NFs. So, this is unique identification for it. Format:- ambassador-NAME</td>
</tr>
<tr>
<td>-</td>
<td>deployment.replicaCount</td>
<td>Replicas of UDR pod</td>
<td>2</td>
<td>Not applicable</td>
<td>Number of UDR pods to be maintained by replica set created with deployment</td>
</tr>
<tr>
<td>Service</td>
<td>Parameter</td>
<td>Description</td>
<td>Default Value</td>
<td>Range of Possible Values (if applicable)</td>
<td>Notes</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------</td>
<td>------------------------------------</td>
<td>---------------</td>
<td>------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>-</td>
<td>hikar.poolsize</td>
<td>Connection pool size</td>
<td>25</td>
<td>Not Applicable</td>
<td>The hikari pool connection size to be created at start up</td>
</tr>
<tr>
<td>-</td>
<td>image.pullPolicy</td>
<td>This setting will tell if image need to be pulled or not</td>
<td>Always</td>
<td>Possible Values</td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>image.repository</td>
<td>Full Image Path</td>
<td>reg-1:5000/ocudr/nudr_datarepository_service</td>
<td>Not applicable</td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>jaeger.service.name</td>
<td>Jaegar Service Name installed in CNE</td>
<td>1.0.0</td>
<td>Not applicable</td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>jaeger.service.port</td>
<td>Jaegar Service Port installed in CNE</td>
<td>9411</td>
<td>Not applicable</td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>logging.level.root</td>
<td>Log Level</td>
<td>WARN</td>
<td>Possible Values</td>
<td>Log level of the UDR pod</td>
</tr>
<tr>
<td>-</td>
<td>maxReplicas</td>
<td>Maximum Replicas</td>
<td>4</td>
<td>Not applicable</td>
<td>Maximum number of pods</td>
</tr>
<tr>
<td>-</td>
<td>minReplicas</td>
<td>Minimum Replicas</td>
<td>1</td>
<td>Not applicable</td>
<td>Minimum number of pods</td>
</tr>
<tr>
<td>-</td>
<td>mysql.dbname</td>
<td>Mysql Database name for UDR</td>
<td>udrdb</td>
<td>Not applicable</td>
<td>This is a read only parameter. Do not change the default value.</td>
</tr>
<tr>
<td>-</td>
<td>mysql.dspassword</td>
<td>Mysql password for UDR database set by MYSQL DBA</td>
<td>udrDbPasswd</td>
<td>Not applicable</td>
<td>Password example: udrpasswd</td>
</tr>
<tr>
<td>-</td>
<td>mysql.dusename</td>
<td>Mysql Username for UDR database set by MYSQL DBA</td>
<td>udruser</td>
<td>Not applicable</td>
<td></td>
</tr>
</tbody>
</table>

Table 6-1  (Cont.) nudr-drservice Micro service
Table 6-1  (Cont.) nudr-drservice Micro service

<table>
<thead>
<tr>
<th>Service</th>
<th>Parameter</th>
<th>Description</th>
<th>Default Value</th>
<th>Range of Possible Values (if applicable)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mysql.primaryhost</td>
<td>Primary MYSQL Host IP or Hostname</td>
<td>ocudr-mysql</td>
<td>Not applicable</td>
<td>UDR will connect Primary MYSQL if not available then it will connect secondary host. Format:- NAME-mysql For MYSQL Cluster use respective IP Address or Service</td>
</tr>
<tr>
<td></td>
<td>mysql.primaryport</td>
<td>Port of MYSQL Database</td>
<td>3306</td>
<td>Not applicable</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>mysql.secondaryhost</td>
<td>Secondary MYSQL Host IP or Hostname</td>
<td>ocudr-mysql</td>
<td>Not applicable</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>mysql.secondaryport</td>
<td>Port of MYSQL Database</td>
<td>3306</td>
<td>Not applicable</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>resources.request.cpu</td>
<td>Cpu Allotment for UDR pod</td>
<td>3</td>
<td>Not applicable</td>
<td>The cpu to allocated for UDR pod during deployment</td>
</tr>
<tr>
<td></td>
<td>resources.target.averageCpuUtil</td>
<td>CPU utilization limit for autoscaling</td>
<td>80</td>
<td>Not Applicable</td>
<td>CPU utilization limit for creating HPA</td>
</tr>
<tr>
<td></td>
<td>service.http2enabled</td>
<td>Enabled HTTP2 support flag for rest server</td>
<td>true/false</td>
<td>true/false</td>
<td>Enable/Disable HTTP2 support for rest server</td>
</tr>
<tr>
<td></td>
<td>service.port.http</td>
<td>HTTP port</td>
<td>80</td>
<td>Not applicable</td>
<td>The http port to be used in UDR service</td>
</tr>
<tr>
<td></td>
<td>service.port.httpss</td>
<td>HTTPS port</td>
<td>443</td>
<td>Not applicable</td>
<td>The https port to be used for UDR service</td>
</tr>
<tr>
<td></td>
<td>service.port.management</td>
<td>Management port</td>
<td>9000</td>
<td>Not applicable</td>
<td>The actuator management port to be used for UDR service</td>
</tr>
</tbody>
</table>
### Table 6-1  (Cont.) nudr-drsservice Micro service

<table>
<thead>
<tr>
<th>Service</th>
<th>Parameter</th>
<th>Description</th>
<th>Default Value</th>
<th>Range of Possible Values (if applicable)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>service.type</td>
<td>UDR service type</td>
<td>ClusterIP</td>
<td>Possible Values- ClusterIP NodePort LoadBalancer</td>
<td>The kubernetes service type for exposing UDR deployment Note: Suggested to be set as ClusterIP (default value) always</td>
</tr>
</tbody>
</table>

### Table 6-2  nudr-nrf-client-service micro service

<table>
<thead>
<tr>
<th>Service</th>
<th>Parameter</th>
<th>Description</th>
<th>Default value</th>
<th>Range or Possible Values (if applicable)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>image.repository</td>
<td>Full Image Path</td>
<td>reg-1:5000/ocudr/nrf_client_service</td>
<td>Not applicable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>image.tag</td>
<td>Tag of Image</td>
<td>1.0.0</td>
<td>Not applicable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>image.pullPolicy</td>
<td>This setting will tell if image need to be pulled or not</td>
<td>Always</td>
<td>Possible Values - Always IfNotPresent Never</td>
<td></td>
</tr>
<tr>
<td></td>
<td>host.baseurl</td>
<td>NRF url for registration</td>
<td><a href="http://ocnrfeendpoint.mynrf.svc.cluster.local/mnrfnfm/v1/ndefinstances">http://ocnrfeendpoint.mynrf.svc.cluster.local/mnrfnfm/v1/ndefinstances</a></td>
<td>Not applicable</td>
<td>Url used for udr to connect and register with NRF</td>
</tr>
<tr>
<td></td>
<td>host.proxy</td>
<td>Proxy Setting</td>
<td>NULL</td>
<td>nrfClient.host</td>
<td>Proxy setting if required to connect to NRF</td>
</tr>
<tr>
<td></td>
<td>ssl</td>
<td>SSL flag</td>
<td>false</td>
<td>true/false</td>
<td>SSL flag to enable SSL with udr nrf client pod</td>
</tr>
<tr>
<td></td>
<td>logging.level.root</td>
<td>Log Level</td>
<td>WARN</td>
<td>Possible Values - WARN INFO DEBUG</td>
<td>Log level of the UDR nrf client pod</td>
</tr>
<tr>
<td></td>
<td>capacityMultiplier</td>
<td>Capacity of UDR</td>
<td>500</td>
<td>Not applicable</td>
<td>Capacity multiplier of UDR based on number of UDR pods running</td>
</tr>
</tbody>
</table>

Tables provides information about the nudr-nrf-client-service micro service.
### Table 6-2 (Cont.) nudr-nrf-client-service micro service

<table>
<thead>
<tr>
<th>Service</th>
<th>Parameter</th>
<th>Description</th>
<th>Default value</th>
<th>Range or Possible Values (if applicable)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>supirange</td>
<td>Supi Range supported with UDR</td>
<td><code>[{&quot;start&quot;: &quot;10000000000&quot;, &quot;end&quot;: &quot;20000000000&quot;}]</code></td>
<td>Valid start and end supi range</td>
<td>Priority to be sent in registration request</td>
</tr>
<tr>
<td>-</td>
<td>priority</td>
<td>Priority</td>
<td>10</td>
<td>Priority to be sent in registration request</td>
<td>Priority to be sent in registration request</td>
</tr>
<tr>
<td>-</td>
<td>fqdn</td>
<td>UDR FQDN</td>
<td>ocudr-ambassador.myudr.svc.cluster.local</td>
<td>Not Applicable</td>
<td>FQDN to used for registering in NRF for other NFs to connect to UDR. Note: Be cautious in updating this value. Should consider namespace used for udr deployment and name resolution setting in k8s.</td>
</tr>
</tbody>
</table>

table provides information about the ocudr-ambassador Micro service
### Table 6-3  ocudr-ambassador Micro service

<table>
<thead>
<tr>
<th>Service</th>
<th>Parameter</th>
<th>Description</th>
<th>Default Value</th>
<th>Range or Possible Values (if applicable)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>ambassador.id</td>
<td>Ambassador Identification</td>
<td>ambassador-ocudr</td>
<td>Not applicable</td>
<td>There can be multiple Ambassador installed in same cluster for different NFs. So, this is unique identification for it.</td>
</tr>
<tr>
<td>-</td>
<td>image.repository</td>
<td>Full Image Path</td>
<td>quay.io/datawire/ambassador</td>
<td>Not applicable</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>image.tag</td>
<td>Tag of Image</td>
<td>0.40.2</td>
<td>Not applicable</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>image.pullPolicy</td>
<td>This setting will tell if image need to be pulled or not</td>
<td>Always</td>
<td>Possible Values - Always IfNotPresent Never</td>
<td>-</td>
</tr>
<tr>
<td>--</td>
<td>service.annotations.getambassador.io/config.ambassador_id</td>
<td>Ambassador Identification</td>
<td>ambassador-ocudr</td>
<td>Not applicable</td>
<td>There can be multiple Ambassador installed in same cluster for different NFs. So, this is unique identification for it. Format:- ambassador-NAME</td>
</tr>
<tr>
<td>--</td>
<td>podAnnotations.prometheus.io/scrape</td>
<td>Prometheus related setting</td>
<td>true</td>
<td>Possible Values - true false</td>
<td>-</td>
</tr>
<tr>
<td>--</td>
<td>podAnnotations.prometheus.io/port</td>
<td>Prometheus related setting</td>
<td>9102</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>--</td>
<td>prometheusExporter.enabled</td>
<td>Prometheus related setting</td>
<td>true</td>
<td>Possible Values - true false</td>
<td>-</td>
</tr>
<tr>
<td>--</td>
<td>resources.request.cpu</td>
<td>CPU Allotment for Ambassador pod</td>
<td>3</td>
<td>Not applicable</td>
<td>The cpu to allocated for ambassador pod during deployment</td>
</tr>
</tbody>
</table>