NVIDIA DGX OS 6 User Guide

NVIDIA

Apr 23, 2024

Contents

Chapter 1. About DGX OS 6

NVIDIA DGX OS provides a customized installation of Ubuntu Linux with system-specific optimizations and configurations, additional drivers, and diagnostic and monitoring tools. It provides a stable, fully-tested, and supported OS to run AI, machine learning, and analytics applications on DGX Supercomputers.

NVIDIA DGX systems are shipped preinstalled with DGX OS to provide a turnkey solution for running AI and analytics workloads. Initial system configuration is deferred to a setup wizard that runs after the first boot. The setup wizard offers users a fast on-boarding experience for using DGX systems.

The DGX OS installer is released as an ISO image to reimage a DGX system. The additional software, the NVIDIA DGX Software Stack, that is included in DGX OS is provided as packages that are available from software repositories over the internet.

You also have the option to install the NVIDIA DGX Software Stack on a regular Ubuntu 22.04 while still benefiting from the advanced DGX features. This installation method supports more flexibility, such as custom partition schemes. Cluster deployments also benefit from this installation method by taking advantage of Ubuntu's standardized automated and non-interactive installation process.

1.1. DGX OS 6 Features

The following are the key features of DGX OS Release 6:

- Based on Ubuntu 22.04 with the latest long-term Linux kernel version 5.15 for the recent hardware and security updates and updates to software packages, such as Python and GCC.
- Includes the NVIDIA-optimized Linux kernel, which supports GPU Direct Storage (GDS) without additional patches.
- Provides access to all NVIDIA GPU driver branches and CUDA toolkit versions.
- Uses the Ubuntu OFED by default with the option to install NVIDIA OFED for additional features.
- Supports Secure Boot (requires Ubuntu OFED).
- Supports DGX H100.

1.2. Installation and Upgrade

This document covers installation and upgrade options for DGX OS. It also provides instructions for setting up the system and installing additional software.

Initial Setup If DGX OS 6 is already installed on your DGX system, refer to *Initial Setup* for information about setting up the system on first boot.

After initial setup, refer to *Upgrading the OS* to perform a *package upgrade* to the latest software package versions.

Upgrading the OS To upgrade your DGX OS to the latest software package versions or for information about performing a *release upgrade* from DGX OS 5 to DGX OS 6, refer to *Upgrading the OS*.

Reimaging the System To restore a DGX system to a default DGX OS installation and erase all data, you can use the ISO image that includes an autonomous installer. Refer to *Reimaging the System* for more information.

Installing DGX Software on Ubuntu To install Ubuntu and the DGX Software Stack, refer to *Installing DGX Software on Ubuntu* for information about automating the installation process, such as a cluster deployment.

Managing and Upgrading Software Components DGX OS and Ubuntu provide additional software packages, including additional NVIDIA software and driver options. Refer to *Managing and Upgrading Software* for more information and installation instructions.

Important: Before you upgrade or install any new software, always consult the *Release Notes* for the latest information about available upgrades. You can find out more about the release cadence and release methods for DGX OS in *Release Guidance*

1.3. Related Documentation

Refer to the following documents that are related to DGX OS 6:

- DGX Documentation

  All documentation for DGX products, including product user guides, software release notes, and firmware update container information

- MIG User Guide

  The Multi-Instance GPU (MIG) feature allows the NVIDIA A100 GPU to be securely partitioned into up to seven discrete GPU instances.

- NGC Private Registry

  How to access the NGC container registry for using containerized deep learning GPU.

- NVSM Software User Guide

  Contains instructions for using the NVIDIA System Manager software.

- DCGM Software User Guide

  Contains instructions for using the Data Center GPU Manager software.

1.4. NVIDIA Enterprise Support

NVIDIA Enterprise Support is the support resource for DGX customers and can assist with hardware, software, or NGC application issues. For more information about how to obtain support, visit NVIDIA Enterprise Support.

Chapter 2. Release Guidance

This information helps you understand the DGX OS release mechanism, release numbering convention, and options to install and upgrade your DGX OS software.

Important: Keeping your DGX OS software up to date is the single most important task for protecting your system. Security-related updates are available from the Ubuntu and NVIDIA repositories. Refer to *Upgrading the OS* for information about performing a *package upgrade* to latest software releases and the Upgrades page on Ubuntu's Wiki for additional information.

Security updates for Ubuntu are announced on Ubuntu Security Notices (USN). The USN website lists known Common Vulnerabilities and Exposures (CVEs) for Ubuntu packages. You can find NVIDIA security announcements on the Product Security website.

2.1. Release Mechanisms

This section provides information about the DGX OS release mechanisms that are available to install or upgrade DGX systems to the latest version of the DGX OS.

2.1.1. DGX OS ISO Image

DGX OS is released in the form of an ISO image that includes an autonomous installer and all the software required for running AI, machine learning, and analytics applications. Basic system configuration is deferred to a setup wizard on first boot. The ISO is intended for cases that require reimaging a single system.

Updated versions of the ISO image are available between releases with the following:

- Critical bug fixes and security mitigations.
- Improved installation experience.
- Support for new DGX systems and hardware components.

Always use the latest ISO image unless you need to restore the system to an earlier version.

2.1.2. Linux Software Repositories

The DGX OS software, upgrades to DGX OS, and a large pool of additional software are available from software repositories in the form of software packages. Software repositories are storage locations from which your system retrieves the software packages. The repositories used by DGX OS are hosted by Canonical for the Ubuntu OS and NVIDIA for DGX specific software and other NVIDIA software.

Software packages contain the files of the respective software component or closely related components. This includes executables, configuration files, documentation, and metadata, such as dependencies.

New versions of these packages contain bug fixes and security updates. These packages provide an upgrade to a DGX OS release. The repositories are also updated between releases for critical bug fixes, security migitations, and additional hardware enablement. Updates for supporting a new system or a new hardware component, such as a network card or disk drive, do not affect existing hardware configurations.

System upgrades are cumulative. Your system always receives the latest package versions, which can be newer than the current DGX OS release. You cannot select which upgrades to make or limit upgrades to a specific DGX OS version by default.

Warning: NVIDIA recommends that you do not update packages individually. For example, do not run `apt install <package-name>`.

2.2. Release Numbering Convention

The NVIDIA DGX OS release numbering convention is <major>.<minor>. The meaning for major and minor are as follows:

Major Release Major releases are typically based on Ubuntu releases. These releases include new kernel versions and new features that might not be backwards compatible. They also include the latest releases of the NVIDIA software.

For example:

- DGX OS 5 is based on Ubuntu 20.04.
- DGX OS 6 is based on Ubuntu 22.04.

Minor Release Minor releases include new NVIDIA features, bug fixes, and security updates.

- Minor releases deliver upgrades to the NVIDIA software that are backward compatible.
- When you upgrade a system, the system remains on the currently installed NVIDIA GPU driver release unless the driver branch reaches the end of support.
- The ISO releases provide a snapshot of all accumulated changes. The release can include a newer NVIDIA GPU driver LTS branch.

Important bug fixes and security updates are provided between releases through the software repositories without having a release number assigned to the change.

2.3. Release Cadence

DGX OS is released twice a year, typically around February and August, for the first two years as minor releases. Updates are provided between releases and after the initial two years for the remainder of the supported duration.

Chapter 3. Release Notes

Note: Software upgrades are cumulative, which means that your systems will always receive the latest versions of all installed software components. The packages in the repositories can also be newer than the current DGX OS release. Read and evaluate the information and advisories from all relevant releases and later upgrades.

3.1. Current Software Versions

The following table shows the current version information of the software packages provided in the NVIDIA and Ubuntu repositories for the NVIDIA DGX Software Stack.

Table 1: Current Software Versions (Last Updated on April 19, 2024)

Component	Version	Additional Information
GPU Driver	550.54.15	RHEL8: RPM installer RHEL9: RPM installer
GPU Driver	535.161.08	RHEL8: RPM installer RHEL9: RPM installer
GPU Driver	470.239.06	For RHEL8 and DGX OS 5 only. RHEL8: RPM installer
CUDA Toolkit	12.4	R550: 12.4 download
CUDA Toolkit	12.2 Update 2	R535: 12.2 update 2 download
CUDA Toolkit	11.4	R470: For RHEL8 and DGX OS 5 only.
MLNX_OFED	23.10-2.1.3.1	23.10-2.1.3.1 download
Inbox OFED	39.0-1	For DGX OS 6 only.
NCCL	2.21.5	
cuDNN	9.1.0	
DCGM	3.3.5	
GPUDirect Storage (GDS)	▶ 1.9 for CUDA Toolkit 12.4 ▶ 1.7.2 for CUDA Toolkit 12.2 Update 2 ▶ 1.0 for CUDA Toolkit 11.4	
NVIDIA Container Toolkit	1.14.6	NVIDIA Container Toolkit includes the following packages: ▶ libnvidia-container-tools: 1.14.6 ▶ libnvidia-container1: 1.14.6 ▶ nvidia-container-toolkit: 1.14.6
nvidia-peer-memory	1.3	

Note:

- CUDA Toolkit is installed by default only for DGX stations and is optional for DGX servers. Refer to the CUDA Release Notes for driver compatibility information.
- For CUDA Toolkit minor version compatibility and the minimum required driver version, refer to CUDA Compatibility.

The following table provides information about the supported OS and matching firmware versions for NVIDIA® OpenFabrics Enterprise Distribution for Linux (MLNX_OFED) version 23.10-2.1.3.1. For more information about this long-term support (LTS) release, refer to

- NVIDIA MLNX_OFED Documentation v23.10-2.1.3.1 LTS
- Software download: version 23.10-2.1.3.1

Table 2: Supported OS and Matching Firmware Versions (Last Updated on April 19, 2024)

OS	**DGX-1, DGX-2 ConnectX-4 (CX-4) or ConnectX-5 (CX-5)**	**DGX A100 ConnectX-6**	**DGX A100 ConnectX-7**	**DGX H100 ConnectX-7**
Ubuntu 22.04 DGX OS 6	CX-5: 16.35.3006 CX-4: 12.28.2006	20.39.3004	28.39.3004	28.39.3004
Ubuntu 20.04 DGX OS 5	CX-5: 16.35.3006 CX-4: 12.28.2006	20.39.3004	28.39.3004	Not applicable
RHEL 9 and RHEL 8	CX-5: 16.35.3006 CX-4: 12.28.2006	20.39.3004	28.39.3004	28.39.3004
COSS 9	Not applicable	20.39.3004	Not applicable	Not applicable

For installation instructions, refer to

- NVIDIA MLNX_OFED: *Installing the Mellanox OFED Drivers*
- ConnectX®-7 adapter cards: *Installing the ConnectX-7 Firmware*
- ConnectX®-6 adapter cards: Firmware Downloads

In addition to upgrading to the versions described in this section, performing a *package upgrade* can update the software component versions, the Ubuntu 22.04 LTS version, and Ubuntu kernel, depending on when you perform the upgrade.

Note: For information about LTS software versions for related networking components, refer to the Networking Long-Term Support Releases page.

Important: This release incorporates the following updates:

- Ubuntu ConnectX drivers and OFED stack.
- Customers are advised to consider these updates and any effect they might have on their application. For example, some MOFED-dependent applications can be affected.
- Best practices support upgrading select systems and verifying that your applications are working as expected before you deploy on additional systems.

3.2. Supported DGX Systems

DGX OS 6 supports the following systems:

- DGX H100 (requires DGX OS ISO 6.0.11 or later)
- DGX H800 (requires DGX OS ISO 6.1.0 or later)
- DGX A100 640 GB
- DGX A100 320 GB
- DGX A800 640 GB
- DGX Station A100 320 GB
- DGX Station A100 160 GB
- DGX Station A800 320 GB
- DGX-2
- DGX-1 (V100)
- DGX Station (V100)

3.3. Update History

This section provides information about important updates to DGX OS 6.

3.3.1. DGX OS 6.2.0 Release: March 22, 2024

Here are the new features in DGX OS 6.2.0:

- The *DGX OS ISO 6.2.0* has been released.
- Added support for single-port ConnectX-7 VPI adapter card for DGX A100 System.
- Support for MLNX_OFED LTS version 23.10-2.1.3.1.
- Continued support for DGX H100 and DGX H800.
- The following changes were made to the repositories and the ISO:
  - OS base: 22.04.3 LTS
  - Kernel: 5.15.0-1046-nvidia
  - NVIDIA GPU Driver: 535.161.07
  - CUDA Toolkit: 12.2-1
  - NCCL: 2.20.3
  - cuDNN: 8.9.7
  - DCGM: 3.3.5
  - GPUDirect Storage: 1.7.2
  - NVSM: 23.12.01
  - Docker Engine: 24.0.7-1
  - NVIDIA Container Toolkit: 1.14.6
  - MIG Configuration Tool: 0.5.5
  - NGC CLI: 3.36.0
  - DLFW: 24.01
  - GDRCopy: 2.4.1

3.3.2. DGX OS 6.1.0 Release: August 11, 2023

- The *DGX OS ISO 6.1.0* has been released.
- The following changes were made to the repositories and the ISO
  - Added support for DGX H800.
  - Kernel update to 5.15.0-1029-nvidia.
  - The gdrcopy packages are removed from ISO but are still available from the repository.
  - The gdrcopy packages can be installed as an option but are no longer installed by default.
  - The dgx-h100-ota-update-meta package is updated to 23.07.1.
  - The nvidia-manage-ofed package is updated to 23.07-1.
  - NVSM is updated to 23.06.2.
  - NVIDIA CUDA Toolkit is updated to 12.2.0.

- NVIDIA Container Toolkit is updated to 1.13.5.

3.3.3. DGX OS 6.0.11 Release: May 17, 2023

- Added support for DGX H100
- The *DGX OS ISO 6.0.11* has been released.
- The following changes were made to the repositories and the ISO
- Kernel version updated to 5.15.0-1025.25
- NVSM updated to 22.12.06

3.3.4. DGX OS 6.0.10 Release: May 3, 2023

- The *DGX OS ISO 6.0.10* has been released.

3.4. DGX OS ISO Releases

This section lists all DGX OS ISO releases with the software versions included in the image.

3.4.1. DGX OS ISO 6.2.0

Component	Version	Notes
Ubuntu	Ubuntu 22.04.3 LTS **Base OS 6.2.0**	
Ubuntu Kernel	5.15.0-1046-nvidia	
GPU Driver	535.161.07	
CUDA Toolkit	CUDA Toolkit 12.2.1	CUDA Toolkit is only installed by default for DGX stations and is optional for DGX servers. Refer to the CUDA Release Notes for driver compatibility information.
Inbox OFED	39.0-1	
NCCL	2.20.3	
cuDNN	8.9.7	
DCGM	3.3.5	
GPUDirect Storage (GDS)	1.7.2	
NVIDIA Container Toolkit	1.14.6	NVIDIA Container Toolkit includes the following packages: ▶ libnvidia-container-tools: 1.14.6 ▶ libnvidia-container1: 1.14.6 ▶ nvidia-container-toolkit: 1.14.6
NVSM	23.12.01	
Docker Engine	24.0.7-1	
GDRCopy	2.4.1	
MIG Configuration Tool	0.5.5	
NGC CLI	3.36.0	
DLFW (BM)	24.01	
ISO	DGXOS-6.2.0-2024-03-15-14-25-26.iso	
MD5 Checksum	`ecd6c5f77d957d41c8132a0d3941cf8c`	

3.4.2. DGX OS ISO 6.1.0

Component	Version	Additional Information
Ubuntu	22.04 LTS	
Optimized Kernel	5.15.0-1029-nvidia	
GPU Driver	R535: 535.54.03	
CUDA Toolkit	12.2.0	**Note:** The CUDA Toolkit is only installed for DGX Stations and option for DGX servers. Refer also to the latest CUDA Release Notes for driver compatibility information.
NCCL	2.18.3	
cuDNN	8.9.1	
DCGM	3.1.8	
Inbox OFED	39.0-1	
GPUDirect Storage (GDS)	2.15.1	
NVSM	23.06.02	
Docker Engine	23.0.4	Refer to Docker Engine
NVIDIA Container Toolkit	1.13.5	NVIDIA Container Toolkit includes the following packages: ▶ libnvidia-container-tools: 1.13.5-1 ▶ libnvidia-container1: 1.13.5-1 ▶ nvidia-container-toolkit: 1.13.5-1
MIG Configuration Tool	0.5.1	Refer to the following NVIDIA mig-parted github pages and systemd
NGC CLI	3.17.0	Refer to the NGC CLI Documentation.
DLFW (BM)	23.06	
ISO	DGXOS-6.1.0-2023-08-09-12-30-10.iso	
MD5 Checksum	`d38620ffa58905330c1efe49b3d7ff53`	

3.4.3. DGX OS ISO 6.0.11

Component	Version	Additional Information
Ubuntu	22.04 LTS	
Optimized Kernel	5.15.0-1025.25	
GPU Driver	R525: 525.105.17	
CUDA Toolkit	12.0.1	**Note:** The CUDA Toolkit is only installed for DGX Stations and option for DGX servers. Refer also to the latest CUDA Release Notes for driver compatibility information.
NCCL	2.17.1	
cuDNN	8.9.0.131	
DCGM	3.1.8	
Inbox OFED	39.0-1	
GPUDirect Storage (GDS)	2.15.1	
NVSM	22.12.06	
Docker Engine	23.0.4	Refer to Docker Engine
NVIDIA Container Toolkit	1.13.1	NVIDIA Container Toolkit includes the following packages: ▶ libnvidia-container-tools: 1.13.1-1 ▶ libnvidia-container1: 1.13.1-1 ▶ nvidia-container-toolkit: 1.13.1-1 ▶ nvidia-docker2: 2.11.0
MIG Configuration Tool	0.5.1	Refer to the following NVIDIA mig-parted github pages and systemd
NGC CLI	3.17.0	Refer to the NGC CLI Documentation.
DLFW (BM)	23.03	
ISO	DGXOS-6.0.11-2023-05-16-16-18-31.iso	
MD5 Checksum	`21d73f97b1e8d3efc15eddabb53c4f17`	

3.4.4. DGX OS ISO 6.0.10

Component	Version	Additional Information
Ubuntu	22.04 LTS	
Optimized Kernel	5.15.0-1023.23	
GPU Driver	R525: 525.105.17	
CUDA Toolkit	12.0.1	**Note:** The CUDA Toolkit is only installed for DGX Stations and option for DGX servers. Refer also to the latest CUDA Release Notes for driver compatibility information.
NCCL	2.17.1	
cuDNN	8.9.0.131	
DCGM	3.1.8	
Inbox OFED	39.0-1	
GPUDirect Storage (GDS)	2.15.1	
NVSM	22.12.04	
Docker Engine	23.0.4	Refer to Docker Engine
NVIDIA Container Toolkit	1.13.1	NVIDIA Container Toolkit includes the following packages: ▶ libnvidia-container-tools: 1.13.1-1 ▶ libnvidia-container1: 1.13.1-1 ▶ nvidia-container-toolkit: 1.13.1-1 ▶ nvidia-docker2: 2.11.0
MIG Configuration Tool	0.5.1	Refer to the following NVIDIA mig-parted github pages and systemd
NGC CLI	3.17.0	Refer to the NGC CLI Documentation.
DLFW (BM)	23.03	
ISO	DGXOS-6.0.10-2023-05-02-19-06-32.iso	
MD5 Checksum	`55ae2430a4ca490e0383f3740a39941e`	

Chapter 4. Initial Setup

This section describes the set up process when the DGX system is powered on for the first time after delivery or after the system is reimaged.

To start the process, you need to accept the End User License Agreements (EULA) and to set up your username and password. To preview the EULA, visit NVIDIA DGX Systems Enterprise Support and click the **DGX EULA** link (under Details).

4.1. Connecting to the DGX System

During the installation and initial configuration steps, you need to connect to the console of the DGX system. There are several ways to connect to the DGX system, including the following:

- Through a virtual keyboard, video, and mouse (KVM) in the BMC.
- A direct connection with a local monitor and keyboard.

Refer to the appropriate DGX product user guide for a list of supported connection methods and specific product instructions:

- DGX H100 System User Guide
- DGX A100 System User Guide
- DGX-2 System User Guide
- DGX-1 User Guide
- DGX Station User Guide
- DGX Station A100 User Guide

4.2. First Boot Setup Wizard

Here are the steps to complete the first boot process. It differs between the DGX systems:

- *First Boot Process for DGX Servers*
- *First Boot Process for DGX Station*

4.2.1. First Boot Process for DGX Servers

Here are the steps to complete the first boot process for DGX servers.

1. If the DGX OS was installed with an encrypted root filesystem, you will be prompted to unlock the drive. See *Advanced Installation Options (Encrypted Root)* When you select this menu item, you have the ability to encrypt the root filesystem of the DGX system for more information.

   Enter `nvidia3d` at the crypt prompt.
2. Accept the EULA to proceed with the DGX system set up.
3. Complete the following steps:
   a. Select your language and locale preferences.
   b. Select the country for your keyboard.
   c. Select your time zone.
   d. Confirm the UTC clock setting.
   e. Create an administrative user account with your name, username, and password.
      - This username is also used as the BMC and GRUB username.
      - The BMC software **will not** accept `sysadmin` for a username, and you will not be able to log in to the BMC with that username.

        > **Warning:** During this step in the procedure, the default BMC Admin User will be disabled and a new BMC Admin user will be created.

      - The username must be composed of lower-case letters.
      - The username will be used for administrative activities instead of the root account.
      - Ensure you enter a strong password.

      If the password that you entered is weak, a warning appears.
   f. Create a BMC admin password. The allowed character length for the BMC password depends on the specific DGX product:
      - DGX-1: 1 - 20 characters
      - DGX-2: 1 - 20 characters
      - DGX A100: 13 - 20 characters
      - DGX Station A100: 13 - 20 characters
      - DGX H100: 13 - 20 characters

      After you create your login credentials, the default credentials will no longer work.
   g. Create a GRUB password.
      - Your GRUB password must have at least 8 characters. If it has less than 8 characters, you cannot click *Continue*.
      - If you continue without entering a password, the GRUB protection will be disabled. For added security, NVIDIA recommends that you set the GRUB password.

h. Create a root filesystem passphrase. This dialog only appears if root filesystem encryption was selected at the time of the DGX OS installation. See *Advanced Installation Options (Encrypted Root)* for more information. When you select this menu item, you have the ability to encrypt the root filesystem of the DGX system.

i. Select a primary network interface for the DGX system. This should typically be the interface that you will use for subsequent system configuration or in-band management. For example:

   - DGX-1: `enp1s0f0`
   - DGX-2: `enp6s0`
   - DGX A100: `enp226s0`
   - DGX H100: `ens6sf0`

   Do not select `enp37s0f3u1u3c2`, `bmc_redfish0`, or similar name, as this interface is intended only for out-of-band management or future support of in-band tools that will access the Redfish APIs.

   After you select the primary network interface, the system attempts to configure the interface for DHCP and prompts you to enter the name server addresses.

   - If no DHCP is available, click *OK* at the Network autoconfiguration failed dialog and manually configure the network.
   - To configure a static address, then click *Cancel* at the dialog after the DHCP configuration completes to restart the network configuration steps.
   - To select a different network interface, after the DHCP configuration completes, click *Cancel* at the dialog to restart the network configuration steps.

j. If prompted, enter the requested networking information, such as the name server or the domain name.

k. Select a host name for the DGX system.

After you complete the first boot process, the DGX system configures the operating system, starts the system services, and displays a login prompt on the console. If the IP of the configured network interface is known, you can log in by using the console or secure shell (SSH).

Caution: Before issuing a reboot, make sure that the NVIDIA RAID configuration service has completed by issuing `sudo systemctl status nvidia-raid-config` until the `Finished NVIDIA RAID Configuration` message appears. It typically takes about 10 minutes for the service to complete.

4.2.2. First Boot Process for DGX Station

Important: You can connect to the system through a remote BMC console using virtual keyboard, video, and mouse. To use the remote BMC console, ensure that the OnBrd/Ext VGA Select SBIOS configuration is set to OnBoard. If the SBIOS is not configured this way, the system starts the user interface on the display port instead of the remote BMC console. Refer to Using DGX Station A100 as a Server Without a Monitor for details on changing the SBIOS configuration.

When you power on your DGX Station for the first time, you are prompted to accept end user license agreements for NVIDIA software. You are then guided through the process to complete the initial Ubuntu OS configuration.

During the configuration process, to prevent unauthorized users from using non-default boot entries and modifying boot parameters, you need to enter a GRUB password.

1. Accept the EULA and click *Continue*.
2. Select your language, for example, *English – English*, and click *Continue*.
3. Select your keyboard, for example, *English (US)* and click *Continue*.
4. Select your location, for example, *Los Angeles*, and click *Continue*.
5. Enter your username and password, enter the password again to confirm it, and click *Continue*.

   Here are some requirements to remember:

   - The username must be composed of lower-case letters.
   - The username will be used instead of the root account for administrative activities.
   - It is also used as the GRUB username.
   - Ensure you enter a strong password.

   If the password that you entered is weak, a warning appears.
6. Enter the GRUB password and click *OK*.
   - Your GRUB password must have at least 8 characters.

     If it has less than 8 characters, you cannot click *Continue*.
   - If you do not enter a password, GRUB password protection will be disabled.
7. If you performed the automated encryption install, you will also be prompted to create a new passphrase for your root filesystem.
   - The default password was seeded with `nvidia3d` which will be disabled after you complete this step.
   - This new passphrase will be used to unlock your root filesystem when the system boots.

Caution: Following a reboot, make sure that the NVIDIA RAID configuration service has completed by issuing `sudo systemctl status nvidia-raid-config` until the "Finished NVIDIA RAID Configuration" message appears. NOTE: It typically takes about 10 minutes for the service to complete.

4.3. Post-Installation Tasks

You can complete the following tasks after you install your DGX system.

4.3.1. Adding Support for Additional Languages to the DGX Station

During the initial Ubuntu OS configuration, you are prompted to select the default language on the *DGX Station*. If the language that you select is in the DGX OS 6 software image, it is installed in addition to English, and you will see that language after you log in to access your desktop. If the language that you select is not included, you will still see English after logging in, and you will need to install the language separately.

The following languages are included in the DGX OS 6 software image:

- English
- Chinese (Simplified)
- French
- German
- Italian
- Portuguese
- Russian
- Spanish

For information about how to install languages, see Install languages

4.3.2. Configuring your DGX Station

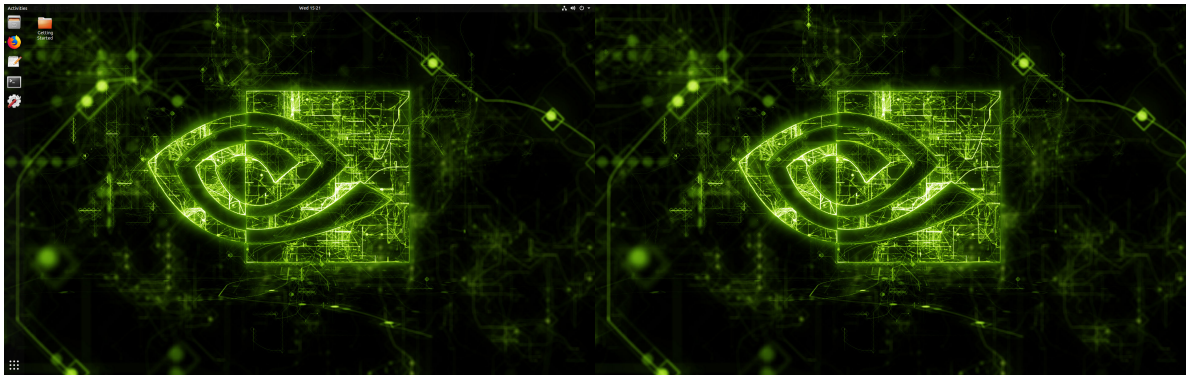
The DGX Display Adaptor card provides DGX OS with multiple display outputs, which allow you to connect multiple monitors to the DGX Station A100. If you plan to use more than one display, configure the DGX Station A100 to use multiple displays **after** you complete the initial DGX OS configuration. See *First Boot Process for DGX Station*.

When you power on your DGX Station for the first time, you are prompted to accept end user license agreements for NVIDIA software. You are then guided through the process to complete the initial Ubuntu OS configuration.

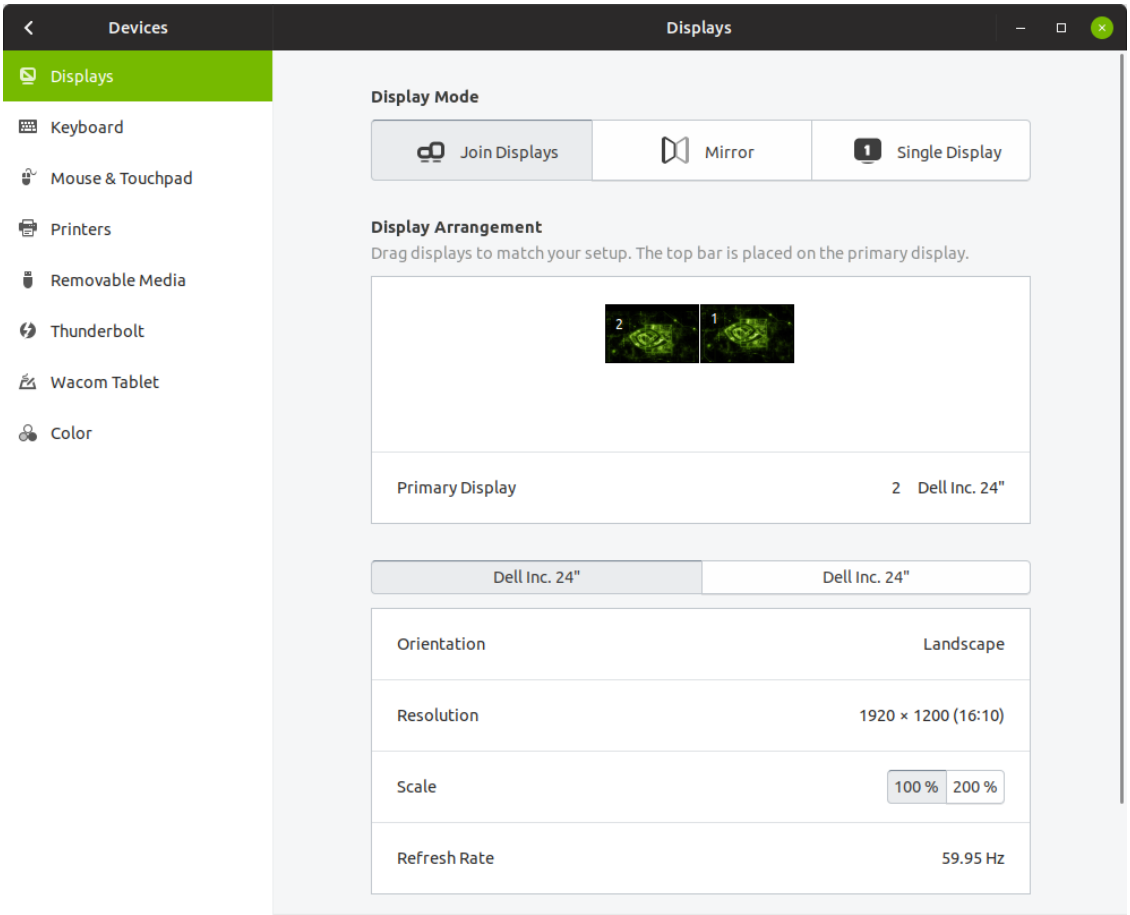
1. Connect the displays that you want to use to the mini DisplayPort (DP) connectors (or the DisplayPort connectors DGX Station V100) at the back of the unit.

   Note: DGX Station A100 also supplies two mini DP to DP adapters if your monitors do not natively support mini DP input.

   Each display is automatically detected as you connect it.



2. **Optional:** If necessary, adjust the display configuration, such as switching the primary display, or changing monitor positions or orientation.
   a. Open the Displays window.
   b. In the Displays window, update the necessary display settings and click Apply



4.3.3. Configuring your DGX Station V100

This information **only** applies to DGX Station V100.

High-resolution displays consume a large quantity of GPU memory. If you connected three 4K displays to the DGX Station V100, the displays might consume most of the GPU memory on the NVIDIA Tesla V100 GPU card to which these displays are connected, especially if you are running graphics-intensive applications.

If you are running memory-intensive compute workloads on the DGX Station V100, and are experiencing performance issues, consider conserving GPU memory by reducing or minimizing the graphics workload.

- To reduce the graphics workload, disconnect any additional displays you connected and use only one display with the DGX Station V100.
- If you disconnect a display from the DGX Station V100, the disconnection is automatically detected, and the display settings are automatically adjusted for the remaining displays.
- To minimize the graphics workload, shut down the display manager and use secure shell (SSH) to remotely log in to the *DGX Station*.
- In DGX OS 6.x, log in to the *DGX Station* remotely, run the following commands:
  - To start the GNOME Display Manager (GDM3):

    ```
    sudo systemctl start gdm3
    ```

  - To stop the GDM3, run the following command:

    ```
    sudo systemctl stop gdm3
    ```

4.3.4. Enabling Multiple Users to Remotely Access the DGX System

To enable multiple users to remotely access the DGX system, an SSH server is installed and enabled on the DGX system.

Add other Ubuntu OS users to the DGX system to allow them to remotely log in to the DGX system through SSH. Refer to Add a new user account for more information.

For information about how to log in remotely through SSH, see Connecting to an OpenSSH Server on the Ubuntu Community Help Wiki.

Important: The DGX system does **not** provide any additional isolation guarantees between users beyond the guarantees that the Ubuntu OS offers. For guidelines about how to secure access to the DGX system over SSH, see Configuring an OpenSSH Server on the Ubuntu Community Help Wiki.

Chapter 5. Reimaging the System

This section provides information about installing the DGX OS by reimaging the system from the DGX OS ISO image.

DGX OS is already preinstalled on new DGX systems and only requires reimaging in limited cases. If your system is already running DGX OS 6, you can skip to *Initial Setup* for instructions about the initial setup of the system. To upgrade a system from DGX OS 5, refer to *Upgrading the OS*.

You also have the option to install Ubuntu and the DGX software manually, for example, if you require custom installation options, such as a specific a drive partition scheme. Refer to *Installing DGX Software on Ubuntu* for more details. It also describes automating the installation process, for example, for cluster deployments.

There are situations where you want to reimage a system, such as the following:

- You want to install the latest version on a new system.
- You need to install an older version.
- The OS becomes corrupted.
- The OS drive is replaced or both drives in a RAID-1 configuration are replaced.
- You want to encrypt the root filesystem.
- You want to revert the DGX system to the originally installed DGX OS.

Warning: Reimaging the system erases all data stored on the OS drives. This includes the `/home` partition, where all users' documents, software settings, and other personal files are stored. If you need to preserve data through the reimaging, you can move the files and documents to the `/raid` directory and install the DGX OS software with the option to preserve the RAID array content.

The reimage process does not change persistent hardware configurations such as MIG settings or data drive encryption.

Important: After completing the installation, refer to *Upgrading the OS* to perform a *package upgrade* to the latest available software versions available since the DGX OS ISO release, including security updates.

5.1. Obtaining the DGX OS ISO Image

Note: Before you begin, ensure that you have an active NVIDIA Enterprise Support account.

To ensure that you install the latest available version of DGX OS, obtain the latest ISO image file from **NVIDIA Enterprise Support**:

1. Go to the Download Center.
2. Click **[Server/Workstation]** -> **[DGX]**, and select **All Downloads** for your system.
3. Click on the download link for the latest ISO release to go to the announcement.
4. Download the ISO image that is referenced in the announcement and save it to your local disk.
5. Run the `md5sum` command to print the MD5 hash and compare it with the value in the announcement. For example:

   ```
   $ md5sum DGXOS-6.1.0-2023-08-09-12-30-10.iso
   ```

   Example Output

   ```
   d38620ffa58905330c1efe49b3d7ff53  DGXOS-6.1.0-2023-08-09-12-30-10.iso
   ```

5.2. Installing the DGX OS Image

Install the DGX OS ISO image in one of the following ways:

- Remotely through the BMC for systems that provide a BMC. Refer to *Installing the DGX OS Image Remotely through the BMC* below for instructions.

  Note: This method is not available for DGX Station (V100)

- Locally from a UEFI-bootable USB flash drive or DVD-ROM.

  Refer to *Installing the DGX OS Image from a USB Flash Drive or DVD-ROM*. After obtaining the DGX OS 6 ISO image from NVIDIA Enterprise Support, create a bootable installation medium, such as a USB flash drive or DVD-ROM, that contains the image.

5.2.1. Installing the DGX OS Image Remotely through the BMC

These instructions describe how to re-image the system remotely through the BMC.

After obtaining the DGX OS 6 ISO image from NVIDIA Enterprise Support, make sure the host that you use for your web browser can access the ISO image file.

1. Log in to the BMC.

   Refer to *Connecting to the DGX System* for more information.

2. Click **[Remote Control]** and then click **[Launch KVM]**.
3. Set up the ISO image as virtual media.
   1. From the top bar, click **[Browse File]** and then locate and select the DGX OS ISO file and click **[Open]**
   2. Click **[Start Media]**.
4. Reset the system and boot the virtual media image.
   1. From the top menu, click **[Power]** and select **[Hard Reset]**, then click **[Perform Action]**.
   2. Click **[Yes]** and then **[OK]** at the Power Control dialogs.

      Wait for the system to power down and then come back online.
   3. Refer to *DGX OS ISO Boot Options* for a description of the GRUB menu options and for instructions on completing the installation process.

5.2.2. Installing the DGX OS Image from a USB Flash Drive or DVD-ROM

After obtaining the DGX OS 6 ISO image from NVIDIA Enterprise Support, create a bootable installation medium, such as a USB flash drive or DVD-ROM, that contains the image.

- To create a bootable USB flash drive, refer to one of the following links for more information:
  - On Linux, refer to *Creating a Bootable USB Flash Drive by Using the dd Command*.
  - On Windows, refer to *Creating a Bootable USB Flash Drive by Using Akeo Rufus*.
- To create a bootable DVD ROM, refer to Burning the ISO on to a DVD-ROM on the Ubuntu Community Help Wiki for more information about the available methods.

5.2.2.1 Creating a Bootable USB Flash Drive by Using the dd Command

On a Linux system, you can use the dd command to create a bootable USB flash drive that contains the DGX OS software image.

Note: To ensure that the resulting flash drive is bootable, use the dd command to perform a device bit copy of the image. If you use other commands to perform a simple file copy of the image, the resulting flash drive may not be bootable.

Ensure that the following prerequisites are met:

- The correct DGX OS software image is saved to your local disk.

  For more information, refer to *Obtaining the DGX OS ISO Image*.
- The USB flash drive meets the following requirement:
  - The USB flash drive has a capacity of at least 16 GB.
  - (DGX A100 only) The partition scheme on the USB flash drive is a GPT partition for UEFI.

Create the bootable USB Flash drive:

1. Plug the USB flash drive into one of the USB ports of your Linux host. Obtain the device name of the USB flash drive by running the lsblk command.

   ```
   lsblk
   ```

   You can identify the USB flash drive from its size, which is much smaller than the size of the SSDs in the DGX software, and from the mount points of any partitions on the drive, which are under `/media`.

   In the following example output, the device name of the USB flash drive is `sde`.

   ```
   NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
   sda      8:0    0   1.8T  0 disk
   |_sda1   8:1    0   121M  0 part /boot/efi
   |_sda2   8:2    0   1.8T  0 part /
   sdb      8:16   0   1.8T  0 disk
   |_sdb1   8:17   0   1.8T  0 part
   sdc      8:32   0   1.8T  0 disk
   sdd      8:48   0   1.8T  0 disk
   sde      8:64   1   7.6G  0 disk
   |_sde1   8:65   1   7.6G  0 part /media/deeplearner/DGXSTATION
   ```

2. As root, convert and copy the image to the USB flash drive.

   ```
   sudo dd if=<path-to-ISO-image> bs=2048 of=<usb-drive-device-name>
   ```

   > **Warning:** The `dd` command erases all data on the device that you specify in the `of` argument. To avoid losing data, ensure that you specify the correct path to the USB flash drive.

5.2.2.2 Creating a Bootable USB Flash Drive by Using Akeo Rufus

On a Windows system, you can use the Akeo Reliable USB Formatting Utility (Rufus) to create a bootable USB flash drive that contains the DGX OS software image.

Ensure that the following prerequisites are met:

- The correct DGX OS software image is saved to your local disk.

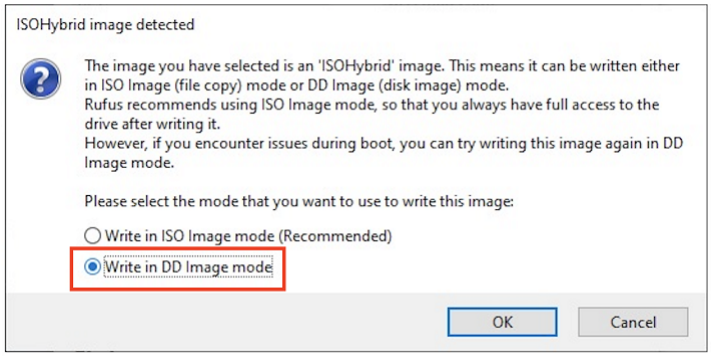
  For more information, refer to *Obtaining the DGX OS ISO Image*.

- The USB flash drive has a capacity of at least 16 GB.

Follow these steps to create the bootable USB Flash drive:

1. Plug the USB flash drive into one of the USB ports of your Windows system.
2. Download and launch the Akeo Reliable USB Formatting Utility (Rufus).
3. In **Drive Properties**, select the following options:
   1. In **Device**, select your USB flash drive.
   2. In **Boot selection**, click **[SELECT]**, locate, and select the DGX OS software image.

   You can leave the other settings at the default.
4. Click **[Start]**. This step prompts you to select whether to write the image in ISO Image mode (file copy) or DD Image mode (disk image).



5. Select **[Write in DD Image mode]** and click **[OK]**.

5.2.3. Booting the DGX OS ISO image

These instructions describe how to boot the DGX OS ISO image locally.

1. Plug the USB flash drive containing the OS image into the DGX system.
2. Connect a monitor and keyboard directly to the DGX system.
3. Boot the system and then press **F11** when the NVIDIA logo appears to access the boot menu.
4. Select the USB volume name that corresponds to the inserted USB flash drive and boot the system from it.

Refer to *DGX OS ISO Boot Options*) for a description of the GRUB menu options and for information about completing the installation process.

5.3. DGX OS ISO Boot Options

This section provides information about the available installation and boot options of the DGX OS ISO installer.

These instructions assume that you have booted the DGX OS ISO, either remotely through the BMC or locally from a USB flash drive.

- When the system boots up, select one of the following options from the GRUB menu:
  - **Install DGX OS <version>**
  - Install DGX OS <version>: Without Reformatting Data RAID (does not mount /raid)
  - Advanced Installation Options - Install DGX OS <version> Without NVIDIA Drivers - Install DGX OS <version> With Encrypted Root - Install DGX OS <version> With Encrypted Root and Without Reformatting Data RAID
  - Boot Into Live Environment
  - Check Media for Defects

  See the following sections for more information about these options.

- Verify that the DGX system booted up and that the image is being installed.

  This process will iterate through the software components and copy and install them showing the executed commands. This process generally takes between 15 and 60 minutes, depending on DGX platform, and how the system is being imaged (for example, BMC over a slow network or locally with a fast USB flash drive).

Note: On DGX servers, the NVIDIA InfiniBand driver is installed and the firmware on the ConnectX cards are updated. This process can take up to 5 minutes for each card. Other system firmware is not updated.

After the installation is complete, the system reboots into the OS, and prompts for configuration information. Refer to *Initial Setup* for more information about how to boot up the DGX system for the first time after reimaging the system.

5.3.1. Install DGX OS

Here are the steps to install your DGX system and reformat the data RAID.

When you accept this option, the installation process repartitions all drives, including the OS and the data drives. The data drives are configured as a RAID array and mounted under the `/raid` directory. This process overwrites all the data and file systems that might exist on the OS and data drives. The RAID array on the DGX data disks is intended to be used as a cache and not for long-term data storage, so reformatting the data RAID should not be disruptive.

These changes are preserved across system reboots.

5.3.2. Install DGX OS without Reformatting the Data RAID

Here are the steps to install your DGX system without reformatting the data RAID.

The RAID array on the DGX data disks is intended for use as a cache and not for long-term data storage, so this should not be disruptive. However, if you are an advanced user and have set up the disks for a non-cache purpose and want to keep the data on those drives, select **Install DGX system without Reformatting the Data RAID** option at the boot menu during the boot installation. This option retains data on the RAID disks, and the following tasks are completed:

- Installs the cache daemon but leaves it disabled by commenting out the `RUN=yes` line in `/etc/default/cachefilesd`
- Creates a `/raid` directory, leaves it out of the file system table by commenting out the entry containing `/raid` in `/etc/fstab`
- Does not format the RAID disks.

When the installation is completed, you can repeat any configuration steps that you had performed to use the RAID disks as other than cache disks. You can always choose to use the RAID disks as cache disks later by enabling `cachefilesd` and adding `/raid` to the file system table:

1. Uncomment the `#RUN=yes` line in `/etc/default/cachefilesd`.
2. Uncomment the `/raid` line in `etc/fstab`.
3. Run the following:

1. Mount `/raid`.

```
sudo mount /raid
```

2. Reload the systemd manager configuration.

```
systemctl daemon-reload
```

3. Start the cache daemon.

```
systemctl start cachefilesd.server
```

These changes are preserved across system reboots.

5.3.3. Advanced Installation Options (Encrypted Root)

When you select this menu item, you have the ability to encrypt the root filesystem of the DGX system.

> **Warning:** Select this option only if you want to encrypt the root filesystem.

Aside from the encrypted root filesystem, the behavior is identical to the default installation.

Selecting **Encrypted Root** instructs the installer to encrypt the root filesystem. The encryption is fully automated and you will be required to manually unlock the root partition by entering a passphrase at the console (through a direct keyboard and mouse connection or through the BMC) each time the system boots.

When you power on your DGX system for the first time, you are prompted to accept end user license agreements for NVIDIA software. You are then guided through the process to complete the initial Ubuntu OS configuration, you can create your passphrase for the drive. If necessary, you can change this passphrase later. For more details see *First Boot Process for DGX Servers* or *First Boot Process for DGX Station*.

> **Warning:** Encryption **cannot be** enabled or disabled after the installation. To change the encryption state again, you need to reimage the drives.

5.3.4. Boot Into a Live Environment

The DGX OS installer image can also be used as a Live image, which means that the image boots up and runs a minimal DGX OS in system memory and does not overwrite anything on the disks in the system.

Live mode does not load drivers, and is essentially a simple Ubuntu Server configuration. This mode can be used as a tool to debug a system when the disks on the system are not accessible or should not be touched.

In a typical operation, this option should not be selected.

5.3.5. Check Disc for Defects

Here is some information about how you can check the disc for defects.

If you are experiencing anomalies when you install the DGX OS, and suspect the installation media might have an issue, selecting this item to complete an extensive test of the install media contents.

The process is time consuming, and the installation media is usually is not the source of the problem. In a typical operation, this option should not be selected.

Chapter 6. Installing DGX Software on Ubuntu

This sections explains the steps for installing and configuring Ubuntu and the NVIDIA DGX Software Stack on DGX systems.

DGX OS provides a customized installation of Ubuntu with additional software from NVIDIA to provide a turnkey solution for running AI and analytics workloads. The additional software, the NVIDIA DGX Software Stack, comprises platform-specific configurations, diagnostic and monitoring tools, and drivers that are required for a stable, tested, and supported OS to run AI, machine learning, and analytics applications on DGX systems.

You also have the option to install the NVIDIA DGX Software Stack on top of a vanilla Ubuntu distribution while still benefiting from the advanced DGX features. This installation method supports more flexibility, such as custom partition schemes.

Cluster deployments also benefit from this installation method by taking advantage of Ubuntu`s standardized automated and non-interactive installation process. Starting with Ubuntu 20.04, the installer introduced a new mechanism for automating the installation allowing system administrators to install a system unattended and non-interactively. You can find information for creating such a *cloud-init* configuration file in *Cloud-init Configuration File*. Refer to Ubuntu Automated Server Installation for more details.

The intended audience are IT professionals managing a cluster of DGX systems and integration partners.

6.1. Prerequisites

The following prerequisites are required or recommended, where indicated.

6.1.1. Ubuntu Software Requirements

The DGX Software Stack requires the following software versions:

- Ubuntu 22.04
- Linux Kernel 5.15 LTS

6.1.2. Access to Software Repositories

The DGX Software Stack is available from repositories that can be accessed from the internet. If your installation does not allow connection to the internet, see appendix *Air-Gapped Installations* for information about installing and upgrading software on "air-gapped" systems.

If you are using a proxy server, then follow the instructions in the section *Network Configuration* for setting up a proxy configuration.

6.2. Installation Considerations

Installing the NVIDIA DGX Software Stack on Ubuntu allows you to select from additional configuration options that would otherwise not be available with the preconfigured DGX OS installer. This includes drive partitioning, filesystem choices, and software selection.

Before you start installing Ubuntu and the NVIDIA DGX Software Stack, you should evaluate the following options. The installation and configuration instructions will be covered in the respective section of this document.

6.2.1. System Drive Mirroring (RAID-1) [recommended]

The DGX H100, DGX A100 and DGX-2 systems embed two system drives for mirroring the OS partitions (RAID-1). This ensures data resiliency if one drive fails. If you want to enable mirroring, you need to enable it during the drive configuration of the Ubuntu installation. It cannot be enabled after the installation.

6.2.2. Data Drive RAID-0 or RAID-5

DGX systems are equipped with multiple data drives that can be configured as RAID-0 for performance or RAID-5 for resiliency. RAID-0 provides the maximum storage capacity and performance, but does not provide any redundancy. If a single SSD in the array fails, all data stored on the array is lost.

RAID-0 is recommended for data caching. You can use cachefilesd to provide caching for NFS shares. The network file system (NFS) is required to take advantage of the cache file system. RAID-5 should be used for persistent data storage.

You have the option to configure RAID and data caching after the initial Ubuntu installation using the `nvidia-config-raid` tool or during the Ubuntu installation. The nvidia-config-raid tool is recommended for manual installation.

Note: The DGX-1 uses a hardware RAID controller that cannot be configured during the Ubuntu installation. You can still use the `nvidia-config-raid` tool or change the configuration in the BIOS.

6.2.3. System Drive Encryption [optional]

Root filesystem encryption is a software-based method to protect the content stored in the system partition(s) from unauthorized access by encrypting the data on-the-fly. It requires users to unlock the filesystem on every boot, either manually by entering a passphrase or automatically using a centralized key server.

System drive encryption can only be enabled during the installation of Ubuntu.

6.2.4. Data Drive Encryption [optional]

Data drive encryption is only supported on DGX H100 and DGX A100 systems equipped with self-encrypting-drives (SED). It can be enabled after Ubuntu is installed using the `nv-encrypt` tool. It requires either to store the keys in the TPM or external key-management.

6.2.5. System Drive Partitioning

Ubuntu uses only a single partition for the entire filesystem by default. This can be configured during the Ubuntu installation for deployments that require a more faceted partition scheme for security reasons. The recommended partitioning scheme is to use only a single partition for the Linux root partition with the ext4 filesystem.

6.3. Installing Ubuntu

There are several methods for installing Ubuntu as described in the Ubuntu Server Guide.

For convenience, this section provides additional instructions that are specific to DGX for installing Ubuntu following the Basic Installation. If you have a preferred method for installing Ubuntu, then you can skip this section.

Steps that are covered in this section:

- Connecting to the DGX system
- Booting from the install media
- Running the Ubuntu installer (including network and storage configuration steps)

6.3.1. Booting from the Installation Media

During the installation and configuration steps, you need to connect to the console of the DGX system. Refer to *Connecting to the DGX System* for more details.

Boot the Ubuntu ISO image in one of the following ways:

- Remotely through the BMC for systems that provide a BMC.

  Refer to the *Reimaging the System Remotely* section in the corresponding DGX user guide listed above for instructions.

- Locally from a UEFI-bootable USB flash drive or DVD-ROM.

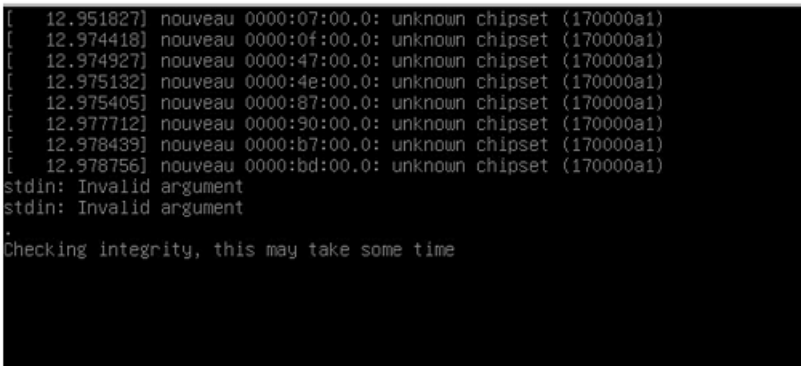
  Refer to *Installing the DGX OS Image from a USB Flash Drive or DVD-ROM* section in the corresponding DGX user guide listed above for instructions.

6.3.2. Running the Ubuntu Installer

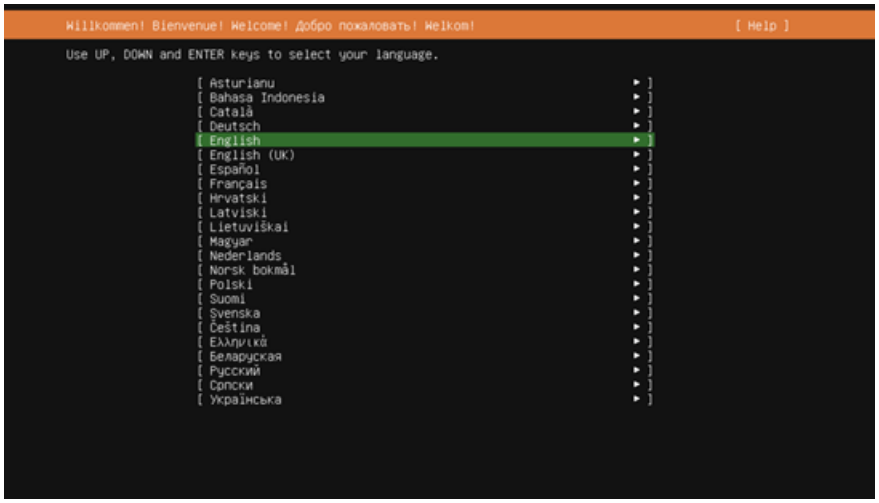
After booting the ISO image, the Ubuntu installer should start and guide you through the installation process.

Note: The screenshots in the following steps are taken from a DGX A100. Other DGX systems have differences in drive partitioning and networking.

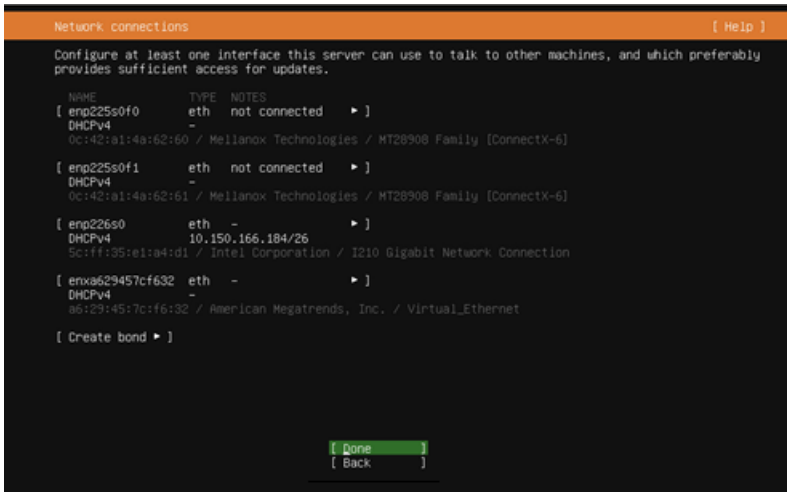
During the boot process of the ISO image, you might see some error messages due to older drivers, etc. They can be safely ignored.



1. Select your language at the welcome screen, then follow the instructions to select whether to update the installer (if offered) and to choose your keyboard..



2. At the Network connections screen, configure your network.



The installer tries to automatically retrieve a DHCP address for all network interfaces, so you should be able to continue without any changes. However, you also have the option to manually configure the interface(s).

3. At the **Guided storage configuration** screen, configure the partitioning and file systems. All DGX systems are shipped preinstalled with DGX OS. The drives are, therefore, already partitioned and formatted. DGX OS installer configures a single ext4 partition for the root partition in addition to the EFI boot partition. You have the following options:

   - Keep the current partition layout and formatting [recommended]
   - Create a custom partition scheme [advanced]
   - Use a single disk with the default Ubuntu partition scheme

   Creating a new custom partition scheme with a RAID configuration is a more involved process and out of the scope for this document. Refer to the Ubuntu installation guide for more information. When you choose the option to use an entire disk, Ubuntu will only use one of the two redundant boot drives.

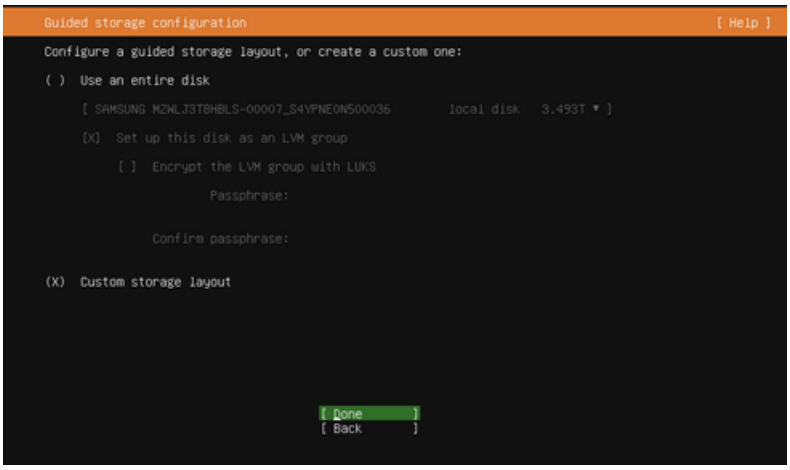
   Note: The RAID level for the data drive can be changed after the installation of Ubuntu.

   The following instructions describe the steps for keeping the current partition layout. It still requires you to re-create and reformat the partitions.

   Note: DGX-1 is using a hardware RAID controller and the RAID membership can only be configured in the RAID controller BIOS. The default configuration consists of two virtual devices:

   - The first virtual device (sda) is a single disk and used as the system drive
   - Second virtual device (sdb) consists of the rest of the disks for data.

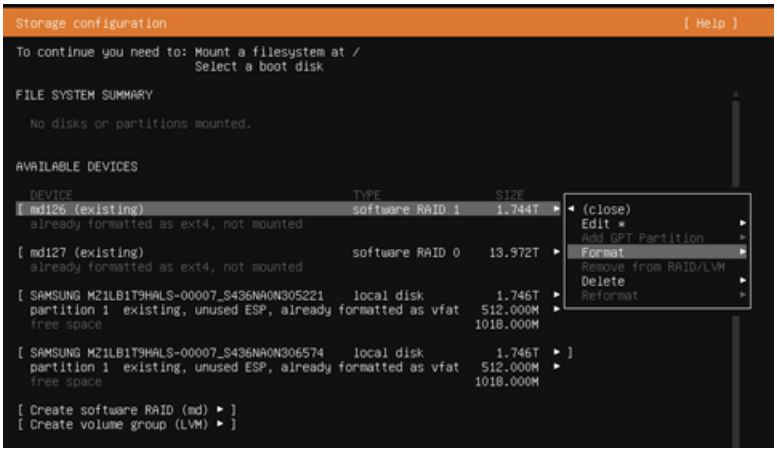
   a. Select Custom storage layout, then click Done.



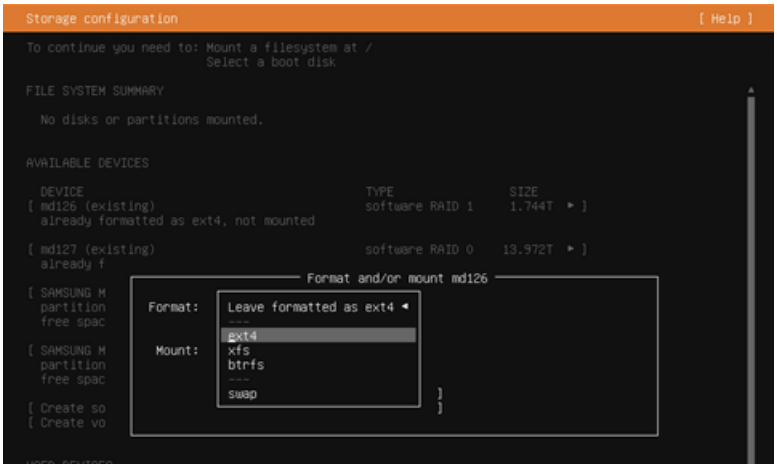
b. Identify the system drive.

The system drive on the DGX-2, DGX A100 and DGX H100 is a RAID 1 array and you should find it easily. The DGX-1 has a hardware RAID controller and you will see a single drive as sda.

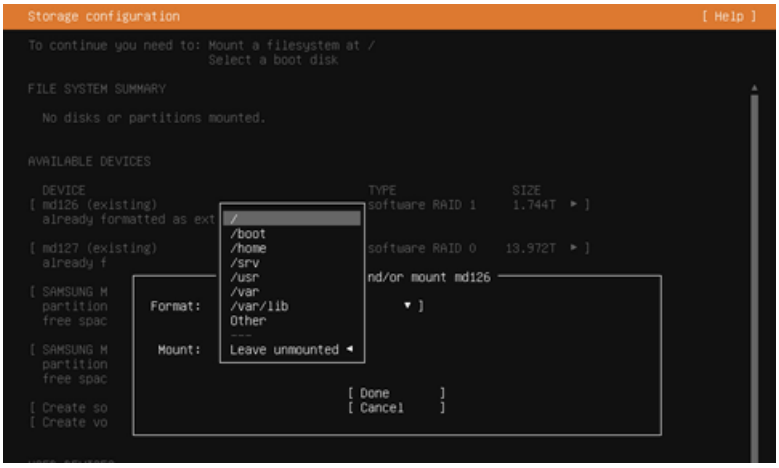
c. Select the system drive and then click Format.



d. Set Format to ext4 (do not select “Leave formatted as <filesystem>”)

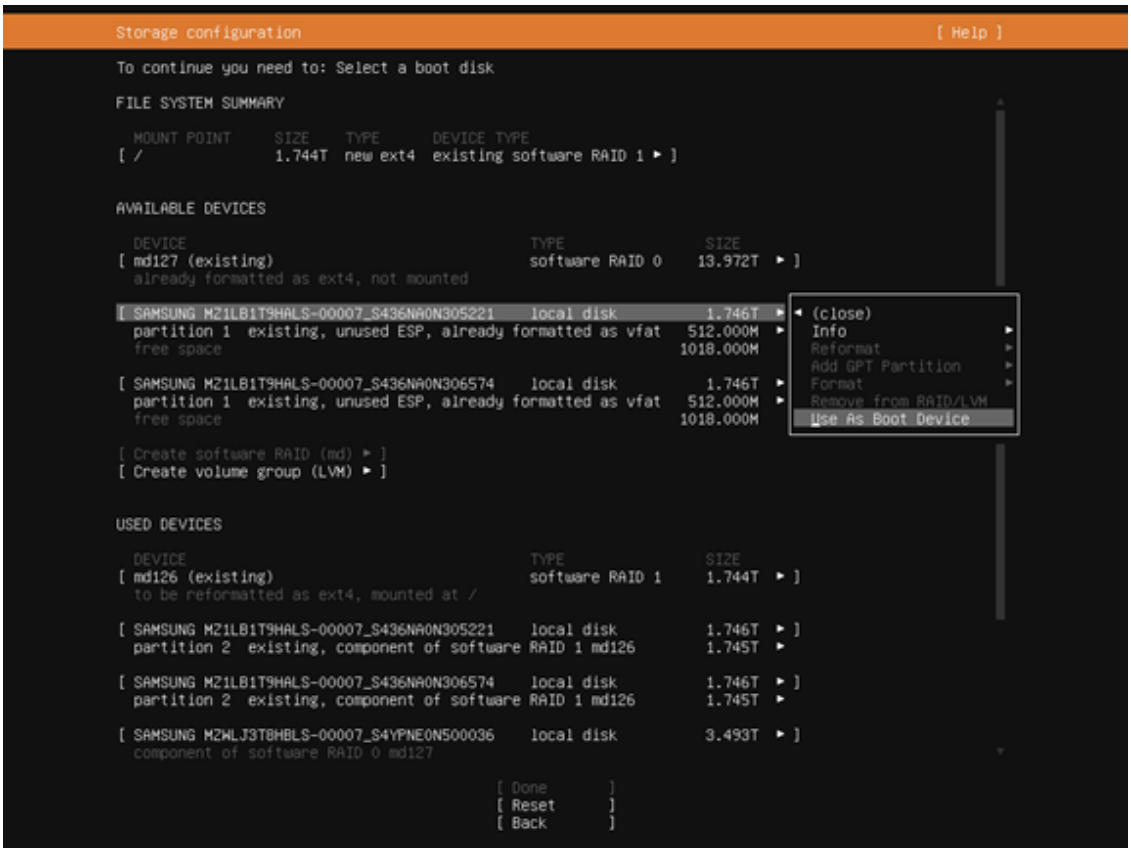


e. Set Mount to “/”:



f. Set the boot flag on the raw devices.

Identify the system drives under AVAILABLE DEVICES (not the RAID array) and select “Use as Boot Device” for the first device. On DGX-2, DGX A100, and DGX H100 that have two drives, repeat this process for the second drive and select “Use as another Boot Device”.



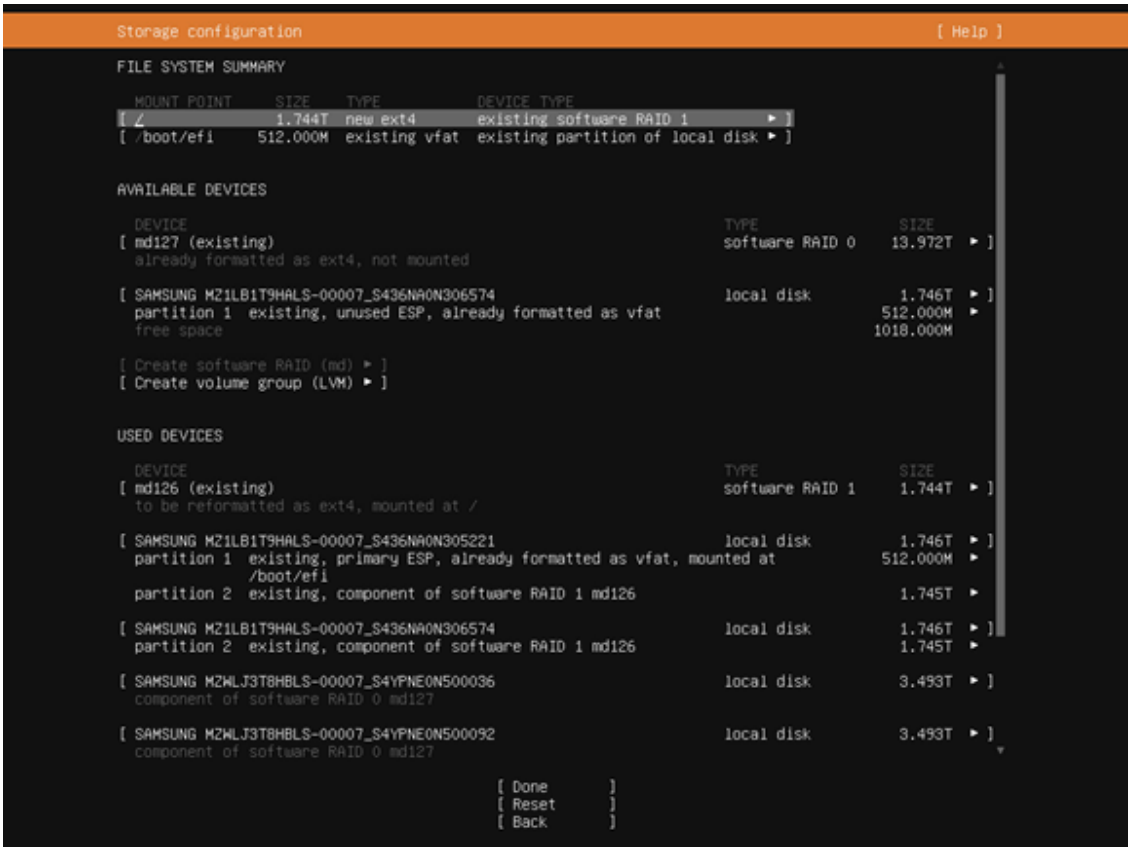
g. Complete the configuration.

- **RAID 0 Array**: In most cases, the RAID 0 array for the data drives will have already been created from the factory. If it hasn’t been created you can either create them in the Storage configurations dialog or by using the `config_raid_array` tool after completing the Ubuntu installation.
- **Enable drive encryption (Optional)**: Note that encryption can only be enabled during

the Storage configuration. It cannot be changed after the installation. To change the encryption state again you need to reinstall the OS. To enable drive encryption, you have to create a virtual group and volume. This is out of the scope for this document. Please refer to the Ubuntu documentation for more details.

- **Swap Partition**: The default installation does not define a swap partition. Linux uses any configured swap partition for temporarily storing data when the system memory is full, incurring a performance hit. With the large memory of DGX systems swapping is not recommended.

The "FILE SYSTEM SUMMARY" at the top of the page should display the root partition on the RAID 1 drive for and a boot/efi partition (the two drives will only show up as a single entry). On DGX-1 with the hardware RAID controller, it will show the root partition on sda.



Select Done and accept all changes.

4. Follow the instructions for the remaining tasks.

   Create a default user in the *Profile setup* dialog and choose any additional SNAP package you want to install in the *Featured Server Snaps* screen.

5. Wait for the installation to complete.

   Log messages are presented while the installation is running.

6. Select **Reboot Now** when the installation is complete to restart the system.

   After reboot, you can log in using the username and password for the user you have created above.

When using LVM, Ubuntu's default partitioning scheme, DGX-2 users may run into https://bugs.launchpad.net/ubuntu/+source/lvm2/+bug/1834250. The "/dev/sda: open failed: No medium found"

messages are harmless, and can be avoided by updating /etc/lvm/lvm.conf with the following filter: "`global_filter = ["r|/dev/sda|"]`":

6.4. Installing the DGX Software Stack

This section requires that you have already installed Ubuntu on the DGX and rebooted the system.

Attention: By installing the DGX Software Stack you are confirming that you have read and agree to be bound by the DGX Software License Agreement. You are also confirming that you understand that any pre-release software and materials available that you elect to install in a DGX may not be fully functional, may contain errors or design flaws, and may have reduced or different security, privacy, availability, and reliability standards relative to commercial versions of NVIDIA software and materials, and that you use pre-release versions at your risk.

6.4.1. Installing DGX System Configurations and Tools

The NVIDIA DGX Software Stack includes system-specific configurations and tools to take advantage of the advanced DGX features. They are provided from NVIDIA repositories in the form of software packages that can be installed on top of a typical Ubuntu installation. All system-specific software components are bundled into meta packages specific to a system:

- system-configurations
- system-tools
- system-tools-extra

For details about the content of these packages, refer to the *Release Notes*.

The following steps enable the NVIDIA repositories and install the system specific packages.

1. Enable the NVIDIA repositories by extracting the repository information.

   This step adds the URIs and configuration preferences to control the package versions that will be installed to the `/etc/apt` directory and the GPG keys for the NVIDIA repositories in the `/usr/share/keyrings` directory.

   ```
   curl https://repo.download.nvidia.com/baseos/ubuntu/jammy/dgx-repo-files.tgz |
   ↪sudo tar xzf - -C /
   ```

2. Update the internal APT database with the latest version information of all packages.

   ```
   sudo apt update
   ```

3. **Recommended**: Upgrade all software packages with the latest versions.

   ```
   sudo apt upgrade
   ```

4. Install the DGX system tools and configurations.

   - For DGX-1, install the DGX-1 configurations and DGX-1 system tools:

```
sudo apt install -y dgx1-system-configurations dgx1-system-tools dgx1-system-
↪tools-extra
```

- For DGX-2, install the DGX-2 configurations and DGX-2 system tools:

```
sudo apt install -y dgx2-system-configurations dgx2-system-tools dgx2-system-
↪tools-extra
```

- For DGX A100, install DGX A100 configurations and DGX A100 system tools:

```
sudo apt install -y dgx-a100-system-configurations dgx-a100-system-tools dgx-
↪a100-system-tools-extra nvidia-utils-525-server
```

- For DGX H100, install DGX H100 configurations and DGX H100 system tools:

```
sudo apt install -y dgx-h100-system-configurations dgx-h100-system-tools dgx-
↪h100-system-tools-extra nvidia-utils-525-server nvfwupd
```

5. Install the `linux-tools-nvidia` package.

```
sudo apt install -y linux-tools-nvidia
```

6. Install the NVIDIA peermem loader package.

```
sudo apt install -y nvidia-peermem-loader
```

7. **Recommended**: Disable unattended upgrades.

   Ubuntu periodically checks for security and other bug fixes and automatically installs updates software packages typically overnight. Because this may be disruptive, you should regularly check for updates and install them manually.

```
sudo apt purge -y unattended-upgrades
```

8. **Recommended**: Enable serial-over-lan console output.

   Note: If you have boot drive encryption enabled, the prompt for entering the passphrase and input will be over the serial console if you install this package.

```
sudo apt install -y nvidia-ipmisol
```

9. **Optional**: Modify the `logrotate` policy to collect more logging information (but size-limited):

```
sudo apt install -y nvidia-logrotate
```

   The configuration changes will take effect only after rebooting the system. To minimize extra reboots, you can defer this step until after the drivers have been installed later in this document.

6.4.2. Configuring Data Drives

The data drives in the DGX systems can be configured as RAID 0 or RAID 5. RAID 0 provides the maximum storage capacity and performance, but does not provide any redundancy.

RAID 0 is often used for data caching. You can use cachefilesd to provide a cache for NFS shares.

Important: You can change the RAID level later but this will destroy the data on those drives.

Except for the DGX-1, the RAID configuration can be configured during the Ubuntu installations. If you have already configured the RAID array during the Ubuntu installation, you can skip the first step and go to step 2.

1. Configure the /raid partition.

   All DGX systems support RAID 0 and RAID 5 arrays.

   - To create a RAID 0 array:

     ```
     sudo /usr/bin/configure_raid_array.py -c -f
     ```

   - To create a RAID 5 array:

     ```
     sudo /usr/bin/configure_raid_array.py -c -f -5
     ```

   The command creates the `/raid` mount point and RAID array, and adds a corresponding entry in `/etc/fstab`.

2. **Optional**: Install tools for managing the self-encrypting drives (SED) for the data drives on the DGX A100 or DGX H100.

   Refer to *Managing Self-Encrypting Drives* for more information.

3. **Optional**: If you wish to use your RAID array for read caching of NFS mounts, you can install `cachefilesd` and set the cachefs option for an NFS share.

   a. Install `cachefilesd` and `nvidia-conf-cachefilesd`.

      This will update the cachefilesd configuration to use the `/raid` partition.

      ```
      sudo apt install -y cachefilesd nvidia-conf-cachefilesd
      ```

   b. Enable caching on all NFS shares you want to cache by setting the `fsc` flag.

      Edit `/etc/fstab` and add the `fsc` flag to the mount options as shown in this example.

      ```
      <nfs_server>:<export_path> /mnt nfs rw,noatime,rsize=32768,wsize=32768,nolock,
      ↪tcp,intr,fsc,nofail 0 0
      ```

   c. Mount the NFS share.

      If the share is already mounted, use the remount option.

      ```
      mount <mount-point> -o,remount
      ```

   d. To validate that caching is enabled, issue the following.

      ```
      cat /proc/fs/nfsfs/volumes
      ```

Look for the text `FSC=yes` in the output of the command. The NFS will be mounted with caching enabled upon subsequent reboot cycles.

6.4.3. Installing the GPU Driver

You have the option to choose between different GPU driver branches for your DGX system. The latest driver release includes new features but might not provide the same support duration as an older release. Consult the Data Center Driver Release Notes for more details and the minimum required driver release for the GPU architecture.

Use the following command to display a list of installed drivers.

1. Ensure to have the latest version of the package database.

```
sudo apt update
```

2. Display a list of all available drivers.

```
sudo apt list nvidia-driver*server
```

Example Output:

```
nvidia-driver-450-server/jammy-updates,jammy-security 450.216.04-0ubuntu0.22.04.1
↪amd64
nvidia-driver-460-server/jammy-updates,jammy-security 525.161.03-0ubuntu0.22.04.1
↪amd64
nvidia-driver-525-server/jammy-updates,jammy-security 525.161.03-0ubuntu0.22.04.1
↪amd64
nvidia-driver-510-server/jammy-updates,jammy-security 515.86.01-0ubuntu0.22.04.1
↪amd64
nvidia-driver-515-server/jammy-updates,jammy-security 515.86.01-0ubuntu0.22.04.1
↪amd64
nvidia-driver-525-server/jammy-updates,jammy-security 525.60.13-0ubuntu0.22.04.1
↪amd64
```

The following steps install the NVIDIA CUDA driver and configure the system. Replace the release version used as an example (525) with the release you want to install. Ensure that the driver release you intend to install is supported by the GPU in the system.

3. Ensure you have the latest version of the package database.

```
sudo apt update
```

4. Ensure you have the latest kernel version installed.

The driver package has a dependency to the kernel and updating the database might have updated the version information.

```
sudo apt install -y linux-nvidia
```

5. Install NVIDIA CUDA driver.

   - For non-NVswitch systems like DGX-1:

```
sudo apt install -y nvidia-driver-525-server linux-modules-nvidia-525-server-
↪nvidia libnvidia-nscq-525 nvidia-modprobe datacenter-gpu-manager nv-
↪persistence-mode
```

- For NVswitch systems like DGX-2, DGX A100, and DGX H100, be sure to also install the fabric-manager package:

```
sudo apt install -y nvidia-driver-525-server linux-modules-nvidia-525-server-
↪nvidia libnvidia-nscq-525 nvidia-modprobe nvidia-fabricmanager-525
↪datacenter-gpu-manager nv-persistence-mode
```

6. Enable the persistenced daemon and other services:
   - For non-NVswitch systems, such as DGX-1:

```
sudo systemctl enable nvidia-persistenced nvidia-dcgm
```

   - For NVswitch systems like DGX-2, DGX A100, and DGX H100, be sure to also enable the NVIDIA fabric manager service:

```
sudo systemctl enable nvidia-fabricmanager nvidia-persistenced nvidia-dcgm
```

7. Reboot the system to load the drivers and to update system configurations.
   a. Issue reboot.

```
sudo reboot
```

   b. After the system has rebooted, verify that the drivers have been loaded and are handling the NVIDIA devices.

```
nvidia-smi
```

   The output should show all available GPUs and the Persistence-Mode *On*:

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 535.54.03  Driver Version: 535.54.03  CUDA Version: 12.2         |
|-------------------------------+----------------------+----------------------+
| GPU Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan Temp Perf Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
| 0 Tesla V100-SXM2...    On | 00000000:06:00.0 Off |                    0 |
| N/A   35C   P0   42W / 300W |      0MiB / 16160MiB |   0%          Default |
|                               |                      |                   N/A |
+-------------------------------+----------------------+----------------------+
| 1 Tesla V100-SXM2...    On | 00000000:07:00.0 Off |                    0 |
| N/A   35C   P0   44W / 300W |      0MiB / 16160MiB |   0%          Default |
|                               |                      |                   N/A |
+-------------------------------+----------------------+----------------------+
...
+-------------------------------+----------------------+----------------------+
| 7 Tesla V100-SXM2...    On | 00000000:8A:00.0 Off |                    0 |
| N/A  35C   P0   43W / 300W  |      0MiB / 16160MiB |   0%          Default |
|                               |                      |                   N/A |
+-------------------------------+----------------------+----------------------+
+-----------------------------------------------------------------------------+
| Processes:                                                                  |
| GPU   GI   CI        PID     Type      Process name            GPU Memory |
|       ID   ID                                                   Usage      |
|=============================================================================|
| No running processes found                                                  |
+-----------------------------------------------------------------------------+
```

6.4.4. Installing the Mellanox OpenFabrics Enterprise Distribution (MLNX_OFED)

DGX systems include high-performance network cards to connect to other systems over Infiniband or Ethernet. You have the option between the driver included in Ubuntu and the Mellanox OpenFabrics Enterprise Distribution (Mellanox OFED or MOFED).

The following steps describe how to install MOFED and all the required additional software in place of the default OFED Ubuntu driver.

1. Install the nvidia-manage-ofed package:

```
sudo apt install -y nvidia-manage-ofed
```

2. Remove the inbox OFED components:

```
sudo /usr/sbin/nvidia-manage-ofed.py -r ofed
```

3. Add the Mellanox OFED components:

```
sudo /usr/sbin/nvidia-manage-ofed.py -i mofed
```

Note: The command installs the latest version of MLNX_OFED that is currently available in the repositories. To install an alternative version other than the latest version, specify the alternative version by using the `-v` option. The following example installs MLNX_OFED version 5.9-0.5.6.0:

```
sudo /usr/sbin/nvidia-manage-ofed.py -i mofed -v 5.9-0.5.6.0
```

4. Reboot the system.

6.4.5. Installing Docker and the NVIDIA Container Toolkit

Containers provide isolated environments with a full filesystem of the required software for specific applications. To use the NVIDIA provided containers for AI and other frameworks on the DGX and GPUs, you need to install Docker and the NVIDIA Container Toolkit. It takes care of providing access to the GPUs to the software running inside the container.

Note that these tools are also required by the Firmware Update Containers for upgrading the system firmware.

1. Install docker-ce, NVIDIA Container Toolkit, and optimizations for typical DL workload.

```
sudo apt install -y docker-ce nvidia-container-toolkit nv-docker-options
```

2. Restart the docker daemon.

```
sudo systemctl restart docker
```

To validate the installation, run a container and check that it can access the GPUs. The following instructions assume that the NVIDIA GPU driver has been installed and loaded.

Note: This validation downloads a container from the NGC registry and requires that the system has internet access.

1. Execute the following command to start a container and run the nvidia-smi tool inside the container:

```
sudo docker run --gpus=all --rm nvcr.io/nvidia/cuda:12.3.2-base-ubuntu22.04
↪nvidia-smi
```

2. Verify that the output shows all available GPUs and has Persistence-Mode set to On.

6.4.6. Installing the NVIDIA System Management (NVSM) Tool [Recommended]

The NVIDIA System Management (NVSM) is a software framework for monitoring NVIDIA DGX nodes in a data center. It allows customers to get a quick health report of the system and is typically required by the NVIDIA support team to resolve issues.

The following steps install and configure NVSM.

1. Install the NVIDIA System Management tool (NVSM):

```
sudo apt install -y nvsm
```

2. Optional: Modify message-of-the-day (MOTD) to display NVSM health monitoring alerts and release information.

```
sudo apt install -y nvidia-motd
```

6.4.7. Additional Software Installed By DGX OS

The Ubuntu and the NVIDIA repositories provide many additional software packages for a variety of applications. The DGX OS Installer, for example, installs several additional software packages to aid system administration and developers that are not installed by default.

The following steps install the additional software packages that get installed by the DGX OS Installer:

1. Install additional software for system administration tasks:

```
sudo apt install -y chrpath cifs-utils fping gdisk iperf ipmitool lsscsi net-
↪tools nfs-common quota rasdaemon pm-utils samba-common samba-libs sysstat vlan
```

2. Install additional software for development tasks:

```
sudo apt install -y build-essential automake bison cmake dpatch flex gcc-multilib
↪gdb g++-multilib libelf-dev libltdl-dev linux-tools-generic m4 swig
```

The NVIDIA CUDA Developer repository provides an easy mechanism to deploy NVIDIA tools and libraries, such as the CUDA toolkit, cuDNN, or NCCL.

6.5. Next Steps and Additional Information

For further installation and configuration options, refer also to these chapters:

- *Managing and Upgrading Software* - installing additional software and changing driver branches
- *Network Configuration* - additional network options and configurations
- *Data Storage Configuration* - RAID configurations and encryption information
- *Running NGC Containers* - running NGC containers on the system

Chapter 7. Upgrading the OS

This section provides information about upgrading an existing DGX OS installation.

If you want to reimage the system with DGX OS to a default state, refer to *Reimaging the System* for more information.

Important: Before you upgrade a system or any installed software, always consult the *Release Notes* for the latest information about available upgrades. You can find out more about the release cadence and release methods for DGX OS in *Release Guidance*

This release incorporates the following updates:

- Ubuntu ConnectX drivers and OFED stack
- Customers are advised to consider these updates and any effect they may have on their application. For example, some MOFED-dependent applications may be affected.
- Best practices support upgrading select systems and verifying that your applications are working as expected before deploying on additional systems.

Here is some information that describes the difference between the different types of upgrades:

- When you perform a **release upgrade**, you currently have the DGX OS 5 installed, and you want to move to DGX OS 6.

  You can upgrade to DGX OS 6 only from the latest DGX OS 5.x release. Refer to *Performing a Release Upgrade from DGX OS 5* for the upgrade instructions. The instructions also provide information about completing an over-the-internet upgrade.
- When you perform **package upgrades**, you want to install upgrades that have been made available in the repositories since the initial DGX OS 6 release. The repositories are periodically updated with packages that include bug fixes and security updates. The NVIDIA repository also includes packages with new features that are available with the latest DGX OS minor version release. Refer to *Performing Package Upgrades* for instructions.

Note: If you want to **change the branch** of a driver or CUDA Toolkit, refer to *Managing and Upgrading Software* for instructions.

Upgrades are cumulative, which means that your systems will install all available upgrades, including upgrades available from Ubuntu, such as the kernel. Performing upgrades will install the latest versions available at the time when the upgrade is performed. **These may be newer than the current DGX OS release**.

Important: The instructions in this chapter upgrade all software for which updates are available from your configured software sources, including applications that you installed yourself. If you want to prevent an application from being upgraded, you can instruct the Ubuntu package manager to keep the current version.

For more information, refer to the Ubuntu Community Help Wiki: Introduction to Holding Packages. It is typically not advised to hold packages as it can disrupt package dependencies.

Important: When you upgrade DGX OS, the system remains on the installed GPU driver branch unless the installed GPU driver branch is end of support. When a GPU driver branch reaches end of support, you will automatically transition to the next supported branch. Refer to *Changing Your GPU Branch* for instructions on manually switching GPU driver branches.

7.1. DGX OS 6 Release Upgrade Advisory

Here is some additional information when you intend to perform a release upgrade from DGX OS 5:

- NGC Containers

  With DGX OS 6, customers should update their NGC containers to container release 20.10.17 or later if they are using multi-node training. For all other use cases, refer to the NCG Framework Containers Support Matrix. Refer to the NVIDIA Deep Learning Frameworks documentation for information about the latest container releases and how to access the releases.

- Ubuntu ConnectX drivers and OFED stack

  For a release upgrade from DGX OS 5, the Mellanox OFED (MOFED) drivers are replaced with the OFED drivers from DGX OS 6.

  Customers are advised to consider these updates and any effect they may have on their application. For example, some MOFED-dependent applications may be affected.

  After you perform the release upgrade, you can replace the OFED drivers with MOFED. Refer to *Installing the Mellanox OFED Drivers* for more information.

  Best practices support upgrading select systems and verifying that your applications are working as expected before deploying on additional systems.

7.2. Getting Release Information for DGX Systems

Here is some information about how you can determine the release information for your DGX systems.

The `/etc/dgx-release` file provides release information, such as the product name and serial number. This file also tracks the history of the DGX OS software updates by providing the following information:

- The version number and installation date of the last version to be installed from an ISO image `DGX_SWBUILD_VERSION`.

- The version number and update date of each over-the-network update applied since the software was last installed from an ISO image (`DGX_OTA_VERSION`).

For DGX OS 6, the `DGX_OTA_VERSION` file indicates the latest ISO version that was released, and upgrades to the system include the changes that were made in the network repository up to the indicated date. You can use this information to determine whether your DGX system is running the current version of the DGX OS software.

To get release information for the DGX system, view the content of the `/etc/dgx-release` file. For example:

```
more /etc/dgx-release

DGX_NAME="DGX Station A100"
DGX_PRETTY_NAME="NVIDIA DGX Station A100"
DGX_SWBUILD_DATE="2022-10-11-17-49-32"
DGX_SWBUILD_VERSION="5.4.1"
DGX_COMMIT_ID="38d36e8"
DGX_PLATFORM="DGX Station A100"
DGX_SERIAL_NUMBER="1632920000024"

DGX_OTA_VERSION="5.5.0"
DGX_OTA_DATE="Mon 10 Apr 2023 10:11:07 PM PDT"

DGX_OTA_VERSION="6.0.10"
DGX_OTA_DATE="Thu Apr 13 04:55:25 PM PDT 2023"
```

7.3. Preparing to Upgrade the Software

This section provides information about the tasks you need to complete before you can upgrade your DGX OS software.

7.3.1. Connect to the DGX System Console

Connect to the console of the DGX system using a direct connection or a remote connection through the BMC. See *Connecting to the DGX System*

Note: SSH can be used to perform the upgrade. However, if the Ethernet port is configured for DHCP, the IP address might change after the DGX server is rebooted during the upgrade, which results in the loss of connection. A loss of connection might also occur if you are connecting through a VPN. If this happens, connect by using a direct connection or through the BMC to continue the upgrade process. Warning: Connect directly to the DGX server console if the DGX is connected to a 172.17.xx.xx subnet.

DGX OS software installs Docker CE, which uses the 172.17.xx.xx subnet by default for Docker containers. If the DGX server is on the same subnet, you will not be able to establish a network connection to the DGX server.

See *Configuring Docker IP Addresses* To ensure that your DGX system can access the network interfaces for Docker containers, Docker should be configured to use a subnet distinct from other network resources used by the DGX system. for instructions on how to change the default Docker network settings after performing the upgrade.

If you are using a GUI to connect to the console, see *Performing Package Upgrades Using the GUI*. You can use the graphical Software Updater application to manage package upgrades on the DGX Station.

7.3.2. Verifying the DGX System Connection to the Repositories

Before you attempt to complete the update, you can verify that the network connection for your DGX system can access the public repositories and that the connection is not blocked by a firewall or proxy.

On the DGX system, enter the following:

```
wget -O f1-changelogs http://changelogs.ubuntu.com/meta-release-lts
```

```
wget -O f2-archive http://archive.ubuntu.com/ubuntu/dists/jammy/Release
```

```
wget -O f3-security http://security.ubuntu.com/ubuntu/dists/jammy/Release
```

```
wget -O f4-nvidia-baseos http://repo.download.nvidia.com/baseos/ubuntu/jammy/x86_64/
↪dists/jammy/Release
```

```
wget -O f5-nvidia-cuda https://developer.download.nvidia.com/compute/cuda/repos/
↪ubuntu2204/x86_64/Release
```

The `wget` commands should be successful, and there should be five files in the directory with non-zero content.

7.4. Performing a Release Upgrade from DGX OS 5

Here you find information on performing a release upgrade from DGX OS 5 to DGX OS 6.

Important: If installed software packages do not have upgrade candidates, and you try to upgrade, an error message will be displayed. You need to use the `--force` option, and upgrade process. Refer to the *Release Notes* for a list of packages that are no longer available in DGX OS 6.

7.4.1. Upgrade DGX OS 5 to the Latest Version

See “Upgrading” in the DGX OS 5 User Guide

Before you can perform the release upgrade of your system, you need to upgrade the current DGX OS 5 to the latest version. These steps upgrade your system to the latest DGX OS 5 release:

1. If you have **DGX OS 5.2 or earlier** installed, please see the “upgrading” section of the DGX OS 5 User Guide and the following release notes for instructions and details:

2. Download information from all configured sources about the latest versions of the packages.

```
sudo apt update
```

3. Install all available upgrades for your **current** DGX OS release.

```
sudo apt -y full-upgrade
```

Note: Depending on which packages were updated when running `sudo apt -y full-upgrade`, you might be prompted to reboot the system before performing `nvidia-release-upgrade`

7.4.2. Performing the Release Upgrade

Follow these steps to upgrade your system from DGX OS 5 to DGX OS 6:

1. Install the `nvidia-release-upgrade` package for upgrading to the latest DGX OS 5 release.

```
sudo apt install -y nvidia-release-upgrade
```

Note: The next step might install a newer GPU driver. To select a specific driver branch, edit the file `/etc/update-manager/release-upgrades.d/nvidia.cfg` and change the Driver-Branch setting.

2. Start the DGX OS release upgrade process.

```
sudo nvidia-release-upgrade
```

If you are using a proxy server, add the `-E` option to keep your proxy environment variables. For example:

```
sudo -E nvidia-release-upgrade
```

Note: Some package upgrades require that you reboot the system before completing the upgrade. Ensure that you reboot the system when prompted.

3. Resolve conflicts.

Refer to *Resolving Release Upgrade Conflicts* for details and instructions.

4. Wait for the upgrade process to complete and press **y** at the prompt that appears when the system upgrade is completed.

```
System upgrade is complete. Restart required To finish the upgrade, a
restart is required. If you select 'y' the system will be restarted.
Continue [yN]
```

The system must be restarted to complete the update process and ensure that any changes are captured by restarted services and runtimes.

Note: If no reboot prompt appeared or if you did not restart the system when prompted, then reboot to complete the update process.

```
sudo reboot
```

After the system is restarted, the upgrade process takes several minutes to perform some final installation steps.

7.4.3. Resolving Release Upgrade Conflicts

During the upgrade, the system might encounter conflicts or require other manual intervention.

- When you are prompted to resolve conflicts in configuration files, evaluate the changes before selecting one of the following options:
  - Accepting the maintainer's version.
  - Keeping the local version.
  - Manually resolving the difference.

  Conflicts in some configuration files might be the result of customizations to the Ubuntu Desktop OS made for DGX OS software. For guidance about how to resolve these conflicts, see the chapter in the *Release Notes* for the release family to which you are upgrading.

  - `/etc/apt/sources.list.d/dgx.list`. You should install the package maintainer's version.
  - `/etc/ssh/sshd_config`. You can keep the local version that is currently installed.

  Conflicts in the following configuration files are the result of customizations to the Ubuntu Desktop OS made for DGX OS 6.

  - `/etc/gdm3/custom.conf.distrib`. You can keep your currently installed version.
  - `/etc/gdm3/custom.conf`. You can keep your currently installed version.
- If you are logged in to the DGX system remotely through secure shell (SSH), you are prompted about whether you want to continue running under SSH.

```
Continue running under SSH?
This session appears to be running under ssh. It is not recommended to perform a
↪upgrade over ssh currently because in case of failure it is harder to recover.
If you continue, an additional ssh daemon will be started at port '1022'.
Do you want to continue?
Continue [yN]
```

  - Enter **y** to continue.
  - An additional sshd daemon is started and the following message is displayed:

```
Starting additional ``sshd`` To make recovery in case of failure easier, an
additional sshd will be started on port '1022'. If anything goes wrong
with the running ssh you can still connect to the additional one. If you
run a firewall, you may need to temporarily open this port. As this is
potentially dangerous it's not done automatically. You can open the port
with e.g.: 'iptables -I INPUT -p tcp --dport 1022 -j ACCEPT' To continue
please press [ENTER]
```

  - Press **Enter**.
- If you are warned that third-party sources are disabled:

```
Third party sources disabled
Some third party entries in your sources.list were disabled. You can re-enable
↪them after the upgrade with the 'software-properties' tool or your package
↪manager.
To continue please press **ENTER**
```

Canonical and DGX repositories are preserved for the upgrade, but any other repositories, for example, Google Chrome or VSCode, will be disabled. After the upgrade, you must manually re-enable any third-party sources that you want to keep.

- Press **Enter**.

- You are asked to confirm that you want to start the upgrade.

```
Do you want to start the upgrade?
Installing the upgrade can take several hours. Once the download has finished,
↪the process cannot be canceled.
Continue [yN] Details [d]
```

  - Press **Enter**.

- **(DGX Station only)** In response to the warning that lock screen is disabled, press **Enter** to continue. **Do not** press **Ctrl+C** to respond to this warning, because pressing **Ctrl+C** terminates the upgrade process.

- If you are prompted to confirm that you want to remove obsolete packages, select one of the options:

```
Remove obsolete packages?
371 packages are going to be removed. Removing the packages can take several
↪hours.
Continue [yN]   Details [d]

- Determine whether to remove obsolete packages and continue with the
  upgrade.

  - Review the list of packages that will be removed.

     To identify obsolete DGX OS Desktop packages, see the lists of obsolete
     packages in the `DGX OS Desktop Release
     Notes <https://docs.nvidia.com/dgx/dgx-os-desktop-release-notes/index.html>`_
↪_
     for all releases after your current release.

  - If the list contains only packages that you want to remove, enter
     **y** to continue with the upgrade.
```

  - Enter **y** to accept the recommended changes, **n** (default) for no, or **d** for more details.

7.4.4. Verifying the Upgrade

Here is are steps to verify your upgrade.

1. Confirm the Linux kernel version.

   For example, when you upgrade to DGX OS 6.0, the Linux kernel version is at least `5.15.0-1023-nvidia`.

2. For the minimum Linux kernel version of the release to which you are upgrading, refer to the release notes for that release.

3. Confirm the NVIDIA Graphics Drivers for Linux version.

   ```
   nvidia-smi
   ```

   For example, for an upgrade to DGX OS 6.0, the NVIDIA Graphics Drivers for Linux version is at least 525.105.17:

   ```
   Thu Apr 27 17:00:38 2023
   +-----------------------------------------------------------------------------+
   | NVIDIA-SMI 525.105.17   Driver Version: 525.105.17   CUDA Version: 12.0     |
   |-------------------------------+----------------------+----------------------+
   ```

7.4.5. Recovering from an Interrupted or Failed Update

If the script is interrupted during the update, because of a loss of power or loss of network connection, depending on the issue, you need to restore power or restore the network connection.

If the system encounters a kernel panic after you restore power and reboot the DGX system, you cannot perform the over-the-network update. You need to reinstall DGX OS 6 with the latest image instead. See *Reimaging the System*.

This section provides information about how to install the DGX OS for instructions and complete the network update.

If you can successfully return to the Linux command line, complete the following steps.

1. Reconfigure the packages.

   ```
   dpkg -a --configure
   ```

2. Fix the broken package installs.

   ```
   apt -f install -y
   ```

3. Determine where the `release-upgrader` was extracted.

   ```
   /tmp/ubuntu-release-upgrader-<random-string>
   ```

4. Start a bash shell, go to the upgrader, and configure.

   ```
   sudo bash
   ```

   ```
   cd /tmp/ubuntu-release-upgrader-<random-string>
   ```

```
RELEASE_UPGRADER_ALLOW_THIRD_PARTY=1 ./jammy --frontend=DistUpgradeViewText
```

Do not reboot at this time.

5. Issue the following command and reboot.

```
bash /usr/bin/nvidia-post-release-upgrade
```

```
reboot
```

7.5. Performing Package Upgrades

NVIDIA and Canonical provide updates to the OS in the form of updated software packages between releases with security mitigations and bug fixes. You should evaluate the available updates in regular intervals and update the system that is based on the threat level.

7.5.1. Enabling Extended Security Maintenance Upgrades

This section provides information about Ubuntu's Extended Security Updates (ESM).

As a DGX OS customer, you are entitled to Extended Security Updates from the Ubuntu Universe repository.

You may see the following Ubuntu Pro message from `ubuntu-advantage-tools` during an `apt upgrade` if security updates are available for packages from the Ubuntu Universe repository:

```
Get more security updates through Ubuntu Pro with 'esm-apps' enabled.
Learn more about Ubuntu Pro at https://ubuntu.com/pro.
```

In addition, DGX users will also get the following NVIDIA message:

```
Your DGX contract entitles you to Extended Security Maintenance updates
for additional packages in the Ubuntu repository. Please
contact NVIDIA Support to get your key to enable this capability."
```

After contacting *NVIDIA Enterprise Support* to obtain an Ubuntu Pro token, you can use the token with the following command to enable Extended Security Maintenance updates:

```
sudo pro attach XXXXX
```

Ubuntu Pro subscription can be checked with the sudo pro status command:

```
sudo pro status
```

7.5.2. Performing Package Upgrades Using the CLI

You should evaluate the available updates in regular intervals and update the system based on the threat level:

- Refer to the Ubuntu Wiki Upgrades for more information about upgrades available for Ubuntu.
- For a list of the known Common Vulnerabilities and Exposures (CVEs), including those that can be resolved by updating the DGX OS software, refer to the Ubuntu Security Notices

If updates are available, you can obtain upgraded packages by completing the following steps:

1. Update the internal database with the list of available packages and their versions.

   ```
   sudo apt update
   ```

2. Review the packages that will be upgraded.

   ```
   sudo apt full-upgrade -s
   ```

   To prevent an application from being upgraded, you can instruct the Ubuntu package manager to “hold packages”. Refer to Holding Packages for more information.

   Note: Holding packages should only be used in extreme rare cases as it can disrupt package dependencies.

3. Upgrade to the latest version.

   ```
   sudo apt full-upgrade
   ```

   When prompted to resolve an issue, answer any questions that appear. Most questions require a **Yes** or **No** response.

   - When prompted to select which the GRUB configuration to use, select the current one on the system.
   - When prompted to select the GRUB install devices, keep the default selection.
   - The other questions will depend on what other packages were installed before the update, and how those packages interact with the update.
   - If a message appears that indicates that the `nvidia-docker.service` failed to start, you can disregard it and continue with the next step. The service will start at that time.

4. When the upgrade is complete, reboot the system.

   ```
   sudo reboot
   ```

Note: Upgrades to the NVIDIA Graphics Drivers for Linux requires a restart to complete the kernel upgrade. If you upgrade the NVIDIA Graphics Drivers for Linux without restarting the DGX system, when you run the `nvidia-smi` command, an error message is displayed.

```
nvidia-smi
Failed to initialize NVML: Driver/library version mismatch
```

7.5.3. Managing Software Upgrades on DGX Station

This section provides information about managing upgrades between DGX OS releases by using a GUI tool on DGX Station.

7.5.4. Performing Package Upgrades Using the GUI

You can use the graphical Software Updater application to manage package upgrades on the *DGX Station*.

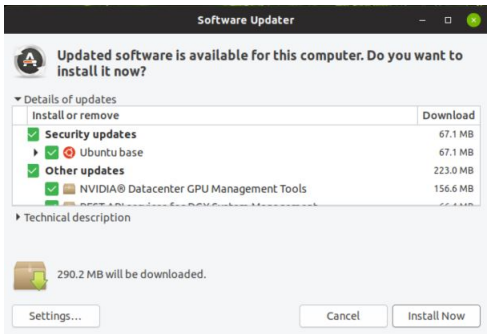
Ensure that you are logged in to your Ubuntu desktop on the **DGX Station** as an administrator user.

1. Press the **Super** key.

   This key is usually found on to the **Alt** key. Refer to What is the Super key? for more information.

   - If you are using a Windows keyboard, the Super key usually has a Windows logo on it, and it is sometimes called the Windows key or system key.
   - If you are using an Apple keyboard, this key is known as the Apple key.

2. In the search bar, type `Software Updater`
3. Open the **Software Updater**, review the available updates, and click **[Install Now]**.

   Screen capture showing the software updater window.

   - If no updates are available, the *Software Updater* informs you that your software is up to date.
   - If an update requires the removal of obsolete packages, you will be warned that not all updates can be installed.

   To continue with the update, complete the following steps:

   a. Click **[Partial Upgrade]**.
   b. Review the list of packages that will be removed. To identify obsolete **DGX Station** packages, see the lists of obsolete packages in the DGX OS Desktop Release Notes for all releases after your current release.
   c. If the list contains only packages that you want to remove, click **[Start Upgrade]**.

4. When prompted to authenticate, type your password into the **[Password]** field and click **[Authenticate]**.
5. When the update is complete, restart **DGX Station**.

Restart the system even if you are not prompted to restart it to complete the updates. Any update to the NVIDIA Graphics Drivers for Linux requires a restart. If you update the NVIDIA Graphics Drivers for Linux without restarting the **DGX Station**, running the `nvidia-smi` command displays an error message.

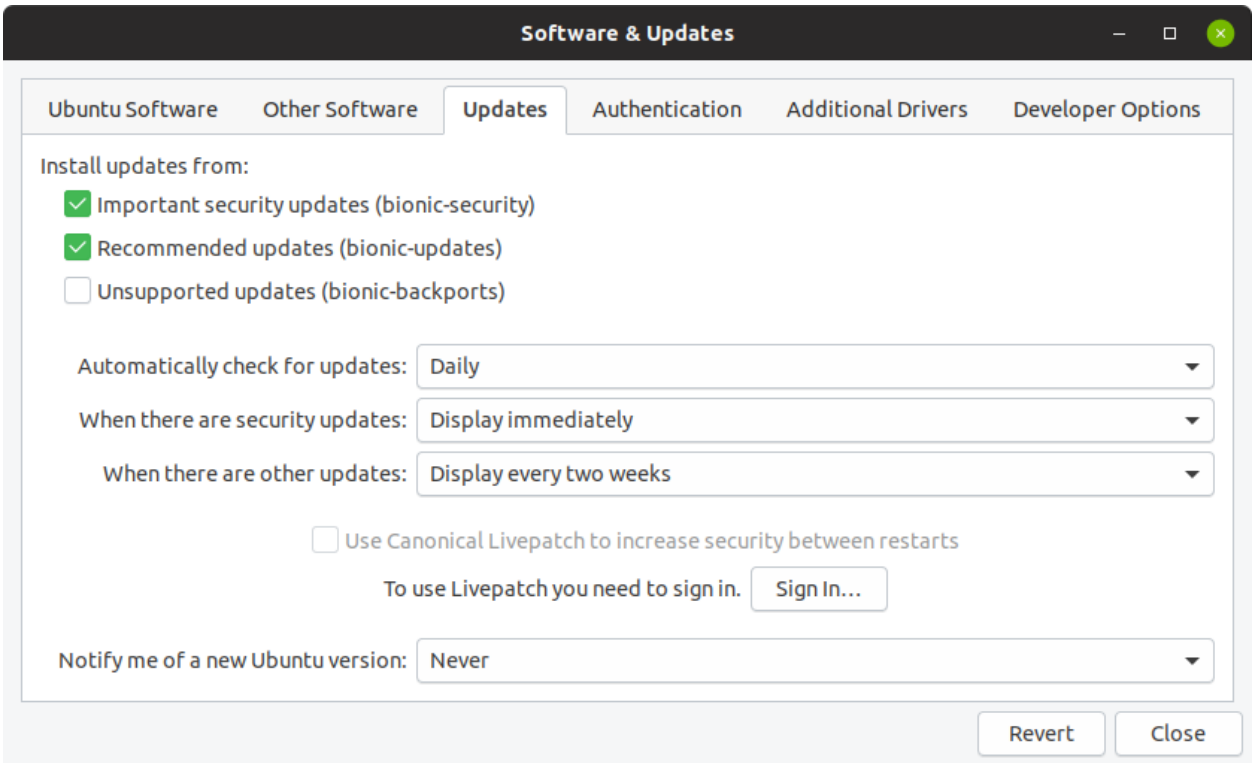
```
nvidia-smi
Failed to initialize NVML: Driver/library version mismatch
```

7.5.5. Checking for Updates to **DGX Station** Software

In **Software & Updates**, you can change your settings to automatically check for package updates and to configure updates from the Ubuntu software repositories. You can also configure your **DGX Station** to notify you of important security updates more frequently than other updates.

In the following example, the **DGX Station** is configured to check for updates daily, to display important security updates immediately, and to display other updates every two weeks.

Screen capture showing the options in the Updates tab of Ubuntu Software & Updates window to check for updates daily, to display important security updates immediately, and to display other updates every two weeks.



Chapter 8. System Configurations

This section provides information about less common configuration options once a system has been installed.

Refer also to *DGX OS Connectivity Requirements* for a list of network ports used by various services.

8.1. Network Configuration

This section provides information about you can configure the network in your DGX system.

8.1.1. Configuring Network Proxies

If your network needs to use a proxy server, you need to set up configuration files to ensure the DGX system communicates through the proxy.

8.1.2. For the OS and Most Applications

Here is some information about configuring the network for the OS and other applications.

Edit the `/etc/environment` file and add the following proxy addresses to the file, below the PATH line.

```
http_proxy="http://<username>:<password>@<host>:<port>/"
ftp_proxy="ftp://<username>:<password>@<host>:<port>/"
https_proxy="https://<username>:<password>@<host>:<port>/"
no_proxy="localhost,127.0.0.1,localaddress,.localdomain.com"
HTTP_PROXY="http://<username>:<password>@<host>:<port>/"
FTP_PROXY="ftp://<username>:<password>@<host>:<port>/"
HTTPS_PROXY="https://<username>:<password>@<host>:<port>/"
NO_PROXY="localhost,127.0.0.1,localaddress,.localdomain.com"
```

Where `username` and `password` are optional.

For example, for the HTTP proxy (both, upper and lower case versions must be changed):

```
http_proxy="http://myproxy.server.com:8080/"
HTTP_PROXY="http://myproxy.server.com:8080/"
```

8.1.3. For the apt Package Manager

Here is some information about configuring the network for the apt package manager.

Edit or create the `/etc/apt/apt.conf.d/myproxy` proxy configuration file and include the following lines:

```
Acquire::http::proxy "http://<username>:<password>@<host>:<port>/";
Acquire::ftp::proxy "ftp://<username>:<password>@<host>:<port>/";
Acquire::https::proxy "https://<username>:<password>@<host>:<port>/";
```

For example:

```
Acquire::http::proxy "http://myproxy.server.com:8080/";
Acquire::ftp::proxy "ftp://myproxy.server.com:8080>/";
Acquire::https::proxy "https://myproxy.server.com:8080/";
```

8.2. Configuring ConnectX from InfiniBand to Ethernet

Many DGX Systems are equipped with NVIDIA ConnectX network controllers and are typically used for cluster communications. By default, the controllers are configured as InfiniBand ports. Optionally, you can configure the ports for Ethernet.

Before or after you reconfigure the port, make sure that the network switch that is connected to the port is also reconfigured to Ethernet or that the port is connected to a different switch that is configured for Ethernet.

The code samples in the following sections show the `mlxconfig` command. The `mlxconfig` command applies to hosts that use the MLNX_OFED drivers. If your host uses the Inbox OFED drivers, then substitute the `mstconfig` command.

You can determine if your host uses the MLNX_OFED drivers by running the `sudo nvidia-manage-ofed.py -s` command. If the output indicates package names beneath the `Mellanox OFED Packages Installed:` field, then the MLNX_OFED drivers are installed.

8.2.1. Determining the Current Port Configuration

Perform the following steps to determine the current configuration for the port.

- Query the devices:

  ```
  sudo mlxconfig -e query | egrep -e Device\|LINK_TYPE
  ```

  The following example shows the output for one of the port devices on an NVIDIA DGX 100 System. The output shows the device path and the default, current, and next boot configuration that are all set to `IB(1)`.

```
Device #9:
Device type:    ConnectX6
Device:         0000:e1:00.0
*        LINK_TYPE_P1                  IB(1)           IB(1)           IB(1)
*        LINK_TYPE_P2                  IB(1)           IB(1)           IB(1)
```

- IB(1) indicates the port is configured for InfiniBand.
- ETH(2) indicates the port is configured for Ethernet.

Determine the device path bus numbers for the slot number of the port that you want to configure. Refer to the following documents for more information:

- DGX H100 Network Ports in the *NVIDIA DGX H100 System User Guide*.
- DGX A100 Network Ports in the *NVIDIA DGX A100 System User Guide*.

8.2.2. Configuring the Port

1. Use the `mlxconfig` command with the `set LINK_TYPE_P<x>` argument for each port you want to configure.

   The following sample command sets port 1 of the controller with PCI ID `e1:00.0` to Ethernet (2):

   ```
   sudo mlxconfig -y -d e1:00.0 set LINK_TYPE_P1=2
   ```

   The following example output is from an NVIDIA DGX A100 System.

   ```
   Device #1:
   ----------

   Device type:    ConnectX6
   Name:           MCX653106A-HDA_Ax
   Description:    ConnectX-6 VPI adapter card; HDR IB (200Gb/s) and 200GbE; dual-
   ↪port QSFP56; PCIe4.0 x16; tall bracket; ROHS R6
   Device:         e1:00.0

   Configurations:                                          Next Boot       New
           LINK_TYPE_P1                                     ETH(2)          IB(1)

    Apply new Configuration? (y/n) [n] : y
   Applying... Done!
   -I- Please reboot machine to load new configurations.
   ```

   Here is an example that sets port 2 to Ethernet:

   ```
   sudo mlxconfig -y -d e1:00.0 set LINK_TYPE_P2=2
   ```

2. (Optional) Run `mlxconfig` again to confirm the change:

   ```
   sudo mlxconfig -e query |egrep -e Device\|LINK_TYPE
   ```

   In the following output, port `LINK_TYPE_2` is set to `ETH(2)` for the next boot. The output shows the device path and the default, current, and next boot configuration.

```
...
Device #9:
Device type:    ConnectX6
Device:         0000:e1:00.0
*        LINK_TYPE_P1                  IB(1)           IB(1)           IB(1)
*        LINK_TYPE_P2                  IB(1)           IB(1)           ETH(2)
```

3. Perform an AC power cycle on the system for the change to take effect.

   Wait for the operating system to boot.

8.2.3. Related Information for Configuring ConnectX from InfiniBand to Ethernet

- Refer to *Inbox OFED vs Mellanox OFED Use Cases* for the command names that apply to the Inbox OFED drivers and the command names that apply to the MLNX_OFED drivers.

8.3. Docker Configuration

To ensure that Docker can access the NGC container registry through a proxy, Docker uses environment variables.

For best practice recommendations on configuring proxy environment variables for Docker, refer to Control Docker with systemd.

8.3.1. Preparing the DGX System to be Used With Docker

Some initial setup of the DGX system is required to ensure that users have the required privileges to run Docker containers and to prevent IP address conflicts between Docker and the DGX system.

8.3.2. Enabling Users To Run Docker Containers

To prevent the `docker` daemon from running without protection against escalation of privileges, the Docker software requires `sudo` privileges to run containers. Meeting this requirement involves enabling users who will run Docker containers to run commands with `sudo` privileges.

You should ensure that only users whom you trust and who are aware of the potential risks to the DGX system of running commands with `sudo` privileges can run Docker containers.

Before you allow multiple users to run commands with `sudo` privileges, consult your IT department to determine whether you might be violating your organization's security policies. For the security implications of enabling users to run Docker containers, see Docker daemon attack surface.

You can enable users to run the Docker containers in one of the following ways:

- Add each user as an administrator user with `sudo` privileges.

- Add each user as a standard user without `sudo` privileges and then add the user to the `docker` group.

This approach is inherently insecure because any user who can send commands to the docker engine can escalate privilege and run root-user operations.

To add an existing user to the `docker` group, run this command:

```
sudo usermod -aG docker user-login-id
```

Where `` `user-login-id `` is the user login ID of the existing user that you are adding to the *docker* group.

8.3.3. Configuring Docker IP Addresses

To ensure that your DGX system can access the network interfaces for Docker containers, Docker should be configured to use a subnet distinct from other network resources used by the DGX system.

By default, Docker uses the `172.17.0.0/16` subnet. Consult your network administrator to find out which IP addresses are used by your network. If your network does not conflict with the default Docker IP address range, no changes are needed and you can skip this section. However, if your network uses the addresses in this range for the DGX system, you should change the default Docker network addresses.

You can change the default Docker network addresses by modifying the `/etc/docker/daemon.json` file or modifying the `/etc/systemd/system/docker.service.d/docker-override.conf` file. These instructions provide an example of modifying the `/etc/systemd/system/docker.service.d/docker-override.conf` to override the default Docker network addresses.

1. Open the `docker-override.conf` file for editing.

```
sudo vi /etc/systemd/system/docker.service.d/docker-override.conf
[Service]
ExecStart=
ExecStart=/usr/bin/dockerd -H fd:// -s overlay2
LimitMEMLOCK=infinity
LimitSTACK=67108864
```

2. Make the changes indicated in bold below, setting the correct bridge IP address and IP address ranges for your network.

   Consult your IT administrator for the correct addresses.

```
[Service]
ExecStart=
ExecStart=/usr/bin/dockerd -H fd:// -s overlay2 --bip=192.168.127.1/24
--fixed-cidr=192.168.127.128/25
LimitMEMLOCK=infinity
LimitSTACK=67108864
```

3. Save and close the `/etc/systemd/system/docker.service.d/docker-override.conf` file.
4. Reload the `systemctl` daemon.

```
sudo systemctl daemon-reload
```

5. Restart Docker.

```
sudo systemctl restart docker
```

8.3.4. Connectivity Requirements for NGC Containers

To run NVIDIA NGC containers from the NGC container registry, your network must be able to access the following URLs:

- http://archive.ubuntu.com/ubuntu/
- http://security.ubuntu.com/ubuntu/
- http://repo.download.nvidia.com/baseos/ubuntu/jammy/x86_64/ (You can access this URL only by using `apt-get` but not in the browser)
- https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86_64/
- https://nvcr.io/

To verify connection to `nvcr.io`, run

```
wget https://nvcr.io/v2
```

You should see connecting verification followed by a 401 error:

```
--2018-08-01 19:42:58-- https://nvcr.io/v2
Resolving nvcr.io (nvcr.io) --> 52.8.131.152, 52.9.8.8
Connecting to nvcr.io (nvcr.io)|52.8.131.152|:443. --> connected.
HTTP request sent, awaiting response. --> 401 Unauthorized
```

8.3.5. Configuring Static IP Addresses for the Network Ports

Here are the steps to configure static IP addresses for network ports.

During the initial boot set up process for your DGX system, one of the steps was to configure static IP addresses for a network interface. If you did not configure the addresses at that time, you can configure the static IP addresses from the Ubuntu command line using the following instructions.

Note: If you are connecting to the DGX console remotely, connect by using the BMC remote console. If you connect using SSH, your connection will be lost when you complete the final step. Also, if you encounter issues with the configuration file, the BMC connection will help with troubleshooting.

If you cannot remotely access the DGX system, connect a display with a 1440x900 or lower resolution, and a keyboard directly to the DGX system.

1. Determine the port designation that you want to configure, based on the physical Ethernet port that you have connected to your network.

   If your network needs to use a proxy server, you need to set up configuration files to ensure the DGX system communicates through the proxy.

   Refer to *Configuring Network Proxies* for the port designation of the connection that you want to configure.

2. Edit the network configuration YAML file, `/etc/netplan/01-netcfg.yaml`, and make the following edits.

   Note: Ensure that your file is identical to the following sample and use spaces and not tabs.

```
# This file describes the network interfaces available on your system
# For more information, see netplan(5).

network:
  version: 2
  renderer: networkd
  ethernets:
    enp226s0:
      dhcp4: no
      addresses:
        - 10.10.10.2/24
      routes:
        - to: default
          via: 10.10.10.1
      nameservers:
        addresses: [ 8.8.8.8 ]
```

   Consult your network administrator for your site-specific values such as network, gateway, and nameserver addresses. Replace `enp226s0` with the designations that you determined in the preceding step.

3. Save the file.
4. Apply the changes.

```
sudo netplan apply
```

Note: If you are not returned to the command line prompt after a information, see Changes, errors, and bugs in the *Ubuntu Server Guide*.

8.4. Managing CPU Mitigations

DGX OS software includes security updates to mitigate CPU speculative side-channel vulnerabilities. These mitigations can decrease the performance of deep learning and machine learning workloads.

If your DGX system installation incorporates other measures to mitigate these vulnerabilities, such as measures at the cluster level, you can disable the CPU mitigations for individual DGX nodes and increase performance.

8.4.1. Determining the CPU Mitigation State of the DGX System

Here is information about how you can determine the CPU mitigation state of your DGX system.

If you do not know whether CPU mitigations are enabled or disabled, issue the following.

```
cat /sys/devices/system/cpu/vulnerabilities/*
```

CPU mitigations are **enabled** when the output consists of multiple lines prefixed with `Mitigation:`.

For example:

```
KVM: Mitigation: Split huge pages
Mitigation: PTE Inversion; VMX: conditional cache flushes, SMT vulnerable Mitigation:
↪Clear CPU buffers; SMT vulnerable
Mitigation: PTI
Mitigation: Speculative Store Bypass disabled via prctl and seccomp Mitigation:
↪usercopy/swapgs barriers and __user pointer sanitization Mitigation: Full generic
↪retpoline, IBPB: conditional, IBRS_FW, STIBP:
conditional, RSB filling
Mitigation: Clear CPU buffers; SMT vulnerable
```

CPU mitigations are **disabled** if the output consists of multiple lines prefixed with `Vulnerable`.

```
KVM: Vulnerable
Mitigation: PTE Inversion; VMX: vulnerable Vulnerable; SMT vulnerable
Vulnerable
Vulnerable
Vulnerable: user pointer sanitization and usercopy barriers only; no swapgs barriers
Vulnerable, IBPB: disabled, STIBP: disabled Vulnerable
```

8.4.2. Disabling CPU Mitigations

Here are the steps to disable CPU mitigations.

> Caution: Performing the following instructions will disable the CPU mitigations provided by the DGX OS software.

1. Install the `nv-mitigations-off` package.

   ```
   sudo apt install nv-mitigations-off -y
   ```

2. Reboot the system.
3. Verify that the CPU mitigations are disabled.

   ```
   cat /sys/devices/system/cpu/vulnerabilities/*
   ```

The output should include several vulnerable lines. See *Determining the CPU Mitigation State of the DGX System* Here is information about how you can determine the CPU mitigation state of your DGX system example output.

8.4.3. Re-enable CPU Mitigations

Here are the steps to enable CPU mitigations again.

1. Remove the `nv-mitigations-off` package.

   ```
   sudo apt purge nv-mitigations-off
   ```

2. Reboot the system.
3. Verify that the CPU mitigations are enabled.

   ```
   cat /sys/devices/system/cpu/vulnerabilities/*
   ```

The output should include several Mitigations lines. See *Determining the CPU Mitigation State of the DGX System* for example output.

8.5. Managing the DGX Crash Dump Feature

This section provides information about managing the DGX Crash Dump feature. You can use the script that is included in the DGX OS to manage this feature.

8.5.1. Using the Script

Here are commands that help you complete the necessary tasks with the script.

- To enable only dmesg crash dumps, run:

  ```
  /usr/sbin/nvidia-kdump-config enable-dmesg-dump
  ```

  This option reserves memory for the crash kernel.
- To enable both dmesg and vmcore crash dumps, run:

  ```
  /usr/sbin/nvidia-kdump-config enable-vmcore-dump
  ```

  This option reserves memory for the crash kernel.
- To disable crash dumps, run:

  ```
  /usr/sbin/nvidia-kdump-config disable
  ```

This option disables the use of `kdump` and ensures that no memory is reserved for the crash kernel.

8.6. Connecting to Serial Over LAN

You can connect to serial over a LAN.

Warning: This applies **only** to systems that have the BMC.

While dumping `vmcore`, the BMC screen console goes blank approximately 11 minutes after the crash dump is started. To view the console output during the crash dump, connect to serial over LAN as follows:

```
ipmitool -I lanplus -H -U -P sol activate
```

8.7. Filesystem Quotas

Here is some information about filesystem quotas.

When running NGC containers you might need to limit the amount of disk space that is used on a filesystem to avoid filling up the partition. Refer to How to Set Filesystem Quotas on Ubuntu 18.04 about how to set filesystem quotas on Ubuntu 18.04 and later.

8.8. Running Workloads on Systems with Mixed Types of GPUs

The DGX Station A100 comes equipped with four high performance NVIDIA A100 GPUs and one DGX Display GPU. The NVIDIA A100 GPU is used to run high performance and AI workloads, and the DGX Display card is used to drive a high-quality display on a monitor.

When running applications on this system, it is important to identify the best method to launch applications and workloads to make sure the high performance NVIDIA A100 GPUs are used. You can achieve this in one of the following ways:

- *Running with Docker Containers*
- *Running on Bare Metal*
- *Using Multi-Instance GPUs*

When you log into the system and check which GPUs are available, you find the following:

```
nvidia-smi -L
GPU 0: Graphics Device (UUID: GPU-269d95f8-328a-08a7-5985-ab09e6e2b751)
GPU 1: Graphics Device (UUID: GPU-0f2dff15-7c85-4320-da52-d3d54755d182)
GPU 2: Graphics Device (UUID: GPU-dc598de6-dd4d-2f43-549f-f7b4847865a5)
GPU 3: DGX Display (UUID: GPU-91b9d8c8-e2b9-6264-99e0-b47351964c52)
GPU 4: Graphics Device (UUID: GPU-e32263f2-ae07-f1db-37dc-17d1169b09bf)
```

A total of five GPUs are listed by `nvidia-smi`. This is because `nvidia-smi` is including the DGX Display GPU that is used to drive the monitor and high-quality graphics output.

When running an application or workload, the DGX Display GPU can get in the way because it does not have direct NVlink connectivity, sufficient memory, or the performance characteristics of the NVIDIA A100 GPUs that are installed on the system. As a result you should ensure that the correct GPUs are being used.

8.8.1. Running with Docker Containers

On the DGX OS, because Docker has already been configured to identify the high performance NVIDIA A100 GPUs and assign the GPUs to the container, this method is the simplest.

A simple test is to run a small container with the [–gpus all] flag in the command and once in the container that is running `nvidia-smi`. The output shows that only the high-performance GPUs are available to the container:

```
docker run --gpus all --rm -it ubuntu nvidia-smi -L
GPU 0: Graphics Device (UUID: GPU-269d95f8-328a-08a7-5985-ab09e6e2b751)
GPU 1: Graphics Device (UUID: GPU-0f2dff15-7c85-4320-da52-d3d54755d182)
GPU 2: Graphics Device (UUID: GPU-dc598de6-dd4d-2f43-549f-f7b4847865a5)
GPU 3: Graphics Device (UUID: GPU-e32263f2-ae07-f1db-37dc-17d1169b09bf)
```

This step will also work when the `--gpus n` flag is used, where n can be `1, 2, 3, or 4`. These values represent the number of GPUs that should be assigned to that container. For example:

```
docker run --gpus 2 --rm -it ubuntu nvidia-smi -L
GPU 0: Graphics Device (UUID: GPU-269d95f8-328a-08a7-5985-ab09e6e2b751)
GPU 1: Graphics Device (UUID: GPU-0f2dff15-7c85-4320-da52-d3d54755d182)
```

In this example, Docker selected the first two GPUs to run the container, but if the `device` option is used, you can specify which GPUs to use:

```
docker run --gpus '"device=GPU-dc598de6-dd4d-2f43-549f-f7b4847865a5,GPU-e32263f2-ae07-
↪f1db-37dc-17d1169b09bf"' --rm -it ubuntu nvidia-smi -L
GPU 0: Graphics Device (UUID: GPU-dc598de6-dd4d-2f43-549f-f7b4847865a5)
GPU 1: Graphics Device (UUID: GPU-e32263f2-ae07-f1db-37dc-17d1169b09bf)
```

In this example, the two GPUs that were not used earlier are now assigned to run on the container.

8.8.2. Running on Bare Metal

To run applications by using the four high performance GPUs, the `CUDA_VISIBLE_DEVICES` variable must be specified **before** you run the application.

Note: This method does not use containers.

CUDA orders the GPUs by performance, so `GPU 0` will be the highest performing GPU, and the last GPU will be the slowest GPU.

Warning: `CUDA_DEVICE_ORDER` variable is set to `PCI_BUS_ID`, this ordering will be overridden.

In the following example, a CUDA application that comes with CUDA samples is run. In the output, GPU 0 is the fastest in a DGX Station A100, and GPU 4 (DGX Display GPU) is the slowest:

```
sudo apt install cuda-samples-11-2
```

```
cd /usr/local/cuda-11.2/samples/1_Utilities/p2pBandwidthLatencyTest
```

```
sudo make
/usr/local/cuda/bin/nvcc -ccbin g++ -I../../common/inc  -m64    --threads
0 -gencode arch=compute_35,code=sm_35 -gencode arch=compute_37,code=sm_37
-gencode arch=compute_50,code=sm_50 -gencode arch=compute_52,code=sm_52
-gencode arch=compute_60,code=sm_60 -gencode arch=compute_61,code=sm_61
-gencode arch=compute_70,code=sm_70 -gencode arch=compute_75,code=sm_75
-gencode arch=compute_80,code=sm_80 -gencode arch=compute_86,code=sm_86
-gencode arch=compute_86,code=compute_86 -o p2pBandwidthLatencyTest.o -c
↪p2pBandwidthLatencyTest.cu
nvcc warning : The 'compute_35', 'compute_37', 'compute_50', 'sm_35', 'sm_37' and 'sm_
↪50' architectures are deprecated, and may be removed in a future release (Use -Wno-
↪deprecated-gpu-targets to suppress warning).
/usr/local/cuda/bin/nvcc -ccbin g++   -m64
-gencode arch=compute_35,code=sm_35 -gencode arch=compute_37,code=sm_37
-gencode arch=compute_50,code=sm_50 -gencode arch=compute_52,code=sm_52
-gencode arch=compute_60,code=sm_60 -gencode arch=compute_61,code=sm_61
-gencode arch=compute_70,code=sm_70 -gencode arch=compute_75,code=sm_75
-gencode arch=compute_80,code=sm_80 -gencode arch=compute_86,code=sm_86
-gencode arch=compute_86,code=compute_86 -o p2pBandwidthLatencyTest
↪p2pBandwidthLatencyTest.o
nvcc warning : The 'compute_35', 'compute_37', 'compute_50', 'sm_35', 'sm_37' and 'sm_
↪50' architectures are deprecated, and may be removed in a future release (Use -Wno-
↪deprecated-gpu-targets to suppress warning).
mkdir -p ../../bin/x86_64/linux/release
cp p2pBandwidthLatencyTest ../../bin/x86_64/linux/release
lab@ro-dvt-058-80gb:/usr/local/cuda-11.2/samples/1_Utilities/p2pBandwidthLatencyTest
↪cd /usr/local/cuda-11.2/samples/bin/x86_64/linux/release
lab@ro-dvt-058-80gb:/usr/local/cuda-11.2/samples/bin/x86_64/linux/release ./
↪p2pBandwidthLatencyTest
[P2P (Peer-to-Peer) GPU Bandwidth Latency Test]
Device: 0, Graphics Device, pciBusID: 1, pciDeviceID: 0, pciDomainID:0
Device: 1, Graphics Device, pciBusID: 47, pciDeviceID: 0, pciDomainID:0
Device: 2, Graphics Device, pciBusID: 81, pciDeviceID: 0, pciDomainID:0
Device: 3, Graphics Device, pciBusID: c2, pciDeviceID: 0, pciDomainID:0
Device: 4, DGX Display, pciBusID: c1, pciDeviceID: 0, pciDomainID:0
Device=0 CAN Access Peer Device=1
Device=0 CAN Access Peer Device=2
Device=0 CAN Access Peer Device=3
Device=0 CANNOT Access Peer Device=4
Device=1 CAN Access Peer Device=0
Device=1 CAN Access Peer Device=2
Device=1 CAN Access Peer Device=3
Device=1 CANNOT Access Peer Device=4
Device=2 CAN Access Peer Device=0
Device=2 CAN Access Peer Device=1
Device=2 CAN Access Peer Device=3
Device=2 CANNOT Access Peer Device=4
Device=3 CAN Access Peer Device=0
Device=3 CAN Access Peer Device=1
Device=3 CAN Access Peer Device=2
```

(continues on next page)

(continued from previous page)

```
Device=3 CANNOT Access Peer Device=4
Device=4 CANNOT Access Peer Device=0
Device=4 CANNOT Access Peer Device=1
Device=4 CANNOT Access Peer Device=2
Device=4 CANNOT Access Peer Device=3
```

Note: In case a device doesn't have P2P access to other one, it falls back to normal memcopy procedure. So you can see lesser Bandwidth (GB/s) and unstable Latency (us) in those cases.

```
P2P Connectivity Matrix
     D\D     0     1     2     3     4
     0       1     1     1     1     0
     1       1     1     1     1     0
     2       1     1     1     1     0
     3       1     1     1     1     0
     4       0     0     0     0     1
Unidirectional P2P=Disabled Bandwidth Matrix (GB/s)
   D\D     0      1      2      3      4
     0 1323.03  15.71  15.37  16.81  12.04
     1  16.38 1355.16  15.47  15.81  11.93
     2  16.25  15.85 1350.48  15.87  12.06
     3  16.14  15.71  16.80 1568.78  11.75
     4  12.61  12.47  12.68  12.55 140.26
Unidirectional P2P=Enabled Bandwidth (P2P Writes) Matrix (GB/s)
   D\D     0      1      2      3      4
     0 1570.35  93.30  93.59  93.48  12.07
     1  93.26 1583.08  93.55  93.53  11.93
     2  93.44  93.58 1584.69  93.34  12.05
     3  93.51  93.55  93.39 1586.29  11.79
     4  12.68  12.54  12.75  12.51 140.26
Bidirectional P2P=Disabled Bandwidth Matrix (GB/s)
   D\D     0      1      2      3      4
     0 1588.71  19.60  19.26  19.73  16.53
     1  19.59 1582.28  19.85  19.13  16.43
     2  19.53  19.39 1583.88  19.61  16.58
     3  19.51  19.11  19.58 1592.76  15.90
     4  16.36  16.31  16.39  15.80 139.42
Bidirectional P2P=Enabled Bandwidth Matrix (GB/s)
   D\D     0      1      2      3      4
     0 1590.33 184.91 185.37 185.45  16.46
     1 185.04 1587.10 185.19 185.21  16.37
     2 185.15 185.54 1516.25 184.71  16.47
     3 185.55 185.32 184.86 1589.52  15.71
     4  16.26  16.28  16.16  15.69 139.43
P2P=Disabled Latency Matrix (us)
   GPU     0      1      2      3      4
     0   3.53  21.60  22.22  21.38  12.46
     1  21.61   2.62  21.55  21.65  12.34
     2  21.57  21.54   2.61  21.55  12.40
     3  21.57  21.54  21.58   2.51  13.00
     4  13.93  12.41  21.42  21.58   1.14

   CPU     0      1      2      3      4
     0   4.26  11.81  13.11  12.00  11.80
```

(continues on next page)

(continued from previous page)

```
     1  11.98   4.11  11.85  12.19  11.89
     2  12.07  11.72   4.19  11.82  12.49
     3  12.14  11.51  11.85   4.13  12.04
     4  12.21  11.83  12.11  11.78   4.02
P2P=Enabled Latency (P2P Writes) Matrix (us)
   GPU     0      1      2      3      4
     0   3.79   3.34   3.34   3.37  13.85
     1   2.53   2.62   2.54   2.52  12.36
     2   2.55   2.55   2.61   2.56  12.34
     3   2.58   2.51   2.51   2.53  14.39
     4  19.77  12.32  14.75  21.60   1.13

   CPU     0      1      2      3      4
     0   4.27   3.63   3.65   3.59  13.15
     1   3.62   4.22   3.61   3.62  11.96
     2   3.81   3.71   4.35   3.73  12.15
     3   3.64   3.61   3.61   4.22  12.06
     4  12.32  11.92  13.30  12.03   4.05
```

Note: The CUDA Samples are not meant for performance measurements. Results may vary when GPU Boost is enabled.

The example above shows the peer-to-peer bandwidth and latency test across all five GPUs, including the DGX Display GPU. The application also shows that there is no peer-to-peer connectivity between any GPU and GPU 4. This indicates that GPU 4 should not be used for high-performance workloads.

Run the example one more time by using the `CUDA_VISIBLE_DEVICES` variable, which limits the number of GPUs that the application can see.

Note: All GPUs can communicate with all other peer devices.

```
 CUDA_VISIBLE_DEVICES=0,1,2,3 ./p2pBandwidthLatencyTest
[P2P (Peer-to-Peer) GPU Bandwidth Latency Test]
Device: 0, Graphics Device, pciBusID: 1, pciDeviceID: 0, pciDomainID:0
Device: 1, Graphics Device, pciBusID: 47, pciDeviceID: 0, pciDomainID:0
Device: 2, Graphics Device, pciBusID: 81, pciDeviceID: 0, pciDomainID:0
Device: 3, Graphics Device, pciBusID: c2, pciDeviceID: 0, pciDomainID:0
Device=0 CAN Access Peer Device=1
Device=0 CAN Access Peer Device=2
Device=0 CAN Access Peer Device=3
Device=1 CAN Access Peer Device=0
Device=1 CAN Access Peer Device=2
Device=1 CAN Access Peer Device=3
Device=2 CAN Access Peer Device=0
Device=2 CAN Access Peer Device=1
Device=2 CAN Access Peer Device=3
Device=3 CAN Access Peer Device=0
Device=3 CAN Access Peer Device=1
Device=3 CAN Access Peer Device=2
```

Note: In case a device doesn't have P2P access to other one, it falls back to normal memcopy proce-

dure. So you can see lesser Bandwidth (GB/s) and unstable Latency (us) in those cases.

```
P2P Connectivity Matrix
     D\D     0     1     2     3
     0       1     1     1     1
     1       1     1     1     1
     2       1     1     1     1
     3       1     1     1     1
Unidirectional P2P=Disabled Bandwidth Matrix (GB/s)
   D\D     0      1      2      3
     0 1324.15  15.54  15.62  15.47
     1  16.55 1353.99  15.52  16.23
     2  15.87  17.26 1408.93  15.91
     3  16.33  17.31  18.22 1564.06
Unidirectional P2P=Enabled Bandwidth (P2P Writes) Matrix (GB/s)
   D\D     0      1      2      3
     0 1498.08  93.30  93.53  93.48
     1  93.32 1583.08  93.54  93.52
     2  93.55  93.60 1583.08  93.36
     3  93.49  93.55  93.28 1576.69
Bidirectional P2P=Disabled Bandwidth Matrix (GB/s)
   D\D     0      1      2      3
     0 1583.08  19.92  20.47  19.97
     1  20.74 1586.29  20.06  20.22
     2  20.08  20.59 1590.33  20.01
     3  20.44  19.92  20.60 1589.52
Bidirectional P2P=Enabled Bandwidth Matrix (GB/s)
   D\D     0      1      2      3
     0 1592.76 184.88 185.21 185.30
     1 184.99 1589.52 185.19 185.32
     2 185.28 185.30 1585.49 185.01
     3 185.45 185.39 184.84 1587.91
P2P=Disabled Latency Matrix (us)
   GPU     0      1      2      3
     0   2.38  21.56  21.61  21.56
     1  21.70   2.34  21.54  21.56
     2  21.55  21.56   2.41  21.06
     3  21.57  21.34  21.56   2.39

   CPU     0      1      2      3
     0   4.22  11.99  12.71  12.09
     1  11.86   4.09  12.00  11.71
     2  12.52  11.98   4.27  12.24
     3  12.22  11.75  12.19   4.25
P2P=Enabled Latency (P2P Writes) Matrix (us)
   GPU     0      1      2      3
     0   2.32   2.57   2.55   2.59
     1   2.55   2.32   2.59   2.52
     2   2.59   2.56   2.41   2.59
     3   2.57   2.55   2.56   2.40

   CPU     0      1      2      3
     0   4.24   3.57   3.72   3.81
     1   3.68   4.26   3.75   3.63
     2   3.79   3.75   4.34   3.71
     3   3.72   3.64   3.66   4.32
```

Note: The CUDA Samples are not meant for performance measurements. Results may vary when GPU Boost is enabled.

For bare metal applications, the UUID can also be specified in the [CUDA_VISIBLE_DEVICES] variable as shown below:

```
CUDA_VISIBLE_DEVICES=GPU-0f2dff15-7c85-4320-da52-d3d54755d182,GPU-dc598de6-dd4d-2f43-
↪549f-f7b4847865a5 ./p2pBandwidthLatencyTest
```

The GPU specification is longer because of the nature of UUIDs, but this is the most precise way to pin specific GPUs to the application.

8.9. Using Multi-Instance GPUs

Multi-Instance GPUs (MIG) is available on NVIDIA A100 GPUs. If MIG is enabled on the GPUs, and if the GPUs have already been partitioned, then applications can be limited to run on these devices.

This works for both Docker containers and for bare metal using the [CUDA_VISIBLE_DEVICES] as shown in the examples below. For instructions on how to configure and use MIG, refer to the NVIDIA Multi-Instance GPU User Guide.

Identify the MIG instances that will be used. Here is the output from a system that has `GPU 0` partitioned into 7 MIGs:

```
nvidia-smi -L
GPU 0: Graphics Device (UUID: GPU-269d95f8-328a-08a7-5985-ab09e6e2b751)
  MIG 1g.10gb Device 0: (UUID: MIG-GPU-269d95f8-328a-08a7-5985-ab09e6e2b751/7/0)
  MIG 1g.10gb Device 1: (UUID: MIG-GPU-269d95f8-328a-08a7-5985-ab09e6e2b751/8/0)
  MIG 1g.10gb Device 2: (UUID: MIG-GPU-269d95f8-328a-08a7-5985-ab09e6e2b751/9/0)
  MIG 1g.10gb Device 3: (UUID: MIG-GPU-269d95f8-328a-08a7-5985-ab09e6e2b751/11/0)
  MIG 1g.10gb Device 4: (UUID: MIG-GPU-269d95f8-328a-08a7-5985-ab09e6e2b751/12/0)
  MIG 1g.10gb Device 5: (UUID: MIG-GPU-269d95f8-328a-08a7-5985-ab09e6e2b751/13/0)
  MIG 1g.10gb Device 6: (UUID: MIG-GPU-269d95f8-328a-08a7-5985-ab09e6e2b751/14/0)
GPU 1: Graphics Device (UUID: GPU-0f2dff15-7c85-4320-da52-d3d54755d182)
GPU 2: Graphics Device (UUID: GPU-dc598de6-dd4d-2f43-549f-f7b4847865a5)
GPU 3: DGX Display (UUID: GPU-91b9d8c8-e2b9-6264-99e0-b47351964c52)
GPU 4: Graphics Device (UUID: GPU-e32263f2-ae07-f1db-37dc-17d1169b09bf)
```

In Docker, enter the MIG UUID from this output, in which `GPU 0` and `Device 0` have been selected.

If you are running on DGX Station A100, restart the `nv-docker-gpus` and docker system services any time MIG instances are created, destroyed or modified by running the following:

```
sudo systemctl restart nv-docker-gpus; sudo systemctl restart docker
```

`nv-docker-gpus` has to be restarted on DGX Station A100 because this service is used to mask the available GPUs that can be used by Docker. When the GPU architecture changes, the service needs to be refreshed.

```
docker run --gpus '"device=MIG-GPU-269d95f8-328a-08a7-5985-ab09e6e2b751/7/0"' --rm -
↪it ubuntu nvidia-smi -L
GPU 0: Graphics Device (UUID: GPU-269d95f8-328a-08a7-5985-ab09e6e2b751)
  MIG 1g.10gb Device 0: (UUID: MIG-GPU-269d95f8-328a-08a7-5985-ab09e6e2b751/7/0)
```

On bare metal, specify the MIG instances:

Note: This application measures the communication across GPUs, and it is not relevant to read the bandwidth and latency with only one GPU MIG.

The purpose of this example is to illustrate how to use specific GPUs with applications, which is illustrated below.

1. Go to the following directory:

```
cd /usr/local/cuda-11.2/samples/bin/x86_64/linux/release
```

2. Run the `p2pBandwidthLatencyTest`

```
CUDA_VISIBLE_DEVICES=MIG-GPU-269d95f8-328a-08a7-5985-ab09e6e2b751/7/0 ./
↪p2pBandwidthLatencyTest
[P2P (Peer-to-Peer) GPU Bandwidth Latency Test]
Device: 0, Graphics Device MIG 1g.10gb, pciBusID: 1, pciDeviceID: 0, pciDomainID:0
```

Note: In case a device doesn't have P2P access to other one, it falls back to normal memcopy procedure. So you can see lesser Bandwidth (GB/s) and unstable Latency (us) in those cases.

```
P2P Connectivity Matrix
     D\D     0
     0       1
Unidirectional P2P=Disabled Bandwidth Matrix (GB/s)
   D\D     0
     0 176.20
Unidirectional P2P=Enabled Bandwidth (P2P Writes) Matrix (GB/s)
   D\D     0
     0 187.87
Bidirectional P2P=Disabled Bandwidth Matrix (GB/s)
   D\D     0
     0 190.77
Bidirectional P2P=Enabled Bandwidth Matrix (GB/s)
   D\D     0
     0 190.53
P2P=Disabled Latency Matrix (us)
   GPU     0
     0   3.57

   CPU     0
     0   4.07
P2P=Enabled Latency (P2P Writes) Matrix (us)
   GPU     0
     0   3.55

   CPU     0
     0   4.07
```

Note: The CUDA Samples are not meant for performance measurements. Results may vary when GPU Boost is enabled.

8.10. Updating the containerd Override File for MIG configurations

When you add MIG instances, the containerd override file does not automatically get updated, and the new MIG instances that you add will not be added to the allow file. When DGX Station A100 starts, after the nv-docker-gpus service runs, a containerd override file is created in the `/etc/systemd/system/containerd.service.d/ directory`.

Note: This file blocks Docker from using the display GPU on the DGX Station A100.

Here is an example of an override file:

```
[Service]
DeviceAllow=/dev/nvidia1
DeviceAllow=/dev/nvidia2
DeviceAllow=/dev/nvidia3
DeviceAllow=/dev/nvidia4
DeviceAllow=/dev/nvidia-caps/nvidia-cap1
DeviceAllow=/dev/nvidia-caps/nvidia-cap2
DeviceAllow=/dev/nvidiactl
DeviceAllow=/dev/nvidia-modeset
DeviceAllow=/dev/nvidia-uvm
DeviceAllow=/dev/nvidia-uvm-tools
```

The service can only add devices of which it is aware. To ensure that your new MIG instances are added to the allow list, complete the following steps:

1. To refresh the override file, run the following commands:

   ```
   sudo systemctl restart nv-docker-gpus
   ```

   ```
   sudo systemctl restart docker
   ```

2. Verify that your new MIG instances are now allowed in the containers. Here is an example of an updated override file:

   ```
   |       ID ID Dev |                  BAR1-Usage |  SM    Unc| CE ENC DEC OFA JPG
   ↪ |
   |                  |                       |        ECC|                       |
   |==================+=======================+===========+=======================|
   | 0      0   0   0 |      0MiB / 81252MiB |  98      0|  7   0   5   1   1    |
   |                  |      1MiB / 13107... |           |                       |
   +------------------+-----------------------+-----------+-----------------------+
   | Processes:                                                                   |
   | GPU   GI     CI     PID Type    Process name    GPU Memory                   |
   |       ID     ID     Usage                                                    |
   |==============================================================================|
   | No running processes found                                                   |
   ```

8.11. Data Storage Configuration

By default, the DGX system includes several drives in a RAID 0 configuration. These drives are intended for application caching, so you

8.11.1. Using Data Storage for NFS Caching

This section provides information about how you can use data storage for NFS caching.

The DGX systems use `cachefilesd` to manage NFS caching.

- Ensure that you have an NFS server with one or more exports with data that will be accessed by the DGX system
- Ensure that there is network access between the DGX system and the NFS server.

8.11.2. Using cachefilesd

Here are the steps that describe how you can mount the NFS on the DGX system, and how you can cache the NFS by using the DGX SSDs for improved performance.

1. Configure an NFS mount for the DGX system.
   a. Edit the filesystem tables configuration.

      ```
      sudo vi /etc/fstab
      ```

   b. Add a new line for the NFS mount by using the local `/mnt` local mount point.

      ```
      <nfs_server>:<export_path> /mnt nfs rw,noatime,rsize=32768,wsize=32768,nolock,
      ↪tcp,intr,fsc,nofail 0 0
      ```

      Here, `/mnt` is used an example mount point.

      - Contact your Network Administrator for the correct values for `<nfs_server>` and `<export_path>`.
      - The nfs arguments presented here are a list of recommended values based on typical use cases. However, `fsc` must always be included because that argument specifies using FS-Cache

   c. Save the changes.
2. Verify that the NFS server is reachable.

   ```
   ping <nfs_server>
   ```

   Use the server IP address or the server name that was provided by your network administrator.
3. Mount the NFS export.

   ```
   sudo mount /mnt
   ```

   `/mnt` is an example mount point.

4. Verify that caching is enabled.

```
cat /proc/fs/nfsfs/volumes
```

5. In the output, find FSC=yes.

   The NFS will be automatically mounted and cached on the DGX system in subsequent reboot cycles.

8.11.3. Disabling cachefilesd

Here is some information about how to disable `cachefilesd`.

If you do not want to enable `cachefilesd` by running:

1. Stop the `cachefilesd` service:

```
sudo systemctl stop cachefilesd
```

2. Disable the `cachefilesd` service permanently

```
sudo systemctl disable cachefilesd
```

8.11.4. Changing the RAID Configuration for Data Drives

Here is information that describes how to change the RAID configuration for your data drives.

Warning: You **must** have a minimum of 2 drives to complete these tasks. If you have less than 2 drives, you **cannot** complete the tasks.

From the factory, the RAID level of the DGX RAID array is RAID 0. This level provides the maximum storage capacity, but it does not provide redundancy. If one SSD in the array fails, the data that is stored on the array is lost. If you are willing to accept reduced capacity in return for a level of protection against drive failure, you can change the level of the RAID array to RAID 5.

Note: If you change the RAID level from RAID 0 to RAID 5, the total storage capacity of the RAID array is reduced.

Before you change the RAID level of the DGX RAID array, back up the data on the array that you want to preserve. When you change the RAID level of the DGX RAID array, the data that is stored on the array is erased.

You can use the `configure_raid_array.py` custom script, which is installed on the system to change the level of the RAID array without unmounting the RAID volume.

- To change the RAID level to RAID 5, run the following command:

```
sudo configure_raid_array.py -m raid5
```

After you change the RAID level to RAID 5, the RAID array is rebuilt. Although a RAID array that is being rebuilt is online and ready to be used, a check on the health of the DGX system reports the status of the RAID volume as unhealthy. The time required to rebuild the RAID array depends on the workload on the system. For example, on an idle system, the rebuild might be completed in 30 minutes.

- To change the RAID level to RAID 0, run the following command:

```
sudo configure_raid_array.py -m raid0
```

To confirm that the RAID level was changed, run the `lsblk` command. The entry in the TYPE column for each drive in the RAID array indicates the RAID level of the array.

8.12. Running NGC Containers

This section provides information about how to run NGC containers with your DGX system.

8.12.1. Obtaining an NGC Account

Here is some information about how you can obtain an NGC account.

NVIDIA NGC provides simple access to GPU-optimized software for deep learning, machine learning , and high-performance computing (HPC). An NGC account grants you access to these tools and gives you the ability to set up a private registry to manage your customized software.

If you are the organization administrator for your DGX system purchase, work with NVIDIA Enterprise Support to set up an NGC enterprise account. Refer to the NGC Private Registry User Guide for more information about getting an NGC enterprise account.

8.12.2. Running NGC Containers with GPU Support

To obtain the best performance when running NGC containers on DGX systems, you can use one of the following methods to provide GPU support for Docker containers:

- Native GPU support (included in Docker 19.03 and later, installed)
- NVIDIA Container Runtime for Docker

This is in the `nvidia-docker2` package.

The recommended method for DGX OS 6 is native GPU support. To run GPU-enabled containers, run `docker run --gpus`.

Here is an example that uses all GPUs:

```
docker run --gpus all …
```

Here is an example that uses 2 GPUs:

```
docker run --gpus 2 …
```

Here is an example that uses specific GPUs:

```
docker run --gpus '"device=1,2"' ...
```

```
docker run --gpus '"device=UUID-ABCDEF-
```

Refer to Running Containers for more information about running NGC containers on MIG devices.

Chapter 9. Installing the ConnectX-7 Firmware

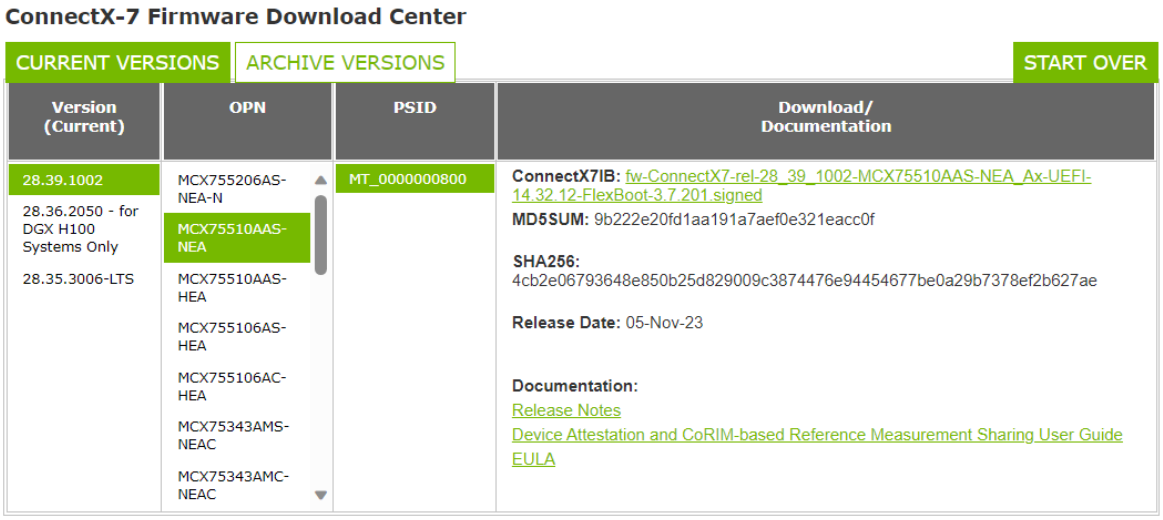
Follow these steps to update firmware for the ConnectX®-7 InfiniBand/Ethernet PCI Express Adapter Cards using the NVIDIA Networking Firmware Downloads Page.

1. Navigate to the NVIDIA Networking Firmware Downloads page.
2. From the **ConnectX Adapter Cards Firmware** table, select version **ConnectX-7** and click the **InfiniBand/Ethernet** network protocol.

ConnectX Adapter Cards Firmware	
Product Line	Network Protocol
ConnectX-7	InfiniBand/Ethernet
ConnectX-6 DE	InfiniBand
ConnectX-6 Lx	Ethernet
ConnectX-6 Dx	Ethernet

   The Firmware for ConnectX®-7 InfiniBand page opens.

3. At the **ConnectX-7 Firmware Download Center** matrix, choose the firmware version and the **OPN** number of the ConnectX-7 adapter card. For example,
   - **Version**: 28.39.1002
   - **OPN**: MCX75510AAS-NEA
4. Click the corresponding **PSID** number, for example, MT_0000000800, to show the firmware information and documentation.



5. Click the **ConnectX7IB** link to download the firmware BIN file.

   For example, `fw-ConnectX7-rel-28_39_1002-MCX75510AAS-NEA_Ax-UEFI-14.32.12-FlexBoot-3.7.201.signed.bin`.

6. After downloading the correct ConnectX-7 firmware, proceed with the installation steps.

Alternatively, you can use the following methods to install the ConnectX-7 adapter card firmware:

- Using the `mstflint` tool

  If you have installed the MTNIC Driver on your machine, you can update the firmware using the `mstflint` tool as described in the mstflint FW Burning Tool README. You can download the `mstflint` tool from the OpenFabrics site at mstflint_SW for Linux.

- Using the NVIDIA Firmware Tools (MFT)

  For details, refer to Updating Firmware for a Single Network Interface Card (NIC).

Chapter 10. Managing Self-Encrypting Drives

The NVIDIA DGX OS software supports the ability to manage self-encrypting drives (SEDs), including setting an Authentication Key for locking and unlocking the drives on NVIDIA DGX H100, DGX A100, DGX A800, DGX Station A100, and DGX-2 systems.

You can manage only the SED data drives. The software cannot be used to manage OS drives even if they are SED-capable.

10.1. Overview

The SED management software is in the `nv-disk-encrypt` package.

The software supports the following configurations:

- NVIDIA DGX H100, DGX A100, DGX A800, DGX Station A100, and DGX-2 systems where all data drives are self-encrypting drives.
- Only SEDs used as data drives are supported.

  The software will not manage SEDs that are OS drives.

The software provides the following functionality:

- Identifies eligible drives on the system.
- Allows you to you assign Authentication Keys (passwords) for each SED as part of the initialization process.
  - Alternatively, the software can generate random passwords for each drive.
  - The passwords are stored in a password-protected vault on the system.
- Once initialized, SEDs are locked upon power loss, such as a system shutdown or drive removal.

  Locked drives get unlocked after power is restored and the root file system is mounted.
- Provides functionality to export the vault.
- Provides functionality for erasing the drives.
- Provides the ability to revert the initialization.

10.2. Installing the Software

Use the package manager to install the `nv-disk-encrypt` package and, optionally, the TPM2 tools package, and reboot the system. You need the TPM tools package if you plan to use the TPM2 to store security keys.

1. Update the packages.

```
sudo apt update
```

2. Install `nv-disk-encrypt`.

```
sudo apt install -y nv-disk-encrypt
```

3. (Optional) Install the TPM tools package.
   - For DGX A100, DGX Station A100, or DGX H100, install the `tpm2-tools` package.

```
sudo apt install -y tpm2-tools
```

   - For DGX-2, install the `tpm-tools` package.

```
sudo apt install -y tpm-tools
```

4. Reboot.

```
sudo reboot
```

If you plan to use TPM2, enable it. Refer to *Configuring Trusted Computing* for more information.

10.3. Configuring Trusted Computing

Here is some information about the controls that are required to configure Trusted Computing (TC).

The DGX H100 system BIOS provides setup controls for configuring the following TC features:

- Trusted Platform Module

  The NVIDIA DGX H100, DGX A100, DGX A800, and DGX Station A100 incorporate Trusted Platform Module 2.0 (TPM 2.0). The DGX-2 incorporates a TPM module. These modules can be enabled from the system BIOS and used in conjunction with the nv-disk-encrypt tool. After being enabled, the `nv-disk-encrypt` tool uses the TPM for encryption and stores the vault and SED authentication keys on the TPM instead of on the file system. Using the TPM is preferred because this allows the vault data to persist even if the system is reimaged.

- Block SID

  Certain drives shipped with the DGX systems support the Block SID authentication feature. Block SID authentication prevents malicious actors from taking ownership of drives and blocks others from using the drives. By default, the DGX BIOS will send the Block SID request. On such setups, you will need to enable the Disable Block Sid feature in the BIOS before proceeding with the initialization steps.

Note: Enabling the “Disable Block SID” option is only valid for one reboot, so if drive encryption needs to be enabled again, then the feature needs to be enabled in BIOS as well.

10.3.1. Determining Whether Drives Support SID

The drive model is a good indicator of whether the drive supports this feature. Issue the following and look for one of the following model strings:

- KCM6DRUL3T84
- KCM6DRUL7T68
- MZQLB7T6HMLA-00007

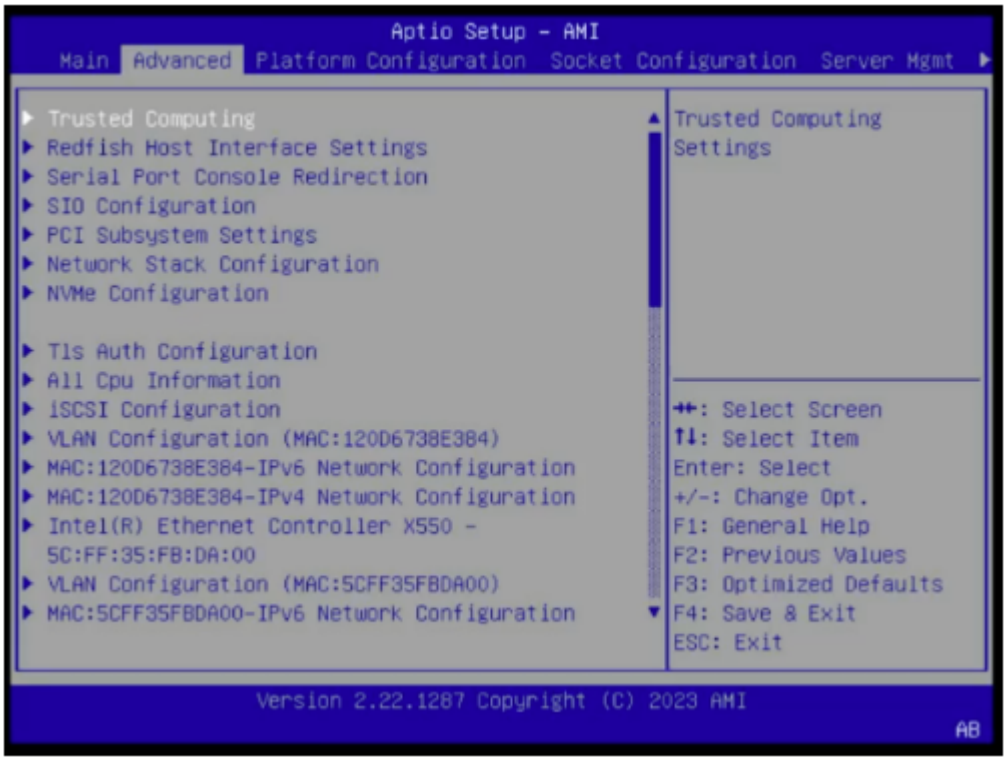
```
sudo nvme list

Node           SN                              Model
------------- ------------------------------ -----
/dev/nvme0n1  70H0A0AHTTHR KCM6DRUL3T84 ...
/dev/nvme1n1  70H0A007TTHR KCM6DRUL3T84
```

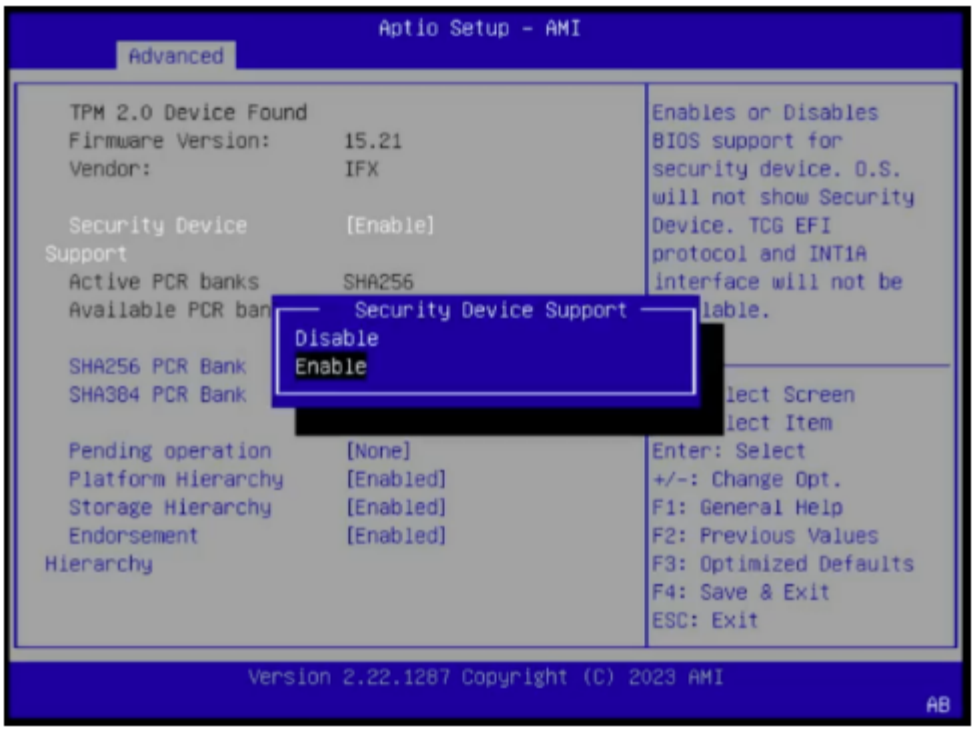
10.3.2. Enabling the TPM and Preventing the BIOS from Sending Block SID Requests

This section provides instructions to enable the TPM and prevent the SBIOS from sending Block SID request. Each task is independent, so you can select which task to complete.

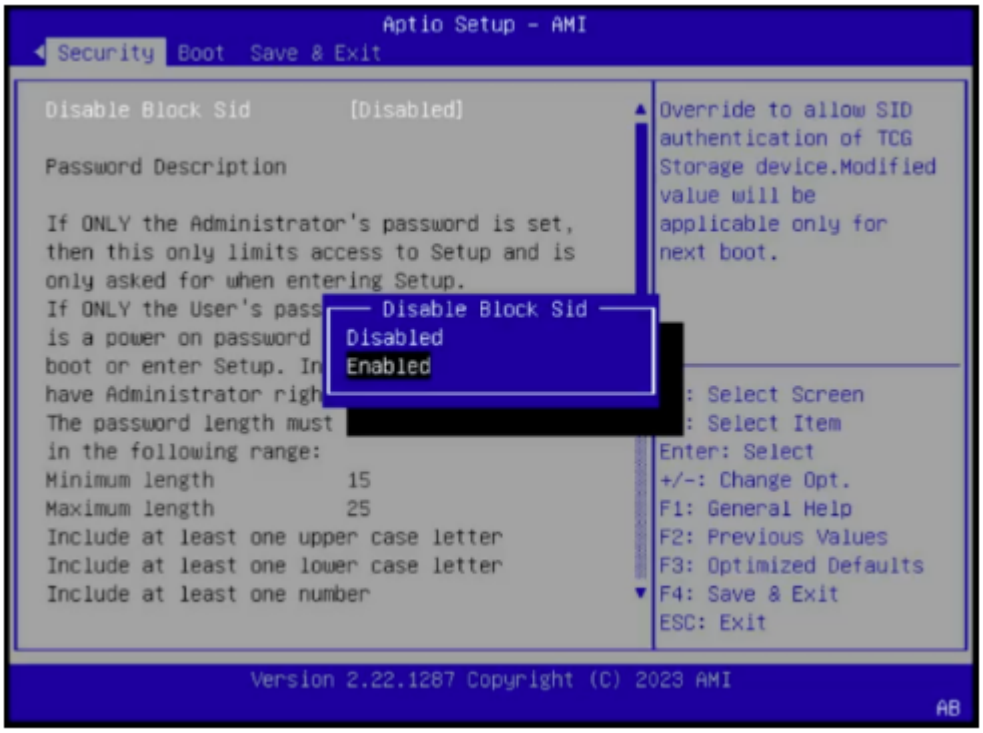
1. Reboot the system, then press [Del] or [F2] at the NVIDIA splash screen to enter the BIOS Setup.
2. Navigate to the Advanced tab on the top menu, then scroll to Trusted Computing and press [Enter].



- To enable TPM, scroll to Security Device and switch the setting to Enabled.



- To disable Block SID, go to the Security tab to the right, and scroll to Disable Block Sid, then switch to Enabled.



3. Save and exit the BIOS Setup to continue the boot process.

   If you disabled Block SID, you will be prompted to accept the request to disable issuing a Block SID Authentication command.

4. Press F10 at the prompt. After the system boots, you can proceed to initialize drive encryption.

10.4. Initializing the System for Drive Encryption

Here is some information about how to initialize the system for drive encryption.

Note: Before initializing drive encryption, review the information in *Configuring Trusted Computing* and follow the configuration instructions if needed.

Initialize the system for drive encryption using the nv-disk-encrypt command.

```
sudo nv-disk-encrypt init [-k <your-vault-password>] [-f <path/to/json-file>] [-g] [-
↪r]
```

Here is a list of the options:

- **-k**: Lets you create the vault password in the command.

  Otherwise, the software will prompt you to create a password before proceeding.

- **-f**: Lets you specify a JSON file that contains a mapping of passwords to drives.

  Refer to "Example 1: Passing in the JSON File" for further instructions.

- **-g**: Generates random salt values (stored in `/etc/nv-disk-encrypt/.dgxenc.salt`) for each drive password.

  Salt values are characters added to a password for enhanced password security. NVIDIA strongly recommends using this option for best security, otherwise the software will use a default salt value instead of a randomly generated one.

- **-r**: Generates random passwords for each drive.

  This avoids the need to create a JSON file or the need to enter a password one by one during the initialization.

10.5. Enabling Drive Locking

After initializing the system for SED management, issue the following command, which uses the `nv-disk-encrypt` command to enable drive locking.

```
sudo nv-disk-encrypt lock
```

After initializing the system and enabling drive locking, the drives will become locked when they lose power. The system will automatically unlock each drive when power is restored to the system and the system is rebooted.

10.6. Initialization Examples

This section provides some initialization examples.

10.6.1. Example 1: Passing in the JSON File

The following instructions in this section describe a method to specify the drive/password mapping ahead of time. This method is useful for initializing several drives at a time and avoids the need to enter the password for each drive after issuing the initialization command, or if you want control of the passwords.

Refer to the following for more information:

- *Determining Which Drives Can be Managed as Self-Encrypting*
- *Creating the Drive/Password Mapping JSON Files and Using it to Initialize the System*

10.6.1.1 Determining Which Drives Can be Managed as Self-Encrypting

Here is some information about how you can determine which drives can be managed as self-encrypting.

Review the storage layout of the DGX system to determine which drives are eligible to be managed as SEDs.

```
sudo nv-disk-encrypt info
```

The default output shows which drives can be used for encryption and which drives cannot. The following status information is provided:

- SED capable: Is this a self-encrypting drive?
- Boot disk: Is this drive currently being used as a boot drive? Does it contain the root filesystem?
- Locked: Is this drive currently in the locked state? Is it able to accept I/O?. It can only be in this state after the following conditions have been met:
  - Locking has been enabled (nv-disk-encrypt init, followed by `nv-disk-encrypt init` lock)
  - The drive is coming back from power-off.
  - The user queries this state prior to it being (automatically) unlocked.
- Lock Enabled: Are locks enabled on this drive? It will be in this state after initialization (`nv-disk-encrypt init`).
- MBR done: This setting is only relevant for drives that support MBR shadowing. On drives that support this feature, this will report 'Y' after initialization (`nv-disk-encrypt init`)

MBR done: This setting is only relevant for drives that support MBR shadowing. On drives that support this feature, this will report 'Y' after initialization (`nv-disk-encrypt init`)

The following example output snippet shows drives than can be used for encryption. Notice SED capable = Y and Boot disk = N.

```
+--------------+--------+-------------------------------------------------------------------------------+¶
| Name | Serial | Status |¶
+--------------+--------+-------------------------------------------------------------------------------+¶
| /dev/nvme3n1 | xxxxx1 | SED capable = Y, Boot disk = N, Locked = N, Lock Enabled = N, MBR done = N |¶
| /dev/nvme6n1 | xxxxx2 | SED capable = Y, Boot disk = N, Locked = N, Lock Enabled = N, MBR done = N |¶
| /dev/nvme9n1 | xxxxx3 | SED capable = Y, Boot disk = N, Locked = N, Lock Enabled = N, MBR done = N |¶
```

The following example output snippet shows drives than cannot be used for encryption. Notice SED capable = Y and Boot disk = Y, or SED capable = N.

```
+--------------+---------+-------------------------------------------------------------------------------+¶
| Name | Serial | Status |¶
+--------------+---------+-------------------------------------------------------------------------------+¶
| /dev/nvme0n1 | xxxxx1  | SED capable = Y, Boot disk = Y, Locked = N, Lock Enabled = N, MBR done = N |¶
| /dev/sr0     | xxxxx2  | SED capable = N, Boot disk = N, Locked = N, Lock Enabled = N, MBR done = N |¶
| /dev/nvme1n1 | xxxxx3  | SED capable = Y, Boot disk = Y, Locked = N, Lock Enabled = N, MBR done = N |¶
| /dev/sda     | unknown | SED capable = N, Boot disk = N, Locked = N, Lock Enabled = N, MBR done = N |¶
```

Alternatively, you can specify the output be presented in JSON format by using the `-j` option.

```
sudo nv-disk-encrypt info -j
```

In this case, drives that can be used for encryption are indicate by the following:

```
"sed_capable": true "used_for_boot": false
```

And drives that cannot be used for encryption are indicated by one of the following:

```
"sed_capable": true "used_for_boot": true
```

Or

```
"sed_capable": false
```

10.6.1.2 Creating the Drive/Password Mapping JSON Files and Using it to Initialize the System

You can initialize the system by creating the drive and password map the JSON files.

1. Create a JSON file that lists all the eligible SED-capable drives that you want to manage.

   Note: These are the list of drives that you obtained from *Determining Which Drives Can be Managed as Self-Encrypting*

   The following example shows the format of the JSON file.

   ```
   {
   "/dev/nvme2n1": "<your-password>",
   "/dev/nvme3n1": "<your-password>",
   "/dev/nvme4n1": "<your-password>",
   "/dev/nvme5n1": "<your-password>",
   }
   ```

   - Ensure that you follow the syntax exactly.
   - Passwords must consist of only upper-case letters, lower-case letters, digits, and/or the following special characters: ~ : @ % ^ + = _ ,

2. Initialize the system and then enable locking.

   The following command assumes you have placed the JSON file in the /tmp directory.

```
sudo nv-disk-encrypt init -f /tmp/<your-file>.json -g
sudo nv-disk-encrypt lock
```

When prompted, enter a password for the vault.

Passwords must consist of only upper-case letters, lower-case letters, digits, and/or the following special characters: ~ : @ % ^ + = _ ,

10.6.2. Example 2: Generating Random Passwords

The commands in this topic use the `-k` and `-r` options so that you are not prompted to enter passwords. You pass the vault password into the command and then the command instructs the tool to generate random passwords for each drive.

The vault password must consist of only upper-case letters, lower-case letters, digits, and/or the following special-characters: ~ : @ % ^ + = _ ,

```
sudo nv-disk-encrypt init -k <your-vault-password> -g -r
sudo nv-disk-encrypt lock
```

10.6.3. Example 3: Specifying Passwords One at a Time When Prompted

If there are a small number of drives, or you do not want to create a JSON file, issue the following command.

```
sudo nv-disk-encrypt init -g
sudo nv-disk-encrypt lock
```

The software prompts you to enter a password for the vault and then a password for each eligible SED.

Passwords must consist of only upper-case letters, lower-case letters, digits, and/or the following special characters: ~ : @ % ^ + = _ ,

10.7. Disabling Drive Locking

To disable drive locking at any time after you initialize, run the following command: $ sudo nv-disk-encrypt disable

This command disables locking on all drives. You can run the initial set up again at any time after this process is complete.

10.8. Enabling Drive Locking

After initializing the system for SED management, issue the following command, which uses the nv-disk-encrypt command to enable drive locking.

```
sudo nv-disk-encrypt lock
```

After initializing the system and enabling drive locking, the drives will become locked when they lose power. The system will automatically unlock each drive when power is restored to the system and the system is rebooted.

10.9. Exporting the Vault

Here is some information about how to export the vault.

To export all drive keys out to a file, use the export function. This requires that you pass in the vault password.

```
sudo nv-disk-encrypt export -k yourvaultpassword
Writing vault data to /tmp/secrets.out
```

10.10. Erasing Your Data

Here is some information about how you can erase your data.

Warning: Be aware when executing this that **all** data will be lost. On DGX H100 systems, these drives generally form a RAID 0 array, and this array will also be destroyed when you perform an erase.

After initializing the system for SED management, use the nv-disk-encrypt command to erase data on your drives after stopping cachefilesd and unmounting the RAID array as follows.

1. Completely stop the RAID.

   ```
   systemctl stop cachefilesd
   sudo umount /raid
   sudo mdadm --stop /dev/md1
   ```

2. Perform the erase.

   ```
   sudo nv-disk-encrypt erase
   ```

This command does the following:

- Sets the drives in an unlocked state.
- Disables locking on the drives.
- Removes the RAID 0 array configuration.

To rebuild the RAID array, issue the following command:

```
sudo /usr/bin/configure_raid_array.py -c -f
```

10.11. Clearing the TPM

If you've lost the password to your TPM, you will not be able to access its contents. In this case, the only way to regain access to the TPM is to clear the TPM's contents. After clearing the TPM, you will need to re-initialize the vault and SED authentication keys.

1. Reboot the system, then press [Del] or [F2] at the NVIDIA splash screen to enter the BIOS Setup.
2. Navigate to the Advanced tab on the top menu, scroll to Trusted Computing, and press [Enter].
3. Clear TPM2.
    1. Scroll to Trusted Computing and press [Enter].
    2. Scroll to Pending Operation and press [Enter].
    3. Select TPM Clear at the Pending Operation popup and press [Enter].
4. Save and exit the BIOS Setup.

10.12. Changing Disk Passwords, Adding Disks, or Replacing Disks

The same steps are needed for changing or rotating passwords, adding disks, or replacing disks.

1. Disable SED management.

    ```
    sudo nv-disk-encrypt disable
    ```

2. Add or replace drives as needed and then rebuild the RAID array. Refer to your system's Service Manual for more information.
3. Enable SED management and assign passwords per the instructions in *Initializing the System for Drive Encryption*.

10.13. Recovering From Lost Keys

NVIDIA recommends backing up your keys and storing them in a secure location. If you lost the key used to initialize and lock your drives, you will not be able to unlock the drive again. If this happens, the only way to recover is to perform a factory-reset, which will result in a loss of data.

SED drives come with a PSID printed on the label; this value can only be obtained by physically examining the drive as exemplified in the following image.

Specify the PSID to reset the drive using the following sedutil-cli command:

```
sudo sedutil-cli ----yesIreallywanttoERASEALLmydatausingthePSID yourdrivesPSID /dev/
↪nvme3n1
```

Chapter 11. Managing and Upgrading Software

DGX OS 6 is an optimized version of the Ubuntu 22.04 Linux distribution with access to a large collection of additional software that is available from the Ubuntu and NVIDIA repositories. You can install the additional software using the `apt` command or through a graphical tool.

Note: The graphical tool is only available for DGX Station and DGX Station A100.

For more information about additional software available from Ubuntu, refer also to Install additional applications

Before you install additional software or upgrade installed software, refer also to the *Release Notes* for the latest release information.

11.1. Upgrading the System

Before installing any additional software, you should upgrade the system to the latest versions. This ensures that you have access to new software releases that have been added to the repositories since your last upgrade. Refer to *Upgrading the OS* for more information and instructions including instructions for enabling Ubuntu's Extended Security Maintenance updates.

Important: You will only see the latest software branches after upgrading DGX OS.

Note: When you switch between software branches, such as the GPU driver or CUDA toolkit, you have to install the package(s) for the new branch. Depending on the software, it will then remove the existing branch or support concurrent branches installed on a system.

11.2. Changing Your GPU Branch

NVIDIA drivers are released as precompiled and signed kernel modules by Canonical and are available directly from the Ubuntu repository. Signed drivers are required to verify the integrity of driver packages and identity of the vendor.

However, the verification process requires that Canonical build and release the drivers with Ubuntu kernel updates after their release cycle is complete, and this process might sometimes delay new driver branch releases and updates. For more information about the NVIDIA driver release, refer to the release notes at NVIDIA Driver Documentation.

Important: The Ubuntu repositories provide the following versions of the signed and precompiled NVIDIA drivers:

- The general NVIDIA display drivers
- The NVIDIA Data Center GPU drivers

On your DGX system, only install the packages that include the NVIDIA Data Center GPU drivers. The metapackages for the NVIDIA Data Center GPU driver have the `-server` or `-server-open` suffix.

11.2.1. Checking the Currently Installed Driver Branch

Before you install a new NVIDIA driver branch, to check the currently installed driver branch, run the following command:

```
apt list --installed nvidia-driver*server
```

11.2.2. Determining the New Available Driver Branches

These steps help you determine which new driver branches are available.

To see the new available NVIDIA driver branches:

1. Update the local database with the latest information from the Ubuntu repository.

   ```
   sudo apt update
   ```

2. Show all available driver branches.

   ```
   apt list nvidia-driver-*-server
   ```

3. Optional: Show the available NVIDIA Open GPU Kernel module branches.

   ```
   apt list nvidia-driver-*-server-open
   ```

   Caution: The NVIDIA Open GPU Kernel module drivers are not supported on NVIDIA DGX-1, DGX-2, and DGX Station systems.

11.2.3. Upgrading Your GPU Branch

To manually upgrade your driver to the latest branch:

1. Install the latest kernel.

```
sudo apt install -y linux-nvidia
```

2. Install the latest NVIDIA GPU driver.

   In the following commands, the trailing - character in `*nvidia*${GPU_BRANCH}*-`, specifies to remove the old driver in the same transaction. Because this operation removes packages from the system, it is important to perform a dry-run first and ensure that the correct packages will be removed.

   Set `GPU_BRANCH` to the latest branch version, such as 535.

   - On **non-Fabric Manager** systems, such as NVIDIA DGX-1, DGX Station, and DGX Station A100, run the following command:

```
GPU_BRANCH=$(dpkg -l | grep nvidia-driver | tr -s " " | cut -d' ' -f3 | cut -d
↪'.' -f1)

# Specify --dry-run to check the packages to install.
sudo apt-get install -y linux-modules-nvidia-535-server-nvidia nvidia-driver-
↪535-server libnvidia-nscq-535 nvidia-modprobe "*nvidia*${GPU_BRANCH}*-" --
↪dry-run

# Install the packages.
sudo apt-get install -y linux-modules-nvidia-535-server-nvidia nvidia-driver-
↪535-server libnvidia-nscq-535 nvidia-modprobe "*nvidia*${GPU_BRANCH}*-"
```

   - On Fabric Manager systems, NVIDIA DGX-2, DGX A100, and DGX H100, run the same command, but append the `nvidia-fabricmanager-535` package:

```
GPU_BRANCH=$(dpkg -l | grep nvidia-driver | tr -s " " | cut -d' ' -f3 | cut -d
↪'.' -f1)

# Specify --dry-run to check the packages to install.
sudo apt-get install -y linux-modules-nvidia-535-server-nvidia nvidia-driver-
↪535-server libnvidia-nscq-535 nvidia-modprobe nvidia-fabricmanager-535
↪"*nvidia*${GPU_BRANCH}*-" --dry-run

# Install the packages.
sudo apt-get install -y linux-modules-nvidia-535-server-nvidia nvidia-driver-
↪535-server libnvidia-nscq-535 nvidia-modprobe nvidia-fabricmanager-535
↪"*nvidia*${GPU_BRANCH}*-"
```

   - To install the NVIDIA Open GPU Kernel module drivers, specify the `-server-open` package name suffix, such as `linux-modules-nvidia-535-server-open-nvidia` and `nvidia-driver-535-server-open`. For example,

```
# Specify --dry-run to check the packages to install.
$ sudo apt-get install -y linux-modules-nvidia-535-server-open-nvidia nvidia-
↪driver-535-server-open libnvidia-nscq-535 nvidia-modprobe nvidia-
↪fabricmanager-535 --dry-run
```

(continues on next page)

(continued from previous page)

```
# Install the packages.
$ sudo apt-get install -y linux-modules-nvidia-535-server-open-nvidia nvidia-
↪driver-535-server-open libnvidia-nscq-535 nvidia-modprobe nvidia-
↪fabricmanager-535
```

Note: The driver versions are only used as an example. Replace the value with the version that you want to install.

3. Reboot the system to ensure the new drivers get loaded:

```
sudo reboot
```

11.3. Installing or Upgrading to a Newer CUDA Toolkit Release

Only DGX Station and DGX Station A100 have a CUDA Toolkit release installed by default. DGX servers are intended to be shared resources that use containers and do not have CUDA Toolkit installed by default. However, you have the option to install a qualified CUDA Toolkit release.

Although the DGX OS supports all CUDA Toolkit releases that interoperate with the installed driver, DGX OS releases might include a default CUDA Toolkit release that might not be the most recently released version. Unless you must use a new CUDA Toolkit version that contains the new features, we recommend that you remain on the default version that is included in the DGX OS release. Refer to the DGX OS Software Release Notes for the default CUDA Toolkit release.

Important: Before you install or upgrade to any CUDA Toolkit release, ensure the release is compatible with the driver that is installed on the system. Refer to CUDA Compatibility for more information and a compatibility matrix.

11.3.1. CUDA Compatibility Matrix and Forward Compatibility

Each CUDA toolkit requires a minimum GPU driver version. This compatibility matrix is documented in CUDA Compatibility

Newer CUDA Toolkits may be used with older GPU drivers if the appropriate forward compatibility package is installed. Refer to: Installing the Forward Compatibility Package

Example:

CUDA toolkit 12.0 requires GPU driver version 525.60.13, however GPU driver 515.43.04 is installed. In order to use CUDA toolkit 12.0 with the older GPU driver, you must install the cuda-compat-12-0 package:

```
sudo apt install cuda-compat-12-0
```

Set either LD_LIBRARY_PATH manually:

```
LD_LIBRARY_PATH=/usr/local/cuda/compat:$LD_LIBRARY_PATH
```

or automatically via the /etc/ld.so.conf file or by adding a file under /etc/ld.so.conf.d/.

11.3.2. Checking the Currently Installed CUDA Toolkit Release

Here is some information about the prerequisite to determine the CUDA Toolkit release that you currently have installed.

Before you install a new CUDA Toolkit release, to check the currently installed release, run the following command:

```
apt list --installed cuda-toolkit-*
```

The following example output shows that CUDA Toolkit 11.0 is installed:

```
apt list --installed cuda-toolkit-*
Listing... Done
cuda-toolkit-11-0/unknown,unknown,now 11.0.3-1 amd64 [installed]
N: There is 1 additional version. Please use the '-a' switch to see it
```

11.3.3. Installing or Upgrading the CUDA Toolkit

These steps help you determine which new CUDA Toolkit releases are available.

To see the new available CUDA Toolkit releases:

1. Update the local database with the latest information from the Ubuntu repository.

   ```
   sudo apt update
   ```

2. Show all available CUDA Toolkit releases.

   ```
   apt list cuda-toolkit-*
   ```

   The following output shows that 11.7, 11.8, 12.0 are the possible CUDA Toolkit versions that can be installed:

   ```
   Listing... Done
   cuda-toolkit-11-7/unknown 11.7.1-1 amd64
   cuda-toolkit-11-8/unknown 11.8.0-1 amd64
   cuda-toolkit-12-0/unknown 12.0.0-1 amd64
   ```

3. To install or upgrade the CUDA Toolkit, run the following:

   ```
   sudo apt install cuda-toolkit-<version>
   ```

11.4. Installing the Mellanox OFED Drivers

DGX OS 6 uses the OFED drivers supplied with the Ubuntu 22.04 distribution. Alternatively, you can install the Mellanox OFED (MOFED) drivers. The Mellanox OFED drivers are tested and packaged by NVIDIA.

DGX OS 6 includes the script `/usr/sbin/nvidia-manage-ofed.py` to assist in managing the OFED stacks.

Run the following command to display a list of OFED-related packages:

```
sudo nvidia-manage-ofed.py -s
```

The command output indicates if the packages are part of the Mellanox stack or the Ubuntu stack.

11.4.1. Using the Mellanox OFED Packages

If you are upgrading from OS 5 to OS 6, refer to Upgrading in the *DGX OS 5 User Guide* before you change drivers.

1. Ensure that you have the latest nvidia-manage-ofed package by running these commands:

   ```
   sudo apt update
   sudo apt upgrade
   ```

2. Remove the inbox OFED components:

   ```
   sudo /usr/sbin/nvidia-manage-ofed.py -r ofed
   ```

3. Add the Mellanox OFED components:

   ```
   sudo /usr/sbin/nvidia-manage-ofed.py -i mofed
   ```

   Note: The command installs the latest version of MLNX_OFED that is currently available in the repositories. To install an alternative version other than the latest version, specify the alternative version by using the `-v` option. The following example installs MLNX_OFED version 5.9-0.5.6.0:

   ```
   sudo /usr/sbin/nvidia-manage-ofed.py -i mofed -v 5.9-0.5.6.0
   ```

4. Reboot the system.

11.4.2. Using the Ubuntu OFED Packages

1. Remove the Mellanox OFED components:

   ```
   sudo /usr/sbin/nvidia-manage-ofed.py -r mofed
   ```

2. Ensure the APT is not configured with the Mellanox repository.

   Remove the `/etc/apt/sources.list.d/mlnx.list` file.

3. Update APT so it no longer has information about the packages from the Mellanox repository:

```
sudo apt update
```

4. Add the Ubuntu OFED components:

```
sudo /usr/sbin/nvidia-manage-ofed.py -i ofed
```

11.4.3. Inbox OFED vs Mellanox OFED Use Cases

The following table describes common MOFED utilities / use cases, and how to accomplish them with inbox OFED tools.

One key difference between MOFED and inbox OFED is that the /dev/mst* devices aren't used by inbox OFED. Devices are addressed by their PCIe Bus:Device:Function instead.

Use Case	Mellanox OFED Method	Inbox OFED Method
Correlate IB devices to network devices	ibdev2netdev	rdma link show
Provide information about bond / mtu	net-interfaces	View the contents of: ► /sys/class/net/bonding_masters ► /sys/class/net/bond<num>/ ► /proc/net/bonding
Reload OFED drivers	openidb	modprobe
Manipulation with device configuration	mlxconfig	mstconfig
FW image burn	flint	mstflint
Collect debug traces	fwtrace	mstfwtrace
Reset operation on device	mlxfwreset	mstfwreset
Configuration Register Access tool	mcra	mstmcra
Manipulation with host privileges	mlxprivhost	mstprivhost
Dump device internal configuration registers	mlxdump	mstregdump
Read device Vital Product Data	mlxvpd	mstvpd

11.4.4. Initialize on Alloc Performance Impact

The `CONFIG_INIT_ON_ALLOC_DEFAULT_ON` Linux kernel configuration option controls whether the kernel fills newly allocated pages and heap objects with zeroes by default. You can override this setting with the `init_on_alloc=<0|1>` kernel parameter. The DGX OS that is preinstalled on NVIDIA DGX System sets `init_on_alloc=1` because this setting is the recommended default by Ubuntu for kernel hardening. However, this setting can have a performance impact on the network interface controller performance on DGX systems because zeroing every buffer page upon allocation is frequent and requires time to complete. This option can impact the performance with the inbox OFED driver more than Mellanox OFED (MOFED) driver. The MOFED driver allocates a much larger page cache which tolerates the increased kernel cost of zeroing pages better. NVIDIA recommends that you keep the default setting, `init_on_alloc=1` for best security. If your deployment permits less strict security and the network interface controller is underperforming, you can try disabling the security feature.

1. Edit the `/etc/default/grub` file and add `init_on_alloc=0` to the `GRUB_CMDLINE_LINUX_DEFAULT` variable.
2. Generate the GRUB bootloader.

```
sudo update-grub
sudo reboot
```

3. Optional: After the system reboots, verify the change took effect.

```
cat /proc/cmdline
```

Example Output

BOOT_IMAGE=/boot/vmlinuz-... init_on_alloc=0

11.4.5. Upgrading Firmware for Mellanox ConnectX Cards

DGX OS 6 uses the open source mstflint program to upgrade firmware on Mellanox cards.

11.4.5.1 Checking the Device Type

Mstflint uses the PCI-E Bus/Device/Function (BDF) identifier to specify devices. To locate all Mellanox Ethernet and Infiniband devices in the system,

1. execute the command:

```
lspci | grep Mellanox | grep -e Infiniband -e Ethernet
```

2. The first column of the output is the BDF, for example:

```
29:00.0 Infiniband controller: Mellanox Technologies MT2910 Family [ConnectX-7]
29:00.1 Ethernet controller: Mellanox Technologies MT2910 Family [ConnectX-7]
```

3. To locate the correct firmware for your device you will need the OPN and PSID of the device.
4. To find the OPN, use the `mstvpd` command:

```
# mstvpd 29:00.0 | grep PN
PN:       MCX755206AS-NEAT-N
```

5. To find the PSID, use the mstflint command:

```
# mstflint -d 29:00.0 q | grep PSID
PSID:                       MT_0000000892
```

11.4.5.2 Download the New Firmware

1. Navigate to https://network.nvidia.com/support/firmware/firmware-downloads/
2. Select the product line, e.g.: ConnectX-6 Infiniband.
3. Select the firmware version.
4. Select the OPN and PSID that matches your device.
5. Select "Download" to download the firmware.
6. Use the "unzip" command to unpack the compressed file to access the .bin file.

11.4.5.3 Program the Firmware

1. Use the mstflint command to program the device:

```
# mstflint -d 29:00.0 -i fw-ConnectX7-rel-28_36_1010-MCX75310AAS-HEA-N_Ax-UEFI-14.
↪29.14-FlexBoot-3.6.901.signed.bin burn
```

2. After installing the new firmware, reboot the system.

11.5. Installing GPUDirect Storage Support

NVIDIA Magnum IO GPUDirect Storage (GDS) enables a direct data path for direct memory access (DMA) transfers between GPU memory and storage. This software avoids a bounce buffer through the CPU.

Note: This section only applies if you intend to use GPUDirect Storage in bare metal.

11.5.1. Prerequisites

- Determine if GDS is supported with your Linux kernel and whether you need to install OFED drivers.

	Optimized NVIDIA Kernel	Generic kernel
Ubuntu OFED	All kernel modules (nvidia-fs, NVMe, NVMf, NFS) related to GDS are part of the Optimized NVIDIA kernel.	Unsupported
Mellanox OFED	All kernel modules related to GDS are part of the Optimized NVIDIA kernel. Install MOFED. Refer to *Installing Mellanox OFED Drivers* and then perform the steps in the following section.	GDS kernel modules are not present in the Ubuntu generic kernel. These kernel modules are patched via MOFED installation. Install MOFED. Refer to *Installing Mellanox OFED Drivers* and then perform the steps in the following section.

For additional help, refer to MLNX_OFED Requirements and Installation in the *NVIDIA GPUDirect Storage Installation and Troubleshooting Guide*.

- For systems other than NVIDIA DGX-1, DGX-2, and DGX Station, to use the latest GDS version, 12.2.2-1, that is provided by nvidia-fs-dkms-2.17.5-1, you must install an NVIDIA Open GPU Kernel module driver. Refer to *Changing Your GPU Branch* for more information about installing the driver.
- For NVIDIA DGX-1, DGX-2, and DGX Station running the generic Linux Kernel, the GPUs in these systems are not supported with the NVIDIA Open GPU Kernel modules. The GDS versions 12.2.2-1 and higher only support the Open GPU Kernel modules.

  For these systems, you must pin the nvidia-fs package to version 2.17.3 or lower and the nvidia-gds package to version 12.2.1-1 or lower.

  1. Create an `/etc/apt/preferences.d/nvidia-fs` file with contents like the following.

     ```
     Package: nvidia-fs
     Pin-Priority: 900
     Pin: version 2.17.3-1

     Package: nvidia-fs-dkms
     Pin-Priority: 900
     Pin: version 2.17.3-1

     Package: nvidia-gds
     Pin-Priority: 900
     Pin: version 12.2.1-1

     Package: nvidia-gds-12-2
     Pin-Priority: 900
     Pin: version 12.2.1-1
     ```

  2. Verify that the nvidia-fs package preference is correct.

     ```
     sudo apt-cache policy nvidia-fs
     ```

     Example Output

     ```
     nvidia-fs:
       Installed: (none)
       Candidate: 2.17.3-1
     ```

(continues on next page)

(continued from previous page)

```
  Version table:
     2.17.5-1 580
        580 https://developer.download.nvidia.com/compute/cuda/repos/
↪ubuntu2204/x86_64  Packages
     2.17.3-1 900
        580 https://developer.download.nvidia.com/compute/cuda/repos/
↪ubuntu2204/x86_64  Packages
     ...
```

3. Verify that the nvidia-gds package preference is correct.

```
sudo apt-cache policy nvidia-gds
```

Example Output

```
nvidia-gds:
  Installed: (none)
  Candidate: 12.2.1-1
  Version table:
     12.2.2-1 580
        580 https://developer.download.nvidia.com/compute/cuda/repos/
↪ubuntu2204/x86_64  Packages
     12.2.1-1 900
        580 https://developer.download.nvidia.com/compute/cuda/repos/
↪ubuntu2204/x86_64  Packages
     ...
```

- For NVIDIA DGX-1, DGX-2, and DGX Station, disable IOMMU to avoid a DMAR penalty.

1. Edit the GRUB configuration file.

```
sudo vi /etc/default/grub
```

2. Add `intel_iommu=off` to the `GRUB_CMDLINE_LINUX_DEFAULT` variable.

   If the variable already includes other options, enter a space to separate the options. Refer to the following example.

```
...
GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 intel_iommu=off"
...
```

3. Generate the GRUB bootloader.

```
sudo update-grub
sudo reboot
```

4. After the system reboots, verify the change took effect.

```
cat /proc/cmdline
```

Example Output

```
BOOT_IMAGE=/boot/vmlinuz-... console=tty0 intel_iommu=off
```

11.5.2. Installing GDS Components for the Optimized NVIDIA Kernel

This procedure applies to both the Ubuntu OFED and Mellanox OFED with the Optimized NVIDIA kernel.

The nvidia-fs kernel module is already part of the Optimized NVIDIA kernel.

- To use GDS with the Optimized NVIDIA kernel, install the CUDA libcufile packages:

```
sudo apt install libcufile-<ver> libcufile-dev-<ver> gds-tools-<ver>
```

  Use the CUDA Toolkit version number in place of <ver>, such as 12-2.

11.5.3. Installing GDS Components for the Generic Kernel

The GDS user-space components, libcufile and tools, are required and are installed by performing this procedure.

To use GDS with the Ubuntu generic kernel, perform the following steps:

1. Set the `NVIDIA_DRV_VERSION` environment variable to the driver version.

```
NVIDIA_DRV_VERSION=$(cat /proc/driver/nvidia/version | grep Module | awk '{print
↪$8}' | cut -d '.' -f 1)
```

2. Install the nvidia-gds package.
   - For NVIDIA DGX-1, DGX-2, and DGX Station that must use version `12.2.1-1`:

```
sudo apt install nvidia-gds-12-2=12.2.1-1  nvidia-dkms-${NVIDIA_DRV_VERSION}-
↪server
```

   - For other NVIDIA DGX Systems:

```
sudo apt install nvidia-gds-<version>  nvidia-dkms-${NVIDIA_DRV_VERSION}-
↪server
```

     Use the CUDA Toolkit version number in place of <version>, such as 12-2.

11.5.4. Enabling Relaxed Ordering for NVMe Drives

The Samsung NVMe drives used in NVIDIA DGX systems support relaxed ordering for I/O operations. Relaxed ordering enables the PCIe bus to complete transactions out of order. NVIDIA recommends enabling this setting when using GPUDirect Storage to improve performance.

- Run the `nvidia-relaxed-ordering-nvme.sh` utility:

```
sudo /bin/nvidia-relaxed-ordering-nvme.sh enable
```

11.5.5. Next Steps

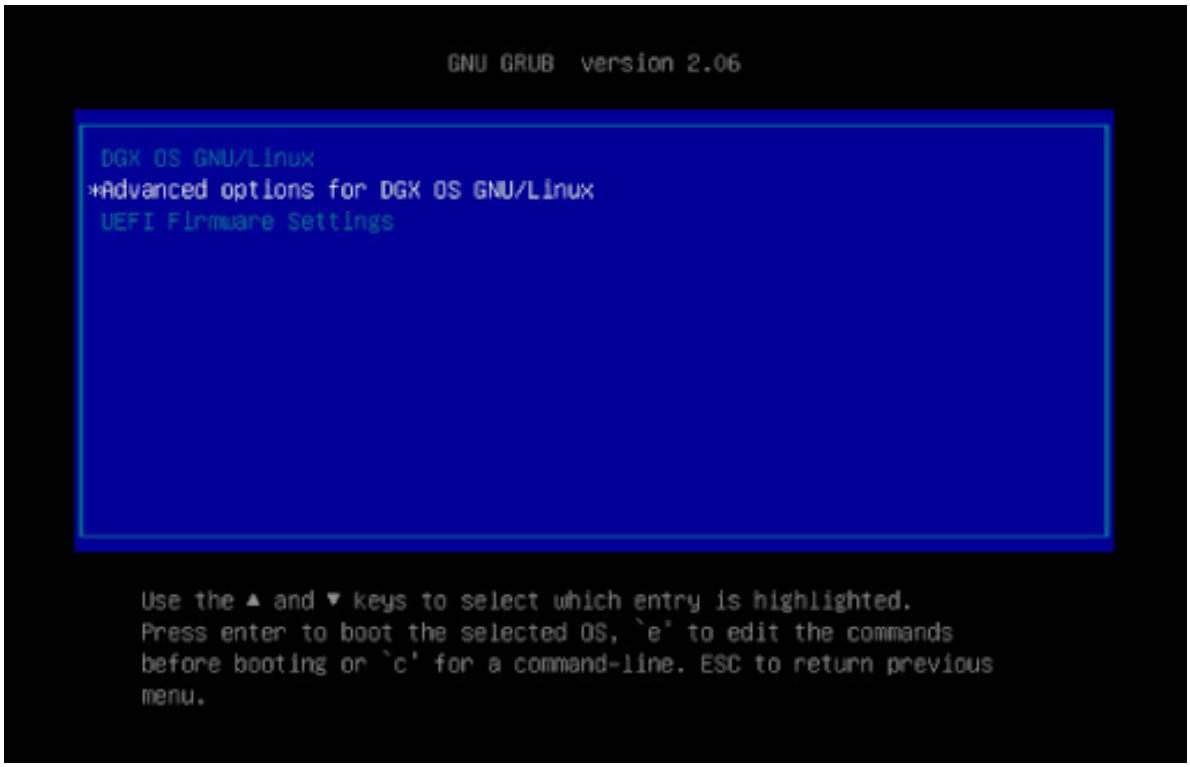
Refer to Verifying a Successful GDS Installation in the *NVIDIA GPUDirect Storage Installation and Troubleshooting Guide*.

11.6. Selecting a different Linux kernel

DGX OS 6 uses a kernel optimized for NVIDIA systems. Install the generic Ubuntu kernel by running `sudo apt install linux-generic`.

11.6.1. Boot the system to the generic kernel one time

1. To boot from the generic kernel first select “Advance options for DGX OS GNU/Linux” in the grub menu:



2. Select the generic kernel:

11.6.2. Boot the system to the generic kernel by default

As shipped, `GRUB_DEFAULT` is set to `0`. This setting indicates to boot the first menu entry on the first menu. The first menu entry is always set to the newest kernel.

To set a different default kernel, you must specify both the menu entry and the submenu item.

1. View the `/boot/grub/grub.cfg` file.
   - Locate the **Advanced options for DGX OS GNU/Linux** submenu line. Copy the ID that follows the `$menuentry_id_option` string. In the following example, the ID begins with `gnulinux-advanced`.

     ```
     submenu 'Advanced options for DGX OS GNU/Linux' $menuentry_id_option
     ↪'gnulinux-advanced-342551e3-c0b6-46da-90c1-d938ff352025'
     ```

   - Further down in the file, locate the menu entry line for the kernel that you want to boot. In the following example, the menu entry is **DGX OS GNU/Linux, with Linux 5.15.0-84-generic**.

     Copy the ID that follows the `$menuentry_id_option` string. In the following example, the ID begins with `gnulinux-5.15.0-84-generic`.

     ```
     menuentry 'DGX OS GNU/Linux, with Linux 5.15.0-84-generic' --class dgx --
     ↪class gnu-linux --class gnu --class os --unrestricted $menuentry_id_option
     ↪'gnulinux-5.15.0-84-generic-advanced-342551e3-c0b6-46da-90c1-d938ff352025'
     ```

2. Edit the `/etc/default/grub` file and set the `GRUB_DEFAULT` value to the ID of the submenu, the greater than character (>), and the ID of the kernel, like the following example.

Example

```
GRUB_DEFAULT='gnulinux-advanced-342551e3-c0b6-46da-90c1-d938ff352025>gnulinux-5.
↪15.0-84-generic-advanced-342551e3-c0b6-46da-90c1-d938ff352025'
```

3. Update the grub configuration.

```
sudo update-grub2
```

Example Output

```
Sourcing file `/etc/default/grub'
Sourcing file `/etc/default/grub.d/hugepage.cfg'
...
Adding boot menu entry for UEFI Firmware Settings ...
done
```

The system uses the newly selected default kernel on the next boot.

Chapter 12. Known Issues

12.1. Incorrect DCGM Version After Upgrade from 5.X to 6.2.0

12.1.1. Issue

During the upgrade process from DGX OS 5.X to DGX OS 6.2.0, the version of the NVIDIA® Data Center GPU Manager (DCGM) is not updated to the latest version.

12.1.2. Workaround

When you upgrade from Base OS 5.x to Base OS 6.2.0, DCGM might not be upgraded automatically. To ensure that the latest DCGM version is installed, run the following commands after the upgrade and reboot are complete:

```
sudo apt update
sudo apt upgrade
```

12.2. Drop IQ2M Usage to Push QMDs

12.2.1. Issue

A potential corruption can occur due to launching kernels on H100 GPUs. The issue is more likely to happen when the GPU is shared among multiple processes. This symptom might manifest in XID 13 error messages, such as `Graphics Exception: SKEDCHECK11_TOTAL_THREADS`. Currently, this issue has no user-controllable workaround and will be resolved in the next release.

Note: At the time of the 6.2.0 release, the software stack does not contain the latest R535TeslaRD8 GPU driver, which will be available in an upcoming release shortly.

12.3. Errors Occur When Loading Mirrored Repositories on Air-Gapped Systems

12.3.1. Issue

When you run the `apt update` command to load mirrored repositories on an air-gapped system, the following error messages appear:

```
File not found - /media/repository/mirror/security.ubuntu.com/ubuntu/dists/jammy-
↪security/main/cnf/Commands-amd64 (2: No such file or directory)
```

```
Failed to fetch file:/media/repository/mirror/security.ubuntu.com/ubuntu/dists/jammy-
↪security/main/cnf/Commands-amd64  File not found - /media/repository/mirror/
↪security.ubuntu.com/ubuntu/dists/jammy-security/main/cnf/Commands-amd64 (2: No such
↪file or directory)
```

12.3.2. Explanation

This issue occurs because a fix for the `apt-mirror` package, which is available in Ubuntu 23.10, has yet to be implemented in the Ubuntu 22.04 repositories. If you are using an `apt-mirror` package

- Version later than 0.5.4-1: Contact NVIDIA Enterprise Services by filing a support case.
- Version 0.5.4-1: Use the following workaround to mirror the repositories.

You can run the following command to determine the version of your `apt-mirror` package:

```
$ dpkg -l | grep apt-mirror

ii  apt-mirror                  0.5.4-1               all             APT sources
↪mirroring tool
```

12.3.3. Workaround

To resolve the issue, follow these instructions using an Ubuntu 23.10 Docker image:

1. On an Ubuntu 20.04 or later system with network access, format a removable USB flash drive and mount that drive at `/media`. For example,

   ```
   sudo mkfs.ext4 device
   sudo mount -t ext4 device /media
   ```

2. Create an empty directory and make it accessible by a user who can access a Docker container, such as joe.

   ```
   mkdir /media/repository
   chown joe /media/repository
   chmod 755 /media/repository
   ```

3. As the user specified in step 2, create the following two files:

   ```
   ./mirror.list

   set base_path /media/repository
   set run_postmirror 0
   set nthreads 20
   set _tilde 0
   deb http://security.ubuntu.com/ubuntu jammy-security main multiverse universe
   ↪restricted
   deb http://archive.ubuntu.com/ubuntu/ jammy main multiverse universe restricted
   deb http://archive.ubuntu.com/ubuntu/ jammy-updates main multiverse universe
   ↪restricted
   deb [ arch=amd64 ] https://developer.download.nvidia.com/compute/cuda/repos/
   ↪ubuntu2204/x86_64/ /
   deb [ arch=amd64 ] https://repo.download.nvidia.com/baseos/ubuntu/jammy/x86_64/
   ↪jammy common dgx
   deb [ arch=amd64 ] https://repo.download.nvidia.com/baseos/ubuntu/jammy/x86_64/
   ↪jammy-updates common dgx
   deb https://developer.download.nvidia.com/hpc-sdk/ubuntu/amd64 /
   ```

(continues on next page)

(continued from previous page)

```
deb [ arch=amd64 ] https://repo.download.nvidia.com/baseos/ubuntu/jammy/x86_64/
↪jammy common dgx
deb [ arch=amd64 ] https://repo.download.nvidia.com/baseos/ubuntu/jammy/x86_64/
↪jammy-updates common dgx
```

```
./Dockerfile

FROM ubuntu:23.10
ENV DEBIAN_FRONTEND=noninteractive
RUN apt update
RUN apt install -y apt-mirror
COPY ./mirror.list /etc/apt/mirror.list
RUN chmod 644 /etc/apt/mirror.list

CMD ["apt-mirror"]
```

4. As the user specified in step 2, run the following commands to build the mirrors on `/media/repository`.

```
docker build -t dgxos6mirror .
docker run --rm -it -v /media/repository/:/media/repository dgxos6mirror
```

Note: This step takes a long time to complete due to nearly 1 TB of data to download.

5. Dismount the `media` directory from the networked system:

```
sudo umount /media
```

6. Move and mount the `media` directory to the target DGX system:

```
sudo mount -t <device> /media
```

7. As root, edit the `sources.list`, `cuda-compute-repo.list`, `dgx.list`, and `nvhpc.list` files to point to the correct local mirrors as follows:

```
/etc/apt/sources.list
deb file:///media/repository/mirror/archive.ubuntu.com/ubuntu/ jammy main
↪restricted universe multiverse
deb file:///media/repository/mirror/archive.ubuntu.com/ubuntu/ jammy-updates main
↪restricted universe multiverse
deb file:///media/repository/mirror/security.ubuntu.com/ubuntu/ jammy-security
↪main restricted universe multiverse
```

```
/etc/apt/sources.list.d/cuda-compute-repo.list
deb [arch=amd64 signed-by=/usr/share/keyrings/cuda_debian_prod.gpg] file:///raid/
↪media/repository/mirror/developer.download.nvidia.com/compute/cuda/repos/
↪ubuntu2204/x86_64/ /
```

```
/etc/apt/sources.list.d/dgx.list
deb [arch=amd64 signed-by=/usr/share/keyrings/dgx_debian_prod.gpg] file:///raid/
↪media/repository/mirror/repo.download.nvidia.com/baseos/ubuntu/jammy/x86_64/
↪jammy common dgx
deb [arch=amd64 signed-by=/usr/share/keyrings/dgx_debian_prod.gpg] file:///raid/
↪media/repository/mirror/repo.download.nvidia.com/baseos/ubuntu/jammy/x86_64/
↪jammy-updates common dgx
```

(continues on next page)

(continued from previous page)

```
/etc/apt/sources.list.d/nvhpc.list
deb [arch=amd64 signed-by=/usr/share/keyrings/nvidia-hpcsdk-archive-keyring.gpg]
↪file:///raid/media/repository/mirror/developer.download.nvidia.com/hpc-sdk/
↪ubuntu/amd64 /
```

8. Review other files in the `sources.list.d` directory to verify that you do not have duplicate entries for the same repositories.
9. Test that your target system can load these repositories.

```
sudo apt update
```

If you see error messages, contact NVIDIA Enterprise Services.

12.4. Reduced Network Communication Speeds on DGX H100 System

12.4.1. Issue

On DGX H100 Systems running DGX OS 6.0, the `MAX_ACC_OUT_READ` configuration for the ConnectX-7 controllers can be too low and result in reduced data transfer rates.

This issue is fixed for DGX H100 systems that ship from NVIDIA with DGX OS 6.1 preinstalled.

For DGX H100 systems that were initially preinstalled with DGX OS 6.0, the workaround in the following section is required even if the system is upgraded or re-imaged to DGX OS 6.1.

12.4.2. Explanation

For DGX H100 Systems, the value is set to 44 by the `nvidia-mlnx-config` service.

You can check the value by using the `mlxconfig` or `mstconfig` command to query each device and view the value of the `MAX_ACC_OUT_READ` configuration.

12.4.3. Workaround

Perform the following steps to set the value to `0`. For DGX OS 6.0, disable the `nvidia-mlnx-config` service.

On systems that were upgraded or re-imaged with DGX OS 6.1, the following `systemctl` commands might return an error message that begins with `Failed to stop...`. This message can occur if the `nvidia-mlnx-config` service is not running or not installed. Continue to perform the remaining steps.

1. For DGX OS 6 systems, stop and disable the `nvidia-mlnx-config` service.

1. Stop the service:

```
sudo systemctl stop nvidia-mlnx-config
```

2. Disable the service so that it does not start after the next boot and revert the configuration:

```
sudo systemctl disable nvidia-mlnx-config
```

2. For all systems, set the configuration based on whether your system uses the MLNX_OFED drivers or the Inbox drivers.

   - MLNX_OFED drivers

```
for dev in $(ls /sys/class/infiniband/); do \
  sudo mlxconfig -y -d ${dev} set ADVANCED_PCI_SETTINGS=1; \
  sudo mlxconfig -y -d ${dev} set MAX_ACC_OUT_READ=0; \
done
```

   - Inbox OFED drivers

```
for bdf in $(ls /sys/bus/pci/devices); do \
  if [[ -e "/sys/bus/pci/devices/${bdf}/infiniband" ]]; then \
    sudo mstconfig -y -d "${bdf}" set ADVANCED_PCI_SETTINGS=1; \
    sudo mstconfig -y -d "${bdf}" set MAX_ACC_OUT_READ=0; \
  fi \
done
```

3. Perform a chassis power cycle so the changes take effect.

12.5. NVSM Raises Alerts for Missing Devices on DGX H100 System

12.5.1. Issue

NVSM reports the following missing devices on NVIDIA DGX H100 Systems:

```
/systems/localhost/pcie/alerts/alert0
    message_details = Device is missing on b1:00.1.
    ...

/systems/localhost/pcie/alerts/alert1
    message_details = Device is missing on b1:00.0.
    ...

 /systems/localhost/pcie/alerts/alert2
    message_details = Device is missing on 0b:00.1.
```

12.5.2. Explanation

NVSM version 22.12.02 is configured with the preceding devices as resources in the DGX H100 System configuration file and the devices are not present in the system at the reported PCI IDs.

You can ignore the alerts for these PCI IDs. The alerts are false positives.

12.5.3. Workaround

After upgrading to a newer version of NVSM, the alerts can persist in local storage. Perform the following steps to remove the NVSM alert database from local storage:

1. Stop the NVSM service:

   ```
   sudo systemctl stop nvsm
   ```

2. Delete the alert database from local storage:

   ```
   sudo rm /var/lib/nvsm/sqlite/nvsm.db
   ```

3. Start the NVSM service:

   ```
   sudo systemctl start nvsm
   ```

12.6. DGX A800 Station/Server: mig-parted config

12.6.1. Issue

DGX Station A800 is not currently supported in the all-balanced configuration of the default mig-parted config file.

12.6.2. Workaround

To add the A800 device ID to the all-balanced configuration:

1. Make a copy of the default configuration.
2. Add device ID 0x20F310DE to the device-filter of the all-balanced config.
3. Point `mig-parted apply` at this new file when selecting a config.

12.7. Erroneous Insufficient Power Error May Occur for PCIe Slots

12.7.1. Issue

Reported in release 4.99.9.

The DGX A100 server reports “Insufficient power” on PCIe slots when network cables are connected.

12.7.2. Explanation

This may occur with optical cables and indicates that the calculated power of the card + 2 optical cables is higher than what the PCIe slot can provide.

The message can be ignored.

12.8. Applications that call the cuCTXCreate API Might Experience a Performance Drop

12.8.1. Issue

Reported in release 5.0.

When some applications call cuCtxCreate, cuGLCtxCreate, or cut Destroy, there might be a drop in performance.

12.8.2. Explanation

This issue occurs with Ubuntu 22.04, but not with previous versions. The issue affects applications that perform graphics/compute interoperations or have a plugin mechanism for CUDA, where every plugin creates its own context, or video streaming applications where computations are needed. Examples include ffmpeg, Blender, simpleDrive Runtime, and cuSolverSp_LinearSolver.

This issue is not expected to impact deep learning training.

12.9. Incorrect nvidia-container-toolkit version after upgrade from 5.X to 6.0

12.9.1. Issue

Reported in release 6.0.

Following a release upgrade from DGX OS 5.X to DGX OS 6.X the version of nvidia-container-toolkit is not updated to the 6.X version.

12.9.2. Explanation

During the release upgrade process the older version of nvidia-container-toolkit has a higher priority than the newer version so it doesn't get updated.

The workaround is to perform an additional `sudo apt update` and `sudo apt upgrade` following the `nvidia-release-upgrade`.

12.10. UBSAN error and mstconfig stack dump in kernel logs at boot

12.10.1. Issue

Reported in release 6.0.

On boot an error message similar to `UBSAN: shift-out-of-bounds in /build/linux-nvidia-s96GJ3/linux-nvidia-5.15.0/debian/build/build-nvidia` may appear, followed by a stack trace.

12.10.2. Explanation

This warning is generate when mstconfig closes a device file. The warning can be safely ignored, and will be fixed in a future release.

12.11. The BMC Redfish interface is not active on first boot after installation

12.11.1. Issue

Reported in release 6.0.

When the system is first booted after installation the BMC Redfish network interface will be in a DOWN state and will not appear when running the `ifconfig` command.

12.11.2. Explanation

The interface is reconfigured after the system has already brought up interfaces, so it doesn't get automatically started.

Running the command `sudo netplan apply` will cause the interface to be started. It will also automatically be started on all subsequent boots even without running the additional `sudo netplan apply`

Chapter 13. DGX OS Connectivity Requirements

In a typical operation, DGX OS runs services to support typical usage of the DGX system.

Some of these services require network communication. The table below describes the port, protocol, direction, and communication purpose for the services. DGX administrators should consider their site-specific access needs and allow or disallow communication with the services as necessary.

13.1. In-Band Management, Storage, and Compute Networks

The following table provides information about the in-band management, storage, and compute networks.

Table 1: In-Band Management, Storage, and Compute Networks

Port (Protocol)	Direction	Use
22 (TCP)	Inbound	SSH
53 (UDP)	Outbound	DNS
80 (TCP)	Outbound	HTTP, package updates
111 (TCP)	Inbound/Outbound	RPCBIND, required by NFS
273 (TCP)		NVIDIA System Management Management
443 (TCP)	Outbound	For internet (HTTP/HTTPS) connection to NVIDIA GPU Cloud. If port 443 is proxied through a corporate firewall, then WebSocket protocol traffic must be supported
1883 (TCP)		Mosquitto Database (used by NVIDIA System Management)

13.2. Out-of-Band Management

The following table provides information about out-of-band management for your DGX system.

Table 2: Out-of-Band Management

Port (Protocol)	Direction	Use
443 (TCP)	Inbound	For BMC web services, remote console services, and CD-media service. If port 443 is proxied through a corporate firewall, then WebSocket protocol traffic must be supported.
623 (UDP)	Inbound	IPMI

Chapter 14. DGX Software Stack

14.1. NVIDIA DGX Software Packages

The following tables list the packages installed as part of the DGX Software Stack, broken out by metapackage name and platform.

Table 1: DGX A100, DGX Station A100, DGX A800, DGX Station A800, DGX H100

DGX A100 and DGX A800	**DGX Station A100 and DGX Station A800**	**DGX H100**
dgx-a100-system-configurations **dgx-a800-system-configurations**	**dgxstation-a100-system-configurations** **dgxstation-a800-system-configurations**	**dgx-h100-system-configurations**
dgx-release	dgx-release	dgx-release
nv-cpu-governor	nv-cpu-governor	nv-cpu-governor
nv-hugepage	nv-hugepage	nv-hugepage
nv-iommu-pt	nv-iommu-pt	nv-iommu-pt
nv-ipmi-devintf	nv-ipmi-devintf	nv-ipmi-devintf
nv-limits	nv-limits	nv-limits
nv-update-disable	nv-update-disable	nv-update-disable
nvgpu-services-list	nvgpu-services-list	nvgpu-services-list
nvidia-acs-disable	nvidia-acs-disable	nvidia-acs-disable
nvidia-crashdump	nvidia-crashdump	nvidia-crashdump
nvidia-esm-hook-epilogue	nvidia-esm-hook-epilogue	nvidia-esm-hook-epilogue
nvidia-fs-loader	nvidia-fs-loader	nvidia-fs-loader
nvidia-kernel-defaults	nvidia-kernel-defaults	nvidia-kernel-defaults
nvidia-nvme-smartd	nvidia-nvme-smartd	nvidia-nvme-smartd

continues on next page

Table 1 – continued from previous page

DGX A100 and DGX A800	DGX Station A100 and DGX Station A800	DGX H100
nvidia-pci-bridge-power	nvidia-pci-bridge-power	nvidia-pci-bridge-power
		nvidia-pci-norealloc
nvidia-redfish-config	nvidia-redfish-config	nvidia-redfish-config
nvidia-relaxed-ordering-gpu	nvidia-relaxed-ordering-gpu	nvidia-relaxed-ordering-gpu
nvidia-relaxed-ordering-nvme	nvidia-relaxed-ordering-nvme	nvidia-relaxed-ordering-nvme
dgx-a100-system-tools-extra **dgx-a800-system-tools-extra**	**dgxstation-a100-system-tools-extra** **dgxstation-a800-system-tools-extra**	**dgx-h100-system-tools-extra**
dgx-release	dgx-release	dgx-release
ipmitool	ipmitool	ipmitool
nv-common-apis	nv-common-apis	nv-common-apis
nv-env-paths	nv-env-paths	nv-env-paths
nvidia-mig-manager		nvidia-mig-manager
nvidia-raid-config	nvidia-raid-config	nvidia-raid-config
nvme-cli	nvme-cli	nvme-cli
tpm2-tools	tpm2-tools	tpm2-tools
dgx-a100-system-tools **dgx-a800-system-tools**	**dgxstation-a100-system-tools** **dgxstation-a800-system-tools**	**dgx-h100-system-tools**
msecli	msecli	msecli

Table 2: DGX-1, DGX-2, and DGX Station

DGX-1	DGX-2	DGX Station
dgx1-system-configurations	**dgx2-system-configurations**	**dgxstation-system-configurations**
dgx-release	dgx-release	dgx-release
nv-ast-modeset		
nv-cpu-governor	nv-cpu-governor	
nv-hugepage	nv-hugepage	nv-hugepage

continues on next page

Table 2 – continued from previous page

DGX-1	DGX-2	DGX Station
	nv-iommu-pt	
nv-ipmi-devintf	nv-ipmi-devintf	
nv-limits	nv-limits	nv-limits
nv-update-disable	nv-update-disable	nv-update-disable
nvgpu-services-list	nvgpu-services-list	nvgpu-services-list
	nvidia-acs-disable	
nvidia-crashdump	nvidia-crashdump	nvidia-crashdump
nvidia-esm-hook-epilogue	nvidia-esm-hook-epilogue	nvidia-esm-hook-epilogue
nvidia-fs-loader	nvidia-fs-loader	nvidia-fs-loader
nvidia-kernel-defaults	nvidia-kernel-defaults	nvidia-kernel-defaults
	nvidia-nvme-smartd	
nvidia-pci-bridge-power	nvidia-pci-bridge-power	
	nvidia-redfish-config	
	nvidia-relaxed-ordering-gpu	
	nvidia-relaxed-ordering-nvme	
dgx1-system-tools	**dgx2-system-tools**	**dgxstation-system-tools**
dgx-release	dgx-release	dgx-release
ipmitool	ipmitool	
nv-common-apis	nv-common-apis	nv-common-apis
nv-env-paths	nv-env-paths	nv-env-paths
	nvidia-raid-config	nvidia-raid-config
	nvme-cli	
	tpm-tools	
dgx1-system-tools-extra	**dgx2-system-tools-extra**	**dgxstation-system-tools-extra**
	msecli	
nvidia-raid-config		
storcli		

The following packages are installed by the **nvidia-mlnx-ofed-misc** metapackage:

- mlnx-fw-updater
- mlnx-pxe-setup
- nvidia-mlnx-config
- nvidia-peermem-loader

The following additional packages are part of the DGX Software Stack:

- nv-docker-options
- nvidia-logrotate
- nvidia-motd
- nvidia-ipmisol

The following table lists all packages that will be installed as part of the system configuration package with more details:

Package	Description	1	2	A	H
dgx-release	Release information	R	R	R	R
nv-ast-modeset	Disable the Aspeed display driver.It can cause issues with connected monitors. The AST2xxx is the BMC used in our servers.	R	R	R	R
nv-enable-nvme-hot-plug	Configure kernel parameters for NVMe hot plug (see also kernel section below).		R		
nv-hugepage	Sets the “transparent_hugepage=madvise” kernel parameter.	R	R	R	R
nv-iommu-pt	Sets iommu=pt for AMD Rome platforms.			R	R
nv-ipmi-devintf	Add the ipmi_devintf module for accessing the BMC using the ipmi tool.	R	R	R	R
nv-limits	Increase the process resource limits for users (ulimits nofile 50000)	R	R	R	R
nv-update-disable	Disable automatic system upgrades. Users need to explicitly upgrade their systems using apt.	R	R	R	R
nvgpu-services-list	Lists GPU-consuming services in JSON format, such as DCGM or NVSM, and required by the firmware update mechanism.	R	R	R	R
nvidia-acs-disable	Disables the PCIe ACS capability to allow for better GPU-direct performance in bare-metal use cases on DGX A100 and DGX H100.			R	R
nvidia-crashdump	Tools to manage kernel crash dumps. They are disabled by default.	R	R	R	R
nv-docker-options	Increases SHMEM and other resources.	R	R	R	R
nvidia-ipmisol	Enables serial output through the BMC using Serial Over LAN (SOL)	O	O	O	O
nvidia-kernel-defaults	Disable ARP for security improvements net.ipv4.conf	R	R	R	R
nvidia-logrotate	Modify the logrotate configuration	O	O	O	O
nvidia-motd	Modify message-of-the-day (MOTD) to display NVSM health monitoring alerts and release information.	O	O	O	O
nvidia-nvme-smartd	Enables SMART monitoring on NVME devices. By default, smartd will skip NVME devices.		R	R	R

continues on next page

Table 3 – continued from previous page

Package	Description	1	2	A	H
nvidia-pci-bridge-power	Sets the bridge power control setting to "on" for all PCI bridges.	R	R	R	R
nvidia-relaxed-ordering-gpu	Sets a reg-key to enable PCIe relaxed-ordering in the GPUs			R	R
nvidia-relaxed-ordering-nvme	Installs a script that users can call to enable relaxed-ordering in NVME devices.			R	R
nvidia-redfish-config	Configures the redfish interface with an interface name and IP address. The interface name is "bmc _redfish0", while the IP address is read from DMI type 42.			R	R

Legend:

1 DGX-1

2 DGX-2

A DGX A100

H DGX H100

R Required package

O Optional package

14.2. DGX Kernel Parameters

Kernel Parameter	Description	Package
ast.modeset=0	Disable the Aspeed display driver. The AST2xxx is the BMC used in our servers. [DGX-1, DGX-2, DGX A100, DGX Station A100, DGX H100]	nv-ast-modeset
crashkernel=1G-:0M	Don't reserve any memory for crash dumps (when crash is disabled = default)	nvidia-crashdump
crashkernel=1G-:512M	Reserve 512MB for crash dumps (when crash is enabled)	nvidia-crashdump
pci=realloc=on	Allows kernel to reallocate PCI resources if allocations done by BIOS are insufficient. This and pcie_ports=native are both required for NVME hot-plug on DGX2.	nv-enable-nvme-hot-plug
pcie_ports=native	Use Linux native services for PME, AER, DPC, PCIe hotplug. I.e. not firmware first. This and pci=realloc=on are both required for NVME hot-plug on DGX2.	nv-enable-nvme-hot-plug

continues on next page

Table 4 – continued from previous page

Kernel Parameter	Description	Package
transparent_hugepage=madvise	Disable huge pages system-wide and only enable them inside MADV_HUGEPAGE madvise regions to prevent applications from allocating more memory resources than necessary.	nv-hugepage
iommu=pt	Enable pass through mode only and disable DMA translations. This enables optimizations for the CPU inside the DGX A100.	nv-iommu-pt
console=ttyS1,115200n8	Set console to serial port 1, using 115200 baud, no parity, 8 data bits [DGX-2 and DGX H100]	nvidia-ipmisol
console=ttyS0,115200n8	Set console to serial port 0, using 115200 baud, no parity, 8 data bits	nvidia-ipmisol

Chapter 15. PXE Boot Setup

The DGX-Server UEFI BIOS supports PXE boot. Several manual customization steps are required to get PXE to boot the Base OS image.

> **Caution:** This document is meant to be used as a reference. Explicit instructions are not given to configure the DHCP, FTP, and TFTP servers. It is expected that the end user's IT team will configure them to fit within their company's security guidelines.

15.1. Pre-requisites

- **TFTP server is setup**
  - TFTP is configured to serve files from /local/tftp/
- **HTTP server is setup**
  - HTTP is configured to serve files from /local/http/
- DHCP server is setup
- IP address is <FTP IP>
- Fully qualified host is <FTP host>

This document is intended to provide detailed step-by-step instructions on how to set up a PXE boot environment for DGX systems. The examples are based on a DGX A100. There are several major components of the solution:

- DHCP server: dnsmasq is used in this doc.
- TFTP server: dnsmasq is also used as a TFTP server.
- HTTP server: HTTP server is used to transfer large files, such is ISO image and initrd. Alternatively, FTP can also be used for this purpose. HTTP in used this doc.
- Syslinux: Linux bootloader software package. section

15.2. Overview of the PXE Server

The PXE Server is divided up into three general areas:

- Bootloader (grub)
- TFTP contents (the kernel and initrd)
- HTTP contents (the ISO image)

The rough directory structure on the TFTP and HTTP server will look like this:

```
/local/
   http/
      base_os_6.0.0/
         base_os_6.0.0.iso
   tftp/
      grub2/
         base_os_6.0.0/
            vmlinuz
            initrd
      grub.cfg
      bootx64.efi
```

The tftp-server (controlled by the xinetd service and configuration found in /etc/xinetd.d/tftp) points to the /local/tftp directory for when the system PXE boots. TFTP is what transfers the bootx64.efi file that is designated in the DHCP server's dhcpd.conf file (see Configure your DHCP server). By default after the bootx64.efi is booted it looks for a grub2/grub.cfg file with the menu options for booting further. That config file will look for its kernel and initrd files relative to the tftp directory.

The following steps will assume the DHCP and PXE servers are configured to use the above directory structure. The lab admin, or whoever is in charge of deploying the PXE environment, should change the directory names and structure to fit their infrastructure.

15.2.1. Configuring the HTTP File Directory and ISO Image

Place a copy of the BaseOS 6.0.0 ISO in `/local/http/base_os_6.0.0/`

15.3. Mount the BaseOS 6.0.0 ISO

Assume your mount point is "/mnt":

```
sudo mount -o loop /local/http/base_os_6.0.0/base_os_6.0.0.iso /mnt
```

Copy the kernel and initrd from the ISO to the TFTP Directory

```
cp /mnt/casper/vmlinuz /local/tftp/grub2/base_os_6.0.0/
cp /mnt/casper/initrd /local/tftp/grub2/base_os_6.0.0/
```

15.4. Configure the TFTP directory

Mount the BaseOS 6.0.0 ISO Assume your mount point is "/mnt":

```
sudo mount -o loop /local/http/base_os_6.0.0/base_os_6.0.0.iso /mnt
```

Copy the kernel and initrd from the ISO to the TFTP Directory

```
cp /mnt/casper/vmlinuz /local/tftp/grub2/base_os_6.0.0/
cp /mnt/casper/initrd /local/tftp/grub2/base_os_6.0.0/
```

Download GRUB Packages For x86_64:

Download the relevant grub packages with the correct architecture specified:

```
wget http://mirror.centos.org/centos/7/updates/x86_64/Packages/grub2-efi-x64-2.02-0.
↪87.el7.centos.7.x86_64.rpm
wget http://mirror.centos.org/centos/7/os/x86_64/Packages/shim-x64-15-8.el7.x86_64.rpm
```

Unpack the RPMs with the following commands:

```
rpm2cpio grub2-efi-x64-2.02-0.86.el7.centos.x86_64.rpm | cpio -idmv
rpm2cpio shim-x64-15-8.el7.x86_64.rpm | cpio -idmv
```

Copy the following binaries from the unpacked RPMs to `/local/tftp/grub2/`:

```
shim.efi
shimx64.efi
grubx64.efi
```

Make a copy of `shimx64.efi` in /local/tftp/grub2/ and name the copy `bootx64.efi`

For arm64:

Download the relevant grub package with the correct architecture specified:

```
wget http://mirror.centos.org/altarch/7/updates/aarch64/Packages/grub2-efi-aa64-2.02-
↪0.87.el7.centos.7.aarch64.rpm
```

Unpack the RPMs with the following commands:

```
rpm2cpio grub2-efi-aa64-2.02-0.87.0.1.el7.centos.9.aarch64.rpm | cpio -idmv
```

Copy the following binary from the unpacked RPM to /local/tftp/grub2/:

```
grubaa64.efi
```

Create the grub configuration file

The contents of the /local/tftp/grub2/grub.cfg file should look something like:

```
set default=0
set timeout=-1
insmod all_video

menuentry 'Install BaseOS 6.0.0' {
  linuxefi /grub2/base_os_6.0.0/vmlinuz fsck.mode=skip autoinstall ip=dhcp url=http://
↪<Server IP>/base_os_6.0.0/base_os_6.0.0.iso nvme-core.multipath=n nouveau.modeset=0
```

(continues on next page)

(continued from previous page)

```
  initrdefi /grub2/base_os_6.0.0/initrd
}
```

Note:

- NOTE 1: The vmlinuz and initrd files are specified relative to the TFTP root - in this example, relative to /local/tftp/. The location of the ISO is relative to the HTTP root - in this example, /local/http/.
- NOTE 2: The kernel boot parameters should match the contents of the corresponding ISO's boot menu, found in /mnt/boot/grub/grub.cfg.
- NOTE 3: In some cases, the transfer of the initrd can time out over FTP. A work around for this is to host the requisite files – initrd, vmlinuz, and the ISO – over HTTP instead. Hosting these over HTTP makes the transfer speedier and more reliable. In this example, we assume that the HTTP server is hosted from /local/http. We will need to copy these files to this location:

```
/local/
   http/
      base_os_6.0.0/
         base_os_6.0.0.iso
         vmlinuz
         initrd
   tftp/
      grub2/
         grub.cfg
         bootx64.efi
         grubaa64.efi
```

When configured this way, the grub.cfg file may look something like:

```
set default=0
set timeout=-1
insmod all_video

menuentry 'Install BaseOS 6.0.0' {
    linuxefi (http,<HTTP Server IP>)/base_os_6.0.0/vmlinuz fsck.mode=skip autoinstall
↪ip=dhcp url=http://<Server IP>/base_os_6.0.0/base_os_6.0.0.iso nvme-core.
↪multipath=n nouveau.modeset=0
    initrdefi (http,<HTTP Server IP>)/base_os_6.0.0/initrd
}
```

Useful parameters for configuring your system's network interfaces:

`ip=dhcp`: tells the initramfs to automatically configure the system's interfaces using DHCP. - If only one interface is connected to the network then this should be enough. - If multiple interfaces are connected to the network, then it will go with the first one that receives a reply.

15.5. Parameters unique to the Base OS installer

- `rebuild-raid` tells the installer to rebuild the data RAID if specified. Installs from the factory should always specify this, but it is optional otherwise
- `md5checkdisc` will not perform an installation when this is specified. It will simply unpack the ISO and check that its contents match with what's described in md5sum.txt
- `offwhendone` powers off the system after the installation. Otherwise, we reboot when done. Factory installs will specify this.
- `nooemconfig` skip oemconfig and create default user "nvidia", seeding initial password. Used for touchless install in PXE install or automatic VM creation/installation.
- `force-ai` allows users to supply their own autoinstall file. If the networking is set up, then users can provide a URL. Otherwise, this has to be one that exists in the installer.

For example:

```
force-ai=/ai/dgx2-ai.yaml
force-ai=http://your-server.com/your-ai.yaml
```

Note: Refer to the note the *Autoinstall Customizations* section for special formatting considerations when using custom autoinstall files along with the `force-ai` parameter.

15.6. Configure DHCP

The DHCP server is responsible for providing the IP address of the TFTP server and the name of the bootloader file in addition to the usual role of providing dynamic IP addresses. The address of the TFTP server is specified in the DHCP configuration file as "next-server", and the bootloader file is specified as "filename". The architecture option can be used to detect the architecture of the client system and used to serve the correct version of the grub bootloader (x86, ia32, arm, etc).

An example of the PXE portion of dhcpd.conf is:

```
next-server <TFTP_Server_IP>;

# x86 UEFI
if option arch = 00:06 {
   filename "grub2/bootx64.efi";
# x64 UEFI
} else if option arch = 00:07 {
   filename "grub2/bootx64.efi";
# ARM 64-bit UEFI
} else if option arch = 00:0b {
   filename "grub2/grubaa64.efi";
} else {
   filename "pxelinux.0";
}
```

15.7. Optional: Configure CX-4/5/6/7 cards to PXE boot

DGX-Servers may also PXE boot using the MLNX CX-4/5/6 cards. If you are logged into the DGX-Server host OS, and running DGX Base OS 4.4 or later, then you can perform this section's steps using the "/usr/sbin/mlnx_pxe_setup.bash" tool, which will enable the UEFI PXE ROM of every MLNX Infiniband device found.

Otherwise, proceed with the manual steps below.

15.8. Query UEFI PXE ROM state

In order to PXE boot from the MLNX CX-4/5/6/7 cards, you must first enable the UEFI PXE ROM of the card you wish to PXE boot from because it is disabled by default. This needs to be performed from the DGX Server host OS itself, it can't be done remotely.

DGX OS 6 provides the in-tree OFED stack by default, but users may optionally install MOFED on top. The commands used to query and enable the UEFI PXE ROM will differ based on whether you are using the in-tree OFED vs. MOFED stack.

15.9. MOFED Instructions

To determine the device name and current configurations of the MLNX CX cards, run "sudo mlxconfig query":

```
user@dgx1server$ sudo mlxconfig query

Device #1:
----------

Device type:    ConnectX4
Name:           MCX455A-ECA_Ax
Description:    ConnectX-4 VPI adapter card; EDR IB (100Gb/s) and 100GbE; single-port
↪QSFP28; PCIe3.0 x16; ROHS R6
Device:         /dev/mst/mt4115_pciconf3

Configurations:                              Next Boot
         ...
         ...
         EXP_ROM_UEFI_x86_ENABLE             False(0)
         ...
         ...

In-tree OFED Instructions
To determine the device name and current configurations of the MLNX CX cards, run
↪"sudo mstconfig query":

user@dgxserver:~$ sudo mstconfig query
```

(continues on next page)

(continued from previous page)

```
Device #1:
----------

Device type:    ConnectX7
Name:           MCX755206AS-NEA_Ax
Description:    NVIDIA ConnectX-7 VPI adapter card; 400Gb/s IB and 200GbE; dual-port
↪QSFP; PCIe 5.0 x16 with x16 PCIe extension option; dual slot; secure boot; no
↪crypto; tall bracket for Nvidia DGX storage
Device:         /sys/bus/pci/devices/0000:b1:00.0/config

Configurations:                              Next Boot
         ...
         ...
         EXP_ROM_UEFI_x86_ENABLE             False(0)
         ...
         ...

Enable UEFI PXE ROM
The "EXP_ROM_UEFI_x86_ENABLE" configuration must be set to True(1) for the MLNX CX
↪card that you wish to PXE boot from, and reboot.

MOFED Instructions

user@dgx1server$ sudo mlxconfig -y -d /dev/mst/mt4115_pciconf3 set EXP_ROM_UEFI_x86_
↪ENABLE=1
user@dgx1server$ sudo reboot

Upon reboot, confirm the configuration was set.

user@dgx1server$ sudo mlxconfig query

Device #1:
----------

Device type:    ConnectX4
Name:           MCX455A-ECA_Ax
Description:    ConnectX-4 VPI adapter card; EDR IB (100Gb/s) and 100GbE; single-port
↪QSFP28; PCIe3.0 x16; ROHS R6
Device:         /dev/mst/mt4115_pciconf3

Configurations:                              Next Boot
         ...
         ...
         EXP_ROM_UEFI_x86_ENABLE             True(1)
         ...
         ...
In-tree OFED Instructions
user@dgxserver:~$ sudo mstconfig -y -d b1:00.0 set EXP_ROM_UEFI_x86_ENABLE=1
user@dgx1server$ sudo reboot

Upon reboot, confirm the configuration was set.

user@dgxserver:~$ sudo mstconfig query

Device #1:
```

(continues on next page)

(continued from previous page)

```
----------

Device type:    ConnectX7
Name:           MCX755206AS-NEA_Ax
Description:    NVIDIA ConnectX-7 VPI adapter card; 400Gb/s IB and 200GbE; dual-port
↪QSFP; PCIe 5.0 x16 with x16 PCIe extension option; dual slot; secure boot; no
↪crypto; tall bracket for Nvidia DGX storage
Device:         /sys/bus/pci/devices/0000:b1:00.0/config

Configurations:                              Next Boot
         ...
         ...
         EXP_ROM_UEFI_x86_ENABLE             True(1)
         ...
         ...
```

15.10. Optional: Configure the DGX-Server to PXE boot automatically

15.10.1. Add PXE to the top of the UEFI boot order

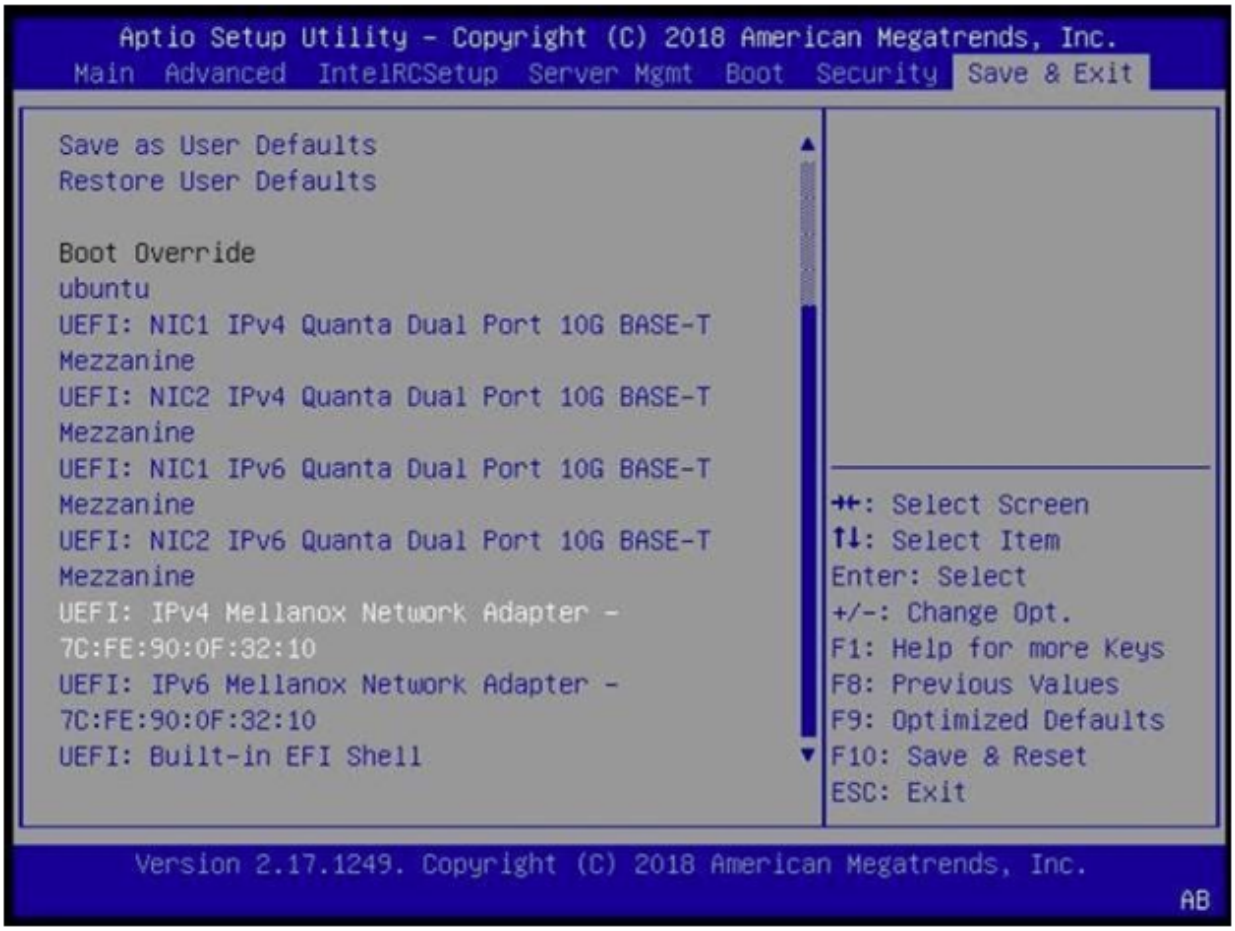
On systems that have a BMC, you can specify the DGX-Server to PXE boot by adding it to the top of the UEFI boot order. This may be done out-of-band via IPMI.

```
ipmitool -I lanplus -H <DGX_BMC_IP> -U <ADMIN> -P <PASSWORD> chassis bootdev pxe
↪options=efiboot
```

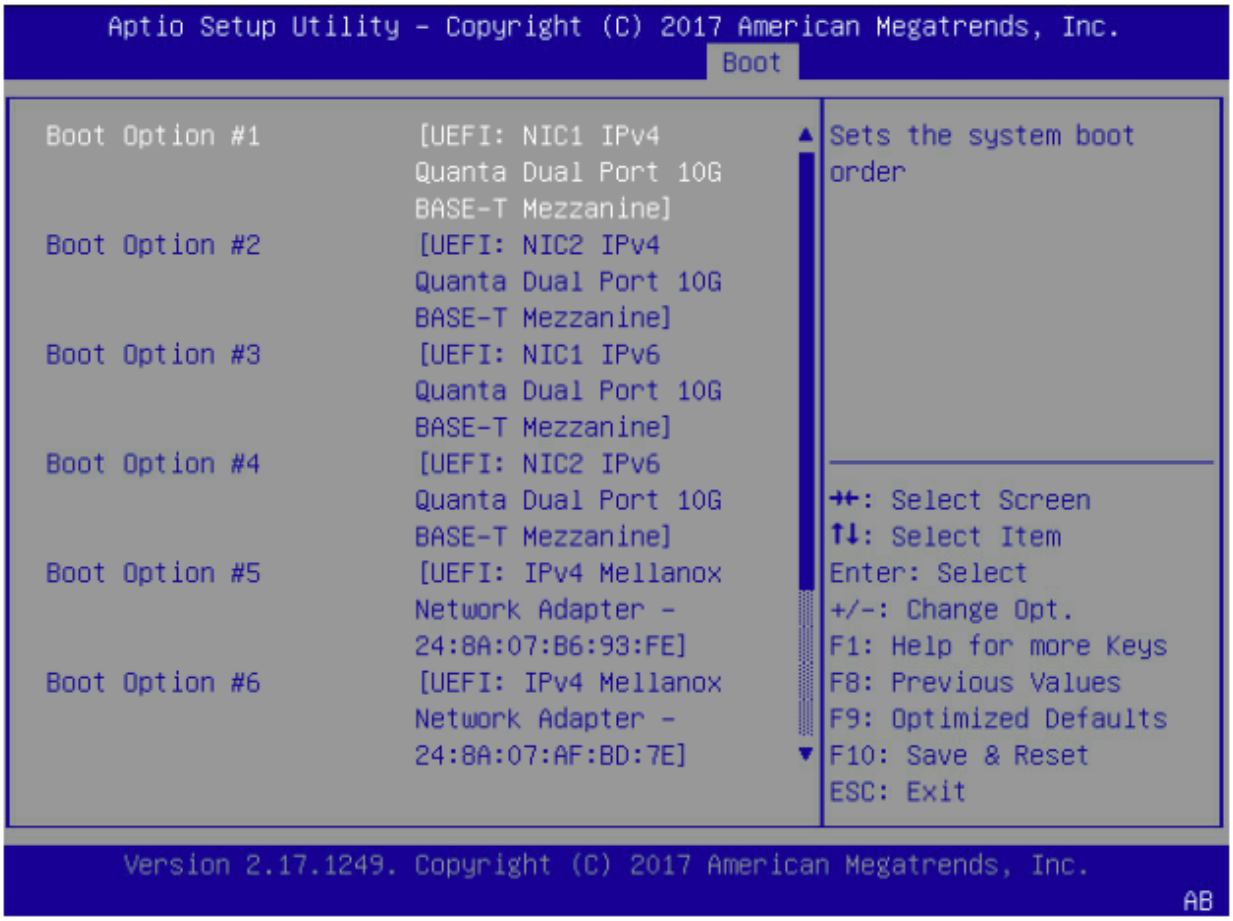
Note: that this only sets the DGX-Server to PXE boot, but doesn't specify the order of network devices to attempt PXE from. This is a limitation of our current UEFI and BMC FW. See the following section to specify the network device boot order.

15.11. Configure network boot priorities

The UEFI Network Drive BBS Priorities allows you to specify the order of network devices to PXE boot from. To modify this, you must reboot your DGX-Server and enter the UEFI boot selection menu by pressing "F2" or "Del" when you see the splash screen. Navigate to the "Boot" menu, and then scroll down to the "UEFI NETWORK Drive BBS Priorities"

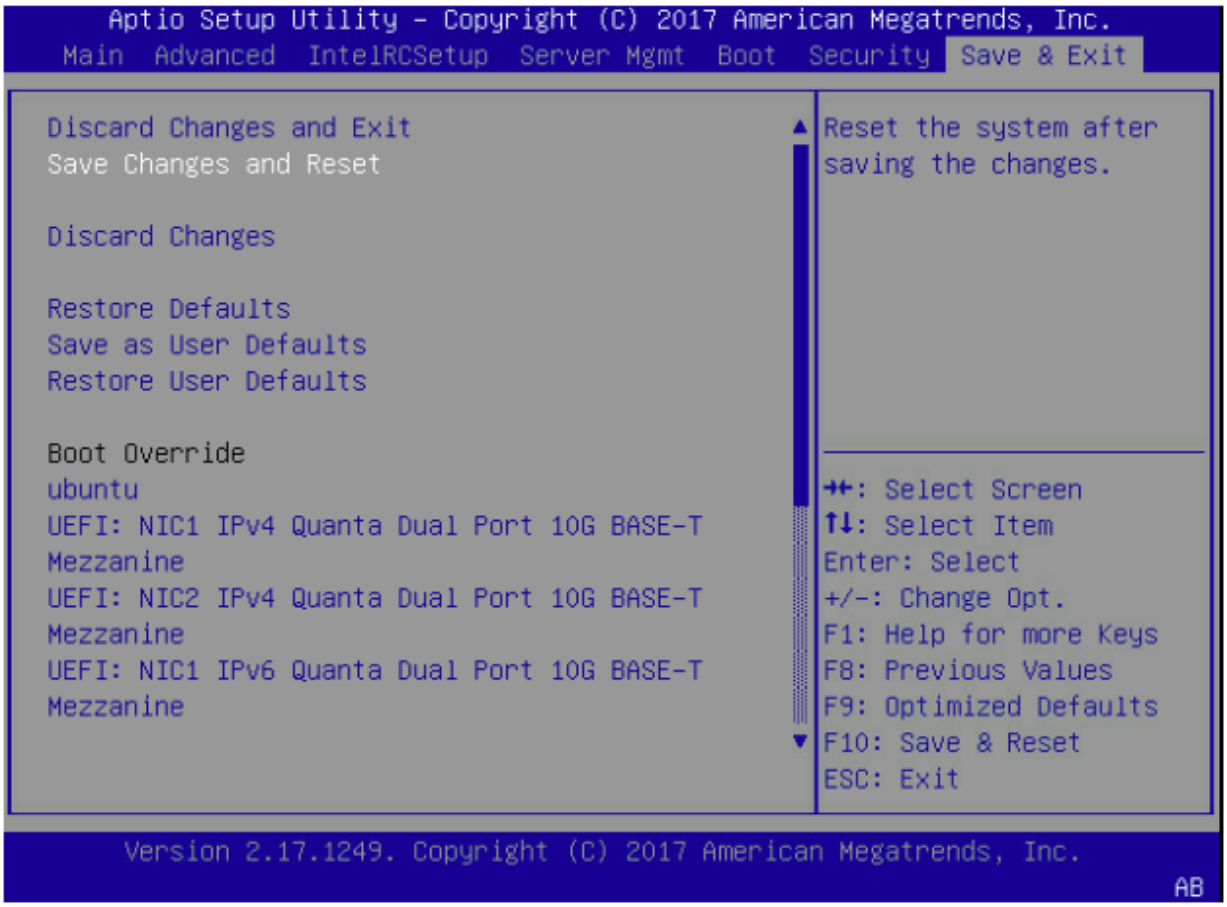


Configure the order of devices to attempt network boots from using this menu.



Save and Exit.

Once you've finished ordering the network boot priorities, then save your changes and reset.



15.12. Make the DGX-Server PXE boot

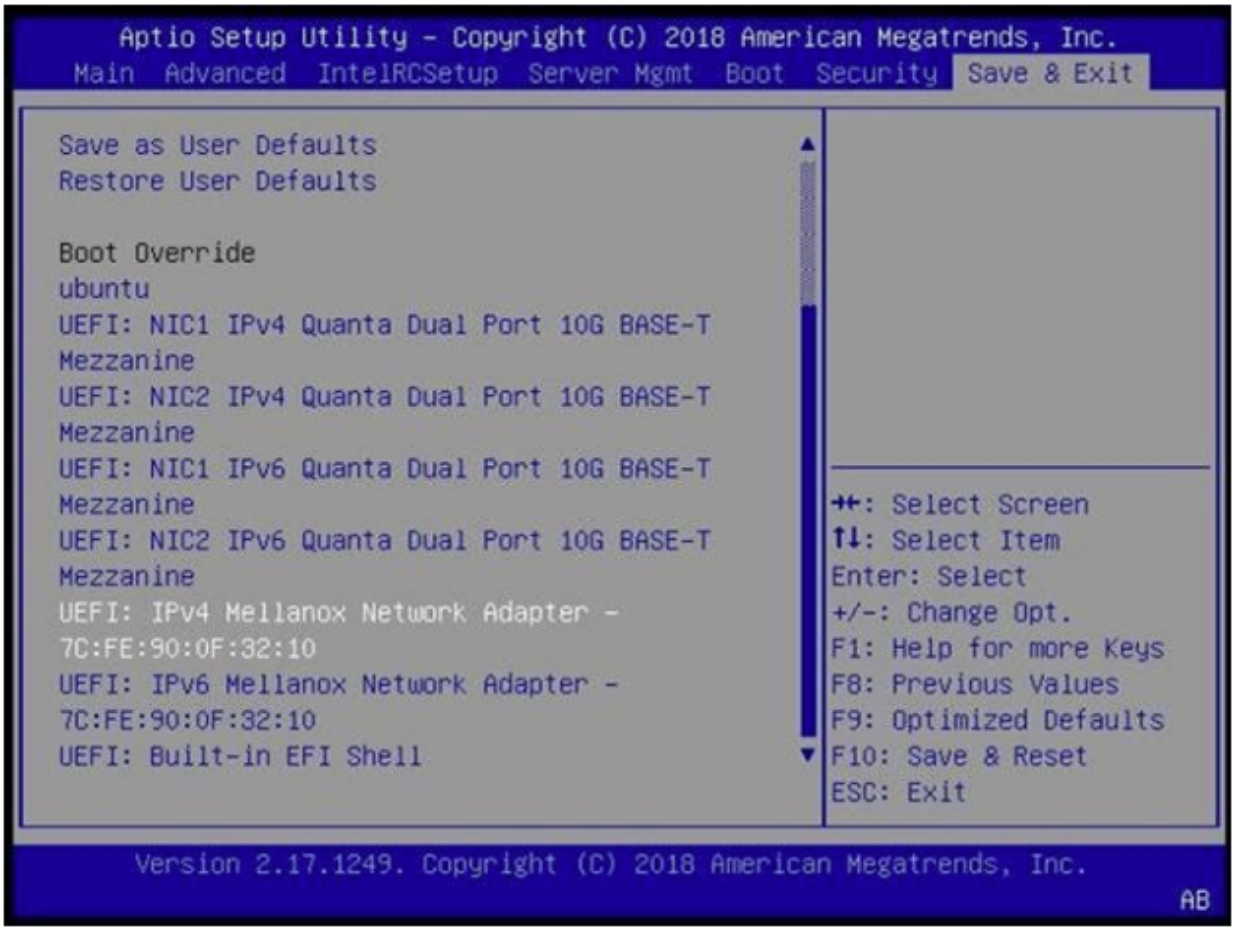
15.12.1. Automated PXE Boot Process

If you've followed the optional steps above, then you can simply reboot, and UEFI will attempt PXE boot using the devices in order specified in the Network Drive BBS Priorities list.

15.12.2. Manual PXE Boot Process

If you want to manually trigger the PXE boot, then reboot your DGX-Server and enter the UEFI boot selection menu by pressing "F2" or "Del" when you see the splash screen.

Navigate to the "Save & Exit" menu, scroll down to the Boot Override section, and choose the appropriate network port to boot from. The MLNX cards will only appear if you enabled the UEFI PXE ROM of that particular card.



Alternatively, you can press “F12” at the SBIOS splash screen, and SBIOS will iterate thru each NIC and try PXE on each one. The order of the NICs attempted is specified by the Network Drive BBS Priorities.

15.13. Other IPMI boot options

For more information about specifying boot order via IPMI, see the “ipmitool” man page, and look at the “chassis” command, and “bootdev” subcommand: https://linux.die.net/man/1/ipmitool

For more information about the IPMI specification, refer to Intelligent Platform Management Interface Specification v2.0 rev. 1.1.

15.14. Autoinstall Customizations

The Base OS 6.x installer has undergone major changes compared to the Base OS 5.x installer. The Base OS 6.x installer now uses subiquity, which supports autoinstall instead of curtin.

Autoinstall and curtin both serve similar purposes but have some syntactic differences – be aware of these when porting old curtin files. There are many autoinstall files that users can reference inside the Base OS 6.x ISO; these are contained in:

```
casper/ubuntu-server-minimal.ubuntu-server.installer.kernel.nvidia.squashfs
```

Users can mount the ISO and then mount this squashfs to view the many autoinstall files that are packed within:

```
mkdir -p /tmp/iso_mnt
mkdir -p /tmp/squash_mnt
sudo mount /path/to/DGXOS-<version>-<date>.iso /tmp/iso_mnt/
sudo mount /tmp/iso_mnt/casper/ubuntu-server-minimal.ubuntu-server.installer.kernel.
↪nvidia.squashfs /tmp/squash_mnt/
find /tmp/squash_mnt/ai/ -name '*.yaml'
```

For some deployments, users may want to use their own autoinstall files. This section will describe some sections contained in the built-in autoinstall files as well as how to perform some common customizations.

Note: The installer expects a unified autoinstall file rather than the typical split vendor/user/metadata format. This means that the user-supplied autoinstall file will need to account for some formatting differences – namely, the `autoinstall:` keyword needs to be dropped and the indentations adjusted accordingly:

```
#
# typical user-data file
#
#cloud-config
autoinstall:
  version: 1
  identity:
    realname: 'DGX User'
    username: dgxuser
    password: '$6$g3vXaGj.MQpP/inN$l6.
↪JtAueRAfMtQweK7qASjxXiEX8Vue3CvRcwON81Rt9BJmlEQKtnfOVSnCqHrTsy88PbMDDHq6k.iM6PWfHr1'

#
# unified autoinstall file
#
version: 1
identity:
  realname: 'DGX User'
  username: dgxuser
  password: '$6$g3vXaGj.MQpP/inN$l6.
↪JtAueRAfMtQweK7qASjxXiEX8Vue3CvRcwON81Rt9BJmlEQKtnfOVSnCqHrTsy88PbMDDHq6k.iM6PWfHr1'
```

15.15. NVIDIA-Specific Autoinstall Variables

The autoinstall files contained in the ISO are platform-specific, and serve as a good starting point for custom versions. Many of them contain variables, prefixed with `CHANGE_` which will be substituted by the installer:

- `CHANGE_STORAGE_REG` This gets removed and uncommented when the boot parameter ai-encrypt-root is not present. Uncommenting this stanza results in the standard disk partitioning scheme without LUKS encryption.
- `CHANGE_STORAGE_ENC` This gets removed and uncommented when the boot parameter "ai-encrypt-root" is present. Uncommenting this stanza results in an encrypted root partition.
- **`CHANGE_BOOT_DISK_NAME_x` This is a disk-name, without the "/dev" prefix. There may be multiple ones (e.g.**

  Note: The installer will find the appropriate disk name to substitute here. Alternatively, the "force-bootdisk" parameter can be used to specify the disk name(s).

- `CHANGE_BOOT_DISK_PATH_x` This is the same as the CHANGE_BOOT_DISK_NAME_x variable above, except that it is prefixed with "/dev/".
- `CHANGE_DESC_PLATFORM` The installer will substitute this with a platform-specific descriptive name.
- `CHANGE_SERIAL_NUMBER` The installer will substitute this with the serial number reported by dmidecode.
- `CHANGE_INSTALL_PKGS` The installer will substitute this value with a list of packages specific to the platform. The lists of packages are specified by the `*-pkgs` files in the `squashfs`
- `CHANGE_REBUILD_RAID` This gets replaced with either "true" or "false" based on whether or not the "rebuild-raid" boot parameter is present.
- `CHANGE_IPMISOL` This gets replaced with either "true" or "false" based on whether or not the "ai-encrypt-root" boot parameter is present. When we set the system up with encryption, we also undo the IPMI serial-over-LAN configuration to ensure that the LUKS passphrase prompt shows up on the console rather than the serial-over-LAN interface.

Attention: While it is possible to replace these values on your own, we strongly recommend letting the installer handle this.

15.16. Common Customizations

In this section, we will describe some common customizations that may be useful in more custom deployments.

15.17. Network Configuration

To configure the network at install time, you can add a “network” section to your autoinstall file. In this example we will create a netplan configuration file that sets the enp1s0f0 interface to use DHCP:

```
network:
 version: 2
 ethernets:
   enp1s0f0:
    dhcp4: yes
```

15.18. Creating a User

To create a user at install time, you can add an “identity” section to your autoinstall file. In this example, we set the system’s hostname to “dgx” and create a user with the name/password of nvidia/nvidia.

```
#  To generate an encrypted password:
#    printf '<plaintext_password>' | openssl passwd -6 -stdin
#
#  For example:
#    printf 'nvidia' | openssl passwd -6 -stdin
identity:
  hostname: dgx
  password: $6$8fqF54QDoaLMtDXJ
↪$J02iNH1xW9hHtzH6APpUX4X4HkRx2xY2ZKy9DQpGOQhW7OOuTk3DwHr9FnAAh1JIyqn3L277Jy9MEzW4MyVsV0
  username: nvidia
```

There are many more examples documented in the Ubuntu autoinstall reference

Chapter 16. Air-Gapped Installations

For security purposes, some installations require that systems be isolated from the internet or outside networks.

An air-gapped system is not connected to an unsecured network, such as the public Internet, to an unsecured LAN, or to other computers that are connected to an unsecured network. The default mechanisms to update software on DGX systems and loading container images from the NGC Container Registry require an Internet connection. On an air-gapped system, which is isolated from the Internet, you must provide alternative mechanisms to update software and load container images.

Since most DGX software updates are completed through an over-the-network process with NVIDIA servers, this section explains how updates can be made when using an over-the-network method is not an option. It also includes a process to install Docker containers.

Here are the methods you can use:

- Download the ISO image, copy it to removable media and then reimage the DGX System from the media.

  This method is available only for software versions that are available as ISO images for download. For details, see *Reimaging the System* This section provides information about how to install the DGX OS.

- Update the DGX software by performing a network update from a local repository.

  This method is available only for software versions that are available for over-the-network updates.

Note: If you are using `apt-mirror` version 0.5.4-1 or higher, the following method described in this topic does not successfully update the repositories. To work around this issue, refer to *Errors Occur When Loading Mirrored Repositories on Air-Gapped Systems* in the *Known Issues* section for an alternative method.

16.1. Creating a Local Mirror of the NVIDIA and Canonical Repositories

Here are the steps to download the necessary packages to create a mirror of the repositories that are needed to update NVIDIA DGX systems. For more information on DGX OS versions and the release notes available, refer to *Release Notes*

Note: These procedures apply only to upgrades in the same major release, such as from 5.x to 5.y. The steps do not support upgrades across major releases, such as from DGX OS 5 to DGX OS 6.

1. Identify the sources that correspond to the public NVIDIA and Canonical repositories that provide updates to the DGX OS.

   You can identify these sources from the `/etc/apt/sources.list` file and the contents of the `/etc/apt.sources.list.d/` directory, or by using [System Settings, Software & Updates].

2. Create and maintain a private mirror of the repository sources that you identified in the previous step.
3. Update the sources that provide updates to the DGX system to use your private repository mirror instead of the public repositories.

To update these sources, modify the `/etc/apt/sources.list` file and the contents of `/etc/apt.sources.list.d/` directory.

16.2. Creating the Mirror of the Repositories

The instructions in this section are to be performed on a system with internet access.

- A system installed with Ubuntu OS is needed to create the mirror because there are several Ubuntu tools that need to be used.
- You must be logged in to the system installed with Ubuntu OS as an administrator user because this procedure requires `sudo` privileges.
- The system must contain enough storage space to replicate the repositories to a file system. The space requirement could be as high as 250 GB.
- An efficient way to move large amount of data is needed, for example, shared storage in a DMZ, or portable USB drives that can be brought into the air-gapped area.

  The data will need to be moved to the systems that need to be updated. Make sure that any portable drives are formatted using ext4 or FAT32.

To create the mirror:

1. Ensure that the storage device is attached to the system with network access and identify the mount point of the device.

   Here is a sample mount point that was used in these instructions:

   ```
   /media/usb/repository
   ```

2. Install the `apt-mirror` package.

   ```
   sudo apt update
   ```

   ```
   sudo apt install apt-mirror
   ```

3. Change the ownership of the target directory to the `apt-mirror` user in the `apt-mirror` group.

   ```
   sudo chown apt-mirror:apt-mirror /media/usb/repository
   ```

The target directory must be owned by the user `apt-mirror` or the replication will not work.

4. Configure the path of the destination directory in `/etc/apt/mirror.list` and use the included list of repositories below to retrieve the packages for both Ubuntu base OS and the NVIDIA DGX OS packages.

```
############# config ##################
#
set base_path /media/usb/repository #/your/path/here
#
# set mirror_path $base_path/mirror
# set skel_path $base_path/skel
# set var_path $base_path/var
# set cleanscript $var_path/clean.sh
# set defaultarch <running host architecture>
# set postmirror_script $var_path/postmirror.sh
set run_postmirror 0
set nthreads 20
set _tilde 0
#
############# end config ##############
# Standard Canonical package repositories:
deb http://security.ubuntu.com/ubuntu jammy-security main multiverse universe
↪restricted
deb http://archive.ubuntu.com/ubuntu/ jammy main multiverse universe restricted
deb http://archive.ubuntu.com/ubuntu/ jammy-updates main multiverse universe
↪restricted
#
deb-i386 http://security.ubuntu.com/ubuntu jammy-security main multiverse
↪universe restricted
deb-i386 http://archive.ubuntu.com/ubuntu/ jammy main multiverse universe
↪restricted
deb-i386 http://archive.ubuntu.com/ubuntu/ jammy-updates main multiverse universe
↪restricted
#
# CUDA specific repositories:
deb http://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86_64/ /
#
# DGX specific repositories:
deb http://repo.download.nvidia.com/baseos/ubuntu/jammy/x86_64/ jammy common dgx
deb http://repo.download.nvidia.com/baseos/ubuntu/jammy/x86_64/ jammy-updates
↪common dgx
#
deb-i386 http://repo.download.nvidia.com/baseos/ubuntu/jammy/x86_64/ jammy common
↪dgx
deb-i386 http://repo.download.nvidia.com/baseos/ubuntu/jammy/x86_64/ jammy-
↪updates common dgx
# Clean unused items
clean http://archive.ubuntu.com/ubuntu
clean http://security.ubuntu.com/ubuntu
```

5. Run `apt-mirror` and wait for it to finish downloading content.

   This will take a long time depending on the network connection speed.

```
sudo apt-mirror
```

6. Eject the removable storage with all packages.

```
sudo eject /media/usb/repository
```

16.3. Configuring the Target Air-Gapped System

Here are the steps that explain how you can configure a target air-gapped DGX OS 6 system.

The instructions in this section are to be performed on the target air-gapped DGX system.

- The target air-gapped DGX system is installed, has gone through the first boot process, and is ready to be updated with the latest packages.
- The USB storage device on which the mirrors were created is attached to the target DGX system.

There are other ways to transfer the data that are not covered in this document as they will depend on the data center policies for the air-gapped environment.

1. Mount the storage device on the air-gapped system to `/media/usb/repository` for consistency2. Configure the apt command to use the file system as the repository in the file `/etc/apt/sources.list` by modifying the following lines.

```
deb file:///media/usb/repository/mirror/security.ubuntu.com/ubuntu jammy-security
↪main multiverse universe restricted
deb file:///media/usb/repository/mirror/archive.ubuntu.com/ubuntu/ jammy main
↪multiverse universe restricted
deb file:///media/usb/repository/mirror/archive.ubuntu.com/ubuntu/ jammy-updates
↪main multiverse universe restricted
```

2. Configure `apt` to use the NVIDIA DGX OS packages in the `/etc/apt/sources.list.d/dgx.list` file.

```
deb file:///media/usb/repository/mirror/repo.download.nvidia.com/baseos/ubuntu/
↪jammy/x86_64/ jammy main dgx
deb file:///media/usb/repository/mirror/repo.download.nvidia.com/baseos/ubuntu/
↪jammy/x86_64/ jammy-updates main dgx
```

3. Configure [apt] to use the NVIDIA CUDA packages in the `/etc/apt/sources.list.d/cuda-compute-repo.list` file.

```
deb file:///media/usb/repository/mirror/developer.download.nvidia.com/compute/
↪cuda/repos/ubuntu2204/x86_64/ /
```

4. Update the `apt` repository.

```
sudo apt update
```

The output from this command is similar to the following example.

```
Get:1 file:/media/usb/repository/mirror/security.ubuntu.com/ubuntu jammy-security
↪InRelease [107 kB]
Get:2 file:/media/usb/repository/mirror/archive.ubuntu.com/ubuntu jammy InRelease
↪[265 kB]
Get:3 file:/media/usb/repository/mirror/archive.ubuntu.com/ubuntu jammy-updates
↪InRelease [111 kB]
Get:4 file:/media/usb/repository/mirror/developer.download.nvidia.com/compute/
↪cuda/repos/ubuntu2204/x86_64  InRelease
```

(continues on next page)

(continued from previous page)

```
Get:5 file:/media/usb/repository/mirror/repo.download.nvidia.com/baseos/ubuntu/
↪jammy/x86_64 jammy InRelease [12.5 kB]
Get:6 file:/media/usb/repository/mirror/repo.download.nvidia.com/baseos/ubuntu/
↪jammy/x86_64 jammy-updates InRelease [12.4 kB]
Get:7 file:/media/usb/repository/mirror/developer.download.nvidia.com/compute/
↪cuda/repos/ubuntu2204/x86_64  Release [697 B]
Get:8 file:/media/usb/repository/mirror/developer.download.nvidia.com/compute/
↪cuda/repos/ubuntu2204/x86_64  Release.gpg [836 B]
Reading package lists... Done
```

5. Upgrade the system using the newly configured local repositories.

```
sudo apt full-upgrade
```

Chapter 17. Cloud-init Configuration File

This section provides instructions for creating a cloud-init configuration file for the Ubuntu Automated Server Installation.

17.1. Modifying the Configuration File

The following text is an outline of the example configuration file. It won't work as is but requires additional modifications as described in the sections. Refer to Ubuntu AUtomated Server Installation for more details.

1. Begin the configuration file with the following header:

```
#cloud-config
autoinstall:
  version: 1
```

2. Define a default user (the example uses Ubuntu), localization, and keyboard layout.

```
##
## Set initial system and user information
## use mkpassword -m sha-512 <password> to create a password
##
  identity:
    realname: DGX Ubuntu User
    hostname: dgx-host
    password: <PASSWORD HASH>
    username: ubuntu
  locale: en_US
  keyboard:
    layout: en
    variant: us
  reporting:
    builtin:
      type: print
```

3. The network section describes the network configuration and supports fixed addresses, DHCP, and various other network options. The names of the network interfaces are system-dependent. These are the primary management ports for various DGX systems. For example:

   - DGX-1: enp1s0f0

- DGX-2: enp6s0
- DGX A100: enp226s0
- DGX H100: ens6f0

```
##
## Network Configuration
##
  network:
    version: 2
    ethernets:
      enp1s0f0:
        dhcp4: yes
```

4. Update the Subiquity installer to the edge channel.

```
refresh-installer:
  channel: edge
  update: yes
```

5. Provide details about the additional NVIDIA repositories. Refer to *Drive Partitioning* below for more information.

```
##
## Enable this for using the remote repositories
##
  apt:
    <Repository details for the CUDA Compute and DGX Repository>

    conf: |
      Dpkg::Options {
        "--force-confdef";
        "--force-confold";
```

6. Configure storage.

   The next section describes the storage configuration, including swap configuration and drive partitioning. By setting the size to 0, we disable the SWAP partition. Refer to *Drive Partitioning*.

   The `reorder_uefi` flag tells the installer not to change the boot order to place the currently booted entry (BootCurrent) to the first option.

```
##
## Storage Configuration
##
  storage:
    config:
      <Partition and other configurations>
    swap:
      size: 0
    grub:
      reorder_uefi: false
```

7. Enable the SSH server.

   You can also set a default SSH key.

```
##
## SSH Server
```

(continues on next page)

(continued from previous page)

```
##
  ssh:
    install-server: yes
    allow-pw: yes
```

8. Provide a list of packages that should be installed.

   Refer to the comments in this text for instructions on changing the package names for specific DGX systems and on enabling or disabling features.

```
##
## Packages
##
  packages:

##
## NVIDIA DGX system configurations and system tools
## Replace dgx-a100 for other DGX systems:
##  dgx1     for DGX-1
##  dgx2     for DGX-2
##  dgx-a100 for DGX A100
##  dgx-h100 for DGX H100
##
    - dgx-a100-system-configurations
    - dgx-a100-system-tools
    - dgx-a100-system-tools-extra

## Remove this if you don’t want to use cachefilesd
    - nvidia-conf-cachefilesd

## Remove this if boot drive encryption is enabled and you don’t
## want the passphrase dialog only visible on the serial console
    - nvidia-ipmisol

##
## NVIDIA CUDA driver and tools
## Change the driver version to the branch you want to install
##
    - datacenter-gpu-manager
    - libnvidia-nscq-525
    - linux-modules-nvidia-525-server-generic
    - nvidia-driver-525-server
    - nvidia-modprobe
    - nv-persistence-mode

## Uncomment these to support the NVswitch on DGX-2, DGX A100, and DGX H100
## Ensure that the driver version matches with the versions above
#    - libnvidia-nscq-525
#    - nvidia-fabricmanager-525

##
## Mellanox drivers and tools
##
    - nvidia-mlnx-config
    - nvidia-mlnx-names
    - nvidia-mlnx-ofed-misc
```

(continues on next page)

(continued from previous page)

```
##
## NVIDIA container support
##
    - docker-ce
    - nv-docker-options
    - nvidia-docker2

##
## NVIDIA system management tools
##
    - nvsm
    - nvidia-motd
```

9. Add any additional software packages you want to install during autoinstall.
10. Finally, add a list of additional commands to be executed at the end of the installation.
    - Disable unattended upgrades
    - Disable the ondemand governor defaulting to performance mode
    - Enable DCGM and OpenIBD services
    - Enable `nv-peer-mem`

```
##
## Commands executed after completion of the installation
##
  late-commands:
    - curtin in-target --target=/target -- apt purge -y unattended-upgrades
    - curtin in-target --target=/target -- systemctl disable ondemand
    - curtin in-target --target=/target -- systemctl enable dcgm openibd
    - curtin in-target --target=/target -- update-rc.d nv_peer_mem defaults
# DGX A100 …
    - curtin in-target -- mlnx_pxe_setup.bash
```

17.2. Drive Partitioning

```
storage:
   config:
   - id: disk-sda
     type: disk
     ptable: gpt
     path: /dev/sda
     name: osdisk
     wipe: superblock-recursive
   - id: partition-sda1
     type: partition
     device: disk-sda
     number: 1
     size: 512M
     flag: boot
     grub_device: true
   - id: partition-sda2
```

(continues on next page)

(continued from previous page)

```
    type: partition
    device: disk-sda
    number: 2
    size: 100G
  - id: format-partition-sda1
    type: format
    fstype: fat32
    label: efi
    volume: partition-sda1
  - id: format-partition-sda2
    type: format
    fstype: ext4
    label: root
    volume: partition-sda2
  - id: root-mount
    type: mount
    path: /
    device: format-partition-sda2
    options: errors=remount-ro
    passno: 1
  - id: boot-mount
    type: mount
    path: /boot/efi
    device: format-partition-sda1
    passno: 1
  - id: disk-sdb
    type: disk
    ptable: gpt
    path: /dev/sdb
    name: raid
    wipe: superblock-recursive
  - id: partition-sdb1
    type: partition
    device: disk-sdb
    number: 1
  - id: format-partition-sdb1
    type: format
    fstype: ext4
    label: raid
    volume: partition-sdb1
  - id: raid-mount
    type: mount
    path: /raid
    device: format-partition-sdb1
    passno: 2
```

Chapter 18. Installing Docker Containers

This method applies to Docker containers hosted on the NVIDIA NGC Container Registry, and requires that you have an active NGC account.

1. On a system with internet access, log in to the NGC Container Registry by entering the following command and credentials.

Username:

Password:

1. Type $oauthtoken exactly as shown for the Username.
2. This is a special username that enables API key authentication. In place of apikey, paste in the API Key text that you obtained from the NGC website.
3. Enter the docker pull command, specifying the image registry, image repository, and tag:
4. Verify the image is on your system using docker images.
5. Save the Docker image as an archive.
6. Transfer the image to the air-gapped system using removable media such as a USB flash drive.
7. Load the NVIDIA Docker image.

```
docker load –i framework.tar
```

1. Verify the image is on your system.

```
docker images
```

Chapter 19. Third-Party License Notices

This NVIDIA product contains third party software that is being made available to you under their respective open source software licenses. Some of those licenses also require specific legal information to be included in the product. This section provides such information.

19.1. Micron msecli

The msecli utility is provided under the following terms:

Micron Technology, Inc. Software License Agreement PLEASE READ THIS LICENSE AGREEMENT ("AGREEMENT") FROM MICRON TECHNOLOGY, INC. ("MTI") CAREFULLY: BY INSTALLING, COPYING OR OTHERWISE USING THIS SOFTWARE AND ANY RELATED PRINTED MATERIALS ("SOFTWARE"), YOU ARE ACCEPTING AND AGREEING TO THE TERMS OF THIS AGREEMENT. IF YOU DO NOT AGREE WITH THE TERMS OF THIS AGREEMENT, DO NOT INSTALL THE SOFTWARE. LICENSE: MTI hereby grants to you the following rights: You may use and make one (1) backup copy the Software subject to the terms of this Agreement. You must maintain all copyright notices on all copies of the Software. You agree not to modify, adapt, decompile, reverse engineer, disassemble, or otherwise translate the Software. MTI may make changes to the Software at any time without notice to you. In addition MTI is under no obligation whatsoever to update, maintain, or provide new versions or other support for the Software. OWNERSHIP OF MATERIALS: You acknowledge and agree that the Software is proprietary property of MTI (and/or its licensors) and is protected by United States copyright law and international treaty provisions. Except as expressly provided herein, MTI does not grant any express or implied right to you under any patents, copyrights, trademarks, or trade secret information. You further acknowledge and agree that all right, title, and interest in and to the Software, including associated proprietary rights, are and shall remain with MTI (and/or its licensors). This Agreement does not convey to you an interest in or to the Software, but only a limited right to use and copy the Software in accordance with the terms of this Agreement. The Software is licensed to you and not sold.

DISCLAIMER OF WARRANTY:

THE SOFTWARE IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. MTI EXPRESSLY DISCLAIMS ALL WARRANTIES EXPRESS ORmIMPLIED, INCLUDING BUT NOT LIMITED TO, NONINFRINGEMENT OF THIRD PARTY RIGHTS, AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE. MTI DOES NOT WARRANT THAT THE SOFTWARE WILL MEET YOUR REQUIREMENTS, OR THAT THE OPERATION OF THE SOFTWARE WILL BE UNINTERRUPTED OR ERROR-FREE. FURTHERMORE, MTI DOES NOT MAKE ANY REPRESENTATIONS REGARDING THE USE OR THE RESULTS OF THE USE OF THE SOFTWARE IN TERMS OF ITS CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE. THE ENTIRE RISK ARISING OUT OF USE OR PERFORMANCE OF THE SOFTWARE REMAINS WITH YOU. IN NO EVENT SHALL MTI, ITS AFFILIATED COMPANIES OR THEIR SUPPLIERS BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, INCIDENTAL, OR SPECIAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF

INFORMATION) ARISING OUT OF YOUR USE OF OR INABILITY TO USE THE SOFTWARE, EVEN IF MTI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Because some jurisdictions prohibit the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply to you.

TERMINATION OF THIS LICENSE: MTI may terminate this license at any time if you are in breach of any of the terms of this Agreement. Upon termination, you will immediately destroy all copies the Software.

GENERAL: This Agreement constitutes the entire agreement between MTI and you regarding the subject matter hereof and supersedes all previous oral or written communications between the parties. This Agreement shall be governed by the laws of the State of Idaho without regard to its conflict of laws rules.

CONTACT: If you have any questions about the terms of this Agreement, please contact MTI's legal department at (208) 368-4500. By proceeding with the installation of the Software, you agree to the terms of this Agreement. You must agree to the terms in order to install and use the Software.

19.2. Mellanox (OFED)

MLNX_OFED <http://www.mellanox.com/> is provided under the following terms:

Copyright (c) 2006 Mellanox Technologies. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Chapter 20. Notices

20.1. Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation ("NVIDIA") makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice.

Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer ("Terms of Sale"). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death, or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer's own risk.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer's sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer's product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or

20.2. Trademarks

Copyright