

Oracle® Communications

Diameter Signaling Router

Virtual Network Functions Manager Installation and User
Guide

Release 4.0

F21479-01

July 2019



Oracle Communications Diameter Signaling Router VNFM Installation and User Guide, Release 4.0.

Copyright © 2011, 2019 Oracle and/or its affiliates. All rights reserved.

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 installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. 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 Xeon 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, Opteron, the AMD logo, and the AMD Opteron 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.



CAUTION: Use only the Upgrade procedure included in the Upgrade Kit.

Before upgrading any system, please access My Oracle Support (MOS) (<https://support.oracle.com>) and review any Technical Service Bulletins (TSBs) that relate to this upgrade.

My Oracle Support (MOS) (<https://support.oracle.com>) is your initial point of contact for all product support and training needs. A representative at Customer Access Support (CAS) can assist you with MOS registration.

Call the CAS 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>.

Change History

[illegible]

Table of Contents

1. Introduction	8
1.1 References	8
1.2 Acronyms	8
1.3 Terminology	9
1.4 Limitations	9
2. Virtual Network Functions Manager Overview	10
2.1 Advantage of Using VNFM	10
3. DSR VNFM Lifecycle Management Interfaces	12
4. DSR VNFM OpenStack Prerequisites	13
5. Install and Configure the DSR VNFM	16
5.1 Access DSR VNFM Using the REST Interface	20
5.2 DSR VNFM Redundancy	21
5.3 Supported VNF's by the DSR VNFM	21
6. Upgrading DSR VNFM	22
7. Deploying DSR VNFs	23
7.1 Create a VNF Instance	23
7.2 Query VNF Instance	26
7.2.1 Query Individual VNF Instance	27
7.2.2 Query All VNF Instances	30
7.3 Instantiating the Network OAM VNF	32
7.4 Instantiating the DR Network OAM VNF	36
7.4.1 Determining the DR NOAM XMI resource IDs	36
7.5 Instantiating the Signaling VNF with Multiple XSI (1, 2 & 4 XSI Interface)	41
7.5.1 Determine the NOAM XMI Resource IDs	43
7.5.2 DSR Signaling VNF with Multiple XSI support (1, 2 & 4 xsi interface only)	43
7.6 Instantiating Multiple Signaling VNFs	67
7.7 Instantiating the APIGW VNF	68
7.8 Instantiating the IDIH VNF	71
7.8.1 Determining the Signaling IMI resource ID:	71
7.9 Instantiating the SDS Network OAM VNF	75
7.10 Instantiating the SDS DR Network OAM VNF	79
7.10.1 Determining the SDS DR NOAM XMI resource IDs	80
7.11 Instantiating the SDS Signaling VNF	84
7.11.1 Determining the Signaling IMI resource IDs	85
7.11.2 Determining the SDS NOAM XMI resource IDs	85
7.12 Scale VNF to Level (Only Scale Out)	90

7.12.1 Scale VNF to Level using InstantiationLevelId.....	91
7.12.2 Scale VNF to Level using ScaleInfo (Arbitrary Size).....	94
8. Discover Stack	98
9. Query LCM Operation	100
9.1 Query Individual LCM Operation	100
9.2 Query All LCM Operation	102
10. Terminating a VNF	107
10.1 Forceful Termination.....	107
10.2 Graceful Termination	108
11. Changing the Default Configurations	110
11.1 Changing Flavor Names.....	110
11.2 Changing Image Names.....	110
11.3 Changing Availability Zone.....	114
11.4 Changing Profile Name	114
12. Openstack Client HTTP/HTTPS Support.....	118
13. Import HTTPS/SSL Certificate into VNFM.....	120
13.1 Recombine Existing PEM Keys and Certificates into VNFM.....	120
13.2 Copy Created Certificate (vnfm_default.jks) into VNFM	121
13.3 VNFM Self Signed Certificate Generation.....	121
14. DSR NOAM IPv6 Migration	123
15. Troubleshooting VNFM.....	124
15.1 Debug VNFM	124
15.2 Enable VNFM Logs with different Log Levels (DEBUG, TRACE, WARN, ERROR).....	124
15.3 Enable VNFM after shutdown or reboot	124
15.4 Resolve HA Alarms on VNF through VNFM Deployed Setup	124
Appendix A. My Oracle Support (MOS)	126

List of Tables

Table 1. Acronyms	8
Table 2. Terminology	9
Table 3. Parameters and Definitions for VNFM Installation	17
Table 4. Supported VNFs and VMs	21
Table 5. Parameters and Definitions for VNF Instance	26
Table 6. Parameters and Definitions for Network OAM VNF	35
Table 7. Parameters and Definitions for DR Network OAM VNF	40
Table 8. Supported Instantiation Levels for DSR Signaling VNF	42
Table 9. Parameters and Definitions for Signaling VNF with Multiple XSI	63
Table 10. Parameters and Definitions for Signaling VNF with Multiple XSI	66
Table 11. Supported Instantiation levels for DSR APIGW VNF	68
Table 12. Parameters and Definitions for APIGW VNF	70
Table 13. Parameters and Definitions for IDIH VNF	74
Table 14. Parameters and Definitions for SDS Network OAM VNF	78
Table 15. Parameters and Definitions SDS DR Network OAM VNF	83
Table 16. Parameters and Definitions for SDS Signaling VNF	89
Table 17. Parameters and Definitions for Scaling VNF to Level using InstantiationLevelId	94
Table 18. Parameters and Definitions for Scaling VNF to Level using ScaleInfo	97
Table 19. Parameters and Definitions for Terminating VNF	107

List of Figures

Figure 1. ETSI MANO specification 10

Figure 2. Supported VNFs 21

Figure 3. VNF Create Instance Request..... 23

Figure 4. Query VNF Instance 26

Figure 5. VNF Instantiate Request..... 42

Figure 6. VNF Create Instance Request..... 85

Figure 7. VNF Scaling 91

Figure 8. VNF LCM Operation 100

Figure 9. Forceful Termination 107

Figure 10. Graceful Termination 109

1. Introduction

This document defines and describes the DSR Virtual Network Functions Manager (DSR VNFM). DSR VNFM is an application that helps in the quick deployment of virtual DSRs by automating the entire deployment process and making it ready to use in the shortest possible time.

The VNFM is responsible for the lifecycle management of virtual network functions (VNFs) under the control of the network function virtualization orchestrator (NFVO).

1.1 References

- DSR Cloud Benchmarking Guide
- Or-VNFM Interface defined by ETSI NFV-SOL 003
- Import a Swagger Specification/Swagger UI
- DSR Cloud Install Guide
- DSR IP Flow Document
- DSR IPv6 Migration Guide

1.2 Acronyms

An alphabetized list of acronyms used in the document.

Table 1. Acronyms

Acronym	Definition
APIGW	Application Program Interface Gateway
DA-MP	Diameter Agent Message Processor
DB	Database
DR	Disaster Recovery
DSR	Diameter Signaling Router
ETSI	European Telecommunications Standards Institute
GUI	Graphical User Interface
HA	High Availability
IP	Internet Protocol
IDIH	Integrated Diameter Intelligence Hub
LCM	Lifecycle Management
MANO	Management and Orchestration
MP	Message Processing or Message Processor
NFVO	Network Functions Virtualization Orchestrator
NOAM	Network Operations and Maintenance
OAM	Operations, Administration, and Maintenance
OHC	Oracle Help Center
OSDC	Oracle Software Delivery Cloud

Acronym	Definition
REST	Representational State Transfer
SOAM	System Operations and Maintenance
STP-MP	Signaling Transfer Point Message Processor
UDR	Usage Detail Records
UI	User Interface
NFVO	Network Function Virtualization Orchestrator
VDSR	Virtual Diameter Signaling Router
VM	Virtual Manager
VNFM	Virtual Network Functions Manager
VNF	Virtual Network Functions
XMI	External Management Interface
XSI	External Signaling Interface

1.3 Terminology

This section describes terminology as it is used within this document.

Table 2. Terminology

Term	Definition
OpenStack controller	OpenStack controller controls the selected OpenStack instance.
Postman	A tool for creating REST requests.
Swagger UI	Swagger UI allows the users to interact with the API resources.
VNF instances	VNF instances are represented by the resources. Using this resource, the client can create individual VNF instance resources, and to query VNF instances.

1.4 Limitations

- Scale-In feature is not supported.
- Terminate VNF deletes the entire stack and is not applicable for terminating a single server.
- Discover VNF stack supports
 - Stacks that are created by the VNFM templates only.
 - The stack created by VNFM GUI, Double Failure of Active VNFM and its Persistent volume.
- Diameter Configuration is required for running the traffic.
- VNF (DSR) Software upgrade/downgrades are currently not in the scope of VNFM.

2. Virtual Network Functions Manager Overview

A VNFM automates lifecycle operations for VNFs. Since, each VNF is managed independently, to deploy a DSR it requires creating and instantiating at least two VNFs (one for the network OAM VNF and one for the signaling VNF). Signaling VNFs can be instantiated any time after the network OAM has been instantiated.

The main objective of the DSR VNFM is to provide an ETSI-compliant VNFM manager. The VNFM would be helpful by:

- Automating lifecycle management (LCM) operations for DSR VNFs. Automation of these operations can reduce their execution time.
- Providing a standardized interface to easily integrate with automation clients, especially ETSI-compliant NFVOs. The DSR VNFM provides a REST API that complies with ETSI NFV-SOL 003.

The VNFM is also helpful in responding quickly to changing customer requirements and delivers solutions for those requirements in a very short time.

The following figure illustrates the interaction between various components of DSR VNFM:

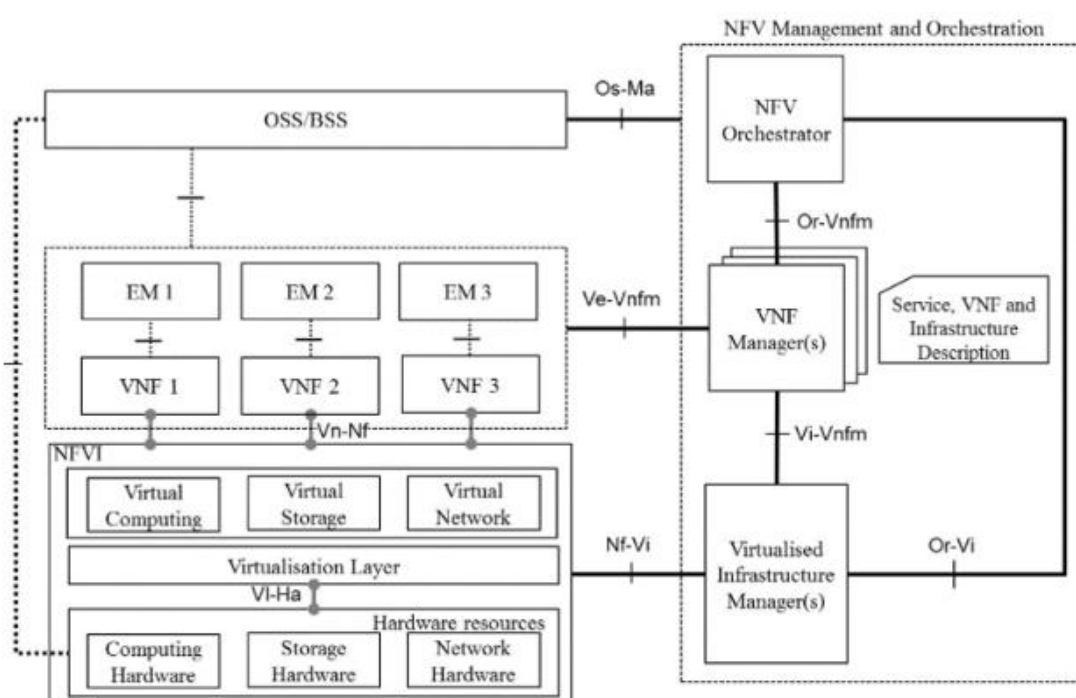


Figure 1. ETSI MANO specification

2.1 Advantage of Using VNFM

Deployment of Virtual DSR (vDSR) was performed using the following methods that required manual processing:

1. VM creation and installation process
2. HEAT Template based installation (HEAT templates require manual updates)

The manual deployment consumes multiple hours to deploy a fully operational DSR and the HEAT template based installation needed more caution since it requires more manual work.

Using DSR VNFM, users can now deploy a fully operational DSR on OpenStack in less than 15 minutes!

This application benefits both the internal and external customers by reducing operating expenses associated with the implementation and by reducing human errors by eliminating manual intervention.

3. DSR VNFM Lifecycle Management Interfaces

The DSR VNFM Lifecycle Management (LCM) interface supports the following operations:

- Create VNF
- Instantiate VNF
- Query Individual / All VNF(s)
- Scale VNF
 - Scale VNF to Level (Scale Out C Level servers of DSR Signaling VNF)
 - Scale VNF to Arbitrary size (Scale Out C Level servers of DSR Signaling VNF)
- Query Individual / All LCM Operation(s)
- Terminating VNF
- Discover VNF

4. DSR VNFM OpenStack Prerequisites

Following are the prerequisites for using the DSR VNFM:

1. An OpenStack instance, QUEEN version.
2. One OpenStack tenant per DSR Signaling VNF. The DSR network OAM VNF may share a tenant with one of the signaling VNFs, if allowed.

Note: The openstack instance must have admin privileges for multi-tenant deployments.

3. A DSR VM image must be in VMDK format as per GA release, named as:

`DSR-8.4.0.1.0_84.23.0.vmdk`

Where `DSR-8.4.0.1.0_84.23.0.ova` is the name of the OVA image delivered with the DSR build. This image must be accessible from every tenant where DSR VMs are deployed.

TPD version used by VNFM is `TPD.install-7.6.1.0.0_88.55.0-OracleLinux6.10-x86_64.qcow2`

4. DSR-specific flavors. VNFM assumes the following flavors are defined on each OpenStack tenant on which the DSR VMs are deployed.

DSR specific Flavors and respective VNF Types

VNF Type	Image Name	Flavor Name
NOAM, DSR-DBSERVER, DSR-DR-NOAM	DSR-8.4.0.1.0_84.23.0.vmdk	dsr.noam
SOAM	DSR-8.4.0.1.0_84.23.0.vmdk	dsr.soam
DA-MP	DSR-8.4.0.1.0_84.23.0.vmdk	dsr.da
IPFE	DSR-8.4.0.1.0_84.23.0.vmdk	dsr.ipfe
STP-MP	DSR-8.4.0.1.0_84.23.0.vmdk	dsr.vstp
SBR	DSR-8.4.0.1.0_84.23.0.vmdk	dsr.sbr
DSR-APIGWADMIN	DSRAPIGW-8.4.0.0.0_84.16.0.vmdk	dsrapigw.admin
DSR-APIGWAPP	DSRAPIGW-8.4.0.0.0_84.16.0.vmdk	dsrapigw.app
UDR	UDR-12.5.1.0.0_17.8.0.vmdk	udr.noam
DSR-IDIHAPP	apps-8.2.1.0.0_82.23.0.vmdk	appl-idih
DSR-IDIHMEDIATION	mediation-8.2.1.0.0_82.23.0.vmdk	med-idih

DSR-IDIHDB	oracle-8.2.1.0.0_82.23.0.vmdk	db-idih
SDS-NOAM, SDS-QS, SDS-DR-NOAM, SDS-DR-QS	SDS-8.4.0.1.0_84.23.0.vmdk	sds.noam
SDS-SOAM	SDS-8.4.0.1.0_84.23.0.vmdk	sds.dpsoam
SDS-DP	SDS-8.4.0.1.0_84.23.0.vmdk	sds.dp

For more information about flavor, see section *DSR VM Configurations* the *DSR VM Configurations* section of *DSR Benchmarking guide* for the minimum resource requirement with respect to each VNF flavor.

Note: To deploy a larger profile VM user needs to create the respective flavor in OpenStack.

Openstack Vim Connection Information

Parameter	Definition	Example
id	Unique Id of the Vim	"vimid"
vimType	Virtual Infrastructure Manager (Openstack)	"OpenStack"
controllerUri	VIM controller Identity API URI	"https://mvl-dev1.us.oracle.com:5000/v3"
username	Username to access openstack controller	"*****"
password	Password to verified credentials for openstack controller	"*****"
userDomain	User Domain name for openstack controller	"default"
projectDomain	Project Domain Id for openstack controller	"Default"
tenant	Tenant name to openstack controller	"VNFM_FT1"

Supported VNFM Network Interfaces

Node Type	IPv4	Multiple XSI	Fixed XMI	Fixed XSI/SBR	Fixed IMI	IPv6 XSI	IPv6 XMI	IPv6 IMI	Cloud-init
DSR									
DSR NOAM	Y	NA	Y	NA	Y	NA	Y	Y	Y
DR DSR NOAM	Y	NA	Y	NA	Y	NA	Y	Y	Y
DSR SOAM	Y	NA	Y	NA	Y	NA	Y	Y	Y
DAMP	Y	Y	Y	Y	Y	Y	Y	Y	Y
vSTP MP	Y	Y	Y	Y	Y	Y	Y	Y	Y
IPFE	Y	Y	Y	Y	Y	Y	Y	Y	Y
IDIH	Y	NA	Y	NA	Y	NA	N	N	Y
SBR	Y	NA	Y	Y (SBR Replication Ports)	Y	NA	Y	Y	PARTIAL
UDR NOAM	Y	Y	Y	Y	Y	Y*	Y*	Y*	Y*
SDS									
SDS NOAM	Y	NA	Y	NA	Y	NA	Y	Y	Y
Query Server	Y	NA	Y	NA	Y	NA	Y	Y	Y
DR SDS NOAM	Y	NA	Y	NA	Y	NA	Y	Y	Y
SDS SOAM	Y	NA	Y	NA	Y	NA	Y	Y	Y
DP Server	Y	NA	Y	NA	Y	NA	Y	Y	Y
APIGW									
APIGWDB	Y	NA	N	N	N	N	N	N	Y
APIGWAdmin	Y	NA	N	N	N	N	N	N	Y
APIGWAPP	Y	NA	N	N	N	N	N	N	Y

Y*: Cloud-init is not supported for IPv6 in UDR due to bug in cloud-init scripts.

5. Install and Configure the DSR VNFM

Perform the steps below to install and configure the DSR VNFM:

1. Get one Linux Box which has already installed Openstack client. If not then install Openstack client in Linux Box to get interact with Openstack through CLI.

Steps to install the Openstack client.

- a. Login as a root user and execute:

```
yum install python-devel
```

- b. Install Openstack client, by executing:

```
pip install python-openstackclient
```

- c. The above command skips importing `heatclient` plugin, install this plugin by executing:

```
pip install python-heatclient
```

2. Identify an OpenStack instance.

Note: The identified OpenStack instance must meet the DSR VNFM OpenStack Prerequisites.

- a. Download the openstack api credential file from openstack.
- b. Download the Openstack RC file.
 - i. Login to openstack GUI.
 - ii. Go to API Access section tab.
 - iii. Click on Download Openstack RC File and download (Identity API v3) file.
- c. Source the downloaded openstack API RC file in Linux BOX where openstack client is installed by executing:

```
source openrc.sh
```

When prompt for password, provide openstack controller password.

3. Download the HEAT templates for VNFM installation.

Note: Download the DSR VNFM 4.0 HEAT templates to your local disk from Oracle Help Center (OHC).

4. Upload the image file to OpenStack:
 - a. From the OpenStack GUI, navigate to **Projects > Compute-Image**.
 - b. Click **Create Image**.
 - c. In the **Create Image** dialog box, select the suggested options for the following fields:
 - i. In the **Image Source** field, select **Image File**.
 - ii. In the **Image File** field, select the **VNFM 4.0 VM** image. The VNFM Image can be obtained from Oracle Software Delivery Cloud (OSDC) Portal.
Image name:
`DSRVNFM_4.0.0.0.0_40.7.0.qcow2`
 - iii. The Minimum Disk and Minimum RAM fields can be left blank.
 - d. The VNFM flavors must be provided with the appropriate values. For information about flavors, see, the *DSR Cloud Benchmarking Guide*.

5. Create the VNFM Volume:

Creating VNFM Volume using the **OpenStack CLI**:

- a. Create the VNFM volume to use as a part of the OpenStack. The VNFM 4.0 supports a volume with the following specifications:

Volume size = 8 GB

Availability-zone = nova

For example:

```
openstack volume create --size 8 --availability-zone nova <Name of the volume>
```

The above command displays the ID assigned to the newly created volume.

Create VNFM Volume, using Openstack GUI:

- a. Navigate to **Project > Volumes – Volumes**
- b. Click **Create Volume**.
- c. In the Create Volume dialog box, perform as suggested for the following fields:

In the **Size (GiB)** field, give 8 as its size.

In the **Availability Zone** field, give nova as its value.

Get the ID of the volume created above and update the `dSrVnfmVolumeId` parameter in the `dSrVnfmParams.yaml` file.

Note:

- To change the images and flavors of VNFCs, configure the respective parameters in:
`/opt/vnfm/config/8.4/VmInfo.xml`
- To change the default properties, configure the respective parameters in:
`/opt/vnfm/config/VnfmProperties.xml`

6. Modify the input parameters:

- a. Edit the HEAT template file `dSrVnfmParams.yaml`

Note:

- The input parameters are given as key/value pairs. Only modify the values (the part to the right side of the colon).
- The formatting is very important in a YAML file. Do not remove any leading spaces or add any lines to the file.

- b. Edit the values as per the guidelines provided in Table 3.

Table 3. Parameters and Definitions for VNFM Installation

Parameter	Value
<code>dSrVnfmVmName</code>	Enter a name for the VM. Alphanumeric characters, as well as "-" and "_" are allowed.
<code>dSrVnfmImage</code>	Enter the name of the image uploaded in the previous step.
<code>dSrVnfmFlavor</code>	Enter the name of a flavor that is loaded onto OpenStack.

vnfmNetwork	Enter the name of a network that external clients can use to talk to the VNFM. (The user can also give an IP along with the network in case of fixed IP deployment) (IPv6 or IPv4)
vimNetwork	Enter the name of a network that VNFM uses to route VIM traffic. (IPv4 or IPv6)
ntpServer	Enter the IP address of an NTP server with which the VNFM synchronizes the time. The OpenStack controller hosts an NTP server so the IP address of the OpenStack controller is usually a good value.
dsrVnfmAZ	Enter the availability zone to place the VNFM. The "nova" is the default availability zone and is usually the right value.
dsrVnfmVolumeId	Enter the volume name to use as persistence storage for the VNFM.
vimRouteAddress	Enter the Openstack network address/subnet mask. This is going to be use communication between VNFM and Openstack (Vim) network. User can provide the list of route address separated by comma.

Note:

- In case of fixed IP deployment for VNFM, the network name and IP must be given in the following syntax for `vnfmNetwork` or `vimNetwork` parameter in `dsrVnfmParams.yaml` file:

```
vnfmNetwork: {"network":"ext-net2","fixed_ip":"10.196.52.175"}
vimNetwork: {"network":"ext-net2","fixed_ip":"10.196.52.176"}
```
- In case of dynamic IP deployment for VNFM, the network name should be given in the following syntax for `vnfmNetwork` or `vimNetwork` parameter in `dsrVnfmParams.yaml` file:

```
vnfmNetwork: {"network":"ext-net2"}
vimNetwork: {"network":"ext-net2"}
```
- User need to give mandatory openstack network address `vimRouteAddress` parameter in `vnfm` parameter.
Syntax: `vimRouteAddress: <openstack Network address>/<subnet mask>`

For example:

```
vimRouteAddress: 10.75.167.0/24
```

In case of list of Openstack cloud:

```
vimRouteAddress: 10.75.167.0/24,10.75.185.0/24
```

- If user is going to communicate with multiple openstack cloud with one vnfm then in that case user give provide multiple openstack network address also while installing vnfm. Otherwise user can add other openstack cloud network later also after vnfm installation through below command:

```
sudo netAdm add --route=net --device=eth1 --gateway=<gateway of eth1> --address=<openstack (VIM) network address> --netmask=<subnetmask>
```

For example:

```
sudo netAdm add --route=net --device=eth1 --gateway=10.75.157.1 --address=10.75.185.0 --netmask=24
```

c. Once editing is done, save the file.

7. Deploy the VNFM using the OpenStack CLI by executing:

```
openstack stack create -t dsrVnfmVm.yaml -e dsrVnfmParams.yaml
<stackName>
```

8. To query the DSR VNFM release details after VNFM deployment, execute:

```
$>./install_vnfm.py --info
```

DSR VNFM Tool release information:

Product Name : DSR VNFM

Product Release : 4.0

Refer the following table when choosing the IP versions:

VNFM External IP Version (REST interface) eth0	VNFM Vim IP Version (VIM interface) eth1	OpenStack Controller VIM IP	DSR IP	Comments
IPv4	IPv4	IPv4	IPv4	Supported All the openstack traffic/packet will go through VIM IP (eth1) and DSR (VNF) traffic through default route (eth0).
IPv6	IPv4	IPv4	IPv6	Supported Default route will add to both interface. Eth0 and Eth1. All the openstack traffic/packet will go through VIM IP (eth1) and DSR (VNF) traffic go through default route (eth0). Note: Not tested due to environment
IPv6	IPv4	IPv4	IPv4	Supported Default route will add to both interface, eth0 and eth1. All the openstack traffic/packet will go through VIM IP (eth1) and DSR (VNF) traffic also go through default route IPv4 (eth1) As Vnfm communicates to DSR. IPV6 cannot communicate to IPv4. So, in this case eth1 will communicate to openstack and DSR.

IPv6	IPv6	IPv4		Not Applicable. The VIM IP version and the controller IP version are different. The communication will never happen. Supported only for IPv6 controller.
IPv6	IPv6	IPv6	IPv6	Not tested The MMI call to VNFs will fail in case of IPv6. Vms will create but cloud init will fail.
IPv4	IPv6	IPv4		Not Applicable. The VIM IP version and the controller IP version are different. The communication will never happen. Supported only for IPv6 controller.

Note:

- **VNFM External IP Version (REST interface) eth0** - Vnfm external IP interface to support the VNFM rest api.
- **VNFM Vim IP Version (VIM interface) eth1** - Vnfm IP that is use to communicate to VIM controller. The `vnfm eth1` IP and `vim controller` IP should be in the same IP version, either IPv4 or IPv6.
- **OpenStack Controller VIM IP** - Openstack controller vim IP that creates the VNF through VNFM. Multiple Openstack vim controller IP can be provided during vnfm installation with vim subnet.
- **DSR IP** - DSR IP is the VNF IP. VNFM eth0 IP will communicate to DSR XMI interface for DSR cloud init LCM operation. So, DSR xmi IP and VNFM eth0 IP should be in the same IP version, either IPv4 or IPv6.

5.1 Access DSR VNFM Using the REST Interface

The DSR VNFM is accessible using a REST interface. There is no provision to access the REST interface through CLI, or GUI, however it can be accessed through a Swagger specification provided for the REST interface. There are many other compatible interfaces that can be used with popular REST testing tools. Some of the most widely used tools that can be used with the REST testing tool are:

Swagger UI

With the Swagger UI, a GUI can be generated from the Swagger specification.

Swagger specifications can be found post VNFM installation at, (https://<VNFM_IP>:8443/docs/vnfm/).

Postman

Another popular tool for creating REST requests is the Postman tool. It is available as a standalone app and as a Chrome browser plugin. You can import a Swagger specification to allow Postman to understand the VNFM REST API in detail, which allows it to assist you while creating requests and interpreting responses.

5.2 DSR VNFM Redundancy

- If the VNFM goes down due to unavoidable circumstances, the data should be restored.
- Persistent Cinder volume is used for this functionality.
- Detach the volume from the existing (not functional) VNFM VM and attach to the new VNFM VM.

5.3 Supported VNF's by the DSR VNFM

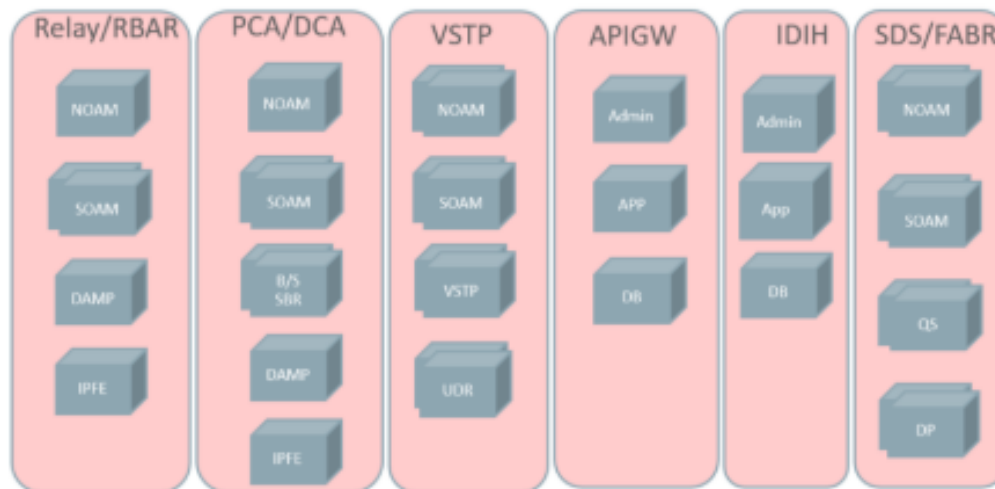


Figure 2. Supported VNFs

Table 4. Supported VNFs and VMs

Supported Dynamic IP VNFs	Supported VNFCs	Supported Dynamic IP VNF	Supported Fixed IP VNF	VNF Dependency
DSR NOAM	NOAM (Active/Standby)	Yes	Yes	
DSR DR NOAM	DR NOAM (Active/Standby)	Yes	Yes	DSR NOAM
DSR Signaling	SOAM (Active/Standby), DA-MP, STP-MP, IPFE, SBR, UDR	Yes	Yes	DSR NOAM
APIGW	DB Server (Active/Standby), Admin Server, Application Server(s)	Yes		
IDIH	APP, MEDIATION, DB Server	Yes	Yes	DSR Signaling
SDS NOAM	NAOM (Active/Standby) and Query Server	Yes	Yes	
SDS DR NOAM	DR NAOM (Active/Standby) and Query Server	Yes	Yes	SDS NOAM
SDS Signaling	SOAM (Active/Standby), DP Server	Yes	Yes	SDS NOAM

6. Upgrading DSR VNFM

The current VNFM stack must be deleted. All the data is stored in the volume that is created during the install procedure. This acts as a persistent storage, so the stack can be safely deleted and the volume is automatically detached from the stack

The user must follow the steps mentioned in the VNFM Installation procedure with the new IMAGE provided. Flavor, Volume need not be created again. The existing volume ID should be given as the volume ID in the `dsrVnfmParams.yaml` file.

Note: VNFM supports both the fixed and dynamic IP support. In order to bring up the new VNFM with the same IP as the existing one, the user can use FIXED IP deployment model.

7. Deploying DSR VNFs

Prerequisites: A virtual infrastructure satisfying the DSR VNFM OpenStack Prerequisites.

7.1 Create a VNF Instance

1. Before a DSR VNF is instantiated, the user must first issue a request to create a VNF instance by using the command **create VNF instance**.
2. Creating a VNF instance informs the VNFM that a user has requested to instantiate a VNF at some point in the future.
3. The VNFM returns a VNF ID that must be saved for future use while performing operations on the same VNF.

Note: Each VNF has its own VNF ID, so if it is required to create a DSR with two signaling VNFs, then issue the request to create a VNF instance three times, once for the network OAM VNF, and once for each signaling VNFs.

For more information about the full list of all inputs and possible outputs of the **create VNF instance** command, see **ETSI NFV-SOL 003**, section 5.4.2.3.1, or the DSR VNFM Swagger specification.

Swagger specifications can be found post VNFM installation at (<https://<VNFM IP>:8443/docs/vnfm/>).

The following image illustrates the VNF instance creation:

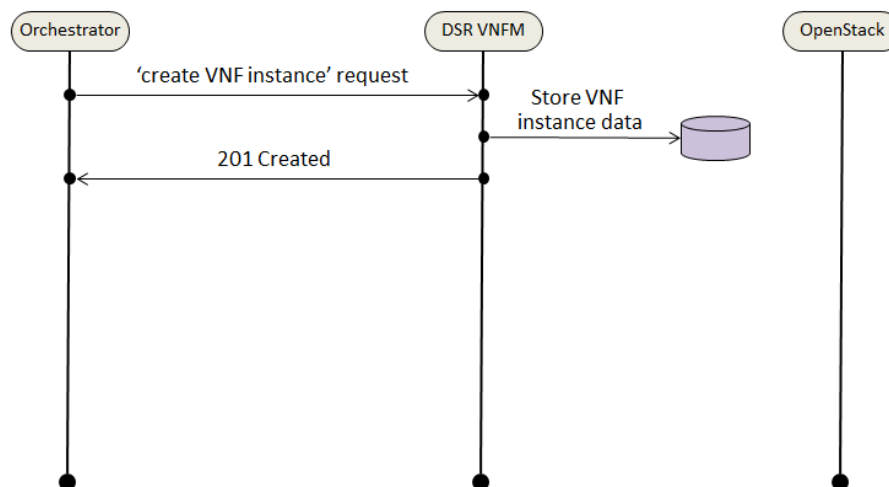


Figure 3. VNF Create Instance Request

Sample Request

Create VNF instance request generated.

Resource URL: https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances

Accept: application/json

Content-Type: application/json

Example for NOAM:

```
{
  "vnfdId": "dsrNetworkOam",
  "vnfInstanceName": "DemoNoam",
  "vnfInstanceDescription": "DemoNoam "
}
```

Example for DR NOAM:

```
{
  "vnfdId": "dsrDrNetworkOam",
  "vnfInstanceName": "DemoDrNoam",
  "vnfInstanceDescription": "DemoDrNoam "
}
```

Example for Signaling:

```
{
  "vnfdId": "dsrSignaling",
  "vnfInstanceName": "DemoSoam",
  "vnfInstanceDescription": "Description"
}
```

Example for APIGW:

```
{
  "vnfdId": "dsrApiGw",
  "vnfInstanceName": "DemoApiGw",
  "vnfInstanceDescription": "Description for APIGW VNF"
}
```

Example for IDIH:

```
{
  "vnfdId": "dsrIdih",
  "vnfInstanceName": "DemoIdih",
  "vnfInstanceDescription": "Description for IDIH VNF"
}
```

Example for SDS NOAM:

```
{
  "vnfdId": "sdsNetworkOam",
  "vnfInstanceName": "DemoSdsNoam",
  "vnfInstanceDescription": "DemoSdsNoam "
}
```


Example for SDS DR NOAM:

```
{
  "vnfdId": "sdsDrNetworkOam",
  "vnfInstanceName": "DemoSdsDrNoam",
  "vnfInstanceDescription": "DemoSdsDrNoam "
}
```

Example for SDS Signaling:

```
{
  "vnfdId": "sdsSignaling",
  "vnfInstanceName": "DemoSdsSoam",
  "vnfInstanceDescription": "DemoSdsSignaling"
}
```

Sample Response

201 Created

Create VNF Instance Response

Content-Type: application/json

Resource URL: https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances

```
{
  "id":"dsrNetworkOam-b44e9a45-b575-4b30-b580-085d8ddd7015",
  "vnfdId":"dsrNetworkOam",
  "instantiationState":"NOT_INSTANTIATED",
  "vnfInstanceName":"DemoNoam",
  "vnfInstanceDescription":"string",
  "vnfProvider":"Oracle",
  "vnfProductName":"DSR",
  "vnfSoftwareVersion":"DSR_8.4.0.0.0_84.8.0",
  "vnfdVersion":"4.0",
  "onboardedVnfPkgInfoId":"N/A",
  "links":{
    "self":{
      "href":"https://<<VNFM HOST
IP>>:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-b44e9a45-b575-4b30-b580-
085d8ddd7015"
    },
    "instantiate":{
```

```

    "href":"https://<<VNFM HOST
IP>>:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-b44e9a45-b575-4b30-b580-
085d8ddd7015/instantiate"
  },
  "scaleToLevel":null,
  "terminate":null
}
}

```

Note: VNFM supports both the secured and the unsecured URL (HTTPS with port 8443 and HTTP with port 8080).

Table 5 describes the parameters used for sending request to VNFM:

Table 5. Parameters and Definitions for VNF Instance

Parameter	Definitions
vnfdId	Identifier of the VNF instance deployment ID to be created
vnfInstanceName	Name of the VNF instance to be created
vnfInstanceDescription	Description of the VNF instance

7.2 Query VNF Instance

The diagram describes a sequence for querying/reading information about a VNF instance.

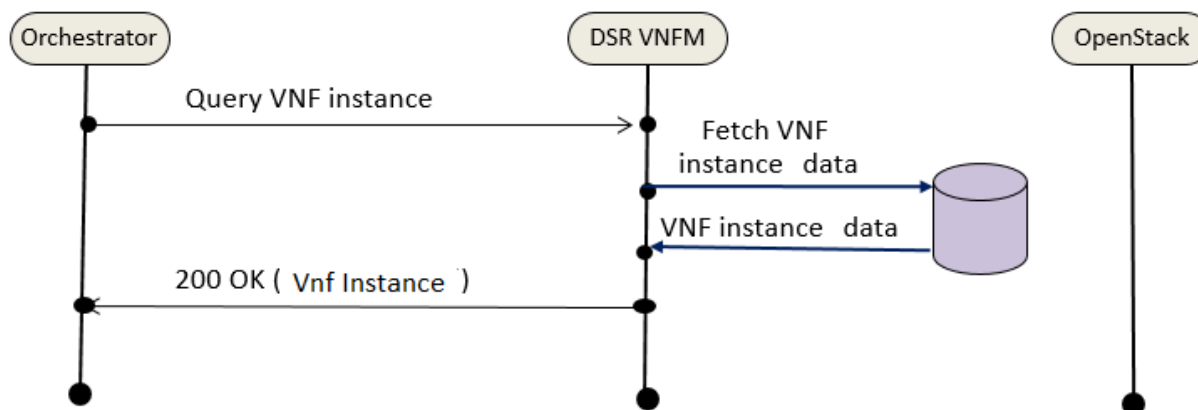


Figure 4. Query VNF Instance

VNF instance query, as illustrated in Figure 4. Query VNF Instance, consists of the following steps:

1. If the NFVO intends to read information about a particular VNF instance, it sends a GET request to the **Individual VNF instance** resource, addressed by the appropriate VNF instance identifier (Vnf Id) in its resource URI.
2. The VNFM returns a **200 OK** response to the NFVO, and includes specific data structure of type **VnfInstance** related to the VNF instance identifier (Vnf Id) in the payload body.
3. If the NFVO intends to query all VNF instances, it sends a GET request to the **VNF instances** resource.

- The VNFM returns a **200 OK** response to the NFVO, and includes zero or more data structures of type **VnfInstance** in the payload body.

7.2.1 Query Individual VNF Instance

Sample Request for Single VNF Instance:

URL: GET: `https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances/<<VNF Instance ID>>`

Sample Response for Single VNF Instances:

URL: GET: `https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances/<<VNF Instance ID>>`

Accept: `application/json`

Content-Type: `application/json`

```
{
  "id": "dsrNetworkOam-793a2420-adab-4347-9667-489ae671b767",
  "vnfdId": "dsrNetworkOam",
  "instantiationState": "NOT_INSTANTIATED",
  "vnfInstanceName": "string",
  "vnfInstanceDescription": "string",
  "vnfProvider": "Oracle",
  "vnfProductName": "DSR",
  "vnfSoftwareVersion": "DSR_8.4.0.0.0_84.8.0",
  "vnfdVersion": "4.0",
  "onboardedVnfPkgInfoId": "N/A",
  "links": {
    "self": {
      "href": "https:// <<VNFM HOST
IP>>:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-793a2420-adab-4347-9667-
489ae671b767"
    },
    "instantiate": {
      "href": "https:// <<VNFM HOST
IP>>:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-793a2420-adab-4347-9667-
489ae671b767/instantiate"
    }
  }
}
```

Response Body for VNF Instances that are Instantiated

```
{
  "id": "dsrNetworkOam-c689e44d-2b93-473f-935a-3bf09957fe9f",
  "vnfdId": "dsrNetworkOam",
```

```

    "instantiationState": "INSTANTIATED",
    "vnfInstanceName": "dsrvnfm",
    "vnfInstanceDescription": "dsrvnfm",
    "vnfProvider": "Oracle",
    "vnfProductName": "DSR",
    "vnfSoftwareVersion": "DSR_8.4.0.0.0_84.8.0",
    "vnfdVersion": "4.0",
    "onboardedVnfPkgInfoId": "N/A",
    "links": {
        "self": {
            "href": "https:// <<VNFM HOST
IP>>:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-c689e44d-2b93-473f-935a-
3bf09957fe9f"
        },
        "instantiate": {
            "href": "https:// <<VNFM HOST
IP>>:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-c689e44d-2b93-473f-935a-
3bf09957fe9f/instantiate"
        },
        "scaleToLevel": {
            "href": "https:// <<VNFM HOST
IP>>:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-c689e44d-2b93-473f-935a-
3bf09957fe9f/scale_to_level"
        },
        "terminate": {
            "href": "https:// <<VNFM HOST
IP>>:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-c689e44d-2b93-473f-935a-
3bf09957fe9f/terminate"
        }
    },
    "instantiatedVnfInfo": {
        "flavourId": "DSR NOAM",
        "vnfState": "STARTED",
        "extCpInfo": {
            "id": null,
            "cpdId": null
        },
        "scaleStatus": [{
            "aspectId": "NOAM",
            "scaleLevel": "2"
        }
    ]
}

```

```
        }],
      },
      "vimConnectionInfo": {
        "id": "vimid",
        "vimType": "OpenStack",
        "interfaceInfo": {
          "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
        },
        "accessInfo": {
          "username": "dsrat.user",
          "password": "xxxxxxx",
          "userDomain": "Default",
          "projectDomain": "default",
          "tenant": "DSRAT_Feature_Test1"
        },
        "extra": {}
      }
    }
  }
```

7.2.2 Query All VNF Instances

Sample Request

URL: GET: https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances

Sample Response

URL: GET: https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances

Accept: application/json

Content-Type: application/json

Response Body for No VNF Instances

```
[ ]
```

Response Body for VNF Instances

```
[
  {
    "id": "dsrNetworkOam-38f694dc-be36-4747-814d-5fccd4fa6163",
    "vnfdId": "dsrNetworkOam",
    "instantiationState": "INSTANTIATED",
    "vnfInstanceName": "string",
    "vnfInstanceDescription": "dsrvnfm",
    "vnfProvider": "Oracle",
    "vnfProductName": "DSR",
    "vnfSoftwareVersion": "DSR_8.4.0.0.0_84.8.0",
    "vnfdVersion": "4.0",
    "onboardedVnfPkgInfoId": "N/A",
    "links": {
      "self": {
        "href": "https:// <<VNFM HOST
IP>>:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-38f694dc-be36-4747-814d-
5fccd4fa6163"
      },
      "instantiate": {
        "href": "https:// <<VNFM HOST
IP>>:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-38f694dc-be36-4747-814d-
5fccd4fa6163/instantiate"
      },
      "scaleToLevel": {
        "href": "https:// <<VNFM HOST IP>>:8443/vnflcm/v1/dsrNetworkOam-
38f694dc-be36-4747-814d-5fccd4fa6163/scale_to_level"
      },
      "terminate": {
```

```

    "href": "https:// <<VNFM HOST IP>>:8443/vnflcm/v1/dsrNetworkOam-
38f694dc-be36-4747-814d-5fccd4fa6163/terminate"
  },
  "instantiatedVnfInfo": {
    "flavourId": "DSR NOAM",
    "vnfState": "STARTED",
    "extCpInfo": {
      "id": null,
      "cpdId": null
    },
    "scaleStatus": [
      {
        "aspectId": "NOAM",
        "scaleLevel": "2"
      }
    ]
  },
  "vimConnectionInfo": {
    "id": "vimid",
    "vimType": "OpenStack",
    "interfaceInfo": {
      "controllerUri": "https://dpc1.us.oracle.com:5000/v3"
    },
    "accessInfo": {
      "username": "dsrvnfm",
      "password": "xxxxxxx",
      "userDomain": "Default",
      "projectDomain": "default",
      "tenant": "dsrvnfm"
    },
    "extra": {}
  },
  {
    "id": "dsrNetworkOam-31fd9dc5-bcce-4dfb-ae21-46f07cd3cba5",
    "vnfdId": "dsrNetworkOam",
    "instantiationState": "NOT_INSTANTIATED",

```

```

"vnfInstanceName": "demo",
"vnfInstanceDescription": "dsrvnfm",
"vnfProvider": "Oracle",
"vnfProductName": "DSR",
"vnfSoftwareVersion": "DSR_8.4.0.0.0_84.8.0",
"vnfdVersion": "4.0",
"onboardedVnfPkgInfoId": "N/A",
"links": {
  "self": {
    "href": "https:// <<VNFM HOST
IP>>:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-31fd9dc5-bcce-4dfb-ae21-
46f07cd3cba5"
  },
  "instantiate": {
    "href": "https:// <<VNFM HOST
IP>>:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-31fd9dc5-bcce-4dfb-ae21-
46f07cd3cba5/instantiate"
  },
  "scaleToLevel": null,
  "terminate": null
}
}
}

```

7.3 Instantiating the Network OAM VNF

Network OAM VNF supports both dynamic and fixed IP deployment.

To start a DSR deployment, it is required to instantiate a DSR network OAM VNF. Before deploying the VNF, make sure the following information is available:

The **VNF ID** for a previously created DSR Network OAM VNF instance

Information about the OpenStack instance on which the VNF must be deployed:

- OpenStack Controller URI
- User Domain Name
- Project Domain Id
- Username
- Password
- Tenant name

The name of a Public Network in your chosen OpenStack instance that will carry OAM traffic.

The IP of an NTP server accessible by VMs within the selected OpenStack instance. The OpenStack controller that controls the selected OpenStack instance normally hosts an NTP server, and is often a good choice.

For more information about the full list of all inputs and possible outputs of the **instantiate VNF** command, see **ETSI NFV-SOL 003**, section 5.4.4.3.1, or the **DSR VNFM Swagger specification**. Swagger specifications can be found post VNFM installation at (<https://<VNFM IP>:8443/docs/vnfm/>).

Sample Request

Instantiating NOAM Request for dynamic IP deployment.

Resource URL: https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances/<VNF ID received from create request>/instantiate

Accept: application/json

Content-Type: application/json

```
{
  "flavourId": "DSR NOAM",
  "instantiationLevelId": "HA",
  "extVirtualLinks": "extVirtualLinks",
  "extManagedVirtualLinks": [],
  "vimConnectionInfo": [ {
    "id": "vimid",
    "vimType": "OpenStack",
    "interfaceInfo": {
      "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
    },
    "accessInfo": {
      "username": "dsrsci.user",
      "password": "xxxxxx",
      "userDomain": "Default",
      "projectDomain": "default",
      "tenant": "DSR CI"
    }
  } ],
  "localizationLanguage": "localizationLanguage",
  "additionalParams": {
    "xmiNetwork": {
      "name": "ext-net3",
      "ipVersion": "IPv4"
    },
    "ntpServerIp": "10.250.32.10"
  }
}
```

```
}
}
```

Instantiating NOAM Request for fixed IP deployment.

```
{
  "flavourId": "DSR NOAM",
  "instantiationLevelId": "HA",
  "extVirtualLinks": "extVirtualLinks",
    "extManagedVirtualLinks": [],

  "vimConnectionInfo": [ {
    "id": "vimid",
    "vimType": "OpenStack",
    "interfaceInfo": {
      "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
    },
    "accessInfo": {
      "username": "dsrcli.user",
      "password": "xxxxxx",
      "userDomain": "Default",
      "projectDomain": "default",
      "tenant": "DSR CI"
    }
  }

  ],
  "localizationLanguage": "localizationLanguage",
  "additionalParams": {
    "xmiNetwork": {
      "name": "ext-net3",
      "ipVersion": "IPv4",
      "fixedIps":
      {
        "primaryNoamIp": "10.75.218.50",
        "secondaryNoamIp": "10.75.218.49",
        "noamVip": "10.75.218.134"
      }
    }
  }
}
```

```

    },
    "ntpServerIp": "10.250.32.10"
  }
}

```

Note: User must identify available IP addresses to be used in the network. If the user provides an IP address which does not exist in the subnet, the stack creation fails.

Sample Response

Instantiating NOAM Request.

```

202 Accepted
Headers:
{
  location: https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
fb21f9d3-43ad-46cd-a03f-7220bb36a5c6
  date: Tue, 29 Jan 2019 10:39:24 GMT
  content-length: 0  content-type:
  application/xml
}

```

Notes:

- The 202 response means that the request was accepted for processing. The VNF might take up to 15 minutes to become fully operational. Use the DSR GUI to determine when the VNF is operational.
- If the VNFM creates a VNF that is operational, but has no Signaling VNFs, then it is required to deploy one or more Signaling VNF, and create the DIAMETER configuration data (peers, connections, etc.) for those VNFs, to perform DIAMETER routing.
- After NOAM VNF deployment, the standby NOAM is automatically changed to **Force StandBy**, purposely to avoid any switchover, while DSR Signaling VNF is deployed. Once DSR Signaling Site is deployed and no more Life Cycle Management operations are planned, change **Force Standby** NOAM to Active by changing the **Max Allowed HA Role** to **Active** on the **Status & Manage -> HA** options in the Active NOAM GUI.
- The supported NOAM Flavor is **DSR NOAM**.
- The supported NOAM instantiation level id is **HA**, that creates two NOAMs.

Table 6 describes the parameters used for sending request to VNFM.

Table 6. Parameters and Definitions for Network OAM VNF

Parameter	Definitions
flavourId	Identifier of the VNF deployment flavor to be instantiated
xmiNetwork	Network that is used to provide access to the DSR entities (GUI, ssh), and for inter-site communication

ntpServerIp	IP of the NTP server
fixedIps	Json object in network to provide IP address
primaryNoamIp	IP address for primary NOAM IP
secondaryNoamIp	IP address for secondary NOAM IP
noamVip	IP address for NOAM VIP
imiNetwork	Network used for internal communication of DSR entities
ipVersion	IP version of the network - "IPv4"/"IPv6"
primaryNoamImiIp	IP address for primary NOAM IP of IMI
secondaryNoamImiIp	IP address for secondary NOAM IP of IMI

7.4 Instantiating the DR Network OAM VNF

DRNOAM is the Disaster recovery NOAM site. The operator can make DRNOAM as the Primary Site, in case both the Active and StandBy NOAM of Primary site fails, and can continue the operations without any disturbance.

DRNOAM supports both dynamic and fixed deployment model.

When a setup is configured with a DR NOAM then first NOAM SG is treated as Primary NOAM Site and second NOAM SG is treated as Secondary NOAM site.

In order to instantiate a DSR DR Network OAM VNF the following information must be available:

The **VNF ID** for a previously created DSR DR Network OAM VNF instance.

Information about the OpenStack instance on which the VNF must be deployed:

- OpenStack Controller URI
- User Domain Name
- Project Domain Id
- Username
- Password
- Tenant name

The name of a Public Network in your chosen OpenStack instance that will carry OAM traffic.

OpenStack resource IDs for the XMI IPs from both DSR NOAM VMs.

Note: The resource IDs can be obtained by examining the DSR Network OAM stack to which the identified DR NOAM VNF would be attached.

Name of Active Primary DSR NOAM VM.

The IP of an NTP server accessible by VMs within the selected OpenStack instance. The OpenStack controller that controls the selected OpenStack instance normally hosts an NTP server, and is often a good choice.

7.4.1 Determining the DR NOAM XMI resource IDs

The following facts must be considered before proceeding with DR NOAM site creation:

- DRNOAM site must be created on separate tenant.

- DRNOAM site is referred as Secondary NOAM. Therefore, we have two sites, Primary and Secondary.
- Secondary Site configuration is done on Primary Active NOAM.
- In the Primary Active NOAM, when second NOAM Server Group gets created, it automatically becomes Secondary.
- Primary Active NOAM communicates to Secondary Active NOAM through the existing comcol replication and merging mechanism.
- Secondary NOAM Site is optional and it does not need to be deployed at the same time as of Primary NOAM.

From the OpenStack GUI:

1. Change your view to the tenant on which the DSR Network OAM VNF was deployed.
2. Go to **Project->Network->Network Topology**. A diagram of all VMs in the tenant is displayed.
Note: The diagram may take a few minutes to display.
3. Click on one of the NOAM VMs.
4. A pop-up appears having information about the specific NOAM VM.
5. Save the resource ID for the XMI port provided in the IP Addresses section of the pop-up.
Note: The IP Addresses section of the popup contains information about the network ports and resource IDs, assigned to the VM.
6. Repeat the previous step for the other NOAM VM.

For more information about the full list of all inputs and possible outputs of the **instantiate VNF** command, see **ETSI NFV-SOL 003**, section **5.4.4.3.1**, or the **DSR VNFM Swagger specification**. Swagger specifications can be found post VNFM installation at (<https://<VNFM IP>:8443/docs/vnfm/>).

Sample Request

Instantiating DR NOAM Request for Dynamic IP deployment.

Resource URL: https://<myhost-IP>:8443/vnfm/v1/vnf_instances/<VNF ID received from create request>/instantiate

Accept: application/json

Content-Type: application/json

```
{
  "flavourId": "DSR DR NOAM",
  "instantiationLevelId": "HA",
  "extVirtualLinks": "extVirtualLinks",
  "extManagedVirtualLinks": [{
    "id": "id1",
    "virtualLinkDescId": "Active NOAM",
    "resourceId": "156d73cf-6e44-456b-a661-14bd0cc2b43c"
  },
  {
    "id": "id2",
    "virtualLinkDescId": "StandBy NOAM",
    "resourceId": "5c638770-5585-44c7-97c7-b4a52a26e5ec"
  }
  ]
}
```

```

    }
  ],
  "vimConnectionInfo": [ {
    "id": "vimid",
    "vimType": "OpenStack",
    "interfaceInfo": {
      "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
    },
    "accessInfo": {
      "username": "dsrsci.user",
      "password": "xxxxxx",
      "userDomain": "Default",
      "projectDomain": "default",
      "tenant": "DSR CI"
    }
  }
],
"localizationLanguage": "localizationLanguage",
"additionalParams": {
  "xmiNetwork": {
    "name": "ext-net3",
    "ipVersion": "IPv4"
  },
  "ntpServerIp": "10.250.32.10",
  "primaryNoamVmName": "NOAM00-ea47f4b1"
}
}

```

Instantiating DR NOAM Request for Fixed IP deployment.

```

{
  "flavourId": "DSR DR NOAM",
  "instantiationLevelId": "HA",
  "extVirtualLinks": "extVirtualLinks",
  "extManagedVirtualLinks": [
    {
      "id": "id1",
      "virtualLinkDescId": "Active NOAM IP's",
    }
  ]
}

```

```

    "resourceId":"38121fc6-310c-4012-9787-b5289dd620b9"
  },
  {
    "id":"id2",
    "virtualLinkDescId":"Secondary NOAM IP's",
    "resourceId":"baa54c8d-1a7a-4b15-8d64-8fe9af50b000"
  }
],
"vimConnectionInfo":[
  {
    "id":"vimid",
    "vimType":"OpenStack",
    "interfaceInfo":{
      "controllerUri":"https://dpc1.us.oracle.com:5000/v3"
    },
    "accessInfo":{
      "username":"dsrvnfm",
      "password":"xxxx",
      "userDomain":"Default",
      "projectDomain": "default",
      "tenant":"dsrvnfm"
    }
  }
],
"localizationLanguage":"localizationLanguage",
"additionalParams":{
  "ntpServerIp":"10.250.32.10",
  "xmiNetwork":{
    "name":"ext-net4",
    "ipVersion":"IPv4",
    "fixedIps":{
      "drPrimaryNoamIp":"10.75.218.167",
      "drSecondaryNoamIp":"10.75.218.174",
      "drNoamVip":"10.75.218.165"
    }
  }
},
"primaryNoamVmName":"NOAM00-9ca5c163"

```

```
}
}
```

Sample Response

Instantiating DR NOAM Response.

```
202 Accepted
Headers:
{
  location: https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
fb21f9d3-43ad-46cd-a03f-7220bb36a5c6
  date: Tue, 21 Feb 2019 10:39:24 GMT
  content-length: 0  content-type:
  application/xml
}
```

Note:

- The 202 response means that the request was accepted for processing. The VNF might take up to 15 minutes to become fully operational. Use the DSR GUI to determine when the VNF is operational.
- The supported NOAM Flavor is **DSR NOAM**.
- The supported NOAM instantiation level id is **HA**.
- Support for IPv6 networks - ipVersion should be "IPv6" in the request Body.

Table below describes the parameters used for sending request to VNFM.

Table 7. Parameters and Definitions for DR Network OAM VNF

Parameter	Definitions
flavourId	Identifier of the VNF deployment flavor to be instantiated
instantiationLevelId	Identifier of the instantiation level of the deployment flavor to be instantiated. If not present, the default instantiation level is HA.
resourceId	The identifier of the resource in the scope of the VIM or the resource provider
xmiNetwork	Network that is used to provide access to the DSR entities (GUI, ssh), and for inter-site communication
imiNetwork	Network used for internal communication of DSR entities
name	Network name, for example; ext-net
ipVersion	IP version IPv4 or IPv6
ntpServerIp	IP of the NTP server
primaryNoamVmName	Primary Active DSR NOAM VM name

drPrimaryNoamIp	IP address of primary DR Noam
drSecondaryNoamIp	IP address of secondary DR Noam
drPrimaryNoamIp	IP address of primary DR Noam
dsPrimaryNoamImiIp	IP address for primary DR NOAM IP of IMI
drSecondaryNoamImiIp	IP address for secondary DR NOAM IP of IMI

7.5 Instantiating the Signaling VNF with Multiple XSI (1, 2 & 4 XSI Interface)

Signaling VNF supports both dynamic and fixed IP deployment.

To deploy the first signaling VNF, the following must be available:

A previously instantiated DSR Network OAM VNF.

The VNF ID for a previously created DSR Signaling VNF instance.

Information about the OpenStack instance on which you want to deploy the VNF:

- OpenStack Controller URI
- User Domain Name
- Project Domain Id
- Username
- Password
- Tenant name

The name of a Public Network in your chosen OpenStack instance that will carry OAM traffic.

The name of a Public Network in your chosen OpenStack instance that will carry Signaling traffic.

Note: This should be a different network than the one that carries OAM traffic.

The IP address of the NTP server accessible by VMs within the selected OpenStack instance. The OpenStack controller that controls your chosen OpenStack instance normally hosts an NTP server, and is often a good choice.

OpenStack resource IDs for the XMI IPs from both NOAM VMs.

Note: The resource IDs can be obtained by examining the network OAM stack to which the identified signaling VNF would be attached.

Name of the active NOAM VM.

Note: To avoid switchover of Active NOAM, make the StandBy NOAM as **Forced Standby** by changing the **Max Allowed HA Role** to **Standby** on **Status & Manage** -> **HA** from Active NOAM GUI.

Name of the NOAM SG.

Figure 1 illustrates the VNF instantiation:

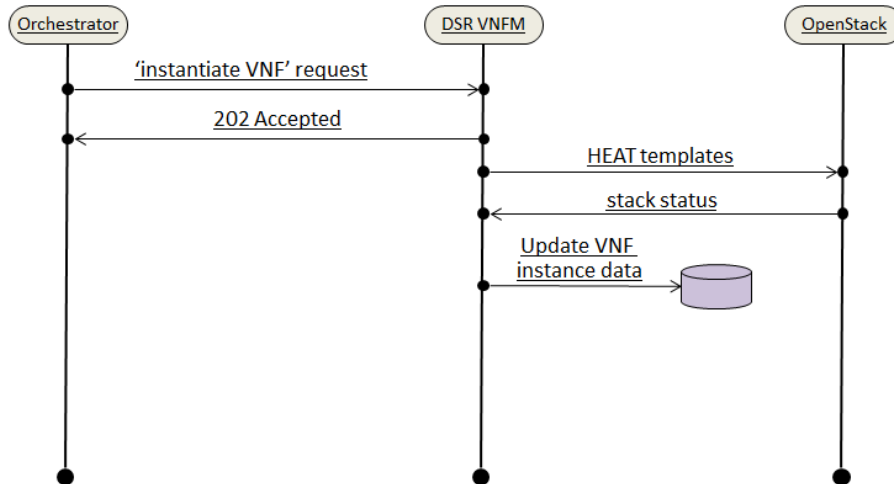


Figure 5. VNF Instantiate Request

Table 8 contains the supported Instantiation levels to instantiate a VNF resource for the DSR Signaling VNF.

Table 8. Supported Instantiation Levels for DSR Signaling VNF

Signaling Flavors supported by VNFM	Small					Medium					Large				
	DAMP	IPFE	STP	SBR	UDR	DAMP	IPFE	STP	SBR	UDR	DAMP	IPFE	STP	SBR	UDR
DIAMETER	2	2	0	0	0	4	2	0	0	0	8	2	0	0	0
SS7	0	0	2	0	0	0	0	4	0	0	0	0	8	0	0
DIAMETER + SS7	2	2	2	0	0	4	2	4	0	0	8	2	8	0	0
DIAMETER + SBR	2	2	0	3	0	4	2	0	6	0	8	2	0	9	0
DIAMETER + SS7 + SBR	2	2	2	3	0	4	2	4	6	0	8	2	8	9	0
DIAMETER + UDR	2	2	0	0	2	4	2	0	0	2	8	2	0	0	2
SS7 + UDR	0	0	2	0	2	0	0	4	0	2	0	0	8	0	2
DIAMETER + SS7 + SBR + UDR	2	2	2	3	2	4	2	4	6	2	8	2	8	9	2

Note:

- In case of SBR flavors, it is mandatory to pass the `sbrNetwork` parameter for instantiation of signaling stack. VNFM always creates Replication port for SBRs.
- In case of UDR flavors, VNFM supports one and two xsi interface.
- Total number of servers allowed per signaling VNF is 48.
- Total number of IPFE servers allowed per signaling VNF is 4.
- Total number of SOAMs for any of the above servers is 2.

For Example: Total number of servers per signaling VNF = No. of SOAM's + No. of DAMP's + No. of IPFE's + No. of STP's + No. of SBR's+ No. of UDR's.

7.5.1 Determine the NOAM XMI Resource IDs

From the OpenStack GUI:

1. Change your view to the tenant on which the DSR Network OAM VNF was deployed.
2. Navigate to **Orchestration->Network->Network Topology**.

A diagram of all VMs in the tenant displays.

Note: The diagram may take a few minutes to display.

3. Click on one of the NOAM VMs.

A screen displays with information about the specific NOAM VM.

4. Save the resource ID for the XMI port provided in the IP addresses section of the screen.

Note: The IP Addresses section of the popup screen contains information about the network ports and resource IDs assigned to the VM.

5. Repeat the previous step for the other NOAM VM.

For more information about the full list of all inputs and possible outputs of the **instantiate VNF** command, see **ETSI NFV-SOL 003**, section **5.4.4.3.1**, or the DSR VNFM Swagger specification. Swagger specifications can be found post VNFM installation at (https://<VNFM_IP>:8443/docs/vnfm/).

7.5.2 DSR Signaling VNF with Multiple XSI support (1, 2 & 4 xsi interface only)

- Multiple XSI supports only DSR Signaling node.
- DAMP vnf supports 1, 2 & 4 xsi interface.
- STPMP vnf supports 1, 2, & 4 xsi interface.
- IPFE vnf supports 1, 2, & 4 xsi interface.
- UDR vnf supports only 1 & 2 xsi interface.
- While passing the `xsiNetwork` through request body. Add list of network in the `xsiNetwork`.

For example:

1 xsiNetwork	2 xsiNetwork	3 xsiNetwork
<pre>"xsiNetwork": [{ "name": "provider- vlan500", "ipVersion": "IPv4" }]</pre>	<pre>"xsiNetwork": [{ "name": "provider- vlan500", "ipVersion": "IPv4" }, { "name": "provider- vlan610", "ipVersion": "IPv4" }]</pre>	<pre>"xsiNetwork": [{ "name": "provider-vlan500", "ipVersion": "IPv4" }, { "name": "provider-vlan610", "ipVersion": "IPv4" }, { "name": "provider-vlan500", "ipVersion": "IPv4" }, { "name": "provider-vlan610", "ipVersion": "IPv4" }]</pre>

The sample request and response provided below represents signaling flavors without SBR such as, DIAMETER, SS7 & DIAMETER+SS7, DIAMETER+UDR, and SS7+UDR, with multiple xsi (1, 2, 4 xsi interface) for Dynamic IP and Fixed IP deployment model.

Sample Request

Instantiating the first signaling VNF request for Dynamic IP deployment model.

Resource URL: `https://<myhost-IP>:8443/vnfm/v1/vnf_instances/<VNF ID received from create request>/instantiate`

Accept: `application/json`

Content-Type: `application/json`

```
{
  "flavourId": "DIAMETER+SS7",
  "instantiationLevelId": "small",
  "extVirtualLinks": "extVirtualLinks",
  "extManagedVirtualLinks": [{
    "id": "",
    "virtualLinkDescId": "active
NOAM",
    "resourceId": "8a4d1ec6-367a-
4b1a-978d-2c4eae3daec3"
  },
  {
    "id": "",
    "virtualLinkDescId": "standby
NOAM",
    "resourceId": "2bed5886-8c97-
4623-8da3-9c500cce71e3"
  }
],
  "vimConnectionInfo": [ {
    "id": "vimid",
    "vimType": "OpenStack",
    "interfaceInfo": {
      "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
    },
    "accessInfo": {
      "username": "dsrcli.user",
      "password": "xxxx",
      "userDomain": "Default",
```

```

        "projectDomain": "default",
        "tenant": "DSR CI"
    },

    },

    "localizationLanguage": "localizationLanguage",
    "additionalParams": {
        "xmiNetwork": {
            "name": "ext-net3",
            "ipVersion": "IPv4"
        },
        "imiNetwork": {
            "name": "imi-private",
            "ipVersion": "IPv4"
        },
        "xsiNetwork": [{
            "name": "ext-net2",
            "ipVersion": "IPv4"
        },
        {
            "name": "ext-net5",
            "ipVersion": "IPv4"
        }
    ],
        "ntpServerIp": "10.250.32.10",
        "primaryNoamVmName": "NOAM00-32cd6138",
        "noamSgName":
"dsrNetworkOam_NOAM_32cd6138_SG"
    }
}

```

Instantiating the first signaling VNF request for Fixed IP deployment.

```

{
    "flavourId": "DIAMETER+SS7",
    "instantiationLevelId": "small",
    "extVirtualLinks": "extVirtualLinks",
    "extManagedVirtualLinks": [

```

```

    {
      "id": "id1",
      "virtualLinkDescId": "",
      "resourceId": "d6be6053-78a9-437a-a139-4dc11792598a"
    },
    {
      "id": "id2",
      "virtualLinkDescId": "",
      "resourceId": "d6be6053-78a9-437a-a139-4dc11792598a"
    }
  ],
  "vimConnectionInfo": [
    {
      "id": "vimid",
      "vimType": "OpenStack",
      "interfaceInfo": {
        "controllerUri": "https://dpc1.us.oracle.com:5000/v3"
      },
      "accessInfo": {
        "username": "dsrvnfm",
        "password": "xxxx",
        "userDomain": "Default",

        "projectDomain": "default",
        "tenant": "dsrvnfm"
      }
    }
  ],
  "localizationLanguage": "localizationLanguage",
  "additionalParams": {
    "xmiNetwork": {
      "name": "ext-net4",
      "ipVersion": "IPv4",
      "fixedIps": {
        "primarySoamXmiIp": "10.75.218.141",
        "secondarySoamXmiIp": "10.75.218.163",
        "soamVip": "10.75.218.97",

```

```

        "dampXmiIps": [
            "10.75.218.38",
            "10.75.218.137"
        ],
        "ipfeXmiIps": [
            "10.75.218.153",
            "10.75.218.126"
        ],
        "stpXmiIps": [
            "10.75.218.67",
            "10.75.218.84"
        ]
    }
},

"imiNetwork": {
    "name": "imi-private",
    "ipVersion": "IPv4",
    "fixedIps": {
        "primarySoamImiIp": "192.167.2.9",
        "secondarySoamImiIp": "192.167.2.10",
        "dampImiIps": [
            "192.167.2.11",
            "192.167.2.12"
        ],
        "ipfeImiIps": [
            "192.167.2.13",
            "192.167.2.14"
        ],
        "stpImiIps": [
            "192.167.2.15",
            "192.167.2.16"
        ]
    }
},

"xsiNetwork": [

```

```

    {
      "name": "ext-net4",
      "ipVersion": "IPv4",
      "fixedIps": {
        "dampXsiIps": [
          "10.75.218.140",
          "10.75.218.155"
        ],
        "ipfeXsiIps": [
          "10.75.218.101",
          "10.75.218.22"
        ],
        "stpXsiIps": [
          "10.75.218.95",
          "10.75.218.108"
        ]
      }
    },
    {
      "name": "ext-net4",
      "ipVersion": "IPv4",
      "fixedIps": {
        "dampXsiIps": [
          "10.75.218.42",
          "10.75.218.122"
        ],
        "ipfeXsiIps": [
          "10.75.218.91",
          "10.75.218.131"
        ],
        "stpXsiIps": [
          "10.75.218.121",
          "10.75.218.83"
        ]
      }
    }
  ],

```



```

    "ntpServerIp": "10.250.32.10",
    "primaryNoamVmName": "NOAM00-",
    "noamSgName": "dsrNetworkOam_NOAM__SG"
  }
}

```

Sample Response

```

202 Accepted
Headers:
{
  location: https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
fb21f9d3-43ad-46cd-a03f-7220bb36a5c6
  date: Tue, 29 Jan 2019 10:39:24 GMT
  content-length: 0  content-type:
  application/xml
}

```

Sample Request

Instantiating the signaling VNF request with SBR (DIAMETER+SBR, DIAMETER+SS7+SBR, DIAMETER+SS7+SBR+UDR) with multiple xsi (1, 2, 4 xsi interface) generated for Dynamic IP deployment model.

Resource URL: https://<<myhost-IP>>:8443/vnfm/v1/vnf_instances/<VNF ID received from create request>/instantiate

Accept: application/json

Content-Type: application/json

```

{
  "flavourId": "DIAMETER+SBR",
  "instantiationLevelId": "small",
  "extVirtualLinks": "extVirtualLinks",
  "extManagedVirtualLinks": [{
    "id": "",
    "virtualLinkDescId": "active
NOAM",
    "resourceId": "8a4d1ec6-367a-
4b1a-978d-2c4eae3daec3"
  },
  {
    "id": "",

```

```

NOAM",
4623-8da3-9c500cce71e3"
    },
    ],
    "vimConnectionInfo": [ {
      "id": "vimid",
      "vimType": "OpenStack",
      "interfaceInfo": {
        "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
      },
      "accessInfo": {
        "username": "dsrsci.user",
        "password": "xxxx",
        "userDomain": "Default",
        "projectDomain": "default",
        "tenant": "DSR CI"
      }
    }
  ]],
  "localizationLanguage": "localizationLanguage",
  "additionalParams": {
    "xmiNetwork": {
      "name": "ext-net3",
      "ipVersion": "IPv4"
    },
    "imiNetwork": {
      "name": "imi-private",
      "ipVersion": "IPv4"
    },
    "xsiNetwork": [{
      "name": "ext-net2",
      "ipVersion": "IPv4"
    },
    {
      "name": "ext-net5",
      "ipVersion": "IPv4"
    }
  ]
}

```

```

    },
    ],
    "sbrNetwork": {
        "name": "ext-net3",
        "ipVersion": "IPv4"
    },
    "ntpServerIp": "10.250.32.10",
    "primaryNoamVmName": "NOAM00-32cd6138",
    "noamSgName":
    "dsrNetworkOam_NOAM_32cd6138_SG"
    }
}

```

Instantiating the signaling VNF request with SBR (DIAMETER+SBR, DIAMETER+SS7+SBR) with multiple xsi (1,2,4 xsi interface) generated for Fixed IP deployment model.

```

{
  "flavourId": "DIAMETER+SBR",
  "instantiationLevelId": "small",
  "extVirtualLinks": "extVirtualLinks",
  "extManagedVirtualLinks": [
    {
      "id": "id1",
      "virtualLinkDescId": "active NOAM",
      "resourceId": "d6be6053-78a9-437a-a139-4dc11792598a"
    },
    {
      "id": "id2",
      "virtualLinkDescId": "standby NOAM",
      "resourceId": "d6be6053-78a9-437a-a139-4dc11792598a"
    }
  ],
  "vimConnectionInfo": [
    {
      "id": "vimid",
      "vimType": "OpenStack",
      "interfaceInfo": {
        "controllerUri": "https://dpc1.us.oracle.com:5000/v3"
      }
    }
  ]
}

```

```

},
"accessInfo":{
  "username":"dsrvnfm",
  "password":"xxxx",
  "userDomain": "Default",
  "projectDomain": "default",
  "tenant":"dsrvnfm"
}
},
"localizationLanguage":"localizationLanguage",
"additionalParams":{
  "xmiNetwork":{
    "name":"ext-net4",
    "ipVersion":"IPv4",
    "fixedIps":{
      "primarySoamXmiIp":"10.75.218.141",
      "secondarySoamXmiIp":"10.75.218.163",
      "soamVip":"10.75.218.97",
      "dampXmiIps":[
        "10.75.218.38",
        "10.75.218.137"
      ],
      "ipfeXmiIps":[
        "10.75.218.153",
        "10.75.218.126"
      ],
      "sbrXmiIps":[
        "10.75.218.67",
        "10.75.218.84",
        "10.75.218.184"
      ]
    }
  },
  "imiNetwork":{
    "name":"imi-private",
    "ipVersion":"IPv4",

```

```

"fixedIps":{
  "primarySoamImiIp":"192.167.2.1",
  "secondarySoamImiIp":"192.167.2.2",
  "dampImiIps":[
    "192.167.2.3",
    "192.167.2.4"
  ],
  "ipfeImiIps":[
    "192.167.2.5",
    "192.167.2.6"
  ],
  "sbrImiIps":[
    "192.167.2.7",
    "192.167.2.8",
    "192.167.2.9"
  ]
},
"sbrNetwork":{
  "name":"ext-net7",
  "ipVersion":"IPv4",
  "fixedIps":{
    "sbrNetworkIps":[
      "10.196.218.95",
      "10.196.218.108",
      "10.196.218.18"
    ]
  }
},
"xsiNetwork":[
  {
    "name":"ext-net4",
    "ipVersion":"IPv4",
    "fixedIps":{
      "dampXsiIps":[
        "10.75.218.140",
        "10.75.218.155"
      ]
    }
  }
]

```

```

],
"ipfeXsiIps":[
"10.75.218.101",
"10.75.218.22"
]
}
},
{
"name":"ext-net4",
"ipVersion":"IPv4",
"fixedIps":{
"dampXsiIps":[
"10.75.218.42",
"10.75.218.122"
],
"ipfeXsiIps":[
"10.75.218.91",
"10.75.218.131"
]
}
}
],
"ntpServerIp":"10.250.32.10",
"primaryNoamVmName":"NOAM00-f1888e6d",
"noamSgName":"dsrNetworkOam_NOAM_f1888e6d_SG"
}
}

```

For signaling flavors with UDR with multiple xsi (1 and 2 XSI interface) for Fixed IP deployment model

```

{
  "flavourId":"DIAMETER+UDR",
  "instantiationLevelId":"small",
  "extVirtualLinks":"extVirtualLinks",
  "extManagedVirtualLinks":[
    {
      "id":"id1",

```

```

        "virtualLinkDescId": "active NOAM",
        "resourceId": "6ba09324-0568-4489-bdb6-bcc9bb6218a3"
    },
    {
        "id": "id2",
        "virtualLinkDescId": "standby NOAM",
        "resourceId": "379e4fce-61a7-4323-8ee3-d548e819042f"
    }
],
"vimConnectionInfo": [
    {
        "id": "vimid",
        "vimType": "OpenStack",
        "interfaceInfo": {
            "controllerUri": "https://dpc1.us.oracle.com:5000/v3"
        },
        "accessInfo": {
            "username": "dsrvnfm",
            "password": "xxxx",
            "userDomain": "Default",

            "projectDomain": "default",
            "tenant": "dsrvnfm"
        }
    }
],
"localizationLanguage": "localizationLanguage",
"additionalParams": {
    "xmiNetwork": {
        "name": "ext-net4",
        "ipVersion": "IPv4",
        "fixedIps": {
            "primarySoamXmiIp": "10.75.218.207",
            "secondarySoamXmiIp": "10.75.218.218",

```

```

        "soamVip":"10.75.218.204",
        "primaryUdrXmiIp":"10.75.218.243",
        "secondaryUdrXmiIp":"10.75.218.223",
        "udrVip":"10.75.218.191",
        "dampXmiIps":[
            "10.75.218.196",
            "10.75.218.213"
        ],
        "ipfeXmiIps":[
            "10.75.218.226",
            "10.75.218.216"
        ]
    }
},

"imiNetwork":{
    "name":"imi-private",
    "ipVersion":"IPv4",
    "fixedIps":{
        "primarySoamImiIp":"192.167.2.1",
        "secondarySoamImiIp":"192.167.2.2",
        "primaryUdrImiIp":"192.167.2.3",
        "secondaryUdrImiIp":"192.167.2.4",
        "dampImiIps":[
            "192.167.2.5",
            "192.167.2.6"
        ],
        "ipfeImiIps":[
            "192.167.2.7",
            "192.167.2.8"
        ]
    }
},

"xsiNetwork":[

```



```

{
  "name": "ext-net4",
  "ipVersion": "IPv4",
  "fixedIps": {
    "dampXsiIps": [
      "10.75.218.214",
      "10.75.218.217"
    ],
    "ipfeXsiIps": [
      "10.75.218.149",
      "10.75.218.238"
    ],
    "primaryUdrXsiIps": [
      "10.75.218.201"
    ],
    "secondaryUdrXsiIps": [
      "10.75.218.215"
    ]
  }
},
{
  "name": "ext-net4",
  "ipVersion": "IPv4",
  "fixedIps": {
    "dampXsiIps": [
      "10.75.218.235",
      "10.75.218.178"
    ],
    "ipfeXsiIps": [
      "10.75.218.225",
      "10.75.218.219"
    ],
    "primaryUdrXsiIps": [
      "10.75.218.175"
    ]
  }
}

```

```

        ],
        "secondaryUdrXsiIps": [
            "10.75.218.230"
        ]
    }
}

],
"ntpServerIp": "10.250.32.10",
"primaryNoamVmName": "NOAM00-a2eaba59",
"noamSgName": "dsrNetworkOam_NOAM_a2eaba59_SG"
}
}

```

Sample Response

Instantiating the signaling VNF with SBR response

```

202 Accepted
Headers:
location: https:// <<VNFM HOST IP>>:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
f00678f4-ea8e-417f-9c5a-e126926db402
date: Wed, 13 Feb 2019 09:55:01 GMT
content-length: 0
content-type: application/xml

```

Sample Request

For signaling flavors with DIAMETER+SS7+SBR+UDR with multiple xsi (2 XSI interface) for Fixed IP deployment model

```

{"flavourId": "DIAMETER+SS7+SBR+UDR",
 "instantiationLevelId": "small",
 "extVirtualLinks": "extVirtualLinks",
 "extManagedVirtualLinks": [
     {
         "id": "id1",
         "virtualLinkDescId": "active NOAM",
         "resourceId": "790bf9f7-8834-4c3a-bd17-5544ef5e6848"
     },
     {

```

```

        "id": "id2",
        "virtualLinkDescId": "standby NOAM",
        "resourceId": "1776d877-f643-45d6-b6da-bf1a540a01d1"
    }
],
"vimConnectionInfo": [
    {
        "id": "vimid",
        "vimType": "OpenStack",
        "interfaceInfo": {
            "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
        },
        "accessInfo": {
            "username": "dsrvnfm",
            "password": "xxxxxx",
            "userDomain": "Default",
            "projectDomain": "default",
            "tenant": "dsrvnfm"
        }
    }
],
"localizationLanguage": "localizationLanguage",
"additionalParams": {
    "xmiNetwork": {
        "name": "ext-net4",
        "ipVersion": "IPv4",
        "fixedIps": {
            "primarySoamXmiIp": "10.75.218.91",
            "secondarySoamXmiIp": "10.75.218.223",
            "soamVip": "10.75.218.36",
            "primaryUdrXmiIp": "10.75.218.180",
            "secondaryUdrXmiIp": "10.75.218.205",
            "udrVip": "10.75.218.121",
            "dampXmiIps": ["10.75.218.242", "10.75.218.194"],
            "ipfeXmiIps": ["10.75.218.159", "10.75.218.198"],
            "stpXmiIps": ["10.75.218.241", "10.75.218.128"],
            "sbrXmiIps": ["10.75.218.147", "10.75.218.209", "10.75.218.105"]
        }
    }
}

```

```

    }
  },
  "sbrNetwork": {
    "name": "ext-net4",
    "ipVersion": "IPv4",
    "fixedIps": {
      "sbrNetworkIps": ["10.75.218.231", "10.75.218.236", "10.75.218.244"]
    }
  },
  "xsiNetwork": [
    {
      "name": "ext-net4",
      "ipVersion": "IPv4",
      "fixedIps": {
        "dampXsiIps": ["10.75.218.238", "10.75.218.47"],
        "ipfeXsiIps": ["10.75.218.239", "10.75.218.93"],
        "stpXsiIps": ["10.75.218.214", "10.75.218.19"],
        "primaryUdrXsiIps": ["10.75.218.228"],
        "secondaryUdrXsiIps": ["10.75.218.235"]
      }
    },
    {
      "name": "ext-net4",
      "ipVersion": "IPv4",
      "fixedIps": {
        "dampXsiIps": ["10.75.218.230", "10.75.218.225"],
        "ipfeXsiIps": ["10.75.218.49", "10.75.218.245"],
        "stpXsiIps": ["10.75.218.170", "10.75.218.224"],
        "primaryUdrXsiIps": ["10.75.218.233"],
        "secondaryUdrXsiIps": ["10.75.218.227"]
      }
    }
  ],
  "ntpServerIp": "10.250.32.10",
  "primaryNoamVmName": "NOAM00-d8fc80a2",

```

```
"noamSgName": "dsrNetworkOam_NOAM_d8fc80a2_SG"
}}
```

Sample Response

```
Instantiating the signaling VNF with DIAMETER+SS7+SBR+UDR response

202 Accepted

Headers:

location: https:// <<VNFM HOST IP>>:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
f00678f4-ea8e-417f-9c5a-e126926db402

date: Wed, 13 Feb 2019 09:55:01 GMT

content-length: 0

content-type: application/xml
```

Sample Request

For signaling flavors with DIAMETER+SS7+SBR+UDR with multiple xsi (2 XSI interface) for Dynamic IP deployment model

```
{ "flavourId": "DIAMETER+SS7+SBR+UDR",
  "instantiationLevelId": "small",
  "extVirtualLinks": "extVirtualLinks",
  "extManagedVirtualLinks": [
    {
      "id": "id1",
      "virtualLinkDescId": "active NOAM",
      "resourceId": "790bf9f7-8834-4c3a-bd17-5544ef5e6848"
    },
    {
      "id": "id2",
      "virtualLinkDescId": "standby NOAM",
      "resourceId": "1776d877-f643-45d6-b6da-bf1a540a01d1"
    }
  ],
}
```

```

"vimConnectionInfo": [
  {
    "id": "vimid",
    "vimType": "OpenStack",
    "interfaceInfo": {
      "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
    },
    "accessInfo": {
      "username": "dsrvnfm",
      "password": "xxxxxx",
      "userDomain": "Default",
      "projectDomain": "default",
      "tenant": "dsrvnfm"
    }
  }
],
"localizationLanguage": "localizationLanguage",
"additionalParams": {
  "xmiNetwork": {
    "name": "ext-net4",
    "ipVersion": "IPv4"
  },
  "sbrNetwork": {
    "name": "ext-net4",
    "ipVersion": "IPv4"
  },
  "xsiNetwork": [
    {
      "name": "ext-net4",
      "ipVersion": "IPv4"
    },
    {
      "name": "ext-net4",
      "ipVersion": "IPv4"
    }
  ],
  "ntpServerIp": "10.250.32.10",

```

```

"primaryNoamVmName": "NOAM00-d8fc80a2",
"noamSgName": "dsrNetworkOam_NOAM_d8fc80a2_SG"
}}

```

Sample Response

```

Instantiating the signaling VNF with DIAMETER+SS7+SBR+UDR response
202 Accepted

Headers:
location: https:// <<VNFM HOST IP>>:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
f00678f4-ea8e-417f-9c5a-e126926db402
date: Wed, 13 Feb 2019 09:55:01 GMT
content-length: 0
content-type: application/xml

```

Notes:

- The 202 response means that the request was accepted for processing. The VNF might take up to 15 minutes to become fully operational. Use the DSR GUI to determine when the VNF is operational.
- If the VNFM creates a VNF that is operational, but has no DIAMETER configuration data, then create the necessary configuration data (peers, connections, etc.) to perform DIAMETER routing.

Table 9. Parameters and Definitions for Signaling VNF with Multiple XSI

Parameter	Definitions
flavourId	Identifier of the VNF deployment flavor to be instantiated
instantiationLevelId	Identifier of the instantiation level of the deployment flavor to be instantiated. If not present, the default instantiation level is HA.
resourceId	The identifier of the resource in the scope of the VIM or the resource provider
xmiNetwork	Network that is used to provide access to the DSR entities (GUI, ssh), and for inter-site communication
imiNetwork	Network used for internal communication of DSR entities
name	Network name, for example; ext-net
ipVersion	IP version IPv4 or IPv6
xsiNetwork	Network used for DSR signaling traffic
ntpServerIp	IP of the NTP server
primaryNoamVmName	Primary Active DSR NOAM VM name
noamSgName	The server group of the NOAM VM

Important Note:

The flavor ID must be selected based on the VMs to be deployed and the instantiation level must be selected based on the number of VMs required.

Only the IPs of the required VM must be provided in the `fixedIp` parameter.

For Example:

"flavorId": "DIAMETER+SS7", "instantiationLevelId": "small" - This brings up 2 SOAM, 2 DAMP, 2 IPFE, 2 STP servers.

The user must provide `primarySoamXmiIp(1)`, `secondarySoamXmiIp(1)`, `soamVip(1)`, `dampXmiIps(2)`, `ipfeXmiIps(2)`, `stpXmiIps(2)`, `dampXsiIps(2)`, `ipfeXsiIps(2)`, `stpXsiIps(2)`

Detailed explanation of XMI, IMI and XSI Network

The detailed explanation of XMI and XSI Network for the additional parameters are provided below:

For XMI Network

```
"xmiNetwork":{
  "name":"<NAME of the network of XMI IPS >",
  "ipVersion":"IPv4",
  "fixedIps":{
    "primarySoamXmiIp":"<ACTIVE SOAM XMI IP>",
    "secondarySoamXmiIp":"<STANDBY SOAM XMI IP>",
    "soamVip":"<SOAM VIP>",
    "dampXmiIps":[
      "<DAMP 00 XMI IP>",
      "<DAMP 01 XMI IP>"
    ],
    "ipfeXmiIps":[
      "<IPFE 00 XMI IP>",
      "<IPFE 01 XMI IP>"
    ],
    "stpXmiIps":[
      "<STP 00 XMI IP>",
      "<STP 01 XMI IP>"
    ]
  }
}
```

For IMI Network


```

"xmiNetwork":{
  "name":"<NAME of the network of XMI IPS >",
  "ipVersion":"IPv4",
  "fixedIps":{
    "primarySoamImiIp":"<ACTIVE SOAM IMI IP>",
    "secondarySoamImiIp":"<STANDBY SOAM IMI IP>",
    "dampImiIps":[
      "<DAMP 00 IMI IP>",
      "<DAMP 01 IMI IP>"
    ],
    "ipfeImiIps":[
      "<IPFE 00 IMI IP>",
      "<IPFE 01 IMI IP>"
    ],
    "stpImiIps":[
      "<STP 00 IMI IP>",
      "<STP 01 IMI IP>"
    ]
  }
}

```

For XSI Network

```

"xsiNetwork":[
  {
    "name":"<NAME of the network of XSI 1>",
    "ipVersion":"IPv4",
    "fixedIps":{
      "dampXsiIps":[
        "<DAMP00 XSI 1 IP>",
        "<DAMP 01 XSI 1 IP>"
      ],
      "ipfeXsiIps":[
        "<IPFE00 XSI 1 IP>",
        "<IPFE01 XSI 1 IP>"
      ],
      "stpXsiIps":[
        "<STP00 XSI 1 IP>",
        "<STP01 XSI 1 IP>"
      ]
    }
  },
  {
    "name":"<NAME of the network of XSI 2>",

```

```

        "ipVersion":"IPv4",
        "fixedIps":{
            "dampXsiIps":[
                "<DAMP00 XSI 2 IP>",
                "<DAMP01 XSI 2 IP>"
            ],
            "ipfeXsiIps":[
                "<IPFE00 XSI 2 IP>",
                "<IPFE01 XSI 2 IP>"
            ],
            "stpXsiIps":[
                "<STP00 XSI 2 IP>",
                "<STP01 XSI 2 IP>"
            ]
        }
    }
}
]

```

Table 10 describes the parameters used for sending request to VNFM.

Table 10. Parameters and Definitions for Signaling VNF with Multiple XSI

Parameter	Definitions
flavourId	Identifier of the VNF deployment flavor to be instantiated
instantiationLevelId	Identifier of the instantiation level of the deployment flavor to be instantiated. If not present, the default instantiation level as declared in the VNFD is instantiated.
resourceId	The identifier of the resource (active NOAM and then standBy NOAM) in the scope of the VIM or the resource provider
xmiNetwork	Network that is used to provide access to the DSR entities (GUI, ssh), and for inter-site communication
xsiNetwork	Network used for DSR signaling traffic
imiNetwork	Network used to provide access to the DSR entities (GUI, ssh), and for internal communication
name	Network name, for example; ext-net
ipVersion	IP version IPv4 or IPv6
xsiNetwork	Network that is used for DSR signaling traffic
ntpServerIp	IP of the NTP server
primaryNoamVmName	Name of primary NOAM VM on which the configured XML is loaded

Parameter	Definitions
noamSgName	The server group of the NOAM VM
primarySoamXmiIp	IP address of primary SOAM
secondarySoamXmiIp	IP address of secondary SOAM
soamVip	VIP of SOAM
dampXmiIps	List of DAMP external management IPs (only if DAMPs are being instantiated)
ipfeXmiIps	List of IPFE external management IPs (only if IPFEs are being instantiated)
stpXmiIps	List of vSTP external management IPs (only if STPs are being instantiated)
dampXsiIps	List of DAMP signaling IPs (only if DAMPs are being instantiated)
ipfeXsiIps	List of IPFE signaling IPs (only if IPFEs are being instantiated)
stpXsiIps	List of STP signaling IPs (only if STPs are being instantiated)
primaryUdrXmiIp	IP address of primary UDR (only if UDRs are being instantiated)
secondaryUdrXmiIp	IP address of secondary UDR (only if UDRs are being instantiated)
udrVip	VIP address of UDR (only if UDRs are being instantiated)
primaryUdrXsiIps	List of primary UDR signaling IPs (only if UDRs are being instantiated)
secondaryUdrXsiIps	List of secondary UDR signaling IPs (only if UDRs are being instantiated)
sbrXmiIps	List of SBR external management IPs (only if SBRs are being instantiated)
sbrNetworkIps	List of SBR replication port IPs (only if SBRs are being instantiated)
primarySoamImiIp	IP address of primary SOAM for IMI
secondarySoamImiIp	IP address of secondary SOAM for IMI
dampImiIps	List of DAMP internal management IPs (only if DAMPs are being instantiated)
ipfeImiIps	List of IPFE internal management IPs (only if IPFEs are being instantiated)
stpImiIps	List of vSTP internal management IPs (only if STPs are being instantiated)
primaryUdrImiIp	IP address of primary UDR for IMI (only if UDRs are being instantiated)
secondaryUdrImiIp	IP address of secondary UDR for IMI (only if UDRs are being instantiated)
sbrImiIps	List of SBR internal management IPs (only if SBRs are being instantiated)

7.6 Instantiating Multiple Signaling VNFs

To instantiate multiple Signaling VNFs, simply repeat the above procedures. You would need to create another DSR Signaling VNF instance, and you must deploy each Signaling VNF on a separate OpenStack instance.

Note: For lab installations, a separate tenant on the same OpenStack instance is acceptable.

7.7 Instantiating the APIGW VNF

To start APIGW deployment, it is required to instantiate an APIGW VNF. Before deploying the VNF, make sure the following information is available:

The VNF ID for a previously created APIGW VNF instance.

Information about the OpenStack instance on which the VNF must be deployed:

- OpenStack Controller URI
- User Domain Name
- Project Domain Id
- Username
- Password
- Tenant name

The name of a public network in the selected OpenStack instance that will carry APIGW traffic.

The name of a public network in the selected OpenStack instance that will carry signaling traffic.

Note: This should be a different network than the one that carries APIGW traffic

The IP of an NTP server accessible by VMs within the selected OpenStack instance. The OpenStack controller that controls the selected OpenStack instance, normally hosts an NTP server, and is often a good choice.

For more information about the full list of all inputs and possible outputs of the **instantiate VNF** command, see **ETSI NFV-SOL 003**, section **5.4.4.3.1**, or the DSR VNFM Swagger specification. Swagger specifications can be found post VNFM installation at (https://<VNFM_IP>:8443/docs/vnfm/).

Table 11 contains the supported Instantiation levels to instantiate the VNF resource for DSR APIGW VNF.

Table 11. Supported Instantiation levels for DSR APIGW VNF

APIGW Flavors supported by VNFM	Small			Medium			Large		
	ADMIN	APP	DB	ADMIN	APP	DB	ADMIN	APP	DB
APIGW	1	1	Active/Standby	1	2	Active/Standby	1	3	Active/Standby

Sample Request

Instantiating APIGW Request generated.

Resource URL: `https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances/< VNF ID received from create request>/instantiate`

Accept: `application/json`

Content-Type: `application/json`

```
{
  "flavourId": "APIGW",
  "instantiationLevelId": "small",
  "extVirtualLinks": "extVirtualLinks",
  "extManagedVirtualLinks": [],
  "vimConnectionInfo": [
    {
      "id": "vimid",
      "vimType": "OpenStack",
      "interfaceInfo": {
        "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
      },
      "accessInfo": {
        "username": "dsrat.user",
        "password": "xxxx",
        "userDomain": "Default",
        "projectDomain": "default",
        "tenant": "DSR AT Dev 2"
      }
    }
  ],
  "localizationLanguage": "localizationLanguage",
  "additionalParams": {
    "ntpServerIp": "10.250.32.10",
    "keyName": "apiGwKey",
    "xmiNetwork": {
      "name": "ext-net3",
      "ipVersion": "IPv4"
    },
    "xsiNetwork": {
      "name": "ext-net2",
```

```

    "ipVersion": "IPv4"
  },
  "externalLoadBalancer": "10.10.10.10",
  "mtu": "9000",
  "dsrMPLList": "10.10.10.4:49152",
  "appServersVolumeIds": ["320f3557-9a0a-4c13-9d19-d4f0f755b941"]
}
}

```

Sample Response

Instantiating APIGW Request

```

202 Accepted
Headers:
{
  location: https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
fb21f9d3-43ad-46cd-a03f-7220bb36a5c6
  date: Tue, 29 Jan 2019 10:39:24 GMT
  content-length: 0 content-type:
  application/xml
}

```

Notes:

The 202 response means that the request was accepted for processing. The VNF might take up to 6 minutes to become fully operational. Use the DSR GUI to determine when the VNF is operational.

The supported flavor is **APIGW**.

The keyName is the name of the key that generates public & private key in openstack dynamically while creating stack and this key is used to communicate over admin to app server & DB server.

One push script executes and enables the OCSG. After successful execution of one push script, the Admin portal and the App portals GUI comes up.

With DSR 8.4 and VNFM 4.0, APIGW is automatically configured, there is no need of manual configuration.

Table 12 describes the parameters used for sending request to VNFM.

Table 12. Parameters and Definitions for APIGW VNF

Parameter	Definitions
flavourId	Identifier of the VNF deployment flavor to be instantiated
instantiationLevelId	Identifier of the instantiation level of the deployment flavor to be instantiated. If not present, the default instantiation level as declared in the VNFD is instantiated.
xmiNetwork	Network that is used to provide access to the DSR entities (GUI, ssh), and for inter-site communication

imiNetwork	Network used for internal communication of DSR entities
xsiNetwork	Network used for DSR signaling traffic
ntpServerIp	IP of the NTP server
keyName	Name of key-pair to be generated
externalLoadBalancer	The external load balancer IP where the API is exposed on
Mtu	Maximum transfer Unit to do scp file. For different cloud values will be different. (Ex: For oort and mvl-dev1 mtu value will be 9000 and for dpc1 it will be 1500.)
dsrMPList	List of DSR MPs
appServersVolumeIds	A JSON Array containing the volume IDs of the volumes created by the user that is mounted to the individual App Servers. The size/length of this array should be equal to the number of App Servers, which in turn depends on the flavor chosen by the user.

7.8 Instantiating the IDIH VNF

To start IDIH deployment, it is required to instantiate a signaling VNF. Before deploying the VNF, make sure the following information is available:

The VNF ID for a previously created IDIH VNF instance.

Information about the OpenStack instance on which the VNF must be deployed:

- OpenStack Controller URI
- User Domain Name
- Project Domain Id
- Username
- Password
- Tenant name

The name of a public network in the selected OpenStack instance that will carry the IDIH traffic.

The IP of an NTP server accessible by VMs within the selected OpenStack instance. The OpenStack controller that controls the selected OpenStack instance normally hosts an NTP server, and is often a good choice.

The network ID of the private network in the selected OpenStack instance that will carry OAM traffic. A signaling stack must be brought up first and then the ID of the internal network generated from this stack must be used for instantiating IDIH.

The name of the internal private network in the selected OpenStack instance that will allow communication between Application, Mediation, and Database servers.

For more information about the full list of all inputs and possible outputs of the **instantiate VNF** command, see **ETSI NFV-SOL 003**, section **5.4.4.3.1**, or the DSR VNFM Swagger specification. Swagger specifications can be found post VNFM installation at (https://<VNFM_IP>:8443/docs/vnfm/).

7.8.1 Determining the Signaling IMI resource ID:

1. Navigate to **Project -> Network -> Networks**.

2. Open the Network used for intra-site communication with Signaling VNF (imi).
3. The IMI resource ID is the ID of this network.

The following table informs about the supported Instantiation levels to Instantiate VNF resource for IDIH VNF:

IDIH Flavors supported by VNFM	Small		
	APP	MEDIATION	DB
IDIH	1	1	1

Sample Request

Instantiating IDIH Request for dynamic IP deployment

Resource URL: `https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances/<VNF ID received from create request>/instantiate`

Accept: `application/json`

Content-Type: `application/json`

```
{
  "flavourId":"IDIH",
  "instantiationLevelId":"small",
  "extVirtualLinks":"extVirtualLinks",
  "extManagedVirtualLinks":[
    {
      "id":"id1",
      "virtualLinkDescId":"Network ID of the network used for intra-site
communication(imi) with Signalling VNF",
      "resourceId":"aae72b3d-d189-4464-a217-58bb0320065b"
    }
  ],
  "vimConnectionInfo":[
    {
      "id":"vimid",
      "vimType":"OpenStack",
      "interfaceInfo":{
        "controllerUri":"https://oortcloud.us.oracle.com:5000/v3"
      },
      "accessInfo":{
        "username":"dsrat.user",
        "password":"xxxx",
        "userDomain":"Default",
        "projectDomain": "default",

```



```

        "tenant": "DSRAT_Feature_Test4"
    }
}
],
"localizationLanguage": "localizationLanguage",
"additionalParams": {
    "ntpServerIp": "10.250.32.10",
    "xmiNetwork": {
        "name": "ext-net3",
        "ipVersion": "IPv4"
    },
    "idihIntNetwork": {
        "idihIntPrivateNetwork": "test",
        "idihIntPrivateSubnet": "test-sub"
    }
}
}

```

Instantiating IDIH Request for fixed IP deployment

```

{
    "flavourId": "IDIH",
    "instantiationLevelId": "small",
    "extVirtualLinks": "extVirtualLinks",
    "extManagedVirtualLinks": [
        {
            "id": "id1",
            "virtualLinkDescId": " Network ID of the network used for intra-site communication (imi) with Signalling VNF",
            "resourceId": "aae72b3d-d189-4464-a217-58bb0320065b"
        }
    ],
    "vimConnectionInfo": [
        {
            "id": "vimid",
            "vimType": "OpenStack",
            "interfaceInfo": {
                "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
            },
            "accessInfo": {
                "username": "dsrat.user",
                "password": "xxxx",
                "userDomain": "Default",
            }
        }
    ]
}

```

```

        "projectDomain": "default",
        "tenant": "DSRAT_Feature_Test4"
    }
},
"localizationLanguage": "localizationLanguage",
"additionalParams": {
    "ntpServerIp": "10.250.32.10",
    "xmiNetwork": {
        "name": "ext-net3",
        "ipVersion": "IPv4",

        "fixedIps": {
            "idihDbXmiIp": "10.75.218.30",
            "idihMedXmiIp": "10.75.218.19",
            "idihAppXmiIp": "10.75.218.49"
        }
    },
    "idihIntNetwork": {
        "idihIntPrivateNetwork": "test",
        "idihIntPrivateSubnet": "test-sub"
    }
}
}

```

Sample Response

Instantiating IDIH Request

```

202 Accepted
Headers:
{
    location: https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
fb21f9d3-43ad-46cd-a03f-7220bb36a5c6
    date: Tue, 29 Jan 2019 10:39:24 GMT
    content-length: 0    content-type:
    application/xml
}

```

Notes:

The 202 response means the request was accepted for processing. The VNF might take up to 6 minutes to become fully operational. Use the DSR GUI to determine when the VNF is operational.

The supported flavor is IDIH.

Table 13 describes the parameters used for sending request to VNFM.

Table 13. Parameters and Definitions for IDIH VNF

Parameter	Definitions
flavourId	Identifier of the VNF deployment flavor to be instantiated
instantiationLevelId	Identifier of the instantiation level of the deployment flavor to be instantiated. If not present, the default instantiation level as declared in the VNFD is instantiated.
resourceId	The Identifier of the Private network (imi) of the Signaling VNF
xmiNetwork	Network that is used to provide access to the DSR entities (GUI, ssh), and for inter-site communication
IdihIntNetwork	Private network for communication between application, mediation and database servers
ntpServerIp	IP of the NTP server
idihDbXmiIp	Fixed IP address of IDIH database server
idihMedXmiIp	Fixed IP address of IDIH mediation server
idihAppXmiIp	Fixed IP address of IDIH application server

7.9 Instantiating the SDS Network OAM VNF

SDS NOAM is a setup of three servers:

- Primary Noam
- Secondary Noam
- Query Server

In order to start a SDS deployment, it is required to instantiate a SDS Network OAM VNF. Before deploying the VNF, the following information must be available:

- The VNF ID for a previously created SDS network OAM VNF instance.
- Information about the OpenStack instance on which the VNF must be deployed:
 - o OpenStack Controller URI
 - o User Domain Name
 - o Project Domain Id
 - o Username
 - o Password
 - o Tenant name
- The name of a public network in the selected OpenStack instance that will carry the OAM traffic.
- The IP of an NTP server accessible by VMs within the selected OpenStack instance. The OpenStack controller that controls the selected OpenStack instance normally hosts an NTP server, and is often a good choice.
 - o Supported for IPv6 networks - `ipVersion` should be "IPv6" in the request Body. The GUI can be accessed by the following URL: `https://[<SDS-NOAM-vIP>]`
For example: `https://[fd0d:deba:d97c:2c:6e41:6aff:fec7:80bf]`

For more information about the full list of all inputs and possible outputs of the **instantiate VNF** command, see **ETSI NFV-SOL 003**, section **5.4.4.3.1**, or the DSR VNFM Swagger specification. Swagger specifications can be found post VNFM installation at (`https://<VNFM IP>:8443/docs/vnfm/`).

Sample Request:

Sample Request for DYNAMIC IP deployment model

Resource URL: `https://<VNFM HOST IP>:8443/vnfm/v1/vnf_instances/<VNF ID received from create request>/instantiate`

Accept: `application/json`

Content-Type: `application/json`

```
{
  "flavourId": "SDS NOAM",
  "instantiationLevelId": "HA",
  "extVirtualLinks": "extVirtualLinks",
    "extManagedVirtualLinks": [],

  "vimConnectionInfo": [ {
    "id": "vimid",
    "vimType": "OpenStack",
    "interfaceInfo": {
      "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
    },
    "accessInfo": {
      "username": "dsrsci.user",
      "password": "xxxxxx",
      "userDomain": "Default",
      "projectDomain": "default",
      "tenant": "DSR CI"
    }
  }
],
  "localizationLanguage": "localizationLanguage",
  "additionalParams": {
    "xmiNetwork": {
      "name": "ext-net8",
      "ipVersion": "IPv4"
    },
    "ntpServerIp": "10.250.32.10"
  }
}
```

Sample Request for Fixed IP deployment model

```
{
```

```

"flavourId": "SDS NOAM",
"instantiationLevelId": "HA",
"extVirtualLinks": "extVirtualLinks",
    "extManagedVirtualLinks": [],

"vimConnectionInfo":[ {
    "id": "vimid",
    "vimType": "OpenStack",
    "interfaceInfo": {
        "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
    },
    "accessInfo": {
        "username": "dsrcli.user",
        "password": "xxxxx",
        "domainuserDomain": "Default",
        "projectDomain": "default",
        "tenant": "DSR CI"
    }
}],
"localizationLanguage": "localizationLanguage",
"additionalParams": {
    "xmiNetwork": {
        "name": "ext-net8",
        "ipVersion": "IPv4",
        "fixedIps":
            {
                "primarySdsNoamIp": "10.75.218.50",
                "secondarySdsNoamIp": "10.75.218.49",
                "sdsQsIp": "10.75.218.134",
                "sdsNoamVip": "10.75.218.114"
            }
    },
    "imiNetwork": {
        "name": "imi-net",
        "ipVersion": "IPv4",
        "fixedIps":

```

```

    {
        "primarySdsNoamImiIp": "192.167.2.5",
        "secondarySdsNoamImiIp": "192.167.2.4",
        "sdsQsImiIp": "192.167.2.3"
    },
    "ntpServerIp": "10.250.32.10"
}

```

Sample response:

```

202 Accepted
Headers:
{
    location:      https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
fb21f9d3-43ad-46cd-a03f-7220bb36a5c6
    date: Tue, 29 Jan 2019 10:39:24 GMT
    content-length: 0   content-type:
    application/xml
}

```

Note:

- The 202 response means that the request was accepted for processing. The VNF might take up to 15 minutes to become fully operational. Use the DSR GUI to determine when the VNF is operational.
- If the VNFM creates a VNF that is operational, but has no signaling VNFs, then it is required to deploy one or more signaling VNF, and create the DIAMETER configuration data (peers, connections, etc.) for those VNFs to perform DIAMETER routing.
- After NOAM VNF deployment, standby NOAM is automatically changed to "**Force StandBy**", purposely to avoid any switchover while DSR Signaling VNF is deployed. Once DSR Signaling Site is deployed and no more Life Cycle Management operations are planned, make "**Force Standby**" NOAM as "**Active**" by changing the "**Max Allowed HA Role**" to "**Active**" on "**Status & Manage -> HA** from **Active NOAM GUI**."
- The supported SDS NOAM Flavor is SDS NOAM.
- The supported SDS NOAM Flavor instantiation level id is HA, which creates 2 SDS NOAMs and 1 Query Server.

The following table describes the parameters used for sending request to VNFM:

Table 14. Parameters and Definitions for SDS Network OAM VNF

Parameter	Definitions
flavourId	Identifier of the VNF deployment flavor to be instantiated

Parameter	Definitions
xmiNetwork	Network that is used to provide access to the DSR entities (GUI, ssh), and for inter-site communication
imiNetwork	Network used for internal communication of DSR entities
ntpServerIp	IP of the NTP server
fixedIps	Json object in network to provide IP address
primarySdsNoamIp	IP address for primary SDS NOAM IP
secondarySdsNoamIp	IP address for secondary SDS NOAM IP
sdsQsIp	IP address for SDS Query Server VIP
sdsNoamVip	IP address for SDS NOAM VIP
primarySdsNoamImiIp	IP address for primary SDS NOAM IP of IMI
secondarySdsNoamImiIp	IP address for secondary SDS NOAM IP of IMI
sdsQsImiIp	IP address for SDS Query Server IP of IMI

7.10 Instantiating the SDS DR Network OAM VNF

SDS DRNOAM is the Disaster Recovery SDS NOAM site. In case both the Active and Standby SDS NOAM of Primary site fails, then the operator can make SDS DRNOAM as the Primary Site and can continue the operations without any disturbance.

When a setup is configured with a SDS DR NOAM then the first SDS NOAM SG is treated as the Primary NOAM Site and the second SDS NOAM SG is treated as Secondary NOAM site.

SDS DR NOAM is a setup of three servers:

- Primary Noam
- Secondary Noam
- Query Server

In order to instantiate a SDS DR Network OAM VNF, the following information must be available:

- The VNF ID for a previously created SDS DR network OAM VNF instance.
- Information about the OpenStack instance on which the VNF must be deployed:
 - OpenStack Controller URI
 - User Domain Name
 - Project Domain Id
 - Username
 - Password
 - Tenant name
- The name of a public network in the selected OpenStack instance that will carry the OAM traffic.
- OpenStack resource IDs for the XMI IPs from both SDS NOAM VMs.

Note: The resource IDs can be obtain by examining the SDS Network OAM stack to which the identified SDS DR NOAM VNF is attached.
- Name of Active Primary SDS NOAM VM.
- The IP of an NTP server accessible by VMs within the selected OpenStack instance. The OpenStack controller that controls the selected OpenStack instance normally hosts an NTP server, and is often a good choice.

7.10.1 Determining the SDS DR NOAM XMI resource IDs

The following facts must be considered before proceeding with SDS DR NOAM site creation:

- SDS DRNOAM site must be created on separate tenant.
- SDS DRNOAM site is referred as Secondary NOAM. Therefore, we have two sites, Primary and Secondary. Secondary Site configuration is done on Primary Active SDS NOAM.
- In the Primary Active SDS NOAM, when second SDS NOAM Server Group gets created, it automatically becomes Secondary.
- The Primary Active SDS NOAM communicates to the Secondary Active SDS NOAM through existing comcol replication and merging mechanism.
- The Secondary SDS NOAM Site is optional and does not require to be deployed at the same time as of the Primary SDS NOAM.

From the OpenStack GUI:

1. Change your view to the tenant on which the DSR Network OAM VNF was deployed.
2. Go to **Project->Network->Network Topology**. A diagram of all VMs in the tenant is displayed.
Note: The diagram may take a few minutes to display.
3. Click on one of the NOAM VMs.
4. A pop-up appears having information about the specific NOAM VM.
5. Save the resource ID for the XMI port provided in the IP Addresses section of the pop-up.
Note: The IP Addresses section of the popup contains information about the network ports and resource IDs, assigned to the VM.
6. Repeat the previous step for the other NOAM VM.

For more information about the full list of all inputs and possible outputs of the **instantiate VNF** command, see **ETSI NFV-SOL 003**, section **5.4.4.3.1**, or the DSR VNFM Swagger specification. Swagger specifications can be found post VNFM installation at (https://<VNFM_IP>:8443/docs/vnfm/).

Sample Request:

Instantiating SDS DR NOAM Request for DYNAMIC IP deployment model

Resource URL: `https://<<VNFM_HOST_IP>>:8443/vnfm/v1/vnf_instances/<VNF_ID_received_from_create_request>/instantiate`

Accept: `application/json`

Content-Type: `application/json`

```
{
  "flavourId": "SDS DR NOAM",
  "instantiationLevelId": "HA",
  "extVirtualLinks": "extVirtualLinks",
  "extManagedVirtualLinks": [{
    "id": "id1",
    "virtualLinkDescId": "active SDS NOAM XMI",
    "resourceId": "156d73cf-6e44-456b-a661-14bd0cc2b43c"
  },
  {
    "id": "id2",
```



```

        "virtualLinkDescId": "standby SDS NOAM XMI",
        "resourceId": "5c638770-5585-44c7-97c7-b4a52a26e5ec"
    },
    ],
    "vimConnectionInfo": [ {
        "id": "vimid",
        "vimType": "OpenStack",
        "interfaceInfo": {
            "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
        },
        "accessInfo": {
            "username": "dsrcli.user",
            "password": "xxxxxx",
            "userDomain": "Default",
            "projectDomain": "default",
            "tenant": "DSR CI"
        }
    }
    ],
    "localizationLanguage": "localizationLanguage",
    "additionalParams": {
        "xmiNetwork": {
            "name": "ext-net3",
            "ipVersion": "IPv4"
        },
        "ntpServerIp": "10.250.32.10",
        "primarySdsNoamVmName": "SDS-NOAM00-ea47f4b1"
    }
}

```

Instantiating SDS DR NOAM Request for Fixed IP deployment model

```

{
    "flavourId": "SDS DR NOAM",
    "instantiationLevelId": "HA",
    "extVirtualLinks": "extVirtualLinks",
    "extManagedVirtualLinks": [{
        "id": "id1",

```

```

        "virtualLinkDescId": "active SDS NOAM XMI",
        "resourceId": "156d73cf-6e44-456b-a661-14bd0cc2b43c"
    },
    {
        "id": "id2",
        "virtualLinkDescId": "standby SDS NOAM XMI",
        "resourceId": "5c638770-5585-44c7-97c7-b4a52a26e5ec"
    }
],
"vimConnectionInfo": [ {
    "id": "vimid",
    "vimType": "OpenStack",
    "interfaceInfo": {
        "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
    },
    "accessInfo": {
        "username": "dsrsci.user",
        "password": "xxxxxx",
        "userDomain": "Default",
        "projectDomain": "default",
        "tenant": "DSR CI"
    }
}
]],
"localizationLanguage": "localizationLanguage",
"additionalParams": {
    "xmiNetwork": {
        "name": "ext-net3",
        "ipVersion": "IPv4",
        "fixedIps":
        {
            "sdsDrPrimaryNoamIp": "10.75.218.50",
            "sdsDrSecondaryNoamIp": "10.75.218.49",
            "sdsDrQueryServerIp": "10.75.218.134",
            "sdsDrNoamVip": "10.75.218.114"
        }
    }
},

```

```

    "imiNetwork": {
      "name": "imi-net",
      "ipVersion": "IPv4",
      "fixedIps":
        {
          "sdsDrPrimaryNoamImiIp": "192.167.2.5",
          "sdsDrSecondaryNoamImiIp": "192.167.2.4",
          "sdsDrQueryServerImiIp": "192.167.2.7"
        }
    },
    "ntpServerIp": "10.250.32.10",
    "primarySdsNoamVmName": "SDS-NOAM00-ea47f4b1"
  }
}

```

Sample response:

```

202 Accepted
Headers:
{
  location:      https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
fb21f9d3-43ad-46cd-a03f-7220bb36a5c6
  date: Tue, 21 Feb 2019 10:39:24 GMT
  content-length: 0   content-type:
  application/xml
}

```

Note:

- The 202 response means that the request was accepted for processing. The VNF might take up to 15 minutes to become fully operational. Use the DSR GUI to determine when the VNF is operational.
- The supported SDS DR NOAM Flavor is SDS DR NOAM.
- The supported SDS DR NOAM Flavor instantiation level id is HA, which creates 2 SDS NOAMs and 1 Query Server.
- Supported for IPv6 networks - ipVersion should be "IPv6" in the request Body.

The following table describes the parameters used for sending request to VNFM:

Table 15. Parameters and Definitions SDS DR Network OAM VNF

Parameter	Definitions
flavourId	Identifier of the VNF deployment flavor to be instantiated

Parameter	Definitions
instantiationLevelId	Identifier of the instantiation level of the deployment flavor to be instantiated. If not present, the default instantiation level is HA.
resourceId	The identifier of the resource (active and then standby SDS NOAM XMI) in the scope of the VIM or the resource provider.
xmiNetwork	Network that is used to provide access to the DSR entities (GUI, ssh), and for inter-site communication
imiNetwork	Network used for internal communication of DSR entities
name	Network name, for example; ext-net
ipVersion	IP version IPv4 or IPv6
ntpServerIp	IP of the NTP server
primarySdsNoamVmName	Primary Active SDS NOAM VM name
sdsDrPrimaryNoamIp	XMI IP of the Primary SDS DR NOAM
sdsDrSecondaryNoamIp	XMI IP of the Secondary SDS DR NOAM
sdsDrQueryServerIp	XMI IP of the SDS DR QUERY NOAM
sdsDrNoamVip	VIP of the SDS DR NOAM
sdsDrPrimaryNoamImiIp	IMI IP of the Primary SDS DR NOAM
sdsDrSecondaryNoamImiIp	IMI IP of the Secondary SDS DR NOAM
sdsDrQueryServerImiIp	IMI IP of the Primary SDS DR NOAM

7.11 Instantiating the SDS Signaling VNF

In order to deploy the SDS signaling VNF, the following information must be available:

- A previously instantiated SDS network OAM VNF.
- The VNF ID for a previously created SDS signaling VNF instance.
- Information about the OpenStack instance on which the VNF must be deployed:
 - OpenStack Controller URI
 - User Domain Name
 - Project Domain Id
 - Username
 - Password
 - Tenant name
- The name of the xmi public network in the selected OpenStack instance that will carry traffic.
- The IP address of the NTP server accessible by VMs within the selected OpenStack instance.
- The OpenStack controller that controls the selected OpenStack instance normally hosts an NTP server, and is often a good choice.
- OpenStack resource IDs for the IMI IP from DSR Signaling and XMI IPs from both NOAM VMs.
Note: The resource IDs can be obtain by examining the SDS Network OAM stack and DSR Signaling stack to which the identified SDS signaling VNF would be attached.
- Name of the Active NOAM VM.
Note: To avoid switchover of Active NOAM, make the StandBy NOAM as "**Forced Standby**" by changing the "**Max Allowed HA Role**" to "**Standby**" on "**Status & Manage -> HA** from **Active NOAM GUI**."
- Name of the NOAM SG

Note: After SDS deployment, the Max Allowed HA Role of Query Server is expected to be Observer but it is Standby. Manually change the Max Allowed HA Role of Query Server from Standby to Observer as follows:

Login to Active SDS Noam GUI and navigate to **Status & Manage --> HA --> Edit-->Change the role of Query Server to Observer**, and click **OK**

- Supported for IPv6 networks - ipVersion should be "IPv6" in the request Body.

The following image illustrates the VNF instantiation:

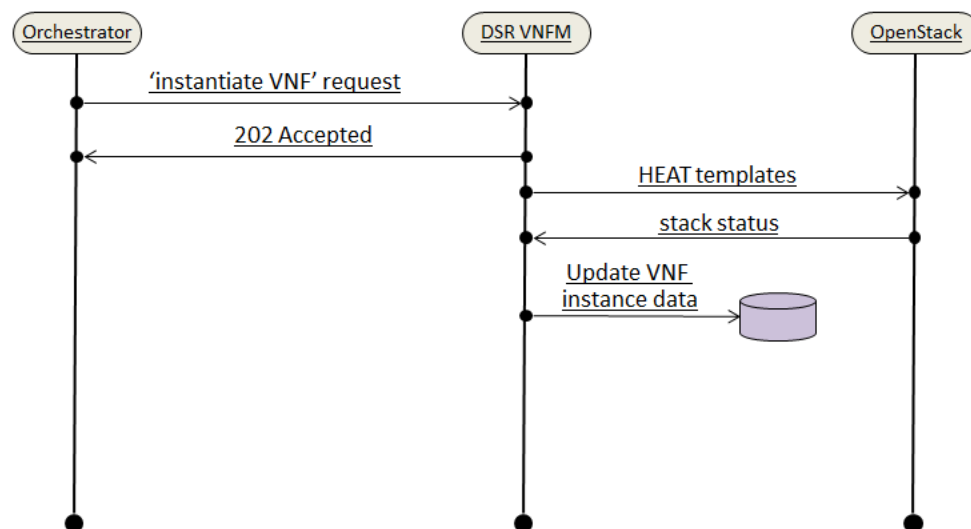


Figure 6. VNF Create Instance Request

The following table informs about the supported Instantiation levels to Instantiate VNF resource for SDS Signaling VNF:

Signaling Flavors supported by VNFM	Small	Medium	Large
	DP Server	DP Server	DP Server
SDS Signaling	1	2	3

The number of DP-SOAM will be 2 for any instantiation level.

7.11.1 Determining the Signaling IMI resource IDs

From the OpenStack GUI:

1. Navigate to **Project -> Network -> Networks**
2. Open the Network used for intra - site communication with Signaling VNF (imi).
3. The IMI resource ID is the ID of this network.

7.11.2 Determining the SDS NOAM XML resource IDs

From the OpenStack GUI:

- Change your view to the tenant on which the DSR Network OAM VNF is deployed.
- Go to **Project->Network->Network Topology**. A diagram of all VMs in the tenant is displayed.
Note: The diagram may take a few minutes to display.

- Click on one of the NOAM VMs.
- A pop-up appears having information about the specific NOAM VM.
- Save the resource ID for the XMI port provided in the IP Addresses section of the pop-up.
Note: The IP Addresses section of the popup contains information about the network ports and resource IDs, assigned to the VM.
- Repeat the previous step for the other NOAM VM and DSR Signaling VM.

For more information about the full listing of all inputs and possible outputs of the command "instantiate VNF", see **ETSI NFV-SOL 003**, section **5.4.4.3.1**, or the DSR VNFM Swagger specification.

Sample Request:

Instantiating the first signaling VNF request generated

URL: `https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances/ < VNF ID received from create request > /instantiate`

Accept: `application/json`

Content-Type: `application/json`

Sample request for Dynamic IP deployment model

```
{
    "flavourId": "sdssignaling",
    "instantiationLevelId": "small",
    "extVirtualLinks": "extVirtualLinks",
    "extManagedVirtualLinks": [{
        "id": "",
        "virtualLinkDescId": "
Network used for intra-site communication in Signaling VNF",
        "resourceId": "8a4d1ec6-
367a-4b1a-978d-2c4eae3daec3"
    },
    {
        "id": "",
        "virtualLinkDescId": "
active SDS NOAM XMI",
        "resourceId": "2bed5886-
8c97-4623-8da3-9c500cce71e3"
    },
    {
        "id": "",
        "virtualLinkDescId": "
standby SDS NOAM XMI",
        "resourceId": "8a4d1ec6-
367a-4b1a-978d-2c4eae3daeg3"
    }
  ]
}
```

```

    ],
    "vimConnectionInfo": [ {
      "id": "vimid",
      "vimType": "OpenStack",
      "interfaceInfo": {
        "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
      },
      "accessInfo": {
        "username": "dsrsci.user",
        "password": "xxxx",
        "userDomain": "Default",
        "projectDomain": "default",
        "tenant": "DSR CI"
      }
    }
  ],
  "localizationLanguage": "localizationLanguage",
  "additionalParams": {
    "xmiNetwork": {
      "name": "ext-net8",
      "ipVersion": "IPv4"
    },
    "ntpServerIp": "10.250.32.10",
    "primarySdsNoamVmName": "SDS-NOAM00-32cd6138",
    "sdsNoamSgName": "sdsNetworkOam_NOAM_32cd6138_SG"
  }
}

```

Sample request for Fixed IP deployment model

```

{
  "flavourId": "sdssignaling",
  "instantiationLevelId": "small",
  "extVirtualLinks": "extVirtualLinks",
  "extManagedVirtualLinks": [{
    "id": "",

```

```

SDS NOAM XMI",
8c97-4623-8da3-9c500cce71e3"
    },
    {
        "id": "",
        "virtualLinkDescId":
        "resourceId": "2bed5886-

"standby SDS NOAM XMI",
367a-4b1a-978d-2c4eae3daeg3"
    }
],
    "vimConnectionInfo": [ {
        "id": "vimid",
        "vimType": "OpenStack",
        "interfaceInfo": {
            "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
        },
        "accessInfo": {
            "username": "dsrcli.user",
            "password": "xxxx",
            "domainuserDomain": "Default",
            "projectDomain": "default",
            "tenant": "DSR CI"
        }
    }
}],
    "localizationLanguage": "localizationLanguage",
    "additionalParams": {
        "xmiNetwork": {
            "name": "ext-net8",
            "ipVersion": "IPv4",

"fixedIps": {
    "primaryDpSoamXmiIp": "10.75.192.5",
    "dpSoamXmiIp": "10.75.192.6",
    "dpSoamVip": "10.75.192.7",
    "dpXmiIps": ["10.75.192.8"]

```



```

    }

    },

    "imiNetwork": {

        "name": "imi-private",

        "ipVersion": "IPv4",

        "fixedIps":{
            "primaryDpSoamImiIp": "192.167.2.1",
            "dpSoamImiIp": "192.167.2.3",
            "dpImiIps": ["192.167.2.5"]
        }

    },

    "ntpServerIp": "10.250.32.10",

    "primarySdsNoamVmName": "SDS-NOAM00-
32cd6138",

    "sdsNoamSgName":
"sdsNetworkOam_NOAM_32cd6138_SG"

    }

}

```

Sample Response:

```

202 Accepted
Headers:
{
    location: https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
fb21f9d3-43ad-46cd-a03f-7220bb36a5c6
    date: Tue, 29 Jan 2019 10:39:24 GMT
    content-length: 0    content-type:
    application/xml
}

```

The following table describes the parameters used for sending request to VNFM:

Table 16. Parameters and Definitions for SDS Signaling VNF

Parameter	Definitions
flavourId	Identifier of the VNF deployment flavor to be instantiated
instantiationLevelId	Identifier of the instantiation level of the deployment flavor to be instantiated. If not present, the default instantiation level as declared in the VNFD is instantiated.

Parameter	Definitions
resourceId	The identifier of the resource (imi Network ID of the signaling VNF, active, standby SDS NOAM XMI) in the scope of the VIM or the resource provider
xmiNetwork	Network that is used to provide access to the DSR entities (GUI, ssh), and for inter-site communication
imiNetwork	Network used to provide access to the DSR entities (GUI, ssh), and for internal communication
name	Network name, for example; ext-net
ipVersion	IP version IPv4 or IPv6
ntpServerIp	IP of the NTP server
primarySdsNoamVmName	Name of primary SDS NOAM VM
sdsNoamSgName	The server group of the SDS NOAM VM
primaryDpSoamXmiIp	IP address for primary SDS DP SOAM IP
dpSoamXmiIp	IP address for secondary SDS DP SOAM IP
dpSoamVip	IP address for SDS SOAM VIP
dpXmiIps	IP address for SDS DP IP
primaryDpSoamImiIp	IP address for primary SDS DP SOAM IP of IMI
dpSoamImiIp	IP address for secondary SDS DP SOAM IP of IMI
dpImiIps	IP address for primary SDS DP IP of IMI

7.12 Scale VNF to Level (Only Scale Out)

The N/B LCM scale_to_level Rest I/F helps in scaling existing VNF's.

Following are the available options while scaling using "scale to VNF level" N/B Interface:

- Scale VNF to Level based on pre-defined sizes (using Instantiation level Id).
- Scale VNF to Level with arbitrary sizes (using scaleInfo).

Note:

- This feature is only supported for Scaling out C-level servers of Signaling Stack.
- The stack must have been instantiated prior to performing scale to level operation.
- Before Scaling the VNF to level, `VnfInstance` Id of the stack must be available.
- The instantiation level for Signaling stack is available under **Instantiating the first signaling VNF** section.
- Scale to Level Request accepts either `instantiationLevelId` or `scaleInfo`.
- Cross deployment scaling is not supported by VNFM - if the user instantiated the VNF in fixed IP deployment model, then he must scale to level using FIXED IP deployment model only and vice versa.

The following image illustrates the VNF Scaling:

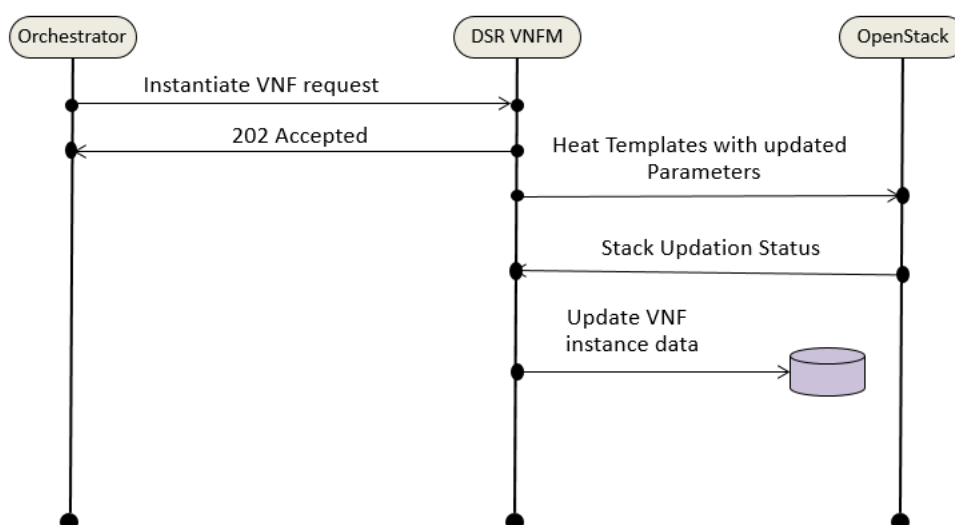


Figure 7. VNF Scaling

7.12.1 Scale VNF to Level using InstantiationLevelId

This option supports Scaling of VNF from a lower instantiation level to higher one, such as Small to Medium.

Sample Request

Scaling VNF to Level Request for Dynamic IP model

Resource URL: `https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances/<VNF ID received from create/instantiate request>/scale_to_level`

Accept: `application/json`

Content-Type: `application/json`

```
{
  "instantiationLevelId": "medium"
}
```

Scaling VNF to Level Request for Fixed IP model

```
{
  "instantiationLevelId": "medium",
  "additionalParams": {
    "xmiNetwork": {
      "fixedIps": {
        "dampXmiIps": ["10.75.218.123", "10.75.218.21"],
        "ipfeXmiIps": ["10.75.218.3", "10.75.218.2"],
      }
    }
  }
}
```

```

        "stpXmiIps": ["10.75.218.42","10.75.218.143"],
        "sbrXmiIps": ["10.75.218.23","10.75.218.19"]
    },
    "sbrNetwork":{
        "fixedIps":{
            "sbrNetworkIps": ["10.75.219.23","10.75.219.123"]
        }
    },
    "xsiNetwork": [{
        "fixedIps":{
            "dampXsiIps": ["10.75.219.23","10.75.219.12"],
            "ipfeXsiIps": ["10.75.219.1","10.75.219.112"],
            "stpXsiIps": ["10.75.219.12","10.75.219.23"]
        }
    }]
}
}

```

Note: The 202 response means that the request was accepted for processing. The VNF might take up to 6 minutes to become fully operational. Use the DSR GUI to determine when the VNF is operational.

Sample Response

```

202 Accepted
Headers:
{
    location: https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
fb21f9d3-43ad-46cd-a03f-7220bb36a5c6
    date: Tue, 29 Jan 2019 10:39:24 GMT
    content-length: 0    content-type:
    application/xml
}

```

Detailed explanation of XMI and XSI Network

Note:

1. The instantiation level must be decided based on the number of VMs required.
2. Only the IPs of the required VM are to be provided in the `fixedIp` parameter and they must be of the same network in that order as used during the instantiation process.

For Example:

"flavorId": "DIAMETER+SS7", "instantiationLevelId": "medium" (scaling from small to medium) - This brings up 2 new DAMPs (DAMP02, DAMP03) , 2 new STP (STP 02, STP 03) servers.

The user needs to provide dampXmiIps(2), stpXmiIps(2), dampXsiIps(2), stpXsiIps(2)

The detailed explanation of XMI and XSI Network for the additional parameters is provided below:

For XMI Network

```
"xmiNetwork":{
  "fixedIps":{
    "dampXmiIps":[
      "<DAMP 02 XMI IP>",
      "<DAMP 03 XMI IP>"
    ],
    "stpXmiIps":[
      "<STP 02 XMI IP>",
      "<STP 03 XMI IP>"
    ]
  }
}
```

For XSI Network

```
"xsiNetwork":[
  {
    "fixedIps":{
      "dampXsiIps":[
        "<DAMP02 XSI 1 IP>",
        "<DAMP03 XSI 1 IP>"
      ],
      "stpXsiIps":[
        "<STP02 XSI 1 IP>",
        "<STP03 XSI 1 IP>"
      ]
    }
  },
  {
    "fixedIps":{
      "dampXsiIps":[
```

```

        "<DAMP02 XSI 2 IP>",
        "<DAMP03 XSI 2 IP>"
    ],
    "stpXsiIps": [
        "<STP02 XSI 2 IP>",
        "<STP03 XSI 2 IP>"
    ]
}
}
]

```

Table 17 describes the parameters used for sending request to VNFM.

Table 17. Parameters and Definitions for Scaling VNF to Level using InstantiationLevelId

Parameter	Definitions
instantiationLevelId	Identifier of the instantiation level of the deployment flavor to be instantiated.
dampXmiIps	List of DAMP external management ips (if new DAMP VMs are to be scaled)
ipfeXmiIps	List of IPFE external management ips (if new IPFE VMs are to be scaled)
stpXmiIps	List of vSTP external management ips (if new vSTP VMs are to be scaled)
sbrXmiIps	List of SBR external management ips (if new SBR VMs are to be scaled)
sbrNetworkIps	List of SBR replication port ips (if new SBR VMs are to be scaled)
dampXsiIps	List of DAMP signaling ips (if new DAMP VMs are to be scaled)
ipfeXsiIps	List of IPFE signaling ips (if new DAMP VMs are to be scaled)
stpXsiIps	List of STP signaling ips (if new DAMP VMs are to be scaled)

Note: During Scaling of SBR's, the newly spawned SBR's are not added to any Server Group, it need to be manually added to the new Server Groups created by the user. One server Group can have maximum two SBR's.

7.12.2 Scale VNF to Level using ScaleInfo (Arbitrary Size)

This option supports Scaling of VNF to arbitrary sizes based on `ScaleInfo`.

Scale VNF to Level using arbitrary size means increasing existing VNFC count within the max allowed VNFC count.

Max allowed VNFC count is the count from existing VNF's flavourId with Large InstantiationLevelId.

Note: Max allowed VNFC count can be referred from Instantiating the first signaling VNF section.

Sample Request:

Scaling VNF to Level Request

Request URL: `https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances/< VNF ID received from create/instantiate request>/scale_to_level`

Accept: `application/json`

Content-Type: `application/json`

```
{
  "scaleInfo": [
    {
      "aspectId": "DAMP",
      "scaleLevel": "3"
    }
  ],
}
```

Note: The 202 response means that the request was accepted for processing. The VNF might take up to 6 minutes to become fully operational. Use the DSR GUI to determine when the VNF is operational.

Sample Response

```
202 Accepted
Headers:
{
  location: https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
fb21f9d3-43ad-46cd-a03f-7220bb36a5c6
  date: Tue, 29 Jan 2019 10:39:24 GMT
  content-length: 0   content-type:
  application/xml
}
```

Important Note:

3. The aspect Id is decided based on the VM to be scaled, scale level is decided based on the number of VMs required.
4. Only the IPs of the required VM must be provided in the `fixedIp` parameter and they must be of the same network in that order as used during the instantiation process.

For Example:

```
"aspectId":"DAMP","scaleLevel":"4" (from scaleLevel 2 to scaleLevel 4) (
scaling from small to medium) - This brings up 2 new DAMPs (DAMP02, DAMP03) servers.
```

The user needs to provide `dampXmiIps(2)`, `dampXsiIps(2)`

Detailed explanation of XMI, IMI and XSI Network

The detailed explanation of XMI and XSI Network for the additional parameters is provided below:

For XMI Network

```

"xmiNetwork":{
  "fixedIps":{
    "dampXmiIps":[
      "<DAMP 02 XMI IP>",
      "<DAMP 03 XMI IP>"
    ]
  }
}

```

For IMI Network

```

"imiNetwork":{
  "fixedIps":{
    "dampImiIps":[
      "<DAMP 02 IMI IP>",
      "<DAMP 03 IMI IP>"
    ]
  }
}

```

For XSI Network

```

"xsiNetwork":[
  {
    "fixedIps":{
      "dampXsiIps":[
        "<DAMP02 XSI 1 IP>",
        "<DAMP03 XSI 1 IP>"
      ]
    }
  },
  {
    "fixedIps":{
      "dampXsiIps":[
        "<DAMP02 XSI 2 IP>",
        "<DAMP03 XSI 2 IP>"
      ]
    }
  }
]

```



```

    }
  ]

```

Table 18 describes the parameters used for sending request to VNFM.

Table 18. Parameters and Definitions for Scaling VNF to Level using ScaleInfo

Parameter	Definitions
scaleInfo	aspectId : VnfType scaleLevel : Target scale level to which the VNF is to be scaled
dampXmiIps	List of DAMP external management ips (if new DAMP VMs are to be scaled)
ipfeXmiIps	List of IPFE external management ips (if new IPFE VMs are to be scaled)
stpXmiIps	List of vSTP external management ips (if new vSTP VMs are to be scaled)
sbrXmiIps	List of SBR external management ips (if new SBR VMs are to be scaled)
sbrNetworkIps	List of SBR replication port ips (if new SBR VMs are to be scaled)
dampXsiIps	List of DAMP signaling ips (if new DAMP VMs are to be scaled)
ipfeXsiIps	List of IPFE signaling ips (if new DAMP VMs are to be scaled)
stpXsiIps	List of STP signaling ips (if new DAMP VMs are to be scaled)
dampImiIps	List of DAMP internal management ips (if new DAMP VMs are to be scaled)
ipfeImiIps	List of IPFE internal management ips (if new IPFE VMs are to be scaled)
stpImiIps	List of vSTP internal management ips (if new vSTP VMs are to be scaled)
sbrImiIps	List of SBR internal management ips (if new SBR VMs are to be scaled)

Note: During Scaling of SBR's, the newly spawned SBR's are not added to any Server Group, it needs to be manually added to the new Server Groups created by the user. One server Group can have maximum two SBR's.

8. Discover Stack

1. It is an LCM Discover Rest I/F.
2. It is used to discover the created stack in OpenStack and save the stack information (parameter file and VNF instance) in the VNFM persistent directory. This information can be used for further requests by the orchestrator. For example, to scale out the stack.
3. Before discovering the stack, make sure the following information is available:
 - The Stack ID of the previously created stack.
 - The following information about the OpenStack instance on which the Stack must be discovered:
 - OpenStack Controller URI
 - Use Domain Name
 - Project Domain Id
 - Username
 - Password
 - Tenant name
4. The Interface discovers the stack and performs the following operations:
 - a. Download the parameter file of the discovered stack.
 - b. Create the Instance file of the discovered stack.
 - c. These two files are saved in `/var/vnfm/instances/<autoDiscovery InstanceId>/` directory.

Sample Request for Discover Interface

```
Request URL: POST:
https://<<VNFM HOST IP>>:8443/vnflcm/v1/discover/<<discover stack id>>

For example:
https://localhost:8443/vnflcm/v1/discover/b30ac203-5fe1-4007-a3ba-078f3422708b

Accept: application/json
Content-Type: application/json
Request Body:
{
  "vimConnectionInfo": [
    {
      "id": "vimid",
      "vimType": "OpenStack",
      "interfaceInfo": {
        "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
      },
      "accessInfo": {
        "username": "dsrat.user",
```

```
        "password": "xxxx",
        "userDomain": "Default",
        "projectDomain": "default",
        "tenant": "DSR AT Dev 1"
    }
}
]
```

Sample Response for Discover Interface

Response Code: 200

```
{
  "vnfInstanceId": "dsrNetworkOam-945cffa107c235bb-43d87678-756b-4f8e-a59c-
d9b7d4dd95a1", "discoverStackId": "7d861391-0ed2-4d0b-9f01-e84e186e9244"
}
```

Note:

- Discover VNF stack supports only those stacks that are created by the VNFM templates.
- Discover VNF stack also supports for stack created by VNFM GUI, Double Failure of Active VNFM and its persistent volume.

9. Query LCM Operation

This resource represents VNF lifecycle management operation occurrences. This resource can be used to query status information about multiple VNF lifecycle management operation occurrences.

The following image illustrates the sequence for querying/reading information about a VNF LCM Operation.

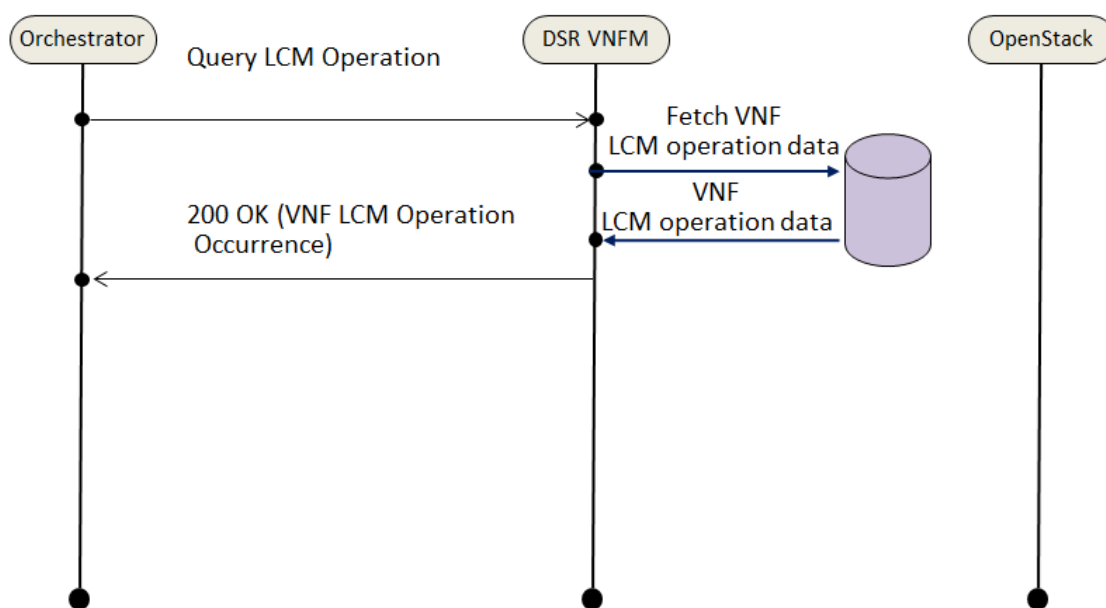


Figure 8. VNF LCM Operation

Query LCM Operation, using the following two ways:

- 1) Query individual LCM Operation
- 2) Query All LCM Operation

9.1 Query Individual LCM Operation

If the NFVO intends to read information about a particular LCM Operation, it sends a GET request to the "Individual LCM operation" resource, addressed by the appropriate VNF LCM Operation occurrence identifier (`vnfLcmOpOccId`) in its resource URI.

The VNFM returns a **200 OK** response to the NFVO, and includes specific data structure of type "VnfLcmOpOcc" related to the VNF LCM Operation occurrence identifier (`vnfLcmOpOccId`) in the payload body.

Sample Request

Query individual LCM Operation

URL: GET: `https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_lcm_op_occs/<<{vnfLcmOpOccId}>>`

Sample Response

```

URL: GET: https://<<VNFM HOST
IP>>:8443/vnfm/v1/vnf_lcm_op_occs/<<{vnfLcmOpOccId}>>
Accept: application/json
Content-Type: application/json
{
  "id": "lcmOp-ec72c7b4-7cea-4201-a0ab-5c0cec66cfa6",
  "operationState": "STARTING",
  "stateEnteredTime": "2019/01/16 05:53:31 UTC",
  "startTime": "2019/01/16 05:53:31 UTC",
  "vnfInstanceId": "dsrNetworkOam-dfc4dcd2-2752-48b4-875d-6cf703ba4134",
  "operation": "INSTANTIATE",
  "operationParams": {
    "flavourId": "DSR NOAM",
    "instantiationLevelId": "small1",
    "extVirtualLinks": "extVirtualLinks",
    "extManagedVirtualLinks": [],
    "vimConnectionInfo": [
      {
        "id": "vimid",
        "vimType": "OpenStack",
        "interfaceInfo": {
          "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
        }
      },
      {
        "id": "vimid",
        "vimType": "OpenStack",
        "interfaceInfo": {
          "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
        }
      }
    ],
    "accessInfo": {
      "username": "dsrat.user",
      "password": "automation",
      "userDomain": "Default",
      "projectDomain": "default",
      "tenant": "DSR AT Dev 2"
    },
    "extra": null
  },
  "localizationLanguage": "localizationLanguage",
  "additionalParams": {
    "ntpServerIp": "10.250.32.10",
    "xmiNetwork": {

```

```

        "name": "ext-net7",
        "ipVersion": "IPv4"
      }
    },
    "links": {
      "self": {
        "href": "https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-ec72c7b4-7cea-4201-a0ab-5c0cec66cfa6"
      },
      "vnfInstance": {
        "href": "https://localhost:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-dfc4dcd2-2752-48b4-875d-6cf703ba4134"
      }
    },
    "isAutomaticInvocation": false,
    "isCancelPending": false
  }
}

```

9.2 Query All LCM Operation

If the NFVO intends to query all LCM Operation, it sends a GET request to the **LCM operation** resource. The VNFM returns a **200 OK** response to the NFVO, and includes zero or more data structures of type "VnfLcmOpOcc" in the payload body.

Sample Request

Query All LCM Operation

URL: GET: https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_lcm_op_occs

Sample Response

```

URL: GET: https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_lcm_op_occs
Response Body for No VNF Instances
[]

Response Body for Query All LCM Operation
[
{
  "id": "lcmOp-ec72c7b4-7cea-4201-a0ab-5c0cec66cfa6",
  "operationState": "STARTING",

```

```

"stateEnteredTime": "2019/01/16 05:53:31 UTC",
"startTime": "2019/01/16 05:53:31 UTC",
"vnfInstanceId": "dsrNetworkOam-dfc4dcd2-2752-48b4-875d-6cf703ba4134",
"operation": "INSTANTIATE",
"operationParams": {
  "flavourId": "DSR NOAM",
  "instantiationLevelId": "small11",
  "extVirtualLinks": "extVirtualLinks",
  "extManagedVirtualLinks": [],
  "vimConnectionInfo": [
    {
      "id": "vimid",
      "vimType": "OpenStack",
      "interfaceInfo": {
        "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
      }
    },
    {
      "username": "dsrat.user",
      "password": "xxxxxx",
      "userDomain": "Default",
      "projectDomain": "default",
      "tenant": "DSR AT Dev 2"
    }
  ],
  "extra": null
},
"localizationLanguage": "localizationLanguage",
"additionalParams": {
  "ntpServerIp": "10.250.32.10",
  "xmiNetwork": {
    "name": "ext-net7",
    "ipVersion": "IPv4"
  }
},
"links": {
  "self": {

```

```

    "href": "https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-ec72c7b4-7cea-4201-a0ab-5c0cec66cfa6"
  },
  "vnfInstance": {
    "href":
"https://localhost:8443/vnflcm/v1/vnf_instances/dsrNetworkOam-dfc4dcd2-2752-48b4-875d-6cf703ba4134"
  }
},
"isAutomaticInvocation": false,
"isCancelPending": false
},
{
  "id": "lcmOp-00574fa7-8c4a-45ac-b7a8-816bfaf70985",
  "operationState": "STARTING",
  "stateEnteredTime": "2019/01/16 06:05:32 UTC",
  "startTime": "2019/01/16 06:05:32 UTC",
  "vnfInstanceId": "dsrSignaling-08db63da-6cac-495f-8480-baf368d21cf7",
  "operation": "INSTANTIATE",
  "operationParams": {
    "flavourId": "DIAMETER",
    "instantiationLevelId": "small",
    "extVirtualLinks": "extVirtualLinks",
    "extManagedVirtualLinks": [
      {
        "id": "id1",
        "resourceId": "31ae9c8b-519e-4316-9a24-45c619646d69"
      },
      {
        "id": "id2",
        "resourceId": "aa9d142d-89d4-40e7-a701-559a993aa5ea"
      }
    ],
    "vimConnectionInfo": [
      {
        "id": "vimid",
        "vimType": "OpenStack",
        "interfaceInfo": {

```



```

    "controllerUri": "https://oortcloud.us.oracle.com:5000/v3"
  },
  "accessInfo": {
    "username": "dsrat.user",
    "password": "xxxxxx",
    "userDomain": "Default",
    "projectDomain": "default",
    "tenant": "DSR AT Dev 2"
  },
  "extra": null
}
],
"localizationLanguage": "localizationLanguage",
"additionalParams": {
  "xmiNetwork": {
    "name": "ext-net7",
    "ipVersion": "IPv4"
  },
  "xsiNetwork": {
    "name": "ext-net7",
    "ipVersion": "IPv4"
  },
  "ntpServerIp": "10.250.32.10",
  "primaryNoamVmName": "NOAM00-03ba4134",
  "noamSgName": "dsrNetworkOam_NOAM_03ba4134_SG"
},
"links": {
  "self": {
    "href": "https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-00574fa7-8c4a-45ac-b7a8-816bfaf70985"
  },
  "vnfInstance": {
    "href": "https://localhost:8443/vnflcm/v1/vnf_instances/dsrSignaling-08db63da-6cac-495f-8480-baf368d21cf7"
  }
},
"isAutomaticInvocation": false,

```

```
    "isCancelPending": false  
  }  
]
```

10. Terminating a VNF

This procedure represents the **Terminate VNF** operation. The client can use this procedure to terminate a VNF instance. The POST method terminates a VNF instance.

Following are the two types of request parameters for the **Terminate VNF** operation:

- **FORCEFUL**: The VNFM deletes the VNF and releases the resources immediately after accepting the request.
- **GRACEFUL**: After accepting the request, the VNFM first validates if the VNF configuration is cleaned up. Once the validation is successful, VNFM deletes the VNF and releases the resources.

Table 19 describes the parameters used for sending request to VNFM.

Table 19. Parameters and Definitions for Terminating VNF

Parameter	Definitions
terminationType	Indicates whether forceful or graceful termination is requested.

10.1 Forceful Termination

The VNFM will delete the VNF immediately after accepting the request. The instance file is updated with VNF Operational State set to **STOPPED**.

Note: If the VNF is still in service, requesting forceful termination can adversely impact the network service.

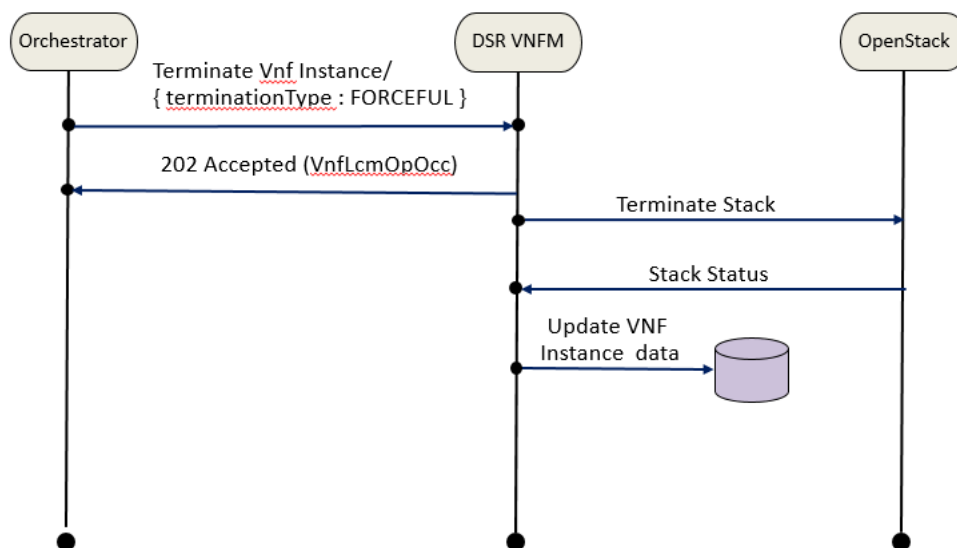


Figure 9. Forceful Termination

Terminating DSR and SDS VNF Instance Forcefully

Sample Request:

Request URL: POST: `https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances/< VNF ID received from create request>/terminate`

Accept: application/json

Content-Type: application/json

```
{
  "terminationType": "FORCEFUL"
}
```

Sample Response:

```
Response Code: 202
{
  location: https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
fb21f9d3-43ad-46cd-a03f-7220bb36a5c6
  date: Tue, 29 Jan 2019 10:39:24 GMT
  content-length: 0   content-type:
  application/xml
}
```

10.2 Graceful Termination

The VNFM first validates if the VNF configuration is cleaned up after accepting the request. Once that configuration is cleaned, the VNFM deletes the VNF. Then the instance file is updated with VNF Operational State set to **STOPPED**.

If AppWorks configurations are not cleaned manually and the orchestrator tries to do graceful termination for that VNF, then the termination of VNF fails.

Note: User must manually cleanup the AppWorks configurations before doing Graceful Termination.

Steps for cleaning up the AppWorks Configuration for Signaling Stack of DSR and SDS:

1. Open corresponding Active NOAM GUI of the Signaling instance.
2. In **Status & Manage** Tab, under **HA**, edit the **Max Allowed HA Role** of instances of the Signaling stack as **OOS**.
3. In Configuration Tab, under Server Groups, edit the corresponding server groups of the instances and uncheck **SG Inclusion** for the Server, and press **OK**. After this step, the excluded Servers must disappear in **Status & Manage -> Server** section.
4. Finally, go to **Configuration -> Servers** section, select the servers that needs to be deleted and click **Delete**.

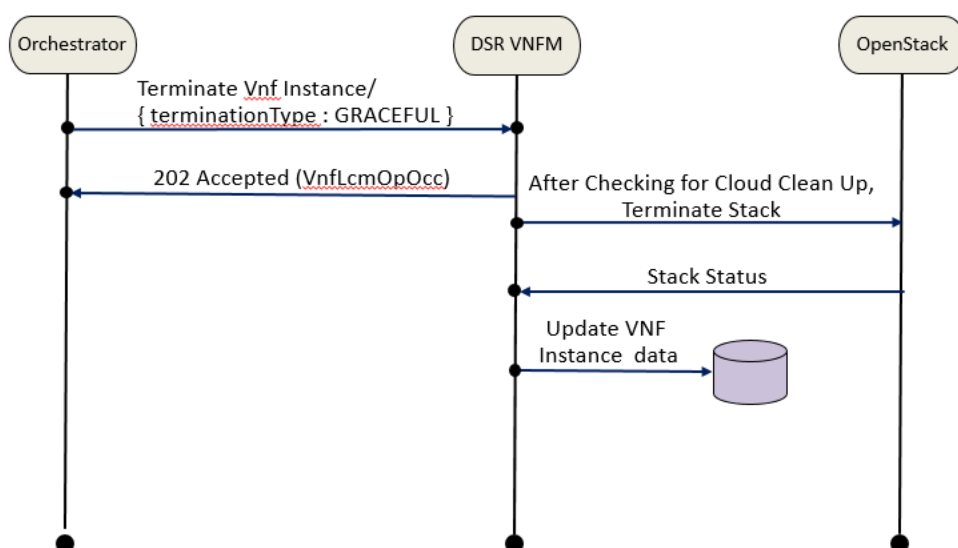


Figure 10. Graceful Termination

Terminating DSR and SDS VNF Instance Gracefully

Sample Request:

Request URL: POST: `https://<<VNFM HOST IP>>:8443/vnfm/v1/vnf_instances/< VNF ID received from create request>/terminate`

Accept: `application/json`

Content-Type: `application/json`

```
{
  "terminationType": "GRACEFUL"
}
```

Sample Response:

```
Response Code : 202
{
  location: https://localhost:8443/vnflcm/v1/vnf_lcm_op_occs/lcmOp-
fb21f9d3-43ad-46cd-a03f-7220bb36a5c6
  date: Tue, 29 Jan 2019 10:39:24 GMT
  content-length: 0  content-type:
  application/xml
}
```

11. Changing the Default Configurations

This section includes information about changing the default configurations through the following files:

- VmInfo.xml
- VnfmProperties.xml

11.1 Changing Flavor Names

To change the flavor names:

1. Log into the VNFM VM.
2. Go to /opt/vnfm/config/8.4/ folder.
3. Edit the file VmInfo.xml
4. Find the tag <flavor> against the VM type (NOAM, SOAM, and so on)
5. Change the default name to user defined name.

Note: The user defined flavor name should be a valid flavor.

11.2 Changing Image Names

1. Log into the VNFM VM
2. Change to /opt/vnfm/config/8.4/ folder
3. Edit the VmInfo.xml
4. Find the tag <image> against the VM type (NOAM, SOAM, and so on)
5. Change the default name to user defined name.

Note: The user defined image name should be a valid image.

The sample VmInfo.xml is provided below:

```
<?xml version="1.0" encoding="UTF-8"?>

<!--
*****
Oracle Corporation, Inc.
Copyright (C) 2016, Oracle and/or its affiliates. All rights reserved
*****
-->

<!--
VM Info file for DSR Release 8.3
This file contains informations related to open stack flavors, Dsr images..
etc of respective node types (NOAM,SOAM,DAMP,SS7,STP,IPFE..) .
-->

<dsratvminfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
<vmdetails>
  <vminfo>
    <name>NOAM</name>
    <flavor>dsr.noam</flavor>
    <image>DSR-8.4.0.1.0_84.23.0.vmdk</image>
  </vminfo>
  <vminfo>
    <name>SOAM</name>
    <flavor>dsr.soam</flavor>
    <image>DSR-8.4.0.1.0_84.23.0.vmdk</image>
  </vminfo>
  <vminfo>
    <name>DA-MP</name>
    <flavor>dsr.da</flavor>
    <image>DSR-8.4.0.1.0_84.23.0.vmdk</image>
  </vminfo>
  <vminfo>
    <name>IPFE</name>
    <flavor>dsr.ipfe</flavor>
    <image>DSR-8.4.0.1.0_84.23.0.vmdk</image>
  </vminfo>
  <vminfo>
    <name>SBR</name>
    <flavor>dsr.sbr</flavor>
    <image>DSR-8.4.0.1.0_84.23.0.vmdk</image>
  </vminfo>
  <vminfo>
    <name>SS7-MP</name>
    <flavor>dsr.ss7</flavor>
    <image>DSR-8.4.0.1.0_84.23.0.vmdk</image>
  </vminfo>
  <vminfo>
    <name>STP-MP</name>
    <flavor>dsr.vstp</flavor>
    <image>DSR-8.4.0.1.0_84.23.0.vmdk</image>
  </vminfo>
  <vminfo>
```

```

        <name>DR-NOAM</name>
        <flavor>dr.noam</flavor>
        <image>DSR-8.4.0.1.0_84.23.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>DSR-APIGWADMIN</name>
        <flavor>dsrapigw.admin</flavor>
        <image>DSRAPIGW-8.4.0.0.0_84.16.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>DSR-APIGWAPP</name>
        <flavor>dsrapigw.app</flavor>
        <image>DSRAPIGW-8.4.0.0.0_84.16.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>UDR</name>
        <flavor>udr.noam</flavor>
        <image>UDR-12.5.1.0.0_17.8.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>DSR-DBSERVER</name>
        <flavor>dsr.noam</flavor>
        <image>DSR-8.4.0.0.0_84.15.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>DSR-IDIHAPP</name>
        <flavor>appl-idih</flavor>
        <image>apps-8.2.1.0.0_82.23.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>DSR-IDIHMEDIATION</name>
        <flavor>med-idih</flavor>
        <image>mediation-8.2.1.0.0_82.23.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>DSR-IDIHDB</name>
        <flavor>db-idih</flavor>

```



```

        <image>oracle-8.2.1.0.0_82.23.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>SDS-NOAM</name>
        <flavor>sds.noam</flavor>
        <image>SDS-8.4.0.1.0_84.23.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>SDS-QS</name>
        <flavor>sds.noam</flavor>
        <image>SDS-8.4.0.1.0_84.23.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>SDS-DR-NOAM</name>
        <flavor>sds.noam</flavor>
        <image>SDS-8.4.0.1.0_84.23.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>SDS-DR-QS</name>
        <flavor>sds.noam</flavor>
        <image>SDS-8.4.0.1.0_84.23.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>SDS-SOAM</name>
        <flavor>sds.dpsoam</flavor>
        <image>SDS-8.4.0.1.0_84.23.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>SDS-DP</name>
        <flavor>sds.dp</flavor>
        <image>SDS-8.4.0.1.0_84.23.0.vmdk</image>
    </vminfo>
    <vminfo>
        <name>DSR-DR-NOAM</name>
        <flavor>dsr.noam</flavor>
        <image>DSR-8.4.0.1.0_84.23.0.vmdk</image>
    </vminfo>

```

```
</vmdetails>
</dsratvminfo>
```

11.3 Changing Availability Zone

1. Log into the VNFM VM.
2. Change to /opt/vnfm/config/ folder
3. Edit the VnfmProperties.xml
4. Find the tag <osAvailabilityZone>
5. Change the default name to user defined name.

Note: The user defined flavor name should be the availability zone.

11.4 Changing Profile Name

1. Log into the VNFM VM
2. Change to /opt/vnfm/config/ folder
3. Edit the VnfmProperties.xml
4. Find the tag <profileName>
5. Change the default name to user defined name.

Note: The user defined image name should be a valid profile name.

The sample VnfmProperties.xml is provided below:

```
<?xml version="1.0" encoding="UTF-8"?>

<!--
*****
Oracle Corporation, Inc.
Copyright (C) 2016, Oracle and/or its affiliates. All rights reserved
*****
-->

<vnfmOpenstackProperties xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
<!-- DSR RELEASE INFORMATION : START-->
<!-- ===== -->
<!-- Information about DSR VNFM release -->
<dsrVnfmVersion>4.0</dsrVnfmVersion>
<!-- Information about DSR supported releases
Include within <dsrVersion></dsrVersion> tags to add new release
-->
```

```

<dsrSupportedReleases>
<dsrRelease>
<releaseNumber>8.4</releaseNumber>
</dsrRelease>
</dsrSupportedReleases>

<!-- Information about DSR supported pre-releases
APPLICABLE ONLY FOR INTERNAL RELEASES
Include within <dsrVersion></dsrVersion> tags to add new release
-->
<dsrSupportedPreReleases>
<dsrRelease>
<releaseNumber>8.3</releaseNumber>
</dsrRelease>
</dsrSupportedPreReleases>
<!-- ===== -->
<!-- DSR RELEASE INFORMATION : END -->

<!-- VNFM Persistent Storage path-->
<vnfmPersistentInstancesDir>/var/vnfm/instances/</vnfmPersistentInstancesDir>
>
<vnfmPersistentLcmOperationsDir>/var/vnfm/lcmoperations/</vnfmPersistentLcmOperationsDir>
>
<!--paths to various files -->
<userOpenstackDir>/var/NSA/LDM/config/openstack</userOpenstackDir>
<heatParameterDir>/var/NSA/LDM/config/openstack/parameter</heatParameterDir>
<userInputVnfmDir>/var/NSA/LDM/config/planning</userInputVnfmDir>
<vmInfoDir>/var/NSA/LDM/config/openstack</vmInfoDir>
<vnfmLogDir>/var/vnfm/logs/</vnfmLogDir>

<!--OpenStack related timers -->
<stackCheckInterval>5000</stackCheckInterval>
<stackCreateCheckCount>60</stackCreateCheckCount>
<stackUpdateCheckCount>60</stackUpdateCheckCount>
<stackDeleteCheckCount>30</stackDeleteCheckCount>
<stackRetrieveDataCount>3</stackRetrieveDataCount>
<retrieveStackRetryCount>25</retrieveStackRetryCount>
<retrieveDeleteStackRetryCount>1</retrieveDeleteStackRetryCount>

```

```

<!--OpenStack network information -->
<dsrImiIpv4CidrSubnet>192.167.1.0/24</dsrImiIpv4CidrSubnet>
<dsrImiIpv6CidrSubnet>2001:db8:1234:0000::/64</dsrImiIpv6CidrSubnet>

<!-- HTTP Request Validator Path -->
<nbrestValidatorsDir>/var/NSA/LDM/validators</nbrestValidatorsDir>

<!-- DSR MMI related parameters -->
<mmiRetryCount>30</mmiRetryCount>
<mmiInterval>60000</mmiInterval>

<!-- DSR access for validation -->
<dsrNoamUsername>guiadmin</dsrNoamUsername>
<dsrNoamPassword>tekware</dsrNoamPassword>

<!-- Total Number of servers per signaling VNF -->
<totalServersPerSignalingVnf>48</totalServersPerSignalingVnf>
<totalIpfeServersPerSignalingVnf>4</totalIpfeServersPerSignalingVnf>

<!-- Topo version for MMI Client -->
<dsrMmiVersion>v2.0</dsrMmiVersion>
<sdsMmiVersion>v1.0</sdsMmiVersion>

<!-- Thread pool executor -->
<corePoolSize>20</corePoolSize>
<maximumPoolSize>30</maximumPoolSize>
<blockingQueueSize>30</blockingQueueSize>
<keepAliveTime>10</keepAliveTime>

<!-- Openstack https client certificate path -->
<osClientCertificatePath>/var/vnfm/certificate/os-client-certificate-
keystore.pem</osClientCertificatePath>

<!-- DSR Profile/Hardware name -->
<profileName>TVOE Guest</profileName>

```

```
<!-- Openstack Availability Zone -->  
<osAvailabilityZone>nova</osAvailabilityZone>  
</vnfmOpenstackProperties>
```

12. Openstack Client HTTP/HTTPS Support

Vnfm support both openstack vim HTTP & HTTPS client.

To support the openstack HTTPS client, user must add the openstack certificate in the below path in a vnfm deployed system:

```
/var/vnfm/certificate/<certificate name>.pem
```

For example: /var/vnfm/certificate/os-client-certificate-keystore.pem

Note: Certificate needs to be in pem format only.

To get the Openstack client certificate, execute:

```
echo -n | openssl s_client -connect <openstack stack ip>:5000 | \
sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' > os-client-certificate-
keystore.pem
```

For example:

```
echo -n | openssl s_client -connect 10.10.20.137:5000 | \
sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' > os-client-certificate-
keystore.pem
```

Above command will fetch the https certificate from openstack and save it in os-client-certificate-keystore.pem file.

To give Openstack HTTPS call through VNFM:

In the request body of any instantiate vnf through VNFM, change VimConnection controllerUri from http to https.

For example:

```
"vimConnectionInfo": [
  {
    "id": "vimid",
    "vimType": "OpenStack",
    "interfaceInfo": {
      "controllerUri": "https://10.10.20.137:5000/v3"
    },
    "accessInfo": {
      "username": "dsrat.user",
      "password": "xxxx",
      "userDomain": "Default",
      "projectDomain": "default",
      "tenant": "tenant name"
    },
    "extra": null
  }
]
```

Note: The `controlleruri` https supports only if the openstack supports https client api.

13. Import HTTPS/SSL Certificate into VNFM

Note: Diameter must be configured for running the traffic.

13.1 Recombine Existing PEM Keys and Certificates into VNFM

If you have an existing private key and certificates for your server's domain in PEM format, combine them into a PKCS keystore, then convert the PKCS keystore into a Java keystore.

Execute the following command:

```
cat <midfile.1.cert.pem> <midfile.2.cert.pem> > intermediates.cert.pem
```

Where <midfile.1.cert.pem> and <midfile.2.cert.pem> are the names of intermediate certificate files.

Note: If you have multiple intermediate certificates, combine them in any order.

1. `openssl pkcs12 -export -in <dsrVnfm.pem> -inkey <dsrVnfm.key> -certfile <intermediate.cert.pem> -passin pass:<existingpassword> -passout pass: xxxx -out vnfm_default.p12 -name "<yourDomainName>"`

For example:

```
openssl pkcs12 -export -in dsrVnfm.pem -inkey dsrVnfm.key -passin pass: xxxx -passout pass:xxxx -out vnfm_default.p12 -name dsrvnfm
```

2. `keytool -importkeystore -srckeystore vnfm_default.p12 -srcstorepass xxxx -srcstoretype PKCS12 -destkeystore vnfm_default.jks -deststorepass xxxx -alias dsrVnfm`

For example:

```
keytool -importkeystore -srckeystore vnfm_default.p12 -srcstorepass xxxx -srcstoretype PKCS12 -destkeystore vnfm_default.jks -deststorepass xxxx -alias dsrVnfm
```

Note: keytool is the java key and certificate management utility provided by Java. It exist in `jre/bin/keytool`.

Where,

- <dsrVnfm.pem>: The existing signed certificate file that matches your existing private key.
- <dsrVnfm.key>: The existing private key file.
- <intermediate.cert.pem>: The existing intermediate certificates that complete the chain from your certificate to a root CA.
- <yourDomainName>: The complete domain name of your server.
- <existingpassword>: The password that allows access to the existing key file.
- <yourpassword>: The password that allows access to your new keystore. Provide at least six characters.

Notes:

destkeystore file name should be same as mention in the command (vnfm_default.jks).

srcstorepass is the password that is given in first command (-passout pass: xxxx) and it should also be same as mention in the command (xxxx)

`deststorepass` is the password that is used to open the certificate file (`vnfm_default.jks`) and should also be same as mention in the command (`xxxxx`), because the same file name and password is used in Tomcat Apache to access the SSL certificate.

13.2 Copy Created Certificate (vnfm_default.jks) into VNFM

Once `vnfm` box is installed, a self-signed certificate is created by VNFM and is placed in the `/var/vnfm/certificate/vnfm_default.jks` directory by default. This certificate is valid for 365 days.

The client must copy their created certificate with same name as `vnfm_default.jks` into `/var/vnfm/certificate/` directory and override the existing `vnfm_default.jks` certificate.

Note: After the making the certificate changes, client must restart the apache tomcat server to reflect the updated certificate in VNFM.

Run the following command to restart the apache tomcat server:

1. `sudo /usr/share/vnfm/apache-tomcat-9.0.6/bin/shutdown.sh`
2. `sudo /usr/share/vnfm/apache-tomcat-9.0.6/bin/startup.sh`

13.3 VNFM Self Signed Certificate Generation

1. Create a `vnfmCert.conf` configuration file as shown in the example below (provide your own details in the respective fields):

```
[ req ]
default_bits = 2048
default_md = sha256
distinguished_name = req_distinguished_name
req_extensions = req_ext
[ req_distinguished_name ]
countryName = Country Name (2-letter code)
stateOrProvinceName = State or Province Name (full name)
localityName = Locality (e.g. city name)
organizationName = Organization (e.g. company name)
commonName = Common Name (your.domain.com)
[ req_ext ]
subjectAltName = @alt_names
[alt_names]
DNS.1 = *.localhost
DNS.2 = 127.0.0.1
DNS.3 = *.oracle.com
DNS.4 = *.oraclecorp.com
```

2. Generate a key pair and a signing request by executing:

```
openssl req -new -keyout dsrVnfm.key -out dsrVnfm.csr -newkey rsa:2048 -
config vnfmCert.conf
```

It will request for password to create private key file.

Note: To skip passphrase in private key, add `-nodes (read: "No DES encryption")` parameter from the command.

Check if CSR contains the SAN by executing:

```
openssl req -noout -text -in sslcert.csr | grep DNS
```

3. Generating a self-signed certificate:

To generate a temporary certificate, which is acceptable for 365 days, execute:

```
openssl x509 -req -days 365 -in dsrVnfm.csr -signkey dsrVnfm.key -sha256 -out dsrVnfm.crt -extfile ca.cnf -extensions req_ext
```

Enter pass phrase for dsrVnfm.key: <type pass phrase of private key>

Check if CSR contains the SAN by executing:

```
openssl req -noout -text -in sslcert.csr | grep DNS
```

4. Convert the CRT to PEM format:

Use the `openssl` tool to convert the CRT to a PEM format that is readable by the reporter:

```
openssl x509 -in dsrVnfm.crt -out dsrVnfm.pem -outform PEM
```

5. To convert the PEM-format keys to Java KeyStores:

```
openssl pkcs12 -export -in dsrVnfm.pem -inkey dsrVnfm.key -passin pass:4srVN6M -passout pass:4srVN6M -out vnfm_default.p12 -name dsrvnfm
```

6. Convert the vnfm_default.p12 to a Java keystore vnfm_default.jks, by executing:

```
keytool -importkeystore -srckeystore vnfm_default.p12 -srcstorepass 4srVN6M -srcstoretype PKCS12 -destkeystore vnfm_default.jks -deststorepass 4srVN6M -alias dsrVnfm
```

Note: After importing certificate into java keystore, it is a good practice to check if the certificate information is correct or not. Keytool is the java jdk tool, which exists in `jdk/bin`.

```
keytool -list -v -keystore [enter keystore name] -storepass [enter keystore password]
```

To delete existing alias from the keystore file, execute (optional):

```
keytool -delete -alias <aliasname> -keystore vnfm_default.jks
```

Note: The `vnfm_default.jks` is the ssl certification file which is being used in VNFM https to establish the ssl connection.

While importing certificate into java keystore, provide `-alias dsrVnfm`. If it prompts to override, type YES.

Use the password "xxxx".

Note: Certificate file name (`vnfm_default.jks`) and alias name (`dsrVnfm`) must be the same as mentioned above.

14. DSR NOAM IPv6 Migration

Prerequisite: The xmi & imi network should have two subnet network each, where 1st will be on IPv4 subnet and 2nd will be on IPv6 subnet.

For example:

Network name: ext-net

Subnets:

Name	Network Address	IP Version	Gateway IP
ext-net-subnet	10.75.189.128/25	IPv4	10.75.189.129
ext-net-ipv6-subnet	2606:b400:605:b818::/64	IPv6	2606:b400:605:b818:6e41:6aff:fec7:80bf

Steps to migrate DSR NOAM on IPv6:

1. Create DSR Noam through VNFM. Provide dual subnet network (xmi & imi) to creating the DSR Noam set up. DSR NOAM will be up & running with IPv4 network interface through VNFM and will create the IPv6 IP address in Openstack for both xmi/imi.
2. Add the allowed address for IPv6 manually through Openstack cli command for both active/standby NOAM.

Note: User should have permission to add the allowed address to port through Openstack cli.

Execute the following command to add the allowed address pair in port:

```
openstack port set --allowed-address ip-address=<vip ipv6 address>
<active noam port id>
```

```
openstack port set --allowed-address ip-address=<vip ipv6 address>
<standby noam port id>
```

For example:

```
openstack port set --allowed-address ip-
address=2606:b400:605:b818:6e41:6aff:fec7:80cf a2d4fe19-d5e8-4a18-b08c-
0057e68d2bde
```

3. Follow the document *Dual IP Stack migration* to add the IPv6 interface for active/standby NOAM xmi, imi and VIP.
4. While adding IPv6 interface, use the same IPv6 IP address for active/standby xmi & imi which is created through VNFM for DSR NOAM.
 - a. Go to Openstack GUI.
 - b. Navigate to **Network -> <network name>** and locate the active/standby & vip port.
 - c. Open the port to obtain the created IPv4 & IPv6 address.

15. Troubleshooting VNFM

15.1 Debug VNFM

To debug issues during VNFM deployment, check the following log files:

- VNFM logs are located in `/var/vnfm/logs/vnfm.log`.
- Tomcat logs are located in `/usr/share/vnfm/apache-tomcat-9.0.16/logs/catalina.out`.

15.2 Enable VNFM Logs with different Log Levels (DEBUG, TRACE, WARN, ERROR)

- Open the file `log4j2.xml` located in `/opt/vnfm/config/`
- Replace `level="INFO"` with `level="DEBUG"` (or `TRACE` or `WARN` or `ERROR`) in `<Logger>` tag and save

Note:

- Default value of level is "INFO"

15.3 Enable VNFM after shutdown or reboot

Perform the following in case of shutdown or reboot:

- The tomcat server is configured to shutdown automatically.
To restart the tomcat server, run the `startup.sh` script, by executing:
`sudo ./startup.sh`
Note: The `startup.sh` script is available in `/usr/share/vnfm/apache-tomcat-9.0.16/bin`
- The `iptables` service starts during the reboot. This must be stopped to enable the REST services:
To stop the `iptables` service:
 - In case the VNFM external interface(eth0) is IPv4, execute:
`sudo service iptables stop`
 - In case of VNFM external interface(eth0) is IPv6:
`sudo service ip6tables stop`

15.4 Resolve HA Alarms on VNF through VNFM Deployed Setup

Perform the following to resolve the HA alarms:

1. Check the ping request and response packets from Server-A and Server-B for which alarm has been raised, by executing:
`tcpdump -i eth1 -n "host <server-A>-imi or <server-B>-imi and port 17401 and udp"`

For example:

```
tcpdump -i eth1 -n "host noam00-17badf67-imi or noam01-17badf67-imi and
port 17401 and udp"
```

2. If ping request or response packets are not coming from any server, then add security group rule ingress (response) or egress (request) to that instance.
3. Check the ping packets again after adding the rule and ensure that `imi` request and response packets are received from each servers, by executing:

```
tcpdump -i eth1 -n "<server-A>-imi or <server-B>-imi and port 17401 and  
udp"
```

4. Now restart the `cmha` process on the node where the alarms are present, by executing:
`pm.set off cmha && sleep 5 && pm.set on cmha`

Note: If the Node is HA Active, then restarting `cmha` will cause switchover.

Appendix A. My Oracle Support (MOS)

MOS (<https://support.oracle.com>) is your initial point of contact for all product support and training needs. A representative at Customer Access Support (CAS) can assist you with MOS registration.

Call the CAS 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 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 MOS, select 2.

You are connected to a live agent who can assist you with MOS registration and opening a support ticket. MOS is available 24 hours a day, 7 days a week, and 365 days a year.

Emergency Response

In the event of a critical service situation, emergency response is offered by the 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.

Locate Product Documentation on the Oracle Help Center

Oracle Communications customer documentation is available on the web at the Oracle Help Center (OHC) site, <http://docs.oracle.com>. You do not have to register to access these documents. Viewing these files requires Adobe Acrobat Reader, which can be downloaded at <http://www.adobe.com>.

1. Access the **Oracle Help Center** site at <http://docs.oracle.com>.
2. Click **Industries**.
3. Under the **Oracle Communications** subheading, click the **Oracle Communications documentation** link. The Communications Documentation page displays. Most products covered by these documentation sets display 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 displays. 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.