Oracle® Communications

Diameter Signaling Router

API Gateway Installation Guide

Release 8.5

F35452-02

September 2021

ORACLE

Oracle Communications API Gateway Installation Guide.

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.

See more information on MOS in the Appendix section.

Table of Contents

List of Figures

List of Tables

List of Procedures

1. Introduction

1.1 Purpose and Scope

This document describes the installation procedures for OpenStack HEAT template.

This document assumes platform-related configuration has already been done.
The audience for this document includes Oracle customers as well as these groups: Software System, Product Verification, Documentation, and Customer Service including Software Operations and First Office Application.
The document describes installation procedure for the following three components for DSR APIGW:

- OCSG Database Server
- OCSG Admin Server
- OCSG Application Server

1.2 References

[1] DSR Cloud Benchmarking Guide
[2] DSR Cloud Installation Guide
[3] DSR API Gateway User Guide

1.3 Acronyms

Acronym	Definition
APIGW	API Gateway
CLI	Command Line Interface
KVM	Kernel-based Virtual Machine
OVA	Open Virtualization Archive
OVM-M	Oracle VM Manager
OVM-S	Oracle VM Server
OHC	Oracle Help Center
SSO	Single Sign On
YAML	Yet Another Markup Language
OCSG	Oracle communications services Gatekeeper
PEM	Privacy Enhanced Mail

1.4 General Procedure Step Format

Figure 1. Example of a procedure step illustrates the general format of procedure steps as they appear in this document. Where it is necessary to explicitly identify the server on which a particular step is to be taken, the server name is given in the title box for the step.

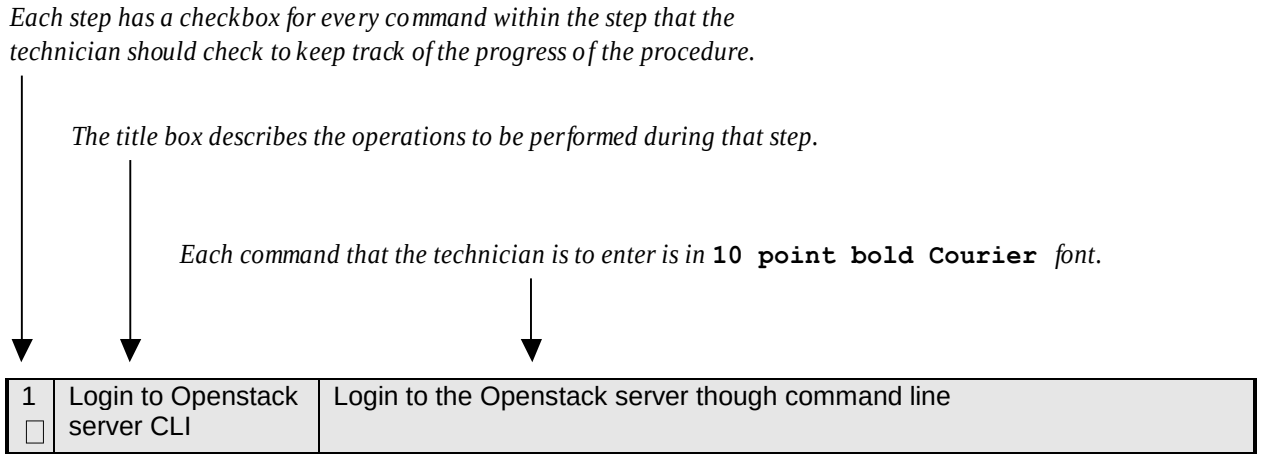


Figure 1. Example of a procedure step

2. Overview

2.1 Prerequisites

Following are the prerequisites for installing DSR APIGW:

1. KVM/OpenStack admin and tenant privileges
2. OCSG Patches must be downloaded from mysupport (if required).
3. DSR APIGW OVA (will be used for Admin and Application Server installation)
4. The **mysql-ndbcluster-7_6_8.qcow2** (will be used for Database Server installation)
5. The **.pem** file must be available in Openstack.
6. DSR APIGW Database server must be configured and accessible from DSR APIGW Admin and Application VMs.
7. Following YAML files are required: For DSR APIGW Admin/Application server: **dsrapigw.yml** and **dsrapigw_env.yml**.
8. Qemu-img tool must be available to convert VMDK to qcow2 format, if required.

3. Installation Overview

This section provides a brief overview of the recommended methods for installing the HEAT template. It also lists the procedures required for installation with estimated times.

3.1 Installation Procedures

The following table illustrates the progression of the installation process by procedure with estimated times. The estimated times and the phases that must be completed may vary due to differences in typing ability and system configuration. The phases outlined in are to be executed in the order they are listed.

Installation and configuration of instances can be performed either on VMware or KVM/Openstack. On KVM/Openstack, user can install and configure instances either manually or using HEAT template. Following are the sections wherein the sequence of procedures are explained.

3.1.1 Install and Configure Instances on VMware

Following table explains the sequence to be followed on VMware:

Table 1: Install and configure instances on VMware

Procedure	Title	Description
Procedure 1	Import DSR APIGW Database and Admin/Application OVAs	Import both DSR APIGW Database, Admin and Application server OVAs
Procedure 8	MySQL NDB Cluster Installation and Configuration	Install and Configure MySQL NDB Cluster
Procedure 2**Error! Reference source not found.**	Create DSR APIGW Admin/Application servers**Error! Reference source not found.**	Create Admin and Application VMs
27Procedure 9	Configure DSR APIGW Admin/Application Server	Install and configure DSR APIGW Admin/Application server

3.1.2 Install and Configure Instances on KVM/Openstack

Following table explains the sequence to be followed on KVM/Openstack:

Table 2: Install and configure instances on KVM/Openstack manually

Procedure	Title	Description
Procedure 3	Import DSR APIGW Database and Admin/Application OVAs (Openstack)	Import both DSR APIGW database, admin, application server OVAs
Procedure 8	MySQL NDB Cluster Installation and Configuration	Install and Configure MySQL NDB Cluster
Procedure 4	Create DSR APIGW Admin/Application VMs (Openstack)	Create DSR APIGW Admin and Application VMs
27Procedure 9	Configure DSR APIGW Admin/Application Server	Install and configure DSR APIGW Admin/Application server
Procedure 5	Download Openstack HEAT template and parameter files	Download the template and parameter files from OHC
Procedure 6	Create DSR APIGW Database and Admin/Application Parameter Files	Create parameter file based on your configuration
Procedure 7	Deploy DSR APIGW Database and Admin/Application using HEAT templates	Deploy the servers using HEAT template

Table 3: Install and configure instances on KVM/Openstack using Heat Template

Procedure	Title	Description
Procedure 5	Download Openstack HEAT template and parameter files	Download the template and parameter files from OHC
Procedure 6	Create DSR APIGW Database and Admin/Application Parameter Files	Create parameter file based on your configuration
Procedure 7	Deploy DSR APIGW Database and Admin/Application using HEAT templates	Deploy the servers using HEAT template
Procedure 8	MySQL NDB Cluster Installation and Configuration	Install and Configure MySQL NDB Cluster
27Procedure 9	Configure DSR APIGW Admin/Application Server	Install and configure DSR APIGW Admin/Application server

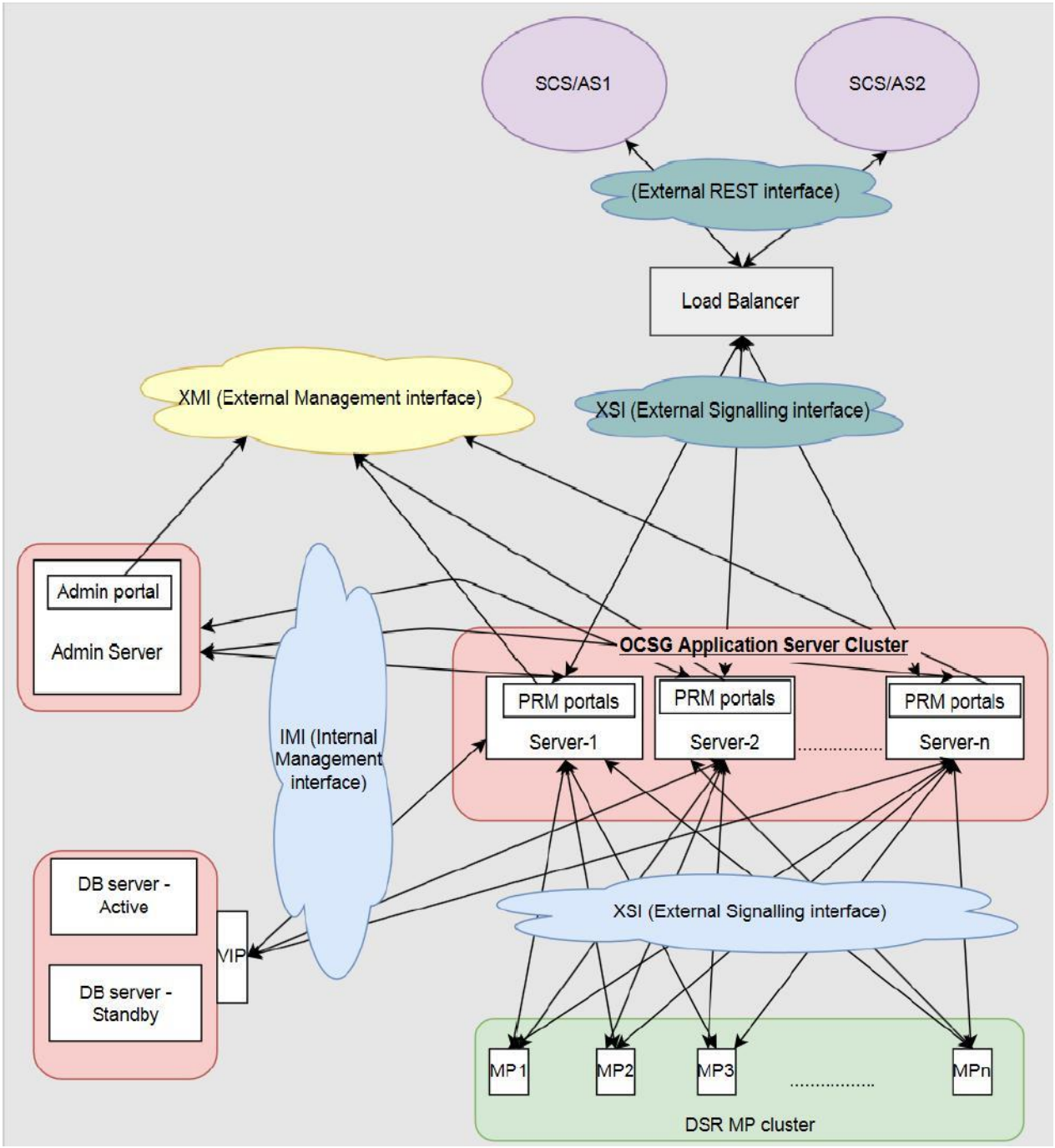
3.2 Network model

Below diagram depicts the supported network model for DSR APIGW deployments. DSR APIGW will be deployed in cluster mode and one to one mapping should be maintained between DSR site and DSR APIGW cluster.

DSR APIGW deployment model will have 3 networks:

1. XMI - External Management Interface, which will expose Administrative portal, Partner management and Partner portals. Ports 9002 will be opened for management traffic on XMI. Links to portals:
    - Admin portal - https:<Adm in-server-XMI-IP>:9002/console
    - Partner management portal - https:<AppServer-XMI-IP>:9002/portal/partner-manager/index/login.html
    - Partner Portal - https://<AppServer-XMI-IP>:9002/portal/partner/index/partnerLogin.html
2. IMI- Internal management interface - This interface will be used within DSR APIGW cluster between DSR APIGW - Database for internal communication.
3. XSI - External Signalling interface - This interface will be used to receive and send network traffic from and to app-servers. Ports 10001 for http traffic and 10002 for https traffic on XSI interface.

Figure 2: Network Model



4. Software Installation on VMware

As mentioned earlier, the host configuration and virtual networks should be done before executing the procedures in this document. It is assumed that at this point the user has access to:

- Consoles of all guests and hosts at all sites
- ssh access to the guests at all sites
- GUI access to hosts at all sites
- A configuration station with a web browser, ssh client, and scp client
- VM Manager Privileges to add OVA's to catalog (VMware only)
- VMware, KVM/OpenStack admin and tenant privileges

4.1 Create Instances on VMware

4.1.1 Import DSR APIGW Database and Admin/Application OVAs (VMware)

Procedure 1. Import DSR APIGW Database and Admin/Application OVAs (VMware)

STEP #	Procedure	Description
This procedure describes steps to import the DSR APIGW Database and Admin/Application OVAs to the VMware catalog or repository. Check off (√) each step as it is completed. Steps with shaded boxes require user input. If this procedure fails, contact My ORACLE Support (MOS) and ask for assistance.		
1 ☐	**VMware client:** Add DSR APIGW Database image	1. Launch the VMware client of your choice. 2. Add the DSR APIGW Database image to the VMware catalog or repository. Follow the instructions provided by the Cloud solutions manufacturer.
2 ☐	**VMware client:** Add DSR APIGW Admin/Application OVA image	1. Launch the VMware client of your choice. 2. Add the DSR APIGW Admin/Application OVA image to the VMware catalog or repository. Follow the instructions provided by the Cloud solutions manufacturer.

4.1.2 Create DSR APIGW Database VMs (VMware)

Note: Refer to DSR APIGW Database section for configuring database. Refer to Configure DSRAPIGW Admin/Application Server section for configuring the admin and application servers.

4.1.3 Create DSR APIGW Admin/Application VMs (VMware)

Procedure 2. Create DSR APIGW Admin/Application servers

STEP #	Procedure	Description
This procedure describes steps to create all admin and application servers. ***Note***: This procedure provides an example for creating an Admin. Follow the same steps to create other guests with their respective VM names and profiles. Check off (√) each step as it is completed. Boxes have been provided for this purpose under each step number. If this procedure fails, contact My ORACLE Support (MOS) and ask for assistance.		
1 ☐	**VMware client:** Create the Admin VM from the OVA image	1. Browse the library or repository that you placed the **DSR APIGW OVA** image. 2. Deploy the OVA image using **vSphere Client** or **vSphere Web Client**. 3. Name the **Admin VM** and select the data store.

STEP #	Procedure	Description
2 ☐	**VMware client:** Configure resources for the Admin VM	Configure the **Admin VM** per the resource profiles defined in Appendix E for the **DSR APIGW** Admin server using the **vSphere Client** or **vSphere Web Client**. Interfaces must be added per described in Network model section.
3 ☐	**VMware client:** Power on Admin VM	1. Power on the **Admin VM** with the **vSphere Client** or **vSphere Web Client**. 2. Monitor the vApps screen's Virtual Machines tab until the Admin VM reports **Powered On** in the Status column.

4
☐
	VMware client: Configure XMI interface	1. Access the **VM console** via the **vSphere Client** or **vSphere Web Client**.
		2. Login as the **admusr** user.
		3. Set the ethX device:
		Note: Where ethX is the interface associated with the XMI network.
		`$ sudo netAdm add --device=<ethX> --address=<IP Address in External Management Network> --netmask=<Netmask>--onboot=yes --bootproto=none`
		4. Add the default route for ethX:
		Note: The below step of adding gateway should be done only to the externally routable network.
		`$ sudo netAdm add --route=default --gateway=<gateway address for the External management network>--device=<ethX>`
		5. Ping the XMI gateway for network verification.
		`$ ping –c3 <Gateway of External Management Network>`
		6. Depending on the number of instances, configuring network interfaces (step 4) should be repeated for each network (imi, XSI1, XSI2 etc.)
		7. Restart network
		`$ service network restart`

5 ☐	**VMware client:** Verify network connectivity	1. Access the **Admin VM console** using the **vSphere Client** or **vSphere web Client**. 2. Login as the **admusr** user. 3. Ping the Admin. `$ ping –c3 <IP Address in External Management Network>`
6 ☐	**VMware client:** Repeat for other Application VMs	Repeat steps 1 through 5 for the Application VMs. Use unique labels for the VM names.

5. Software Installation on KVM/Openstack

5.1 Create Instances on KVM/OpenStack Manually

5.1.1 Import DSR APIGW Database and Admin/Application OVAs (Openstack)

Procedure 3. Import DSR APIGW Database and Admin/Application OVAs (Openstack)

STEP #	Procedure	Description
This procedure adds the DSR APIGW Admin/Application and Database OVA files to the glance image catalog. Check off (√) each step as it is completed. Steps with shaded boxes require user input. If this procedure fails, contact My ORACLE Support (MOS) and ask for assistance.		
1 ☐	**Openstack Controller:** Preparation	Create instance flavors.

2
☐
	Openstack Controller: Add DSR APIGW Database OVA image	1. Copy the DSR APIGW Database OVA file from Oracle repository to the OpenStack control node.
		`$ scp <user_name>@<Oracle Repository server>:<path-to-OVA>/DSR-8.5.0.0.0_90.10.0.ova.`
		2. In an empty directory, unpack the OVA file using **tar**.
		`$ tar xvf DSR-x.x.x.x.x.ova`
		3. One of the unpacked files has a **.vmdk** suffix. This is the VM image file that must be imported.
		`DSR-x.x.x.x.x-disk1.vmdk`
		4. Source the OpenStack **admin** user credentials.
		`$ . keystonerc_admin`
		5. Select an informative name for the new image.
		`dsr-8.5.x.x.x-original`
		6. Import the image using the **glance** utility from the command line**.**
		`openstack image create --disk-format vmdk --container-format bare --public --file dsrapigw-x.x.x.x.vmdk dsrapigw-x.x.x.x-original`
		This process takes about 5 minutes, depending on the underlying infrastructure.
		This step is complete.
		In case you want to convert vmdk file to qcow2 format, refer to Appendix F.
		Note: This process will take about 5 minutes, depending on the underlying infrastructure.

3 ☐	**Openstack Controller:** Add DSR APIGW OVA	Repeat above steps to add `DSRAPIGW-8.5.0.0.0_90.10.0-x86_64.iso` DSR APIGW OVA.

5.1.2 Create DSR APIGW Database VMs (Openstack)

Note: Refer to DSR APIGW Database section for configuring database. Refer to Configure DSR APIGW Admin/Application Server section for configuring the admin and application servers.

5.1.3 Create DSR APIGW Admin/Application VMs (Openstack)

Procedure 4. Create DSR APIGW Admin/Application VMs (Openstack)

Step#	Procedure	Description
This procedure describes steps to configure all VMs i.e. Admin and Application Servers. ***Note***: This procedure provides an example for creating an Admin. Follow the same steps to create other guests with their respective VM names and profiles. Check off (√) each step as it is completed. Boxes have been provided for this purpose under each step number. If this procedure fails, contact My ORACLE Support (MOS) and ask for assistance.		
1 ☐	**Openstack Controller:** Name the new VM instance	1. Create an informative name for the new instance: **Admin**. 2. Examine the interfaces must be added per described in Network model section.

2
☐
	Openstack Controller: Create and boot the Admin and Application VM instance from the glance image	1. Get the following configuration values.
		a. The DSR APIGW Admin/Application image ID.
		`$ glance image-list`
		b. The flavor ID.
		`$ nova flavor-list`
		c. The network ID(s)
		`$ neutron net-list`
		d. An informative name for the instance.
		• **Admin**
		• **Application**
		1. Create and boot the VM instance.
		Refer to Appendix E regarding the resource profile.
		The instance must be owned by the tenant user, not the admin user. Source the credentials of the DSR tenant user and issue the following command. Use one **--nic** argument for each IP/interface. Number of IP/interfaces for each VM type must confirm with the OCDSR Network to Device Assignments defined in [1].
		Note: IPv6 addresses should use the **v6-fixed-ip** argument instead of **v4-fixed-ip**.
		Admin server
		`nova boot --image <image ID> --flavor <flavor id> --nic net-id=<XMI network id>,v4-fixed-ip=<XMI ip address> --nic net-id=<IMI network id>,v4-fixed-ip=<IMI ip address> <instance name>`
		App server
		`nova boot --image <image ID> --flavor <flavor id> --nic net-id=<XMI network id>,v4-fixed-ip=<XMI ip address> --nic net-id=<IMI network id>,v4-fixed-ip=<IMI ip address> --nic net-id=<XSI network id>,v4-fixed-ip=<XSI ip address> <instance name>`
		2. View the newly created instance using the nova tool.
		`$ nova list --all-tenants`
		The VM takes approximately 5 minutes to boot and may be accessed through both network interfaces and the Horizon console tool.

Step#	Procedure	Description
3 ☐	**Openstack Controller:** Check if interface is configured	If DHCP is enabled on Neutron subnet, VM configures the VNIC with the IP address provided in step 2 above. To verify, ping the IP address provided with nova boot… command (step 2): `$ ping <IP-Provided-During-Nova-Boot>` If the ping is successful, ignore step 5 to configure the interface manually.

4
☐
	Openstack GUI: Manually configure interface, if not already done (Optional)	***Note***: If the instance is already configured with an interface and successfully pinging (step 4), then **ignore** this step to configure the interface manually.
		1. Log into the **Horizon** GUI as the DSR tenant user.
		2. Go to the **Compute/Instances** section.
		3. Click the **Name** field of the newly created instance.
		4. Select the **Console** tab.
		5. Login as the **admusr** user.
		6. Configure the network interfaces, conforming with the Network model section.
		`$ sudo netAdm add --onboot=yes --device=eth0 --address=<ip> --netmask=<net mask>`
		Note: The below step of adding gateway should be done only to the externally routable network.
		`$ sudo netAdm add --route=default --device=eth0 --gateway=<gateway ip>`
		Verify network connectivity by pinging Gateway of network.
		`$ ping –c3 <Gateway>`
		Under some circumstances, it may be necessary to configure as many as 6 or more interfaces.
		7. Depending on the number of instances, configuring network interfaces (step 6) should be repeated for each network (imi, XSI1, XSI2 etc.)
		8. Restart network
		`$ service network restart`
		9. Reboot the Admin VM. It takes approximately 5 minutes for the VM to complete rebooting.
		`$ sudo init 6`
		The new VM should now be accessible via both network and Horizon consoles.

5 ☐	Repeat for other application VMs	Repeat steps 1 through 4 for the other application VMs. Use unique labels for the VM names. Assign addresses to all desired network interfaces.

5.2 Create Instances on KVM/Openstack using Heat Template

5.2.1 Download Openstack Template and Parameter Files

Procedure 5. Download Openstack HEAT template and parameter files

Step #	Procedure	Description
This procedure instructs to select required templates and environment files to be provided while deploying DSR APIGW and DSR stacks. **Prerequisite**: All the respective infrastructures has to be up and running Check off (√) each step as it is completed. Boxes have been provided for this purpose under each step number. If this procedure fails, contact My ORACLE Support (MOS) and ask for assistance.		
1 ☐	Login to Oracle document repository - OHC	Login to the Oracle Document Repository. Link: http://docs.oracle.com/en/industries/communications/diameter-signaling-router/index.html
2 ☐	Select the DSR Release	Select the respective release folder Example: Release 8.5.x
3 ☐	Download HEAT templates	Login to Openstack controller and navigate to home directory where you want to store the HEAT templates. Download the **HEAT Templates** zip file.
4 ☐	**Openstack Controller:** Unzip the HEAT templates to a folder in Openstack	1. Create a new folder with any name for storing the heat templates, under home directory. Example: '/home/heat_templates' 2. Store the downloaded heat templates zip file in Step 3, to the above created folder. Example : '/home/heat_templates/exampleHeat.zip' 3. Unzip the downloaded heat templates. ***unzip /home/heat_templates/exampleHeat.zip***
5 ☐	Determine the Template and Environment Files	The HEAT templates downloaded contains files for all scenarios. Determine the appropriate template and parameter files with respect to your requirement. The YAML files for DSR APIGW admin/application servers are **dsrapigw.yml** and **dsrapigw_env.yml.** The YAML files for DSR APIGW admin/application servers are **dsrapigw.yml** and **dsrapigw_env.yml**.

5.2.2 Create DSR APIGW Database and Admin/Application Parameter Files

Procedure 6. Create DSR APIGW Database and Admin/Application Parameter Files

STEP #	Procedure	Description
This procedure instructs how to manually create input parameters file to be provided while deploying DSR APIGW and DSR. **Prerequisite**: All the respective infrastructures has to be up and running Check off (√) each step as it is completed. Boxes have been provided for this purpose under each step number.x` If this procedure fails, contact My ORACLE Support (MOS) and ask for assistance.		
1 ☐	**Openstack Controller:** Login to Openstack controller	Login to the Openstack controller though command line.
2 ☐	**Openstack Controller:** Create the parameter file	1. Navigate to the folder which is already created in the above procedure for storing the templates. 2. Create an empty parameter file in this folder, following the below naming convention just to identify the purpose of the file: For DSR APIGW Admin/Application: **<DSR APIGW Name>_Params.yml**

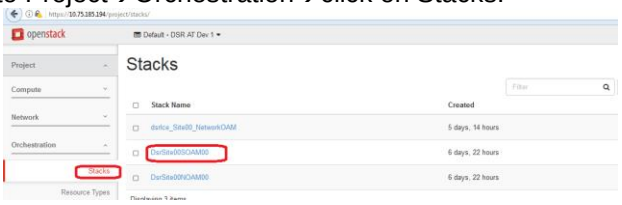
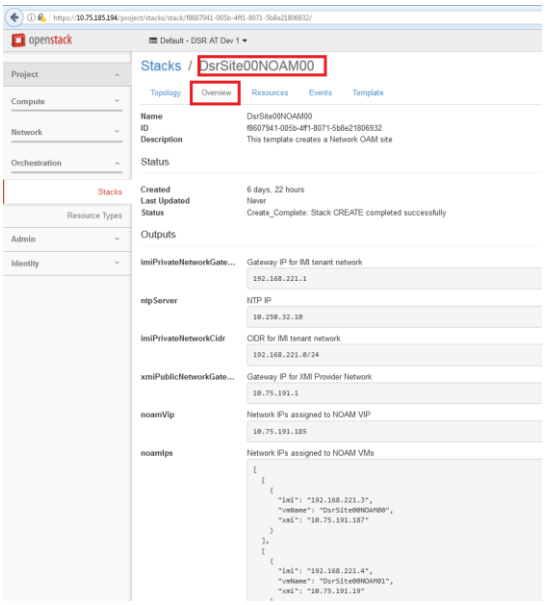
STEP #	Procedure	Description
		Example: *dsrapigw_Params.yml*
3 ☐	**Openstack Controller:** Sample File	Refer to 8 for a sample file with the values. **Note:** It is important to keep the Example File handy as this will help in understanding the use of each Key Value pair which is described in the steps below while creating the Parameter File.
4 ☐	**Openstack Controller:** Populate the parameters file as follows	Refer **8** to create the parameter file in YAML format. **Note: Make sure the below guidelines are followed while working with the YAML files.** • The file must end with .yaml extension. • YAML must be case-sensitive and indentation-sensitive. • YAML doesn't support the use of tabs. Instead of tabs, it uses spaces. 1. This file is in YAML format and it contains 'key:value' pairs 2. The first key should be 'parameters:' and then followed by the remaining required key/value pairs for the topology 3. Refer to 8 for all required key value pairs

5.2.3 Deploy DSR APIGW Database and Admin/Application using HEAT Templates

This section describes the procedure to deploy DSR APIGW Database and Admin/Application using HEAT templates.

Procedure 7. Deploy DSR APIGW Database and Admin/Application using HEAT templates

STEP #	Procedure	Description
This procedure instructs how to deploy HEAT templates to create DSR APIGW admin and application stacks. **Prerequisite**: All the respective infrastructures has to be up and running. The required input files are all available. Check off (√) each step as it is completed. Boxes have been provided for this purpose under each step number. If this procedure fails, contact My ORACLE Support (MOS) and ask for assistance.		
1 ☐	**Openstack Controller:** Login to OpenStack server CLI	If not already done, login to the OpenStack CLI
2 ☐	**Openstack Controller:** Prepare the input files required for the deployment	It is required to provide the parameter file as input while deploying the HEAT templates to create DSR APIGW admin and application stacks.
3 ☐	**Openstack Controller:** Deploy DSR APIGW stack	Execute the below OpenStack command to create DSR APIGW admin and application stack, passing the above 3 input files. Make sure the Template and Parameter files are selected with respect to DSR APIGW admin and application stack. `openstack stack create -e <ParameterFile.yaml> -t <TemplateFile>`
4 ☐	**Openstack Controller:** Verify the stack creation status	After the OpenStack create commands are executed, execute the below command to see the stack creation status: `$ openstack stack show <stackname>`

STEP #	Procedure	Description
		+--------------------------------------+-----------+-------------------+----------------------+ | ID | Name | Status | Created | +--------------------------------------+-----------+-------------------+----------------------+ | (uuid) | teststack | CREATE_IN_PROGRESS | (timestamp) | +--------------------------------------+-----------+-------------------+----------------------+ It will take approximately 2 minutes to complete the creation. Execute the command again to verify the status `$ openstack stack show <stackname>` +--------------------------------------+----------------+-----------------+ | ID | Stack Name | Stack Status | +--------------------------------------+----------------+-----------------+ | 950ed51a-cca7-478a-81e4-3d61562c045d | teststack | CREATE_COMPLETE |
5 ☐	**Openstack Controller:** Retrieve required IP's from created stacks	a) Login to openstack GUI with valid credentials.  b) Go to Project→Orchestration→click on Stacks.  c) Click on the stack that you have created (<stackname>) and then click on "Overview". After clicking you can see all IP details of specific stack that is created. 
6 ☐	**Openstack GUI:** Manually configure interface, if not already done (Optional)	1. Log into the openstack Horizon GUI 2. Go to the **Compute/Instances** section 3. Click the **Name** field of the newly created instance. 4. Select the **Console** tab

STEP #	Procedure	Description

		5. Login as the root user.
		
		6. Configure the network interfaces
		
		`$ netAdm add --onboot=yes --device=eth0 --address=<xmi ip> --netmask=<xmi net mask>`
		Note: The below step of adding gateway should be done only to the externally routable network.
		`$ netAdm add --route=default --device=eth0 --gateway=<xmi gateway ip>`
		
		7. Verify network connectivity by pinging Gateway of XMI network.
		`$ ping -c3 <XMI Gateway ip>`
		
		8. Depending on the number of instances, configuring network interfaces (step 6) should be repeated for each network (imi, XSI1, XSI2 etc.)
		
		9. Restart network
		
		`$ service network restart`

Note: Refer to DSR APIGW Database section for configuring database. Refer to Configure DSR APIGW Admin/Application Server section for configuring the admin and application servers.

6. DSR APIGW Database

The NDB Cluster is a technology that enables clustering of in-memory databases in a shared- nothing system. The shared-nothing architecture enables the system to work with very inexpensive hardware, and with a minimum of specific requirements for hardware or software.

The NDB Cluster is designed not to have any single point of failure. In a shared-nothing system, each component is expected to have its own memory and disk, and the use of shared storage mechanisms such as network shares, network file systems, and SANs is not recommended or supported.

The NDB Cluster integrates the standard MySQL server with an in-memory clustered storage engine called NDB (which stands for "Network DataBase").

An NDB Cluster consists of a set of computers, known as hosts, each running one or more processes. These processes, known as nodes, may include MySQL servers (for access to NDB data), data nodes (for storage of the data), one or more management servers, and possibly other specialized data access programs. When data is stored by the NDB storage engine, the tables (and table data) are stored in the data nodes. Such tables are directly accessible from all other MySQL servers (SQL nodes) in the cluster. Thus, in a payroll application storing data in a cluster, if one application updates the salary of an employee, all other MySQL servers that query this data can see this change immediately.

In addition, a MySQL server that is not connected to an NDB Cluster cannot use the NDB storage engine and cannot access any NDB Cluster data.

The data stored in the data nodes for NDB Cluster can be mirrored; the cluster can handle failures of individual data nodes with no other impact than that a small number of transactions are aborted due to losing the transaction state. Because transactional applications are expected to handle transaction failure, this should not be a source of problems.

Individual nodes can be stopped and restarted, and can then rejoin the system (cluster). Rolling restarts (in which all nodes are restarted in turn) are used in making configuration changes and software upgrades.

6.1 Install and Configure MySQL NDB Cluster

Procedure 8. MySQL NDB Cluster Installation and Configuration

STEP#	Procedure	Description
This procedure installs and configures the MySQL NDB Cluster. Check off (√) each step as it is completed. Boxes have been provided for this purpose under each step number. If this procedure fails, contact My ORACLE Support (MOS) and ask for assistance.		
1. ☐	Bring up VM's with mysql image	NDB Cluster has minimum of 6 VM's: 2 VM's for Management Node, 2 VM's for Data Node, 2 VMs for SQL Node.
2. ☐	SSH to VM's with pem file	SSH to VM's with pem file, by executing: `# ssh -i <pem file> root@ipaddress of vm`
3. ☐	Set the password for root	Set the password for root, by executing: `passwd`
4. ☐	Disable the firewall	Disable the firewall on all nodes, by executing: `systemctl disable firewalld`

5.
☐
	Update `config.ini` file	Update config.ini file on all management nodes and on all data nodes under the location `/var/lib/mysql-cluster` to provide correct ipaddresses of all nodes. Modify DataMemory parameter on all management nodes and data nodes as per the suggestion provided in the below sample file:
		`[ndb_mgmd]`
		`#Management Node1 db1`

STEP#	Procedure	Description
		NodeId=1 HostName=<Ipaddress of Managment Node1> [ndb_mgmd] #Management Node2 NodeId=2 HostName=<Ipaddress of Managment Node2> [ndbd default] NoOfReplicas=2 # Number of replicas DataMemory=3072M # Memory allocate for data storage. Assuming data node is having 4GB RAM. If 64GB RAM profile is used for data nodes , please change it to 48GB. Suggested to use Maximum of 75% of your RAM. IndexMemory=384M # Memory allocate for index storage. Assuming data node is having 4GB RAM. If 64GB RAM profile is used for data nodes, please change it to 6144M. Suggested to use Maximum of 9.375% of your RAM. #Directory for Data Node DataDir=/var/lib/mysql-cluster MaxNoOfAttributes=1000000 StopOnError=0 #<Some more parameters will go here. Verify in corresponding nodes.> [ndbd] #Data Node 1 NodeId=3 HostName=<Ipaddress of Data Node1> [ndbd] #Data Node 2 NodeId=4 HostName=<Ipaddress of Data Node2> [mysqld] NodeId=5 HostName=<Ipaddress of SQL Node1> #SQL Node1 [mysqld] NodeId=6

STEP#	Procedure	Description

6.
☐
	Disable and stop mysqld process	Disable & stop mysqld process running on all datanodes, by executing:
		
		`systemctl disable mysqld`
		
		`systemctl stop mysqld`

7.
☐
	Update `my.cnf` file on data nodes	Update `my.cnf` file on all data nodes, by executing:
		
		`vi /etc/my.cnf`
		
		`[mysqld] #` This section already exists, so just add from below lines to `my.cnf` file
		
		`max_connections = 350`
		
		`wait_timeout = 300`
		
		`interactive_timeout = 300`
		
		`ndbcluster`
		
		`ndb-connectstring=<Ip Address1 of Mgmt Node>,<Ip Address2 of Mgmt Node>`
		
		`[mysql_cluster]`
		
		`ndb-connectstring=<Ip Address1 of Mgmt Node>,<Ip Address2 of Mgmt Node>`
		
		Note: Make sure to update the actual Ip addresses of management nodes.

8.
☐
	Update `my.cnf` file on SQL nodes	Update `my.cnf` file on all SQL nodes as following:
		
		`vi /etc/my.cnf`
		
		[`mysqld`]# This section already exists, so just add from below lines to `my.cnf` file
		
		`max_connections = 350`
		
		`wait_timeout = 300`
		
		`interactive_timeout = 300`
		
		`ndbcluster`
		
		`ndb-connectstring=<Ip Address1 of Mgmt Node>,<Ip Address2 of Mgmt Node># IP address for server management node default_storage_engine=ndbcluster# Define default Storage Engine used by MySQL`
		
		`[mysql_cluster]`
		`ndb-connectstring=<Ip Address1 of Mgmt Node>,<Ip Address2 of Mgmt Node># IP address for server management node`
		
		Note: Make sure to update the actual Ip addresses of management nodes.

STEP#	Procedure	Description
9. ☐	Connect SQL node to data node	Execute the following command on all SQL nodes to connect the SQL nodes to Data node: `rm -f /var/lib/mysql-cluster/config.ini`

10. ☐	Restart the MySQL service	Restart the MySQL service on all SQL nodes, by executing:
		
		`systemctl start mysqld`
		`systemctl enable mysqld`

11. ☐	Configure root password on all the SQL nodes	Get the temporary password that is required in the next step, by executing: `grep 'temporary' /var/log/mysqld.log`

12. ☐	Set the MySQL credentials	Set the MySQL credentials by executing the following command and provide the password for the root user obtained from the previous step and provide new password on all SQL nodes:
		
		`mysql_secure_installation`
		
		Note: Please make note of the password set for mysql during this process.
		
		`[root@vmdk-datanode ~]# mysql_secure_installation`
		
		Reset the password and answer the following questions as suggested:
		
		`Change the password for root ? ((Press y|Y for Yes, any other key for No) : y`
		
		`Do you wish to continue with the password provided? (Press y|Y for Yes, any other key for No) : y`
		
		`Remove anonymous users? (Press y|Y for Yes, any other key for No) : y`
		
		`Disallow root login remotely? (Press y|Y for Yes, any other key for No) :n`
		
		`Remove test database and access to it? (Press y|Y for Yes, any other key for No) : n`
		
		`Reload privilege tables now? (Press y|Y for Yes, any other key for No) : y`
		
		Once all the questions are answered, the following message is displayed:
		
		`Success & All Done`

13. ☐	Login to MySQL	Login to MySQL, by executing: `mysql -u root -p` **Note**: Use the configured password for the root user.
14. ☐	Create a new user	Create a new user, by executing: `CREATE USER 'mysqluser'@'localhost' IDENTIFIED BY '<newpasswordhere>';`

STEP#	Procedure	Description
		Here a user named `mysqluserand` is created provided with necessary grants. **Note**: Any username of your choice can be created but changes need to be made accordingly for all commands provided in further steps. It is suggested to create same username on all SQL nodes.
15. ☐	MySQL user password should never be expired	Set the MySQL user password to never expire, by executing: `ALTER USER 'mysqluser'@'localhost' PASSWORD EXPIRE NEVER;`
16. ☐	Create database gatekeeper	Create database gatekeeper, by executing: `CREATE DATABASE gatekeeper;`
17. ☐	Grant privileges	Grant privileges to the newly created user, by executing: `GRANT ALL PRIVILEGES ON gatekeeper.* TO 'mysqluser'@'%' Identified By '<newpasswordhere>'; FLUSH PRIVILEGES; EXIT;`
18. ☐	Login to mysql	Login to mysql with the newly created user and provide the corresponding password, by executing: `mysql -u mysqluser -p`
19. ☐	Check visibility of gatekeeper database	Check that the gatekeeper database is visible, by executing: `SHOW DATABASES;`
20. ☐	Verify on all SQL nodes	Make sure that steps applicable for SQL node are followed on all SQL/API nodes, such as from steps 9 to 18.

21. ☐	Initialize `ndb_mgmd` process	To initialize the `ndb_mgmd` process on all NDB management nodes, execute:
		
		`sudo chmod +x /etc/init.d/ndb_mgmd chkconfig --add ndb_mgmd`
		
		`service ndb_mgmd start`

22. ☐	Start `ndbd` process	To start `ndbd` process on all the data nodes, execute: `sudo chmod +x /etc/init.d/ndbd chkconfig --add ndbd service ndbd start`
23. ☐	Restart all the VM's of the NDB cluster	Restart all the VM's of the NDB cluster. Once all the VM's are up and available then the NDB Cluster should be working fine. But make sure management node's are started first before data nodes and SQL nodes. It would be good to follow the order of VM's restart, **Management nodes → Data Nodes → SQL Nodes**. Verify that cluster is up and running as per specified in the next step.
24. ☐	Check the other node status from NDB management	Check the status of other nodes from the NDB management node, by executing: `ndb_mgm> show`
25.	Create tables on	Create tables on `NDBCluster SQL node1` as below:

STEP#	Procedure	Description

☐	NDBCluster SQL node1	SSH to SQL Node VM as root and scp the provided schema (`gatekeeper.sql`) file to the current folder location.
		
		Enter the command:
		
		`mysql -u <new mysql user> -p gatekeeper < schemafile`
		
		Provide the new mysql user password at prompt.
		
		Example:
		`mysql -u mysqluser -p gatekeeper < gatekeeper.sql`
		
		The `mysqluser` in the above example has to be replaced with appropriate new user created for mysql.
		
		Verify that all tables are created.

26.	Create stored procedure on all SQL nodes	Create the required stored procedure on all SQL nodes as below:
		
		`mysql -u mysqluser -p gatekeeper < scef_apn_rate_control_sp.sql`
		
		`mysql -u mysqluser -p gatekeeper < scef_me_groupreport_gt_delete.sql`
		
		`mysql -u mysqluser -p gatekeeper < scef_me_groupreport_gt_get.sql`
		
		`mysql -u mysqluser -p gatekeeper < scef_me_groupreport_gt_update.sql`
		
		Note: Copy the `gatekeeper.sql` and `scef_apn_rate_control_sp.sql` to Database SQL Nodes that are available in `Admin/AppServer` VMs under the location `/u02/app/oracle/scripts/.`
		
		Copy the `gatekeeper.sql`, `scef_apn_rate_control_sp.sql`, `scef_me_groupreport_gt_get.sql` and `scef_me_groupreport_gt_update.sql` to Database SQL Nodes that are available in `Admin/AppServerVMs` under the location `/u02/app/oracle/scripts/.`

27.	Grant required permission	SSH to all SQL nodes as root and grant the required permissions, by executing:
		
		`mysql -u root -p`
		`GRANT SELECT ON mysql.proc TO 'mysqluser'@'%';`

28.	Remove the file `auto.cnf` and restart all the SQL nodes	SSH to all SQL nodes to remove the file `/var/lib/mysql/auto.cnf` by executing below command and restart all SQL node VMs : `rm -f /var/lib/mysql/auto.cnf`

Note: It is good to take regular backups of the database at least once in a week on regular basis. Refer to section 3 of *DSR API Gateway Disaster Recovery Guide* for the detailed steps. Follow the approach mentioned in Section 3.1.1 section. For manual approach, refer to section 3.1.2.

7. Configure DSR APIGW Admin/Application Server

Procedure 9. Configure DSR APIGW Admin/Application Server

STEP #	Procedure	Description
This procedure describes how to install and configure DSR APIGW Admin and Application Servers. **Prerequisite**: All the respective infrastructures must be up and running. Check off (√) each step as it is completed. Boxes have been provided for this purpose under each step number. If this procedure fails, contact My ORACLE Support (MOS) and ask for assistance.		
1 ☐	Copy the .pem file (key-pair) used to create the VMs to Admin server in any location.	1. Log in to the Openstack controller console. 2. Copy the pem file from the opentack controller to the Admin server in any location by executing the following command: `$ scp -i /root/dsr-keypair.pem /root/ dsr-keypair.pem admusr@<aminserverip>:/u02` **Note**: PEM certificates are frequently used for web servers as they can easily be translated into readable data using a simple text editor. When a PEM encoded file is opened in a text editor, it contains different headers and footers. For creating a PEM file, refer to Appendix C.
2 ☐	Log in to the Admin server and fill in the `ocsg.properties` file with all required input data for the script.	1. Log in to the **Admin** server. 2. Navigate to ***/u02/app/oracle/scripts/*** `$ cd /u02/app/oracle/scripts/` 3. Edit the `ocsg.properties` file to add respective property values in the file. For information about the properties and its parameters, refer to Appendix B.

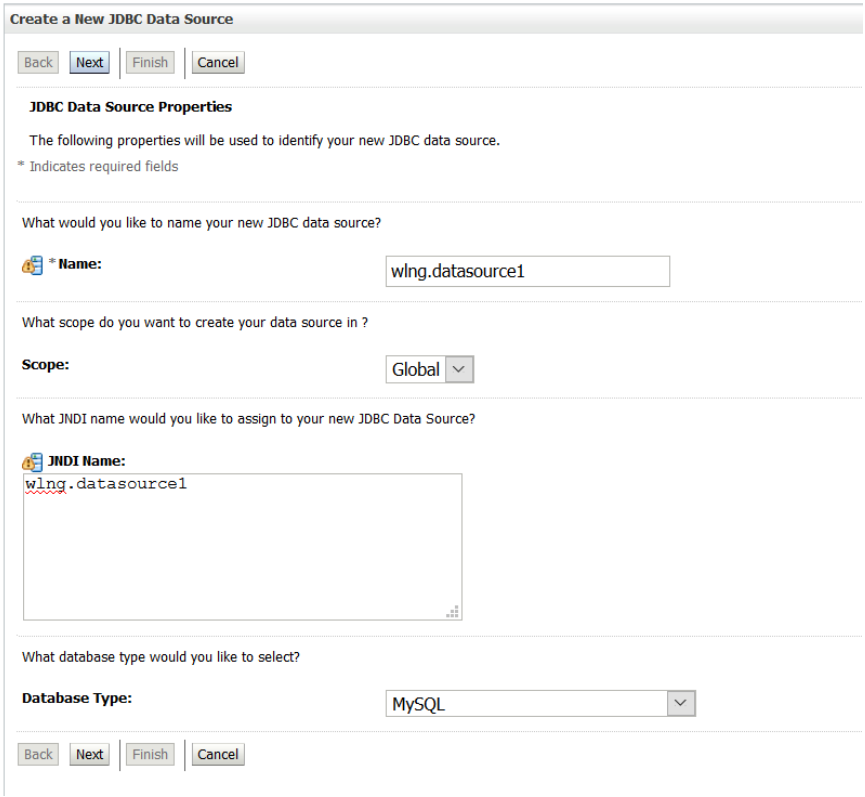
3
☐
	Modify database details in `dsrapigw_default_params.rsp`	1. Log in to the **Admin** server and all **App servers**.
		2. Navigate to `/u02/app/oracle/scripts/`
		`$ cd /u02/app/oracle/scripts/`
		3. Edit the `dsrapigw_default_params.rsp` file and modify the following parameters:
		`DATABASE_USER_NAME=<provide mysql user created while configuring database. Ex:mysqluser>`
		`DATABASE_USER_PASSWORD=<provide password of above mentioned user>`
		`DATABASE_HOST_NAME=<ipaddress of sqlnode1>`
		`DATABASE_PORT=3306`

4 ☐	Execute the script	Execute the script from the Admin server: 1. Log in to the **Admin** server. 2. Navigate to `/u02/app/oracle/scripts` 3. Execute the command: `python configureOCSGSingleTier.py`
5 ☐	Monitor the screen or verify the log file for completion	From the Admin server, verify the logs as follows: 1. Log in to the **Admin** server. 2. Navigate to `/u02/app/oracle/scripts` 3. Execute the command: `vim ocsg_install.log` **Note**: The log file name is configured in the `ocsg.properties` file. Installation takes a few minutes to complete, and a message "`DSR APIGW Configuration Successful`" is displayed after the completion.

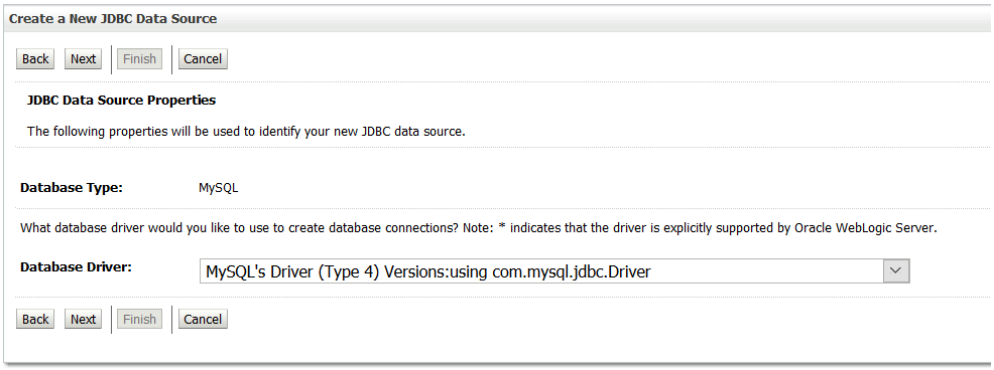
STEP #	Procedure	Description
6 ☐	Verify the interface accessibility	Verify the interface accessibility by opening the GUI Interface IP in a browser window. For the port information, refer to the Network model.
7 ☐	Creation of Multi data source	See Multi Data Source Creation to create multi data sources.

8. Multi Data Source Creation

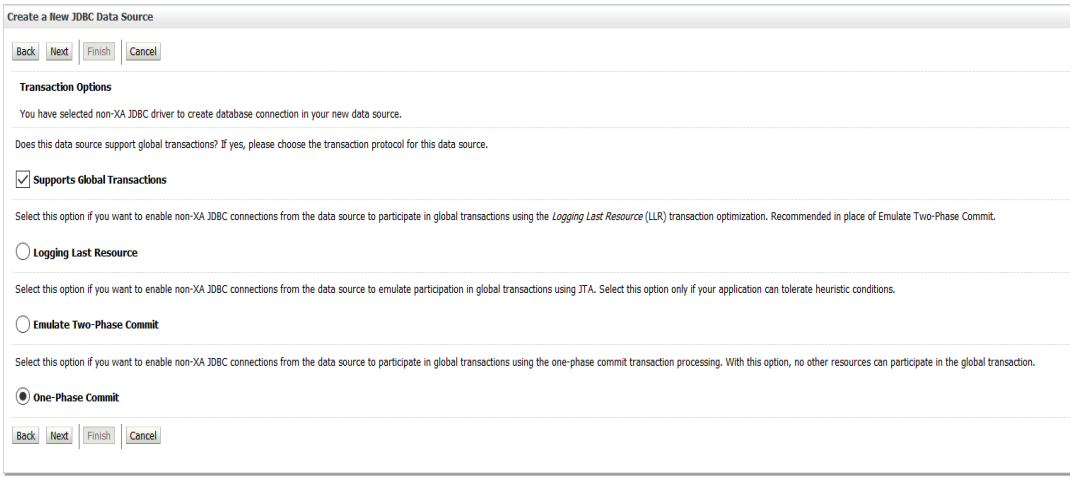
1. Shutdown all the App servers.
2. Click **Lock & Edit** option on the left-hand side of the **Change Center** section in the Admin (Weblogic) console.
3. Navigate to Data source page, **Domain Structure** → **Services** → **Data Sources**.
4. Create data source named `wlng.datasource1` by clicking on **New** in the section "**Data Sources** (Filtered - More Columns Exist)" and select the option "**Generic Data Source**". Provide **Name**, **JNDI Name** & select **Database Type** as shown in the screen below and click **Next**.



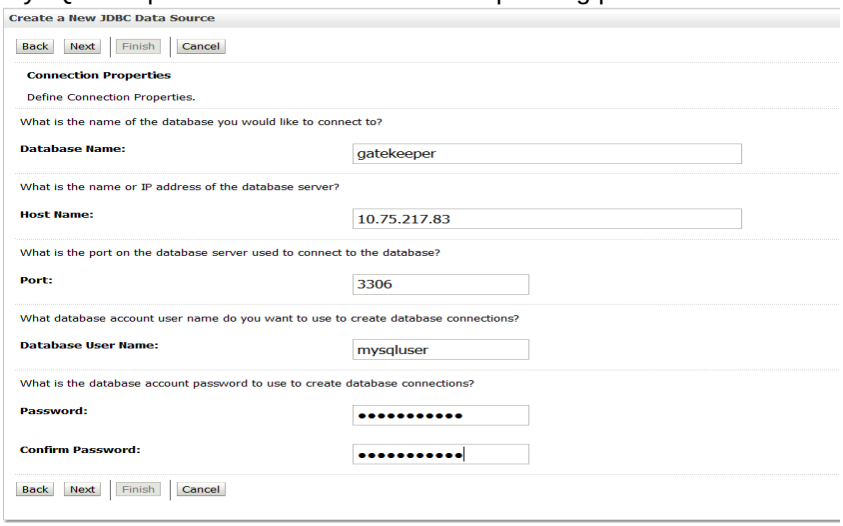
5. Select **Database Driver** as shown in the screen below and click **Next**:



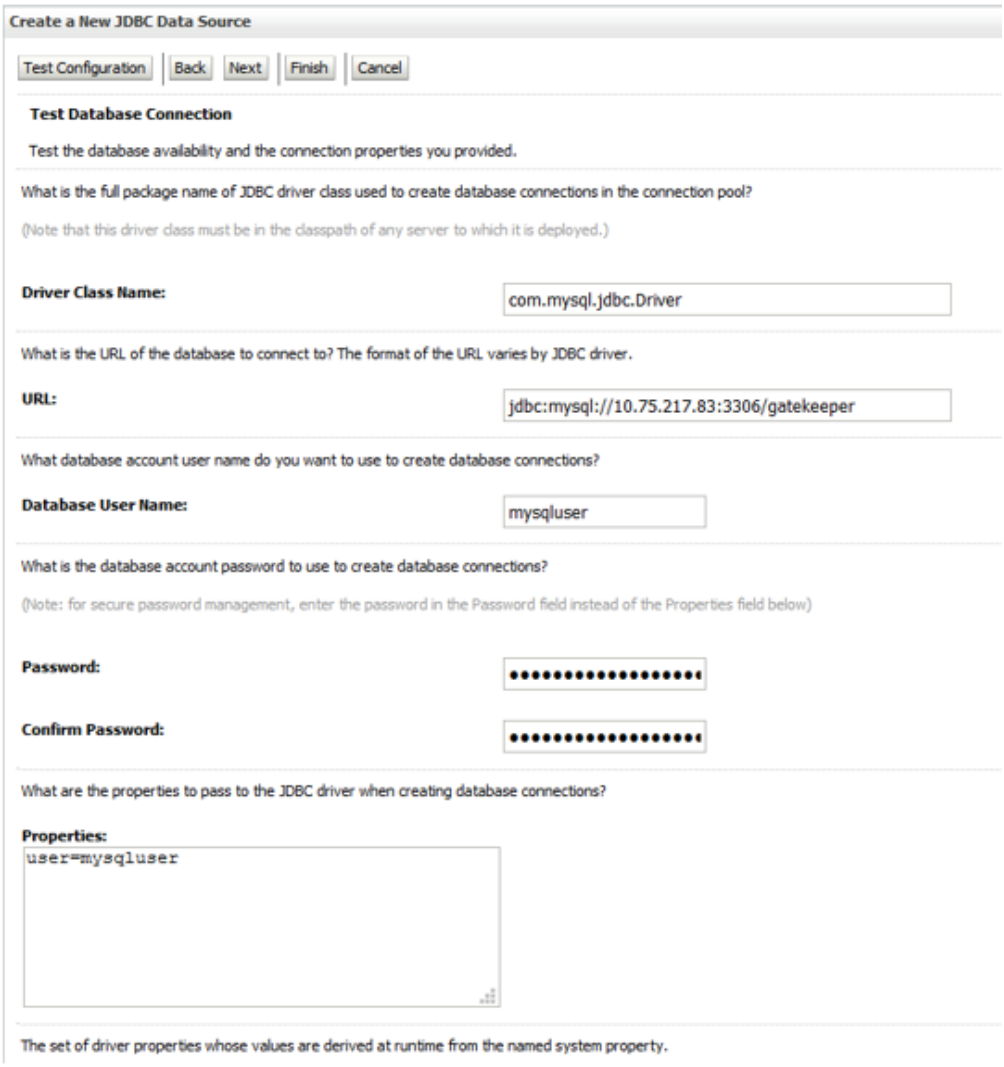
6. Check the box against "**Supports Global Transactions**", and the **One-phase Commit** option would be auto selected, then click **Next**.



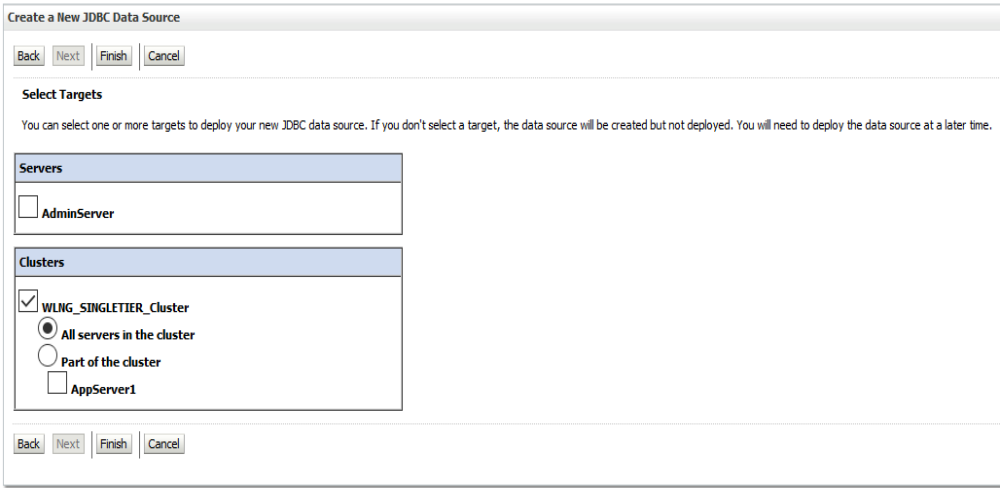
7. Provide **Database Name** as "`gatekeeper`", **Host Name** as "`Ip of SQL Node1`", **Port** as "3306", **Database User Name** is the new user name created while configuring SQL on SQL node1 during MySQL setup creation. Provide the corresponding password and click **Next**.



8. In this step, modify the URL as `jdbc:mysql://ipaddress:3306/gatekeeper?useUnicode=yes&characterEncoding=UTF-8` i.e., add "`?useUnicode=yes&characterEncoding=UTF-8`" at the end of existing url. The remaining can be left as is and test the connection by clicking on **Test Configuration** option, which should show connection is fine. Once connection is tested and found to be fine. Click **Next**.

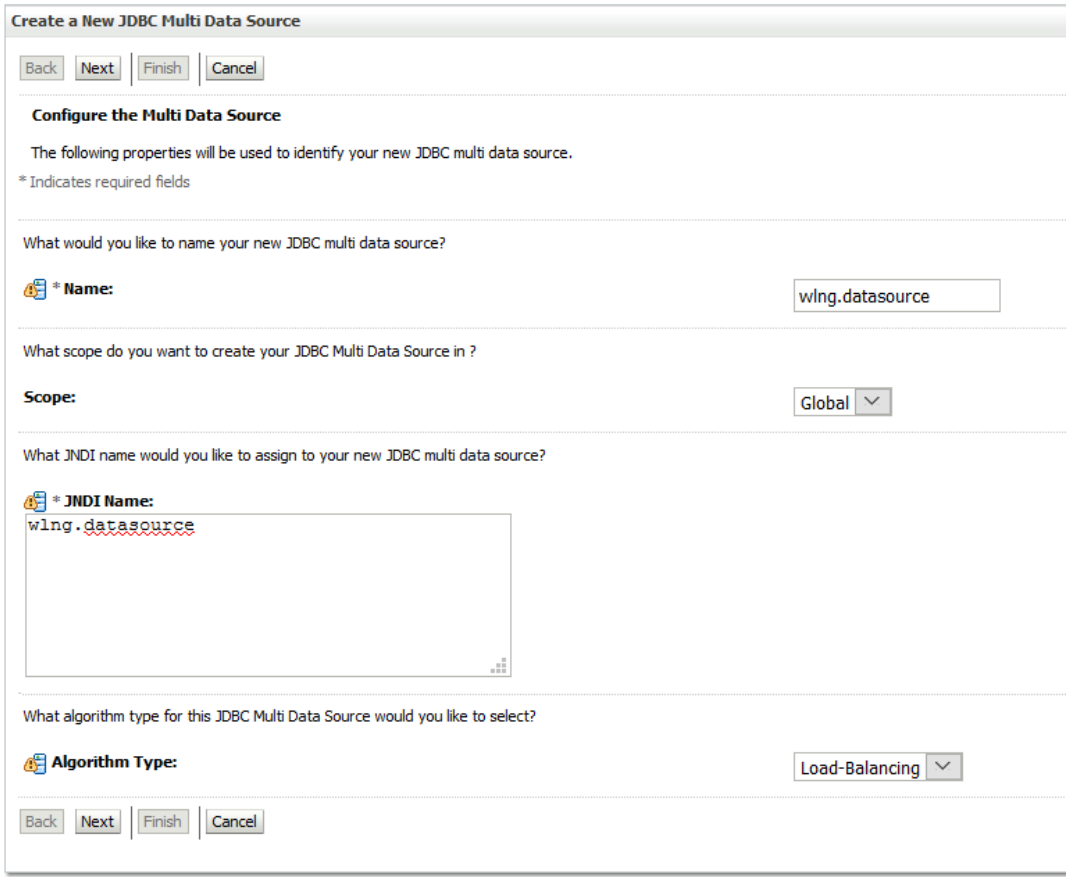


9. Check infront of `WLNG_SINGLETIER_Cluster` as shown in the screen below and click **Finish**.

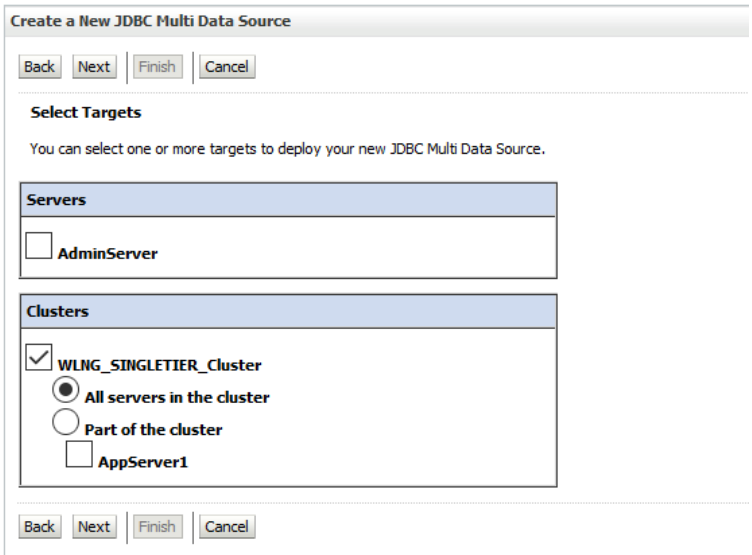


10. Similarly create datasource "`wlng.datasource2`" by repeating steps followed to create `wlng.datasource1`. The `wlng.datasource2` needs to be provided with SQL node2 details. If more datasources are required, please repeat the steps followed to create `wlng.datasource1` and provide corresponding SQL node details.
11. Delete the existing datasource "`wlng.datasource`" as we have to create multi datasource with the same name.
    To delete datasource "`wlng.datasource`":
    a. Navigate to **Domain Structure** → **Services** → **Data Sources**
    b. Check the box against this datasource and click **Delete** in the section "Data Sources (Filtered - More Columns Exist)".
    c. Click **Yes** when prompted for re-confirmation.

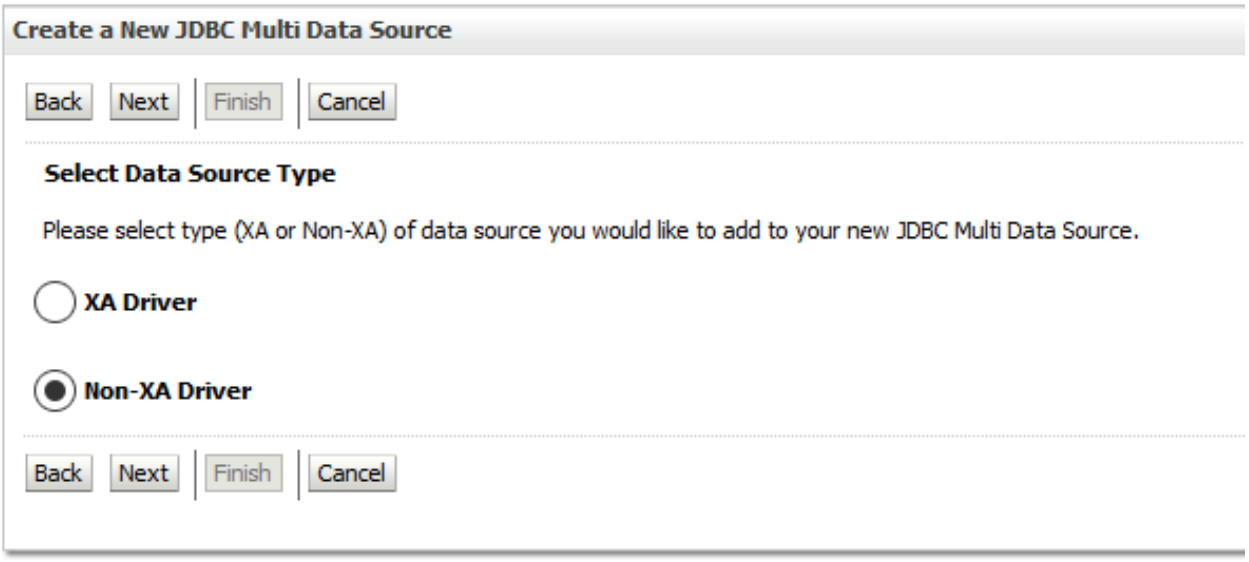
12. Click **New** in the "Data Sources (Filtered - More Columns Exist)" section to create new multi datasource by selecting the option "**Mutli Data Source**". Provide **Name**, **JNDI Name** and **Algorithm Type**, as shown in the screen below and click **Next**:



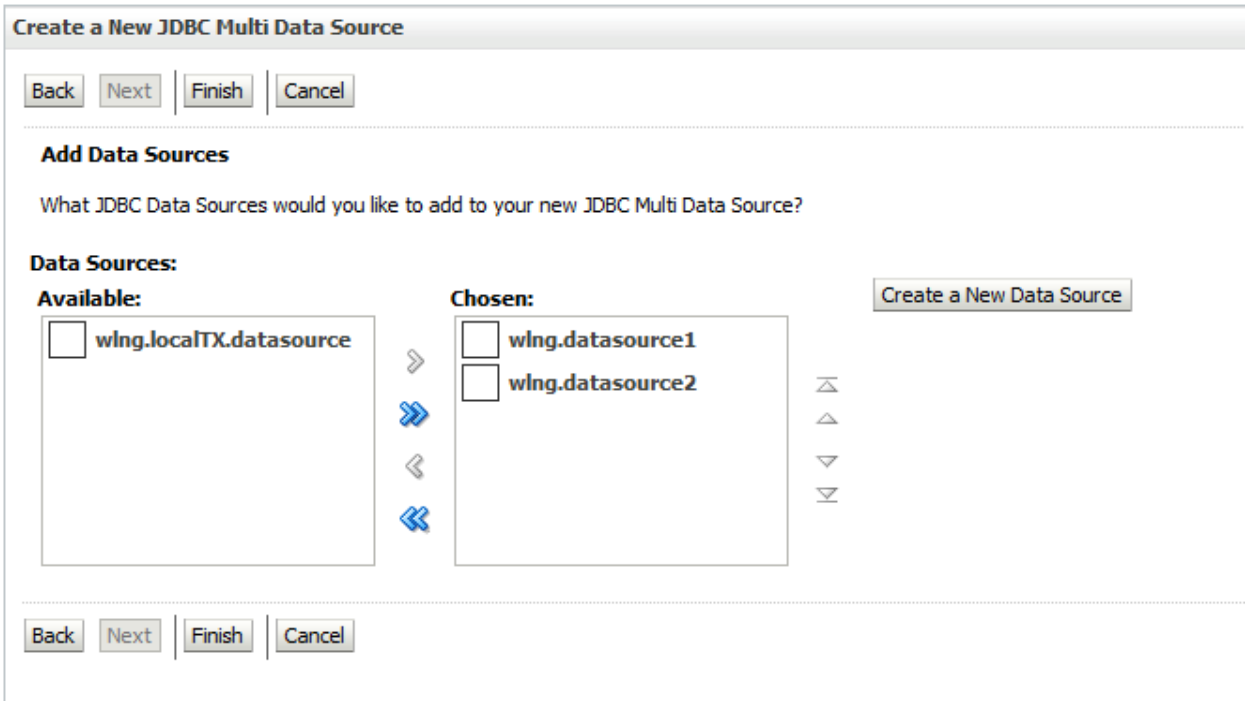
13. Check against "`WLNG_SINGLETIER_Cluster`" as shown in screen below and click **Next**.



14. Choose **Non-XA Driver** as shown in the screen below and click **Next**.

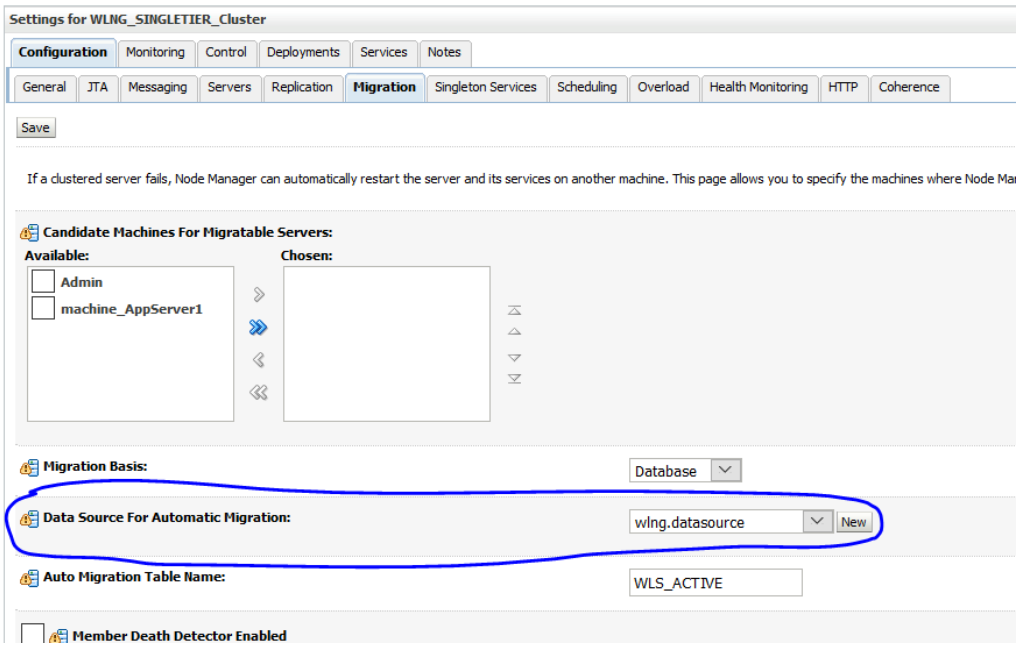


15. The `wlng.datasource1` and `wlng.datasource2` will be available under "**Available**" area, move them under "**Chosen**" area as shown in the screen below and click **Finish**.

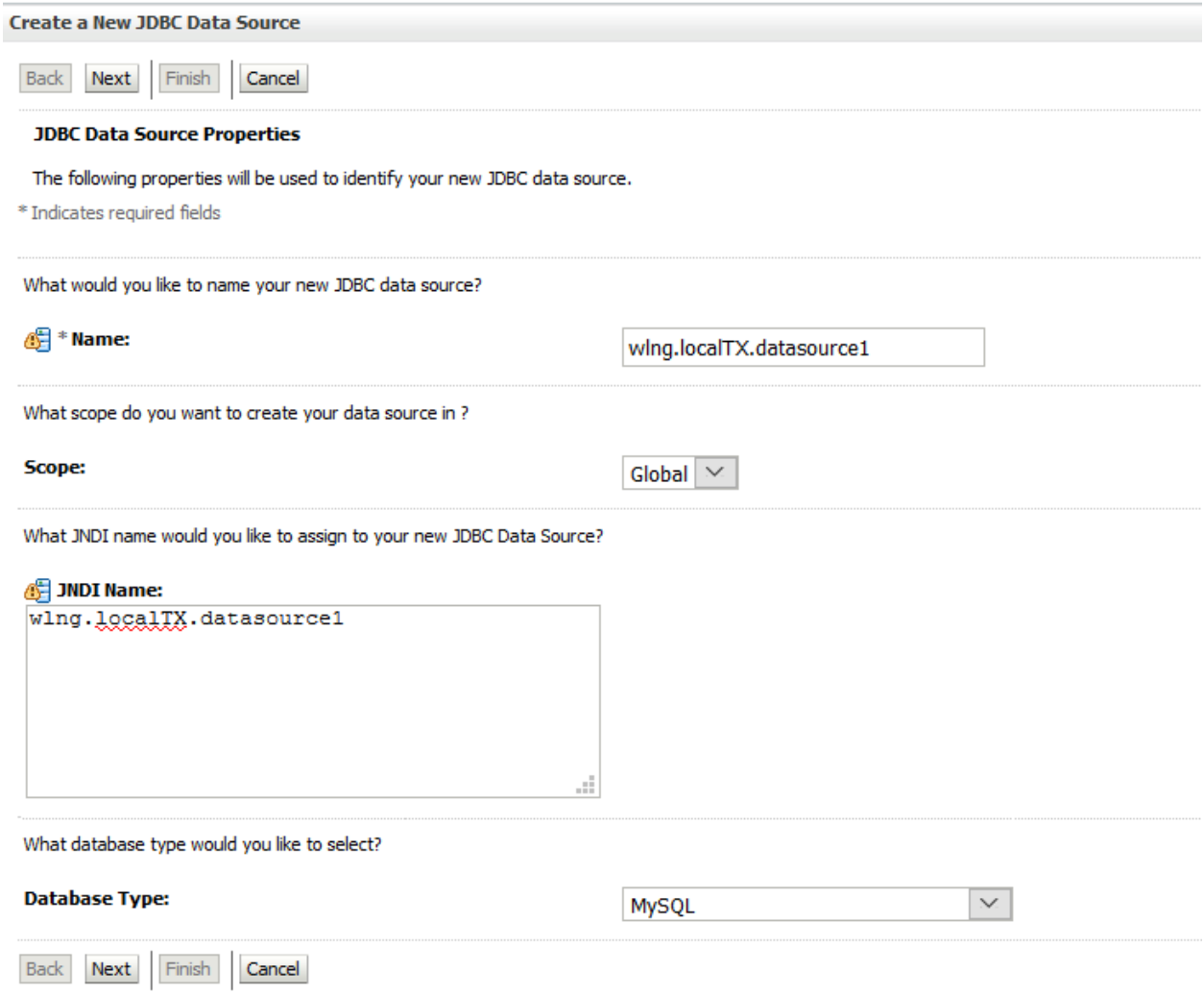


Note: There will be `wlng.datasource`**1,** `wlng.datasource`**2**, and so on. If there are more datasources then they would also need to be moved under **Chosen** section.

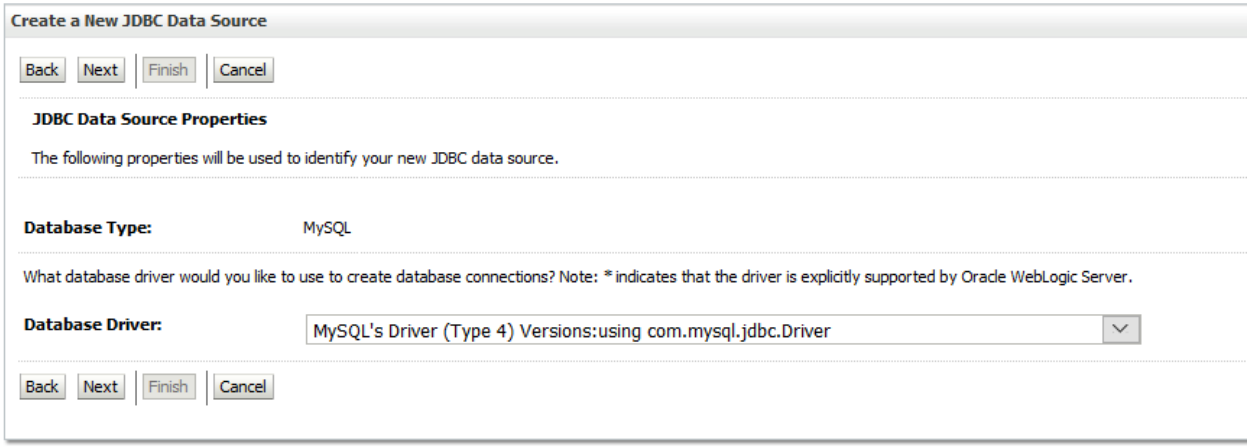
16. Navigate to **DomainStructure** → **Environment** → **Clusters.**
    a. Click on **WLNG_SINGLETIER_Cluster**
    b. Navigate to **Migration** tab under **Configuration** tab.
    c. Change **"Data Source For Automatic Migration:"** to `wlng.datasource` as shown in the screen below and click **Save**.



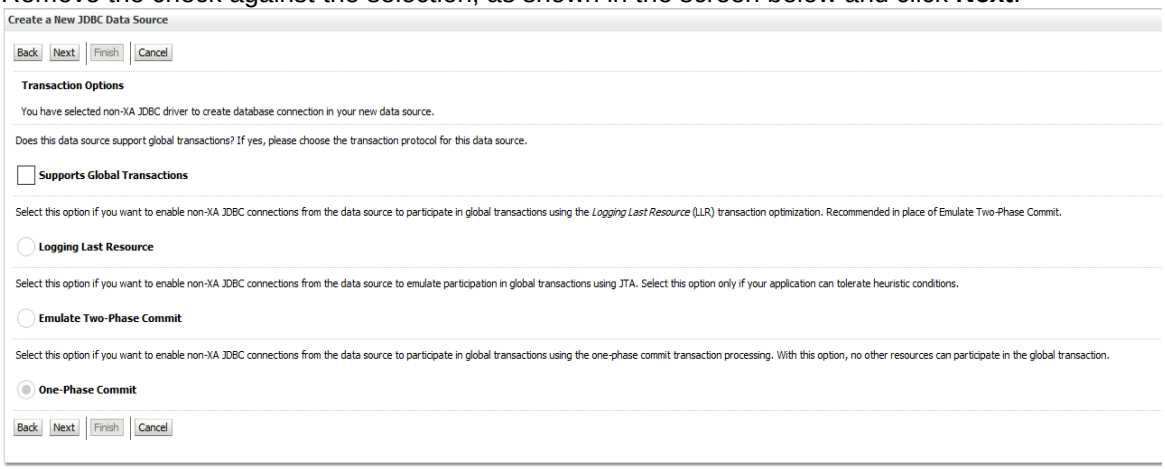
17. Now navigate back to **Data Sources** page, click **New** and select "**Generic Data Source**". Provide **Name**, **JNDI Name & Database Type** as shown in the screen below and click **Next**.



18. Select Database Driver as shown in the screen below:

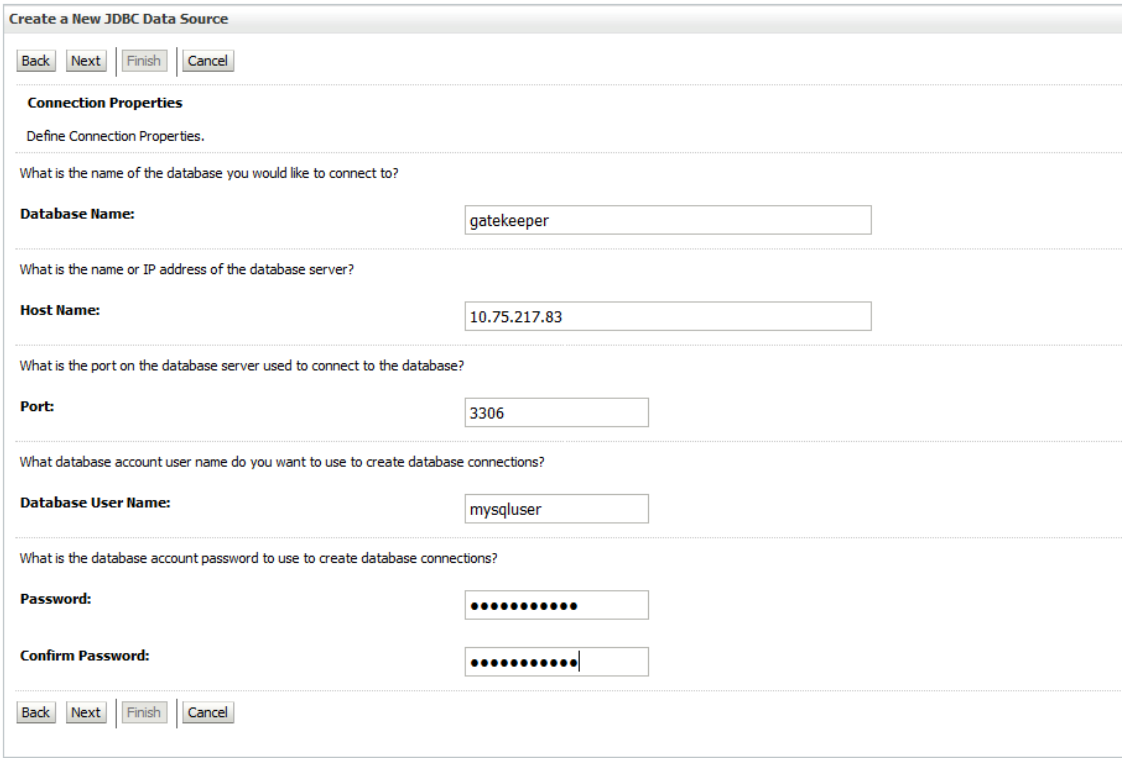


19. Remove the check against the selection, as shown in the screen below and click **Next**.



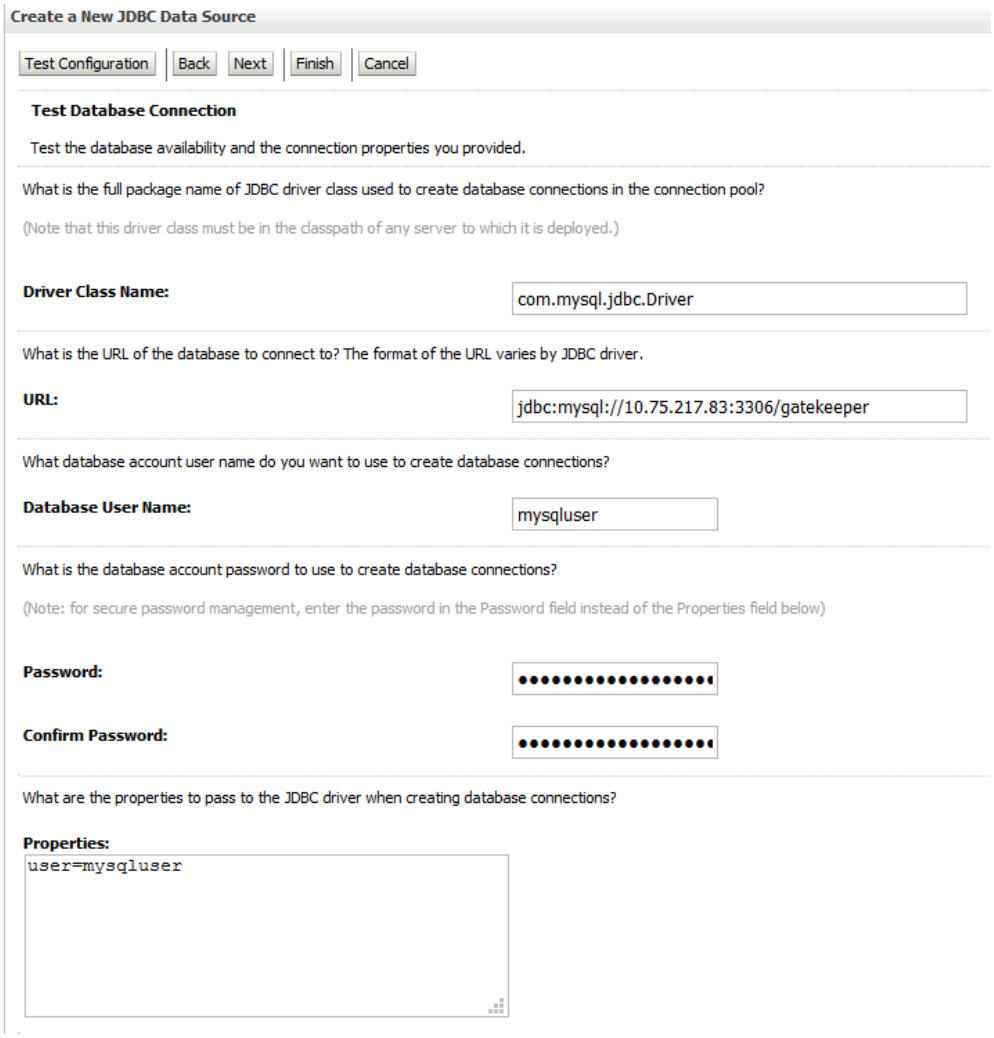
Note: The checkbox against "**Supports Global Transactions**" is not checked.

20. Provide **Database Name** as "gatekeeper", **Host Name** as "Ip of SQL Node1", Port as "3306", **Database User Name** is the new user name created while configuring SQL on SQL node1 during MySQL setup creation. Provide the corresponding password and click **Next**.



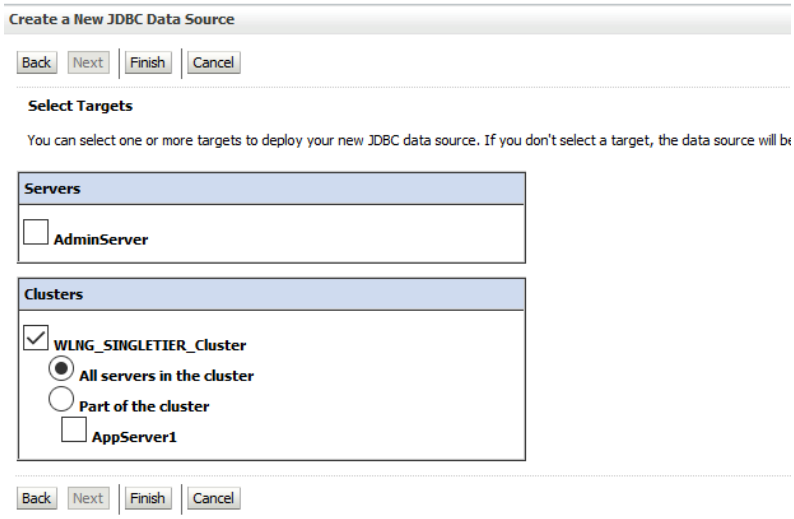
21. In this step, modify URL to look as
`jdbc:mysql://ipaddress:3306/gatekeeper?useUnicode=yes&characterEncoding=UT`

F-8 i.e., after correcting ipaddress of SQL node1, add "?useUnicode=yes&characterEncoding=UTF-8" at the end of url. The remaining can be left as is. Test the connection by clicking on **Test Configuration** that would show the connection is fine. Once connection is tested and found to be fine. Click **Next**.



Do not miss to update the URL in the above step with jdbc:mysql://ipaddress:3306/gatekeeper?useUnicode=yes&characterEncoding=UTF-8, otherwise cyrillic cannot be supported.

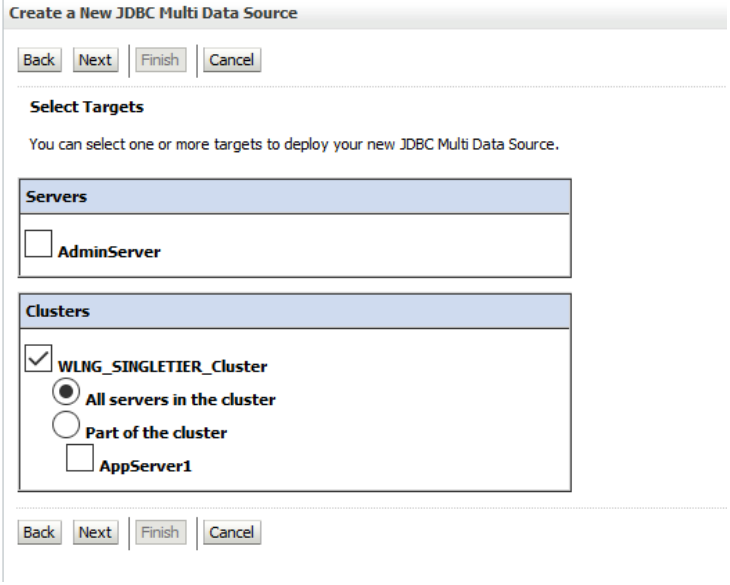
22. Check against "WLNG SINGLETIER Cluster" as shown in the screen below and click **Next**.



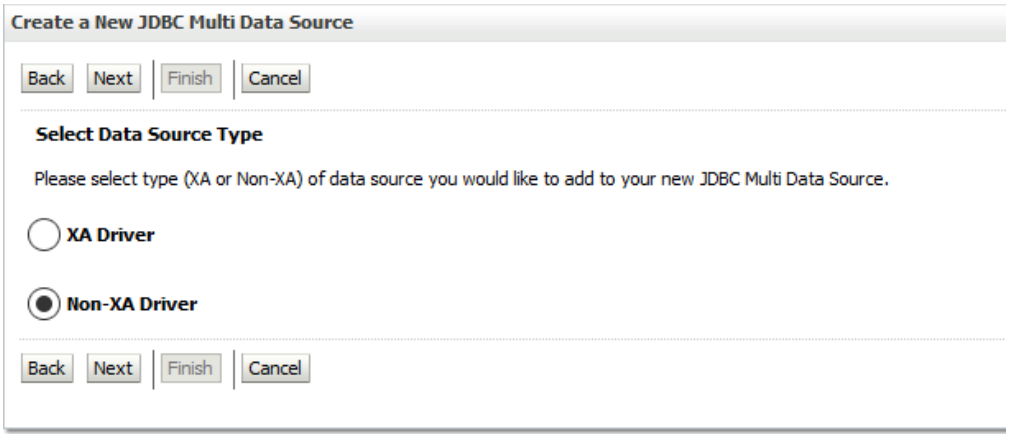
23. Similarly create another data source named "wlng.localTX.datasource2" by repeating steps followed to create "wlng.localTX.datasource1". If more data sources are required, repeat the

steps followed to create `wlng.datasource1` and provide corresponding SQL node details. The `wlng.localTX.datasource2` should be provided with SQL node 2 details.

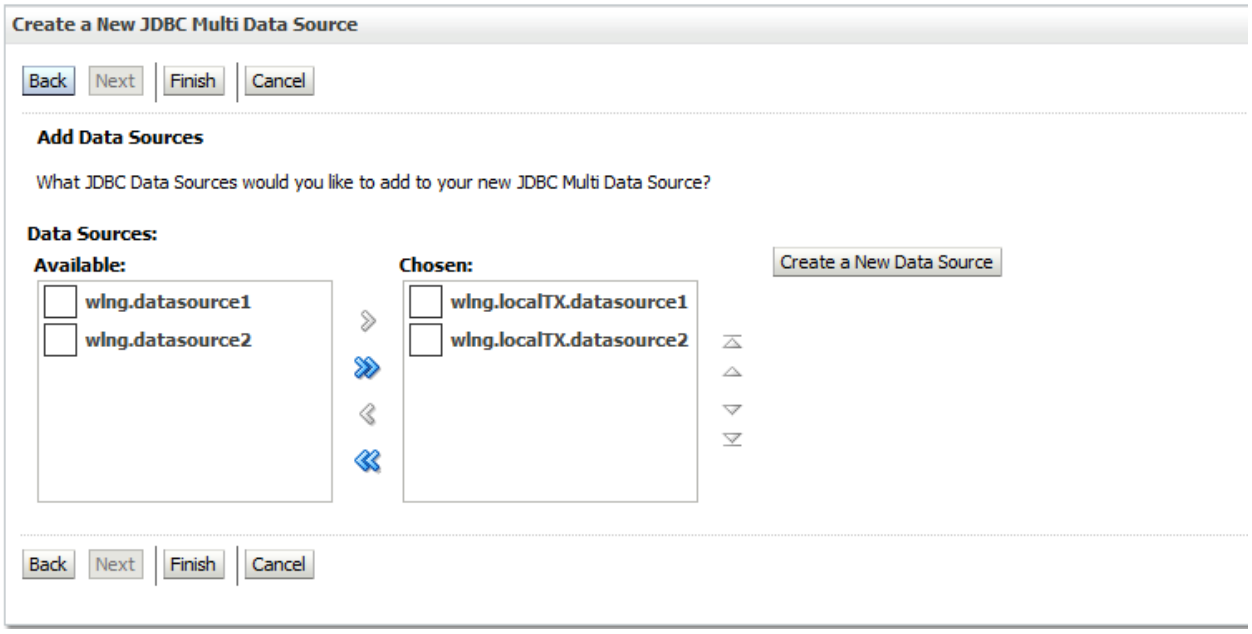
24. Delete the existing datasource named "`wlng.localTX.datasource`" to create multi data source with the same name. To delete datasource "`wlng.localTX.datasource`", check the box against this datasource and click **Delete** in the section "**Data Sources** (Filtered - More Columns Exist)" on the data source. Click **Yes** when asked for re-confirmation.
25. On Data Source page, click **New** and select "**Multi Data Source**" to create data source named "`wlng.localTX.datasource`".
    Name: `wlng.localTX.datasource`
    JNDI Name: `wlng.localTX.datasource`
    Algorithm Type: Failover
    Check against "**WLNG_SINGLETIER_Cluster**" as shown in the screen below and click **Next**.



26. Choose Non-XA Driver as shown in the screen below and click **Next**.

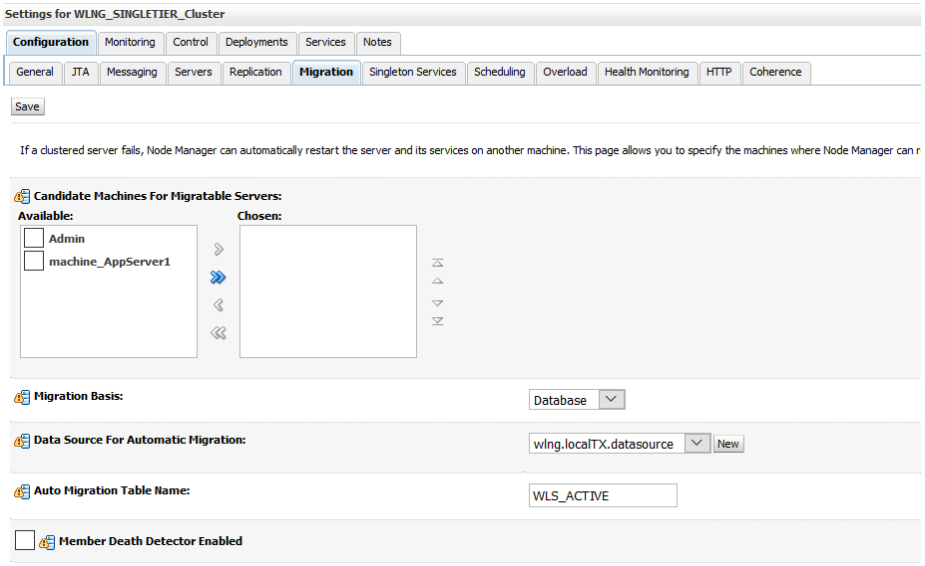


27. The `wlng.localTX.datasource1` and `wlng.localTX.datasource2` will be available under "**Available**", and must be moved under "**Chosen**" as shown in the screen below, then click **Finish**.



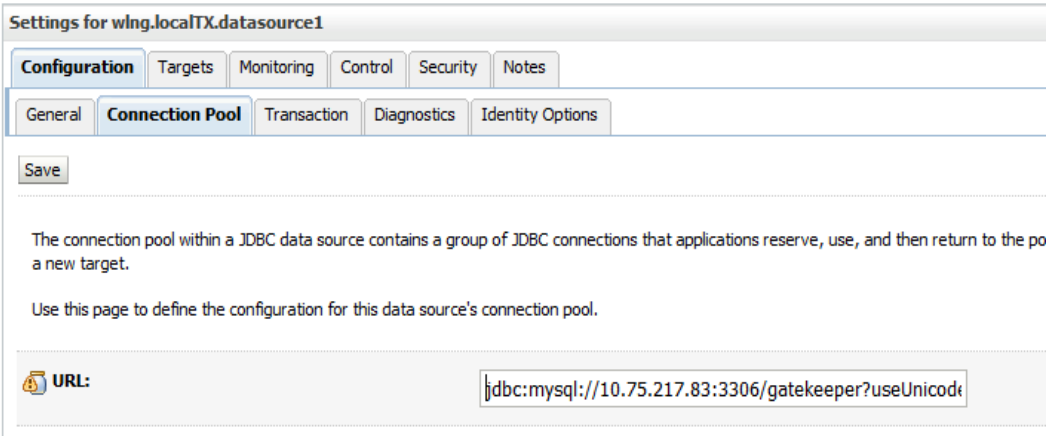
Note: There will be `wlng.localTX.datasource1`, `wlng.localTX.datasource1`, and so on. If there are more datasources then they also need to be moved under **Chosen** section.

28. Navigate to **DomainStructure** → **Environment** → **Clusters**
    a. Click `WLNG_SINGLETIER_Cluster`
    b. Navigate to **Migration** under **Configuration** tab.
    c. Change "**Data Source For Automatic Migration:**" to **wlng.localTX.datasource** as shown in the screen below and click **Save**.



29. Navigate to datasource page, make sure to verify that JDBC url is in the format "`jdbc:mysql://ipaddress:3306/gatekeeper?useUnicode=yes&characterEncoding=UTF-8`" in all datasources. This can be verified by navigating to datasource page and click on datasources (`wlng.datasource1, wlng.datasource2, wlng.localTX.datasource1, wlng.localTX.datasource2`, etc).
30. Navigate to **Connection Pool** under **Configuration** tab (for each datasource). Change the **Maximum capacity** to 40 for all `wlng.datasourceX` and for `wlng.localTX.datasourceX` it should be 75.
    a. Change value of "**Statement Cache Size:**" from 10 to 200.
    b. Click **Save**.
    c. Click **Advanced** at the bottom of the page and make the following changes:
        i. Check the checkbox against "**Test Connections On Reserve**"
        ii. Change "**Seconds to Trust an Idle Pool Connection:**" to **10**
        iii. Change "**Test Frequency**" value from 120 to 5
        iv. Change "**Connection Count of Refresh Failures Till Disable:**" to **1**
        v. Change "**Count of Test Failures Till Flush:**" to **1**

vi. The changes suggested above must be repeated on all the available data sources.



31. After performing all the above steps, click on **Activate Changes** button on the left side of the **Change Center** section to commit the new changes.

Start all the Appservers and verify that OCSG is up and running on all Appservers without any issue.

Appendix A. Example Parameter file

A.1. Guidelines to create parameter file

Basic guidelines to follow while working with YAML files:
9. The file must be ended with .yaml extension.

10. YAML must be case-sensitive and indentation-sensitive.

11. YAML does not support the use of tabs. Instead of tabs, it uses spaces.

YAML is a human-friendly data serialization standard for all programming languages.
The values of the **key:value** can be broadly classified into the following types:

Type	Description	Examples
string	A literal string.	“String param”
number	An integer or float.	“2”; “0.2”
comma_delimited_list	An array of literal strings that are separated by commas. The total number of strings should be one more than the total number of commas.	[“one”, “two”]; “one, two”; ***Note***: “one, two” returns [“one”, ” two”]
json	A JSON-formatted map or list.	{“key”: “value”}
boolean	Boolean type value, which can be equal “t”, “true”, “on”, “y”, “yes”, or “1” for true value and “f”, “false”, “off”, “n”, “no”, or “0” for false value.	“on”; “n”

A.2. Parameter file for DSR APIGW Database

The parameter file defines the topology details. This includes all VM details such as the number of VMs, flavors, network names, etc. It is a list of key/value pairs. By referring to the **parameters** definition section in the template file, the initialization of the parameters has to be done in this section.

File Naming Convention

It is not mandatory to have a specific name for the file; but just to provide a self-explanatory name for the file, it is recommended to follow this convention:

<DSR Name>_<Site Name>_<NetworkOam >_Params.yaml

For example:

- dsrCloudInit_Site00_NetworkOam_Params.yaml

Sample File

Network OAM params file

```
parameters:
    numPrimaryNoams: 1
    numNoams: 1
    noamImage: DSR-60147
    noamFlavor: dsr.noam
    primaryNoamVm Names: ["DsrSite00NOAM00"]
    noamVm Names: ["DsrSite00NOAM01"]
    noamAZ: nova
    xmiPublicNetwork: ext-net
    imiPrivateNetwork: imi
    imiPrivateSubnet: imi-sub
    im iPrivateSubnetCidr: 192.168.321.0/24
    ntpServer: 10.250.32.10
    noamSG: Site00_NOAM_SG
```

Network OAM params file (Fixed IP)

```
parameters:
   numPrimaryNoams: 1
   numNoams: 1
   noamImage: DSR-8.5.0.0.0_90.10.0.vmdk
   noamFlavor: dsr.noam
   primaryNoamVm Names: ["DsrSite00NOAM00"]
   noamVm Names: ["DsrSite00NOAM01"]
   noamAZ: nova
   primaryNoamXmiIps: ["10.196.12.83"]
```

noamXmiIps: ["10.196.12.84"]
noamVip: 10.196.12.85
xmiPublicNetwork: ext-net3
imiPrivateNetwork: imi
imiPrivateSubnet: imi-sub
im iPrivateSubnetCidr: 192.168.321.0/24
ntpServer: 10.75.185.194
noamSG: Site00_NOAM_SG

A.3. Parameter file for DSR APIGW Admin/Application

Following are the HEAT template files:

- dsrapigw.yml
- dsrapigw_env.yml

Following is the list of parameters used to configure DSR APIGW Admin/Application stack.

Table 4: Example Parameter file

Parameter category	Parameter Name	Type	Description
Common parameters	key_name	String	Name of key-pair to be used for compute instance
	image_id	String	Oracle Linux image to be used for compute instance
Number of VMs	num_app	Number	Number of AT servers to be confirgured as per the requirement
VM flavors	flavor_admin	String	Admin server VM profile
	flavor_app	String	AT server VM profile
IP Network	networks_admin	Json	List of networks (one or more) on admin server
	networks_app	Json	List of networks (one or more) on application server
hostname	hostname_admin	String	Hostname of the admin server
	user_name	String	User name of the admin server
	password	String	Password fo the admin server

Appendix B. OCSG Properties file

Following table lists the user data to be filled in OCSG properties file.

Table 2: OCSG Properties file

Section	Parameter Name	Description
Admin	servers	Add Admin server name and IP. For example: **servers** = ["AdminServer:xxx.xxx.xxx.xxx"] **Note**: It is recommend to follow the name of Admin server as 'Adm inServer' Example: # servers = *["AdminServer:11.11.11.11"]* IMI Interface address servers = ["AdminServer: xxx.xxx.xxx.xxx "] XMI Interface address xmiInterfaces = ["AdminServer: xxx.xxx.xxx.xxx "]

	backupServers	This is the DSRAPIGW DB server address where data is backed up. DR procedure will use this data.
		# Provide the Ipaddress of SQL node1.
		# Admin server should have access to this server using the key/pem file.
		backupServer = xxx.xxx.xxx.xxx
		# This is the location in the DSRAPIGW DB server where the data should be backed up.

App	servers	Add App server name and IP. Add comma seperated entries for multiple servers. For example, servers = ["AppServer1:xxx.xxx.xxx.xxx", "AppServer2:xxx.xxx.xxx.xxx"] **Note**: It is mandatory to follow the name of App servers as 'AppServer1', 'AppServer2' etc.
App	xmiInterfaces	XMI Interface address for all AppServers in ["Ip1","Ip2"...] format. For example, xmiInterfaces = ["AppServer1: xxx.xxx.xxx.xxx ", "AppServer2: xxx.xxx.xxx.xxx "]

Section	Parameter Name	Description

App	`xsiInterfaces`	XSI Interface address for all AppServers in ["Ip1","Ip2"...] format.
		For example,
		`xsiInterfaces = ["AppServer1: xxx.xxx.xxx.xxx ", "AppServer2: xxx.xxx.xxx.xxx "]`
		`To add multiple XSIs to each AppServer the format should be,`
		`["AppServer1:XSI1-IP","AppServer2:XSI2","AppServer2:XSI1-IP","AppServer2:XSI2"]`

App	`exteralLoadbalancerIP`	IP used to publish T8 APIs. This IP will be used when displaying T8 API access URLs in Partner and API management Portal. `exteralLoadbalancerIP =` ***xxx.xxx.xxx.xxx***
Servers	`cleanUpBeforeInstall`	If the script failed to execute while running, the server will be in a bad shape for a fresh install. Keeping cleanUpBeforeInstall as "yes" will clean up the server and make it ready for script re-run.
Servers	`ntp`	Provide NTP server IP `ntp` = ***xxx.xxx.xxx.xxx***
Servers	`mtu`	Maximum transmission unit. The script copies multiple files from Admin server to App server. Before copying the MTU has to be set. Recommended value is "9000". `mtu` = *9000*
Servers	`apiroot`	This variable is part of the API creation. <apiroot> is prefixed to the context uri of the APIs exposed. # For example, the API name of Device triggering is "apiroot-dt"
Servers	`dsrMpList`	Provice DSR MP XSI Ip list in format, MP1-XSI-IP:port,MP2-XSI1-IP:port.....
Files	`pemfile`	Provide the .pem file location. `pemfile` = */u02/software/ocsg-db-key.pem*
Files	`logfile`	Custom log file for Installation. Change log file name if required. `logfile` = *ocsg_install.log*

Section	Parameter Name	Description
Files	presentFolder	The scripts will be present in this location. This property should not be changed presentFolder = */u02*
Files	targetFolder	The scripts will be copied to this location. This property should not be changed targetFolder = */u03*
Files	targetPath	Provide the location of the scripts. This property should not be changed targetPath = */app/oracle/*
Files	scripts	Provide the folder name where scripts need to be stored. This property should not be changed. scripts = *scripts*
Files	extendWizard	Custom scripts will be present here. This property should not be changed. extendWizard = *extend_wizard/*
Files	SCEFPackage_EAR	Default EAR file name. This property should not be changed. SCEFPackage_EAR = *SCEFHandlers.ear*
Files	nodemgr	Node manager service file name. This property should not be changed nodemgr = *nodemgr*
Files	DefaultJar	Location of ocsg_generic_jar. This property should not be changed defaultJar = */usr/TKLC/dsrapigw/ocsg_generic_jar*
Files	volumeName	Provide the Volume name, This property should not be changed volumeName = *ocsgv*
Files	volumeSize	Volume size in GB. Script woll create a new volume of this size. This field should not be changed volumeSize = 10
Files	inventoryLoc	Inventory log location of OCSG. This property should not be changed inventoryLoc = */u02/inventory*
Files	cdr_targetFolder	Target folder cdrs cdr_targetFolder = /u04
Files	cdr_volumeSize	cdr Volume size in GB. Script will create a new volume of this size cdr_volumeSize = 5
Files	cdr_volumeName	Provide the Volume name

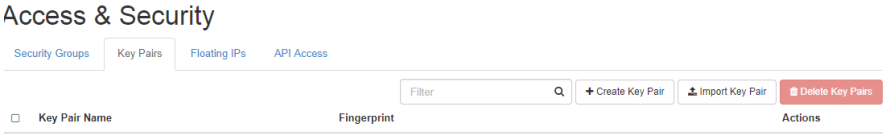
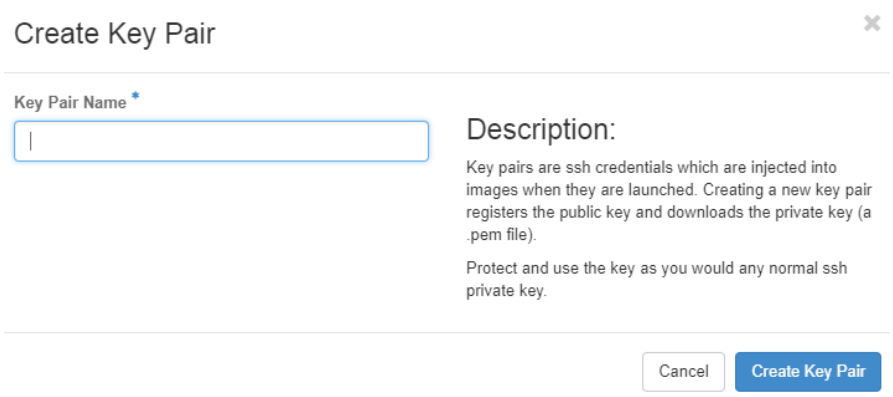
Section	Parameter Name	Description
		cdr_volumeName = cdrv
Files	customslaxsd	CustomSLA XSD Definition file name customslaxsd = customslaxsd.xsd
Credentials	mysqlJdbcServerUrl	MySQL DB credentials. Provide IP address of the DSR API GW database setup SQL Node1. jdbc:mysql://<db-server-ip>:3306/gatekeeper For Example, mysqlJdbcServerUrl = jdbc:mysql://30.30.30.17:3306/gatekeeper
Credentials	mysqlUserName	**mysqlUserName** = *ocsg_auto* Note: MySQL credentials to be updated in dsrapigw_default_params.rsp file available in the location /u02/app/oracle/scripts on all VM's of Admin & Appservers. Refer point# 3 of section 7. Configure DSR APIGW Admin/Application Server" in this document for more details.
Credentials	weblogicUser	Provide the DSR API GW Admin portal credentials. weblogicUser = *weblogic* weblogicPassword = *tekelec123*
Credentials	weblogicPassword	
Credentials	nodeManagerUser	Provide the Nodemanager credentials which will be used in all Admin and AppServers nodeManagerUser = *nodemanager* nodeManagerPassword = *tekelec123*
Credentials	nodeManagerPassword	
Credentials	operatorUser	A new operator will be created with this details to access partner relationship management portal. operatorUser = *oracleop3* operatorPassword = *tekelec123*
Credentials	operatorPassword	
Credentials	adminServerUser	Below is the ssh user name in Admin and AppServers adminServerUser = *admusr* appServerUser = *admusr*
Credentials	appServerUser	
Ports	adminListenPort appListenPort appListenPortSSL	These are the default ports opened on IMI network should not be changed, these ports are used only for internal communication adminListenPort = *7001* appListenPort = *8001* appListenPortSSL = *8002*
Ports	adminIMIPorts adminXMIPorts	Ports to be enabled in IP Firewall on Admin server:

Section	Parameter Name	Description
		`adminIMIPorts` = 7001,5556,7002,9876,8050,3075,9090,7 `adminXMIPorts` = 9002

Ports	`appIMIPorts`
	`appXMIPorts`
	`appXSIPorts`
		Ports to be enabled in IP Firewall on AppServers:
		`appIMIPorts` = 8001,8002,9876,5556,8050,3075,9090,7
		`appXMIPorts` = 9002
		`appXSIPorts` = 10001,10002,1883,5685,5656,3868

Appendix C. Create PEM file for Openstack

Procedure 10. Create PEM File for openstack

STEP #	Procedure	Description
This procedure describes how to create PEM file for openstack. **Prerequisite**: All the respective infrastructures has to be up and running Check off (√) each step as it is completed. Boxes have been provided for this purpose under each step number. Contact My ORACLE Support (MOS) and ask for assistance.		
1 ☐	Login to Openstack	Login to the Openstack.
2 ☐	Go to Create KeyPair option	1. Navigate to Project -> Compute -> Access & Security - > Key Pairs 2. Click Create Key Pair button 
3 ☐	Create Key Pair	Enter the required Key Pair Name and click Create Key Pair. 

Appendix D. Create PEM file for VM

Procedure 11. Create PEM File for VM

STEP #	Procedure	Description
This procedure describes how to create PEM file for VM. **Prerequisite**: All the respective infrastructures has to be up and running Check off (√) each step as it is completed. Boxes have been provided for this purpose under each step number. Contact My ORACLE Support (MOS) and ask for assistance.		
1 ☐	**VMware client:** Login to client	Open the VMware client
2 ☐	**Admin Server**: Create Keys	Execute `ssh-keygen` It will create public and private keys on the Admin server. **Note**: Do not provide any input.
3 ☐	**Admin Server**: Copy id_rsa	Copy the id_rsa to a pem file on Admin server: `cp /home/admusr/.ssh/id_rsa /u02/key.pem`
4 ☐	**Admin Server**: Install Keys	Install the keys on App Server by running command: `ssh-copy-id admusr@<IP of AppServer>` Execute command on Admin server (repeat for all App servers).
5 ☐	**Admin Server**: Copy ssh keys	Check from Admin server if you are able to login to App Server without password by running command. `ssh -i /u02/key.pem admusr@@<IP of AppServer >` You need to copy the ssh keys to all App servers. Once done, edit the **ocsg.properties** file to point to correct pem file.

Appendix E. Resource Profile for DSR APIGW Database and Admin/Application

Following table provides list of resource provide for DSR APIGW Database, Admin and Application servers.

Table 5: Resource Profile for DSR APIGW

DSR APIGW	vCPU	RAM (GB)	Disk (GB)	Network Interfaces
Admin Server	4	6	70	2
Application Server	12	16	70	3*
Management Node (DB)	4	6	70	
Data Node (DB)	12	64	200	
SQL Node (DB)	8	16	70	

***Note**: Multiple XSI Network interfaces are supported for App servers. Maximum 16 network XSI interfaces are supported.

Appendix F. Convert vmdk to qcom2 format

STEP #	Procedure	Description
This procedure describes how to convert vmdk to qcom2 format. **Prerequisite**: All the respective infrastructures has to be up and running. Check off (√) each step as it is completed. Boxes have been provided for this purpose under each step number. Contact My ORACLE Support (MOS) ask for assistance.		
1 ☐	Login to Qemu-img tool	Login to the Qemu-img tool.

2
☐
	Convert the file format	Convert vmdk to qcow2 format
		
		Use the qemu-img tool to create a qcow2 image file using this command:
		
		`qemu-img convert -f vmdk -O qcow2 <VMDK filename> <QCOW2 filename>`
		
		Example:
		
		`qemu-img convert -f vmdk -O qcow2 DSR-82_12_0.vmdk DSR-82_12_0.qcow2`
		
		Note: Install the qemu-img tool (if not already installed) using this yum command:
		`sudo yum install qemu-img`
		
		Import the coverted qcow2 image using the “glance” utility from the command line.
		`$ glance image-create --name dsr-x.x.x-original --is-public True --is-protected False --progress --container-format bare --disk-format qcow2 --file DSR-x.x.x-disk1.qcow2`

Appendix G. Sample Network Element and Hardware Profiles

To enter all the network information for a network element into an AppWorks-based system, a specially formatted XML file needs to be filled out with the required network information. The network information is needed to configure both the NOAM and any SOAM network elements.
It is expected that the maintainer/creator of this file has networking knowledge of this product and the customer site at which it is being installed. The following is an example of a network element XML file.
The SOAM network element XML file needs to have same network names for the networks as the NOAM network element XML file has. It is easy to accidentally create different network names for NOAM and SOAM network elements, and then the mapping of services to networks are not possible.

```
<?xml version="1.0"?>
<networkelement>
    <name>NE</name>
    <networks>
        <network>
            <name>XMI</name>
            <vlanId>3</vlanId>
            <ip>10.2.0.0</ip>
            <mask>255.255.255.0</mask>
            <gateway>10.2.0.1</gateway>
            <isDefault>true</isDefault>
        </network>
        <network>
            <name>IMI</name>
            <vlanId>4</vlanId>
            <ip>10.3.0.0</ip>
            <mask>255.255.255.0</mask>
            <nonRoutable>true</nonRoutable>
        </network>
    </networks>
</networkelement>
```

Note:NetworkElement Name shall be unique while creating multiple Network Element.

Appendix H. 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, there are multiple layers of menus selections. Make the selections in the sequence shown below on the Support telephone menu:

1) For the first set of menu options, select 2, “New Service Request”. You will hear another set of menu options.
2) In this set of menu options, select 3, “Hardware, Networking and Solaris Operating System Support”. A third set of menu options begins.
3) In the third set of options, select 2, “ Non-technical issue”. Then you will be connected to a live agent who can assist you with MOS registration and provide Support. Identifiers. Simply mention you are a Tekelec Customer new to MOS.