Oracle RAC on KVM Hypervisor Virtualized by Unified Compute Platform

Implementation Guide

MK-SL-260-00
January 2023

Feedback

Hitachi Vantara welcomes your feedback. Please share your thoughts by sending an email message to SolutionLab@HitachiVantara.com. To assist the routing of this message, use the paper number in the subject and the title of this white paper in the text.

Revision history

Changes	Date
Initial release	January 20, 2023

Chapter 1: Introduction and overview

This guide provides comprehensive steps to design and implement Hitachi Solution for databases for Oracle Real Application Clusters Virtualized on Oracle Kernel-based Virtual Machine (KVM) Hypervisor with Hitachi Advanced Server DS220 G2 servers and Hitachi Virtual Storage Platform E1090 storage systems.

Walk through the planning and deployment of an on-premises environment with an Oracle KVM hypervisor as the foundation. This design uses an Oracle KVM Hypervisor on the host as a virtualization technology. This solution includes configuring storage, configuring the network, and best practices for designing and streamlining the environment using Oracle Linux Virtualization Manager (OLVM).

Because the Oracle KVM hypervisor virtualization technique is used, the native operating system on the server machine is Oracle Linux 8. In this guide, two DS220 G2 bare metal servers are connected to VSP E1090 over a Fibre Channel network. The storage area network is accessible by both hosts, and zoning configured on switches to allow LUN access to only dedicated servers.

The two bare metal servers are configured with Oracle Linux 8.6. On top of it, Oracle KVM hypervisor software is installed for host virtualization. OLVM is used as a management server to install, configure, and manage all the virtualization resources such as network, storage, and VMs across the KVM hosts and logical volume management (LVM) VMs.

Virtualization benefits

Virtualization technology includes the following benefits:

- Reduces the overhead of purchasing multiple servers and managing them.
- Minimizes infrastructure and software licensing costs.
- Transfers between VMs and LUNs can be easily migrated from one physical device to another.
- Simplifies backup of the VM with encapsulation.
- Uses different configurations of physical servers for hardware platform independence.
- Allows effective use of resources with enhanced utilization.
- Lowers RPO and RTO.

Intended audience

This guide is designed for technical professionals who are looking for end-to-end installation and configuration of Oracle RAC database over virtualized UCP solutions provided by Hitachi Vantara. They should be proficient in Oracle database architecture and administration, and have experience working with servers, networking, and storage.

Note: These procedures were developed in a lab environment. Many factors affect production environments beyond prediction or duplication in a lab environment. Follow recommended practice by conducting proof-of-concept testing for acceptable results before implementing this solution in your production environment. Test the implementation in a non- production, isolated test environment that otherwise matches your production environment.

Chapter 2: Solution components

The following table lists the hardware components used in this implementation.

Vendor	Hardware	Description	Version	Quantity
Hitachi Vantara	Hitachi Virtual Storage Platform E1090	6 × CHA pairs (8 × 32 Gbps Fibre Channel ports in use) 1024 GB cache memory 48 × 1.9 TB NVMe SDDs	93-06-01-80/00	1
Hitachi Vantara	Hitachi Advanced Server DS220 G2	2 × Intel Xeon Platinum 8368 38C CPU @ 2.40 GHz 768 GB (64 GB ×12) DIMM DDR5 Synchronous Registered (Buffered) 3200 MHz	BIOS: S5XH3A12.H03 BMC: 3.16.06 CPLD: 07	2
		2 × Intel E810 dual port 25 GbE NIC cards	Driver: ice Driver Version:0.8.2-k Firmware: 2.42	
		2 × Emulex LightPulse LPe35002-M2 2 Port 32 Gb Fibre Channel Adapter	Driver: lpfc Driver Version: 12.8.0.10 Firmware: 12.8.542.26	
Hitachi Vantara	Hitachi Advanced Server DS120 G2	2 × Intel Xeon Processors 4310, 12-core, 2.10 GHz, 120W 256 GB (32 GB ×8) DIMM DDR4-3200 Synchronous Registered	BIOS: S5XH3A12.H03 BMC: 3.16.06 CPLD: 07	2

Vendor	Hardware	Description	Version	Quantity
		(Buffered) 3200 MHz 1 × 256 GB NVMe 0.3DWPD M.2 SSD for boot		
		1 × Dual Port 25 GbE NIC Intel E810 PCIe card	Driver Version:1.8.1.6 Firmware: 7.30	
		1 × Emulex LightPulse LPe35002-M2 2-Port 32 Gb Fibre Channel Adapter	Driver: lpfc Driver Version:12.8.0.10 Firmware:12.8.542.26	
Brocade	G720 Fibre Channel switches	48 × 32 Gbps ports Fibre Channel switch 32 Gbps SFPs	Kernel: 2.6.34.6 Fabric OS: v9.0.1c	2
Cisco	Nexus 93180YC-FX	48 × 10/25 GbE port 6 × 40/100 Gbps Quad SFP (QSFP28) ports	BIOS version: 07.65 NXOS version: 9.3.7	2
	Cisco-C92348GC-X	1 GE 48-Port Gb Ethernet Switch	BIOS version: 5.37 NXOS version: 9.3.7	1

The following table lists the minimum hardware requirements.

Server Number	List	Details
1	Server Hardware Configuration	16 Gb of RAM Memory
		Modern Intel/AMD x86_64 CPU
		80 GB of disk space
2	KVM Virtual Machine requirement	2 vCPUs
		4 Gb of RAM
		50 Gb hard disk
3	OLVM management host	2 vCPUs
		6 Gb RAM, 30 Gb hard disk

Note: These hardware components and software versions were used in a lab environment. This may vary in a production environment.

Chapter 3: Server and application architecture

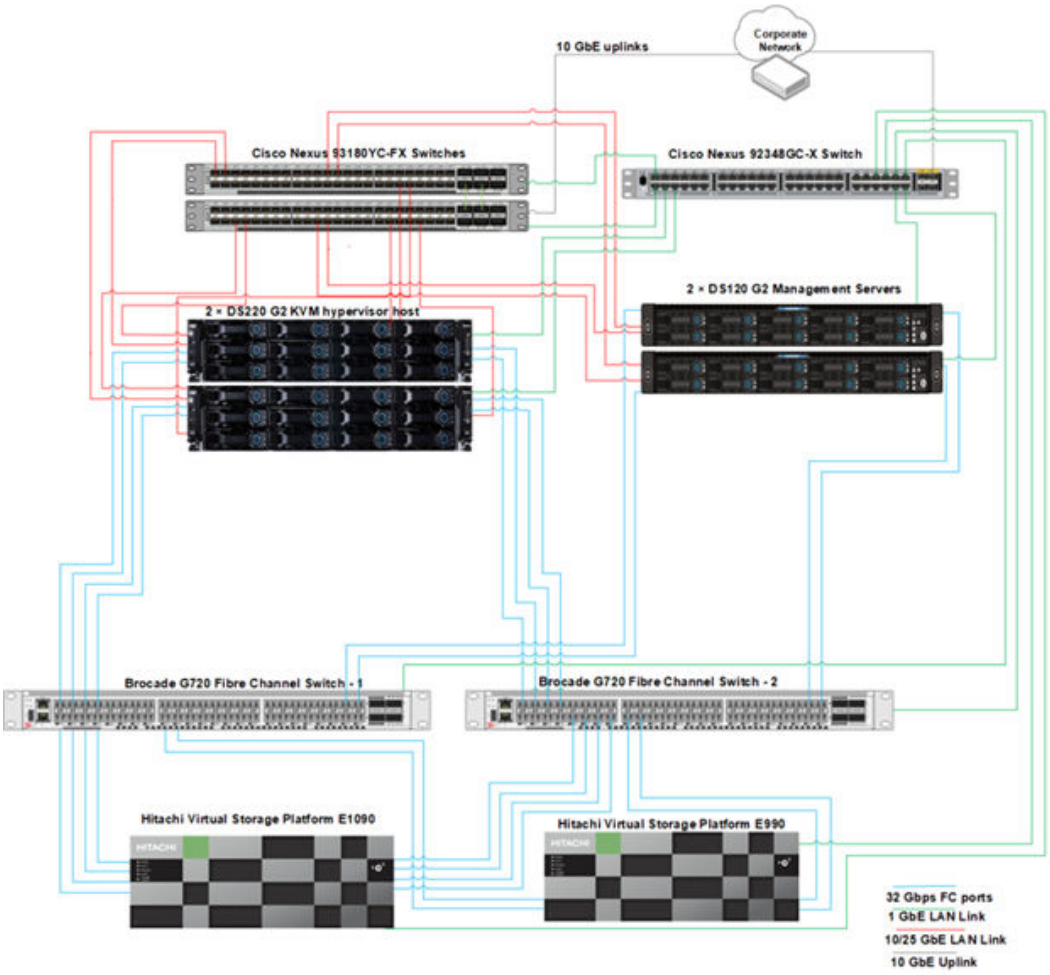
This implementation uses two Hitachi Advanced Server DS220 G2 servers as compute nodes and two Hitachi Advanced Server DS120 G2 servers as management nodes for installation of a two-node Oracle RAC database cluster using a KVM hypervisor on a virtualized platform. This provides the compute power for the Oracle RAC database to handle complex database queries and a large volume of transaction processing in parallel. The following table lists a summary of the server configuration for this solution.

Hitachi Advanced Server	Server	Server Name	Role	CPU Core	RAM
Bare Metal host 1 DS220 G2	KVM hypervisor VM1	rac01	Oracle RAC node 1	36	768 GB (64 GB × 12)
Bare Metal host 2 DS220 G2	KVM hypervisor VM2	rac02	Oracle RAC node 2	36	768 GB (64 GB × 12)
VM host 3 DS120 G2	Management server Oracle Linux Virtual Management (OLVM)	Olvm-host	Manager for KVM hypervisor	18	256 GB (32 GB × 8)

DS220 G2 servers are configured with the following:

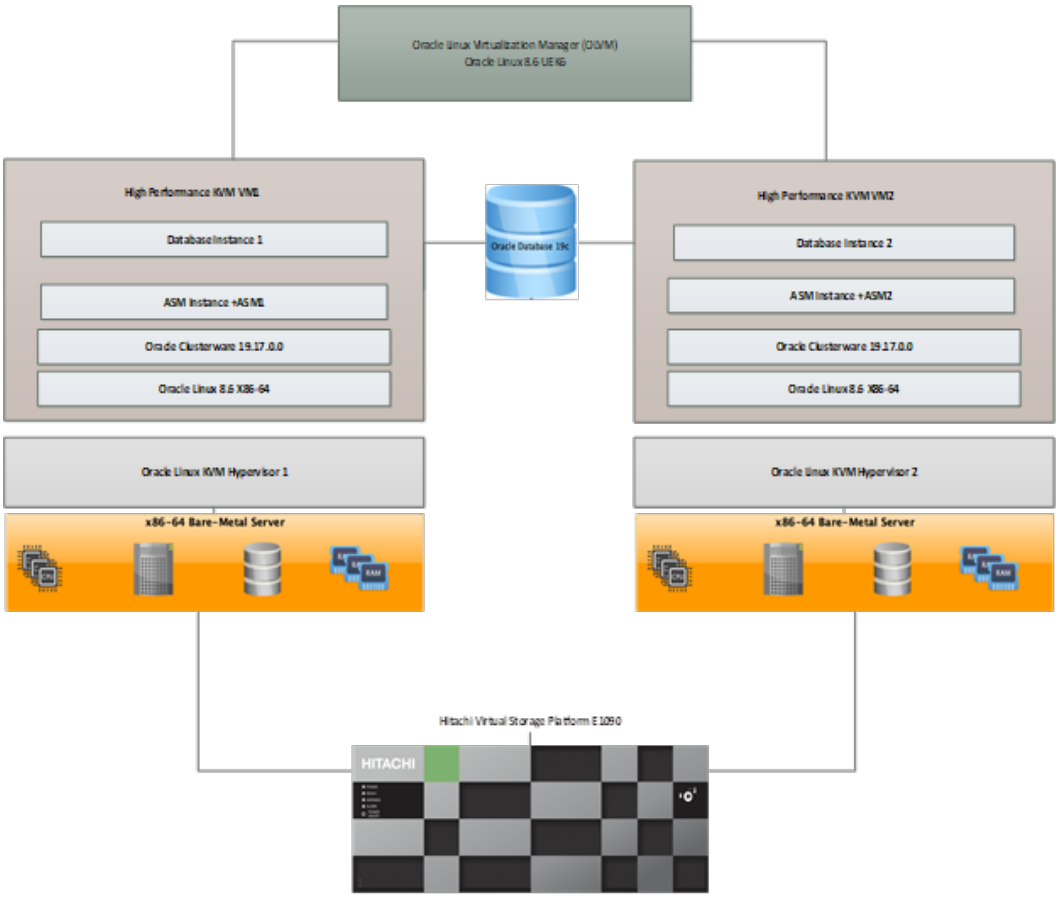
- Fully redundant hardware
- Dual fabric connectivity between hosts and storage

The following illustration shows the high-level architecture diagram using Hitachi Virtual Storage Platform E1090 and Hitachi Advanced Server DS220 G2 for a 2-Node Oracle 19c RAC configuration.



Note: Management servers are not used in this implementation, they are shown for reference only.

The following diagram shows the OLVM flowchart for this implementation.



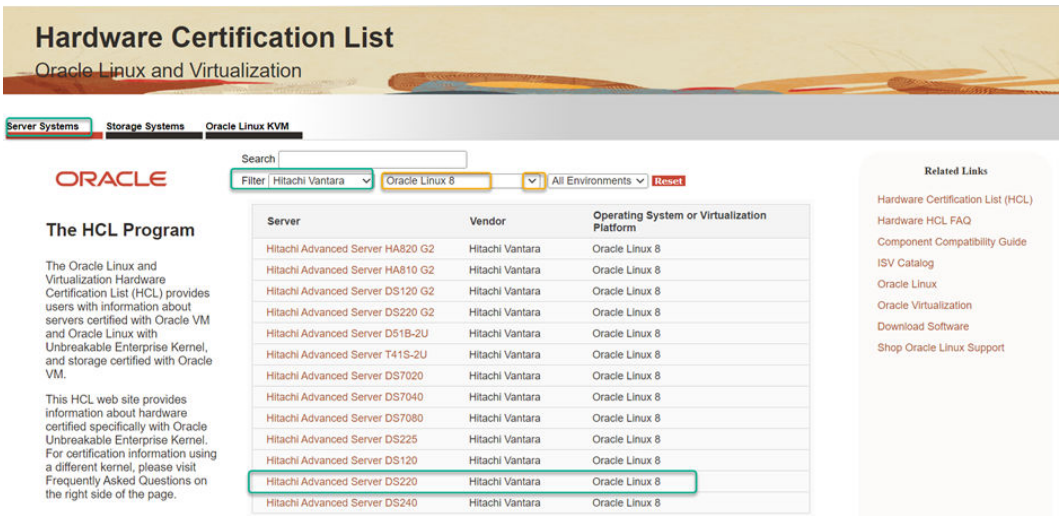
Chapter 4: Compatibility matrix

Before starting, check the compatibility of software and hardware components at their respective vendor site.

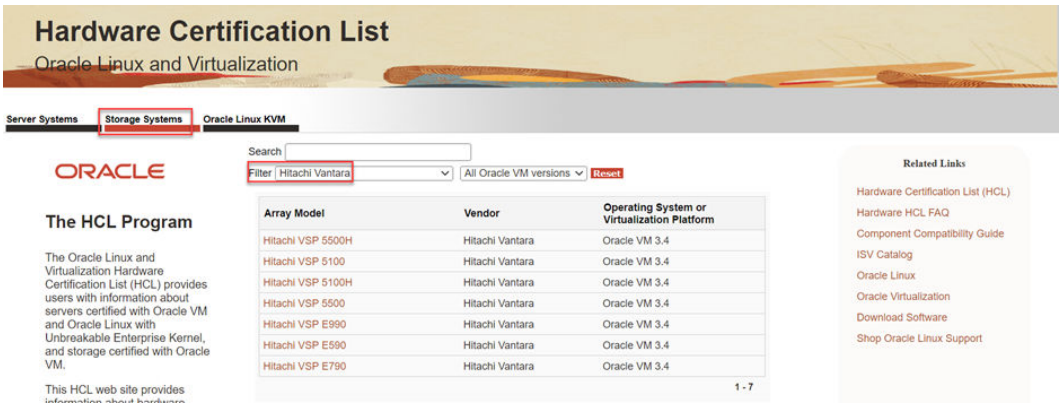
Hardware, OS, and database software compatibility

To check OS (Linux) compatibility with Hitachi hardware see the following URL:

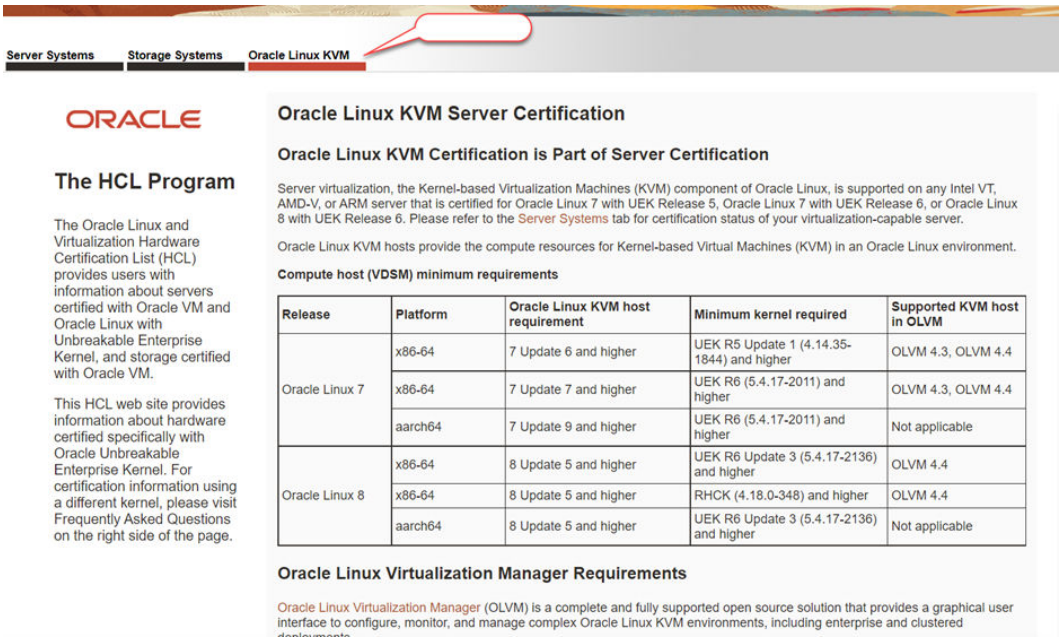
https://linux.oracle.com/ords/f?p=117:1::::RP
This page shows the Server Systems tab.



You can also check OS software compatibility from the Storage Systems tab.



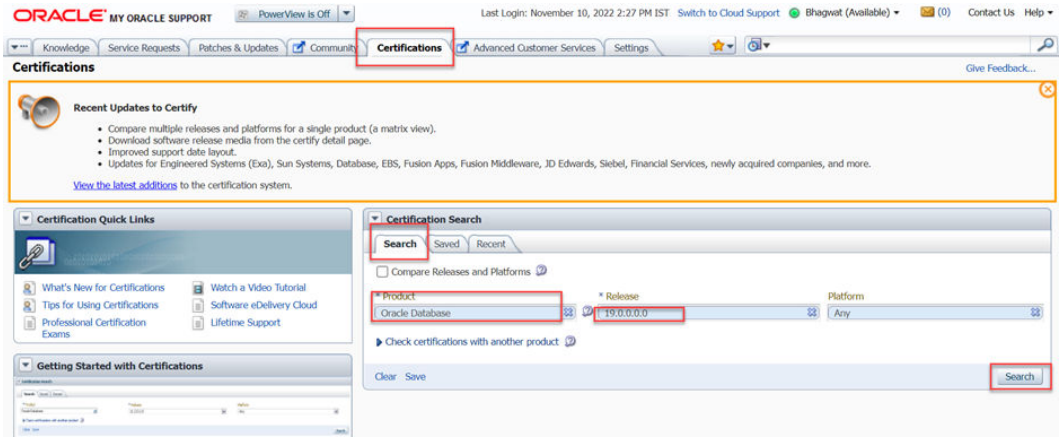
Check KVM compatibility from the Oracle Linux KVM tab.



Release	Platform	Oracle Linux KVM host requirement	Minimum kernel required	Supported KVM host in OLVM
Oracle Linux 7	x86-64	7 Update 6 and higher	UEK R5 Update 1 (4.14.35-1844) and higher	OLVM 4.3, OLVM 4.4
	x86-64	7 Update 7 and higher	UEK R6 (5.4.17-2011) and higher	OLVM 4.3, OLVM 4.4
	aarch64	7 Update 9 and higher	UEK R6 (5.4.17-2011) and higher	Not applicable
Oracle Linux 8	x86-64	8 Update 5 and higher	UEK R6 Update 3 (5.4.17-2136) and higher	OLVM 4.4
	x86-64	8 Update 5 and higher	RHCK (4.18.0-348) and higher	OLVM 4.4
	aarch64	8 Update 5 and higher	UEK R6 Update 3 (5.4.17-2136) and higher	Not applicable

Software compatibility

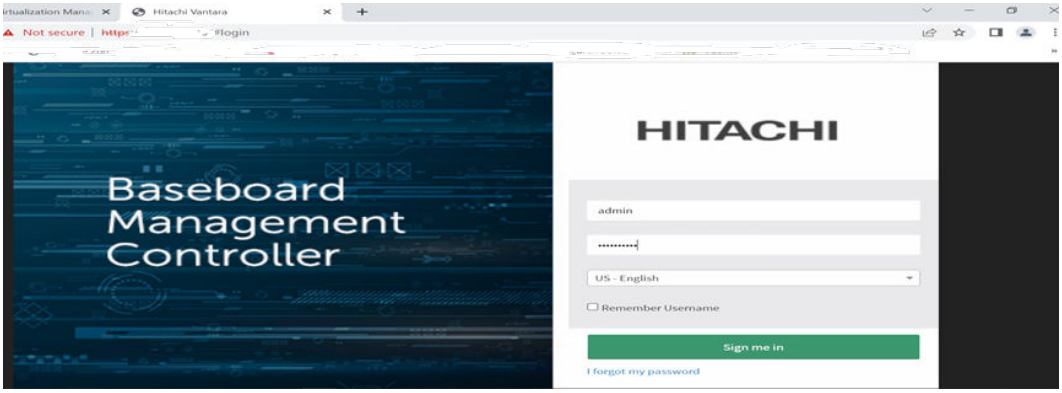
See https://support.oracle.com/ and browse to the Certifications tab to check the database version compatibility with Oracle Linux or any other operating system (support account credentials are needed).



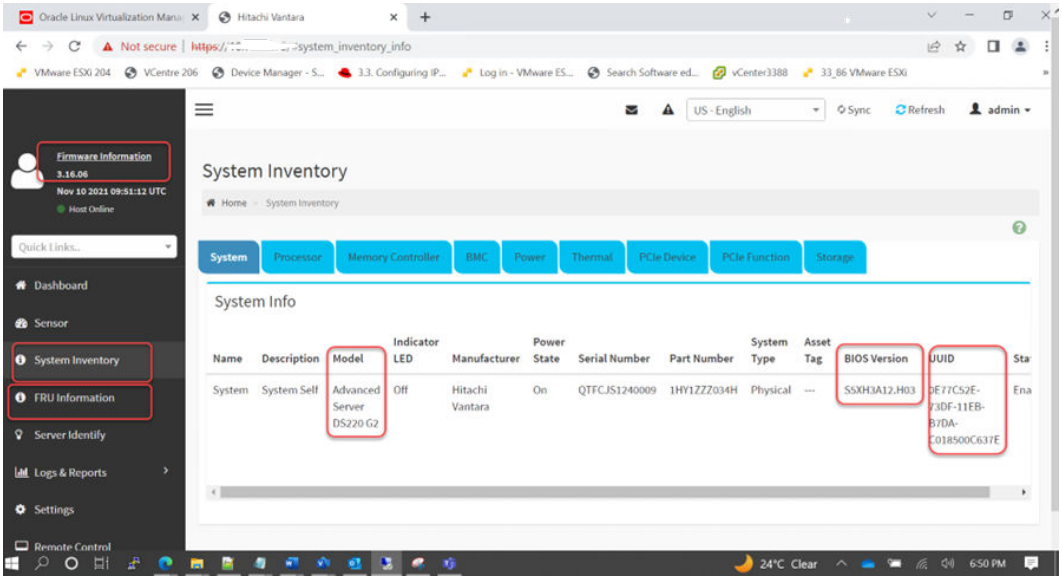
Chapter 5: Hardware pre-checks

After the hardware stack is ready in the lab and mounted in rack, verify that all components are intact and in good condition. Log in to the bare metal host using the BMC console (iLO for Advanced Server HA800 series servers), verify firmware, BIOS, NIC, HBA and other components status.

The Hitachi Advanced Server DS220 BMC management console and login screen for bare metal host looks like the following illustration. The home page shows component names, status, and version details.



On the home page, check firmware information, software versions, model numbers, and the hardware BIOS version. On the same page you can check other hardware-related information such as processors and PCIe devices.



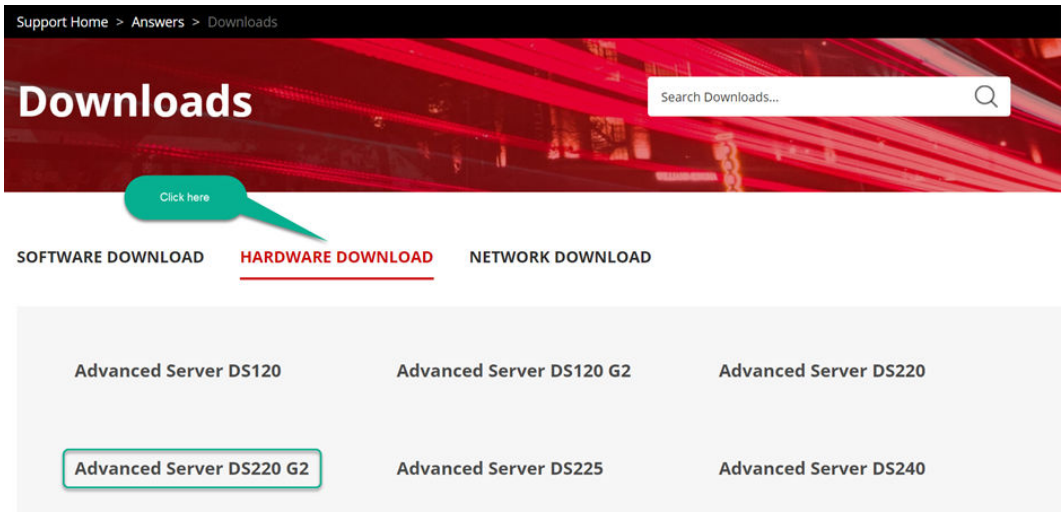
Upgrade firmware and BIOS

If the firmware and BIOS versions are not the latest, download them from the Hitachi Support Connect portal link at https://support.hitachivantara.com/en/user/answers/downloads.html (Hitachi login credentials are required).

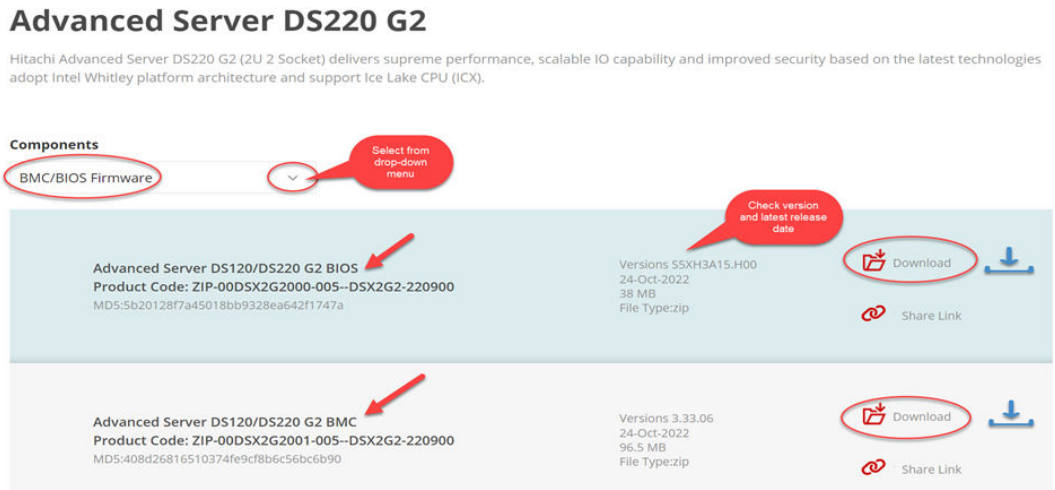
> **Note:** See the `README.txt` file before continuing the upgrade and follow standard practices.

Procedure

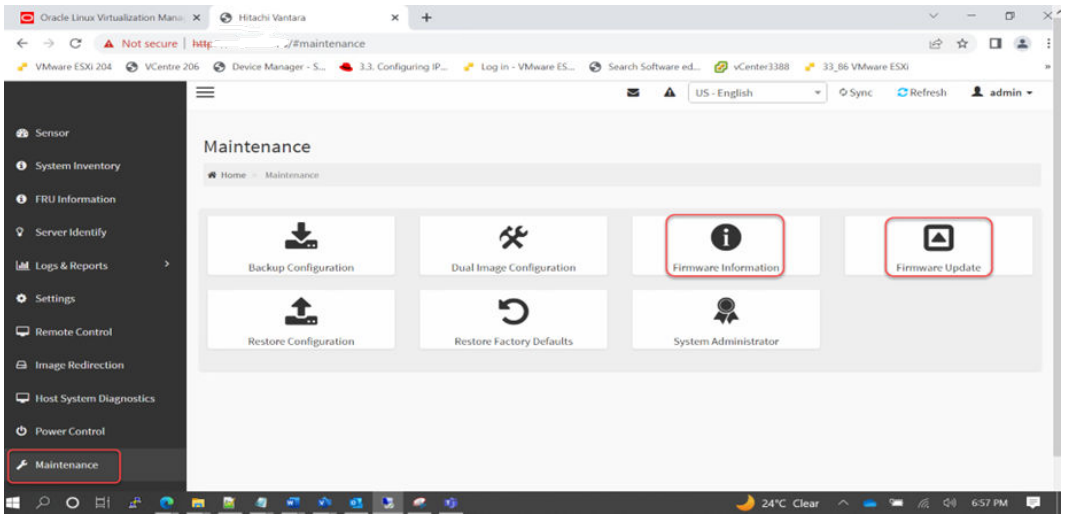
1. After logging in click **Hardware Download**.



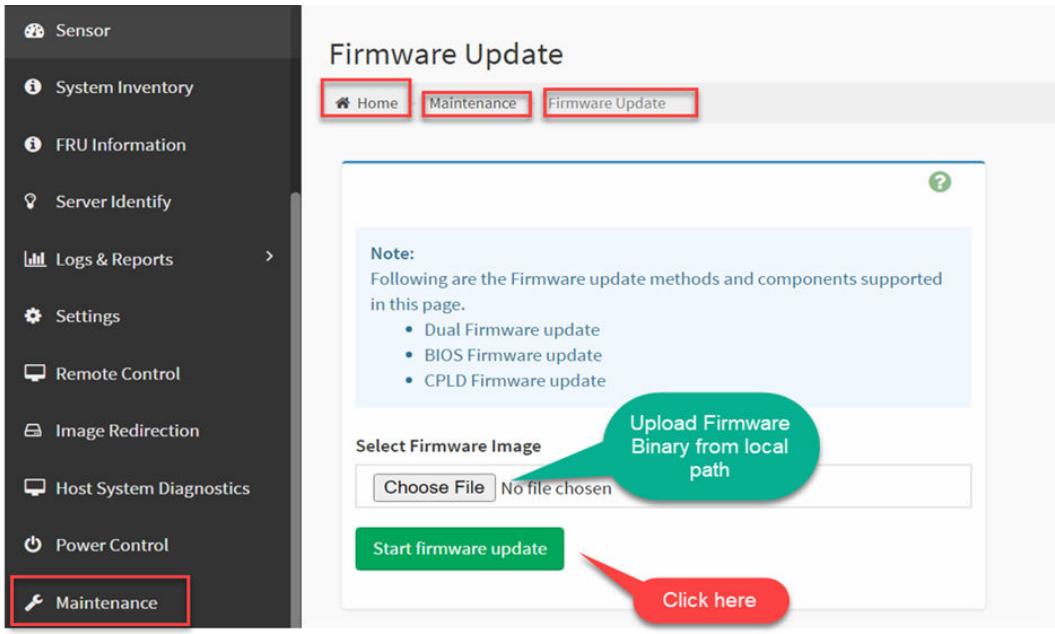
2. Select **Advanced Server DS220 G2**.



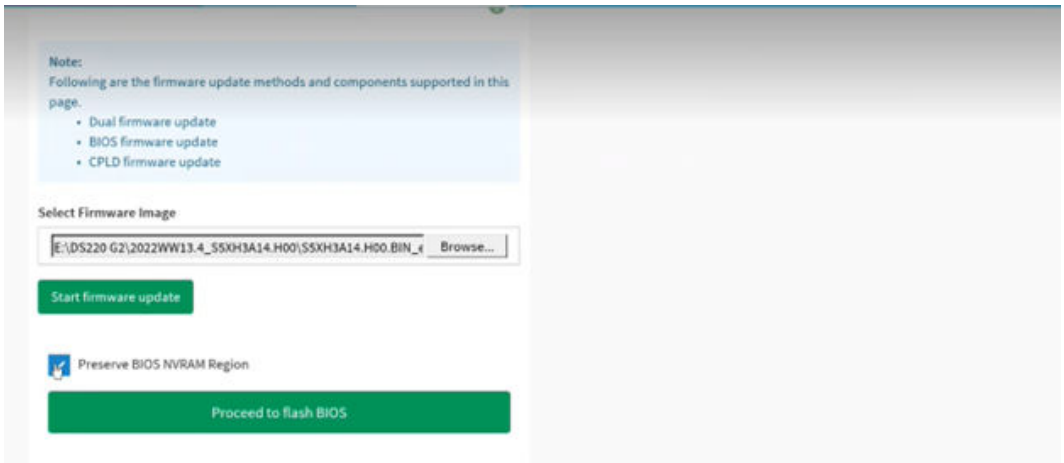
3. In the **Components** section, select **BMC/BIOS Firmware** from the drop down menu and start downloading software.
4. After the download is successful, copy software binaries to their associated directories and log in to the BMC console.
5. Go to **Maintenance** > **Firmware Update** > **Choose File (Upload binary)** > **Start Firmware Update**.



6. Click **Choose File** (browse from local path).

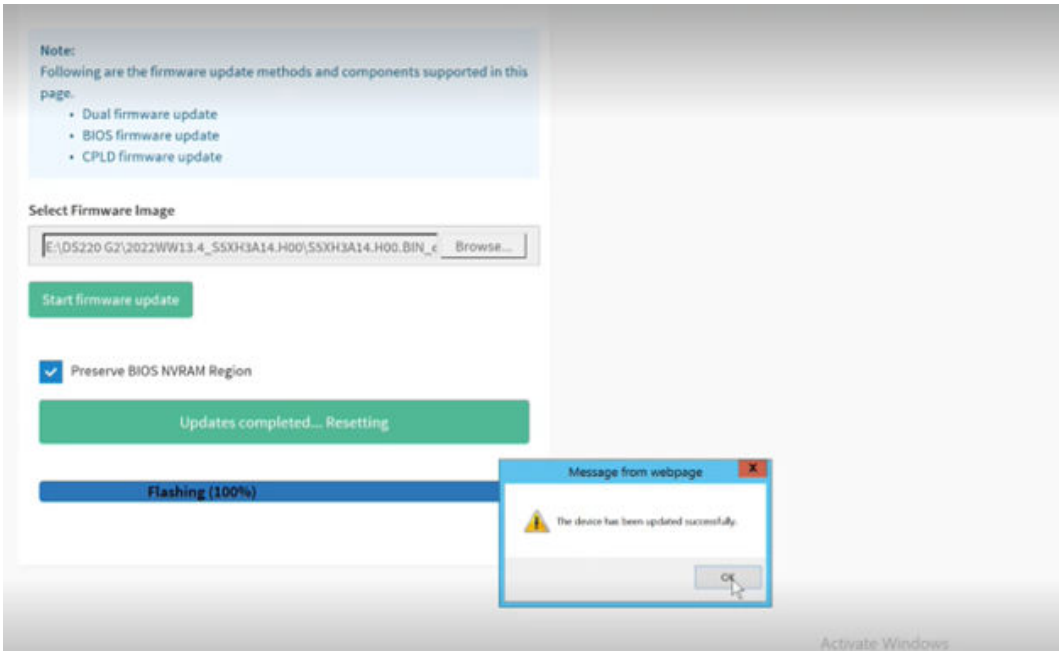


7. Select the **Preserve BIOS NVRAM Region** check box and click **Proceed** to flash the BIOS.



Result

The message *The device has been updated successfully* appears.



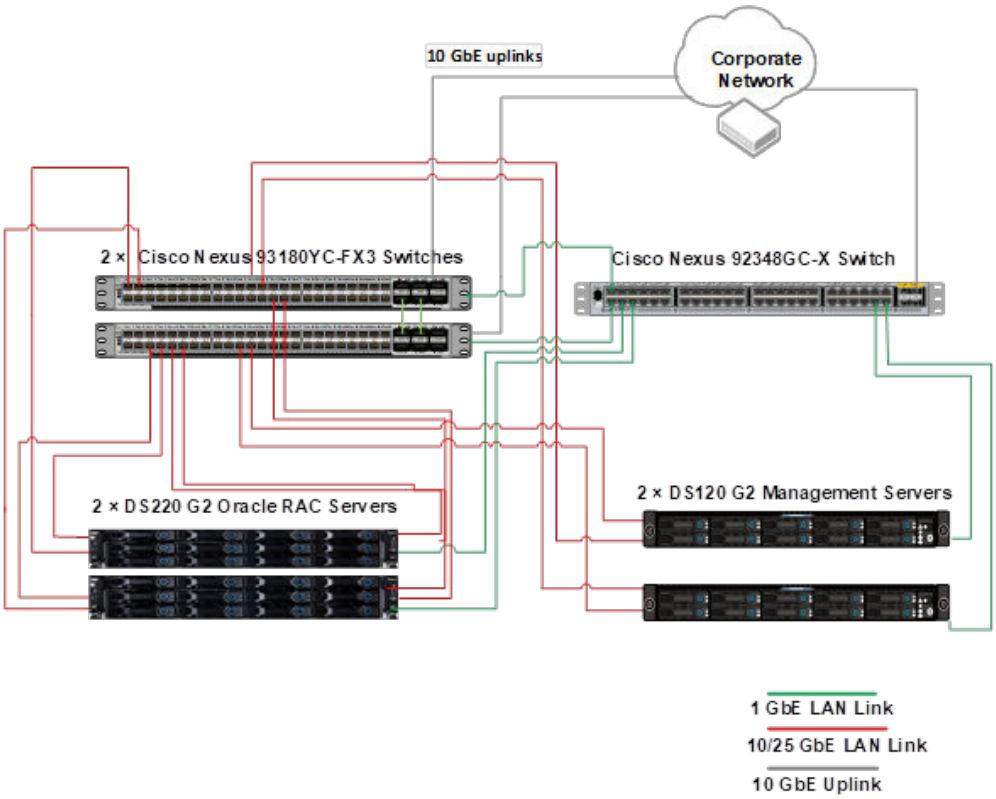
Note: After the firmware update is successful, power cycle the server for the latest BIOS version to take effect.

Chapter 6: Network configuration

The network is the most crucial part between the storage system and servers. To maintain resiliency, two paths are needed for storage access.

Hitachi Vantara recommends using pairs of 25 Gbps NICs for the cluster interconnect network and public network with dual ports. Use NIC bonding to provide failover and load balancing of interconnections within a server.

Note: When creating NIC bonding pairs, ports should be used on different cards to avoid single points of failure (SPoF).



Note:

Management servers are not used in this implementation; they are shown for reference only.

SAN zoning

Zoning is a fabric-based service in a storage area network (SAN) that groups together hosts and storage nodes that require communication. Zoning means restricting the scope of an initiator (host) to a particular target (storage system) in the fabric. An initiator can see only the devices from a particular storage system that is zoned to it. Zoning provides security to data by restricting unauthorized access at the switch level.

A zone is made up of several devices grouped by their Worldwide Names (WWN), or is a group of switch ports. Devices can only see other devices in the same zone, so zones enable servers and storage devices they use to be isolated from other servers and their storage devices.

If one server has two HBAs and dual ports each, two paths are needed from the server to access storage LUNs logically to avoid any storage access failure and to maintain redundancy.

See *Managing Fibre Channel switches* at https://knowledge.hitachivantara.com/Documents/Converged/UCP_Advisor/4.0.0/Managing_Fibre_Channel_switches for details.

Determine WWNN or WWPN information

To create a zone, determine WWNN or WWPN information of components. This information is used to create zone aliases for zone A (Server > Switch) and zone B (Switch > Storage PORT ID).

Path1: `Server/host HBA 1 (WWNN) > Fabric switch port 1(any port on switch device) (WWPN) > Storage PORT (WWNN)`

Path2: `Server/host HBA 2 (WWNN) > Fabric switch port 2(any port on switch device) (WWPN) > Storage PORT (WWNN)`

Procedure

1. Run the following command from the server to determine the HBA port WWNN/WWPN.

```
# more /sys/class/fc_host/host?/port_name
```

```
[root@ig-virt01 host20]# cat /sys/class/fc_host/host20/port_name
0x100000109bd8222e
[root@ig-virt01 host20]# cat /sys/class/fc_host/host19/port_name
0x100000109bd8222d
[root@ig-virt01 host20]# cat /sys/class/fc_host/host18/port_name
0x100000109bd8226d
[root@ig-virt01 host20]# cat /sys/class/fc_host/host17/port_name
0x100000109bd8226c
```

You can also run the following command.

```
# systool -c fc_host -v | grep port_name
```

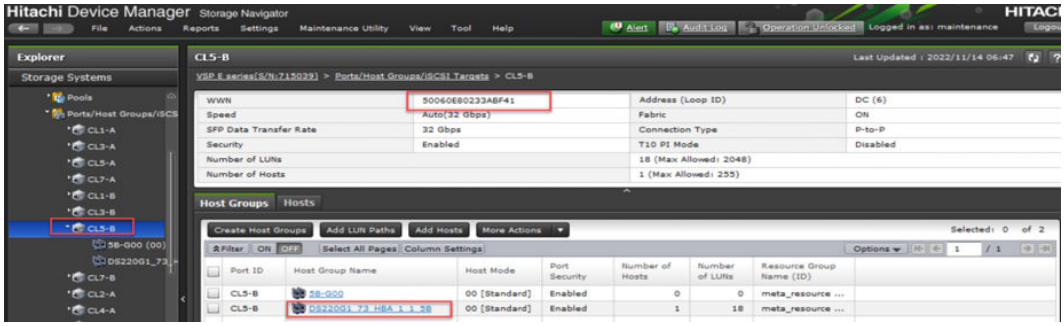
```
[root@ig-virt01 fc_host]# systool -c fc_host -v | grep port_name
    port_name           = "0x100000109bd8226c"
    port_name           = "0x100000109bd8226d"
    port_name           = "0x100000109bd8222d"
    port_name           = "0x100000109bd8222e"
```

You can also run the following command.

```
# ls -l /sys/class/fc_host/
```

2. Log in to the SAN switch, and determine its WWN.

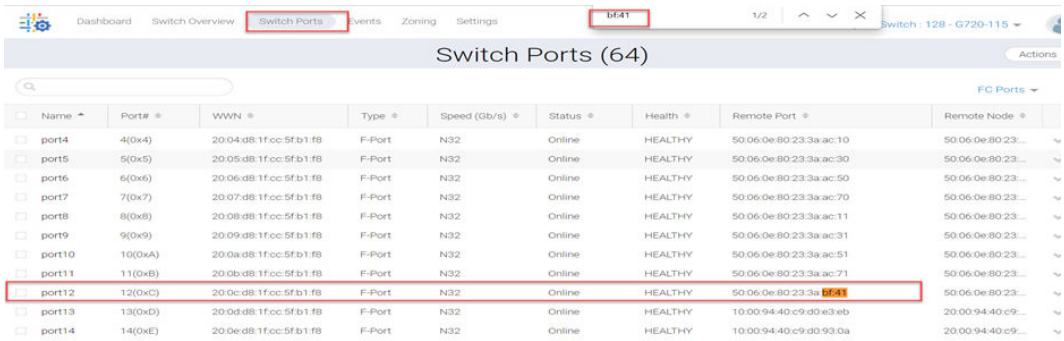
3. Log in to Hitachi Storage Navigator and determine the storage port information (CL5-B with WWN number 50060Eb0233ABF41 in this example).



Now we have HBA, switch, and storage WWNs. Use this information to create zone aliases.

4. Log in to the SAN switch (https://<ip address>/) with user credentials.
5. On the home page, under the **Switch Ports** tab, view the switch WWN and remote host server HBA WWN.

   The following illustration shows storage port WWN (50060Eb0233ABF41) connected to port12 on the switch.



Zone aliases

A zone alias is a name given to an object or set of objects for zoning purposes. Zone aliases simplify zone administration by eliminating the repetitive entry of WWNs or port numbers.

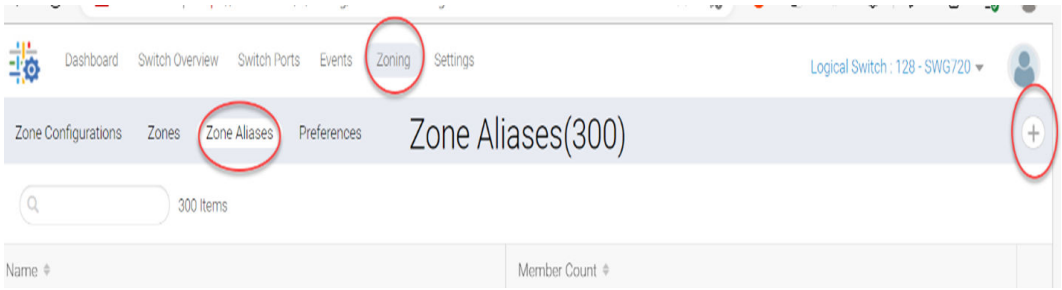
After assigning a zone alias to one or more objects, you can perform zoning operations on the alias instead of having to specify the individual ports and WWNs for the objects.

See Managing zone aliases https://knowledge.hitachivantara.com/Documents/Converged/UCP_Advisor/4.0.0/Managing_Fibre_Channel_switches/08_Managing_zone_aliases for details.

Create a zone alias for storage ports

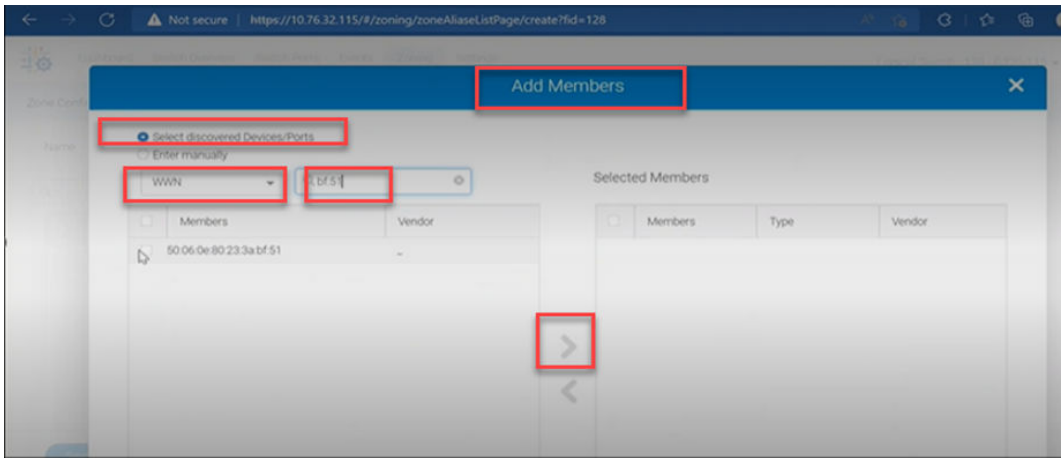
Procedure

1. Log in to the SAN switch controller and choose **Zoning** > **Zone aliases** > > **+ (add member)**

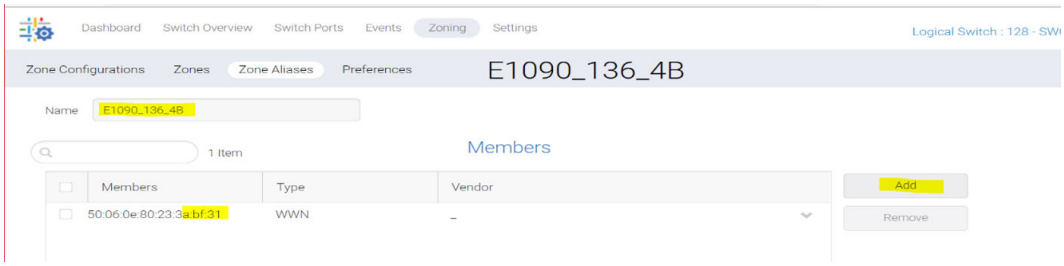


.

2. Provide the zone alias name and search WWN number of the storage (50060Eb0233ABF41 or 51).



3. Confirm the zone alias (for example, E1090_136_4B).



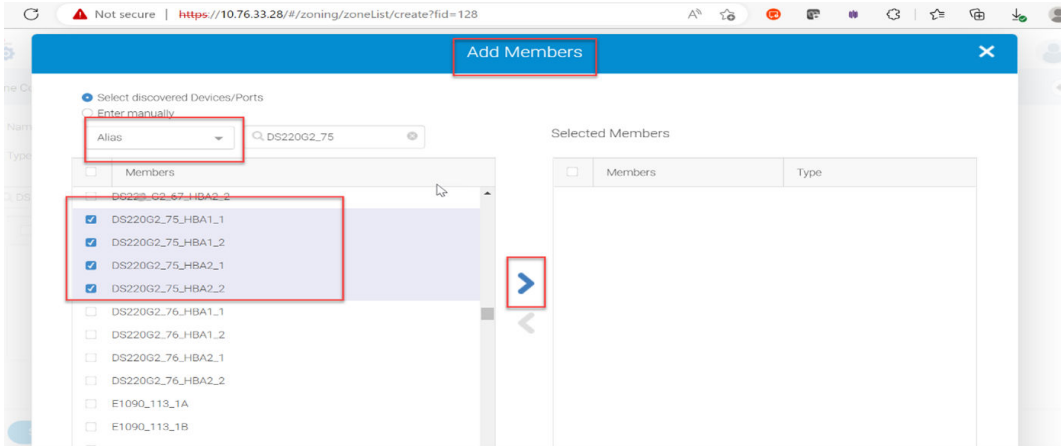
Note: This is an example for reference purposes. The WWN number and alias names will be different in your environment.

Create a zone alias for HBA ports

Procedure

1. Click **Zoning** > **Zone aliases** > **Search with WWN number**.
2. When the results appear, click the associated check boxes and press **Add** > **Save**.

   The following example shows four dual port HBA aliases: DS220G2_75_HBA1_1, DS220G2_75_HBA1_2, DS220G2_75_HBA2_1, and DS220G2_75_HBA2_2.



Result

There are two zones as follows:

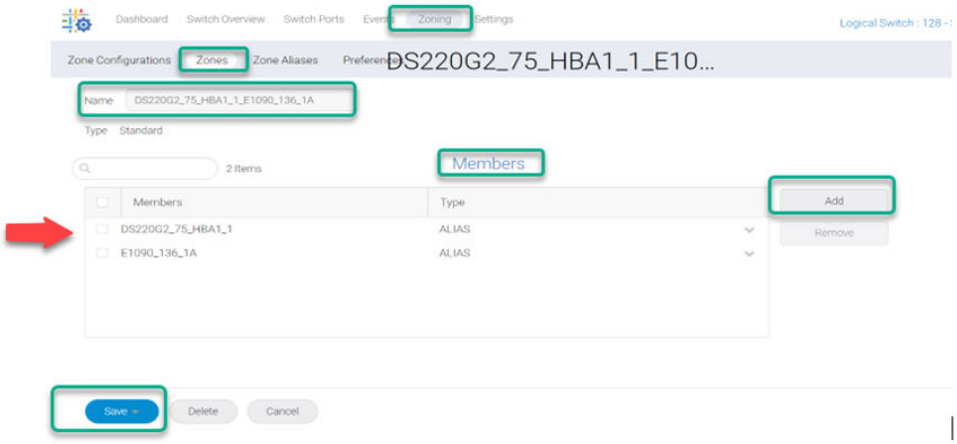
- Storage <-> SAN switch > E1090_136_4B
- SAN Switch <-> Server (HBA port) > DS220G2_75_HBA1_1

Create a zone for storage port zone aliases and HBA port zone aliases

The storage port zone aliases and HBA port zone aliases need a zone and associated name. In the following example, zone DS220G2_75_HBA1_1_E1090_136_1A) is created.

Procedure

1. Go to **Zoning** > **Zones** > **Specify Zone Name** > **Search for Member (zone aliases created earlier)** > **Add** > **Save**.



See *Managing zones* at https://knowledge.hitachivantara.com/Documents/Converged/UCP_Advisor/4.0.0/Managing_Fibre_Channel_switches/09_Managing_zones for more information.

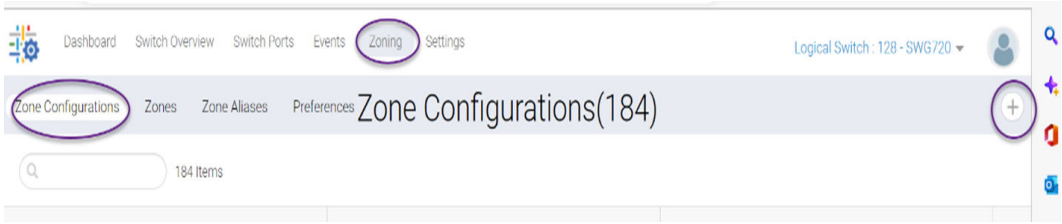
2. Create a zone for each HBA port.
3. After the zones are created, add them to the zone configuration.

Zone configuration

A zone configuration is a set of SAN zones. SAN zoning is a fabric-based service for grouping the devices in a SAN into logical segments to control communications between those devices.

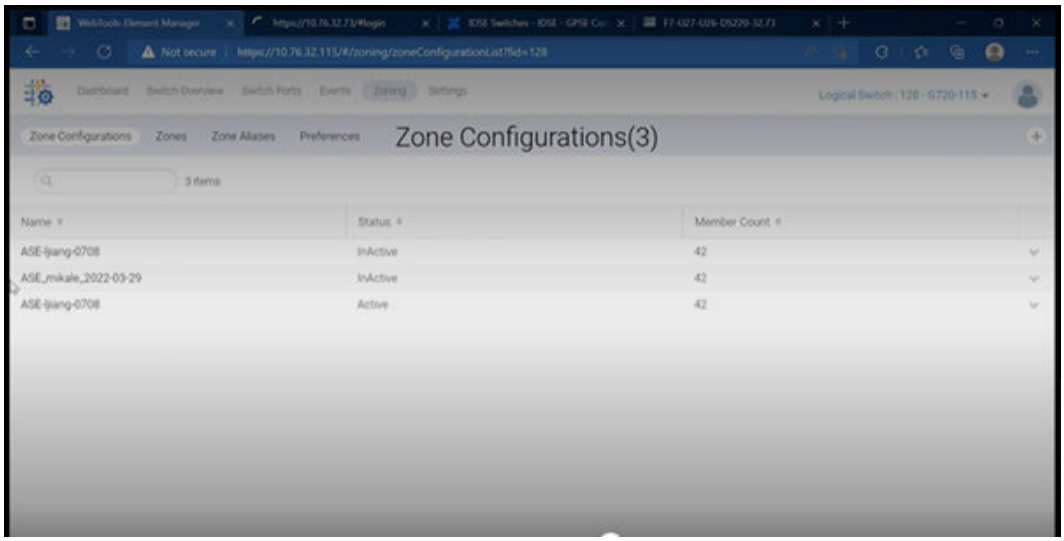
Procedure

1. Go to **Zoning** > **Zone Configurations** > **+ (Add)**.

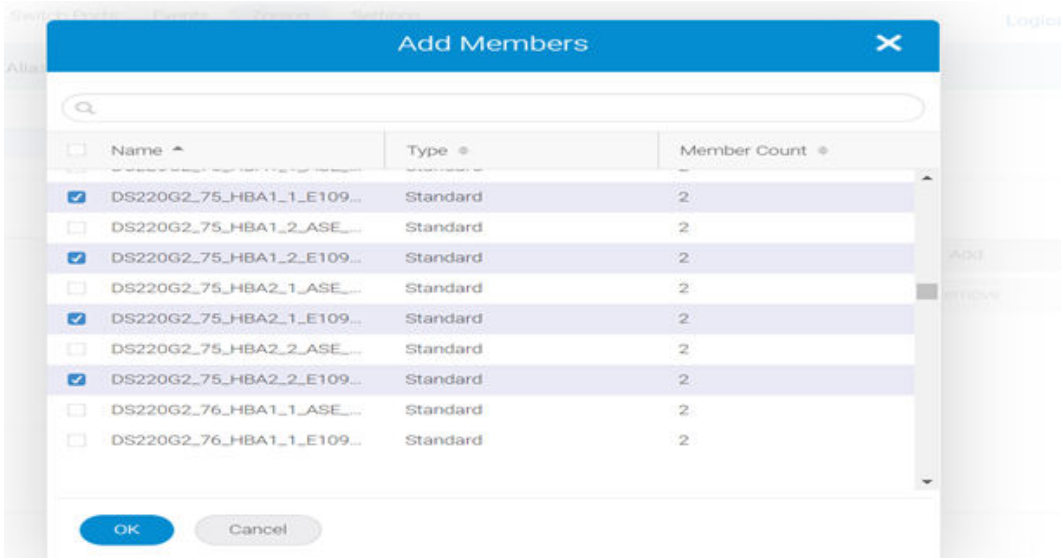


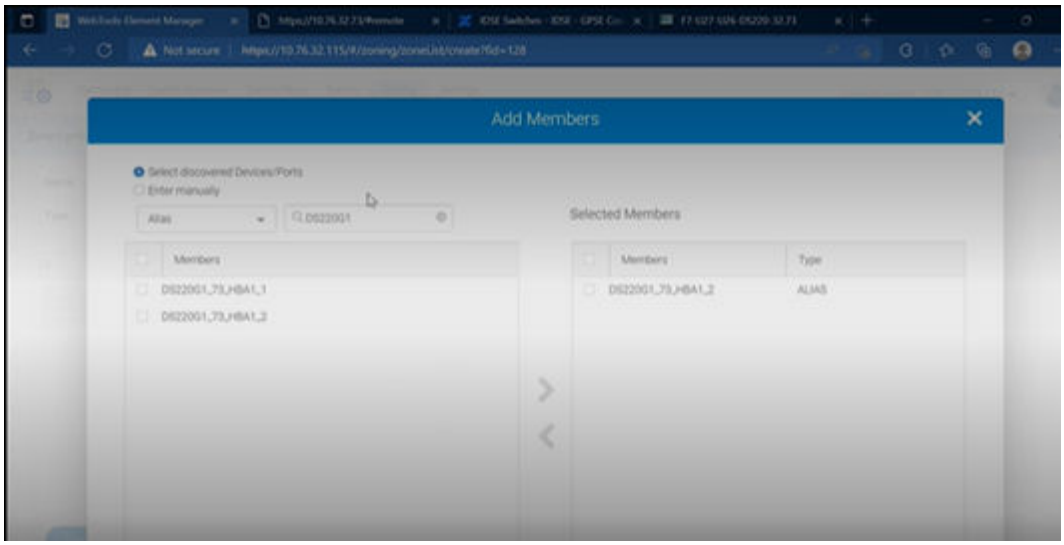
2. Provide a name for the zone configuration (for example, ASE_mkale_0920) and add zone members.

   Verify the existing zones that were created previously, as shown before the zone configuration.



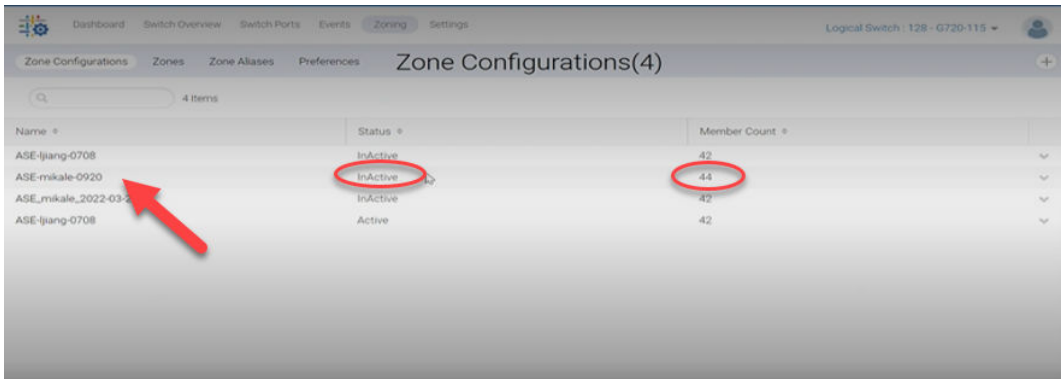
3. Add the zone members that were created.



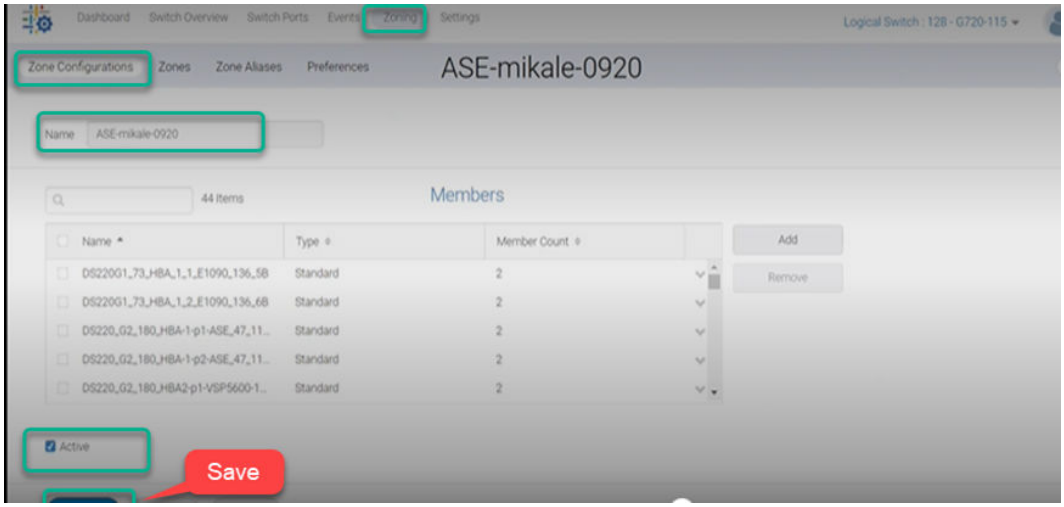


Note: Note down the member count before and after zone configuration.

After Zone configuration (ASE_mkale_0920), the new zone with a status of InActive needs to be changed to Active.



Before zone configuration, the Member Count was 42 and after addition, zone configuration is 44 members. A zone configuration with 44 members must be activated.



Note: Only one zone configuration can be enabled at a time.

Chapter 7: Storage configuration

Hitachi Device Manager Storage Navigator is used to administer storage tasks such as capacity management, availability management, continuity management, and financial management.

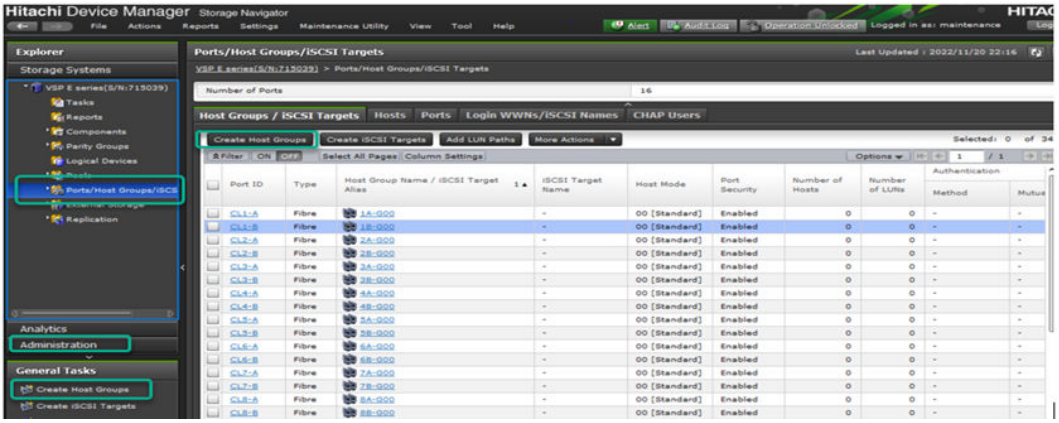
Create host groups

Log in to Hitachi Device Manager Storage Navigator to create host groups so LUNs created on storage systems can be mapped and visible on the server.

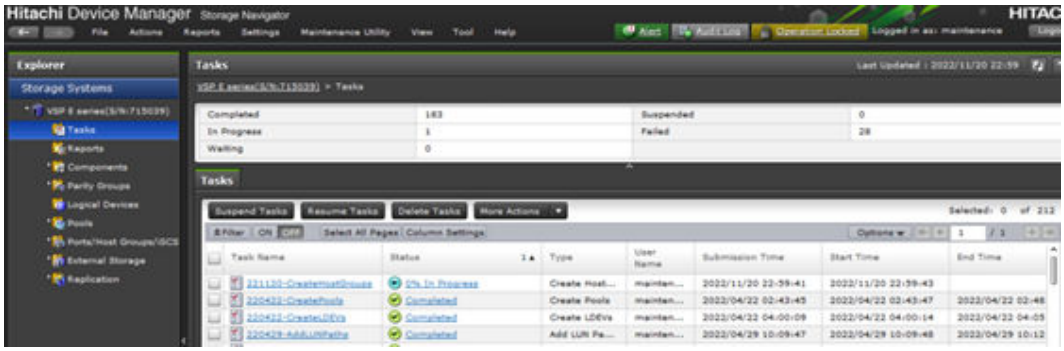
See *Configuring host groups* at https://knowledge.hitachivantara.com/Documents/Management_Software/SVOS/9.3/Volume_Management_-_VSP_G130%2C_G%2F%2FF350%2C_G%2F%2FF370%2C_G%2F%2FF700%2C_G%2F%2FF900/Provisioning/13_Configuring_host_groups for details.

Procedure

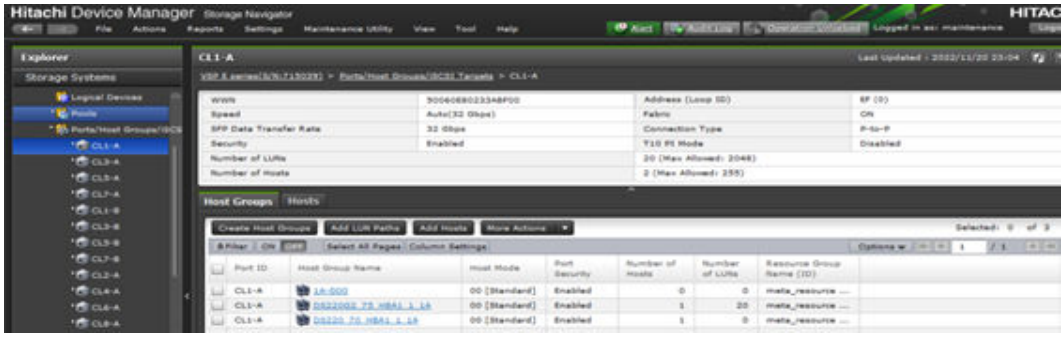
1. Log in to Hitachi Device Manager and select **Ports/Host Groups/iSCSI** > **Create Host Groups**.



2. Go to **Host Group Name** > **Resource Group** > **Host Mode** > **Add New Host** > **Add** > > **Finish**.
3. Go to **Tasks** and monitor the progress (it takes a few minutes).



4. Upon completion, verify host group creation.



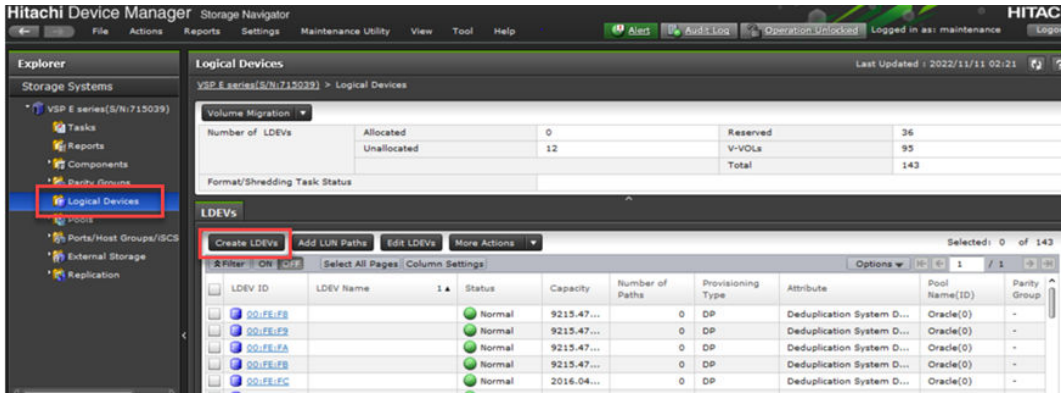
Create LUNs

Creating LUNs on storage systems is necessary in a SAN environment to use disk space in chunks so that it can be assigned to specific hosts. Make sure the SAN area is accessible to hosts with two HBAs to ensure fault tolerance.

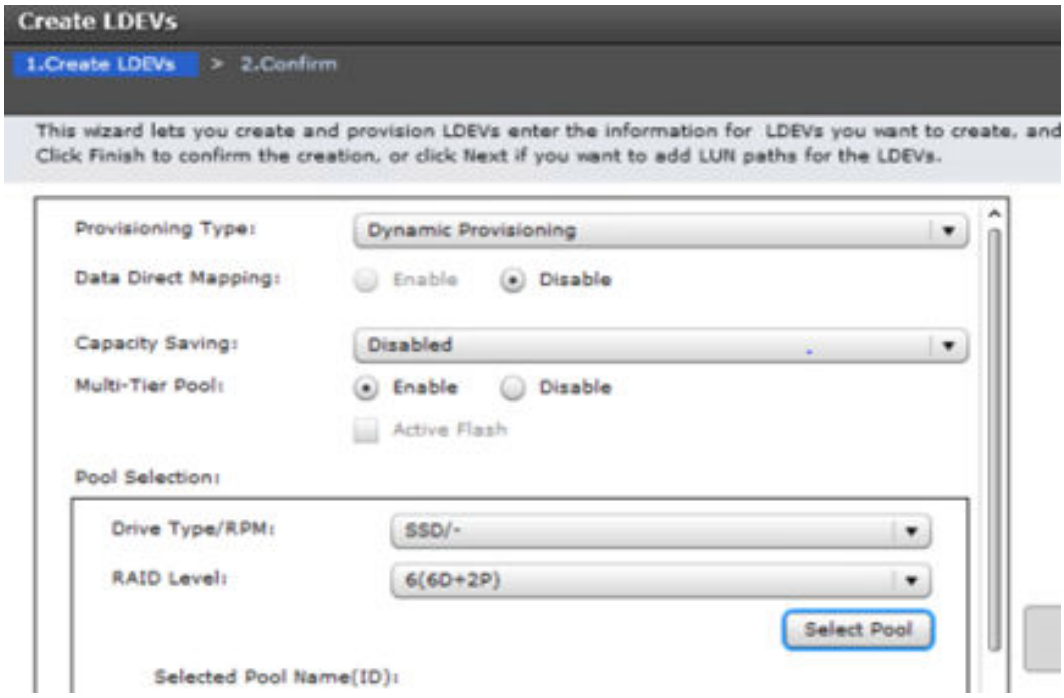
Note: Before LUN creation, make sure you have sufficient space available in the storage system as well as the dynamic provisioning pool.

Procedure

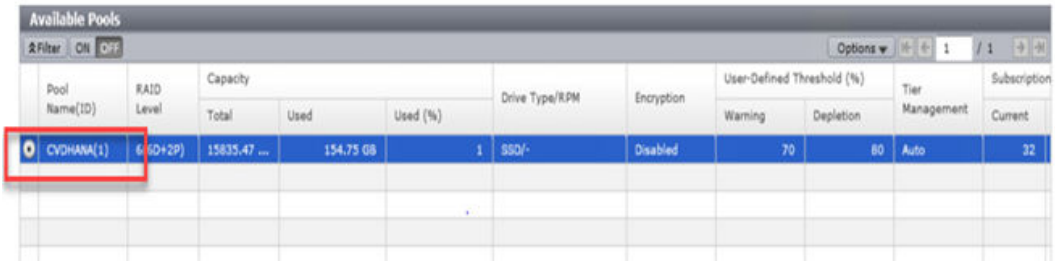
1. Log in to Hitachi Device Manager and select **Logical Devices** > **Create LDEVs**.



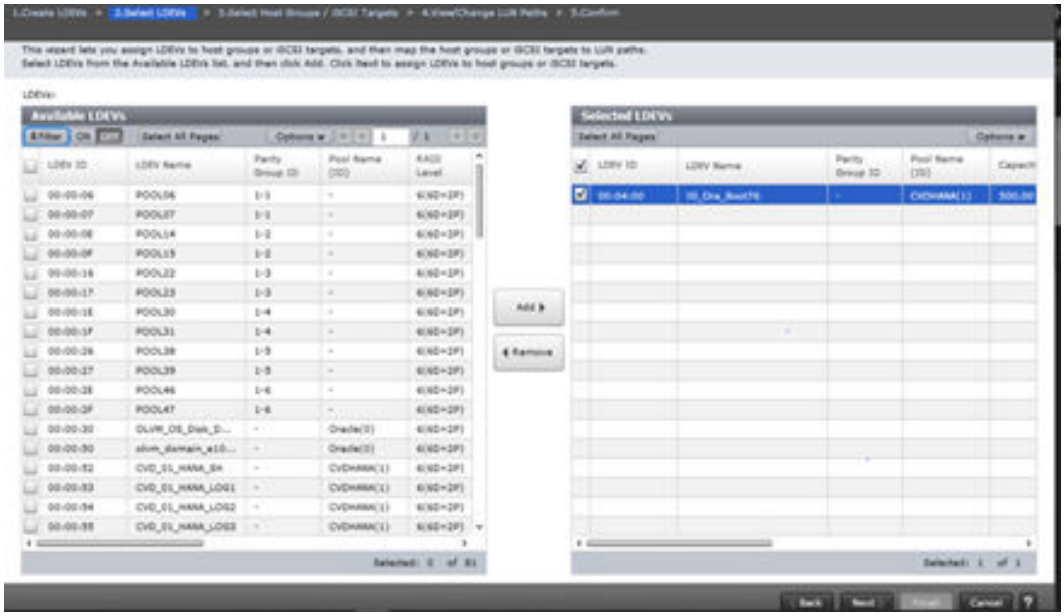
2. Select the Provisioning Type, Capacity Saving, and Pool Selection and then click **Select Pool**.



3. Make a selection from the list of Available Pools.

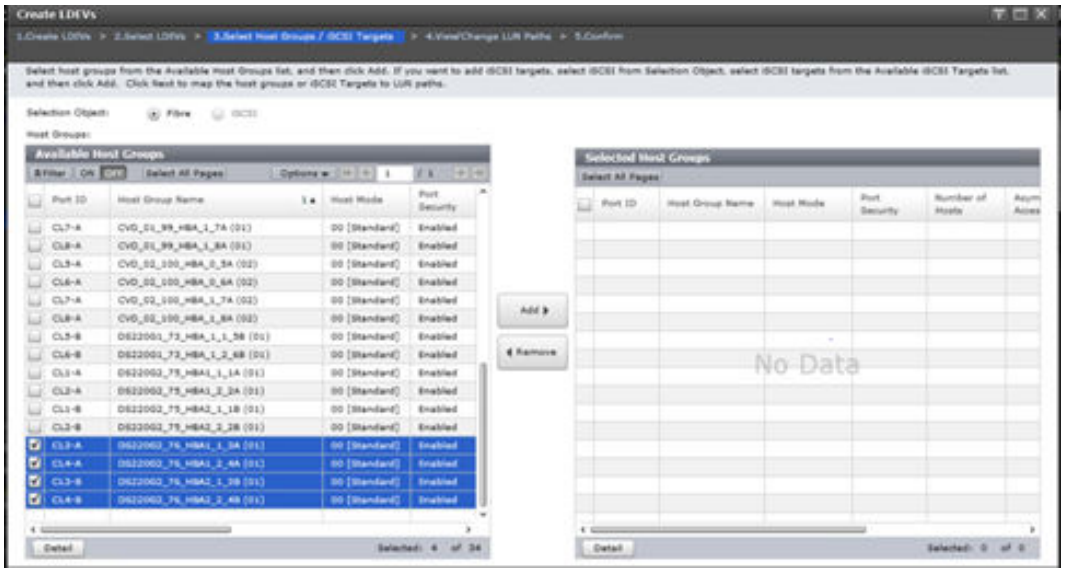


4. Select an LDEV ID and Name and click **Next**.

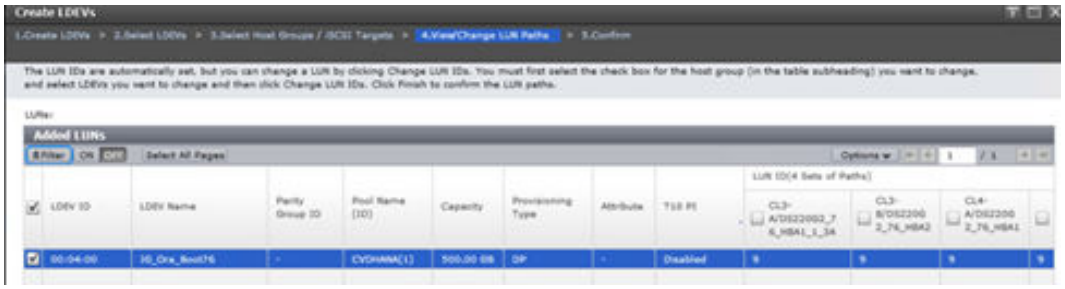


> **Note:** In this example, there is already a data pool on the storage system. If a data pool does not exist, then create one with the help of the storage administrator.

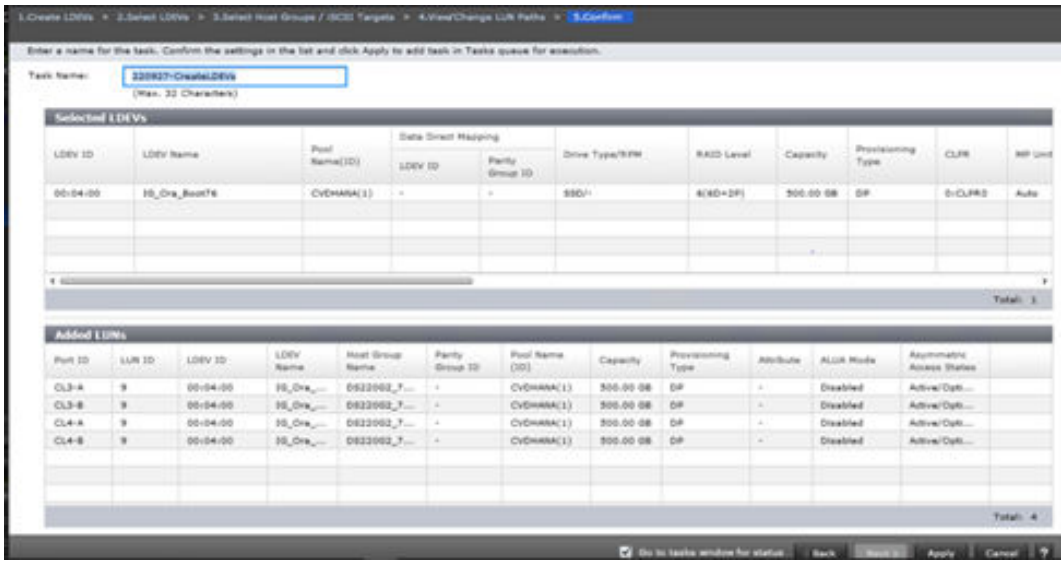
5. Select the name of the hostgroup on which the LUN will be mapped or visible on the host. Select multiple paths to the LUN to avoid any single point of failure, and then click **Add**.



6. Verify LUN and path details.

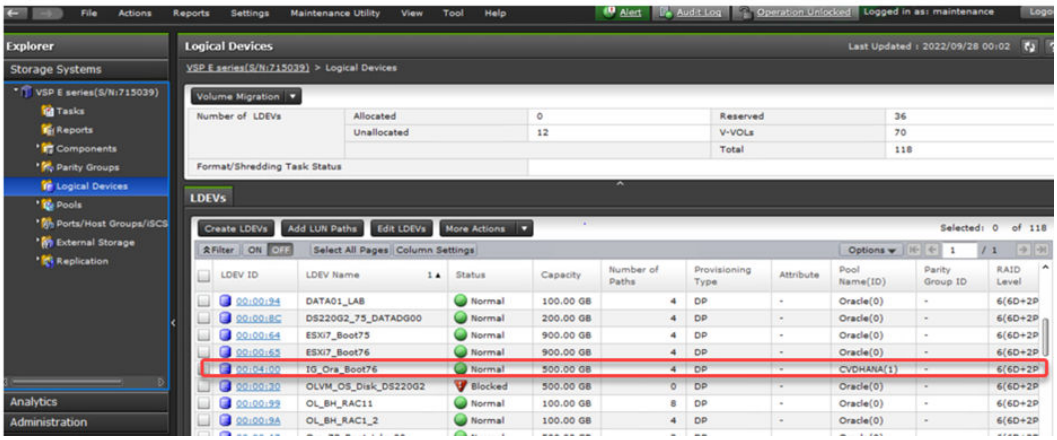


7. Confirm details.



Result

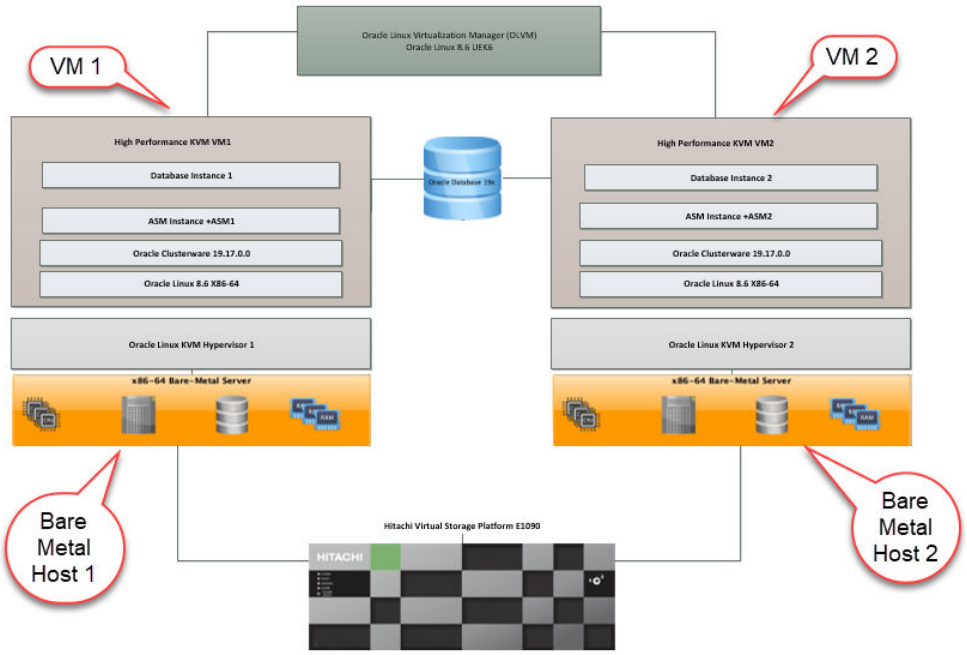
After the LUN is created and the details are verified, the new LUN is listed under Logical Devices.



Chapter 8: Cluster software installation and configuration

Two bare metal hosts are needed to configure clusters. With the help of virtualization, we can configure multiple VMs on each bare metal host.

The following illustration shows the components and associated software used to implement the virtualized solution.



There are two Hitachi Advanced Server DS220 G2 bare metal hosts, and LUNs are shared across these hosts for the Oracle clusterware environment. Oracle Linux 8.6 uek6 (x86_64) is installed on both hosts. After the OS installation, the Oracle KVM hypervisor is installed for virtualization. On top of the KVM hypervisor, VMs are created on each bare metal host with the help of Oracle Linux Virtualization Manager (OLVM), which resides on a separate management host.

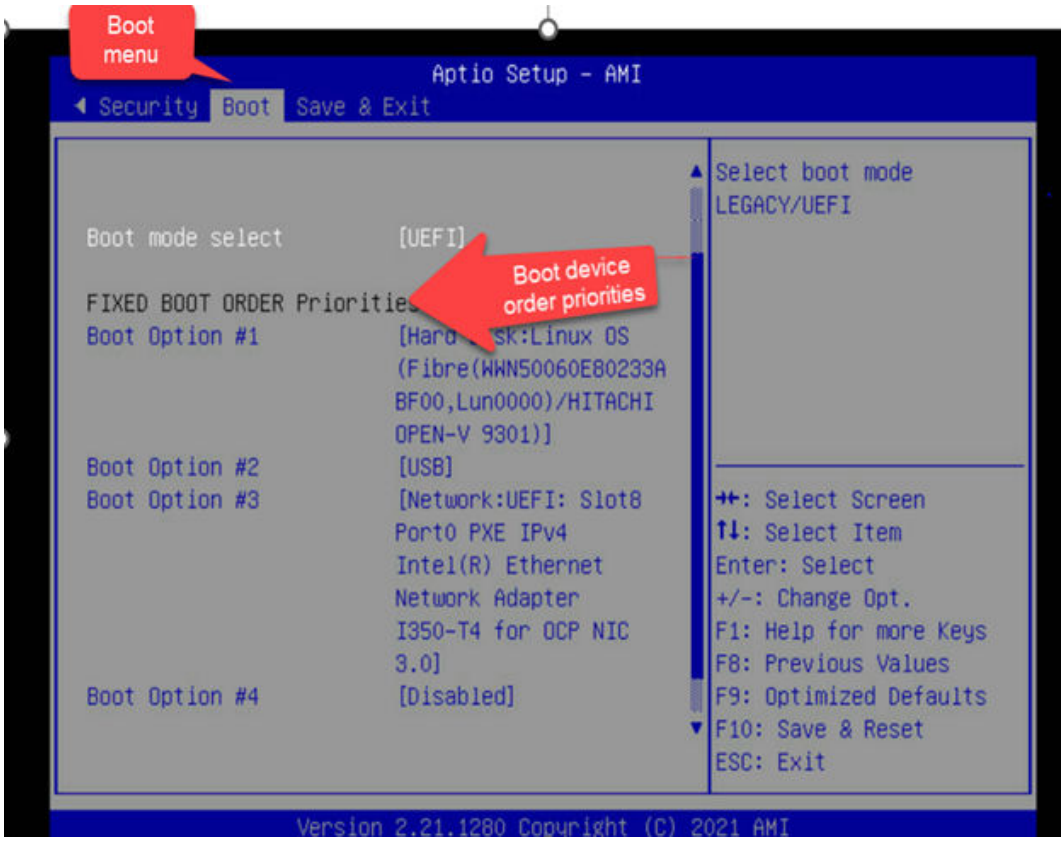
Install the operating system

Before you begin

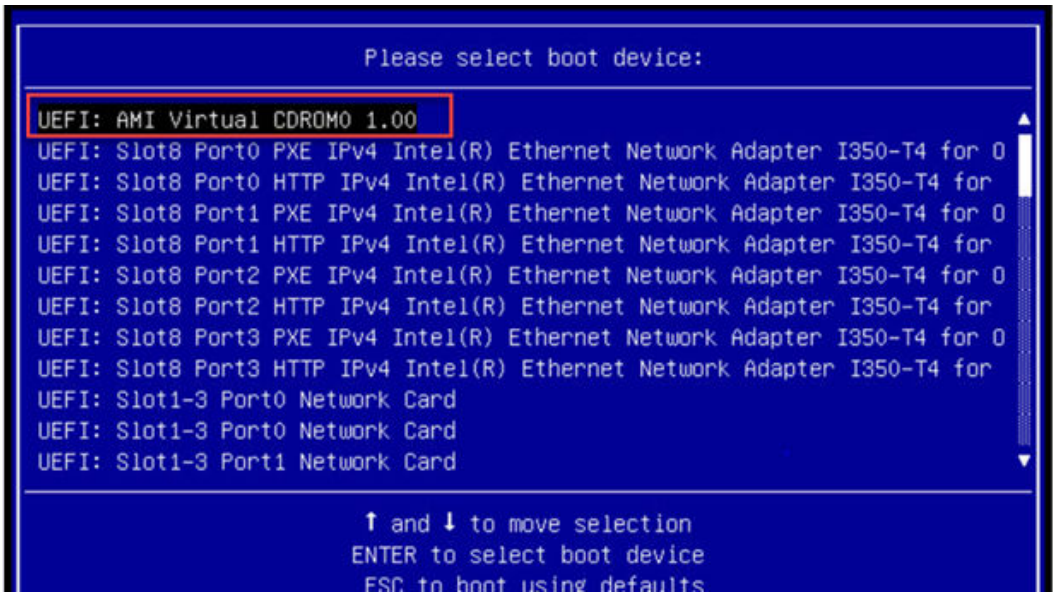
- Before proceeding with the OS installation, both bare metal hosts must be configured with boot LUNs provisioned from Hitachi Device Manager.
- Download the Oracle Linux ISO images from Oracle Software Delivery Cloud at https://edelivery.oracle.com/linux.

Procedure

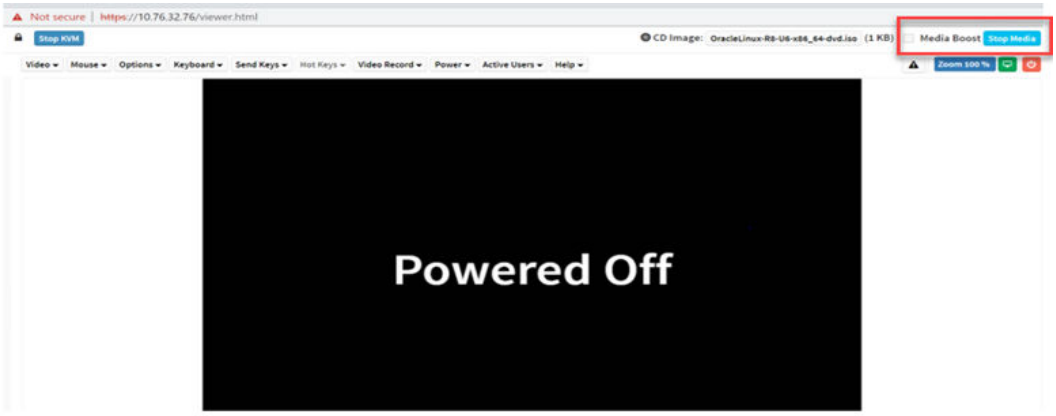
1. Log in to the bare metal host at `http://<IP address>/` using admin credentials.
2. Log in to the BMC host, go to the Boot menu, and find the boot LUN to be used for OS installation.
3. If the boot LUN is visible, go to the Boot menu by pressing the arrow □ keys on the keyboard and selecting **Boot Option #1**.



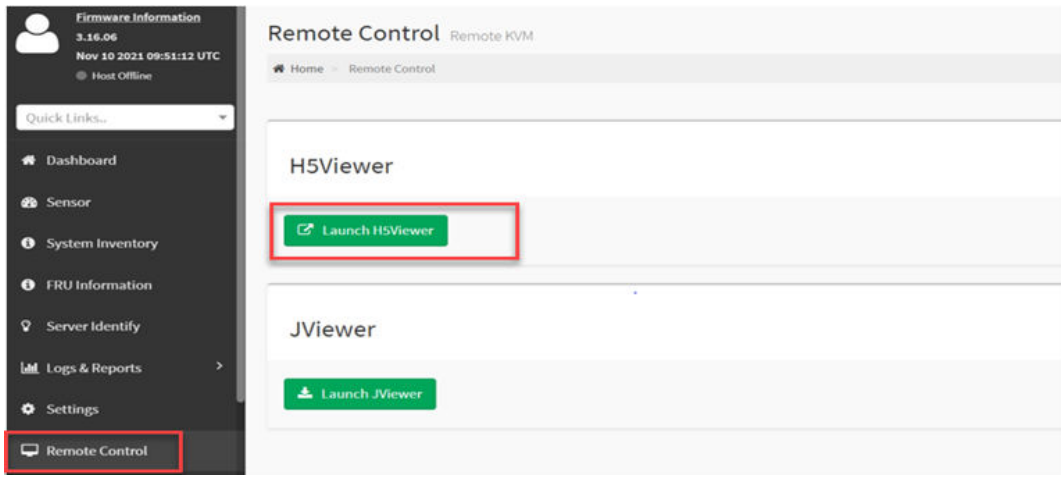
4. Because this is an empty server the OS will boot primarily from CDROM. Go to the Boot menu (Press F11 or F2 for the Boot menu during startup) and select CDROM for the boot device.



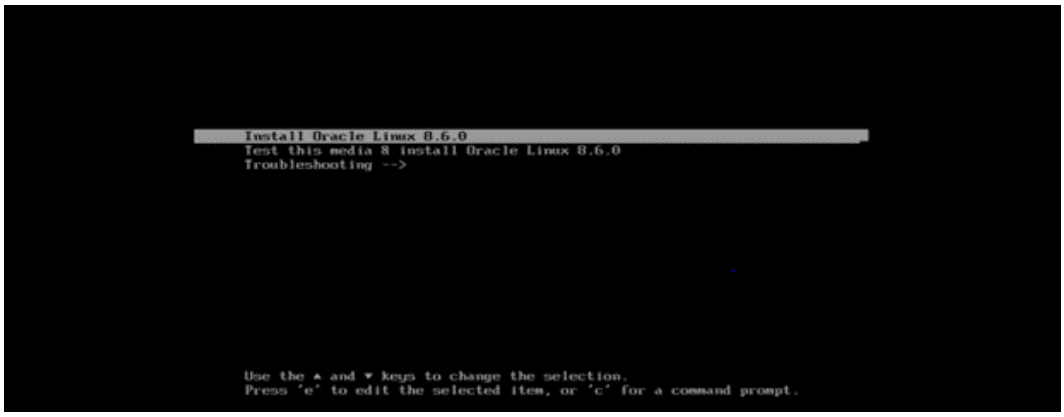
5. Click **Media Boost** and upload the ISO image that was downloaded previously.



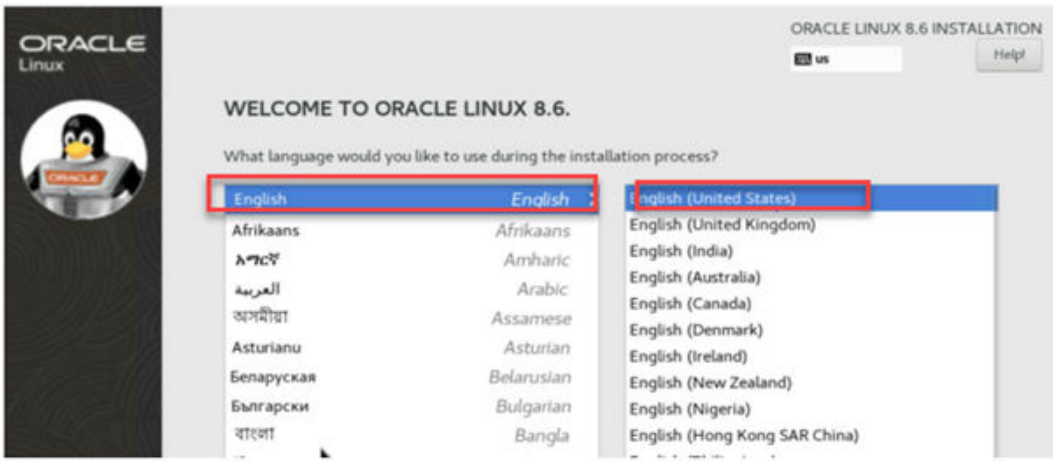
6. Go to **BMC** > **Remote Control** > **Launch H5Viewer**.



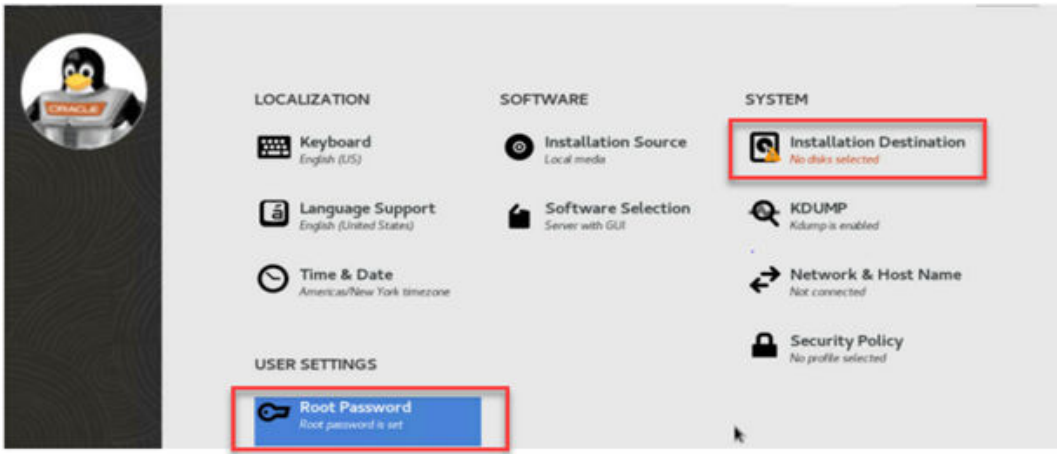
7. Walk through the Oracle Linux 8.6.0 installation.



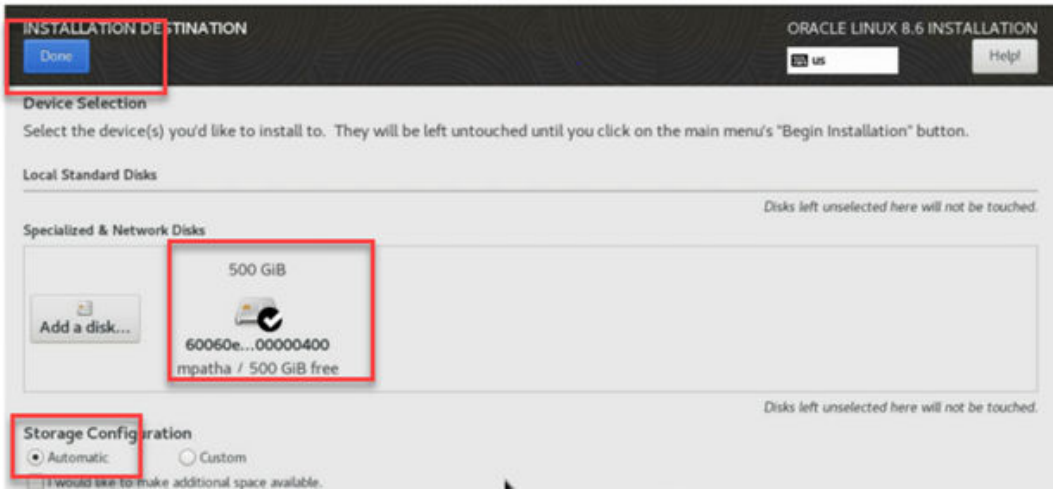
8. Select English (United States) as the language.



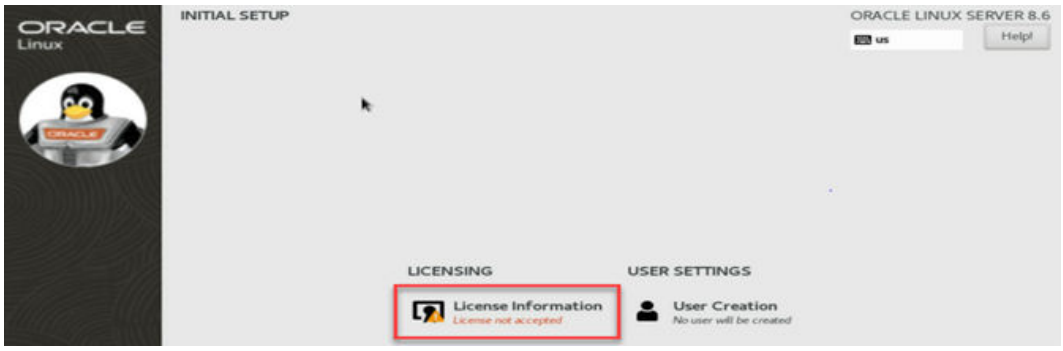
9. Select the installation destination, for example, the Boot LUN that is assigned.



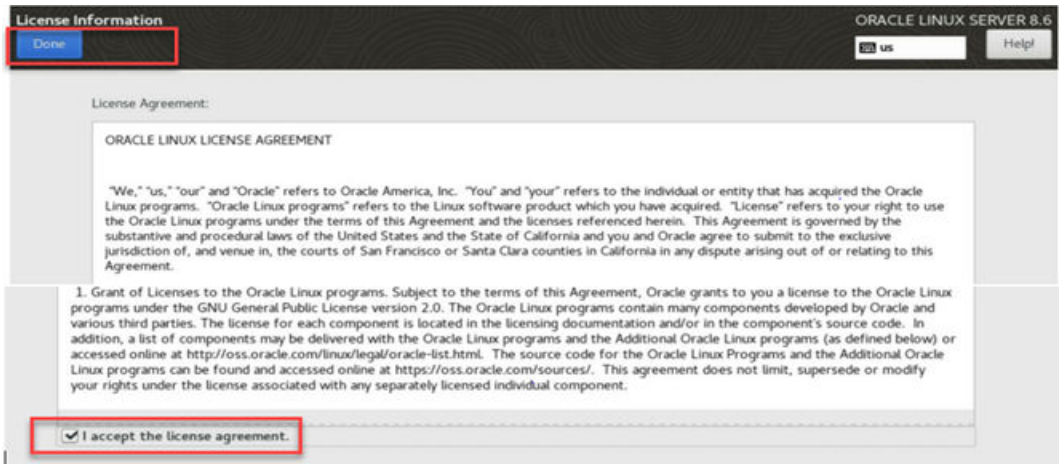
10. Select the boot disk (LUN) that was created earlier and configure storage as needed. Also specify the root user password.



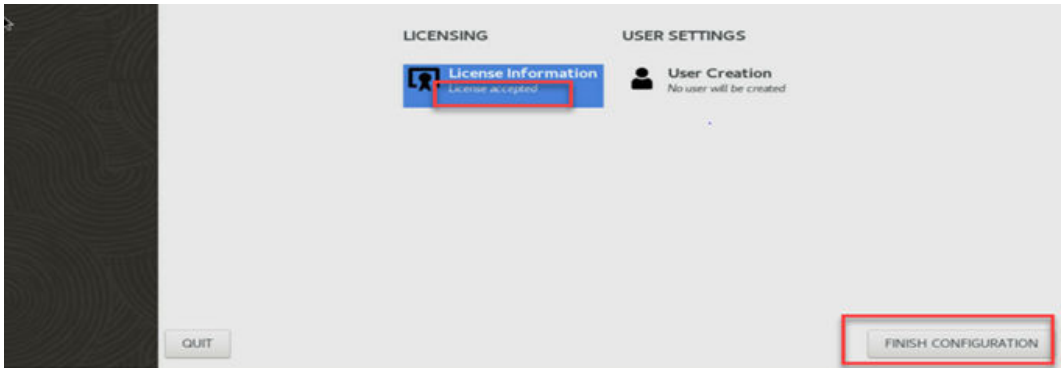
11. Restart the system and accept the license agreement.



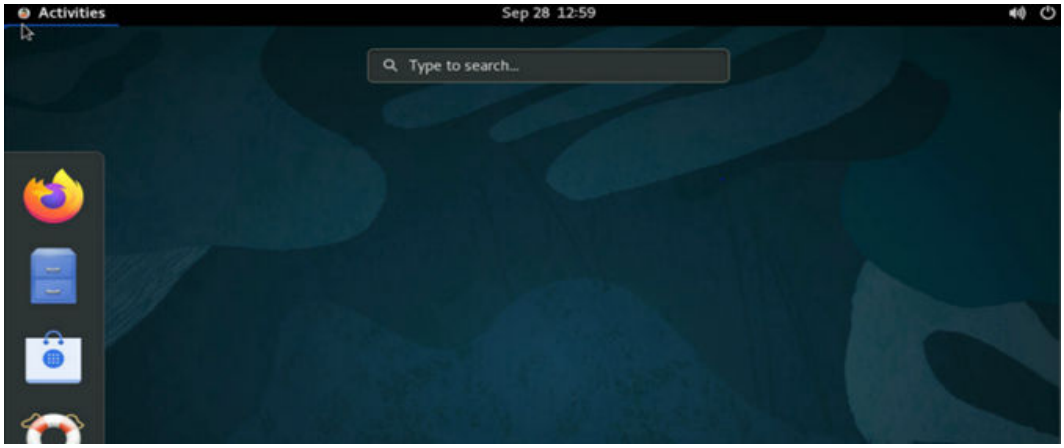
12. Click **Done**.



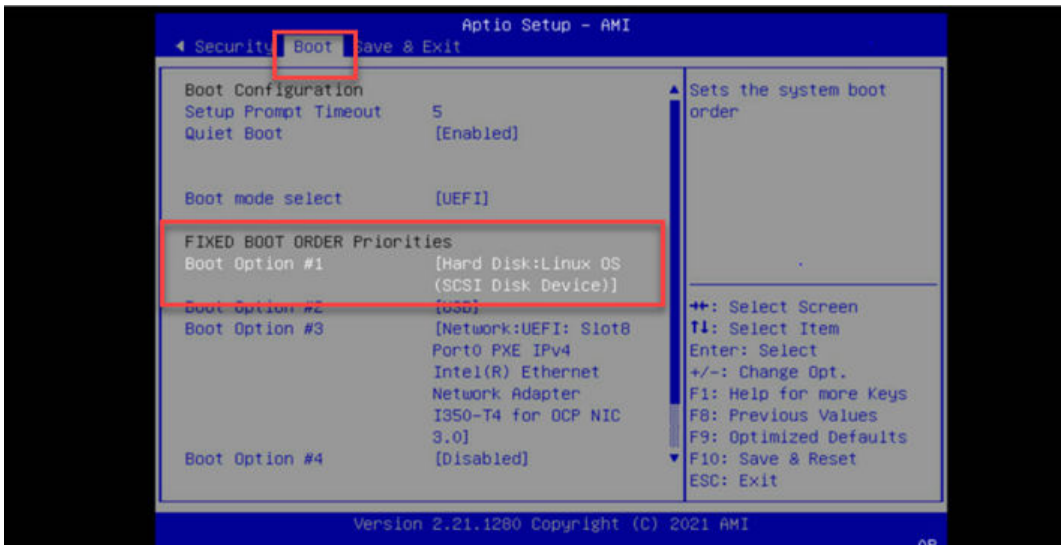
13. Click **FINISH CONFIGURATION**.



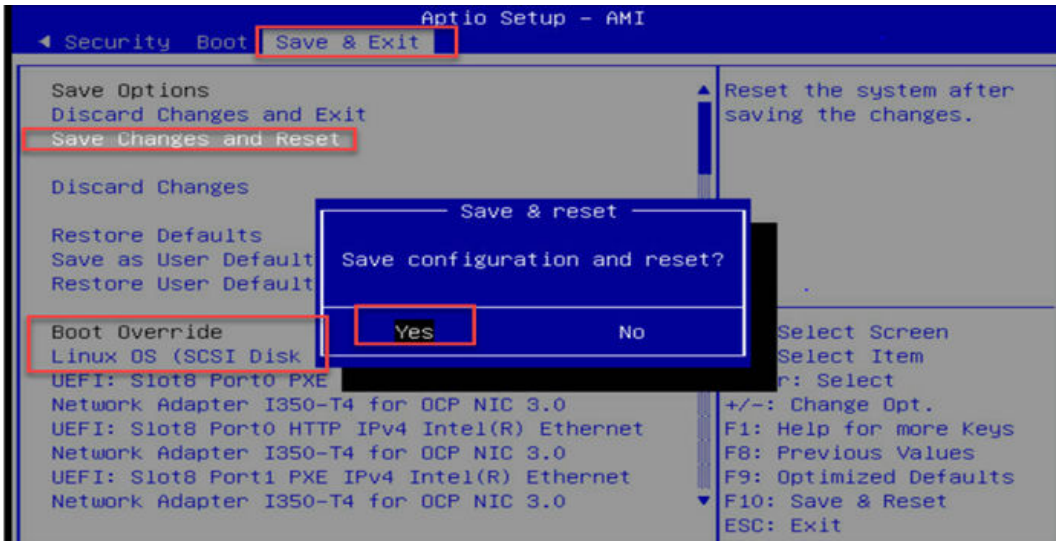
Oracle Linux 8 installation is complete and the following screen appears.



14. After installation, restart the node and enter the Boot menu by pressing F11 to change the Boot order from **CDROM** to **Disk**.



15. Select **Linux OS (SCSI Disk)** and then **Save & Reset**.



Perform LUN discovery

After the KVM host is ready, verify that the boot LUN and other mapped LUNs are visible on both KVM hosts.

Procedure

1. To discover the mapped LUNs, run the following commands.

```
[root@hpeorakvm1 ~]# /usr/bin/rescan-scsi-bus.sh
Scanning SCSI subsystem for new devices
Scanning host 0 for  all SCSI target IDs, all LUNs
 Scanning for device 0 0 0 48 ...
OLD: Host: scsi0 Channel: 00 Id: 00 Lun: 48
      Vendor: HITACHI  Model: OPEN-V           Rev: 9301
      Type:   Direct-Access                    ANSI SCSI revision: 03
 Scanning for device 0 0 0 38 ...
OLD: Host: scsi0 Channel: 00 Id: 00 Lun: 38
      Vendor: HITACHI  Model: OPEN-V           Rev: 9301
      Type:   Direct-Access                    ANSI SCSI revision: 03
 Scanning for device 0 0 0 28 ...
OLD: Host: scsi0 Channel: 00 Id: 00 Lun: 28
      Vendor: HITACHI  Model: OPEN-V           Rev: 9301
      Type:   Direct-Access                    ANSI SCSI revision: 03
 Scanning for device 0 0 0 46 ...
```

```
[root@ hpeorakvm1~]# lsscsi
[0:0:0:0]    cd/dvd  AMI      Virtual CDROM0   1.00  /dev/sr0
[1:1:123:0]  enclosu QCT      D52BQ-2U         0340  -
[3:0:0:0]    disk    AMI      Virtual HDisk0   1.00  /dev/sda
[3:0:0:1]    disk    AMI      Virtual HDisk1   1.00  /dev/sdb
[6:0:0:0]    disk    HITACHI  OPEN-V           9301  /dev/sdc
[6:0:0:1]    disk    HITACHI  OPEN-V           9301  /dev/sdd
[6:0:0:2]    disk    HITACHI  OPEN-V           9301  /dev/sde
[6:0:0:3]    disk    HITACHI  OPEN-V           9301  /dev/sdf
[6:0:0:4]    disk    HITACHI  OPEN-V           9301  /dev/sdg
[6:0:0:5]    disk    HITACHI  OPEN-V           9301  /dev/sdh
[6:0:0:6]    disk    HITACHI  OPEN-V           9301  /dev/sdi
[6:0:0:7]    disk    HITACHI  OPEN-V           9301  /dev/sdj
[6:0:0:8]    disk    HITACHI  OPEN-V           9301  /dev/sdk
[6:0:0:9]    disk    HITACHI  OPEN-V           9301  /dev/sdl
[6:0:0:10]   disk    HITACHI  OPEN-V           9301  /dev/sdm
[6:0:0:11]   disk    HITACHI  OPEN-V           9301  /dev/sdn
[6:0:0:12]   disk    HITACHI  OPEN-V           9301  /dev/sdo
[6:0:0:13]   disk    HITACHI  OPEN-V           9301  /dev/sdbc
[6:0:0:14]   disk    HITACHI  OPEN-V           9301  /dev/sdbg
[6:0:0:15]   disk    HITACHI  OPEN-V           9301  /dev/sdbm
[6:0:0:16]   disk    HITACHI  OPEN-V           9301  /dev/sdbq
[6:0:0:17]   disk    HITACHI  OPEN-V           9301  /dev/sdbt
[6:0:0:18]   disk    HITACHI  OPEN-V           9301  /dev/sdby
[6:0:0:19]   disk    HITACHI  OPEN-V           9301  /dev/sdcc
[18:0:0:0]   disk    HITACHI  OPEN-V           9301  /dev/sdp
[18:0:0:1]   disk    HITACHI  OPEN-V           9301  /dev/sdq
[18:0:0:2]   disk    HITACHI  OPEN-V           9301  /dev/sdr
[18:0:0:3]   disk    HITACHI  OPEN-V           9301  /dev/sds
[18:0:0:4]   disk    HITACHI  OPEN-V           9301  /dev/sdt
[18:0:0:5]   disk    HITACHI  OPEN-V           9301  /dev/sdu
[18:0:0:6]   disk    HITACHI  OPEN-V           9301  /dev/sdv
[18:0:0:7]   disk    HITACHI  OPEN-V           9301  /dev/sdw
[18:0:0:8]   disk    HITACHI  OPEN-V           9301  /dev/sdx
[18:0:0:9]   disk    HITACHI  OPEN-V           9301  /dev/sdy
[18:0:0:10]  disk    HITACHI  OPEN-V           9301  /dev/sdz
[18:0:0:11]  disk    HITACHI  OPEN-V           9301  /dev/sdaa
[18:0:0:12]  disk    HITACHI  OPEN-V           9301  /dev/sdab
[18:0:0:13]  disk    HITACHI  OPEN-V           9301  /dev/sdbd
[18:0:0:14]  disk    HITACHI  OPEN-V           9301  /dev/sdbh
```

```
[18:0:0:15]  disk    HITACHI  OPEN-V           9301  /dev/sdbk
[18:0:0:16]  disk    HITACHI  OPEN-V           9301  /dev/sdbo
[18:0:0:17]  disk    HITACHI  OPEN-V           9301  /dev/sdbv
[18:0:0:18]  disk    HITACHI  OPEN-V           9301  /dev/sdbx
[18:0:0:19]  disk    HITACHI  OPEN-V           9301  /dev/sdcb
[19:0:0:0]   disk    HITACHI  OPEN-V           9301  /dev/sdac
[19:0:0:1]   disk    HITACHI  OPEN-V           9301  /dev/sdad
[19:0:0:2]   disk    HITACHI  OPEN-V           9301  /dev/sdae
[19:0:0:3]   disk    HITACHI  OPEN-V           9301  /dev/sdaf
[19:0:0:4]   disk    HITACHI  OPEN-V           9301  /dev/sdag
[19:0:0:5]   disk    HITACHI  OPEN-V           9301  /dev/sdah
[19:0:0:6]   disk    HITACHI  OPEN-V           9301  /dev/sdai
[19:0:0:7]   disk    HITACHI  OPEN-V           9301  /dev/sdaj
[19:0:0:8]   disk    HITACHI  OPEN-V           9301  /dev/sdak
[19:0:0:9]   disk    HITACHI  OPEN-V           9301  /dev/sdal
[19:0:0:10]  disk    HITACHI  OPEN-V           9301  /dev/sdam
[19:0:0:11]  disk    HITACHI  OPEN-V           9301  /dev/sdan
[19:0:0:12]  disk    HITACHI  OPEN-V           9301  /dev/sdao
[19:0:0:13]  disk    HITACHI  OPEN-V           9301  /dev/sdbf
[19:0:0:14]  disk    HITACHI  OPEN-V           9301  /dev/sdbj
[19:0:0:15]  disk    HITACHI  OPEN-V           9301  /dev/sdbn
[19:0:0:16]  disk    HITACHI  OPEN-V           9301  /dev/sdbr
[19:0:0:19]  disk    HITACHI  OPEN-V           9301  /dev/sdcd
[root@ig-virt01 ~]#
```

Configure a network for KVM hosts

To configure a resilient network, two dual port NICs on both KVM hosts are used to create network bonding.

The `nmcli` command line utility is used to create four network bonding interfaces (ens65f0, ens65f1, ens67f0, and ens67f1) followed by IP assignments for each configured bond.

Oracle Real Application Cluster Database requires the following separate networks:

- Private Network also called the cluster interconnect — This network must be scalable. In addition, it must meet the low latency needs of the network traffic generated by the cache synchronization of RAC clusters and inter-node communication among the nodes in the cluster.
- Public Network — This network provides client connections to the applications and Oracle Real Application Clusters.

The networks are configured as follows:

- A pair of 25 Gbps NICs are used for the private and public interconnect in this solution.
- Use NIC bonding to provide failover and load balancing of interconnections within a server.
- Set all NICs to full duplex mode.
- Configure network bonding as follows:
  - `ens65f0 + ens67f1` □ bond0 – public network
  - `ens65f1 + ens67f0` □ bond1 – private network

Procedure

1. Run the following commands to configure public IP addresses.

```
# nmcli connection show
# nmcli dev status
# nmcli connection add type bond con-name bond0 ifname bond0 bond.options
"mode=active-backup"
# nmcli con add type ethernet slave-type bond con-name bond0:1 ifname ens65f0
master bond0
# nmcli con add type ethernet slave-type bond con-name bond0:2 ifname ens67f1
master bond0
# nmcli con mod bond0 ipv4.addresses "10.76.33.94/24"   <<<<< Put IP address
which should be resolve on DNS.
# nmcli con mod bond0 ipv4.gateway 10.76.33.1
# nmcli con mod bond0 ipv4.dns "10.33.32.111"
# nmcli con mod bond0 ipv4.method manual
# nmcli con up bond0
```

Note: When creating NIC bonding pairs, ports should be used on different cards to avoid single points of failure.

2. Run the following commands to configure private IP addresses.

```
# nmcli dev status
# nmcli connection add type bond con-name bond1 ifname bond1 bond.options
"mode=active-backup"
# nmcli con add type ethernet slave-type bond con-name bond1:1 ifname ens65f1
master bond1
# nmcli con add type ethernet slave-type bond con-name bond1:2 ifname ens67f0
master bond1
# nmcli con mod bond1 ipv4.addresses 192.168.1.94/24" <<<< This is a private IP
so give any IP from a private IP range.
# nmcli con mod bond1 ipv4.gateway 192.168.1.1
# nmcli con mod bond1 ipv4.method manual
# nmcli con up bond1
```

See *Configure network bonding* at https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/configuring_and_managing_networking/configuring-network-bonding_configuring-and-managing-networking for details.

> **Note:** A DNS entry is not required for private IP addresses (that is, the Bond1 private network).

See the activity log for details.

```
[root@ig-virt01 network-scripts]#
[root@ig-virt01 network-scripts]# nmcli connection add type bond con-name bond0 ifname bond0 bond.options "mode=active-backup"
Connection 'bond0' (7d1895ed-ad1a-4678-9068-7045bffbd678) successfully added.
[root@ig-virt01 network-scripts]#
[root@ig-virt01 network-scripts]# nmcli con add type ethernet slave-type bond con-name bond0:1 ifname ens65f0 master bond0
Connection 'bond0:1' (5dc139c5-7541-4c70-9534-b0552aac6a64) successfully added.
[root@ig-virt01 network-scripts]#
[root@ig-virt01 network-scripts]# nmcli con add type ethernet slave-type bond con-name bond0:2 ifname ens67f1 master bond0
Connection 'bond0:2' (5518062b-96f8-4b88-9ec5-337c3f4c37b7) successfully added.
[root@ig-virt01 network-scripts]#
[root@ig-virt01 network-scripts]# nmcli con mod bond0 ipv4.addresses "10.76.33.120/24"
[root@ig-virt01 network-scripts]# nmcli con mod bond0 ipv4.gateway 10.76.33.1
[root@ig-virt01 network-scripts]# nmcli con mod bond0 ipv4.dns "10.76.32.111"
[root@ig-virt01 network-scripts]# nmcli con mod bond0 ipv4.method manual
[root@ig-virt01 network-scripts]# nmcli con modify bond0 connection.autoconnect-slaves 1
[root@ig-virt01 network-scripts]# nmcli con up bond0
Connection successfully activated (master waiting for slaves) (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/18)
[root@ig-virt01 network-scripts]#
```

```
[root@ig-virt01 network-scripts]# nmcli con show
NAME        UUID                                  TYPE      DEVICE
bond0       7d1895ed-ad1a-4678-9068-7045bffbd678  bond      bond0
ovirtmgmt   61e30133-b413-4c19-bc76-abdc211abfa1  bridge    ovirtmgmt
bond0:1     5dc139c5-7541-4c70-9534-b0552aac6a64  ethernet  ens65f0
bond0:2     5518062b-96f8-4b88-9ec5-337c3f4c37b7  ethernet  ens67f1
ens67f3     82d8a653-c22f-4192-82ff-2e22a08cc648  ethernet  ens67f3
ens8f0      342cd06e-4245-4244-9e54-ac686a47fca6  ethernet  ens8f0
ens65f0     2a293831-4730-e3b0-bcfb-3e59e095e907  ethernet  --
ens65f1     6489a246-cf0e-4a90-8529-a462b4046902  ethernet  --
ens65f2     4dfa47ba-85a6-458e-bcb7-ba8619f5f615  ethernet  --
ens65f3     19b85517-6c1b-4cdb-8301-299d9439864e  ethernet  --
ens67f0     3c84d215-54f5-4469-afc9-3505e6a93178  ethernet  --
ens67f1     17fb7998-6166-9023-36f6-0f0e60c660c0  ethernet  --
ens67f2     c612d65f-7d4e-4673-b8e3-de9c9996ec42  ethernet  --
ens8f1      6c3cbf98-fcde-496f-aa1f-830335e0bca7  ethernet  --
ens8f2      3303ebaa-deaf-4630-9b39-f585a32a6951  ethernet  --
ens8f3      f6962890-c4ae-4007-95ef-b0fea689ff50  ethernet  --
usb0        7abf5f51-2981-4e8b-8553-39c32ca91035  ethernet  --
[root@ig-virt01 network-scripts]#
```

```
[root@ig-virt01 network-scripts]#
[root@ig-virt01 network-scripts]# nmcli connection add type bond con-name bond1 ifname bond1 bond.options "mode=active-backup"
Connection 'bond1' (bf92e998-1b44-410f-9c97-9f9a83713310) successfully added.
[root@ig-virt01 network-scripts]# nmcli con add type ethernet slave-type bond con-name bond1:1 ifname ens65f1 master bond1
Connection 'bond1:1' (8cf5557a-d5c7-47d4-9825-1a97c7ea1089) successfully added.
[root@ig-virt01 network-scripts]# nmcli con add type ethernet slave-type bond con-name bond1:2 ifname ens67f0 master bond1
Connection 'bond1:2' (27fdec12-43ad-4a7e-a093-2a89e7e7769f) successfully added.
[root@ig-virt01 network-scripts]# nmcli con mod bond1 ipv4.addresses "192.168.1.201/24"
[root@ig-virt01 network-scripts]# nmcli con mod bond1 ipv4.gateway 192.168.1.1
[root@ig-virt01 network-scripts]# nmcli con mod bond1 ipv4.method manual
[root@ig-virt01 network-scripts]# nmcli con modify bond1 connection.autoconnect-slaves 1
[root@ig-virt01 network-scripts]# nmcli con up bond1
Connection successfully activated (master waiting for slaves) (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/34)
[root@ig-virt01 network-scripts]#
[root@ig-virt01 network-scripts]# nmcli dev status
DEVICE          TYPE      STATE         CONNECTION
bond1           bond      connected     bond1
ovirtmgmt       bridge    connected     ovirtmgmt
ovirtDm220Pub   bridge    connected     ovirtDm220Pub
bond0           bond      connected     bond0
ens65f0         ethernet  connected     bond0:1
ens65f1         ethernet  connected     bond1:1
ens67f0         ethernet  connected     bond1:2
ens67f1         ethernet  connected     bond0:2
ens67f3         ethernet  connected     ens67f3
ens8f0          ethernet  connected     ens8f0
ens65f2         ethernet  disconnected  --
ens65f3         ethernet  disconnected  --
ens67f2         ethernet  disconnected  --
ens8f1          ethernet  disconnected  --
ens8f2          ethernet  disconnected  --
ens8f3          ethernet  disconnected  --
usb0            ethernet  disconnected  --
virbr0          bridge    unmanaged     --
lo              loopback  unmanaged     --
[root@ig-virt01 network-scripts]#
```

Chapter 9: Virtualization configuration

To virtualize bare metal hosts configure VMs on them, we need software that virtualizes the hosts. In this implementation guide we used the Oracle KVM hypervisor software on top of Oracle Linux 8.6. See the *Oracle Linux KVM User's Guide* at https://docs.oracle.com/en/operating-systems/oracle-linux/kvm-user/ for more details.

Oracle Linux KVM

The Kernel-based Virtual Machine (KVM) is opensource software. KVM is a full virtualization solution for Linux on x86_64 hardware containing virtualization extensions (Intel VT or AMD-V). It consists of a loadable kernel module, `kvm.ko`, that provides the core virtualization infrastructure and a processor specific module, `kvm-intel.ko` or `kvm-amd.ko`.

The KVM feature provides a set of modules that enable you to use the Oracle Linux kernel as a hypervisor. KVM supports both x86_64 and aarch64 processor architectures and is supported on Oracle Linux 7 and Oracle Linux 8 operating systems using either RHCK or any UEK release as of Unbreakable Enterprise Kernel Release 4.

Using KVM, you can run multiple virtual machines running unmodified Linux or Windows images. Each virtual machine has private virtualized hardware: a network card, disk, and graphics adapter. The kernel component of KVM is included in mainline Linux as of release 2.6.20.

Virtualization packages

Oracle Linux provides several virtualization packages that enable you work with KVM. You can install virtualization packages from the Oracle Linux YUM server or from the Unbreakable Linux Network (ULN). In most cases, the following packages are the minimum required for a virtualization host:

- `libvirt` — This package provides an interface to KVM, as well as the `libvirtd` daemon for managing guest virtual machine
- `qemu-kvm` — This package installs the QEMU emulator that performs hardware virtualization so that guests can access host CPU and other resources.
- `virt-install` — This package provides command line utilities for creating and provisioning guest virtual machines.
- `virt-viewer` — This package provides a graphical utility that can be loaded into a desktop environment to access the graphical console of a guest virtual machine.

See the following references for more information:

- https://www.linux-kvm.org/page/Main_Page
- https://libvirt.org/
- https://www.qemu.org

Install KVM on Oracle Linux 8

Before you begin

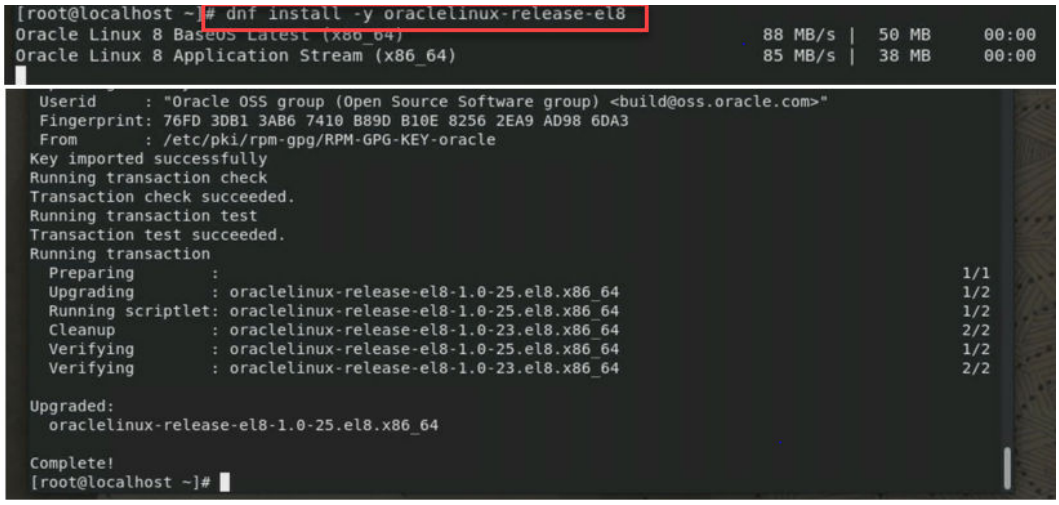
Verify that your system has the correct YUM repository or ULN channel enabled for the virtualization package versions that you want to install.

Procedure

1. Log in as the root user on the target Oracle Linux system.
2. For Oracle Linux 8 run the following commands to install the base virtualization packages and additional utilities.

```
# dnf install -y oraclelinux-release-el8
# dnf config-manager --enable ol8_appstream ol8_kvm_appstream
# dnf update
# dnf module install virt
# dnf install virt-install virt-viewer
# systemctl enable libvirtd
# systemctl start libvirtd.service
# systemctl status libvirtd
# virt-host-validate qemu
# yum repolist all
# yum repolist all|grep -I ol8_UEKR6
# dnf config-manager -enable ol8_UEKR7
# dnf update -y
```

3. The following examples show activity logs for reference.



```
[root@localhost ~]# dnf config-manager --enable ol8_appstream ol8_kvm_appstream
[root@localhost ~]#
```

```
[root@localhost ~]# dnf update
Last metadata expiration check: 0:08:20 ago on Thu 29 Sep 2022 02:05:18 AM EDT.
Dependencies resolved.
================================================================================================================================
 Package                                  Architecture    Version                                                   Repository                  Size
================================================================================================================================
Installing:
 kernel                                   x86_64          4.18.0-372.26.1.0.1.el8_6                                 ol8_baseos_latest          8.1 M
Upgrading:
 NetworkManager                           x86_64          1:1.36.0-7.0.1.el8_6                                      ol8_baseos_latest          2.3 M
 NetworkManager-adsl                      x86_64          1:1.36.0-7.0.1.el8_6                                      ol8_baseos_latest          149 k
 NetworkManager-bluetooth                 x86_64          1:1.36.0-7.0.1.el8_6                                      ol8_baseos_latest          175 k
 NetworkManager-config-server             noarch          1:1.36.0-7.0.1.el8_6                                      ol8_baseos_latest          136 k
 NetworkManager-libnm                     x86_64          1:1.36.0-7.0.1.el8_6                                      ol8_baseos_latest          1.8 M
 NetworkManager-team                      x86_64          1:1.36.0-7.0.1.el8_6                                      ol8_baseos_latest          153 k
 NetworkManager-tui                       x86_64          1:1.36.0-7.0.1.el8_6                                      ol8_baseos_latest          346 k
 NetworkManager-wifi                      x86_64          1:1.36.0-7.0.1.el8_6                                      ol8_baseos_latest          194 k
 NetworkManager-wwan                      x86_64          1:1.36.0-7.0.1.el8_6                                      ol8_baseos_latest          181 k
 alsa-sof-firmware                        noarch          1.9.3-4.el8_6                                             ol8_baseos_latest          781 k
 bash                                     x86_64          4.4.20-4.el8_6                                            ol8_baseos_latest          1.5 M
 binutils                                 x86_64          2.30-113.0.3.el8                                          ol8_baseos_latest          5.9 M
 bpftool                                  x86_64          4.18.0-372.26.1.0.1.el8_6                                 ol8_baseos_latest          8.8 M
 btrfs-progs                              x86_64          5.15.1-0.el8                                              ol8_UEKR6                  866 k
 sssd-common-2.6.2-4.0.2.el8_6.1.x86_64                   sssd-common-pac-2.6.2-4.0.2.el8_6.1.x86_64
 sssd-ipa-2.6.2-4.0.2.el8_6.1.x86_64                      sssd-kcm-2.6.2-4.0.2.el8_6.1.x86_64
 sssd-krb5-2.6.2-4.0.2.el8_6.1.x86_64                     sssd-krb5-common-2.6.2-4.0.2.el8_6.1.x86_64
 sssd-ldap-2.6.2-4.0.2.el8_6.1.x86_64                     sssd-nfs-idmap-2.6.2-4.0.2.el8_6.1.x86_64
 sssd-proxy-2.6.2-4.0.2.el8_6.1.x86_64                    swtpm-0.7.0-3.20211109gitb79fd91.module+el8.6.0+20743+999ad699.x86_64
 swtpm-libs-0.7.0-3.20211109gitb79fd91.module+el8.6.0+20743+999ad699.x86_64   swtpm-tools-0.7.0-3.20211109gitb79fd91.module+el8.6.0+20743+999ad699.x86_64
 systemd-239-58.0.1.el8_6.7.x86_64                        systemd-container-239-58.0.1.el8_6.7.x86_64
 systemd-libs-239-58.0.1.el8_6.7.x86_64                   systemd-pam-239-58.0.1.el8_6.7.x86_64
 systemd-udev-239-58.0.1.el8_6.7.x86_64                   tuned-2.18.0-2.0.1.el8_6.1.noarch
 tzdata-2022c-1.el8.noarch                                unzip-6.0-46.0.1.el8.x86_64
 vim-common-2:8.0.1763-19.0.1.el8_6.4.x86_64              vim-enhanced-2:8.0.1763-19.0.1.el8_6.4.x86_64
 vim-filesystem-2:8.0.1763-19.0.1.el8_6.4.noarch          vim-minimal-2:8.0.1763-19.0.1.el8_6.4.x86_64
 webkit2gtk3-2.36.7-1.el8_6.x86_64                        webkit2gtk3-jsc-2.36.7-1.el8_6.x86_64
 xfsprogs-5.4.0-1.0.3.el8.x86_64                          xorg-x11-server-Xorg-1.20.11-5.el8_6.2.x86_64
 xorg-x11-server-Xwayland-21.1.3-2.el8_6.3.x86_64         xorg-x11-server-common-1.20.11-5.el8_6.2.x86_64
 xz-5.2.4-4.el8_6.x86_64                                  xz-libs-5.2.4-4.el8_6.x86_64
Installed:
 grub2-tools-efi-1:2.02-123.0.9.el8_6.8.x86_64      kernel-4.18.0-372.26.1.0.1.el8_6.x86_64      kernel-core-4.18.0-372.26.1.0.1.el8_6.x86_64
 kernel-modules-4.18.0-372.26.1.0.1.el8_6.x86_64    kernel-uek-5.4.17-2136.311.6.1.el8uek.x86_64 libatomic-8.5.0-10.1.0.1.el8_6.x86_64

Complete!
[root@localhost ~]#
```

```
[root@localhost ~]# dnf module install virt
Oracle Linux 8 BaseOS Latest (x86_64)                                          136 kB/s | 3.6 kB     00:00
Oracle Linux 8 Application Stream (x86_64)                                     322 kB/s | 3.9 kB     00:00
Oracle Linux 8 KVM Application Stream (x86_64)                                 7.1 MB/s | 319 kB     00:00
Dependencies resolved.
================================================================================================================================
 Package                                  Architecture    Version                                                   Repository                  Size
================================================================================================================================
Upgrading:
 libvirt-daemon                           x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream              419 k
 libvirt-daemon-config-network            x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream               67 k
 libvirt-daemon-driver-interface          x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream              209 k
 libvirt-daemon-driver-network            x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream              235 k
 libvirt-daemon-driver-nodedev            x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream              220 k
 libvirt-daemon-driver-nwfilter           x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream              235 k
 libvirt-daemon-driver-qemu               x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream              923 k
 libvirt-daemon-driver-secret             x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream              198 k
 libvirt-daemon-driver-storage            x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream               66 k
 libvirt-daemon-driver-storage-core       x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream              253 k
 libvirt-daemon-driver-storage-disk       x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream               76 k
 libvirt-daemon-driver-storage-gluster    x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream               78 k
 libvirt-daemon-driver-storage-iscsi      x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream               73 k
 libvirt-daemon-driver-storage-iscsi-direct x86_64        8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream               75 k
 libvirt-daemon-driver-storage-logical    x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream               77 k
 libvirt-daemon-driver-storage-mpath      x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream               70 k
 libvirt-daemon-driver-storage-rbd        x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream               81 k
 libvirt-daemon-driver-storage-scsi       x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream               73 k
 libvirt-daemon-kvm                       x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream               65 k
 libvirt-libs                             x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream              4.7 M
Installing group/module packages:
 libguestfs                               x86_64          1:1.44.0-5.0.1.module+el8.6.0+20659+3dcf7c70              ol8_appstream              769 k
 libvirt-client                           x86_64          8.0.0-5.4.0.1.module+el8.6.0+20743+999ad699               ol8_appstream              414 k
Installing dependencies:
 bind-export-libs                         x86_64          32:9.11.36-3.el8                                          ol8_baseos_latest          1.1 M
 dhcp-client                              x86_64          12:4.3.6-47.0.1.el8                                       ol8_baseos_latest          318 k
 dhcp-common                              noarch          12:4.3.6-47.0.1.el8                                       ol8_baseos_latest          207 k
 dhcp-libs                                x86_64          12:4.3.6-47.0.1.el8                                       ol8_baseos_latest          148 k
 hivex                                    x86_64          1.3.18-23.module+el8.6.0+20659+3dcf7c70                   ol8_appstream              113 k
 ipcalc                                   x86_64          0.2.4-4.el8                                               ol8_baseos_latest           38 k
 libguestfs-appliance                     x86_64          1:1.44.0-5.0.1.module+el8.6.0+20659+3dcf7c70              ol8_appstream              2.1 M
 scrub                                    x86_64          2.5.2-16.el8                                              ol8_appstream               44 k
 supermin                                 x86_64          5.2.1-1.module+el8.6.0+20659+3dcf7c70                     ol8_appstream              713 k
 syslinux                                 x86_64          6.04-5.el8                                                ol8_baseos_latest          578 k
 syslinux-extlinux                        x86_64          6.04-5.el8                                                ol8_baseos_latest          140 k
 syslinux-extlinux-nonlinux               noarch          6.04-5.el8                                                ol8_baseos_latest          386 k
 syslinux-nonlinux                        noarch          6.04-5.el8                                                ol8_baseos_latest          553 k
Installing module profiles:
 virt/common

Transaction Summary
================================================================================================================================
Install  15 Packages
Upgrade  20 Packages

Total download size: 16 M
Is this ok [y/N]:
```

```
[root@localhost ~]# dnf install virt-install virt-viewer
Last metadata expiration check: 0:06:26 ago on Thu 29 Sep 2022 02:05:13 AM EDT.
Dependencies resolved.
================================================================================================================================
 Package                        Architecture          Version                                          Repository                      Size
================================================================================================================================
Installing:
 virt-install                   noarch                3.2.0-5.el8                                      ol8_kvm_appstream               40 k
 virt-viewer                    x86_64                9.0-12.el8                                       ol8_appstream                  426 k
Installing dependencies:
 libgovirt                      x86_64                0.3.7-4.el8                                      ol8_appstream                   86 k
 python3-argcomplete            noarch                1.9.3-6.el8                                      ol8_appstream                   60 k
 python3-libvirt                x86_64                8.0.0-1.1.module+el8.6.0+20743+999ad699          ol8_appstream                  332 k
 virt-manager-common            noarch                3.2.0-5.el8                                      ol8_kvm_appstream              1.0 M

Transaction Summary
================================================================================================================================
Install  6 Packages

Total download size: 1.9 M
Installed size: 8.8 M
Is this ok [y/N]:
Downloading Packages:
(1/6): python3-argcomplete-1.9.3-6.el8.noarch.rpm                                   1.3 MB/s |  60 kB     00:00
(2/6): libgovirt-0.3.7-4.el8.x86_64.rpm                                             1.8 MB/s |  86 kB     00:00
(3/6): python3-libvirt-8.0.0-1.1.module+el8.6.0+20743+999ad699.x86_64.rpm           6.1 MB/s | 332 kB     00:00
(4/6): virt-install-3.2.0-5.el8.noarch.rpm                                           38 kB/s |  40 kB     00:01
(5/6): virt-viewer-9.0-12.el8.x86_64.rpm                                            322 kB/s | 426 kB     00:01
(6/6): virt-manager-common-3.2.0-5.el8.noarch.rpm                                   605 kB/s | 1.0 MB     00:01
--------------------------------------------------------------------------------------------------------------------------------
Total                                                                               1.1 MB/s | 1.9 MB     00:01
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                          1/1
  Installing       : python3-libvirt-8.0.0-1.1.module+el8.6.0+20743+999ad699.x86_64                                           1/6
  Installing       : python3-argcomplete-1.9.3-6.el8.noarch                                                                   2/6
  Installing       : virt-manager-common-3.2.0-5.el8.noarch                                                                   3/6
  Installing       : libgovirt-0.3.7-4.el8.x86_64                                                                             4/6
  Installing       : virt-viewer-9.0-12.el8.x86_64                                                                            5/6
  Installing       : virt-install-3.2.0-5.el8.noarch                                                                          6/6
  Running scriptlet: virt-install-3.2.0-5.el8.noarch                                                                          6/6
  Verifying        : libgovirt-0.3.7-4.el8.x86_64                                                                             1/6
  Verifying        : python3-argcomplete-1.9.3-6.el8.noarch                                                                   2/6
  Verifying        : python3-libvirt-8.0.0-1.1.module+el8.6.0+20743+999ad699.x86_64                                           3/6
  Verifying        : virt-viewer-9.0-12.el8.x86_64                                                                            4/6
  Verifying        : virt-install-3.2.0-5.el8.noarch                                                                          5/6
  Verifying        : virt-manager-common-3.2.0-5.el8.noarch                                                                   6/6

Installed:
  libgovirt-0.3.7-4.el8.x86_64          python3-argcomplete-1.9.3-6.el8.noarch        python3-libvirt-8.0.0-1.1.module+el8.6.0+20743+999ad699.x86_64
  virt-install-3.2.0-5.el8.noarch       virt-manager-common-3.2.0-5.el8.noarch        virt-viewer-9.0-12.el8.x86_64

Complete!
[root@localhost ~]#
```

All the rpms required for KVM are installed.

4. Enable and start the `libvirtd` daemon to start KVM services.

```
[root@localhost ~]# systemctl start libvirtd
[root@localhost ~]# systemctl status libvirtd
● libvirtd.service - Virtualization daemon
   Loaded: loaded (/usr/lib/systemd/system/libvirtd.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2022-09-29 02:23:13 EDT; 6s ago
     Docs: man:libvirtd(8)
           https://libvirt.org
Main PID: 169099 (libvirtd)
    Tasks: 21 (limit: 32768)
   Memory: 48.0M
   CGroup: /system.slice/libvirtd.service
           ├─ 5729 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-script=/usr/libexec/libvirt_leaseshelper
           ├─ 5730 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-script=/usr/libexec/libvirt_leaseshelper
           └169099 /usr/sbin/libvirtd --timeout 120

Sep 29 02:23:13 localhost.localdomain systemd[1]: Starting Virtualization daemon...
Sep 29 02:23:13 localhost.localdomain systemd[1]: Started Virtualization daemon.
Sep 29 02:23:13 localhost.localdomain dnsmasq[5729]: read /etc/hosts - 2 addresses
Sep 29 02:23:13 localhost.localdomain dnsmasq[5729]: read /var/lib/libvirt/dnsmasq/default.addnhosts - 0 addresses
Sep 29 02:23:13 localhost.localdomain dnsmasq-dhcp[5729]: read /var/lib/libvirt/dnsmasq/default.hostsfile
[root@localhost ~]#
```

5. Check the status of `qemu`.

```
[root@localhost ~]# virt-host-validate qemu
  QEMU: Checking for hardware virtualization                                 : PASS
  QEMU: Checking if device /dev/kvm exists                                   : PASS
  QEMU: Checking if device /dev/kvm is accessible                            : PASS
  QEMU: Checking if device /dev/vhost-net exists                             : PASS
  QEMU: Checking if device /dev/net/tun exists                               : PASS
  QEMU: Checking for cgroup 'cpu' controller support                         : PASS
  QEMU: Checking for cgroup 'cpuacct' controller support                     : PASS
  QEMU: Checking for cgroup 'cpuset' controller support                      : PASS
  QEMU: Checking for cgroup 'memory' controller support                      : PASS
  QEMU: Checking for cgroup 'devices' controller support                     : PASS
  QEMU: Checking for cgroup 'blkio' controller support                       : PASS
  QEMU: Checking for device assignment IOMMU support                         : PASS
  QEMU: Checking if IOMMU is enabled by kernel                               : PASS
  QEMU: Checking for secure guest support                                    : WARN (Unknown if this platform has Secure Guest support)
[root@localhost ~]#
```

Switch to the Oracle KVM stack

On an existing Oracle Linux 8 system, you can switch from the default KVM stack to the Oracle KVM stack in the `virt:kvm_utils` stream by running the following commands:

```
# sudo dnf module remove virt -y –all   <<<<< Remove any packages from the existing
default virt stream
# sudo dnf module reset virt -y   <<<<< Reset the virt module state so that it is
neither enabled nor disabled
# sudo dnf module enable virt:kvm_utils -y   <<<<< Enable the virt:kvm_utils module
and stream
# sudo dnf --allowerasing distro-sync   <<<<< Perform any necessary package upgrade
or downgrade operations to handle dependencies for the enabled module and stream
# sudo dnf module install virt:kvm_utils -y   <<<<< Install the base packages from
the virt:kvm_utils stream
```

Note: Although you can switch to the Oracle KVM stack and install the packages while using RHCK, the stack is not compatible. You must be running a current version of UEK to use this software.

With this installation, node 1 is ready with Oracle Linux 8.6 and KVM hypervisor installed on it to make the host virtualized so we can create multiple VMs and share resources among them.

Note: Repeat the previous steps on node 2 as well.

The following table shows progress to this point.

Sr.no.	Task Description	Node 1	Node 2	Status
1.	Hardware pre-checks	Myhost1	Myhost2	Done
2.	Configure LUN (storage)	Boot LUN	Boot LUN	Done
3.	Configure Zoning	✓	✓	Done
4.	OS (OL8.6) installation on bare metal host	✓	✓	Done
5.	Configure network bonding and assign IP addresses (Public/Private)	✓	✓	Done
6.	Install Oracle Linux KVM hypervisor	✓	✓	Done

OLVM Management host

When both nodes are ready for virtualization, we must create another host for Oracle Linux virtualization manager which will act as a management agent to manage both KVM hosts, VMs, and all other resources running on them.

See Oracle Linux Virtualization Manager (OLVM) (on page 46) for the next steps.

Chapter 10: Oracle Linux Virtualization Manager (OLVM)

Oracle Linux Virtualization Manager introduction and requirements

Oracle Linux Virtualization Manager (OLVM) is a management server that manages KVM stacks. It creates and allocates resources and performs maintenance activities.

OLVM is a server virtualization management platform based on the open source oVirt project. It is used to configure, monitor, and manage an Oracle Linux KVM environment, including hosts, virtual machines, storage, networks, and users. You can access OLVM through the Administration Portal or VM Portal.

OLVM also provides a Representational State Transfer (REST) Application Programming Interface (API) for managing your KVM infrastructure, allowing you to integrate OLVM with other management systems or to automate repetitive tasks with scripts.

To install Oracle Linux Virtualization Manager, we performed a fresh installation of Oracle Linux 8.6 on a separate host, installed the **ovirt-engine** package, and then ran the `engine-setup` command to configure OLVM.

Install the OS (Oracle Linux 8.6)

Download the installation ISO for Oracle Linux 8.6 from the Oracle Software Delivery Cloud at https://edelivery.oracle.com. See the section titled *Install the OS on Bare Metal Hosts*.

Install the OLVM engine

The main component of Oracle Linux Virtualization Manager is the **oVirt engine** (engine), which is a JBoss-based Java application that runs as a web service and provides centralized management for server and desktop virtualization. The engine provides many features including:

- Managing the Oracle Linux KVM hosts.
- Creating, deploying, starting, stopping, migrating, and monitoring virtual machines.
- Adding and managing logical networks.
- Adding and managing storage domains and virtual disks.

- Configuring and managing cluster, host, and virtual machine high availability.
- Migrating and editing live virtual machines.
- Continuously balancing loads on virtual machines based on resource usage and policies.
- Monitoring all objects in the environment such as virtual machines, hosts, storage, networks.

See the OLVM installation Guide for details.

Run the following commands to install and configure the engine.

```
# dnf config-manager --enable ol8_baseos_latest
# dnf install oracle-ovirt-release-el8
# dnf clean all
# dnf repolist
# dnf install ovirt-engine
```

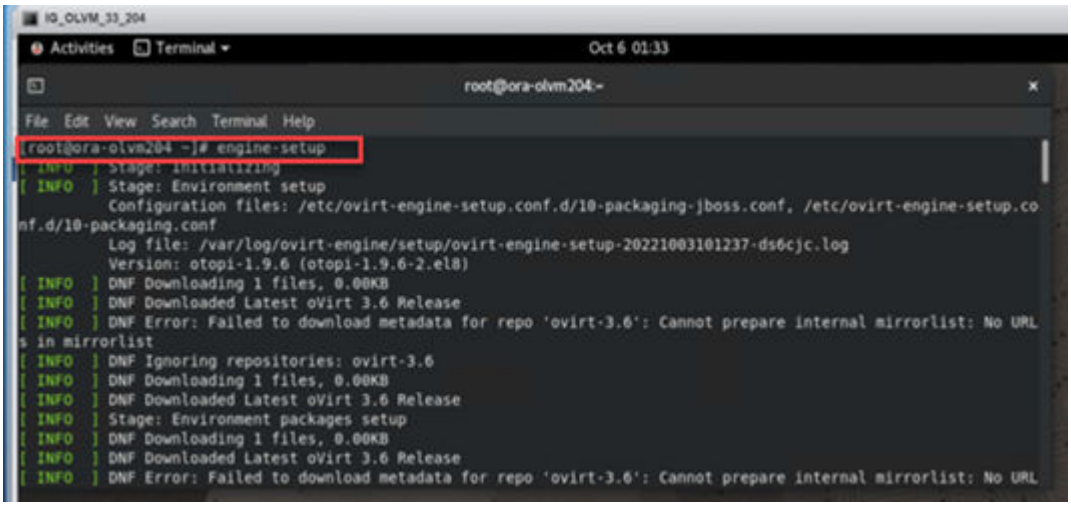
Configure the OLVM engine

After you install the OLVM engine, run the `engine-setup` command to configure the Manager, which sends a series of prompts.

Procedure

1. Log in to the host using **root** credentials through the GUI.
2. Open a terminal session and run the following commands:

```
[root@olvmhost]# cd /etc/ovirt-engine-setup.conf.d
[root@olvmhost]# ./engine-setup
[root@olvmhost]#Configure Engine on this host (Yes, No) [Yes]:
```



3. After answering the prompts, Setup displays a list of the values you entered. Review the list carefully and then press **Enter** to configure the Manager.
4. When the configuration is complete, details about how to log in to the Administration Portal are displayed.
5. Image I/O Proxy: The Image I/O Proxy (`ovirt-imageio-proxy`) enables you to upload virtual disks into storage domains.

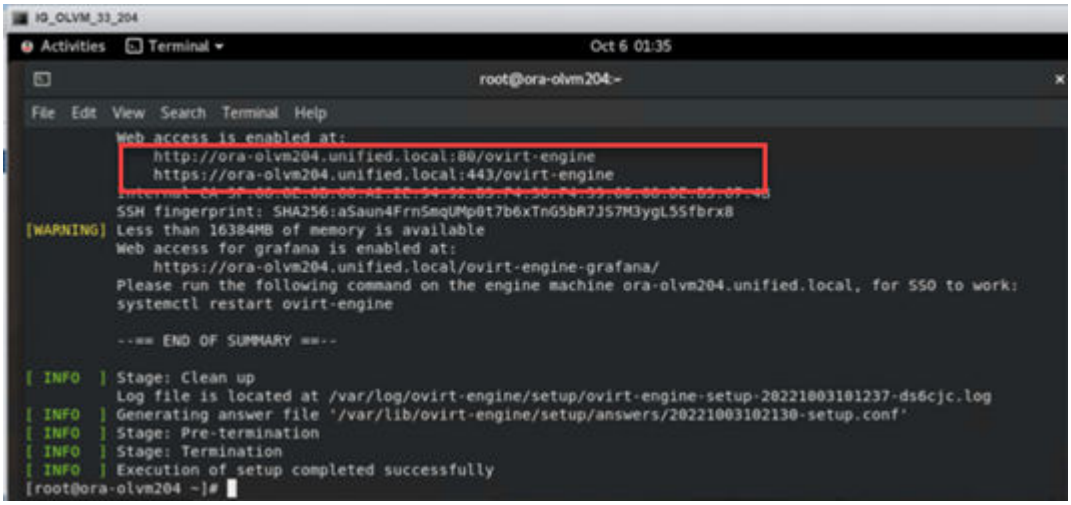
```
# systemctl status ovirt-imageio-daemon
```

6. After the installation finishes a summary of Web URLs that can be used to log in to the administration portal is provided. Note the URL details.

   The following shows an activity log for reference.

```
---== SUMMARY ==---
[ INFO  ] Restarting httpd
          Please use the user 'admin@internal' and password specified in order
to login
          Web access is enabled at:
          http://olvmhost.unified.local:80/ovirt-engine
         https://olvmhost.unified.local:443/ovirt-engine
          Internal CA 3F:68:8E:0B:68:A2:2E:94:92:B3:F4:36:F4:39:00:08:DE:B3:67:4B
          SSH fingerprint: SHA256:aSaun4FrnSmqUMp0t7b6xTnG5bR7JS7M3ygL5Sfbrx8
[WARNING] Less than 16384MB of memory is available
          Web access for grafana is enabled at:
              https://olvmhost.unified.local/ovirt-engine-grafana/
          Please run the following command on the engine machine
olvmhost.unified.local, for SSO to work:
          systemctl restart ovirt-engine

          ---== END OF SUMMARY ==--

[ INFO  ] Stage: Clean up
          Log file is located at /var/log/ovirt-engine/setup/ovirt-engine-setup-
20221003101237-ds6cjc.log
[ INFO  ] Generating answer file '/var/lib/ovirt-engine/setup/answers/
```

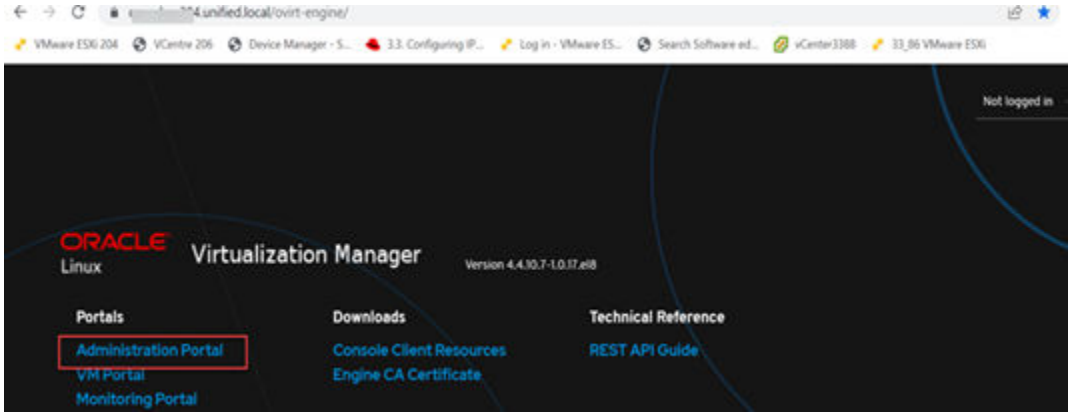
```
20221003102130-setup.conf'
[ INFO  ] Stage: Pre-termination
[ INFO  ] Stage: Termination
[ INFO  ] Execution of setup completed successfully
[root@olvmhost]#
```

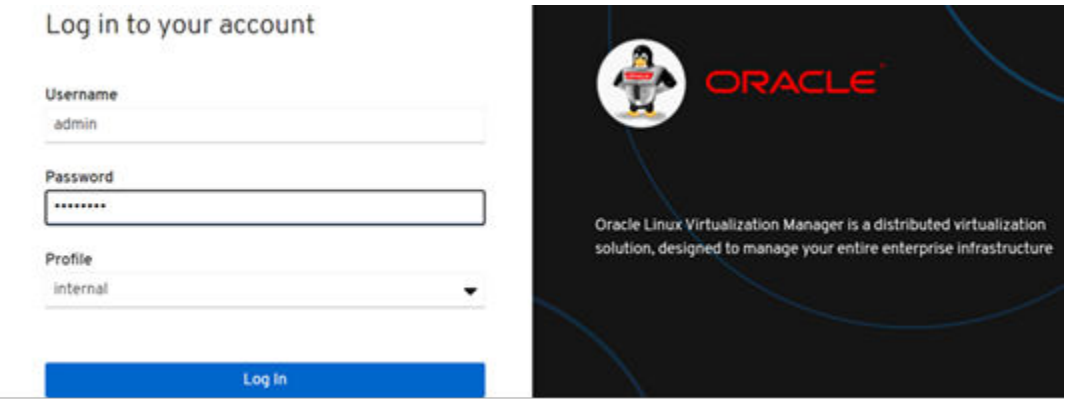


Access the OLVM administration portal

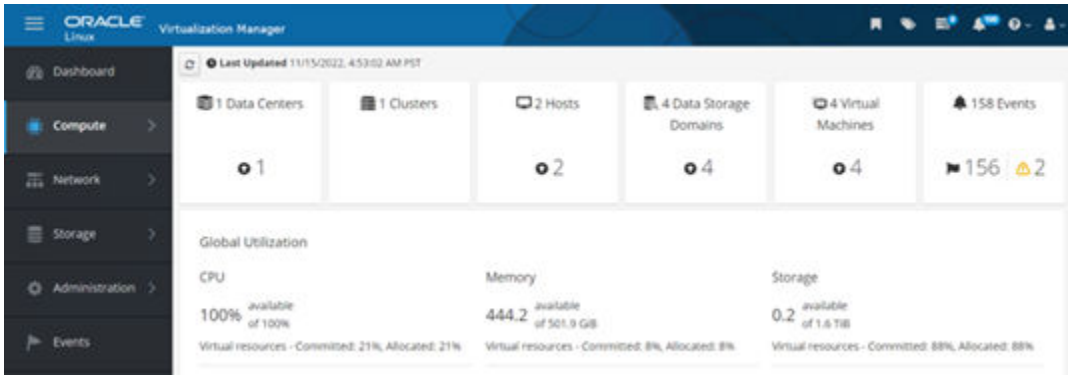
After successful installation of OLVM, access the administration portal to add, configure, and manage KVM hosts.

Log in to the **OLVM administration portal**.





The following figure shows the home page of the administration portal.



Upon login, configure clusters, data centers, and storage disks.

Clusters

Oracle Linux Virtualization Manager creates a default cluster in the default data center during installation. You can use the default cluster or set up new clusters.

See the Oracle Linux Virtualization Manager Administrator's Guide > Administration tasks for details.

Data centers

Oracle Linux Virtualization Manager creates a default data center during installation. You can use the default data center or set up new data centers. A data center requires a functioning cluster, host, and storage domain to operate in your virtualization environment.

> **Note:** The new data center remains in an Uninitialized state until a cluster, host, and storage domain are configured for it.

Storage

Oracle Linux Virtualization Manager uses a centralized storage system for virtual machine disk images, ISO files, and snapshots. You can use Network File System (NFS), Internet Small Computer System Interface (iSCSI), or Fibre Channel Protocol (FCP) storage. You can also configure local storage attached directly to hosts.

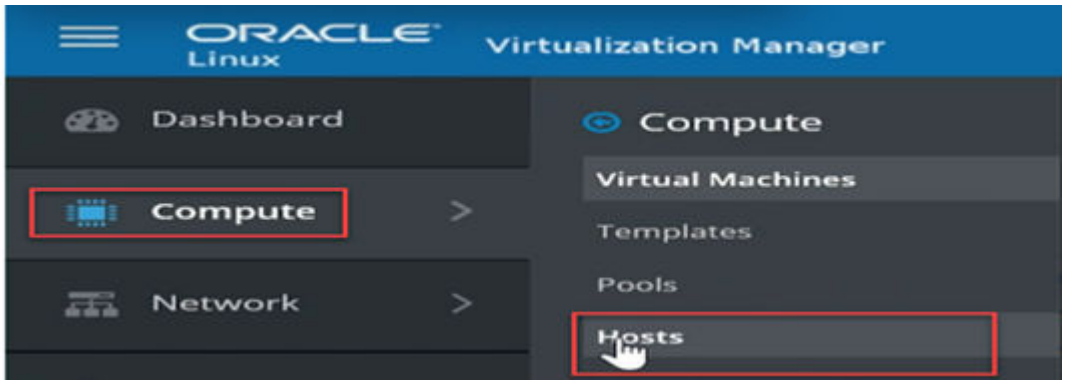
Storage devices in Oracle Linux Virtualization Manager are referred to as data domains, which are used to store virtual hard disks, snapshots, ISO files, and templates. Every data center must have at least one data domain. Data domains cannot be shared between data centers.

Add KVM hosts to the Manager

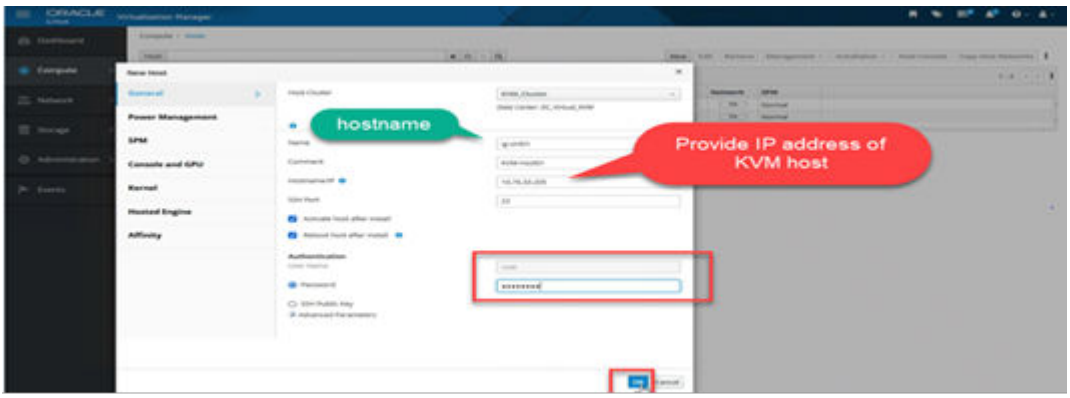
Add hosts and perform management tasks such VM creation and network creation to OLVM.

Procedure

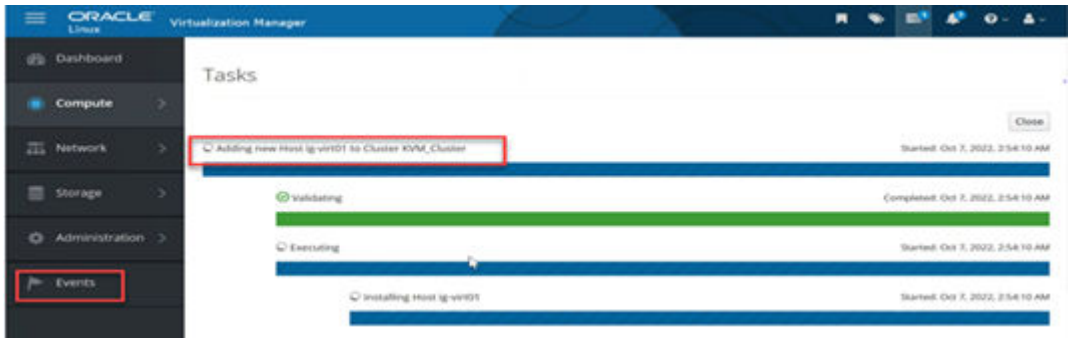
1. Log in to **OLVM administration portal** > **Compute** > **Hosts** to add KVM hosts that will be administered.

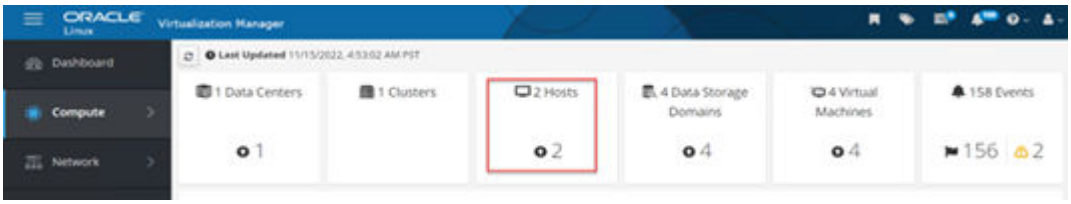


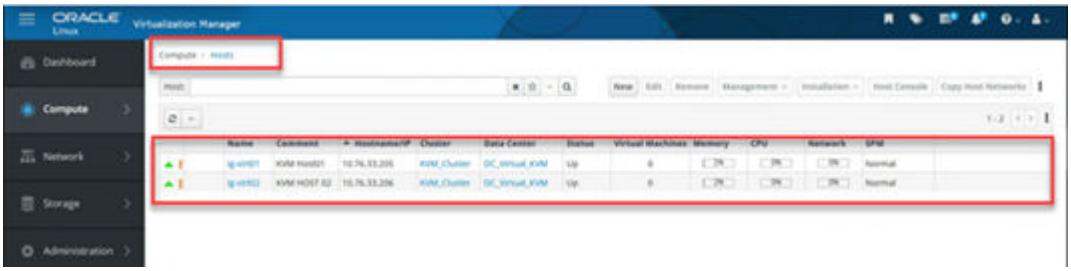
2. Provide the host name of **KVM target host** > **IP address** > **root** as well as username and password. The target host will be added to OLVM. Similarly, add another host.



3. View the progress of the host addition in the **Events** section.







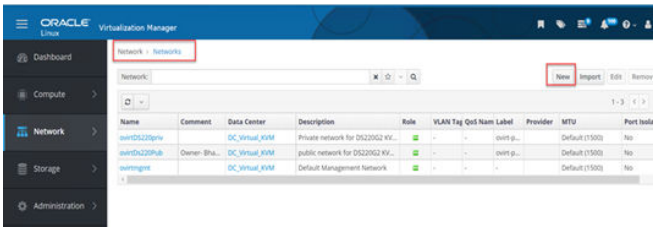
Create a logical network for VMs in OLVM

Create a virtual machine network that is assigned to the KVM host that was added in Add KVM hosts to the Manager (on page 51). This network is used as the virtual machine network for VMs created in Create virtual machines (on page 55).

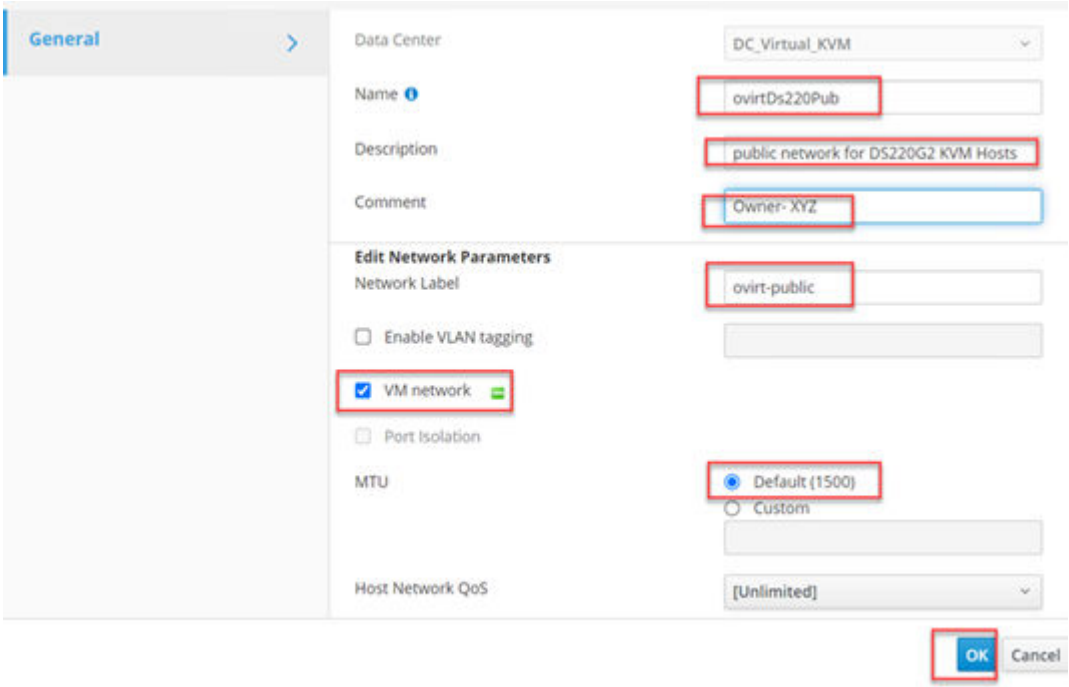
To create a virtual machine network:

Procedure

1. Go to **Network** > **Networks** > **New**.

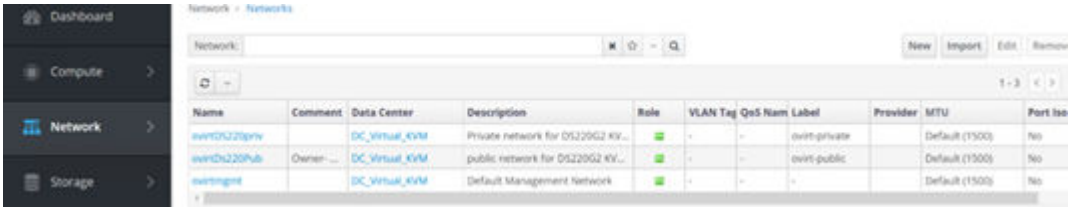


The new **Logical Network** dialog box opens with the **General** tab selected on the sidebar.

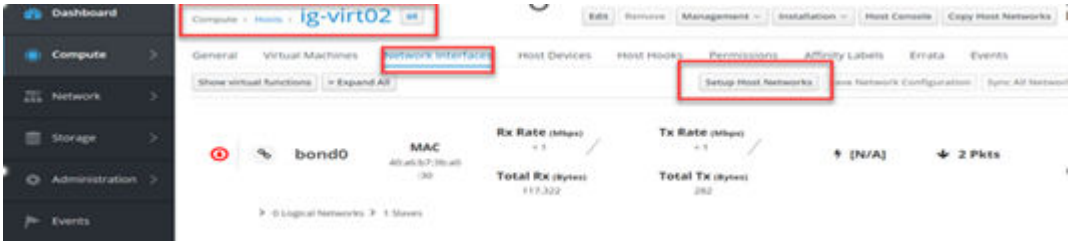


Note: For Oracle RAC optimum performance, we recommend configuring jumbo frames with an MTU size of 9000.

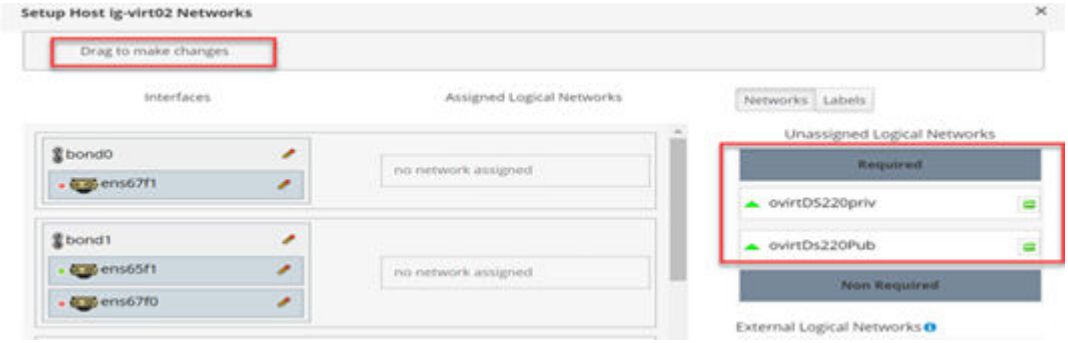
2. Similarly, create the **ovirtDS220Priv** network for private communication between nodes.



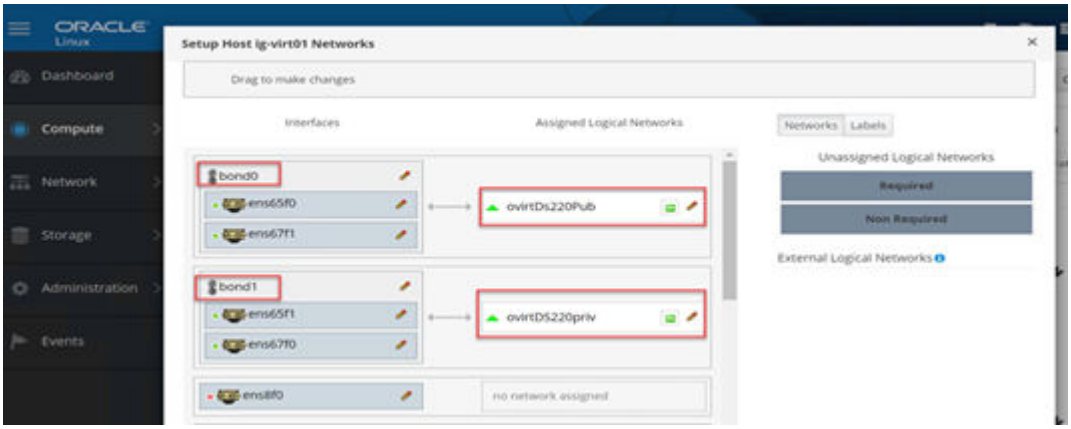
3. After the logical network is created, assign a virtual machine network to a KVM host.



4. Drag **ovirtDS220priv** and assign it to bond1 as a private network. Similarly, assign **ovirtDS220Pub** to bond0 as a public network.



The network looks like the following.

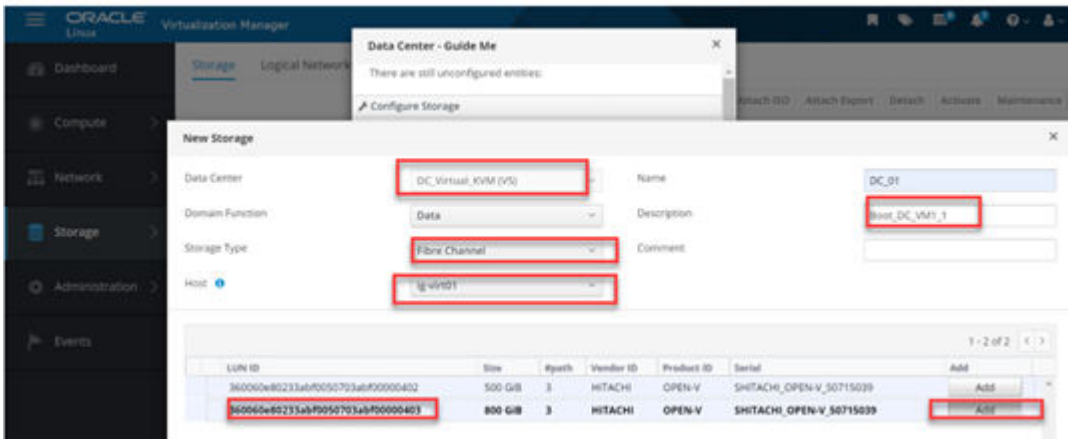


Configure and create VMs on KVM hosts

Configure storage domains for VM boot LUNs

To create VMs on KVM hosts, a separate bootable disk is needed for new VMs that are created from the storage domain. Therefore, create a disk that is the right size for a boot disk for VM1.

- Select the LUN ID created for the boot disk.



Create virtual machines

- **Types of virtual machines (VMs)**

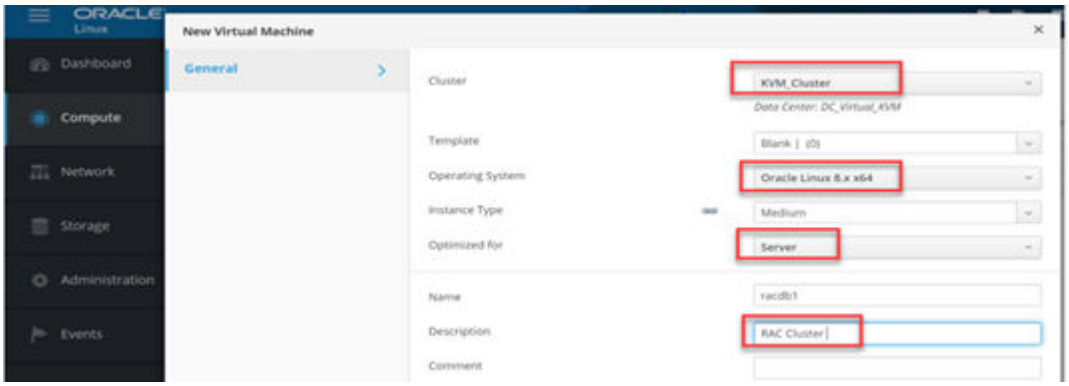
There are two types of VMs configured for Hitachi solutions for Oracle database, namely VMs optimized for server class and VMs optimized for high performance.

We have compared the performance results for different Oracle database workloads with standard recommended configurations along with CPU hard partitioning for server class VMs and high performance VMs. We noticed comparatively best results for random, transactional, analytics, and database background processes-related workloads with high performance VMs that are closer to bare metal performance. Therefore high performance VMs are recommended over server class VMs.

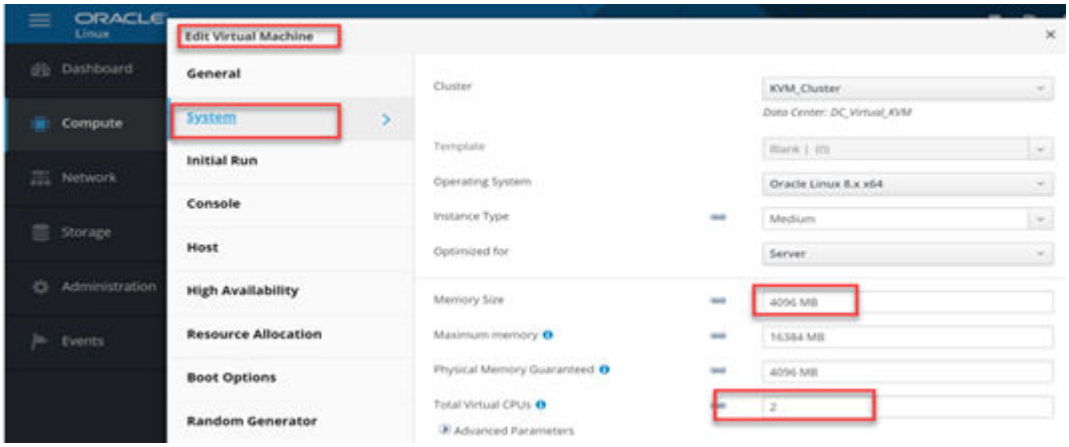
See the High Performance VMtech note for details.

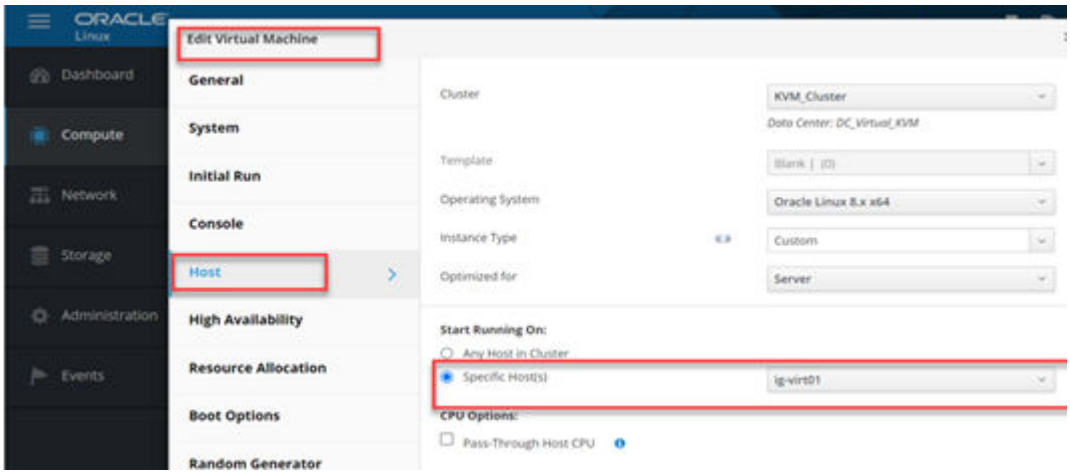
Procedure

1. Go to **Compute** > **Virtual machines** > **New**.



2. Provide memory and CPU parameters.





VMs are created on KVM hosts.

3. Install Oracle Linux 8 on the VMs.

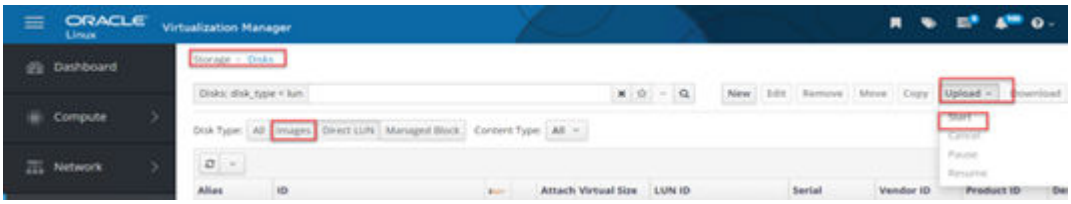
Name	Comment	Hostname/IP	Cluster	Data Center	Status	Virtual Machines	Memory
ig-virt01	KVM Host01	10.76.33.205	KVM_Cluster	DC_Virtual_KVM	Up	0	3%
ig-virt02	KVM Host02	10.76.33.121	KVM_Cluster	DC_Virtual_KVM	Up	0	3%

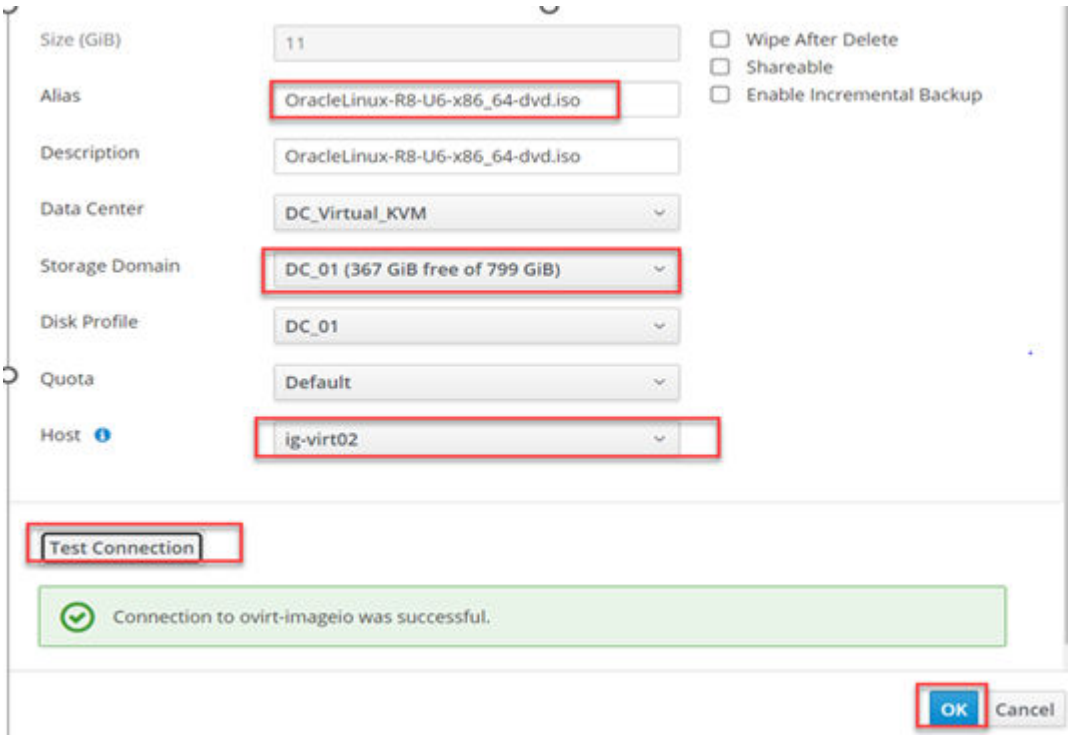
Install the Oracle OS on VMs for cluster software

Procedure

1. Go to **disk** > **Images** > **upload Oracle Linux 8 ISO image** > **host (on which VM resides)** > **Test connection**.

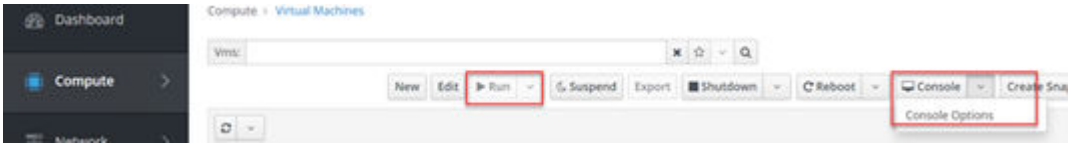
   Note: For a test check, the `ovirt-imageio-proxy` daemon should be running on the **OLVM host**.



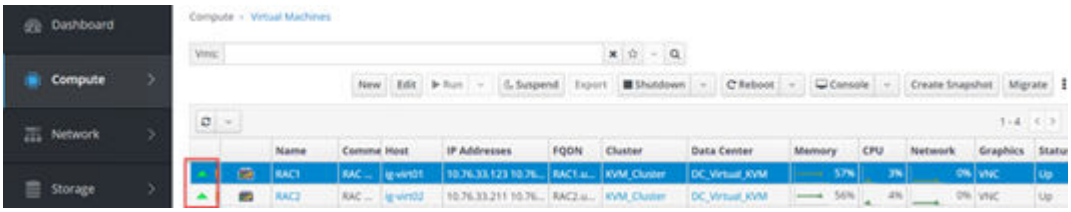


   The ISO image will start uploading.

2. Click **Start VM** to begin the OS install. Open the VM console and click **Install OS**.



3. Similarly, create one more VMs and install the OS for Oracle RAC on node 2. When both VMs are up and running, the status is green on the home page.


Dashboard
Compute
Network
Storage
Compute › Virtual Machines
New
Edit
Suspend
Export
Shutdown
Reboot
Console
Create Snapshot
Migrate
Name
Comme
Host
IP Addresses
FQDN
Cluster
Data Center
Memory
CPU
Network
Graphics
RAC1
RAC2
KVM_Cluster
DC_Virtual_KVM
VNC
Up

Chapter 11: Installation and configuration of Oracle Grid infrastructure

Now we have created two VMs for RAC configuration. We can begin to process the RAC pre-requisites.

Prepare VMs for Grid software

Prepare VMs for Grid software as the first step in the installation and configuration of the Oracle Grid infrastructure.

Install required RPMs on VM

```
# yum update -y
# yum install -y oracle-database-preinstall-19c.x86_64
or
# yum -y install oracle-database-preinstall-19c
# cd /tmp
# wget https://publicyum.oracle.com/repo/OracleLinux/OL8/addons/x86_64/getPackage/
oracleasm-support-2.1.12-1.el8.x86_64.rpm
# yum localinstall ./oracleasm-support-2.1.12-1.el8.x86_64.rpm
# yum install oracleasm-support
# wget https://download.oracle.com/otn_software/asmlib/oracleasmlib-2.0.17-
1.el8.x86_64.rpm
# yum localinstall ./oracleasmlib-2.0.17-1.el8.x86_64.rpm
# yum install bind* -y
# sysctl -p
```

OS level prerequisites

Procedure

1. Assign an IP address to nodes.

```
#vi /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
#Public IP
10.76.33.123    racdb1      racdb1.unified.local
10.76.33.124    racdb2      racdb2.unified.local
#Private IP
192.168.1.201   racdb1-priv     racdb1-priv.unified.local
```

```
192.168.1.202   racdb2-priv     racdb2-priv.unified.local
#Virtual IP
10.76.33.125    racdb1-vip     racdb1-vip.unified.local
10.76.33.126    racdb2-vip     racdb2-vip.unified.local
#Scan IP
10.76.33.127    racdb12-scan   racdb12-scan.unified.local
10.76.33.128    racdb12-scan   racdb12-scan.unified.local
10.76.33.129    racdb12-scan   racdb12-scan.unified.local
```

2. Create a directory structure and file system.

```
# lvcreate -n home -L 50GB ol
# lvcreate -n u01 -L 50GB ol
# lvcreate -n u02 -L 50GB ol
# mkfs -t xfs /dev/mapper/ol-home
# mkfs -t xfs /dev/mapper/ol-u01
# mkfs -t xfs /dev/mapper/ol-u02
# mount -t auto /dev/mapper/ol-home /home
# cd /
# mkdir -p /u01
# mount -t auto /dev/mapper/ol-u01 /u01
# mkdir -p /u02
# mount -t auto /dev/mapper/ol-u02 /u02
# lsblk
```

Note: Make an entry in `/etc/fstab` to auto-mount the file system after a restart of a node.

3. Create groups and users for RAC.

```
# groupadd -g 54327 asmdba
# groupadd -g 54328 asmoper
# groupadd -g 54329 asmadmin
# groupadd -g 54422 dba
# useradd -u 54322 -g oinstall -G dba grid
# usermod -G asmdba,asmoper,asmadmin,dba grid
# usermod -G asmdba,asmoper,asmadmin oracle  <<<< if oracle user already
presents then change its mode
# Passwd grid
# Passwd oracle
```

4. Create directories for grid and Oracle software binaries.

```
# mkdir -p /u01/app/grid
# chmod -R 775 /u01
# chown -R grid:oinstall /u01
# mkdir -p /u01/app/oraInventory
# chown -R grid:oinstall /u01/app/oraInventory
# mkdir -p /u01/softwares
# chown -R grid:oinstall /u01/softwares
```

```
# mkdir -p /u02/app/oracle
# chmod -R 775 /u02
# chown -R oracle:oinstall /u02
# mkdir -p /u02/app/oracle/product/19c/dbhome_1

# su - grid
$ vi .bash_profile
export TMP=/tmp
export TMPDIR=$TMP
export ORACLE_BASE=/u01/app/grid
export ORACLE_HOME=/u01/app/19c/grid_home1
export ORACLE_SID=+ASM1
export ORACLE_TERM=xterm
export BASE_PATH=/usr/sbin:$PATH
export PATH=$ORACLE_HOME/bin:$BASE_PATH
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:/lib:/usr/lib
export CLASSPATH=$ORACLE_HOME/JRE:$ORACLE_HOME/jlib:$ORACLE_HOME/rdbms/jlib

alias grid=' ./home/oracle/grid.env'
alias db=' ./home/oracle/db.env'

#su - oracle
$vi .bash_profile

export TMP=/tmp
export TMPDIR=$TMP
export ORACLE_BASE=/u02/app/oracle
export ORACLE_HOME=/u02/app/oracle/product/19c/dbhome_1
export GRID_HOME=/u01/app/19c/grid_home1
export ORACLE_SID=<Instance_name>
export ORACLE_TERM=xterm
export BASE_PATH=/usr/sbin:$PATH
export PATH=$ORACLE_HOME/bin:$BASE_PATH
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:/lib:/usr/lib
export CLASSPATH=$ORACLE_HOME/JRE:$ORACLE_HOME/jlib:$ORACLE_HOME/rdbms/jlib
```

5. Stop the firewall.

```
# systemctl stop firewall.service
# systemctl stop firewalld
# systemctl status firewalld
# systemctl disable firewalld
```

6. Run the `chroney ntp` configuration.

```
# systemctl enable chronyd.service
# systemctl restart chronyd.service
# chronyc -a 'burst 4/4'
# chronyc -c makestep
```

7. Configure temporary OS settings to avoid passwordless SSH user issues or errors during installation. These settings can be reverted after installation.
   a. Change scp.

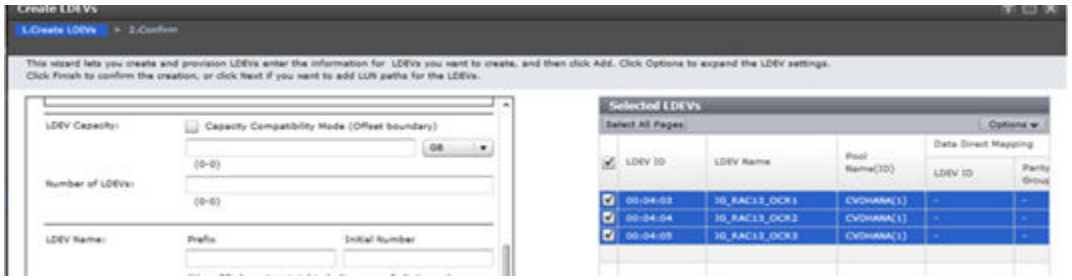
```
# cd
# mv /usr/bin/scp /usr/bin/scp.orig
# vi /usr/bin/scp   <<<< add below lines
/usr/bin/scp.orig -T $*
# chmod 555 /usr/bin/scp
```

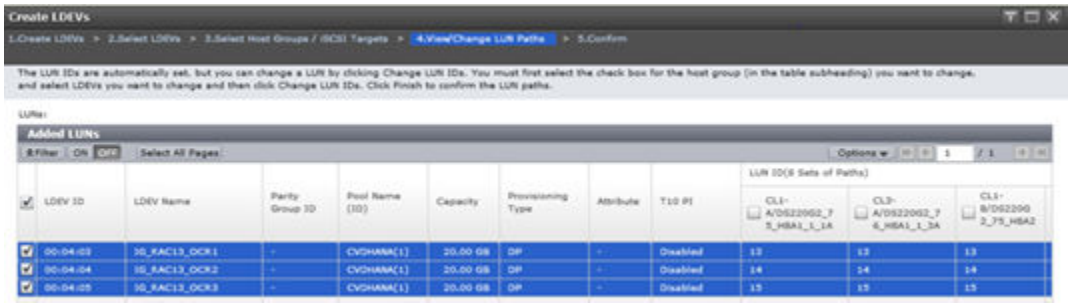
   b. Disable SELINUX.

```
# cat /etc/selinux/config|grep 'SELINUX=d'
# getenforce
# vi /etc/selinux/config
SELINUX=disabled
```

Create a shared LUN for OCR

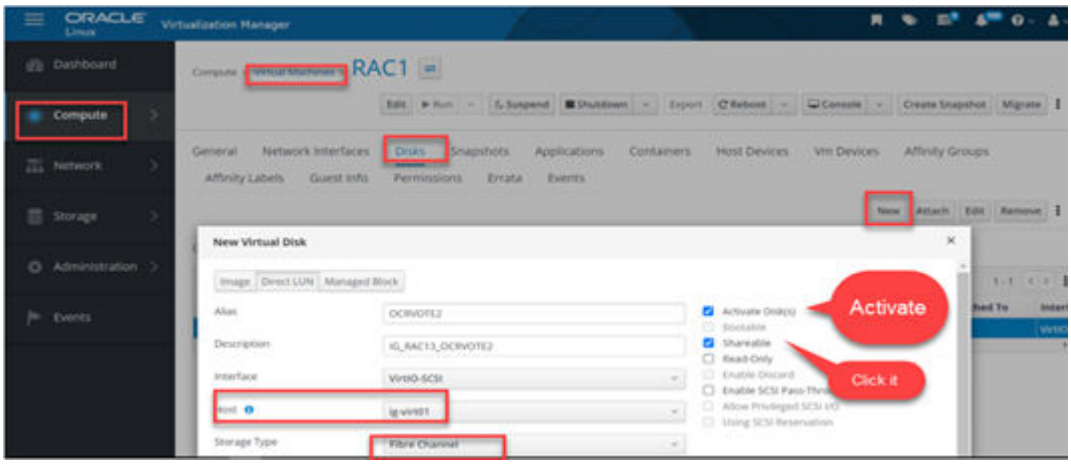
Grid software needs a shared disk to store OCR and votedisk files. Create three LUNs for an OCR disk of 20 Gb in size and share it between both VMs.

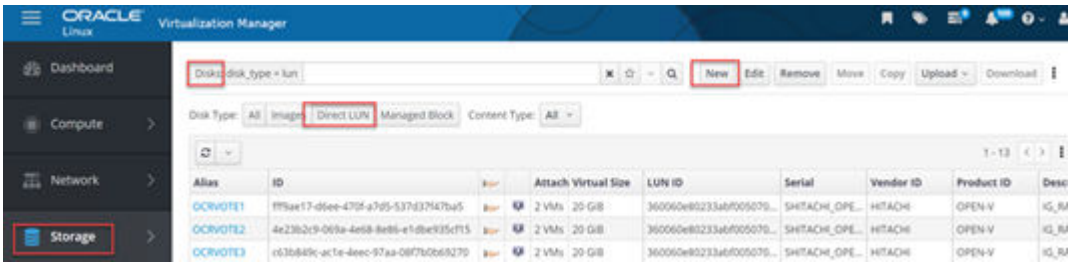


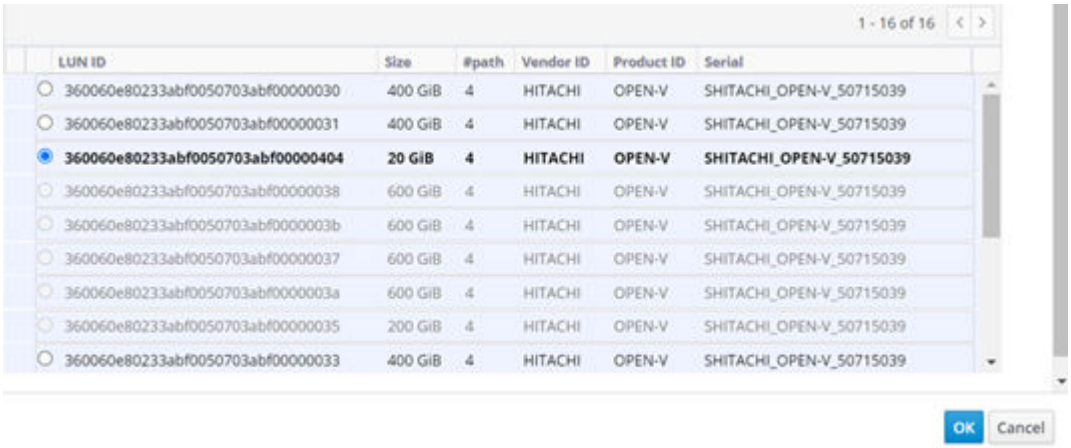


Procedure

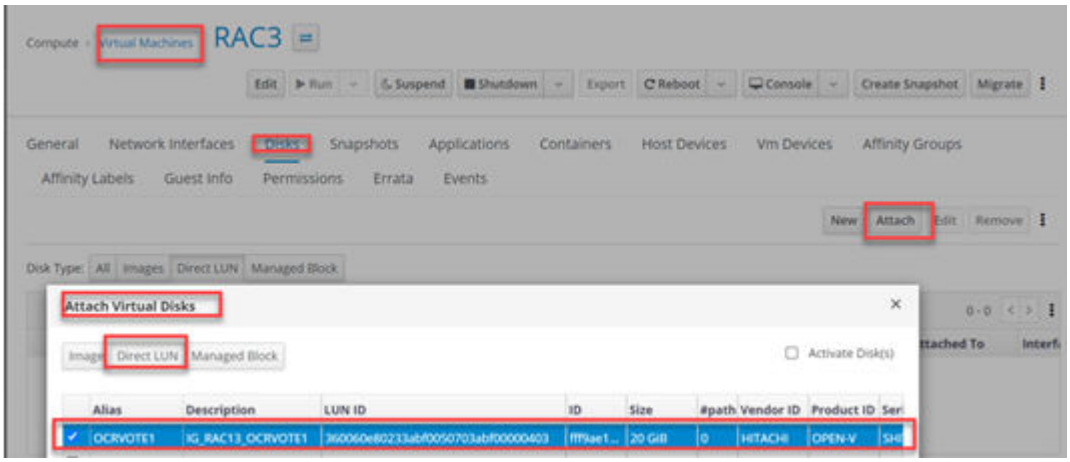
1. Go to **OLVM** > **Compute** > **Virtual Machines** > **VM1** > **Disks** > **New** > **Direct LUN**.



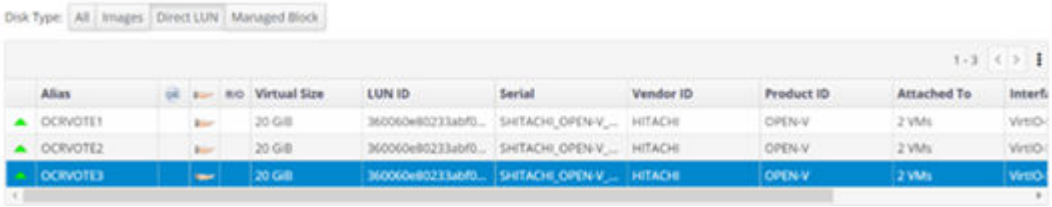




2. Go to **VM2** > **Disks** > **Attach** > **Direct LUN** for the second LUN.



3. Attach the other two OCR disks as well as other ASM disks for database installation and activate them as shown.



Configure UDEV rules for shared disks

UDEV uses files with rules that determine how it identifies devices and creates device names. The UDEV daemon (`udevd`) reads the rules files at system startup and stores the rules in memory.

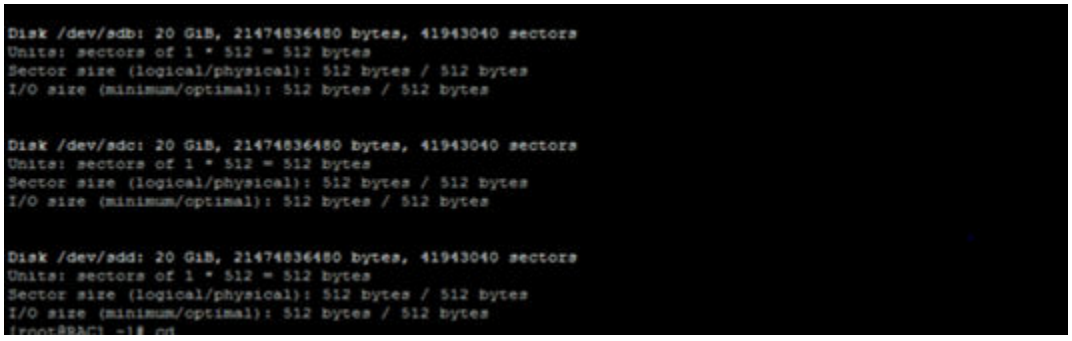
In the older kernels, the `/dev` directory contained static device files. But with dynamic device creation, device nodes for only those devices that are present in the system are created.

If the kernel discovers a new device or an existing device goes offline, the kernel sends an event action (`uevent`) notification to `udevd`, which matches the in-memory rules against the device attributes in `/sys` to identify the device. As part of device event handling, rules can specify additional programs that should run to configure a device. Rules files, which have the file extension `.rules`, are in stored in the following directories:

```
/lib/udev/rules.d          <<<<< Contains default rules files. Do not edit these files
/etc/udev/rules.d/*.rules   <<<<< Contains customized rules files. You can modify
these files.
```

See **Doc ID 1528148.1** on the Oracle support site for additional configuration details.

The following shows shared disks (sdb, sdc, and sdd) mounted to VMs. To create UDEV the same number of WWN disks are needed to be uniquely identified on both hosts.



Procedure

1. Format all three disks on node1 (this is not required on node 2).

```
# fdisk /dev/sdc
Welcome to fdisk (util-linux 2.32.1).
Changes will remain in memory only, until you decide to write them.
```

```
Be careful before using the write command.
Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x05446bfb.
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1):
First sector (2048-33554431, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-33554431, default 33554431):
Created a new partition 1 of type 'Linux' and of size 16 GiB.
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
# fdisk -l /dev/sdc*
Disk /dev/sdc: 16 GiB, 17179869184 bytes, 33554432 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x05446bfb
Device     Boot Start      End  Sectors Size Id Type
/dev/sdc1        2048 33554431 33552384  16G 83 Linux
```

2. Find the WWN of the disk.

```
# /lib/udev/scsi_id -g -u -d /dev/sdb
or
# /usr/lib/udev/scsi_id -g -u -d /dev/sdb
or
# lsblk -o +WWN,serial
or
# udevadm info --query=all --name=/dev/sdo | egrep "WWN|SERIAL"
```

```
# cd /etc/udev/rules.d
# vi 99-oracle-asmdevices.rules
KERNEL=="sd*", PROGRAM=="scsi_id --page=0x83 --whitelisted --device=/dev/%k",
RESULT=="36000c29c97c3b1d37878a1aff92426fb" SYMLINK+="asm-OCRVD1", OWNER="grid",
GROUP="oinstall", MODE="0660"
KERNEL=="sd*", PROGRAM=="scsi_id --page=0x83 --whitelisted --device=/dev/%k",
RESULT=="36000c29d15638ed7d62c304bb3dc749f" SYMLINK+="asm-OCRVD2", OWNER="grid",
GROUP="oinstall", MODE="0660"
KERNEL=="sd*", PROGRAM=="scsi_id --page=0x83 --whitelisted --device=/dev/%k",
```

```
RESULT=="36000c29d15638ed7d62c304bb3dc749f" SYMLINK+="asm-OCRVD2", OWNER="grid",
GROUP="oinstall", MODE="0660"
```

Note: Change the highlighted number with the actual LUN ID for your environment and save the file.

3. Run the following commands.

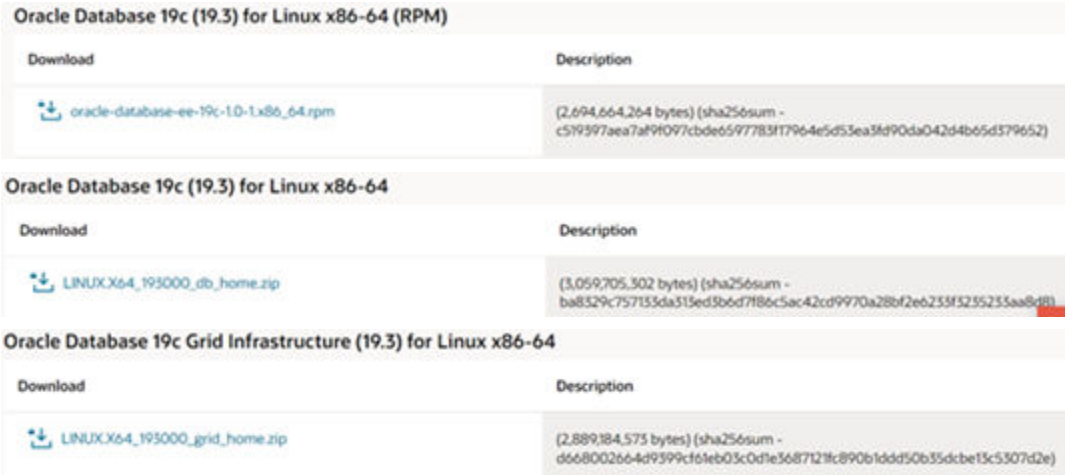
```
# udevadm control --reload-rules && udevadm trigger --action=add
# udevadm trigger
```

4. Copy the `99-oracle-asmdevices.rules` file from **node1** to **node2** at `/etc/udev/rules.d` and run the previous commands on VM2 to reload UDEV.

Download Oracle RAC 19c Grid software and copy to VMs

Download Oracle database and 19c grid software from the following link:

https://www.oracle.com/database/technologies/oracle19c-linux-downloads.html



Start a remote putty session on **VM1**, and copy software binaries on **host1**, and unzip the downloaded file.

Table 1 Summary

Sr.no.	Task Description	Node 1	Node 2	Status
1	Hardware pre-checks	Myhost1	Myhost2	Done
2	Configure LUN (storage)	Boot LUN	Boot LUN	Done
3	Configure Zoning	Yes	Yes	Done
4	OS (OL8.6) installation on bare metal host	Yes	Yes	Done

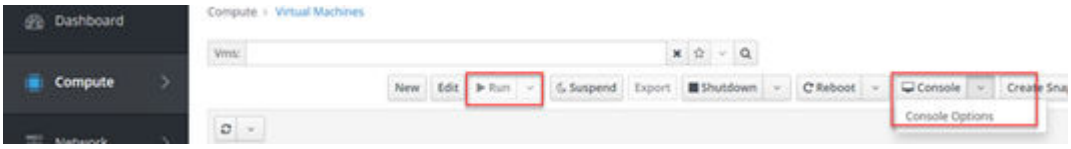
Sr.no.	Task Description	Node 1	Node 2	Status
5	Configure network bonding and assign IPs (Pub/Private)	Yes	Yes	Done
6	Install Oracle Linux KVM hypervisor	Yes	Yes	Done
7	Install Oracle Linux Virtualization Manager (OLVM) and add KVM hosts	It is a separate management host		Done
8	Create Logical network for VM in OLVM	Yes	Yes	Done
9	Create VM on KVM host through OLVM	Yes	Yes	Done
10	Install OS on VM and GRID pre-requisite	Yes	Yes	Done
11	Download GI software to VM	Yes	Yes	Done

Grid installation

To install Grid software, use the remote GUI console to VM1 from OLVM.

Procedure

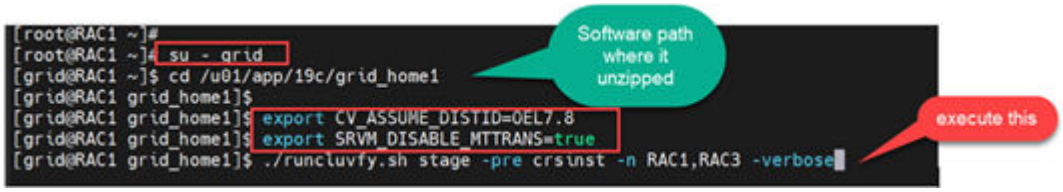
1. Log in with **GRID** user credentials.



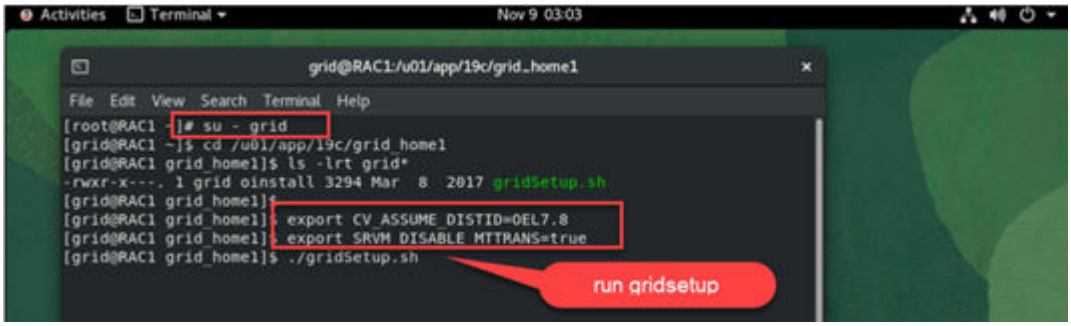
2. Log in with Grid and go to the directory where the GI software is unzipped.

```
# cd /u01/app/19c/grid_home1/cv/rpm
# yum install cvuqdisk-1.0.10-1.rpm <<<<< install RPM
```

3. Run the **./runcluvfy.sh** utility to check RAC prerequisites.



4. Upon completion check failures and take corrective action and then proceed with GI software installation.

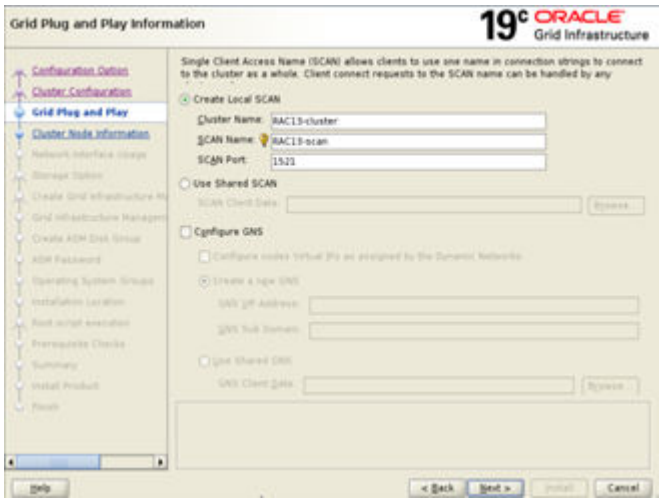


A GUI opens to install GI software. Select an icon and proceed with the installation. Change directory path, location, and permissions as needed.

a. Choose **Configure Oracle Grid infrastructure for a new Cluster**.

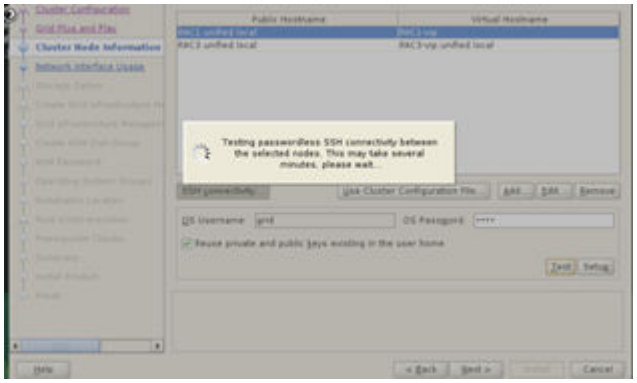
b. Choose **Configure an Oracle standalone Cluster**.

c. Specify **Cluster name** and **SCAN listener** name with **port** number.

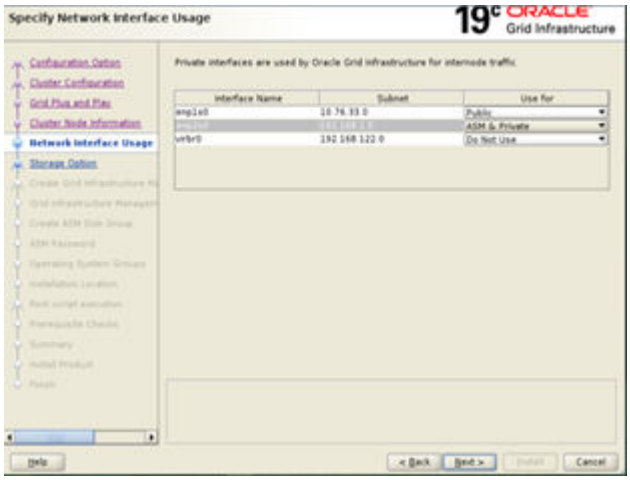


d. Give **2nd node name**, **details** for the cluster (give all nodes name that are going to be part of cluster).

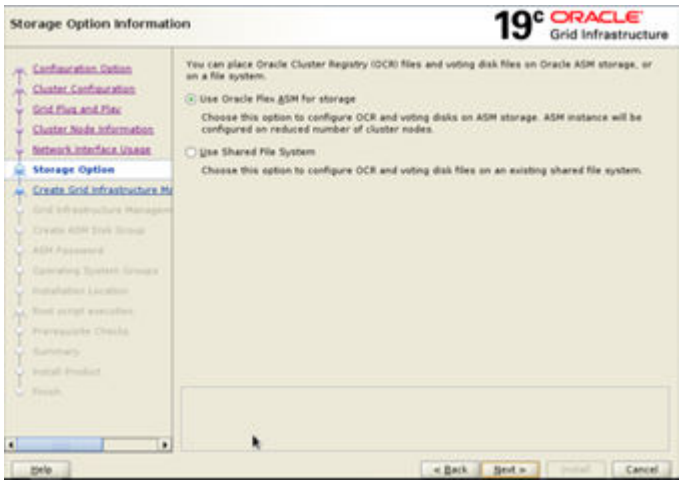
e. Check SSH connectivity between cluster nodes by clicking **SSH connectivity**.



f. Choose **Network interface usage** for the environment configuration.

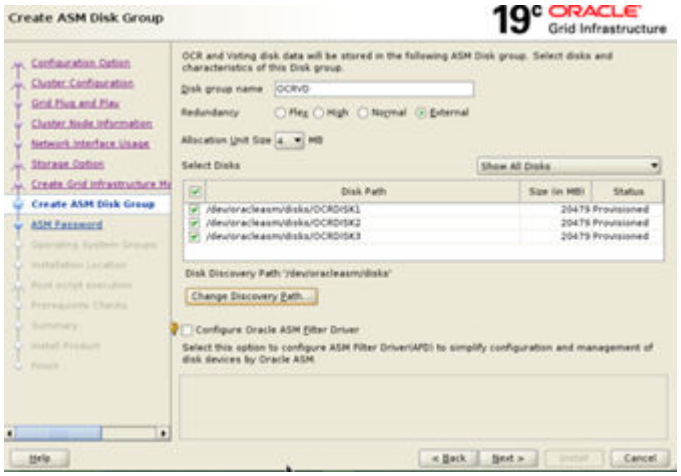


g. Choose a storage option (we used Flex ASM for OCR, vote disk).



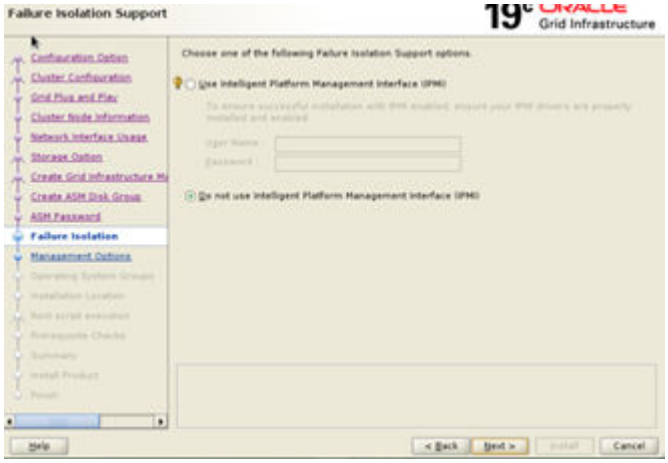
h. There is no need to configure a management repository.

i. Choose **ASM disks for OCR** and **Vote disk** to be **Present**.

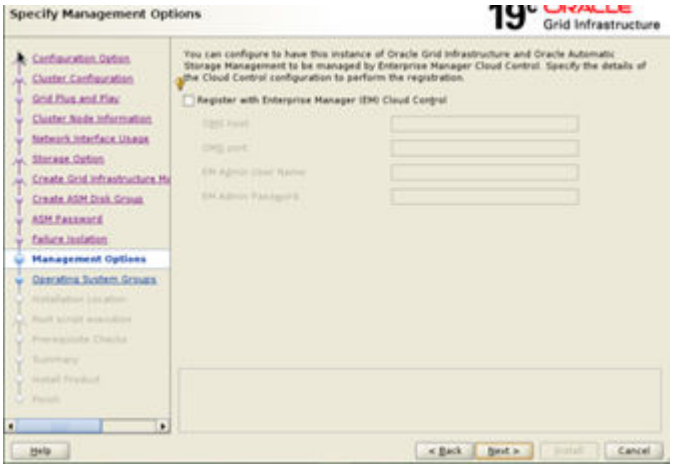


j. Set a password for SYS to log in to ASM instances.

k. Choose **Do not use IPMI**.



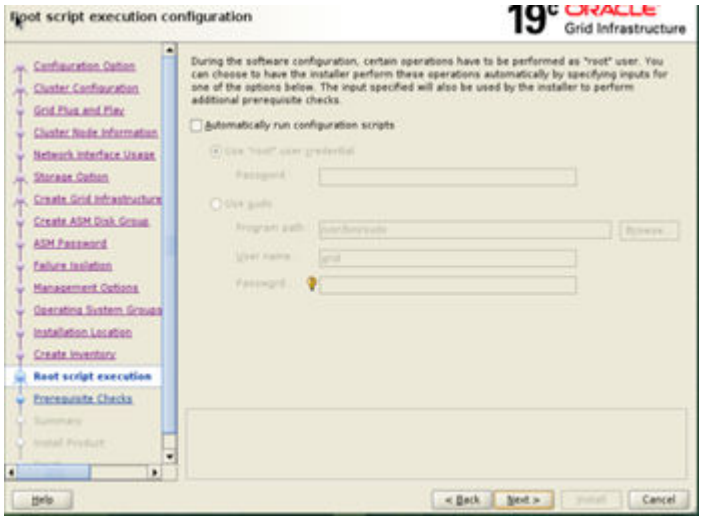
l. Choose **EM** to configure enterprise manager (EM) control, otherwise click **NEXT**.



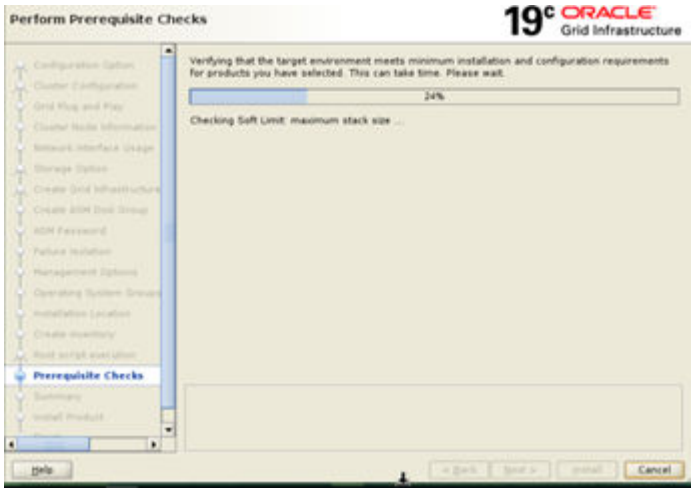
m. Choose groups and environment ownership as needed.

n. Specify the Oracle base for Grid software.

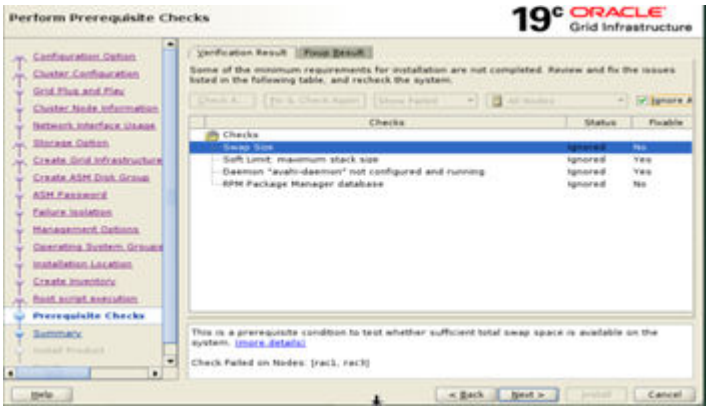
o. Specify the inventory location for Grid software.

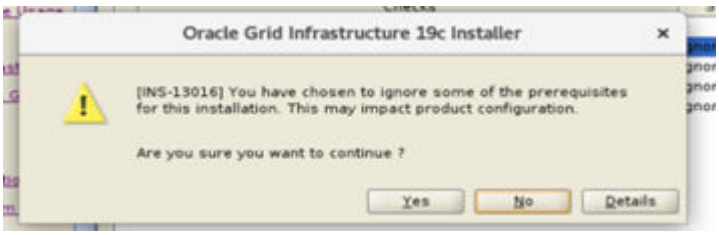


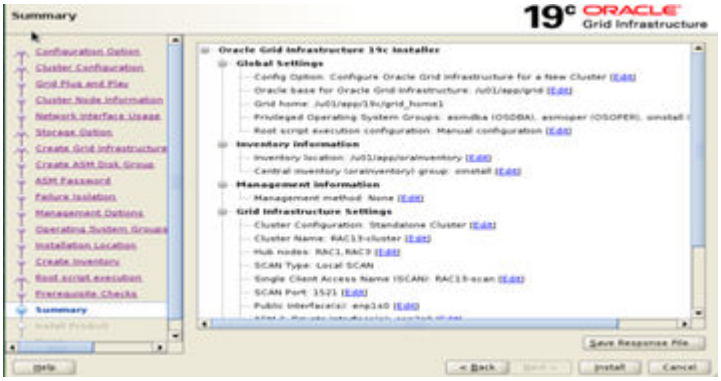
p. Monitor the progress of prerequisite checks.



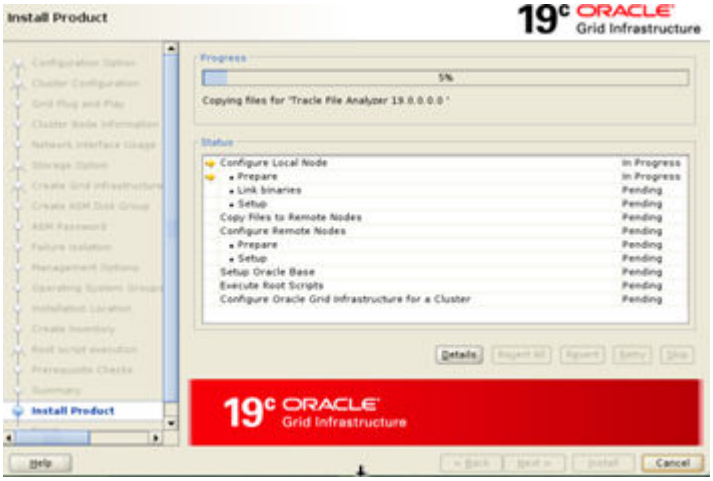
q. Check for any prerequisite failures and take corrective action.

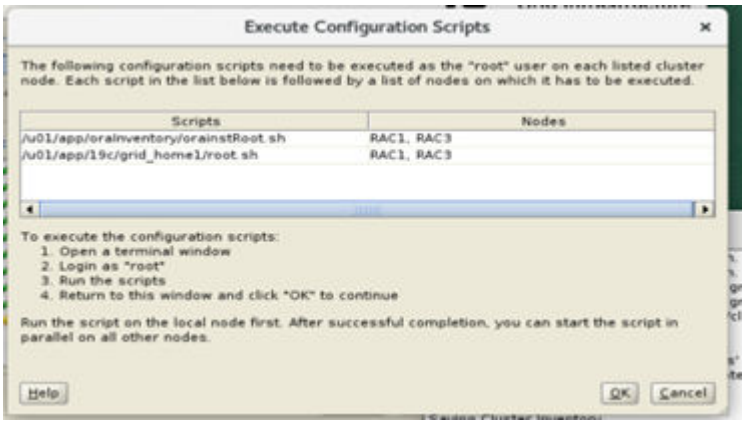






r. Check the progress of Grid software installation.





s. Run `orainstRoot.sh` and `root.sh` on nodes according to the **sequence** given on the screen.

```
[root@RAC1 ~]# /u01/app/oraInventory/orainstRoot.sh
Changing permissions of /u01/app/oraInventory.
Adding read,write permissions for group.
Removing read,write,execute permissions for world.

Changing groupname of /u01/app/oraInventory to oinstall.
The execution of the script is complete.
[root@RAC1 ~]#
```

```
[root@RAC3 ~]# /u01/app/oraInventory/orainstRoot.sh
Changing permissions of /u01/app/oraInventory.
Adding read,write permissions for group.
Removing read,write,execute permissions for world.

Changing groupname of /u01/app/oraInventory to oinstall.
The execution of the script is complete.
[root@RAC3 ~]#
```

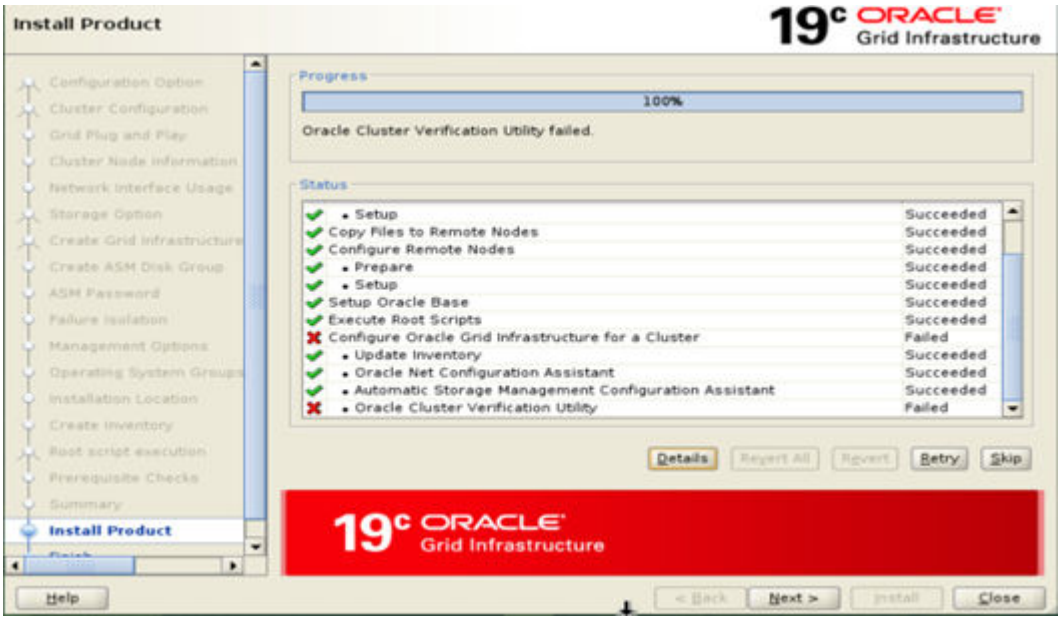
```
[root@RAC1 ~]# /u01/app/19c/grid_home1/root.sh
Performing root user operation.

The following environment variables are set as:
    ORACLE_OWNER= grid
    ORACLE_HOME=  /u01/app/19c/grid_home1

Enter the full pathname of the local bin directory: [/usr/local/bin]:
   Copying dbhome to /usr/local/bin ...
   Copying oraenv to /usr/local/bin ...
   Copying coraenv to /usr/local/bin ...


Creating /etc/oratab file...
Entries will be added to the /etc/oratab file as needed by
Database Configuration Assistant when a database is created
Finished running generic part of root script.
Now product-specific root actions will be performed.
Relinking oracle with rac_on option
Using configuration parameter file: /u01/app/19c/grid_home1/crs/install/crsconfig_params
The log of current session can be found at:
  /u01/app/grid/crsdata/rac1/crsconfig/rootcrs_rac1_2022-11-09_05-12-24AM.log
2022/11/09 05:12:31 CLSRSC-594: Executing installation step 1 of 19: 'SetupTFA'.
2022/11/09 05:12:31 CLSRSC-594: Executing installation step 2 of 19: 'ValidateEnv'.
2022/11/09 05:12:31 CLSRSC-363: User ignored prerequisites during installation
2022/11/09 05:12:31 CLSRSC-594: Executing installation step 3 of 19: 'CheckFirstNode'.
2022/11/09 05:12:33 CLSRSC-594: Executing installation step 4 of 19: 'GenSiteGUIDs'.
2022/11/09 05:12:33 CLSRSC-594: Executing installation step 5 of 19: 'SetupOSD'.
2022/11/09 05:12:33 CLSRSC-594: Executing installation step 6 of 19: 'CheckCRSConfig'.
2022/11/09 05:12:39 CLSRSC-594: Executing installation step 7 of 19: 'SetupLocalGPNP'.
2022/11/09 05:12:42 CLSRSC-594: Executing installation step 8 of 19: 'CreateRootCert'.
2022/11/09 05:12:45 CLSRSC-594: Executing installation step 9 of 19: 'ConfigOLR'.
2022/11/09 05:12:52 CLSRSC-4002: Successfully installed Oracle Trace File Analyzer (TFA) Collector.
2022/11/09 05:12:56 CLSRSC-594: Executing installation step 10 of 19: 'ConfigCHMOS'.
2022/11/09 05:12:56 CLSRSC-594: Executing installation step 11 of 19: 'CreateOHASD'.
2022/11/09 05:12:59 CLSRSC-594: Executing installation step 12 of 19: 'ConfigOHASD'.
2022/11/09 05:12:59 CLSRSC-330: Adding Clusterware entries to file 'oracle-ohasd.service'
2022/11/09 05:13:22 CLSRSC-594: Executing installation step 13 of 19: 'InstallAFD'.
2022/11/09 05:13:22 CLSRSC-594: Executing installation step 14 of 19: 'InstallACFS'.
2022/11/09 05:13:25 CLSRSC-594: Executing installation step 15 of 19: 'InstallKA'.
2022/11/09 05:13:28 CLSRSC-594: Executing installation step 16 of 19: 'InstConfig'.
```

```
2022/11/09 05:13:28 CLSRSC-594: Executing installation step 16 of 19: 'InitConfig'.

ASM has been created and started successfully.

[DBT-30001] Disk groups created successfully. Check /u01/app/grid/cfgtoollogs/asmca/asmca-221109AM051357.log for details.

2022/11/09 05:14:45 CLSRSC-482: Running command: '/u01/app/19c/grid_home1/bin/ocrconfig -upgrade grid oinstall'
CRS-4256: Updating the profile
Successful addition of voting disk 8cae166847e34f23bf5e50138043ed3d.
Successfully replaced voting disk group with +OCRVD.
CRS-4256: Updating the profile
CRS-4266: Voting file(s) successfully replaced
##  STATE    File Universal Id                File Name Disk group
--  -----    -----------------                --------- ---------
 1. ONLINE   8cae166847e34f23bf5e50138043ed3d (/dev/oracleasm/disks/OCRDISK1) [OCRVD]
Located 1 voting disk(s).
2022/11/09 05:15:55 CLSRSC-594: Executing installation step 17 of 19: 'StartCluster'.
2022/11/09 05:16:55 CLSRSC-343: Successfully started Oracle Clusterware stack
2022/11/09 05:16:55 CLSRSC-594: Executing installation step 18 of 19: 'ConfigNode'.
2022/11/09 05:17:59 CLSRSC-594: Executing installation step 19 of 19: 'PostConfig'.
2022/11/09 05:18:17 CLSRSC-325: Configure Oracle Grid Infrastructure for a Cluster ... succeeded
[root@RAC1 ~]#
```

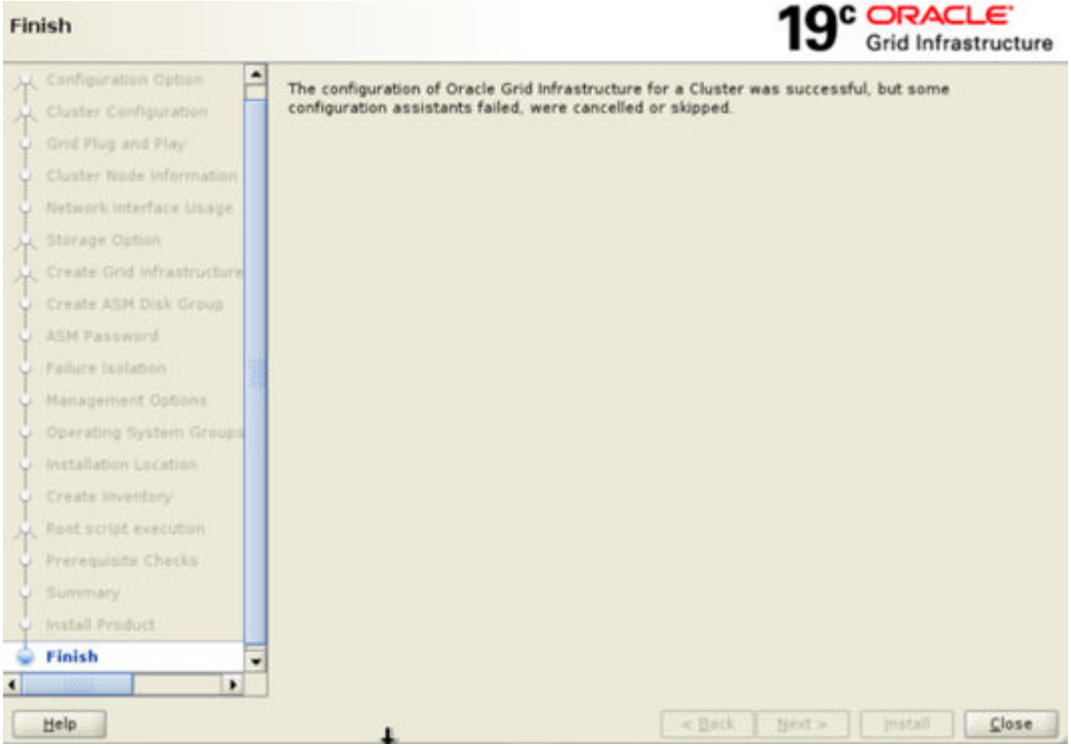
```
[root@RAC3 ~]# /u01/app/19c/grid_home1/root.sh
Performing root user operation.

The following environment variables are set as:
    ORACLE_OWNER= grid
    ORACLE_HOME=  /u01/app/19c/grid_home1

Enter the full pathname of the local bin directory: [/usr/local/bin]:
   Copying dbhome to /usr/local/bin ...
   Copying oraenv to /usr/local/bin ...
   Copying coraenv to /usr/local/bin ...


Creating /etc/oratab file...
Entries will be added to the /etc/oratab file as needed by
Database Configuration Assistant when a database is created
Finished running generic part of root script.
Now product-specific root actions will be performed.
Relinking oracle with rac_on option
Using configuration parameter file: /u01/app/19c/grid_home1/crs/install/crsconfig_params
The log of current session can be found at:
  /u01/app/grid/crsdata/rac3/crsconfig/rootcrs_rac3_2022-11-09_05-21-17AM.log
2022/11/09 05:21:20 CLSRSC-594: Executing installation step 1 of 19: 'SetupTFA'.
2022/11/09 05:21:20 CLSRSC-594: Executing installation step 2 of 19: 'ValidateEnv'.
2022/11/09 05:21:20 CLSRSC-363: User ignored prerequisites during installation
2022/11/09 05:21:20 CLSRSC-594: Executing installation step 3 of 19: 'CheckFirstNode'.
2022/11/09 05:21:21 CLSRSC-594: Executing installation step 4 of 19: 'GenSiteGUIDs'.
2022/11/09 05:21:21 CLSRSC-594: Executing installation step 5 of 19: 'SetupOSD'.
2022/11/09 05:21:21 CLSRSC-594: Executing installation step 6 of 19: 'CheckCRSConfig'.
2022/11/09 05:21:21 CLSRSC-594: Executing installation step 7 of 19: 'SetupLocalGPNP'.
2022/11/09 05:21:22 CLSRSC-594: Executing installation step 8 of 19: 'CreateRootCert'.
2022/11/09 05:21:22 CLSRSC-594: Executing installation step 9 of 19: 'ConfigOLR'.
2022/11/09 05:21:29 CLSRSC-594: Executing installation step 10 of 19: 'ConfigCHMOS'.
2022/11/09 05:21:29 CLSRSC-594: Executing installation step 11 of 19: 'CreateOHASD'.
2022/11/09 05:21:30 CLSRSC-594: Executing installation step 12 of 19: 'ConfigOHASD'.
2022/11/09 05:21:30 CLSRSC-330: Adding Clusterware entries to file 'oracle-ohasd.service'
2022/11/09 05:21:40 CLSRSC-4002: Successfully installed Oracle Trace File Analyzer (TFA) Collector.
2022/11/09 05:21:46 CLSRSC-594: Executing installation step 13 of 19: 'InstallAFD'.
2022/11/09 05:21:47 CLSRSC-594: Executing installation step 14 of 19: 'InstallACFS'.
```

```
2022/11/09 05:21:47 CLSRSC-594: Executing installation step 14 of 19: 'InstallACFS'.
2022/11/09 05:21:48 CLSRSC-594: Executing installation step 15 of 19: 'InstallKA'.
2022/11/09 05:21:49 CLSRSC-594: Executing installation step 16 of 19: 'InitConfig'.
2022/11/09 05:21:56 CLSRSC-594: Executing installation step 17 of 19: 'StartCluster'.
2022/11/09 05:22:38 CLSRSC-343: Successfully started Oracle Clusterware stack
2022/11/09 05:22:38 CLSRSC-594: Executing installation step 18 of 19: 'ConfigNode'.
2022/11/09 05:22:47 CLSRSC-594: Executing installation step 19 of 19: 'PostConfig'.
2022/11/09 05:22:51 CLSRSC-325: Configure Oracle Grid Infrastructure for a Cluster ... succeeded
[root@RAC3 ~]#
```

t. Monitor the installation progress.



u. Grid software installation completes successfully.

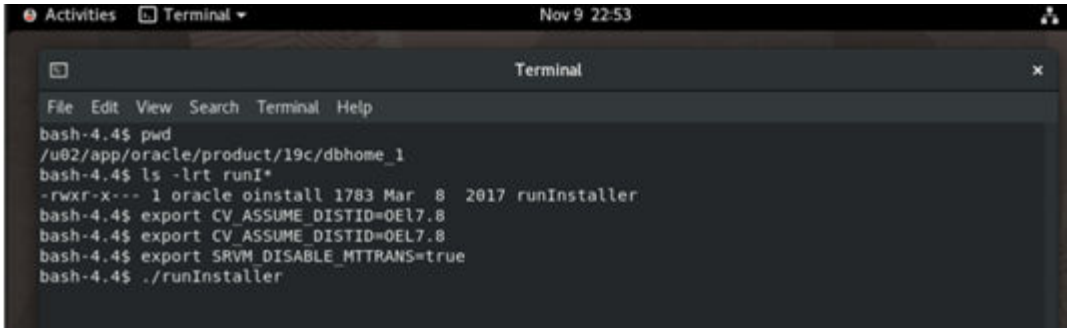


Chapter 12: Installation and configuration of Oracle RDBMS

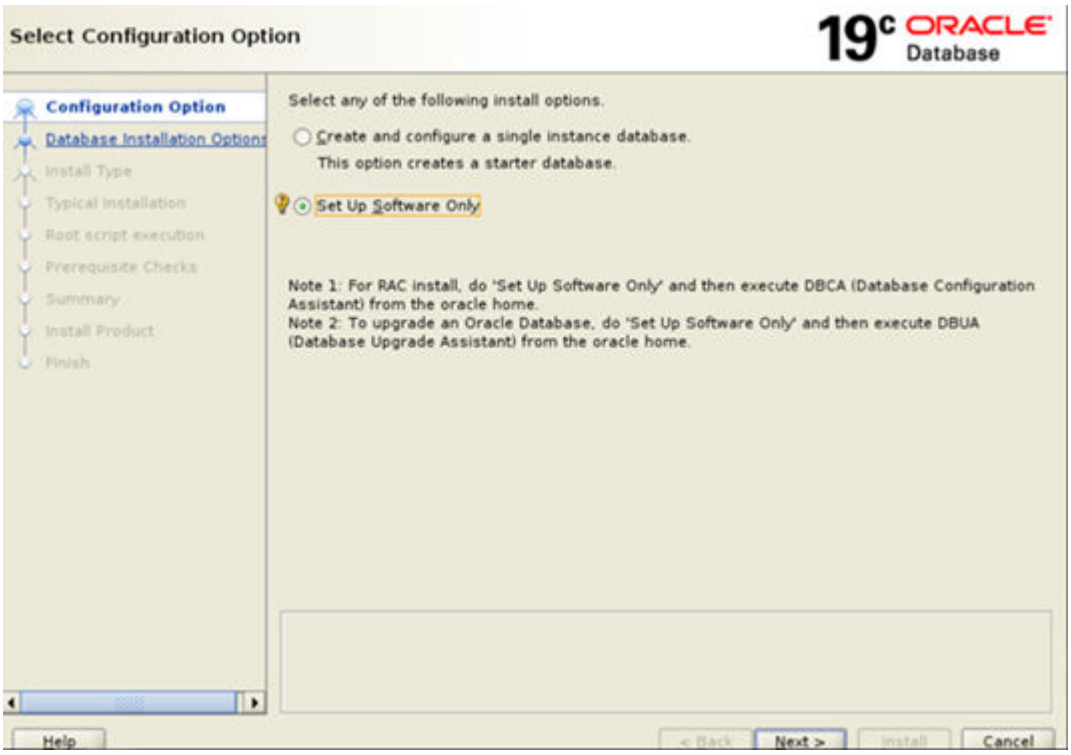
RDBMS installation

Procedure

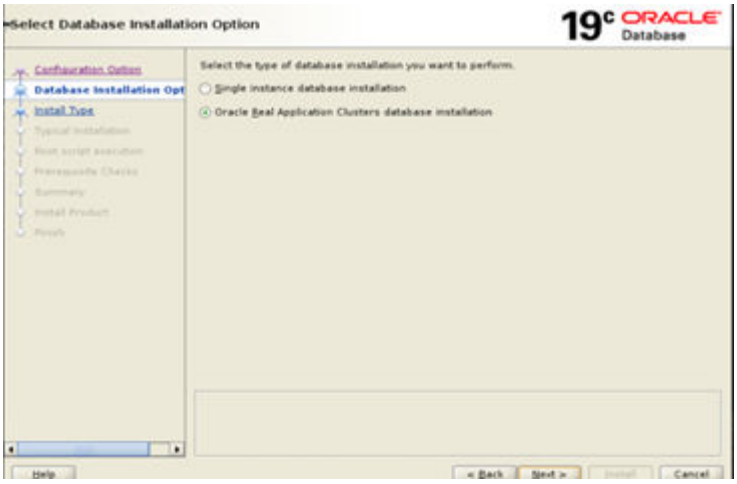
1. For database software installation, log in with Oracle user credentials and unzip the DB software. Go to the location where the software is unzipped and set environment variables.



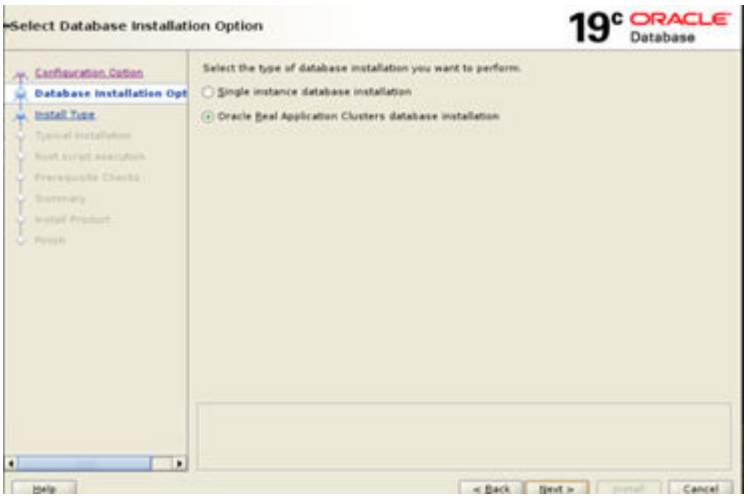
2. Choose **Set Up Software Only** (optionally you can choose to create a database as well).



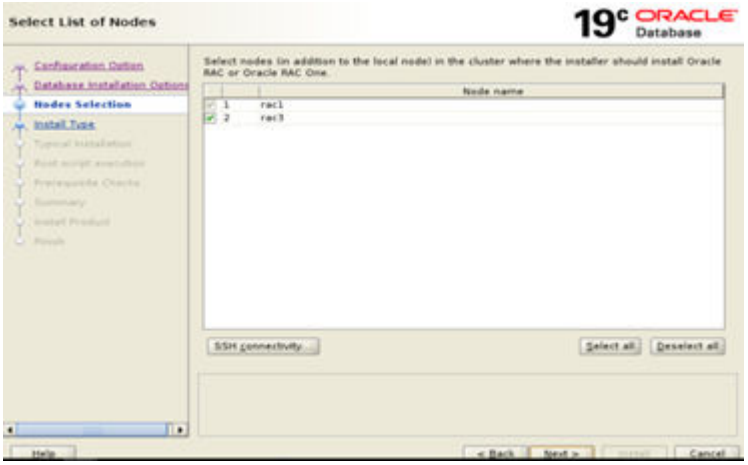
3. Choose the database installation type (RAC or standalone database).



4. Choose **Real application cluster** (choose standalone if you are not using RAC).

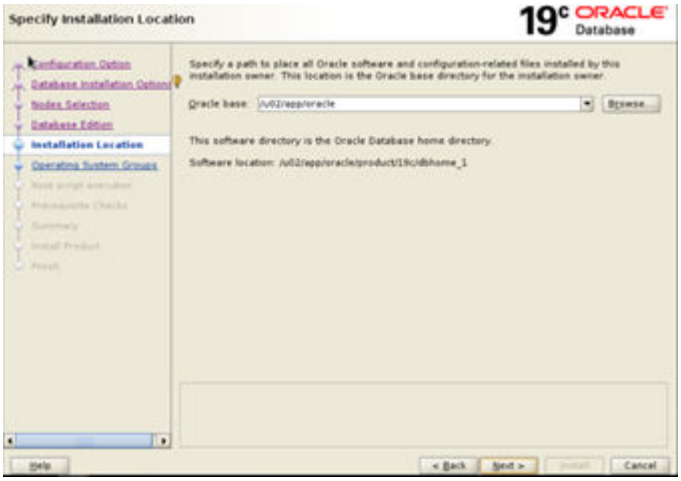


5. Choose the nodes on which RDBMS software must be installed.

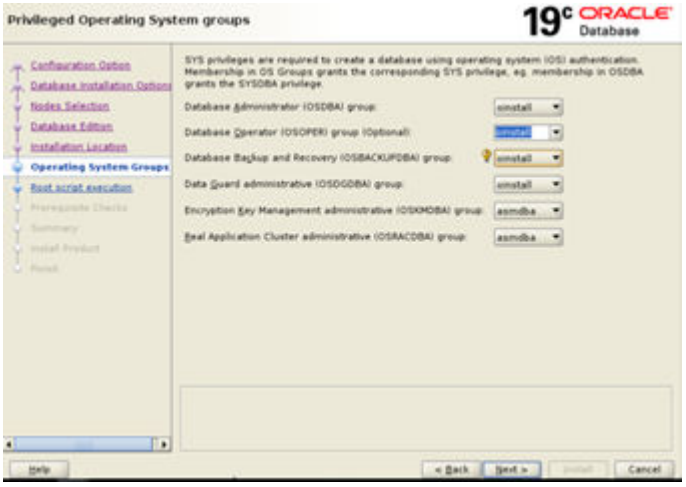


6. Choose Enterprise edition or Standard edition (according to the license purchased).

7. Specify the RDBMS software location.

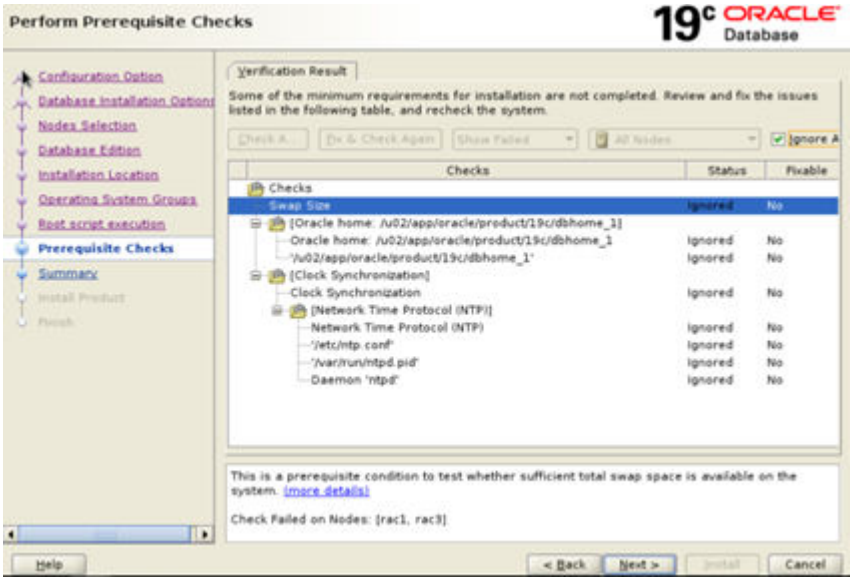


8. Specify OS groups.

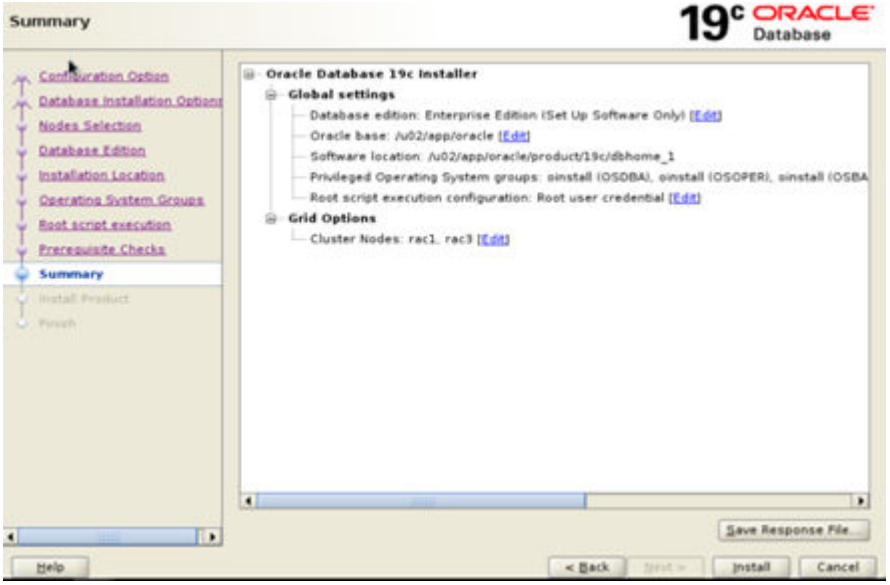


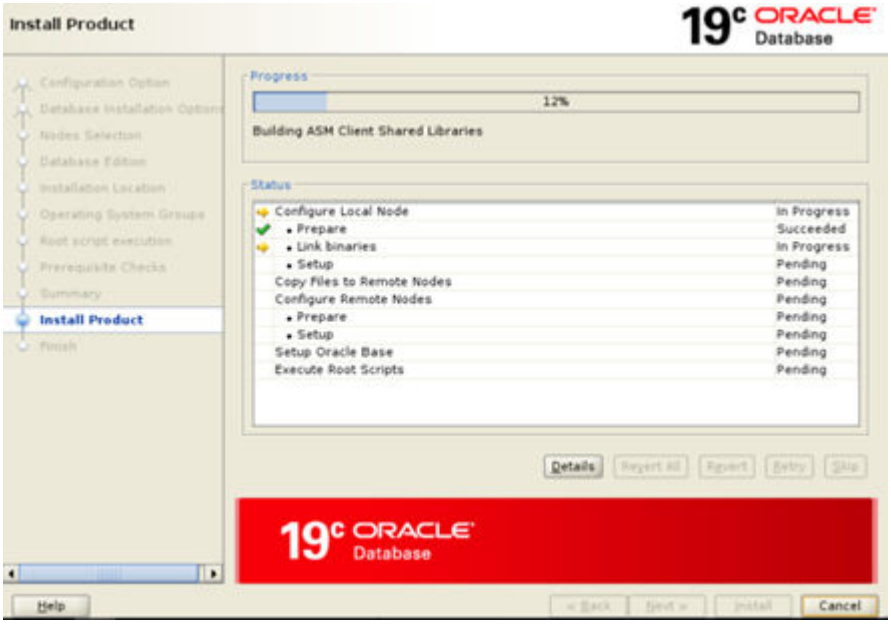
9. Enter the root user password if you would like to run the `root.sh` script automatically. We recommend running it manually from a separate putty session using root user credentials.

10. Check for any failures and correct them.

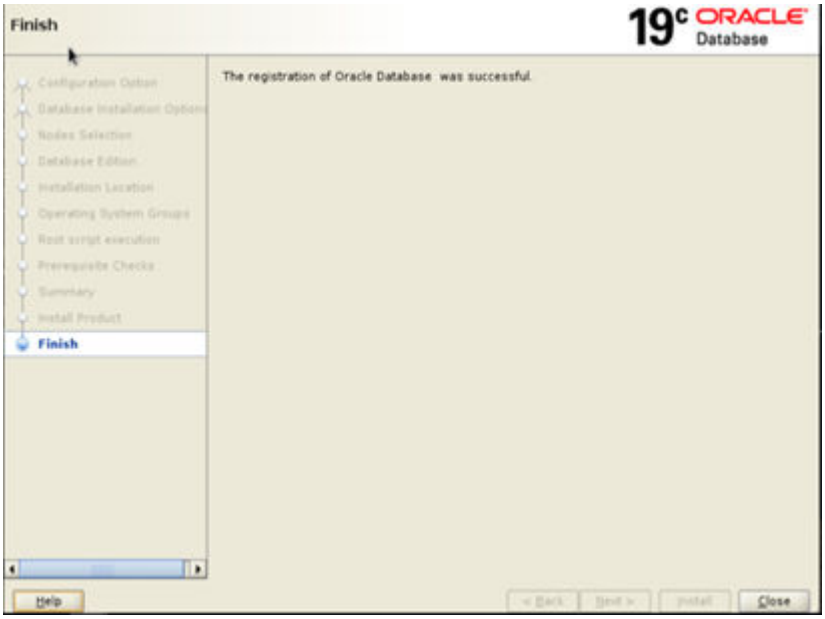

Perform Prerequisite Checks
19c ORACLE Database
Configuration Option
Database Installation Options
Nodes Selection
Database Edition
Installation Location
Operating System Groups
Root script execution
Prerequisite Checks
Summary
Verification Result
Some of the minimum requirements for installation are not completed. Review and fix the issues listed in the following table, and recheck the system.
Checks
Status
Fixable
Swap Size
Ignored
No
This is a prerequisite condition to test whether sufficient total swap space is available on the system. (more details)
Check Failed on Nodes: [rac1, rac3]
Help
< Back
Next >
Install
Cancel


Oracle Database 19c Installer
[INS-13016] You have chosen to ignore some of the prerequisites for this installation. This may impact product configuration.
Are you sure you want to continue ?
Yes
No
Details


Summary
19c ORACLE Database
Configuration Option
Database Installation Options
Nodes Selection
Database Edition
Installation Location
Operating System Groups
Root script execution
Prerequisite Checks
Summary
Oracle Database 19c Installer
Global settings
Database edition: Enterprise Edition (Set Up Software Only) [Edit]
Oracle base: /u02/app/oracle [Edit]
Software location: /u02/app/oracle/product/19c/dbhome_1
Root script execution configuration: Root user credential [Edit]
Grid Options
Cluster Nodes: rac1, rac3 [Edit]
Save Response File...
Help
< Back
Install
Cancel



11. The database RDBMS software is installed successfully.



Database creation

Separate disk groups for **DATA**, **FRA**, and **REDO** are needed for database creation.

Create n number of LUNs from Storage Navigator that are the necessary size. In this implementation, we have created a **200 GB LUN** for each disk group. Add these newly created disks to UDEV rules and format them.

We can also use **ASMLib** instead of UDEV rules to create ASM disks that can be further used for respective disk group creation.

Database creation using Oracle ASMLib

ASMLib is an optional support library for the Automatic Storage Management (ASM) feature of the Oracle Database. ASM simplifies database administration and greatly reduces kernel resource usage. It eliminates the need for the DBA to directly manage potentially thousands of Oracle database files.

You can choose to use Oracle Automatic Storage Management library driver (Oracle ASMLib) or set UDEV rules for device persistence.

See Configuring Storage Device Path Persistence Using Oracle ASMLIB for details.

Procedure

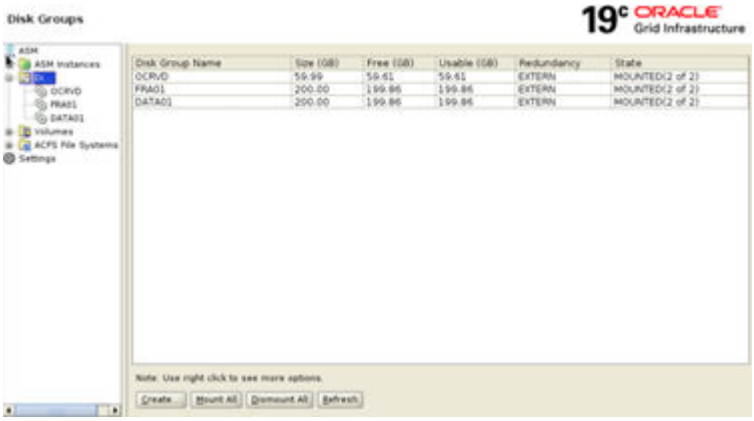
1. Log in to a Grid user and ASM instance.

```
[grid@RAC1 ~]$ . oraenv
ORACLE_SID = [+ASM1] ? +ASM1
ORACLE_HOME = [/home/oracle] ? /u01/app/19c/grid_home1
The Oracle base remains unchanged with value /u01/app/grid
[grid@RAC1 ~]$ asmca
[grid@RAC1 ~]$
```

2. Click **Disk Groups**.

3. Specify **Disk group name** and then check **Select disks**.

4. Create the disk groups.



Database creation using DBCA

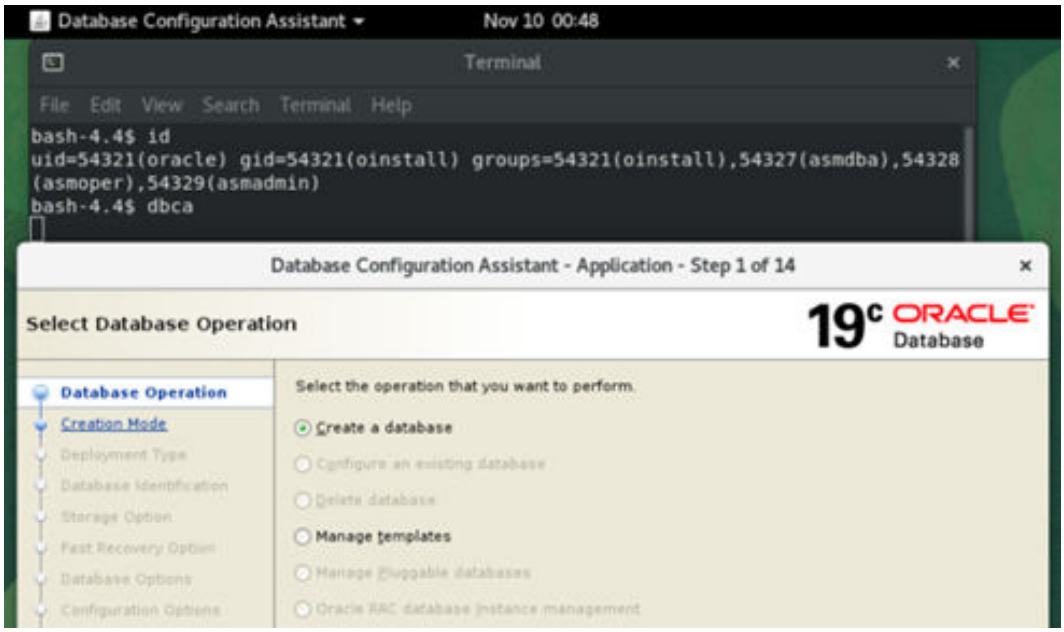
You can use the graphical interface to create a database.

Procedure

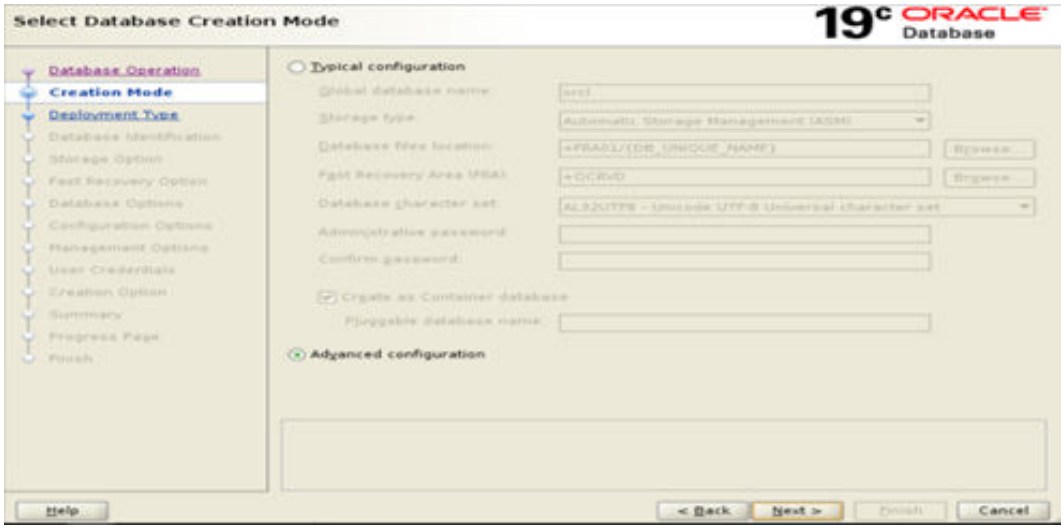
1. Log in to the host using Oracle user credentials and set environment variables.

```
bash-4.4$ export CV_ASSUME_DISTID=OEL7.6
bash-4.4$ export SRVM_DISABLE_MTTRANS=true
bash-4.4$ dbca
```

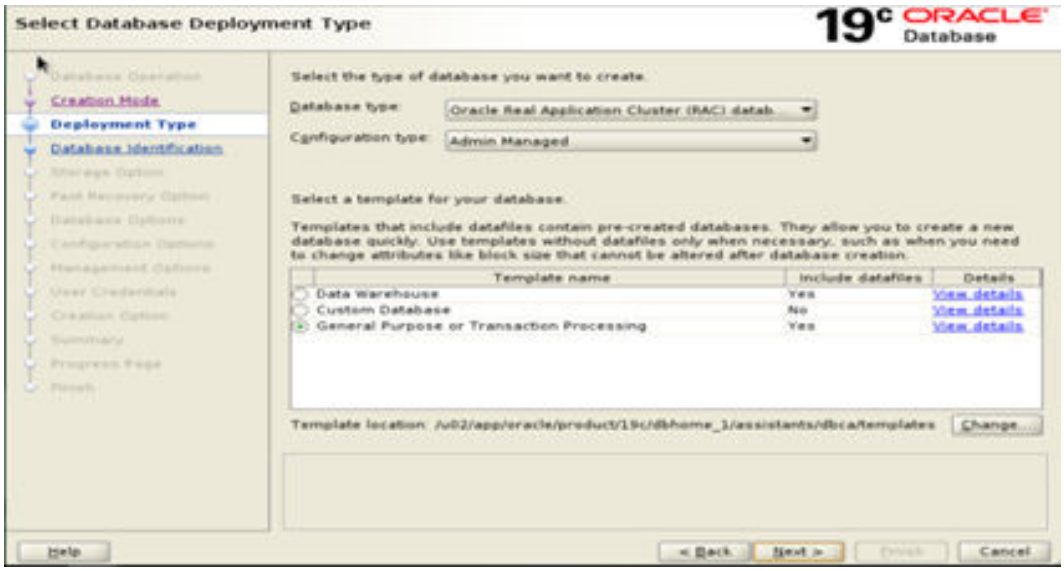
2. Choose **Create a database**.



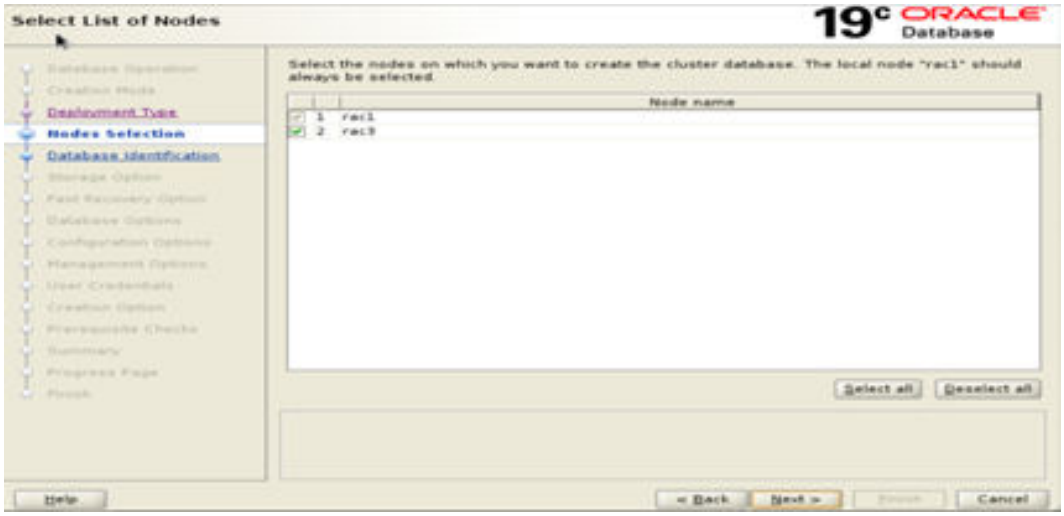
3. Choose **Advanced configuration**.



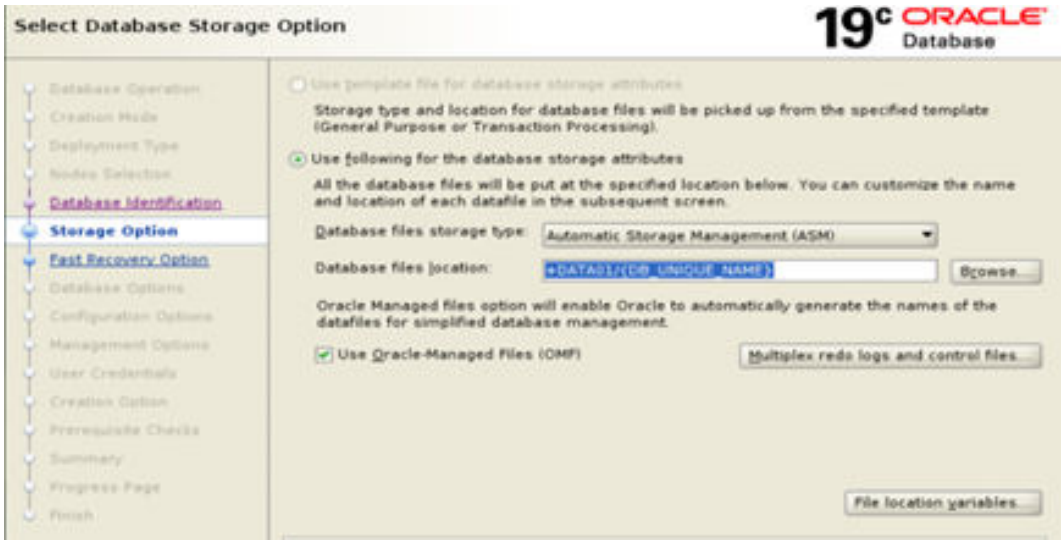
4. Select a database template such as **Data warehouse**, **Custom**, or **General (OLTP)**.



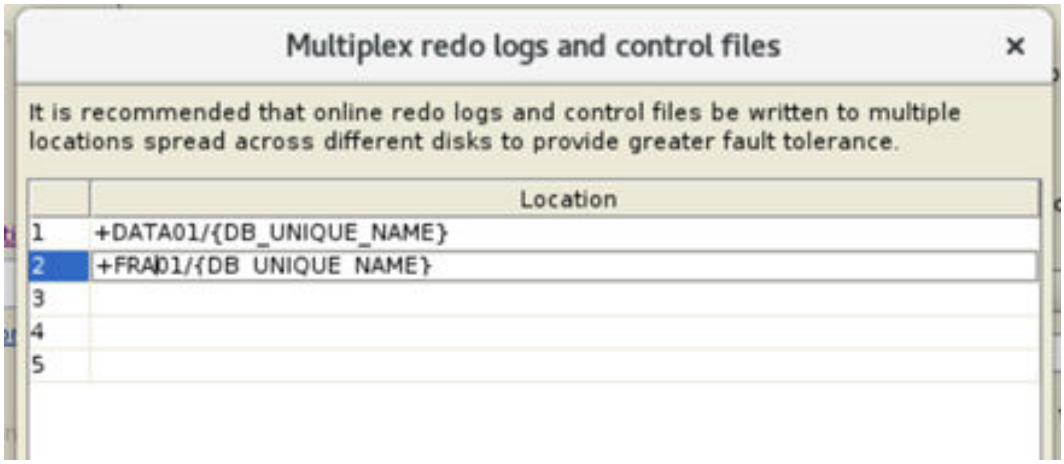
5. Choose the nodes on which the database will be created.



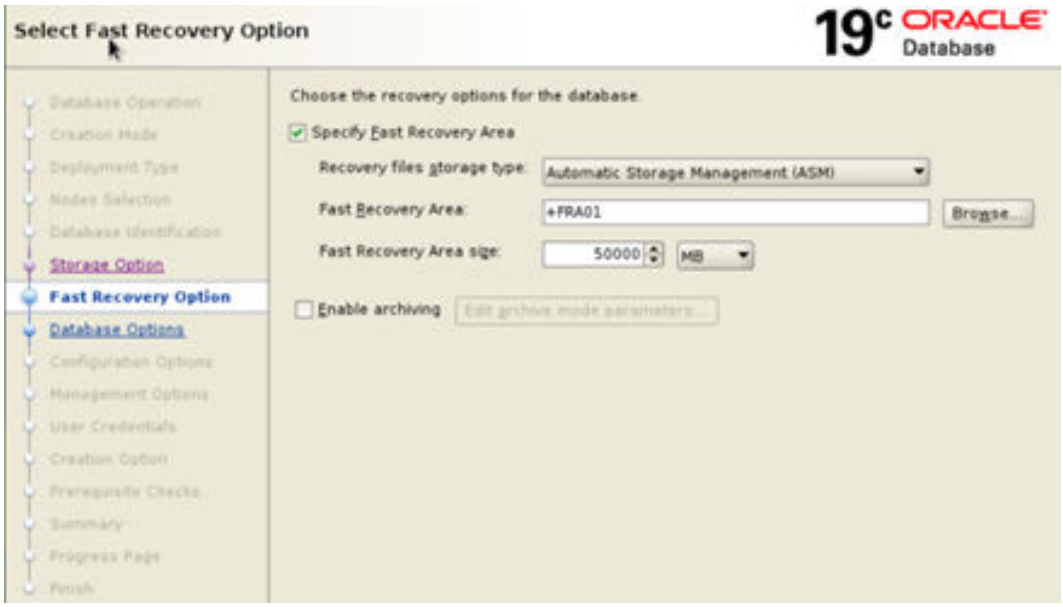
6. Choose storage attributes such as **Automatic Storage Management (ASM)** or **OMF (Oracle Managed Files)**.



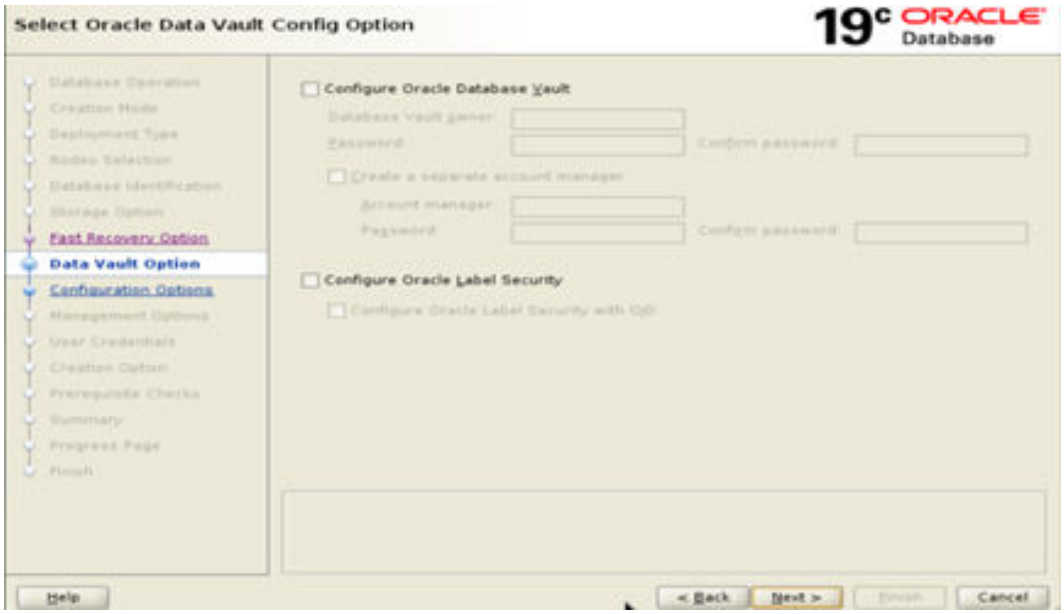
7. Choose disk groups for Oracle files (default tablespace, FRA, and archive location).



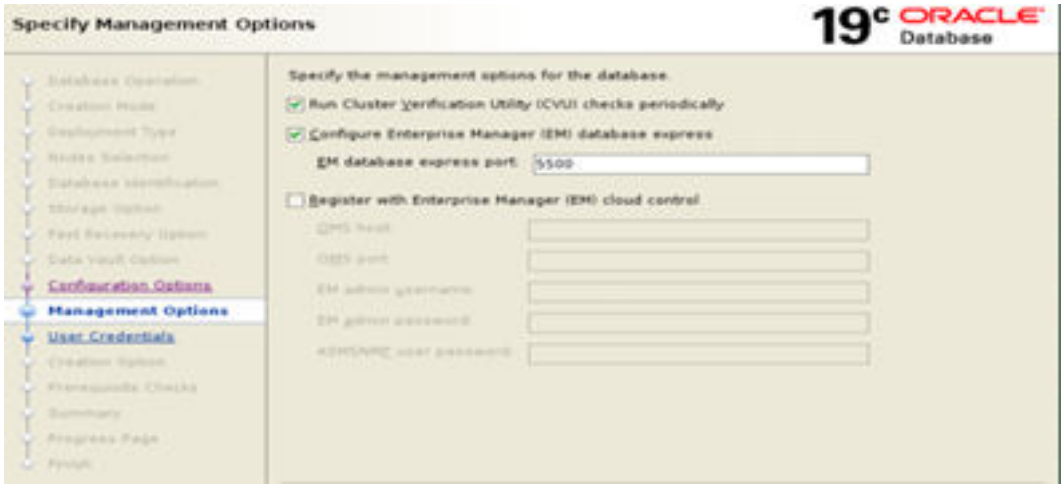
8. Select **Specify Fast Recovery Area** and specify its location (it is recommended to use FRA for recovery-related files for faster recovery in case of any failure).



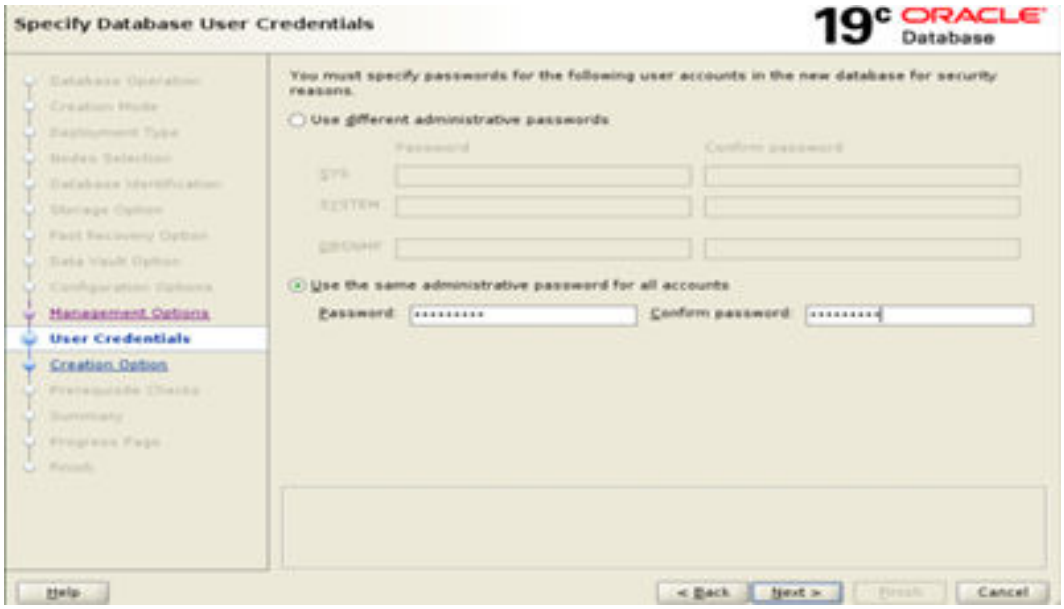
9. Select **Configure Oracle Database Vault**, otherwise click **Next**.



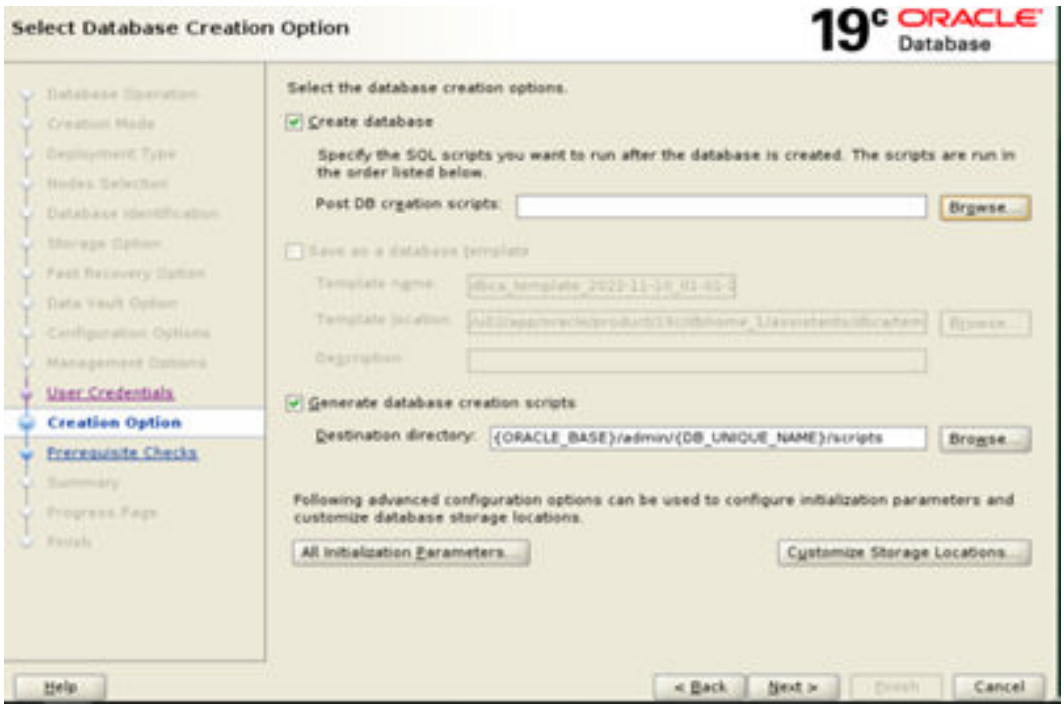
10. Choose **Management Options** to configure Enterprise Manager (EM).



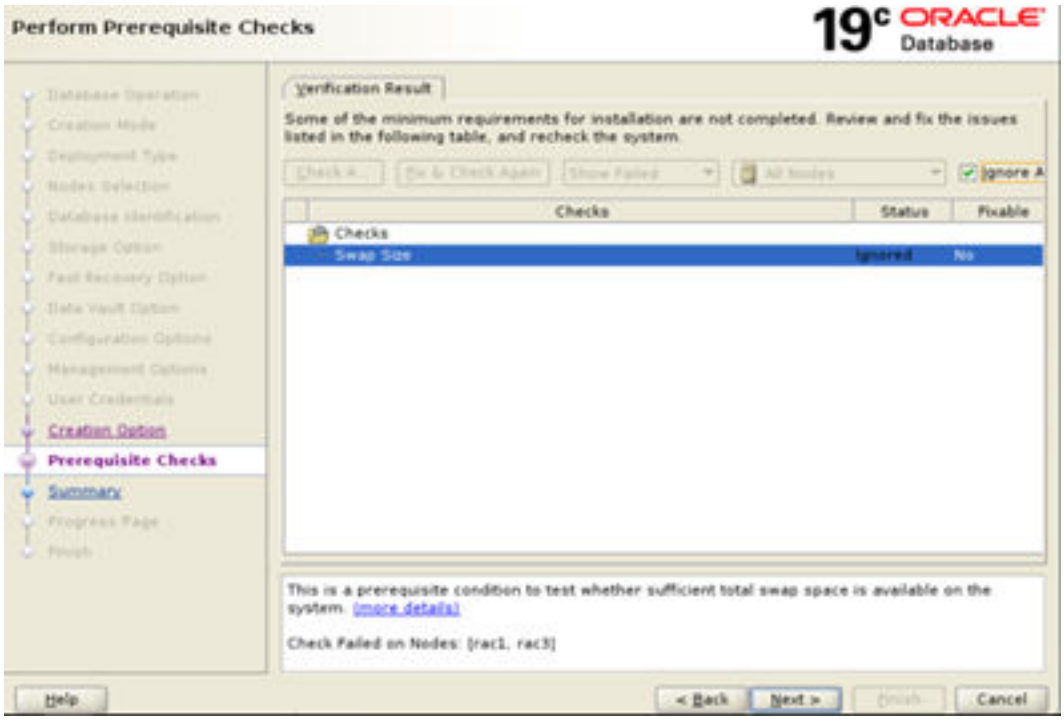
11. Select **User Credentials** to specify passwords (they can be different or the same for all accounts).



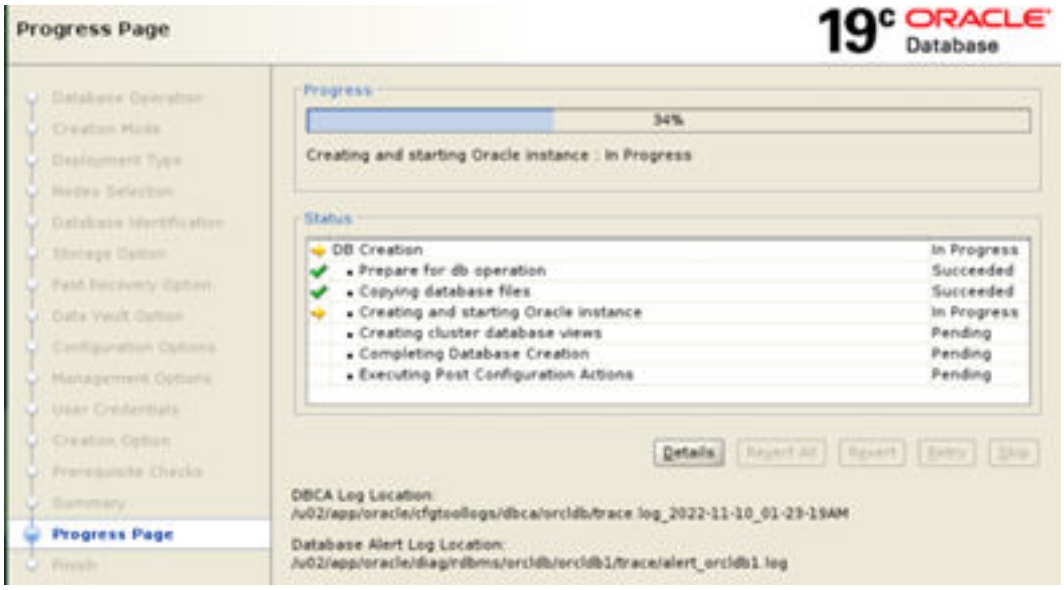
12. Select **Creation Option** to specify information for database scripts.



13. Select **Creation Option** to verify that prerequisites have been met.



14. Monitor the database creation progress.



Result

The database is created successfully.


Finish
19c ORACLE Database
Database Operation
Creation Mode
Deployment Type
Nodes Selection
Database Identification
Storage Option
Fast Recovery Option
Data Vault Option
Configuration Options
Management Options
User Credentials
Creation Option
Prerequisite Checks
Summary
Progress Page
Finish
Database creation complete. For details check the logfiles at:
/u02/app/oracle/cfgtoollogs/dbca/orcldb.
Database Information:
Global Database Name: orcldb
System Identifier(SID) Prefix: orcldb
Server Parameter File name: +DATA01/ORCLDB/PARAMETERFILE/spfile.270.1120365375
EM Database Express URL: https://RAC13-scan:5500/em
Note: All database accounts except SYS, SYSTEM and DBSNMP are locked. Select the Password Management button to view a complete list of locked accounts or to manage the database accounts (except DBSNMP). From the Password Management window, unlock only the accounts you will use. Oracle strongly recommends changing the default passwords immediately after unlocking the account.
Password Management

Chapter 13: Conclusion

Hitachi Virtual Storage Platform E1090 and Hitachi Advanced Server DS220 G2 have been tested and validated as an ideal platform for a virtualized environment using the Oracle KVM hypervisor.

This solution was tested on Oracle RAC database as well as with multiple environments running at the same time.

See the *Hitachi Solution for Databases – Oracle RAC Virtualized on OLVM with DS220 G2 and VSP E1090 Reference Architecture Guide* at https://knowledge.hitachivantara.com/Documents/Application_Optimized_Solutions/Oracle/Hitachi_Solution_for_Databases_%E2%80%93_Oracle_RAC_Virtualized_on_OLVM_with_DS220_G2_and_VSP_E1090_Reference_Architecture_Guide for details about sizing, network design, and configuration.

Hitachi Vantara

Corporate Headquarters
2535 Augustine Drive
Santa Clara, CA 95054 USA

HitachiVantara.com/contact