AlloyDB Omni using HammerDB on DS220 G2 and VSP One Block

Implementation Guide

MK-SL-299-00
July 2024

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	July 2024

Implementation Guide

This guide provides the steps to implement Hitachi Solution for databases for AlloyDB Omni on Hitachi Advanced Server DS220 G2 using Hitachi Virtual Storage Platform One Block storage systems.

Walking through the planning and deployment of an on-premises environment with an RDBMS AlloyDB Omni is at the heart of this guide. This design uses an AlloyDB Omni database on the compute node. This implementation includes configuring storage, configuring the operating system, and configuring the AlloyDB Omni environment.

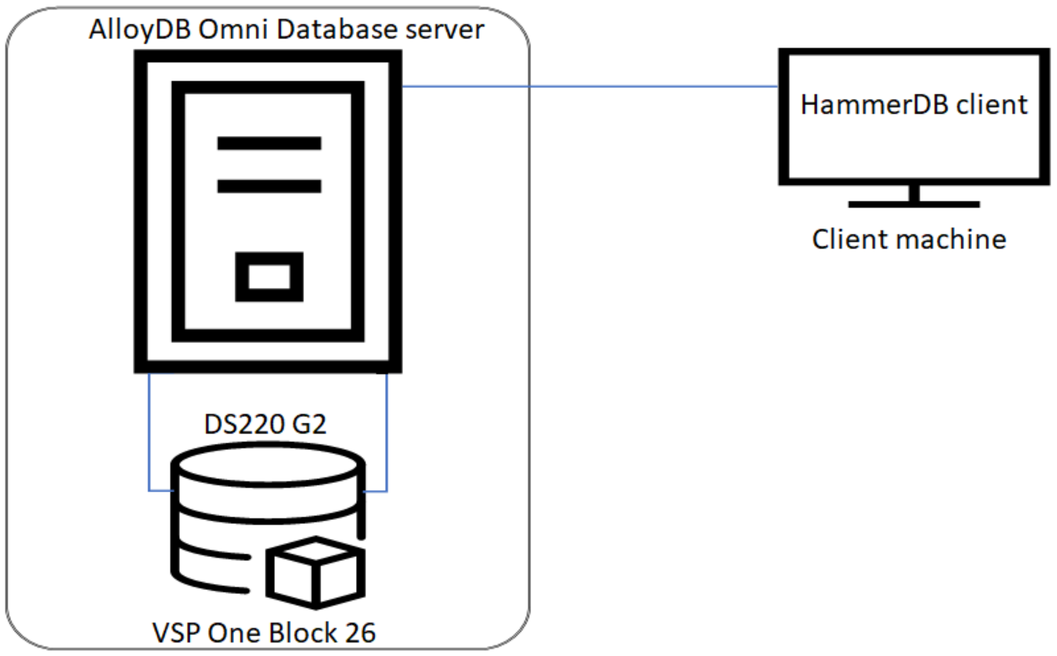
The native operating system on the server is Ubuntu. In this guide, two DS220 G2 bare metal servers are used. One is connected to a VSP One Block storage system over a Fibre Channel network. The storage area network is accessible by compute nodes with native multipathing and zoning configured on switches to allow LUN access to only dedicated servers. The second server is used as client machine that runs sample workloads using HammerDB.

The two bare metal servers are configured with the Ubuntu 20.04 operating system. On top of that, AlloyDB Omni 15.5 software is installed as database software. On the second server HammerDB 4.6 client software is installed to load and run workloads against AlloyDB Omni database.

In this document will see how storage and software are configured while deploying this solution.

Note: Hitachi Solution for AlloyDB Omni is supported by the entire Hitachi portfolio of servers and storage and is not limited to DS220 G2 and VSP One Block.

The following illustration shows the layout of the reference environment.



Storage configuration

In this guide, Hitachi Virtual Storage Platform One Block was used to provide highly available and stable block storage infrastructures for database technologies.

Use this information as a guide to set up and install the storage system.

Hitachi Virtual Storage Platform One Block installation details

Use the corresponding hardware reference guide and quick-start component spreadsheet to complete the VSP One Block installation.

1. Prepare and connect the storage system.
2. Prepare and connect the disk expansion trays to the storage controller.
3. Install disks into the controller or expansion trays.
4. Turn the storage system on.
5. Run the initial startup wizard.
6. Run the initial setup wizard.

The following table lists the storage system components.

Hardware	Description
VSP One Block	CTLAS Controller 768 GB cache memory (12 × 64GB DIMM) Drive Model: SNB5B-R1R9NC / NMVe 1.9 TiB

Hardware	Description
	Number of Drives: 12 Number of Fibre Channel Ports Connected: 2 Fibre Channel Port speed negotiated: 32 Gbps Pool Size: 13.9 TiB Encryption: Enabled Compression Ratio: 2.70:1

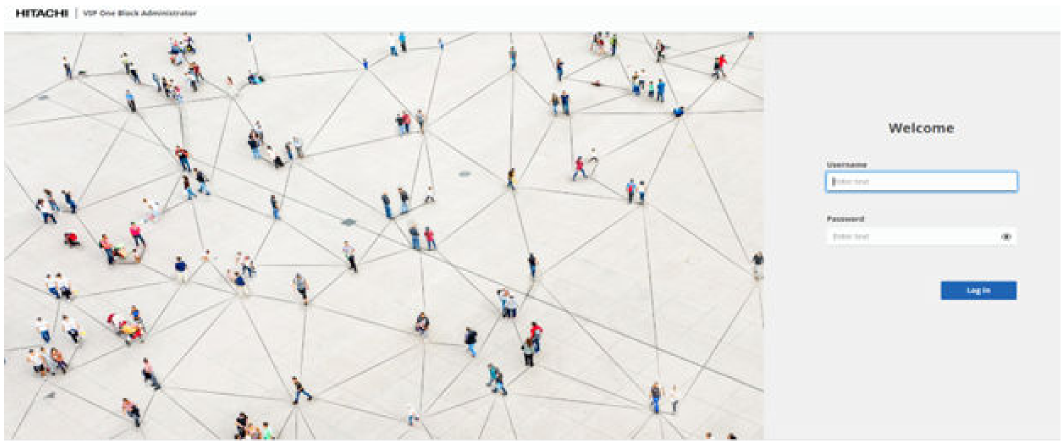
Note: While the lab configuration used these products, the Hitachi solution supports the entire portfolio of Hitachi servers and storage systems.

Pool creation

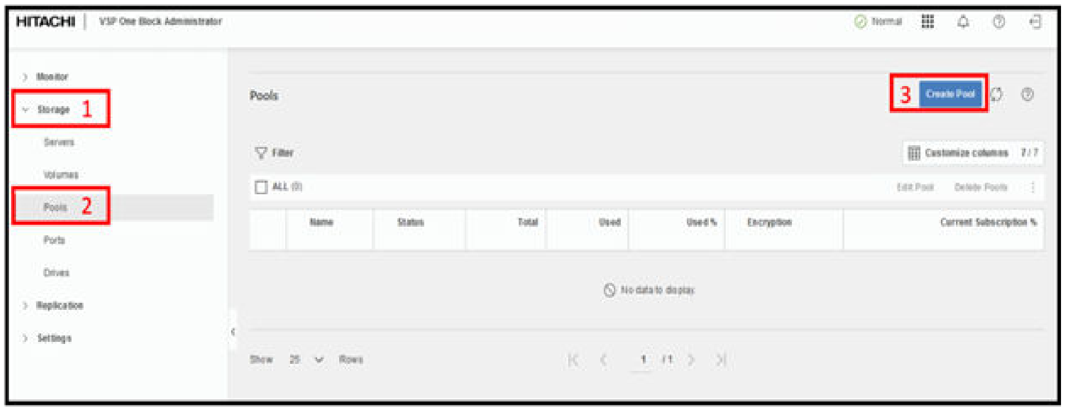
Use this procedure to create a pool.

Procedure

1. Open an internet browser and navigate to https://<ip address>/ui.

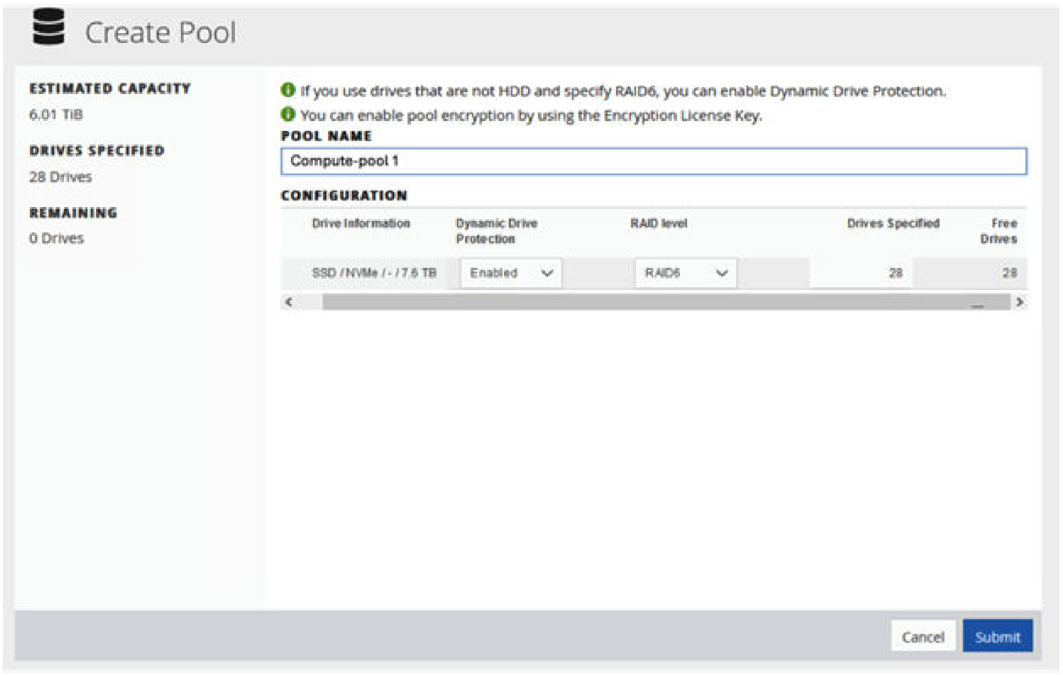


2. Log in with the `maintenance` username and the associated password.
3. Click **Storage** > **Pools** > **Create pool**



4. Enter `Compute Pool 1` as the default pool **Name**.
5. Always set the pool to RAID6.
6. Create the pool with all available space so it can house the management LDEV and the heartbeat LDEVs.
7. Select the pool type as **DDP** and choose all available drives to assign to the pool.
8. Click **Submit**.

   Note: The following image is an example. The drive size and amount used will not match your order.


Create Pool

ESTIMATED CAPACITY
6.01 TiB
DRIVES SPECIFIED
28 Drives
REMAINING
0 Drives

If you use drives that are not HDD and specify RAID6, you can enable Dynamic Drive Protection.
You can enable pool encryption by using the Encryption License Key.

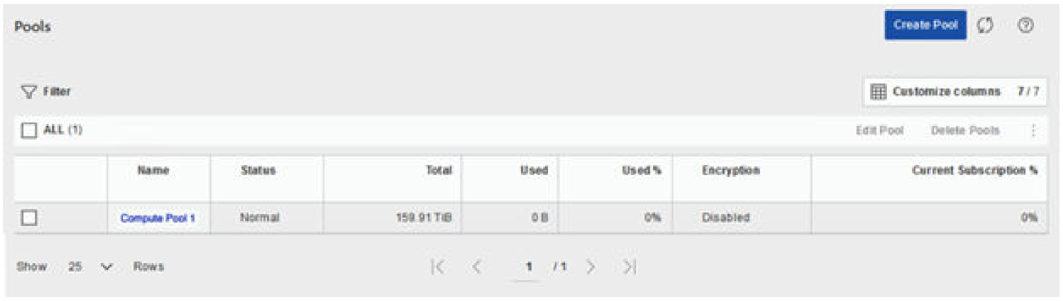
POOL NAME
Compute-pool 1

CONFIGURATION

Drive Information	Dynamic Drive Protection	RAID level	Drives Specified	Free Drives
SSD / NVMe / - / 7.6 TB	Enabled	RAID6	28	28

9. After the pool has been created it will be visible, similar to the following image.

   Note: This illustration is an example. The amount of usable space might be different for your environment.


Pools

	Name	Status	Total	Used	Used %	Encryption	Current Subscription %
☐	Compute Pool 1	Normal	159.91 TiB	0 B	0%	Disabled	0%

Virtual volume creation

Use this procedure to create virtual volumes out of the dynamic drive protection pool to manage virtual machines and the HA heartbeat Storage Repository (SR). The following table provides the volume details.

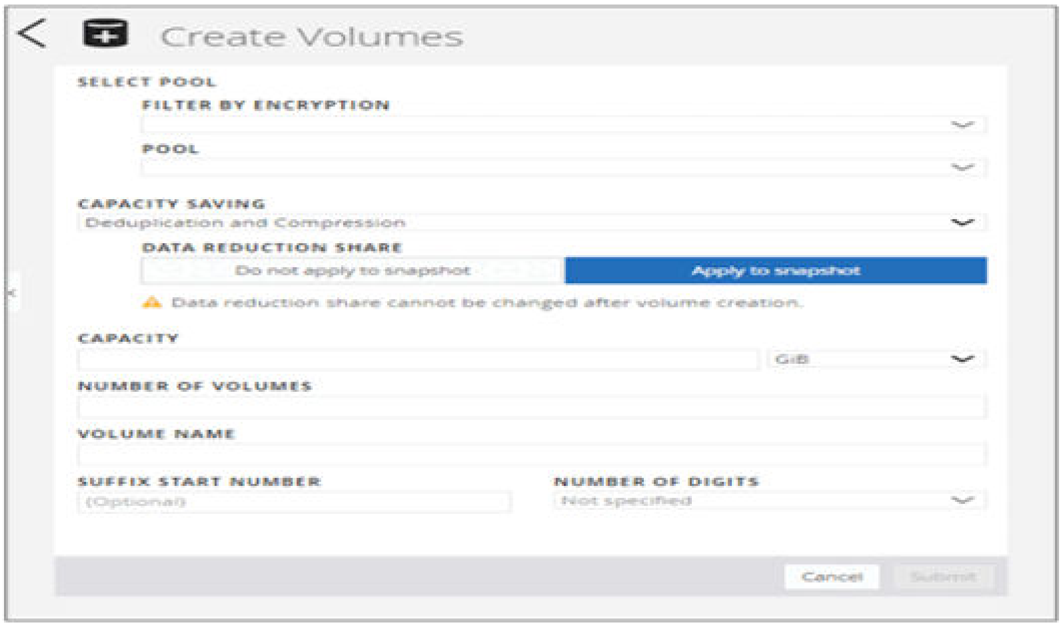
Virtual Volume LDEV ID	Capacity	Volume properties	Pool name
00:0D:00	400 MB	HA-HEARTBEAT-MGMT-LUN1	Compute Pool 1
00:0D:01	1 TB	Management-LUN1	
0:01:00	400 MB	HA-HEARTBEAT-Compute-Pool-1	

The rest of the space will be left unused, because the database can have different size volumes every time, depending on your needs.

Note: The number of LUNs for the compute pool depends on the total capacity. A 400 MB heartbeat LUN (or Quorum disk) must be defined for each compute cluster/pool.

Procedure

1. Under **Storage Systems**, and then click the arrow next to **Pools** to expand the selection.
2. Click **Compute Pool (1)**.
3. Click the **Volumes** tab.
4. Click **Create Volumes**.
5. Select **POOL**.
   - For FILTER BY ENCRYPTION choose **Disabled**.
   - For CAPACITY SAVING, select **Compression**.
   - For DATA REDUCTION SHARE, select **Apply** to snapshot.
   - For CAPACITY, enter the size of the volume based on the previous table.
   - For NUMBER OF VOLUMES, enter **1**.
   - For SUFFIX START NUMBER, leave at **(Optional)**.
   - For NUMBER OF DIGITS, leave at **Not specified**.



6. Click **Submit**.

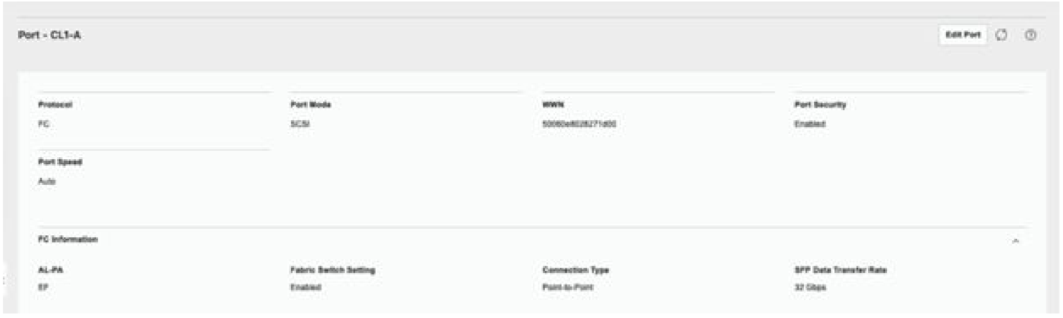
Next steps

Repeat this procedure until all 3 volumes are created.

Configure ports

Procedure

1. In the Web-based GUI for VSP One Block, click **Storage** > **Ports**.
2. Select the ports to configure, and then click **Edit Ports**.



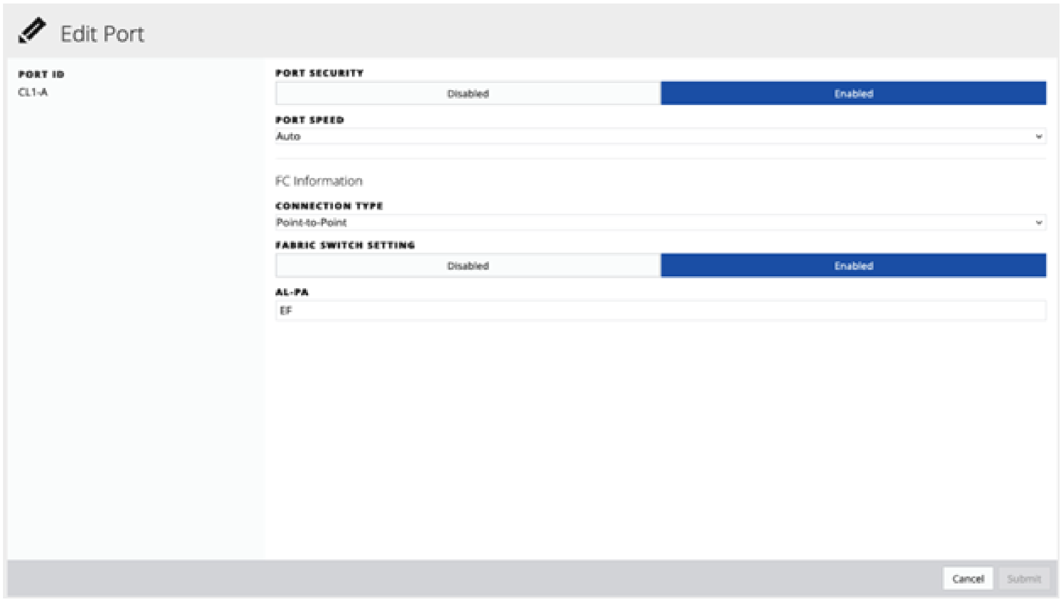
Port - CL1-A

Protocol: FC
Port Mode: SCSI
WWN: 50060e8028271d00
Port Security: Enabled
Port Speed: Auto

FC Information

AL-PA: EF
Fabric Switch Setting: Enabled
Connection Type: Point-to-Point
SFP Data Transfer Rate: 32 Gbps

3. Configure the ports:
   - Port Security: **Enabled**
   - Port Speed: **32 Gbps**, **64 Gbps**, or fastest available speed
   - Connection Type: **P-to-P**
   - Fabric Switch Setting: select **Enabled**
   - AL-PA : Leave the default value or enter a different value



PORT ID
CL1-A

PORT SECURITY
Disabled | Enabled

PORT SPEED
Auto

FC Information

CONNECTION TYPE
Point-to-Point

FABRIC SWITCH SETTING
Disabled | Enabled

AL-PA
EF

4. Click **Submit**.

Create servers (previously known as host groups)

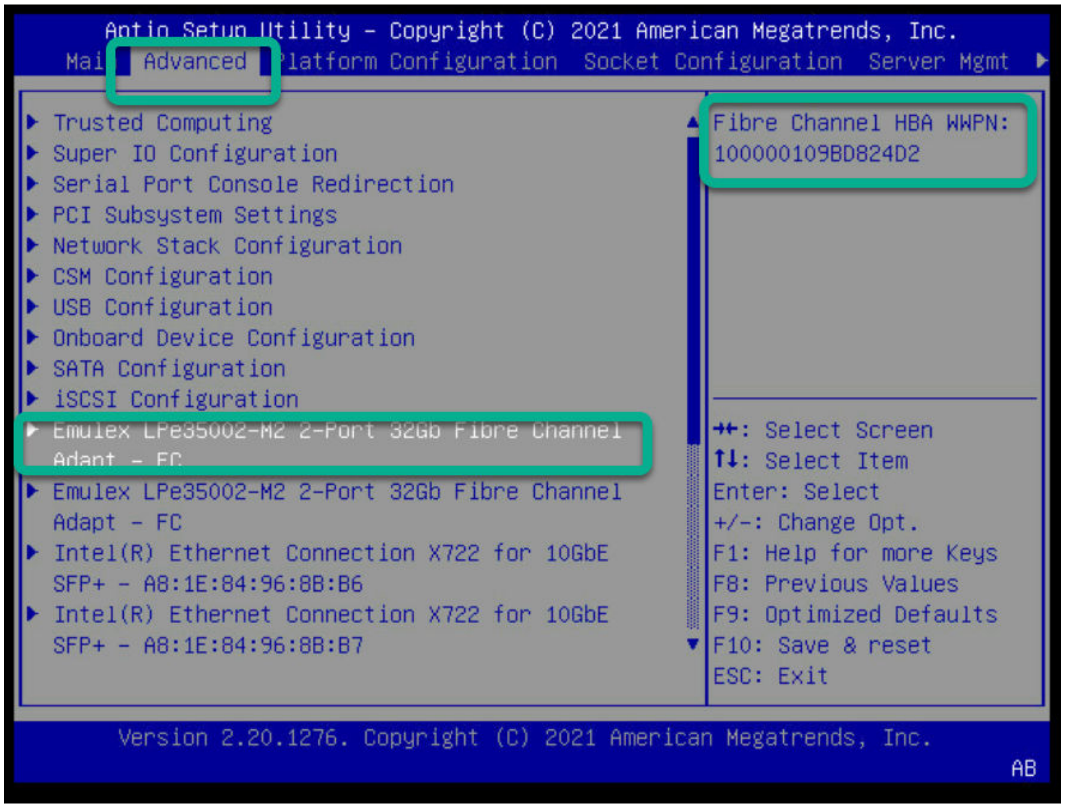
Note: For a single host with no cluster (or Xen pool): Create one server and place one WWN in the single server. For multiple hosts and clusters (or pools): Create one server and place all the WWNs for the clustered hosts in the single server. This ensures that when adding LDEVs to the server, all hosts see the same LUNs. This creates consistency with LUN presentation across all hosts.

Procedure

1. Obtain the HBA information from the target server using the BIOS menu. Use one of the following procedures that follow to get the HBA WWN.
   a. Gather the HBA WWN from the BIOS:
      i. Launch the browser and connect to the BMC IP address.

         Note: The default login username and password are located underneath the pull-out tag of each server node.

      ii. On the BMC interface, go to the boot menu by pressing F2.
      iii. Click **Advanced**.

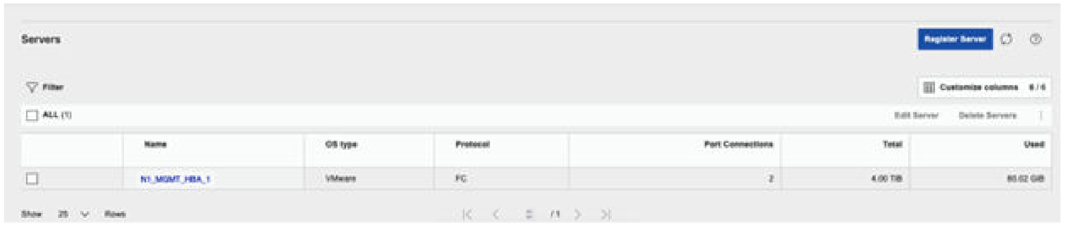


```
Aptio Setup Utility - Copyright (C) 2021 American Megatrends, Inc.
Main  Advanced  Platform Configuration  Socket Configuration  Server Mgmt ▶

▶ Trusted Computing                                    Fibre Channel HBA WWPN:
▶ Super IO Configuration                               100000109BD824D2
▶ Serial Port Console Redirection
▶ PCI Subsystem Settings
▶ Network Stack Configuration
▶ CSM Configuration
▶ USB Configuration
▶ Onboard Device Configuration
▶ SATA Configuration
▶ iSCSI Configuration
▶ Emulex LPe35002-M2 2-Port 32Gb Fibre Channel         ↔: Select Screen
  Adapt - FC                                           ↑↓: Select Item
▶ Emulex LPe35002-M2 2-Port 32Gb Fibre Channel         Enter: Select
  Adapt - FC                                           +/-: Change Opt.
▶ Intel(R) Ethernet Connection X722 for 10GbE          F1: Help for more Keys
  SFP+ - A8:1E:84:96:8B:B6                             F8: Previous Values
▶ Intel(R) Ethernet Connection X722 for 10GbE          F9: Optimized Defaults
  SFP+ - A8:1E:84:96:8B:B7                             F10: Save & reset
                                                       ESC: Exit

Version 2.20.1276. Copyright (C) 2021 American Megatrends, Inc.
                                                                          AB
```

b. In the Web-based GUI, click **Storage** > **Servers** > **Register Server**.

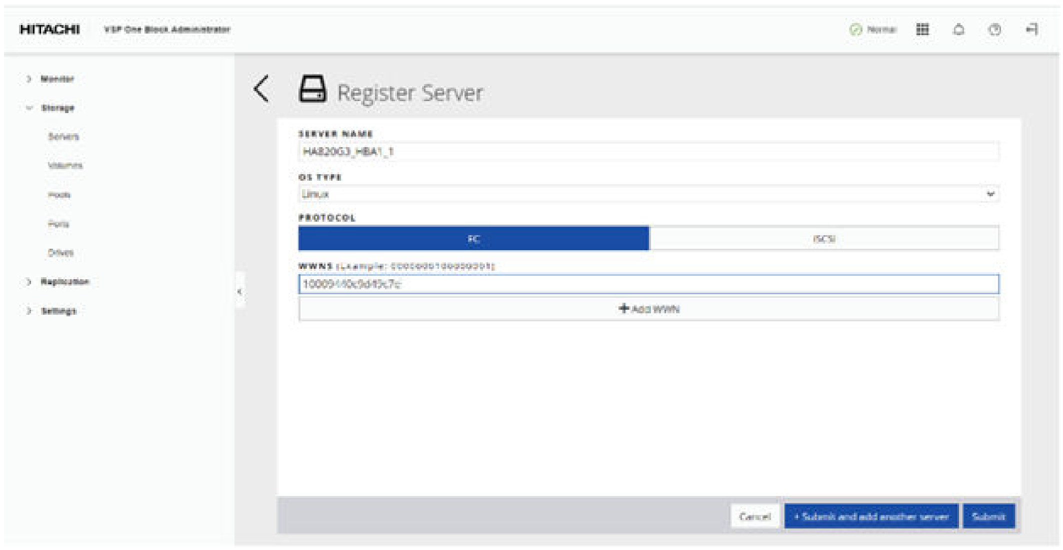


Servers

	Name	OS type	Protocol	Port Connections	Total	Used
☐	N1_MGMT_HBA_1	VMware	FC	2	4.00 TiB	60.02 GiB

- SERVER NAME: Type a server name, for example: N1_MGMT_HBA1_1
- OS TYPE: Select the correct OS from the dropdown list
- PROTOCOL: Select FC.
- HBA WWN: Type the HBA WWN, for example: 100000109BD824D2

After entering all the input, you will see a screen similar to the following.


Register Server

SERVER NAME: HA820G3_HBA1_1

OS TYPE: Linux

PROTOCOL: FC | iSCSI

WWNS (Example: 0000000100000001): 100091400c9d49c7c

2. Click **Submit**.

> **Note:** In this example the HA820G3 was used, but you can use a different server.

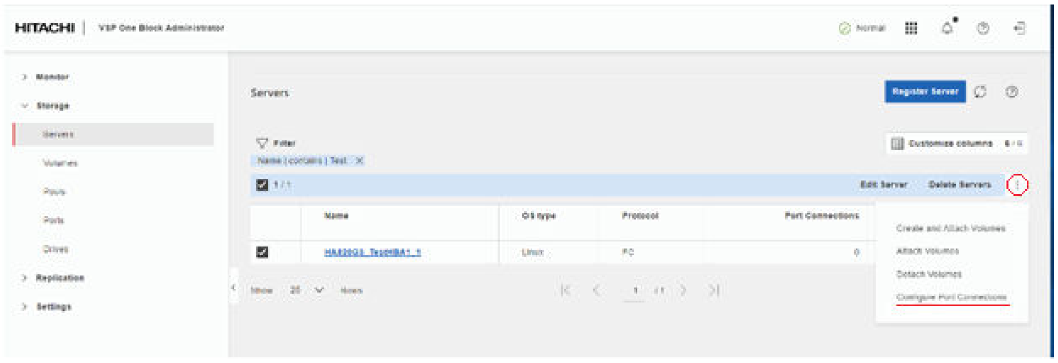
Next steps

Repeat this procedure until all servers are created/registered.

Configure port connections

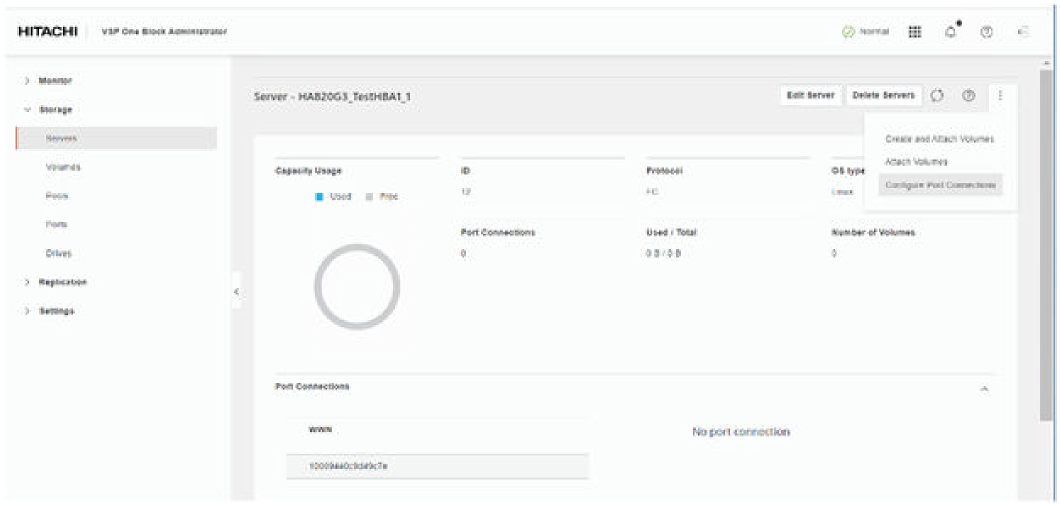
Procedure

1. In the Web-based GUI click **Storage** > **Servers**.
2. Select the checkbox for the server.
3. Click the three dots, and then click **Configure Port Connections**.


Servers

Filter: Name | contains | Test

	Name	OS type	Protocol	Port Connections
✓	HA820G3_TestHBA1_1	Linux	FC	0



Server - HA820G3_TestHBA1_1

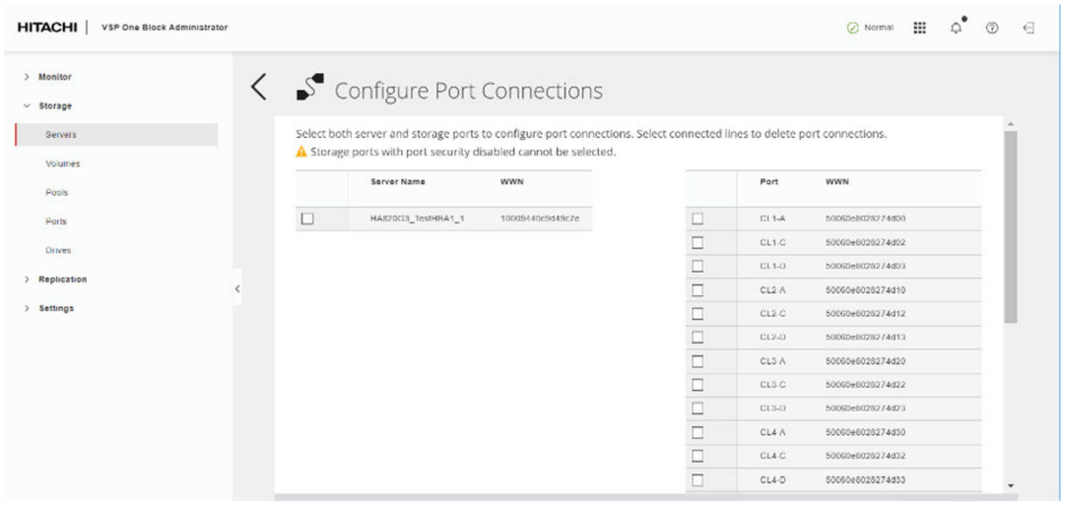
Capacity Usage	ID	Protocol	OS type
	12	FC	Linux
	Port Connections	Used / Total	Number of Volumes
	0	0 B / 0 B	0

Port Connections

WWN
10009440c9d49c7e

No port connection

You will see a screen similar to the following.



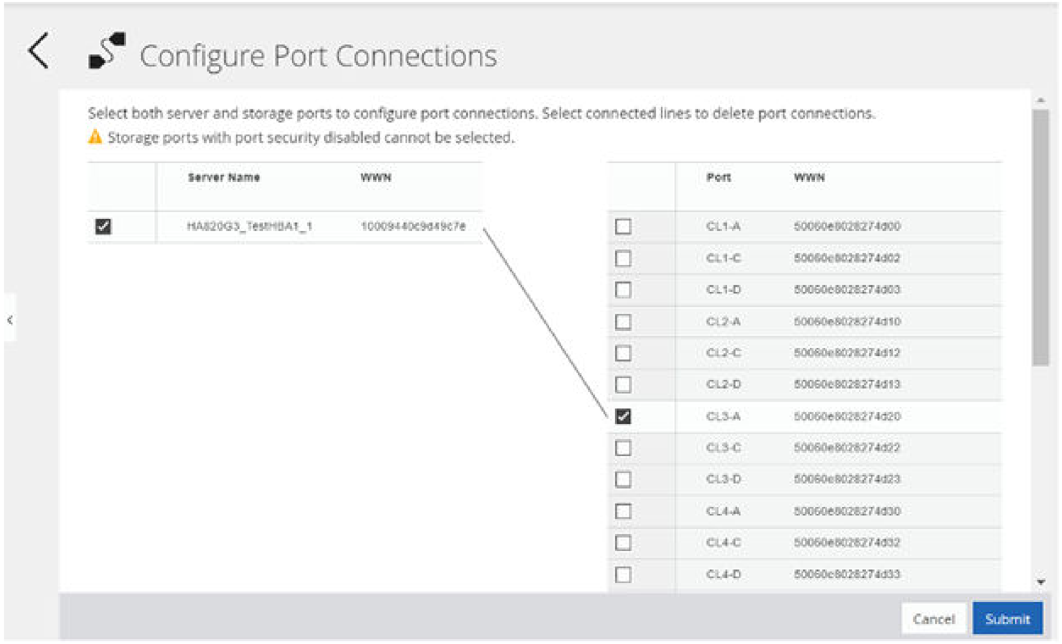
Configure Port Connections

Select both server and storage ports to configure port connections. Select connected lines to delete port connections.
⚠ Storage ports with port security disabled cannot be selected.

	Server Name	WWN
☐	HA820G3_TestHBA1_1	10009440c9d49c7e

	Port	WWN
☐	CL1-A	50060e8028274d00
☐	CL1-C	50060e8028274d02
☐	CL1-D	50060e8028274d03
☐	CL2-A	50060e8028274d10
☐	CL2-C	50060e8028274d12
☐	CL2-D	50060e8028274d13
☐	CL3-A	50060e8028274d20
☐	CL3-C	50060e8028274d22
☐	CL3-D	50060e8028274d23
☐	CL4-A	50060e8028274d30
☐	CL4-C	50060e8028274d32
☐	CL4-D	50060e8028274d33

4. Select server and storage ports as follows to make the necessary connections.



Configure Port Connections

Select both server and storage ports to configure port connections. Select connected lines to delete port connections.
⚠ Storage ports with port security disabled cannot be selected.

	Server Name	WWN
☑	HA820G3_TestHBA1_1	10009440c9d49c7e

	Port	WWN
☐	CL1-A	50060e8028274d00
☐	CL1-C	50060e8028274d02
☐	CL1-D	50060e8028274d03
☐	CL2-A	50060e8028274d10
☐	CL2-C	50060e8028274d12
☐	CL2-D	50060e8028274d13
☑	CL3-A	50060e8028274d20
☐	CL3-C	50060e8028274d22
☐	CL3-D	50060e8028274d23
☐	CL4-A	50060e8028274d30
☐	CL4-C	50060e8028274d32
☐	CL4-D	50060e8028274d33

5. Click **Submit**.

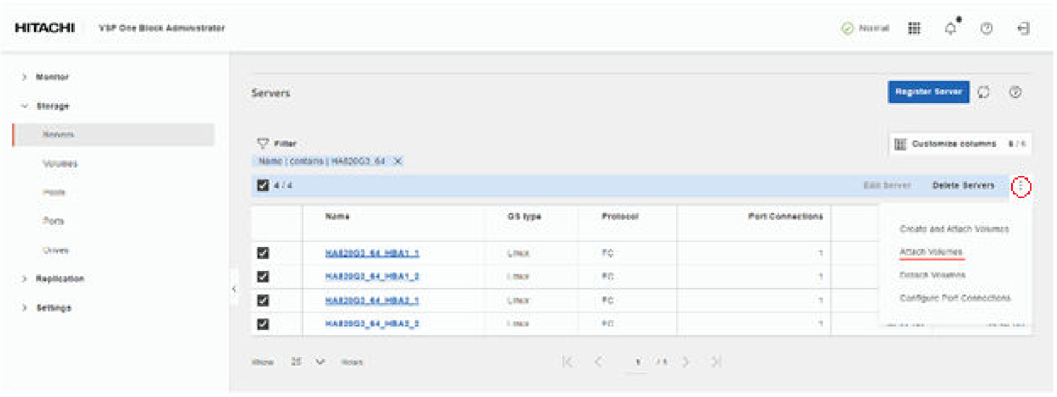
Attach volumes to servers

Add LDEVs according to the following table.

LDEV ID	Storage Port	Server
00:0D:00 00:0D:01 00:01:00	CL1-A/CL4-A	All servers
00:0D:00 00:0D:01 00:01:00	CL2-A/CL3-A	All servers

Procedure

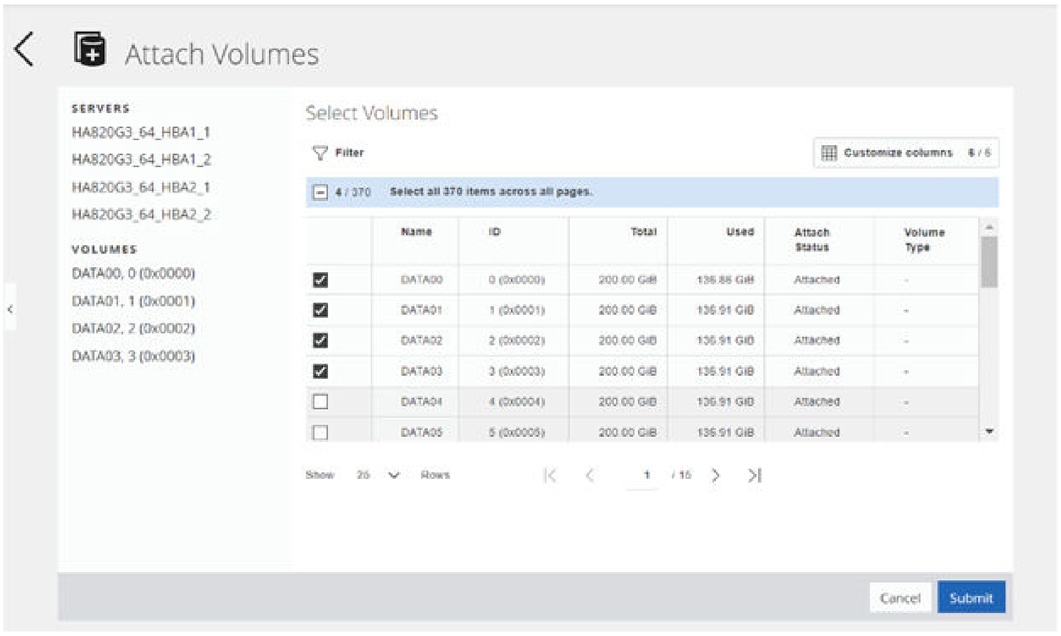
1. In the Web-based GUI click **Storage** > **Servers**.
2. Select the checkbox for the server.
3. Click the three dots, and then click **Attach Volumes**.



	Name	OS type	Protocol	Port Connections
✓	HA820G3_64_HBA1_1	Linux	FC	1
✓	HA820G3_64_HBA1_2	Linux	FC	1
✓	HA820G3_64_HBA2_1	Linux	FC	1
✓	HA820G3_64_HBA2_2	Linux	FC	1

4. Select the volumes to be attached as follows and click **Submit**.

> **Note:** This screenshot is an example, and you will not see all the volumes listed here. There will only be the three that you created earlier.



Select Volumes

SERVERS
HA820G3_64_HBA1_1
HA820G3_64_HBA1_2
HA820G3_64_HBA2_1
HA820G3_64_HBA2_2

VOLUMES
DATA00, 0 (0x0000)
DATA01, 1 (0x0001)
DATA02, 2 (0x0002)
DATA03, 3 (0x0003)

4 / 370 Select all 370 items across all pages.

	Name	ID	Total	Used	Attach Status	Volume Type
✓	DATA00	0 (0x0000)	200.00 GiB	136.86 GiB	Attached	-
✓	DATA01	1 (0x0001)	200.00 GiB	136.91 GiB	Attached	-
✓	DATA02	2 (0x0002)	200.00 GiB	136.91 GiB	Attached	-
✓	DATA03	3 (0x0003)	200.00 GiB	136.91 GiB	Attached	-
☐	DATA04	4 (0x0004)	200.00 GiB	136.91 GiB	Attached	-
☐	DATA05	5 (0x0005)	200.00 GiB	136.91 GiB	Attached	-

Next steps

Repeat this procedure to attach all the volumes required to configure the cluster and database over all the registered servers.

Install the operating system

In this guide we used Hitachi Advanced Server DS220 G2 as a compute node and client node. The detailed server configuration is listed in the following table.

Vendor	Hardware	Detail Description	Quantity
Hitachi Vantara	Hitachi Advanced Server DS220 G2	2 × Intel Xeon Gold 6342 (24C, 2.8GHz, 230W) 512 GB memory (16 × 32 GB DDR4 R-DIMM 3200MHz) 256 GB NVMe 0.3DWPD M.2 SSD, Intel E810 Dual Port 25GbE LP SFP28 Eth, LPE35002-M2 2port 32G FC ADAPTER	2

The following are the recommended system requirements:

- 2 GHz dual-core processor
- 4 GB memory
- 25 GB available disk space for storage (less if installing the minimum version)
- DVD drive or USB port

Use this procedure to install the operating system on the server and client nodes. We installed Ubuntu version 20.04 on both nodes.

These are the high-levels steps for installing the operating system:

1. Launch a remote console on the bare metal server and then verify that it is turned off.
2. Verify that the compute node is turned off.
3. Mount the installation image.
4. Load and configure the operating system.
5. Select the installation destination.
6. Install the operating system.
7. Complete the initial setup and customization.

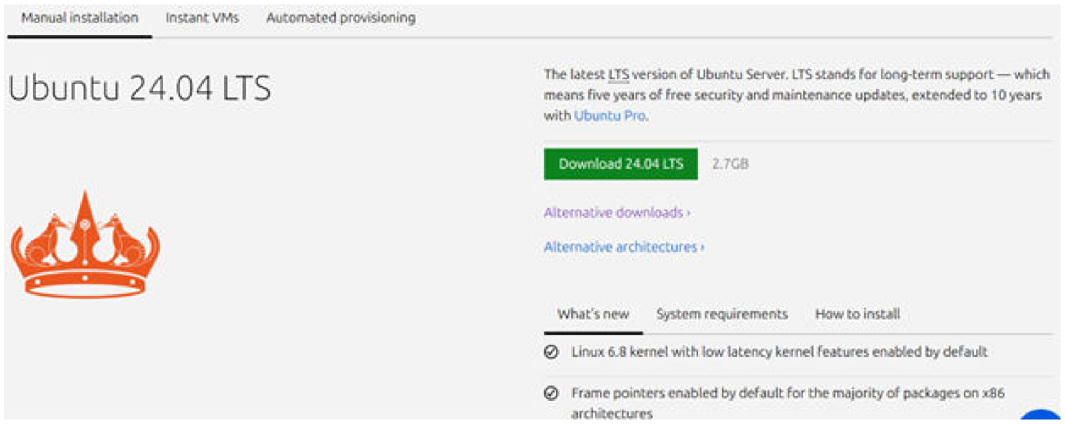
Before you begin

Before you begin, configure the BMC IP addresses for all servers.

Procedure

1. Launch the remote console on the bare metal server.
   a. Open an HTML5-based web browser and enter the IP address of a bare metal server into the address bar.
   b. Log in using the following user credentials.
      Username: `admin`

      Password: (If you do not have the initial password, contact your Hitachi Vantara representative.)
   c. Click **Remote Console**.
   d. To start the remote console, click **Launch KVM (Java)**.
2. Verify that the compute node is turned off. If the Power button on the top-right corner of the remote console is green, click the Power button to turn the node off.
3. Download Software.
   a. In a browser go to https://ubuntu.com/download and download the server software.



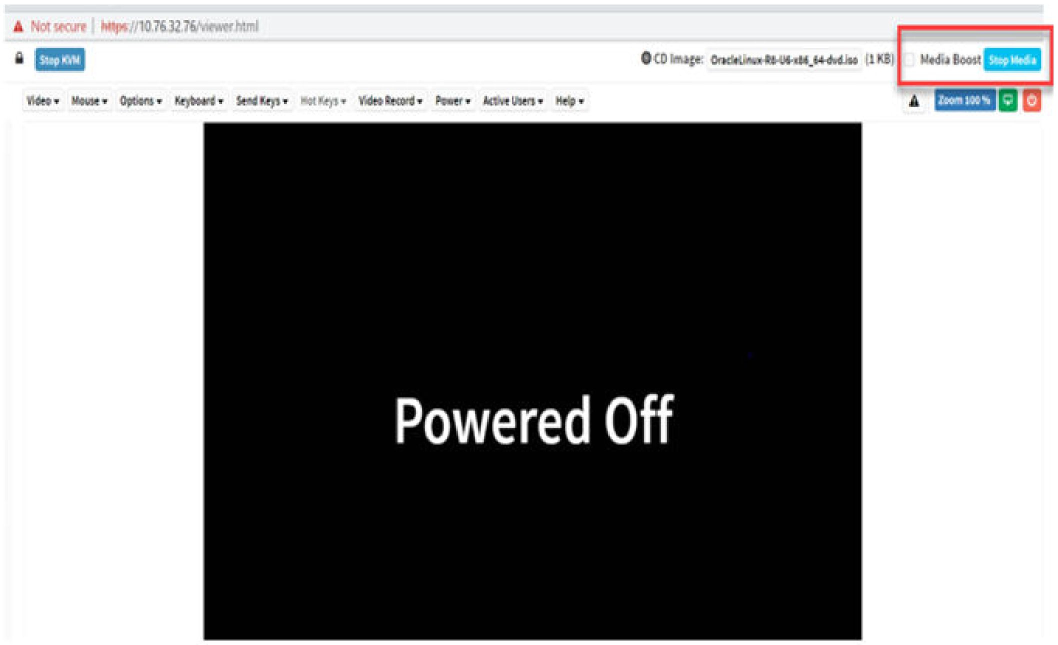
Ubuntu 24.04 LTS

The latest LTS version of Ubuntu Server. LTS stands for long-term support — which means five years of free security and maintenance updates, extended to 10 years with Ubuntu Pro.

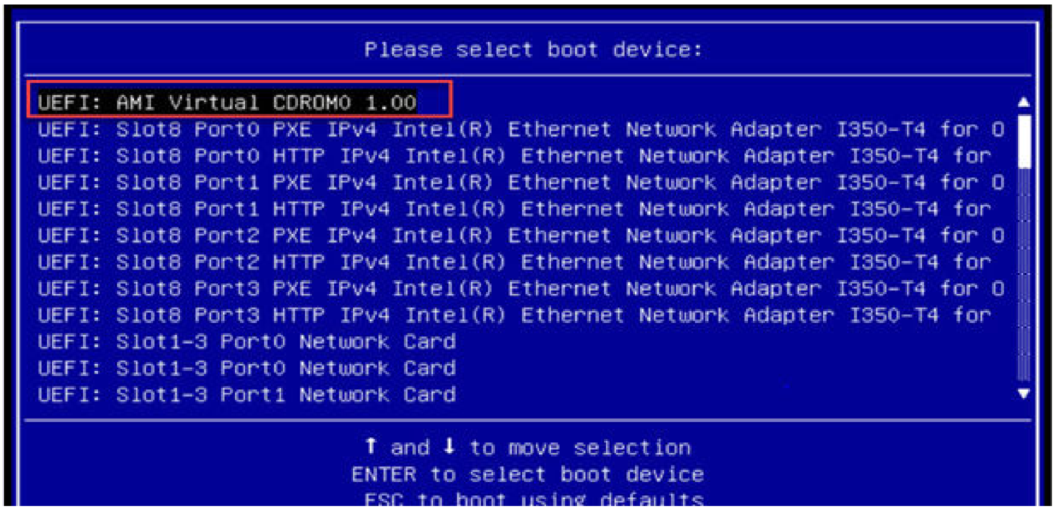
Linux 6.8 kernel with low latency kernel features enabled by default

Frame pointers enabled by default for the majority of packages on x86 architectures

   The download is an .iso file. You can use it to create a bootable USB drive. Save the file to the location of your choice.
4. Mount the installation image (ISO file).
   a. Click **Media boost** and upload the ISO image that was downloaded previously.
   b. Use the Power button to turn on the server.



c. Open the Boot menu by pressing F11 while the server is starting and select **UEFI:AMI Virtual CDROM0 1.00**. The Ubuntu ISO loads automatically.

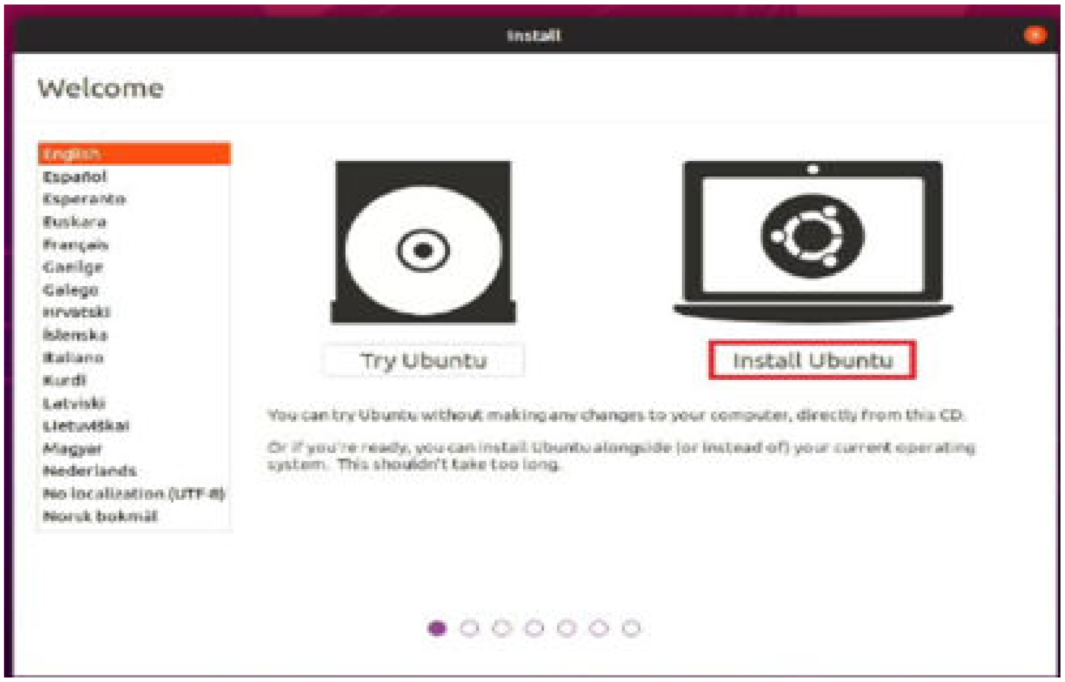


```
                    Please select boot device:

UEFI: AMI Virtual CDROM0 1.00
UEFI: Slot8 Port0 PXE IPv4 Intel(R) Ethernet Network Adapter I350-T4 for O
UEFI: Slot8 Port0 HTTP IPv4 Intel(R) Ethernet Network Adapter I350-T4 for
UEFI: Slot8 Port1 PXE IPv4 Intel(R) Ethernet Network Adapter I350-T4 for O
UEFI: Slot8 Port1 HTTP IPv4 Intel(R) Ethernet Network Adapter I350-T4 for
UEFI: Slot8 Port2 PXE IPv4 Intel(R) Ethernet Network Adapter I350-T4 for O
UEFI: Slot8 Port2 HTTP IPv4 Intel(R) Ethernet Network Adapter I350-T4 for
UEFI: Slot8 Port3 PXE IPv4 Intel(R) Ethernet Network Adapter I350-T4 for O
UEFI: Slot8 Port3 HTTP IPv4 Intel(R) Ethernet Network Adapter I350-T4 for
UEFI: Slot1-3 Port0 Network Card
UEFI: Slot1-3 Port0 Network Card
UEFI: Slot1-3 Port1 Network Card

                    ↑ and ↓ to move selection
                    ENTER to select boot device
                    ESC to boot using defaults
```

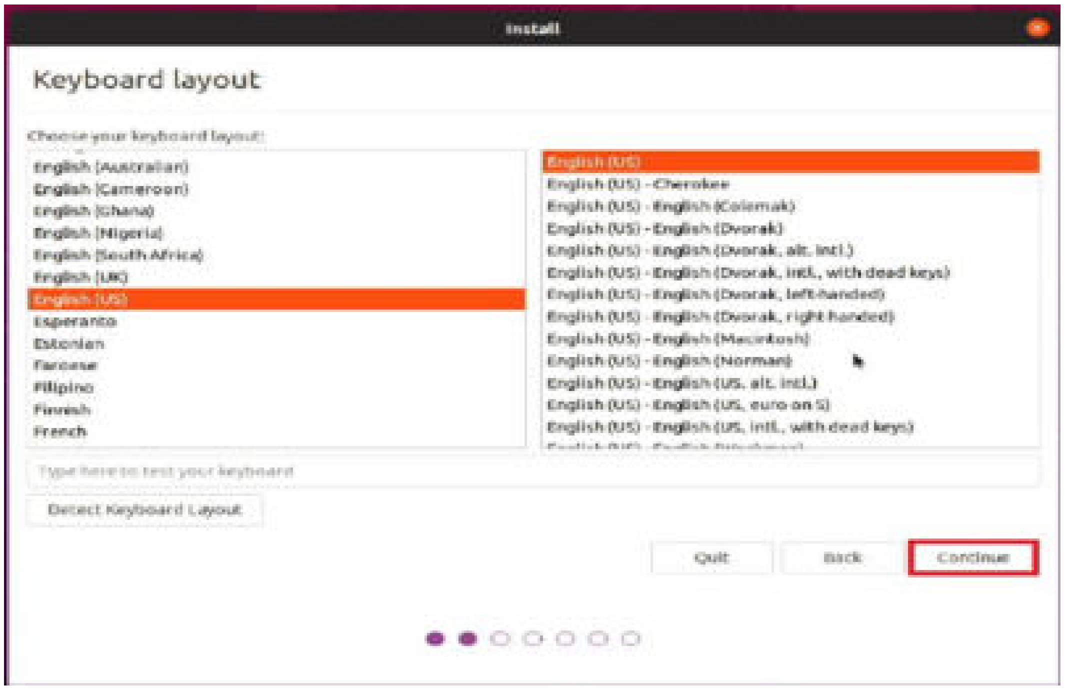
d. Select **Install Ubuntu** on the Welcome screen.



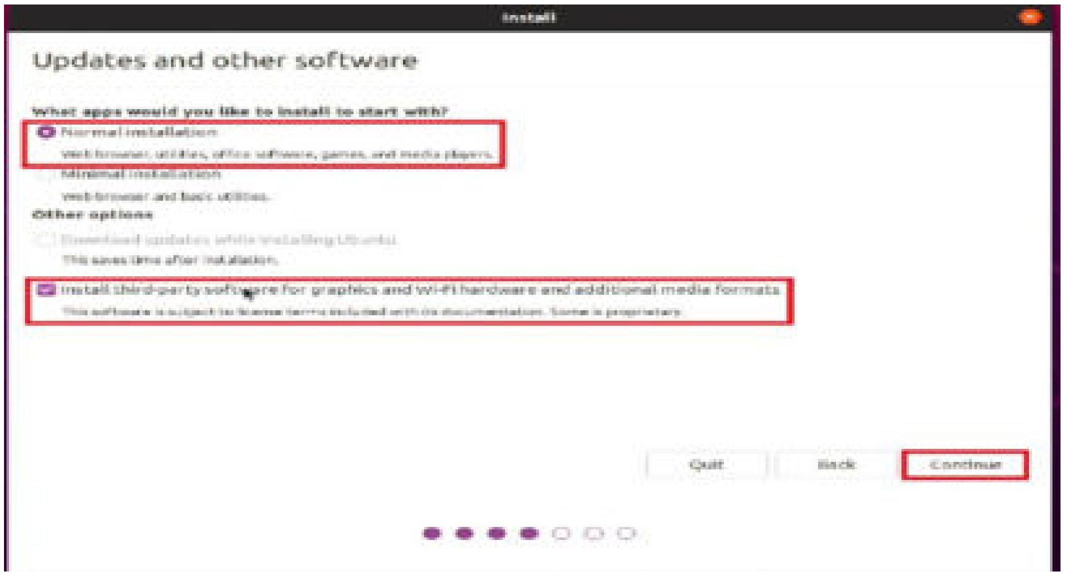
You can try Ubuntu without making any changes to your computer, directly from this CD.

Or if you're ready, you can install Ubuntu alongside (or instead of) your current operating system. This shouldn't take too long.

e. Select the keyboard layout and language and click **Continue**.


Install
Keyboard layout
Choose your keyboard layout:
English (Australian)
English (Cameroon)
English (Ghana)
English (Nigeria)
English (South Africa)
English (UK)
English (US)
Esperanto
Estonian
Faroese
Filipino
Finnish
French
English (US)
English (US) - Cherokee
English (US) - English (Colemak)
English (US) - English (Dvorak)
English (US) - English (Dvorak, alt. intl.)
English (US) - English (Dvorak, intl., with dead keys)
English (US) - English (Dvorak, left-handed)
English (US) - English (Dvorak, right-handed)
English (US) - English (Macintosh)
English (US) - English (Norman)
English (US) - English (US, alt. intl.)
English (US) - English (US, euro on 5)
English (US) - English (US, intl., with dead keys)
Type here to test your keyboard
Detect Keyboard Layout
Quit
Back
Continue

f. Select **Normal Installation** and **Install third-party software**, and then click **Continue**.


Install

Updates and other software

What apps would you like to install to start with?

[✓ Normal installation] Web browser, utilities, office software, games, and media players.

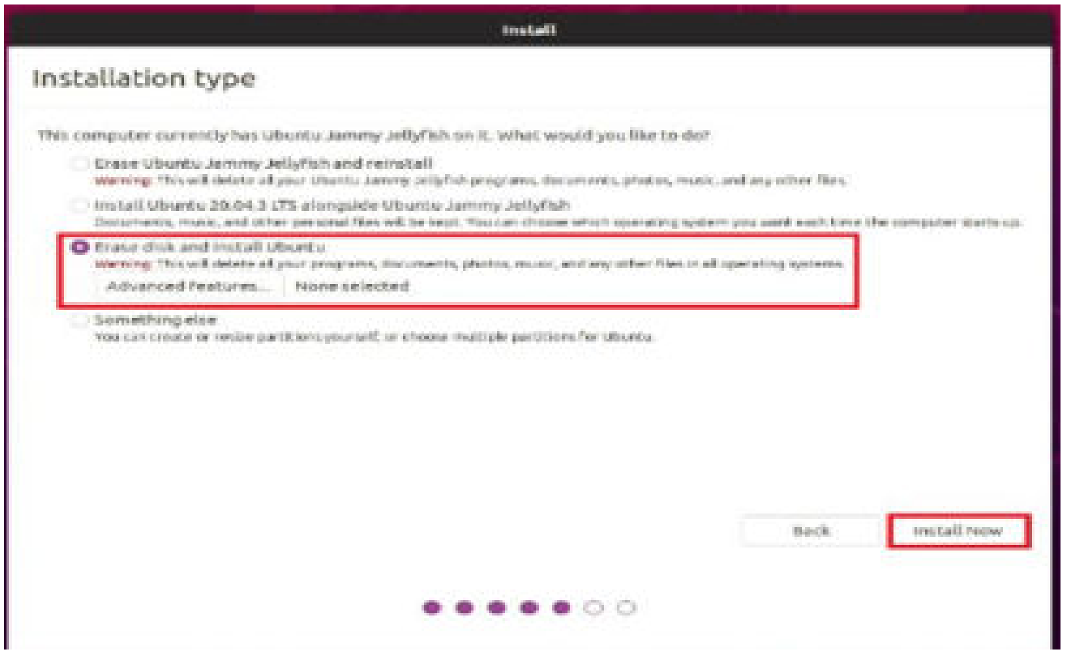
Minimal installation Web browser and basic utilities.

Other options

Download updates while installing Ubuntu This saves time after installation.

[✓ Install third-party software for graphics and Wi-Fi hardware and additional media formats] This software is subject to license terms included with its documentation. Some is proprietary.

g. Choose the installation type and click **Install now**.


Install

Installation type

This computer currently has Ubuntu Jammy Jellyfish on it. What would you like to do?

Erase Ubuntu Jammy Jellyfish and reinstall
Warning: This will delete all your Ubuntu Jammy Jellyfish programs, documents, photos, music, and any other files.

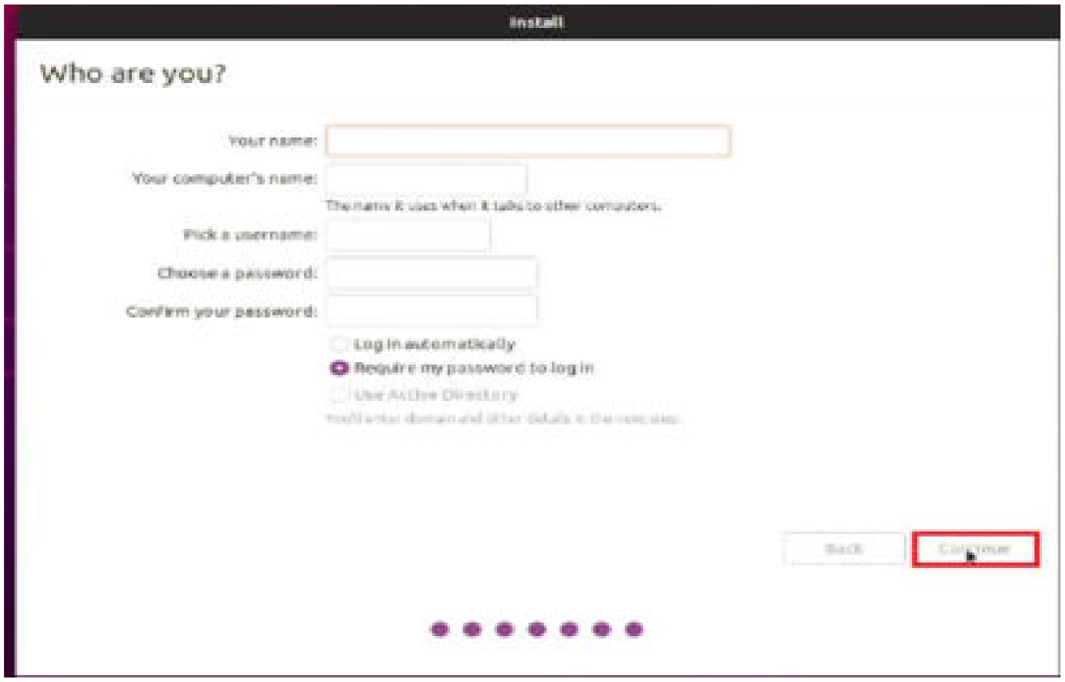
Install Ubuntu 20.04.3 LTS alongside Ubuntu Jammy Jellyfish
Documents, music, and other personal files will be kept. You can choose which operating system you want each time the computer starts up.

[✓ Erase disk and install Ubuntu]
Warning: This will delete all your programs, documents, photos, music, and any other files in all operating systems.
Advanced Features... None selected

Something else
You can create or resize partitions yourself, or choose multiple partitions for Ubuntu.

h. Choose the geographical location and select **Continue**.

i. Fill out information about you and select **Continue**.



Who are you?

Your name:

Your computer's name:

The name it uses when it talks to other computers.

Pick a username:

Choose a password:

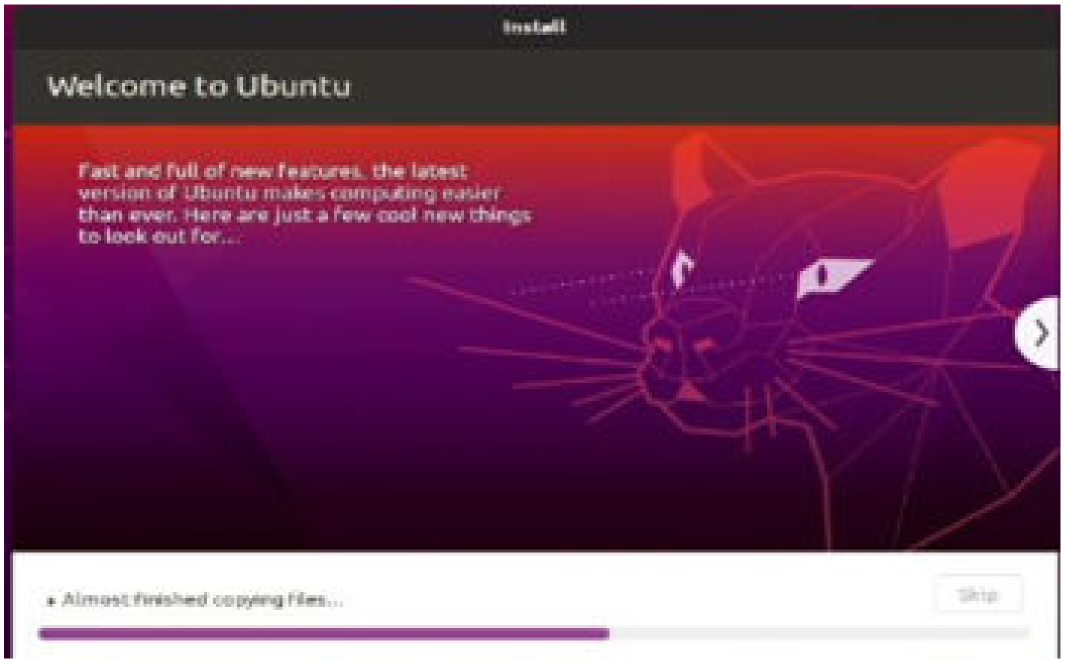
Confirm your password:

Log in automatically [✓ Require my password to log in]

Use Active Directory

You'll enter domain and other details in the next step.

Ubuntu installation begins, and you can monitor the progress bar on the screen.



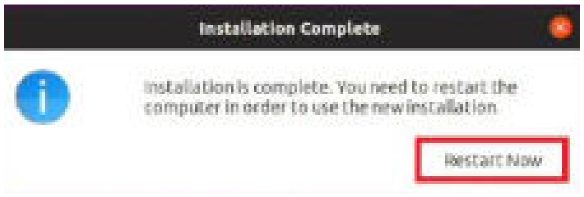
Welcome to Ubuntu

Fast and full of new features, the latest version of Ubuntu makes computing easier than ever. Here are just a few cool new things to look out for...

Almost finished copying files...

Result

When the installation completes the “Restart now” message appears.



Installation is complete. You need to restart the computer in order to use the new installation.

Next steps

After installation completes, configure multipathing for storage disks.

Multipath configuration

Multipathing (DM-Multipath) is a native multipathing in Linux. Device Mapper Multipathing (DM-Multipath) can be used for redundancy and to improve the performance.

- The configuration file is */etc/multipath.conf*
- Take a backup of the file. Edit the configuration file to ensure you have the following entries uncommented out.

```
defaults {
user_friendly_names yes
find_multipaths yes
}
devices {
device {
vendor HITACHI
product Hi-SDS
path_grouping_policy group_by_prio
prio alua
path_checker readsector0
no_path_retry 6
dev_loss_tmo infinity}
}
multipaths {
multipath {
wwid 360060e8028279d005080279d00000001
alias data01}
}
```

Start the multipath service if it is not started by default, and check multipathing status. It will show two paths to the storage device.

```
root@omni:/# multipath -ll
mpatha (360060e8028279d005080279d00000000) dm-1 HITACHI,OPEN-V
size=2.0T features='0' hwhandler='0' wp=rw
`-+- policy='service-time 0' prio=1 status=active
  |- 6:0:0:0  sdc     8:32  active ready running
  `- 17:0:0:0 sdd     8:48  active ready running
root@omni:/#
```

Create and mount disks on the filesystem for database installation.

```
root@omni:/# pvs
  PV             VG        Fmt  Attr PSize   PFree
  /dev/nvme0n1p3 ubuntu-vg lvm2 a--  235.42g 135.42g
root@omni:/# pvcreate /dev/mapper/mpatha
  Physical volume "/dev/mapper/mpatha" successfully created.
```

```
root@omni:/#

root@omni:/# vgcreate pgsql_vg /dev/mapper/mpatha
  Volume group "pgsql_vg" successfully created

root@omni:/# vgs
  VG        #PV #LV #SN Attr   VSize   VFree
  pgsql_vg    1   0   0 wz--n-  <2.00t  <2.00t
  ubuntu-vg   1   1   0 wz--n- 235.42g 135.42g
root@omni:/#

root@omni:/# lvcreate -n db1_lv -L 1500GB pgsql_vg
  Logical volume "db1_lv" created.
root@omni:/#

root@omni:/# mkfs.ext4 /dev/mapper/pgsql_vg-db1_lv
mke2fs 1.45.5 (07-Jan-2020)
Discarding device blocks: done
Creating filesystem with 393216000 4k blocks and 98304000 inodes
Filesystem UUID: a3a2c3c4-a0c2-4c07-af6c-43627e9b142f
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424, 20480000, 23887872, 71663616, 78675968,
        102400000, 214990848

Allocating group tables: done
Writing inode tables: done
Creating journal (262144 blocks): done
Writing superblocks and filesystem accounting information: done

root@omni:/# mkdir db1
root@omni:/# mount /dev/mapper/pgsql_vg-db1_lv /db1
root@omni:/# df -h /db1
/dev/mapper/pgsql_vg-db1_lv        1.5T   28K  1.4T   1% /db1

root@omni:/# vi /etc/fstab
root@omni:~# cat /etc/fstab
/dev/mapper/pgsql_vg-db1_lv     /db1    ext4    noatime,discard,errors=panic
0       1
root@omni:~#
```

Now the server is ready to install AlloyDB Omni database software.

Install AlloyDB Omni

Follow the detailed steps from https://cloud.google.com/alloydb/docs/omni/quickstart#install to install AlloyDB Omni.

The following is a log of the high level steps.

```
root@omni:~# alloydb database-server install --data-dir=/db1/alloydb_omni --pg-
port=5432
Initializing AlloyDB Omni instance...
Creating postgres user...
Copying installation files...
Creating AlloyDB Omni directories...
Installing systemd services...
Updating data plane configuration file...
Starting AlloyDB Omni...
Starting system check for AlloyDB Omni on local machine...

Checking for minimum of 2 vCPUs...SUCCESS
└─Detected 96 vCPUs.
Checking for minimum of 2.00GB of RAM...SUCCESS
└─Detected total RAM: 503.43GB.
Checking for Linux kernel...SUCCESS
└─Linux kernel detected.
Checking Linux kernel version is 4.18+...SUCCESS
└─Linux kernel version: 5.4.0-182-generic
Checking Linux distribution is Debian-based or RHEL...SUCCESS
└─Compatible Debian-based distribution detected.
Checking cgroups v2 is enabled...SUCCESS
└─cgroups v2 is enabled.
Checking Docker is installed...SUCCESS
└─Docker installation found.
Checking Docker daemon is running...SUCCESS
└─Docker service is currently active.
Checking Docker server version is 20.10+...SUCCESS
└─Compatible Docker server version: 24.0.5
Checking for conflicting pre-existing users...SUCCESS
└─User postgres exists with expected ID 2345.

System check passed. Starting AlloyDB Omni...
Configuring memory kernel parameters...
Configuring data directory...
Cleaning old pgtmp-loop and swap files...
Setting up core-dump directory...
Setting up swap file...
Starting memory cleanup...
Removing old data plane containers...
Writing dynamic PostgreSQL parameters...
Resolving data plane images...
Image found locally: gcr.io/alloydb-omni/pg-service:15.5.2
Image found locally: gcr.io/alloydb-omni/memory-agent:15.5.2
Starting AlloyDB Omni containers...
AlloyDB Omni database started.
root@omni:~#
```

Now connect to the database.

```
root@omni:/mnt# docker exec -it pg-service bash
postgres@omni:/$ psql -h localhost -U postgres
```

AlloyDB Omni is installed successfully.

Install the HammerDB client

Log in to the client node and follow the instructions to install AlloyDB client software and connect the database remotely.

```
root@client1:~# apt install -y postgresql-client
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libpq5 postgresql-client-12 postgresql-client-common
Suggested packages:
  postgresql-12 postgresql-doc-12
The following NEW packages will be installed:
  libpq5 postgresql-client postgresql-client-12 postgresql-client-common
0 upgraded, 4 newly installed, 0 to remove and 58 not upgraded.
Need to get 1,202 kB of archives.
After this operation, 4,477 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 libpq5 amd64 12.18-
0ubuntu0.20.04.1 [116 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 postgresql-client-
common all 214ubuntu0.1 [28.2 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 postgresql-client-
12 amd64 12.18-0ubuntu0.20.04.1 [1,054 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 postgresql-client
all 12+214ubuntu0.1 [3,940 B]
Fetched 1,202 kB in 0s (2,463 kB/s)
Selecting previously unselected package libpq5:amd64.
(Reading database ... 109244 files and directories currently installed.)
Preparing to unpack .../libpq5_12.18-0ubuntu0.20.04.1_amd64.deb ...
Unpacking libpq5:amd64 (12.18-0ubuntu0.20.04.1) ...
Selecting previously unselected package postgresql-client-common.
Preparing to unpack .../postgresql-client-common_214ubuntu0.1_all.deb ...
Unpacking postgresql-client-common (214ubuntu0.1) ...
Selecting previously unselected package postgresql-client-12.
Preparing to unpack .../postgresql-client-12_12.18-0ubuntu0.20.04.1_amd64.deb ...
Unpacking postgresql-client-12 (12.18-0ubuntu0.20.04.1) ...
Selecting previously unselected package postgresql-client.
Preparing to unpack .../postgresql-client_12+214ubuntu0.1_all.deb ...
Unpacking postgresql-client (12+214ubuntu0.1) ...
Setting up postgresql-client-common (214ubuntu0.1) ...
Setting up libpq5:amd64 (12.18-0ubuntu0.20.04.1) ...
Setting up postgresql-client-12 (12.18-0ubuntu0.20.04.1) ...
update-alternatives: using /usr/share/postgresql/12/man/man1/psql.1.gz to
provide /usr/share/man/man1/psql.1.gz (psql.1.gz) in auto mode
Setting up postgresql-client (12+214ubuntu0.1) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.15) ...
root@client1:~#
root@client1:~#
```

We can now download and install the latest version of HammerDB Client. HammerDB is used for data loading and running pre-defined sample OLTP and OLAP workloads. These workloads perform transactions into to the AlloyDB Omni database. Upon successful completion of workloads transaction summaries are available. A transaction summary is a baseline for measuring storage IO, latency, and throughput.

Complete the following steps to install HammerDB (4.6 version) client software.

```
#mkdir hammerdb
#pushd hammerdb
#curl -OL
https://github.com/TPC-Council/HammerDB/releases/download/v4.6/HammerDB-4.6-
Linux.tar.gz
#tar zxvf HammerDB-4.6-Linux.tar.gz
#cd HammerDB-4.6
#./hammerdbcli
 HammerDB CLI v4.6
 Copyright (C) 2003-2022 Steve Shaw
 Type "help" for a list of commands
 Initialized SQLite on-disk database /tmp/hammer.DB using existing tables    (45,056
KB)
 hammerdb> librarycheck
Checking database library for PostgreSQL
Success ... loaded library Pgtcl for PostgreSQL
```

A confirmation appears stating that HammerDB client installed successfully.

Now we need to load data into AlloyDB Omni database to conduct OLTP and OLAP tests.

OLTP (TPC-C) workload configuration

Create the `setup.env` file for environment variables to fetch.

Where,

PGHOST= IP address of the AlloyDB Omni database server.

PGPORT= Port number on which the AlloyDB Omni database is running/listening.

NUM_WAREHOUSE= Number of warehouses to load data. The database size depends on this. With 576 warehouses, the database size would be approximately 55 GB.

NUM_USERS= Number of concurrent users for benchmarking.

```
root@client1:/hammerdb/HammerDB-4.6# vi setup.env

# Private IP of the AlloyDB primary instance
export PGHOST=192.168.xxx.xxx
# Postgres default port address. You do not need to change it unless you use non-
default port
address.
export PGPORT=5432 # default port to connect with postgres
# Number of TPC-C warehouses to load. This determines the overall database size.
export NUM_WAREHOUSE=576
# Number of users for running the benchmark.
export NUM_USERS=256
EOF
root@client1:/hammerdb/HammerDB-4.6# ./setup.env
```

Now create the `build-tpcc.sh` file to construct the database.

```
root@client1:/hammerdb/HammerDB-4.6# vi build-tpcc.sh

#!/bin/bash -x
source ./setup.env
./hammerdbcli << EOF
# CONFIGURE PARAMETERS FOR TPCC BENCHMARK
# --------------------------------------
dbset db pg
dbset bm tpc-c
# CONFIGURE POSTGRES HOST AND PORT
# --------------------------------------
diset connection pg_host $PGHOST
diset connection pg_port $PGPORT
# CONFIGURE TPCC
# --------------------------------------
diset tpcc pg_superuser postgres
diset tpcc pg_user tpcc
diset tpcc pg_dbase tpcc
# SET NUMBER OF WAREHOUSES AND USERS TO MANAGE EACH WAREHOUSE
# THIS IMPORTANT METRIC ESTABLISHES THE DATABASE SCALE/SIZE
# --------------------------------------
diset tpcc pg_count_ware $NUM_WAREHOUSE
diset tpcc pg_num_vu 10
# LOG OUTPUT AND CONFIGURATION DETAILS
# --------------------------------------
vuset logtotemp 1
print dict
# CREATE AND POPULATE DATABASE SCHEMA
# --------------------------------------
buildschema
vudestroy
```

```
quit
EOF
```

To load data into the database, run `build-tpcc.sh` as shown.

```
root@client1:/hammerdb/HammerDB-4.6# chmod +x ./build-tpcc.sh
root@client1:/hammerdb/HammerDB-4.6# mkdir -p results
root@client1:/hammerdb/HammerDB-4.6# chmod 777 results
root@client1:/hammerdb/HammerDB-4.6# ulimit -n 32768
root@client1:/hammerdb/HammerDB-4.6# nohup ./build-tpcc.sh > results/build-tpcc.out
2>&1
```

Data loading will take about an hour depending on the size and number of users. After data loading is done, the following tests were run.

Create run-tpcc.sh, run in the background and monitor the log.

```
root@client1:/hammerdb/HammerDB-4.6# vi  run-tpcc.sh

#!/bin/bash -x
source ./setup.env
./hammerdbcli << EOF
dbset db pg
dbset bm tpc-c
# CONFIGURE PG HOST and PORT
# -------------------------
diset connection pg_host $PGHOST
diset connection pg_port $PGPORT
# CONFIGURE TPCC DB
# -------------------------
diset tpcc pg_superuser postgres
diset tpcc pg_user postgres
diset tpcc pg_dbase tpcc
# BENCHMARKING PARAMETERS
# -------------------------
diset tpcc pg_driver timed
diset tpcc pg_rampup 10
diset tpcc pg_duration 60
diset tpcc pg_vacuum false
diset tpcc pg_partition false
diset tpcc pg_allwarehouse true
diset tpcc pg_timeprofile true
diset tpcc pg_connect_pool false
diset tpcc pg_dritasnap false
diset tpcc pg_count_ware $NUM_WAREHOUSE
diset tpcc pg_num_vu 1
loadscript
print dict
vuset logtotemp 1
vuset vu $NUM_USERS
vucreate
```

```
vurun
quit
EOF

root@client1:/hammerdb/HammerDB-4.6# chmod +x run-tpcc.sh
root@client1:/hammerdb/HammerDB-4.6# mkdir -p results
root@client1:/hammerdb/HammerDB-4.6# ulimit -n 32768
root@client1:/hammerdb/HammerDB-4.6# sudo nohup ./run-tpcc.sh > results/run-tpcc.out
2>&1
```

Go to the results directory to review the monitor log.

```
root@client1:/hammerdb/HammerDB-4.6/results# tail -10f run-tpcc.out
Vuser 257:Initializing xtprof time profiler
Vuser 257:VU 257 : Assigning WID=256 based on VU count 256, Warehouses = 576 (1 out
of 3)
Vuser 257:VU 257 : Assigning WID=512 based on VU count 256, Warehouses = 576 (2 out
of 3)
Vuser 257:VU 257 : Assigning WID=192 based on VU count 256, Warehouses = 576 (3 out
of 3)
Vuser 257:Processing 10000000 transactions with output suppressed...
Vuser 1:Rampup 3 minutes complete ...
Vuser 1:Rampup 4 minutes complete ...
Vuser 1:Rampup 5 minutes complete ...
Vuser 1:Rampup 6 minutes complete ...
Vuser 1:Rampup 7 minutes complete ...
Vuser 1:Rampup 8 minutes complete ...
Vuser 1:Rampup 9 minutes complete ...
Vuser 1:Rampup 10 minutes complete ...
Vuser 1:Rampup complete, Taking start Transaction Count.
Vuser 1:Timing test period of 60 in minutes
Vuser 1:1 ...,
Vuser 1:2 ...,
```

OLAP (TPC-H) workload configuration

Create the `setup.env` file for environment variables to fetch.

Where,

PGHOST= IP address of the AlloyDB Omni database server.

PGPORT= Port number on which the AlloyDB Omni database running/listening.

TPCH_SCALE= Defines database size to build. With a 10 scale factor, a 20 GB database will be created.

```
setup.env

# Private IP of the AlloyDB primary instance
export PGHOST=192.168.xx.xx
# Postgres default port address. You do not need to change it unless you use non-
default port
```

```
address.
export PGPORT=5432 # default port to connect with postgres
# TPC-H Scale Factor (determines the size of the database that we want to build).
export TPCH_SCALE=10
EOF
```

Now create the `build-tpch.sh` file to construct the database.

```
vi build-tpch.sh

#!/bin/bash -x
source ./setup.env
./hammerdbcli << EOF
# CONFIGURE PARAMETERS FOR TPC-H BENCHMARK
# --------------------------------------
dbset db pg
dbset bm tpc-h
# CONFIGURE POSTGRES HOST AND PORT
# --------------------------------------
diset connection pg_host $PGHOST
diset connection pg_port $PGPORT
# CONFIGURE TPC-H
# --------------------------------------
diset tpch pg_tpch_superuser postgres
diset tpch pg_tpch_user postgres
diset tpch pg_tpch_dbase tpch
# --------------------------------------
diset tpch pg_scale_fact $TPCH_SCALE
diset tpch pg_num_tpch_threads 32
diset tpch pg_refresh_on false
diset tpch pg_refresh_verbose false
diset tpch pg_degree_of_parallel 8
vuset vu 1
# logging
vuset logtotemp 1
vuset timestamps 0
vuset unique 0
# load and run benchmarking script
loadscript
buildschema
# terminate when completed
vudestroy
quit
EOF
```

To load data into the database, execute `build-tpch.sh` as shown.

```
root@client1:/hammerdb/HammerDB-4.6# chmod +x ./build-tpch.sh
root@client1:/hammerdb/HammerDB-4.6# mkdir -p results
root@client1:/hammerdb/HammerDB-4.6# chmod 777 results
```

```
root@client1:/hammerdb/HammerDB-4.6# ulimit -n 32768
root@client1:/hammerdb/HammerDB-4.6# sudo nohup ./build-tpch.sh > results/build-
tpch.out 2>&1
```

Data loading will take a considerable amount of time depending on the scale factor and number of users loading data. After data loading is done, run the following tests.

Create run-tpch.sh, and then run in the background and monitor the log.

```
root@client1:/hammerdb/HammerDB-4.6# vi run-tpch.sh

#!/bin/bash -x
source ./setup.env
./hammerdbcli << EOF
# CONFIGURE PARAMETERS FOR TPC-H BENCHMARK
# --------------------------------------
dbset db pg
dbset bm tpc-h
# CONFIGURE POSTGRES HOST AND PORT
# --------------------------------------
diset connection pg_host $PGHOST
diset connection pg_port $PGPORT
# CONFIGURE TPC-H
# --------------------------------------
diset tpch pg_tpch_superuser postgres
diset tpch pg_tpch_user postgres
diset tpch pg_tpch_dbase tpch
diset tpch pg_scale_fact $TPCH_SCALE
diset tpch pg_num_tpch_threads 1
diset tpch pg_refresh_on false
diset tpch pg_refresh_verbose false
diset tpch pg_degree_of_parallel 8
diset tpch pg_trickle_refresh 1000
diset tpch pg_tpch_tspace pg_default
diset tpch pg_tpch_gpcompat false
diset tpch pg_tpch_gpcompress false
diset tpch pg_cloud_query false
diset tpch pg_rs_compat false
diset tpch pg_update_sets 1
diset tpch pg_total_querysets 1
vuset vu 1
# logging
vuset logtotemp 1
vuset timestamps 0
vuset unique 0
# load tpc-h script and run benchmark
loadscript
# Warmup run
vurun
# Measurement run
vurun
```

```
# terminate when completed
waittocomplete
vudestroy
quit
EOF

root@client1:/hammerdb/HammerDB-4.6# chmod +x run-tpch.sh
root@client1:/hammerdb/HammerDB-4.6# mkdir -p results
root@client1:/hammerdb/HammerDB-4.6# ulimit -n 32768
root@client1:/hammerdb/HammerDB-4.6# sudo nohup ./run-tpch.sh > results/run-tpch.out
2>&1
```