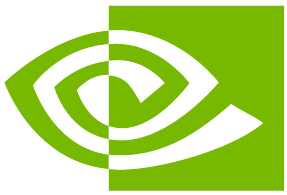


NVIDIA AI Enterprise

Release 5.2

NVIDIA Corporation

Oct 24, 2024

InnerLinkColorHTML000000

Overview

Chapter 1. What's Included

The NVIDIA compute software "stack" consists of various software products in the system software or infrastructure that are required to bootstrap a system with NVIDIA GPUs and be able to run accelerated AI or HPC workloads.

1.1. Infrastructure and Workload Management Components

All software for managing and optimizing the infrastructure and workloads is packaged into the NVIDIA AI Enterprise Infrastructure Branch. For more information, refer to the NVIDIA AI Enterprise Release Branches.

Table 1: Infrastructure and Workload Management Components

Component	Description	NGC Link	Documentation
NVIDIA vGPU (C-Series) Host Driver	NVIDIA driver to be deployed in the hypervisor for virtualized environments.	NVIDIA vGPU (C-Series) Host Driver on NGC	NVIDIA vGPU Host Driver Documentation
NVIDIA vGPU (C-Series) Guest Driver	NVIDIA virtual GPU software driver to be deployed in the VM or on a bare metal Operating System to enable multiple virtual machines (VMs) to have simultaneous, direct access to a single physical GPU.	NVIDIA vGPU (C-Series) Guest Driver on NGC	NVIDIA vGPU Guest Driver Documentation
NVIDIA Data Center Driver	Enable GPU acceleration for AI and Deep Learning in data centers.	GPU Driver on NGC	NVIDIA Data Center Driver Documentation
GPU Operator	NVIDIA GPU Operator simplifies the deployment of NVIDIA AI Enterprise by automating the management of all NVIDIA software components needed to provision GPUs in Kubernetes.	GPU Operator on NGC	NVIDIA GPU Operator Documentation
Network Operator	NVIDIA Network Operator simplifies the provisioning and management of NVIDIA networking resources in a Kubernetes cluster.	Network Operator on NGC	NVIDIA Network Operator Documentation
NVIDIA NIM Operator	NVIDIA NIM Operator enables cluster administrators to operate the software components and services required to run LLM, embedding, and other models using NVIDIA NIM microservices in Kubernetes.	NIM Operator on NGC	NVIDIA NIM Operator Documentation
Base Command Manager Essentials	NVIDIA Base Command Manager essentials streamlines cluster provisioning, workload management, and infrastructure monitoring across data centers and edge locations. It provides all the tools you need to deploy and manage an AI data center. NVIDIA Base Command Manager Essentials comprises the features of NVIDIA Base Command Manager that are certified for use with NVIDIA AI Enterprise.	Base Command Manager Essentials on NGC	NVIDIA Base Command Manager Documentation
NVIDIA License System	The NVIDIA License System is used to serve a pool of floating licenses to NVIDIA-licensed products. The NVIDIA License System is configured with licenses obtained from the NVIDIA Licensing Portal.	N/A	NVIDIA License System Documentation

Chapter 2. Release Notes

Refer to the Previous Releases section to review NVIDIA AI Enterprise documentation v5.1 - v1.0.

To review AI Enterprise documentation v5.2 and more recent, choose a version from the bottom left navigation selector toggle.

2.1. 5.2 Release

These are the NVIDIA AI Enterprise 5.2 Release Notes. Refer to the documentation of the products that are part of NVIDIA AI Enterprise to learn about limitations and issues.

Announcements

- The NVIDIA NIM Operator is now part of NVIDIA AI Enterprise. It automates the deployment and management of NVIDIA NIM microservices for production AI pipelines and enhances inference performance with capabilities such as intelligent model pre-caching for lower initial inference latency and faster autoscaling.

Compatibility

For more information, refer to the *NVIDIA AI Enterprise Product Support Matrix*.

Chapter 3. Infrastructure Support Matrix

3.1. Supported NVIDIA Infrastructure Software

Table 1: Supported Infrastructure Software

Product	Version	x86	ARM
NVIDIA GPU Data Center Driver	550.127.05	Supported	Supported
NVIDIA vGPU Drivers (Host and Guest)	17.4	Supported	Not Supported
NVIDIA Container Toolkit	v1.16.2	Supported	Supported
NVIDIA GPU Operator	v24.6.2	Supported	Supported
NVIDIA Network Operator	v24.7.0	Supported	Supported
NVIDIA Base Command Manager Essentials (BCME)	10.24.09	Supported	Supported
NVIDIA NIM Operator	1.0.0	Supported	Not Supported

3.2. Supported NVIDIA GPUs and Networking

NVIDIA AI Enterprise is supported on the following NVIDIA GPUs with compatible third-party servers listed on the NVIDIA-certified systems page.

Specific NVIDIA AI Enterprise supported products may not support all OS or GPU; refer to the individual product release notes for any discrepancies.

NVIDIA Ada Lovelace

- NVIDIA L40S
- NVIDIA L40
- NVIDIA L20
- NVIDIA L4
- NVIDIA L2
- NVIDIA RTX 6000 Ada
- NVIDIA RTX 5880 Ada
- NVIDIA RTX 5000 Ada

NVIDIA Ampere

- NVIDIA A800
- NVIDIA A100X
- NVIDIA A100
- NVIDIA A40
- NVIDIA A30X
- NVIDIA A30
- NVIDIA A16
- NVIDIA A10
- NVIDIA A10G
- NVIDIA A2
- NVIDIA RTX A6000
- NVIDIA RTX A5500
- NVIDIA RTX A5000

NVIDIA Grace Hopper

- NVIDIA GH200

NVIDIA Hopper

- NVIDIA H800
- NVIDIA H200[1]
- NVIDIA H100
- NVIDIA H20

[1] NVIDIA H200 is not supported with NVIDIA vGPU 17.

NVIDIA Turing

- NVIDIA Quadro RTX 8000 Passive
- NVIDIA Quadro RTX 6000 Passive
- NVIDIA T4

NVIDIA Volta

- NVIDIA V100

Multi-node requires an Ethernet NIC that supports RoCE. For best performance, NVIDIA recommends using an NVIDIA Mellanox ConnectX and an NVIDIA GPU.

Table 2: Supported Ethernet NICs and SuperNICs

Product Family	Architecture
NVIDIA ConnectX-6 NIC	NVIDIA ConnectX-6
NVIDIA ConnectX-6 Dx NIC	NVIDIA ConnectX-6 Dx
NVIDIA ConnectX-7 NIC	NVIDIA ConnectX-7
NVIDIA BlueField-3 SuperNIC	NVIDIA BlueField-3

3.3. Supported Platforms

NVIDIA AI Enterprise is supported on NVIDIA DGX servers in bare-metal deployments with the NVIDIA data center driver for Linux, which is included in the DGX OS software.

Table 3: Supported Platforms

Accelerated Platform	Architecture
NVIDIA DGX A100	NVIDIA Ampere
NVIDIA DGX H200	NVIDIA Hopper
NVIDIA DGX H100	NVIDIA Hopper
NVIDIA HGX A100	NVIDIA Ampere
NVIDIA HGX H200	NVIDIA Hopper
NVIDIA HGX H100	NVIDIA Hopper
NVIDIA IGX Orin[2]	NVIDIA Ada Lovelace

[2] With optional RTX 6000 Ada Lovelace GPU.

Note

DGX platforms, HGX H200, and IGX Orin is not supported with NVIDIA vGPU.

3.4. Bare Metal

Refer to the following platform support matrix for NVIDIA AI Enterprise if you have a dedicated physical server on-premises.

Table 4: Containers Orchestrated with Kubernetes on Bare Metal

Orchestration Platform			Operating System		NVIDIA AI Enterprise Infrastructure Support		
Name	**Versions**	**Engine**	**Name**	**Versions**	**GPU Operator**	**Network Operator**	**GPU Driver Support**[?]
Charmed Kubernetes	▶ 1.28 ▶ 1.29 ▶ 1.30	Containerd	Ubuntu	▶ 20.04 LTS ▶ 22.04 LTS	Supported	Supported	vGPU Guest/Datacenter
HPE Ezmeral Runtime Enterprise	5.6	Containerd	RedHat Enterprise Linux	▶ 8.6 ▶ 8.8 ▶ 8.10	Supported	Not Supported	vGPU Guest/Datacenter
RedHat OpenShift[4]	▶ 4.13 ▶ 4.14 ▶ 4.15 ▶ 4.16	CRI-O	RedHat CoreOS	▶ 4.13 ▶ 4.14 ▶ 4.15 ▶ 4.16	Supported	Supported	vGPU Guest/Datacenter
Nutanix NKP	1.27	Containerd	Ubuntu	▶ 20.04 LTS ▶ 22.04 LTS	Supported	Not Supported	vGPU Guest/Datacenter
Upstream Kubernetes	▶ 1.24 ▶ 1.25 ▶ 1.26 ▶ 1.27 ▶ 1.28 ▶ 1.29 ▶ 1.30	Containerd	RedHat Enterprise Linux	▶ 8.6 ▶ 8.8 ▶ 8.10	Supported	Supported[5]	vGPU Guest/Datacenter
			Ubuntu	▶ 20.04 LTS ▶ 22.04 LTS			
		CRI-O	RedHat Enterprise Linux	▶ 8.6 ▶ 8.8 ▶ 8.10	Not Supported	Not Supported	
			Ubuntu	▶ 20.04 LTS ▶ 22.04			

Table 5: Standalone Containers on Bare Metal

Container			Operating System		NVIDIA AI Enterprise Infrastructure Support		
Name	Versions	Engine	Name	Versions	GPU Operator	Network Operator	GPU Driver Support
Non-Kubernetes (standalone containers)		Docker/Pod	RedHat Enterprise Linux	▶ 7.9 ▶ 8.6 ▶ 8.8 ▶ 8.10 ▶ 9.2 ▶ 9.4	N/A	N/A	vGPU Guest/Datacenter
			SUSE Linux Enterprise Server	15 SP2 and later			
			Ubuntu	▶ 20.04 LTS ▶ 22.04 LTS			

3.5. Virtualized

Refer to the following platform support matrix for NVIDIA AI Enterprise if you have a physical server separated into multiple virtual servers on-premises.

[3] The NVIDIA vGPU Guest Driver is optional in bare metal deployments. Use of the NVIDIA vGPU Guest Driver allows licensing enforcement.

[4] RedHat OpenShift Virtualization is supported with RedHat OpenShift 4.15 and 4.16 and only with GPU Passthrough and NVIDIA vGPU (exclusive of MIG-backed NVIDIA vGPUs).

[5] Network Operator is supported with RHEL 8.6, 8.8, and 8.10.

Table 6: Containers Orchestrated with Kubernetes on Virtualized Environments

Orchestration Platform			Guest Operating System		Hypervisor		NVIDIA AI Enterprise Infrastructure Support			
Name	Versions	Engine	Name	Versions	Name	Versions	GPU Operator	Network Operator	GPU Driver Support (inside the virtual machine)	
									vGPU	Passthrough[7]
Charmed Kubernetes	▶ 1.28 ▶ 1.29 ▶ 1.30	Containerd	Ubuntu	▶ 20 LTS ▶ 22 LTS	Ubuntu[7]	▶ 20.04 LTS ▶ 22.04 LTS	Supported	Supported	vGPU Guest	vGPU Guest/Data center
					VMware vSphere	▶ 7.0 update 2 and later ▶ 8.0 and later				
Red Hat OpenShift[8]	▶ 4.13 ▶ 4.14 ▶ 4.15 ▶ 4.16	CRI-O	Red Hat CoreOS	▶ 4.1 ▶ 4.1 ▶ 4.1 ▶ 4.1	Red Hat Enterprise Linux[7]	▶ 7.9 ▶ 8.6 ▶ 8.8 ▶ 8.10 ▶ 9.2 ▶ 9.4	Supported	Supported	vGPU Guest	vGPU Guest/Data center
					VMware vSphere	▶ 7.0 update 2 and later ▶ 8.0 and later				

[6] NVIDIA vGPU provides management (GPU Partitioning, live migration, etc.) and monitoring capabilities that aren't available with GPU Passthrough and the Datacenter GPU Driver.

[7] NVIDIA vGPU software is supported on Linux with KVM platforms **only** by specific hypervisor software vendors. For information about which NVIDIA vGPU software releases and hypervisor software releases are supported, consult the documentation from your hypervisor vendor. For information about Linux distributions supported as a guest OS, consult the documentation from your hypervisor vendor. A guest OS release must be supported not only by NVIDIA vGPU software but also by your virtualization software vendor.

[8] NVIDIA AI Enterprise supports every patch release for the listed RedHat OpenShift release provided that RedHat also supports it. When a release or patch release is no longer supported by RedHat, it is no longer supported by NVIDIA AI Enterprise.

Table 7: Standalone Containers on Virtualized Environments

Container			Guest Operating System		Hypervisor		NVIDIA AI Enterprise Infrastructure Support			
Name	Versions	Engine	Name	Versions	Name	Versions	GPU Operator	Network Operator	GPU Driver Support (inside the virtual machine)	
									vGPU	Passthrough[?]
Non-Kubernetes (standalone containers)		Docker/Podman	Red Hat Enterprise Linux	▶ 7.9 ▶ 8.6 ▶ 8.8 ▶ 8.1 ▶ 9.2 ▶ 9.4	Red Hat Enterprise Linux[?]	▶ 7.9 ▶ 8.6 ▶ 8.8 ▶ 8.9 ▶ 9.2 ▶ 9.4	N/A	N/A	vGPU Guest	vGPU Guest/Data center
					VMware vSphere	▶ 7.0 update 2 and later ▶ 8.0 and later				
			SUSE Linux Enterprise Server	15 SP2 and later	VMWare vSphere	▶ 7.0 update 2 and later ▶ 8.0 and later	N/A	N/A	vGPU Guest	vGPU Guest/Data center
			Ubuntu	▶ 20 LTS ▶ 22 LTS	Ubuntu[?]	▶ 20.04 LTS ▶ 22.04 LTS	N/A	N/A	vGPU Guest	vGPU Guest/Data center
					VMware					

[9] Obtain the NVIDIA Virtual GPU Manager software directly from Nutanix through the My Nutanix portal (My Nutanix account required). If the NVIDIA AI Enterprise release is not available from the My Nutanix portal, contact Nutanix.

Table 8: Non-containerized Applications on Hypervisors and Guest Operating Systems Supported with vGPU

Hypervisor		Guest Operating System		NVIDIA AI Enterprise Infrastructure Support
Name	**Version**	**Name**	**Version**	**vGPU**
VMware ESX	ESXi 8.0	Debian	12	vGPU Guest
		RedHat Enterprise Linux	▶ 8.8 ▶ 8.10 ▶ 9.2 ▶ 9.4	vGPU Guest
		Ubuntu	▶ 20.04 LTS ▶ 22.04 LTS ▶ 24.04 LTS	vGPU Guest
		SUSE Linux Enterprise Server	▶ 12 SP3+ ▶ 12 SP5 ▶ 15 SP2	vGPU Guest
	ESXi 7.0 Update 2	RedHat Enterprise Linux	▶ 8.8 ▶ 8.10 ▶ 9.2 ▶ 9.4	vGPU Guest
		Ubuntu	▶ 20.04 LTS ▶ 22.04 LTS ▶ 24.04 LTS	vGPU Guest
		SUSE Linux Enterprise Server	▶ 12 SP3+ ▶ 12 SP5 ▶ 15 SP2	vGPU Guest
Ubuntu[7]	▶ 20.04 LTS ▶ 22.04 LTS ▶ 24.04 LTS	Ubuntu	▶ 20.04 LTS ▶ 22.04 LTS ▶ 24.04 LTS	vGPU Guest
RedHat Enterprise Linux (RHEL) with KVM[7]	RHEL 8.8	RedHat Enterprise Linux	▶ 8.8 ▶ 8.10	vGPU Guest
	RHEL 8.10	RedHat Enterprise Linux	▶ 8.8 ▶ 8.10	vGPU Guest
	RHEL 9.2	RedHat Enterprise Linux	▶ 8.8 ▶ 8.10 ▶ 9.2	vGPU Guest

Note

If the NVIDIA AI Enterprise release that you need is not available from the My Nutanix portal, contact Nutanix. Then, follow the instructions on the My Nutanix portal to obtain the correct NVIDIA AI Enterprise graphics drivers from the NVIDIA Licensing Portal.

3.6. Base Command Manager Essentials

Table 9: Base Command Manager

Orchestration Platform			Operating System		NVIDIA AI Enterprise Infrastructure Support		
Name	Versions	Engine	Name	Versions	GPU Operator	Network Operator	GPU Driver Support[?]
Upstream Kubernetes	▶ 1.27 ▶ 1.28 ▶ 1.29 ▶ 1.30	Containerd	Ubuntu	▶ 20.04 LTS ▶ 22.04 LTS ▶ 24.04 LTS	Supported	Supported	Data center
			RedHat Enterprise Linux	▶ 8 ▶ 9	Supported	Supported	Data center
Slurm (non-Kubernetes)	▶ 23.02 ▶ 23.11 ▶ 24.05	N/A	Ubuntu	▶ 20.04 LTS ▶ 22.04 LTS ▶ 24.04 LTS	N/A	N/A	Data center
			RedHat Enterprise Linux	▶ 8 ▶ 9	N/A	N/A	Data center
PBS Pro	▶ 2024.1.1 ▶ 2022.1.6	N/A	Ubuntu	▶ 20.04 LTS ▶ 22.04 LTS ▶ 24.04 LTS	N/A	N/A	Data center
			RedHat Enterprise Linux	▶ 8 ▶ 9	N/A	N/A	Data center

3.7. Public Cloud

3.7.1. Managed Kubernetes

Refer to the following platform support matrix for NVIDIA AI Enterprise if you have a virtual server that runs in a cloud computing environment and is accessible remotely.

Table 10: Managed Kubernetes

Cloud Service Provider	Orchestration Platform			Operating System		NVIDIA AI Enterprise Infrastructure Support		
	Name	K8s Versions	Engine	Name	Versions	GPU Operator	Network Operator	GPU Driver Support
AWS	Amazon Elastic Kubernetes Service (EKS)	▶ 1.24 ▶ 1.25 ▶ 1.26 ▶ 1.27 ▶ 1.28 ▶ 1.29 ▶ 1.30	Containerd	Ubuntu	▶ 20.04 LTS ▶ 22.04 LTS	Supported	Not Supported	vGPU Guest/Datacenter
Google	Google Kubernetes Engine (GKE)	▶ 1.24 ▶ 1.25 ▶ 1.26 ▶ 1.27 ▶ 1.28 ▶ 1.29 ▶ 1.30	Containerd	Ubuntu	22.04 LTS	Supported	Not Supported	vGPU Guest/Datacenter
Microsoft	Azure Kubernetes Service (AKS)	▶ 1.24 ▶ 1.25 ▶ 1.26 ▶ 1.27 ▶ 1.28 ▶ 1.29 ▶ 1.30	Containerd	Ubuntu	22.04 LTS	Supported	Not Supported	vGPU Guest/Datacenter

N/A	RedHat OpenShift	▶ 4.13	CRI-O	RedHat CoreOS	▶ 4.13	Supported	Not Supported	vGPU Guest/Datacenter

3.7.2. Standard GPU Instances

Table 11: Standard Instances for Kubernetes and Standalone Containers

Cloud Service Provider	Virtual Machine (VM) Instance with GPU	Product Family
Alibaba	gn7e	NVIDIA A10
	gn7i	NVIDIA A10
	gn7s	NVIDIA A30
	gn6i	NVIDIA T4
	gn6e	NVIDIA V100
	gn6v	NVIDIA V100
Amazon Web Services (AWS)	EC2 P3	NVIDIA V100
	EC2 P4	NVIDIA A100
	EC2 P5	NVIDIA H100
	EC2 G4	NVIDIA T4
	EC2 G5	NVIDIA A10G
	EC2 G6	NVIDIA L4
	EC2 G6e	NVIDIA L40S
Azure	NCads_H100_v5-series	NVIDIA H100
	NCCads_H100_v5-series	NVIDIA H100
	NCv3-series	NVIDIA V100
	NCasT4_v3-series	NVIDIA T4
	NC_A100_v4-series	NVIDIA A100
Google Cloud Platform (GCP)	A3 VM	NVIDIA H100
	A2 VM	NVIDIA A100
	G2 VM	NVIDIA L4
	N1 VM	▶ NVIDIA T4 ▶ NVIDIA V100
Oracle Cloud Infrastructure (OCI)	BM.GPU3	NVIDIA V100
	▶ BM.GPU4 ▶ BM.GPU.A100	NVIDIA A100
	BM.GPU.A10	NVIDIA A10

continues on next page

Table 11 – continued from previous page

Cloud Service Provider	Virtual Machine (VM) Instance with GPU	Product Family
Tencent Cloud	BM.GPU.H100.8	NVIDIA H100
	VM.GPU3	NVIDIA V100
	VM.GPU.A10	NVIDIA A10
	PNV4	NVIDIA A10
	GT4	NVIDIA A100
	▶ GN10Xp ▶ GN10X	NVIDIA V100
	▶ GN7 ▶ GN7vi ▶ GI3X	NVIDIA T4
Volcano Engine	ecs.gni2	NVIDIA A10

3.7.3. NVIDIA GPU Optimized VMI on CSP Marketplace

For ease of use in the cloud, NVIDIA provides compute-optimized and validated base Virtual Machine Instances (VMI) to run standalone NVIDIA AI containers through CSP marketplaces. Each VMI includes key technologies and software from NVIDIA for rapid deployment, management, and scaling of AI workloads in the modern hybrid cloud.

Table 12: NVIDIA GPU Optimized VMI on CSP Marketplace

Cloud Service Provider	VMI Name	GPUs	K8s Support	Standalone Container
AWS	NVIDIA AI Enterprise	Listed in *Standard GPU Instances*	Not Supported	Supported
Azure	NVIDIA AI Enterprise	Listed in *Standard GPU Instances*	Not Supported	Supported
GCP	NVIDIA AI Enterprise	Listed in *Standard GPU Instances*	Not Supported	Supported

3.8. CPU-Only Server Support

NVIDIA AI Enterprise will support the following CPU-enabled frameworks:

- TensorFlow
- PyTorch
- Triton Inference Server with FIL backend
- NVIDIA RAPIDS with XGBoost and Dask

The CPU-enabled frameworks are supported on CPU-only servers that are part of the NVIDIA Certified Systems list.

Chapter 4. Quick Start Guide

This Quick Start Guide will help you rapidly deploy and configure NVIDIA AI Enterprise, enabling AI workloads on bare-metal, public cloud, or virtualized environments. Follow these steps to seamlessly set up and optimize your infrastructure for accelerated AI and deep learning tasks.

If you need complete instructions for installing and configuring NVIDIA AI Enterprise, are using NVIDIA AI Enterprise in an NVIDIA vGPU deployment, or are using multiple nodes, refer to the *NVIDIA AI Enterprise Deployment Guide*.

Note

These instructions do not apply to NVIDIA DGX systems. For information about how to use these systems, refer to NVIDIA DGX Systems.

Attention

If you have previously purchased NVIDIA AI Enterprise, have an NVIDIA Enterprise Account, and an NGC account, you can directly go to *Installing NVIDIA AI Enterprise Software Components*.

4.1. Activating the Accounts for Getting NVIDIA AI Enterprise

After your order for NVIDIA AI Enterprise is processed, you will receive an order confirmation message. This message contains information that you need for getting NVIDIA AI Enterprise and technical support from NVIDIA. To get NVIDIA AI Enterprise and technical support from NVIDIA, you must have an NVIDIA Enterprise Account, which provides login access to the following NVIDIA software components:

- **NVIDIA NGC** - provides access to all enterprise software, services, and management tools included in NVIDIA AI Enterprise
- **NVIDIA Enterprise Support Portal** - provides access to to support services for NVIDIA AI Enterprise
- **NVIDIA Licensing Portal** - provides access to your entitlements and options for managing your NVIDIA AI Enterprise license servers. Note that installing and managing NVIDIA AI Enterprise license servers are a prerequisite for deploying vGPU software.

These components can be accessed from the NVIDIA Application Hub. To activate your account and access NVIDIA AI Enterprise, follow these steps:

- *Creating your NVIDIA Enterprise Account*
- *Linking an Evaluation Account to an NVIDIA Enterprise Account for Purchased Licenses*

4.1.1. Before You Begin

Before you attempt the procedures in this guide, ensure that the following prerequisites are met:

- You have a third-party NVIDIA-certified server platform that supports NVIDIA AI Enterprise.
- One or more NVIDIA GPUs that support NVIDIA AI Enterprise are installed in your server platform.
- You have a valid NVIDIA software subscription.
- If you are using a GPU that is supplied with NVIDIA AI Enterprise software, such as the NVIDIA H100 PCIe GPU, your NVIDIA AI Enterprise license for H100 has been activated.

For information about supported hardware and software, and any known issues for this release of NVIDIA AI Enterprise, refer to the *NVIDIA AI Enterprise Release Notes*.

4.1.2. Your Order Confirmation Message

After your order for NVIDIA AI Enterprise is processed, you will receive an order confirmation message to which your NVIDIA Entitlement Certificate is attached. Your NVIDIA Entitlement Certificate contains your product activation keys and provides instructions for using the certificate.

If you are a data center administrator, follow the instructions in the NVIDIA Entitlement Certificate to use the certificate. Otherwise, forward your order confirmation message, including the attached NVIDIA Entitlement Certificate, to a data center administrator in your organization.

4.1.3. NVIDIA Enterprise Account Requirements

You must have a suitable NVIDIA Enterprise Account to get access to NVIDIA AI Enterprise and technical support from NVIDIA.

Whether or not you have a suitable NVIDIA Enterprise Account depends on whether you have previously purchased NVIDIA AI Enterprise.

- If you have previously purchased NVIDIA AI Enterprise, you already have a suitable NVIDIA Enterprise Account.

To use this account to get NVIDIA AI Enterprise, download the software assets that you require from the NVIDIA AI Enterprise Infra Release 5 collection on NVIDIA NGC. For details, refer to *Accessing the NVIDIA AI Enterprise Software Suite*.

- If you have obtained an evaluation license but have not previously purchased NVIDIA AI Enterprise, you do **not** have a suitable NVIDIA Enterprise Account. To create a suitable NVIDIA Enterprise Account, follow the **Register** link in the instructions for using the certificate to create an account for your **purchased** licenses. You can choose to create a separate account for your purchased licenses or link your existing account for an evaluation license to the account for your purchased licenses.

- To create a separate account for your purchased licenses, follow the instructions in *Creating your NVIDIA Enterprise Account*, specifying a different email address than the address with which you created your existing account.
- To link your existing account for an evaluation license to the account for your purchased licenses, follow the instructions in *Linking an Evaluation Account to an NVIDIA Enterprise Account for Purchased Licenses*, specifying the email address with which you created your existing account.

- If you have not previously purchased NVIDIA AI Enterprise, you do **not** have a suitable NVIDIA Enterprise Account.

To create a suitable NVIDIA Enterprise Account, follow the **Register** link in the instructions for using the certificate to create your account. For details, refer to *Creating your NVIDIA Enterprise Account*.

4.1.3.1 Creating your NVIDIA Enterprise Account

If you do not have an NVIDIA Enterprise Account, you must create an account to get access to the NVIDIA AI Enterprise software components and technical support from NVIDIA. For details on these software components, refer to *Activating the Accounts for Getting NVIDIA AI Enterprise*.

If you have an account that was created for an evaluation license and you want to access licenses that you purchased, you must repeat the registration process when you receive your purchased licenses. You can choose to create a separate account for your purchased licenses or link your existing account for an evaluation license to the account for your purchased licenses.

- To create a separate account for your purchased licenses, perform this task, specifying a different email address than the address with which you created your existing account.
- To link your existing account for an evaluation license to the account for your purchased licenses, follow the instructions in *Linking an Evaluation Account to an NVIDIA Enterprise Account for Purchased Licenses*, specifying the email address with which you created your existing account.

Before you begin, ensure that you have your order confirmation message.

1. In the instructions for using your NVIDIA Entitlement Certificate, follow the **Register** link.
2. Fill out the form on the NVIDIA Enterprise Account Registration page and click **REGISTER**. A message confirming that an account has been created appears. An email instructing you to log in to your account on the NVIDIA Application Hub is sent to the email address you provided.
3. Open the email instructing you to log in to your account and click **Log In**.
4. On the NVIDIA Application Hub Login page that opens, in the text-entry field, type the email address you provided and click **Sign In**.
5. On the Create Your Account page that opens, provide and confirm a password for the account and click **Create Account**. A message prompting you to verify your email address appears. An email instructing you to verify your email address is sent to the email address you provided.
6. Open the email instructing you to verify your email address and click **Verify Email Address**. A message confirming that your email address is confirmed appears.

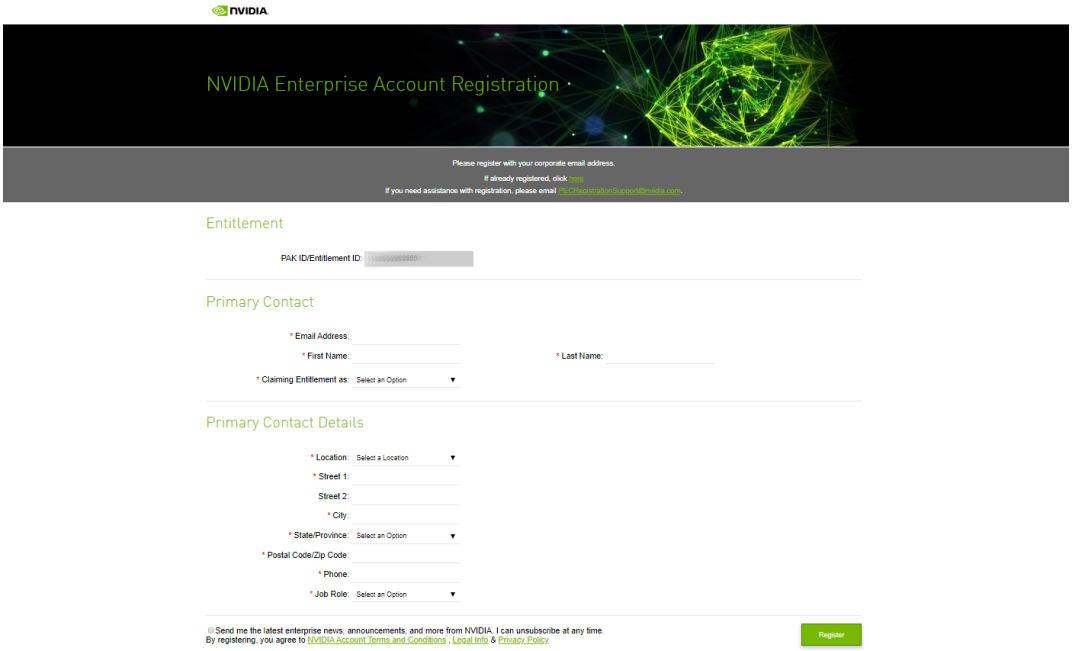
From the NVIDIA Application Hub page, you can now log in to the components that are listed in *Activating the Accounts for Getting NVIDIA AI Enterprise*.

4.1.3.2 Linking an Evaluation Account to an NVIDIA Enterprise Account for Purchased Licenses

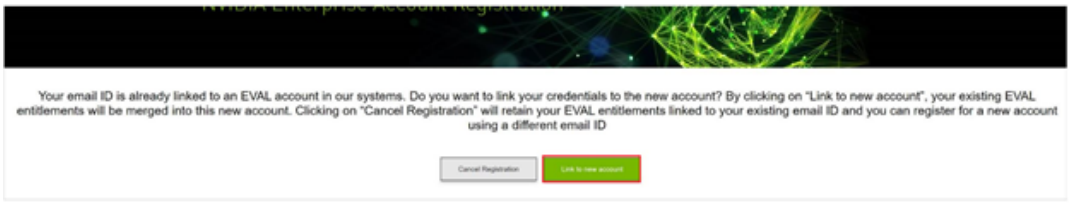
If you have an account that was created for an evaluation license, you must repeat the registration process when you receive your purchased licenses. To link your existing account for an evaluation license to the account for your purchased licenses, register for an NVIDIA Enterprise Account with the email address with which you created your existing account.

If you want to create a separate account for your purchased licenses, follow the instructions in *Creating your NVIDIA Enterprise Account*, specifying a different email address than the address with which you created your existing account.

1. In the instructions for using the NVIDIA Entitlement Certificate **for your purchased licenses**, follow the **Register** link.
2. Fill out the form on the NVIDIA Enterprise Account Registration page, specifying the email address with which you created your existing account, and click **Register**.



3. When a message stating that your email address is already linked to an evaluation account is displayed, click **LINK TO NEW ACCOUNT**.



4. Log in to the NVIDIA Licensing Portal with the credentials for your existing account.

4.2. Installing NVIDIA AI Enterprise Software Components

4.2.1. The NVIDIA NGC Catalog

NVIDIA AI Enterprise components are distributed through the NVIDIA NGC Catalog. Infrastructure and workload management components are distributed as resources in the NVIDIA AI Enterprise Infra Release 5 collection. Tools for AI development and use cases are available from the NVIDIA AI Enterprise Software Suite.

4.2.1.1 Accessing the NVIDIA AI Enterprise Infrastructure Resources

Infrastructure and workload management components of NVIDIA AI Enterprise are distributed as resources in the NVIDIA AI Enterprise Infra Release 5 collection.

The **NVIDIA AI Enterprise Infra Release 5** collection offers infrastructure management and orchestration software bundled together to efficiently manage and scale AI workloads, and contains the following resources:

- GPU Operator
- Network Operator
- NVIDIA NIM Operator
- vGPU Host Driver
- Containerized vGPU Guest Driver
- Containerized NVIDIA GPU Datacenter Driver
- NVIDIA Base Command Manager Essentials

Before downloading any NVIDIA AI Enterprise software assets, ensure that you have signed in to NVIDIA NGC from the NVIDIA NGC Sign In page.

1. Go to the NVIDIA AI Enterprise Infra Release 5 collection on NVIDIA NGC.
2. Click the **Entities** tab and select the resource that you are interested in.
3. Click **Download** and, from the menu that opens, choose to download the resource by using a direct download in the browser, the displayed wget command, or the CLI.

4.2.1.2 Accessing the NVIDIA AI Enterprise Software Suite

Tools for AI development and use cases are available from the NVIDIA AI Enterprise Software Suite and are distributed through the NVIDIA NGC Catalog.

Before downloading any NVIDIA AI Enterprise software assets, ensure that you have signed in to NVIDIA NGC from the NVIDIA NGC Sign In page.

1. View the NVIDIA AI Enterprise Software Suite on NVIDIA NGC.
   - Go to the NVIDIA AI Enterprise Supported page on NVIDIA NGC.
   - Visit the NVIDIA NGC site and set the **NVIDIA AI Enterprise Support** filter.

2. Browse the NVIDIA AI Enterprise Software Suite to find software assets that you are interested in.
3. For each software asset that you are interested in, click the asset to learn more about or download the asset.

4.2.2. NVIDIA AI Enterprise Deployment Options

The following table outlines the most common approaches used to deploy NVIDIA AI Enterprise.

Table 1: NVIDIA AI Enterprise Deployment Options

Category	Getting Started	Deployment Guide	Reference Tutorials
Deploy on bare metal	▶ *Installing NVIDIA AI Enterprise on Bare Metal Ubuntu* ▶ *Obtaining NVIDIA Base Command Manager Essentials*	NVIDIA AI Enterprise Bare Metal Deployment Guide	▶ *Running ResNet with TensorRT* ▶ *Running ResNet-50 with TensorFlow* ▶ *Running NVIDIA NIM on Bare Metal Ubuntu 22.04*
Deploy on the public cloud	*Installing NVIDIA AI Enterprise on Microsoft Azure*	NVIDIA AI Enterprise Cloud Deployment Guide	*Running an LLM NIM on Microsoft Azure using NVIDIA AI Enterprise*
Deploy in virtualized environments using NVIDIA vGPU (C-Series)	*Installing NVIDIA AI Enterprise on VMware vSphere*	NVIDIA AI Enterprise VMware Deployment Guide	

4.3. Installing NVIDIA AI Enterprise on Bare Metal Ubuntu 22.04

This section of the NVIDIA AI Enterprise Quick Start Guide provides minimal instructions for a bare-metal, single-node deployment of NVIDIA AI Enterprise using Docker on a third-party NVIDIA-certified system.

4.3.1. NVIDIA AI Enterprise Software Prerequisites

To enable NVIDIA GPU acceleration for compute and AI workloads running in data centers:

1. Download the NVIDIA GPU data center drivers from this location.
2. Select **Linux 64 bits, Ubuntu 22.04** as the Operating System to download the `.run` file.

4.3.2. Installing NVIDIA AI Enterprise using the TRD Driver on Ubuntu 22.04 from a `.run` File

Installation of the NVIDIA AI Enterprise software driver for Linux requires:

- Compiler toolchain
- Kernel headers

Prerequisites

Ensure you follow the pre-installation steps.

> **Note**
>
> If you prefer to use a Debian package, refer to the Debian instructions.

Steps

1. Log into the system and check for updates.

```
sudo apt-get update
```

2. Install the GCC compiler and the make tool in the terminal.

```
sudo apt-get install build-essential
```

3. Copy the NVIDIA AI Enterprise Linux driver package, for example, `NVIDIA-Linux-x86_64-550.90.12.run`, to the host machine where you are installing the driver.

   Where `x_x:xxx.xx.xx` is the current NVIDIA AI Enterprise version and driver version.

4. Navigate to the directory containing the NVIDIA Driver `.run` file. Then, add the Executable permission to the NVIDIA Driver file using the `chmod` command.

```
sudo chmod +x NVIDIA-Linux-x86_64-xxx.xx.xx-grid.run
```

5. From a console shell, run the driver installer as the root user, and accept the defaults.

```
sudo sh ./NVIDIA-Linux-x86_64-xxx.xx.xx-grid.run
```

6. Reboot the system.

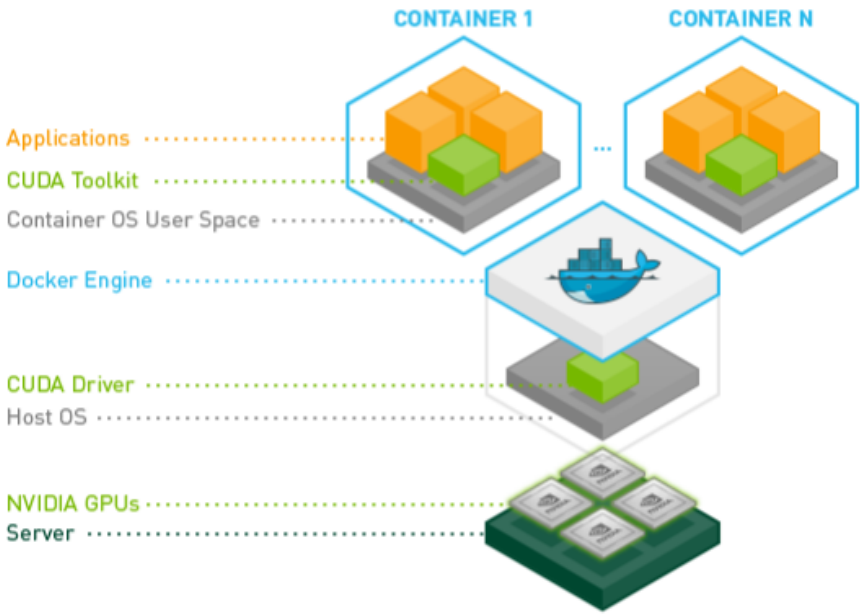
```
sudo reboot
```

7. After the system has rebooted, confirm that you can see your NVIDIA vGPU device in the output from `nvidia-smi`.

```
nvidia-smi
```

4.3.3. Installing the NVIDIA Container Toolkit

The NVIDIA Container Toolkit allows users to build and run GPU accelerated Docker containers. The toolkit includes a container runtime library and utilities to configure containers to leverage NVIDIA GPUs automatically. Complete documentation and frequently asked questions are available on the repository wiki.



1. **Install Docker** - Refer to Install Docker Engine on Ubuntu | Docker Documentation for the installation procedure for Ubuntu.
2. **Install the NVIDIA Container Toolkit** - Refer to Installing the NVIDIA Container Toolkit | NVIDIA Documentation for the installation procedure to enable the Docker repository and install the NVIDIA Container Toolkit.
3. After the NVIDIA Container Toolkit is installed, to configure the Docker container runtime, refer to the Configuration | NVIDIA Documentation.

4.3.4. Verifying the Installation of NVIDIA Container Toolkit

1. Run the `nvidia-smi` command contained in the latest official NVIDIA CUDA Toolkit image that is compatible with the release of the NVIDIA CUDA Toolkit driver that is running on your machine.

   Note

   Do not use a release of the NVIDIA CUDA Toolkit image later than the release of the NVIDIA CUDA Toolkit driver that is running on your machine. For a list of all NVIDIA CUDA Toolkit images, refer to nvidia/cuda on Docker Hub.

   ```
   $ sudo docker run --rm --runtime=nvidia --gpus all ubuntu nvidia-smi
   ```

2. Start a GPU-enabled container on any two available GPUs.

   ```
   $ docker run --runtime=nvidia --gpus 2 nvidia/cuda:12.4.0-base-ubuntu22.
   ↪04 nvidia-smi
   ```

3. Start a GPU-enabled container on two specific GPUs identified by their index numbers.

   ```
   $ docker run --runtime=nvidia --gpus '"device=1,2"' nvidia/cuda:12.4.0-
   ↪base-ubuntu22.04 nvidia-smi
   ```

4. Start a GPU-enabled container on two specific GPUs with one GPU identified by its UUID and the other GPU identified by its index number.

   ```
   $ docker run --runtime=nvidia --gpus '"device=UUID-ABCDEF,1"' nvidia/
   ↪cuda:12.4.0-base-ubuntu22.04 nvidia-smi
   ```

5. Specify a GPU capability for the container.

   ```
   $ docker run --runtime=nvidia --gpus all,capabilities=utility nvidia/
   ↪cuda:12.4.0-base-ubuntu22.04 nvidia-smi
   ```

4.3.5. Installing Software Distributed as Container Images

The NGC container images accessed through the NVIDIA NGC Catalog include the AI and data science applications and frameworks. Each container image for an AI and data science application or framework contains the entire user-space software stack that is required to run the application or framework, namely, the CUDA libraries, cuDNN, any required Magnum IO components, TensorRT, and the framework.

Ensure that you have completed the following tasks in the NGC Private Registry User Guide:

- Generating Your NGC API Key
- Accessing the NGC Container Registry

Perform this task from the host machine. Obtain the Docker pull command to download each of the following applications and deep learning framework components from the listing for the application or component in the NGC Public Catalog.

- Applications
  - NVIDIA NIMs
  - NVIDIA Clara Parabricks
  - NVIDIA DeepStream
  - NVIDIA Riva
  - MONAI - Medical Open Network for Artificial Intelligence
  - RAPIDS
  - RAPIDS Accelerator for Apache Spark
  - TAO
- Deep learning framework components
  - NVIDIA TensorRT
  - NVIDIA Triton Inference Server
  - PyTorch
  - TensorFlow 2

4.3.6. Running ResNet-50 with TensorFlow

1. Launch the **TensorFlow 1** container image on all GPUs in interactive mode, specifying that the container will be deleted when stopped.

```
$ sudo docker run --gpus all -it --rm \
nvcr.io/nvidia/tensorflow:21.07-tf1-py3
```

2. From within the container runtime, change to the directory that contains test data for cnn example.

```
# cd /workspace/nvidia-examples/cnn
```

3. Run the ResNet-50 training test with FP16 precision.

```
# python resnet.py --layers 50 -b 64 -i 200 -u batch --precision fp16
```

4. Confirm that all operations on the application are performed correctly and that a set of results is reported when the test is completed.
5. Press **Ctrl+P**, **Ctrl+Q** to exit the container runtime and return to the Linux command shell.

4.3.7. Running ResNet-50 with TensorRT

1. Launch the NVIDIA TensorRT container image on all GPUs in interactive mode, specifying that the container will be deleted when stopped.

```
$ sudo docker run --gpus all -it --rm nvcr.io/nvidia/tensorrt:21.07-py3
```

2. From within the container runtime, change to the directory that contains test data for the ResNet-50 convolutional neural network.

```
# cd /workspace/tensorrt/data/resnet50
```

3. Run the ResNet-50 convolutional neural network with FP32, FP16, and INT8 precision and confirm that each test is completed with the result PASSED.
   1. To run ResNet-50 with the default FP32 precision, run this command:

```
# trtexec --duration=90 --workspace=1024 --percentile=99 --avgRuns=100 \
--deploy=ResNet50_N2.prototxt --batch=1 --output=prob
```

   2. To run ResNet-50 with FP16 precision, add the `--fp16` option:

```
# trtexec --duration=90 --workspace=1024 --percentile=99 --avgRuns=100 \
--deploy=ResNet50_N2.prototxt --batch=1 --output=prob --fp16
```

   3. To run ResNet-50 with INT8 precision, add the `--int8` option:

```
# trtexec --duration=90 --workspace=1024 --percentile=99 --avgRuns=100 \
--deploy=ResNet50_N2.prototxt --batch=1 --output=prob --int8
```

4. Press **Ctrl+P**, **Ctrl+Q** to exit the container runtime and return to the Linux command shell.

4.3.8. Running NVIDIA NIM on Bare Metal Ubuntu 22.04

NVIDIA Inference Microservices (NIMs) provide a streamlined path for developing AI-powered enterprise applications and deploying AI models in production.You can download and run the NIM of your choice from the NVIDIA NGC Catalog. Follow the instructions below to deploy a Llama3 8B Instruct NIM on your bare metal host machine setup and run inference.

1. Pull and run `meta/llama3-8b` using Docker (this will download the full model and run it in your local environment).

```
$ docker login nvcr.io
Username: $oauthtoken
Password: <PASTE_API_KEY_HERE>
```

2. Pull and run NVIDIA NIM. This will download the optimized model for your infrastructure.

```
export NGC_API_KEY=<PASTE_API_KEY_HERE>
export LOCAL_NIM_CACHE=~/.cache/nim
mkdir -p "$LOCAL_NIM_CACHE"
docker run -it --rm \
    --gpus all \
    --shm-size=16GB \
```

(continues on next page)

(continued from previous page)

```
    -e NGC_API_KEY=$NGC_API_KEY \
    -v "$LOCAL_NIM_CACHE:/opt/nim/.cache" \
    -u $(id -u) \
    -p 8000:8000 \
    nvcr.io/nim/meta/llama3-8b-instruct:1.0.0
```

3. Make a local API call.

```
curl -X 'POST' \
'http://0.0.0.0:8000/v1/chat/completions' \
-H 'accept: application/json' \
-H 'Content-Type: application/json' \
-d '{
    "model": "meta/llama3-8b-instruct",
    "messages": [{"role":"user", "content":"Write a limerick about the
↪wonders of GPU computing."}],
    "max_tokens": 64
}'
```

For more information about running inference on this locally deployed LLM NIM, refer to Launch NVIDIA NIM for LLMs.

4.4. Installing NVIDIA AI Enterprise on Public Cloud

NVIDIA AI Enterprise can be run on Amazon Web Services (AWS), Google Cloud, Microsoft Azure, Oracle Cloud Infrastructure (OCI), Alibaba Cloud, and Tencent Cloud.

This section of the NVIDIA AI Enterprise Quick Start Guide provides minimal instructions for deploying NVIDIA AI Enterprise on Microsoft Azure using the NVIDIA AI Enterprise VMI.

4.4.1. Installing NVIDIA AI Enterprise on Microsoft Azure using the NVIDIA AI Enterprise VMI

The NVIDIA AI Enterprise On-Demand VMI spins up a GPU-accelerated Compute Engine VM instance in minutes with pre-installed software for accelerating your Machine Learning, Deep Learning, Data Science, and HPC workloads.

The On-Demand VMI is preconfigured with the following software:

- Ubuntu Operating System
- NVIDIA GPU Data Center Driver
- Docker-ce
- NVIDIA Container Toolkit
- CSP CLI, NGC CLI
- Miniconda, JupyterLab, Git

- Token Activation Script

Getting started with NVIDIA AI Enterprise in your Enterprise (On-Demand) VMI cloud instance can be broken down into two simple steps:

1. Authorize the VMI cloud instance with NVIDIA NGC by copying over the provided instance ID token into the Activate Subscription page on NGC. There are four key steps to complete this part of the process:
    1. Get an identity token from the VMI.
    2. Activate your NVIDIA AI Enterprise subscription with the token.
    3. Generate an API key for accessing the catalog.
    4. Put the API key on the VMI.

    Follow the instructions outlined here to complete this step.
2. Pulling and running NVIDIA AI Enterprise Containers. Refer to this section of the Cloud Deployment Guide for pulling and running NGC container images through the NVIDIA NGC Catalog.

Detailed instructions on how to install NVIDIA AI Enterprise on the public cloud can be found in the Microsoft Azure Overview.

4.4.2. Running an LLM NIM on Microsoft Azure using NVIDIA AI Enterprise

Refer to the Appendix section of the Cloud Deployment Guide for pulling and running NGC container images through the NVIDIA NGC Catalog.

4.5. Installing NVIDIA AI Enterprise in Virtualized Environments using NVIDIA vGPU (C-Series)

The NVIDIA AI Enterprise VMware Deployment Guide offers detailed information for deploying NVIDIA AI Enterprise on a third-party NVIDIA-certified system running VMware vSphere using NVIDIA vGPU (C-Series).

4.5.1. NVIDIA AI Enterprise Software Prerequisites

1. NVIDIA vGPU (C-Series) Host Driver
2. NVIDIA vGPU (C-Series) Guest Driver
3. NVIDIA License System

To download the NVIDIA vGPU (C-Series) software drivers, follow the instructions in *Accessing the NVIDIA AI Enterprise Infrastructure Resources*.

The NVIDIA AI Enterprise license entitles customers to download NVIDIA vGPU (C-Series) software through *NVIDIA NGC*. Deploying NVIDIA AI Enterprise using vGPU requires a valid license that is enforced through the installation and the NVIDIA License System. The NVIDIA License System is used to

serve a pool of floating licenses to licensed NVIDIA software products and is configured with licenses obtained from the NVIDIA Licensing Portal.

NVIDIA License System supports the following types of service instances:

- Cloud License Service (CLS) instance. A CLS instance is hosted on the NVIDIA Licensing Portal.
- Delegated License Service (DLS) instance. A DLS instance is hosted on-premises at a location that is accessible from your private network, such as inside your data center.

An NVIDIA vGPU (C-Series) client VM with a network connection obtains a license by leasing it from an NVIDIA License System service instance. The service instance serves the license to the client over the network from a pool of floating licenses obtained from the NVIDIA Licensing Portal. The license is returned to the service instance when the licensed client no longer requires the license.

To activate an NVIDIA vGPU (C-Series), software licensing must be configured for the vGPU VM client when it is booted. NVIDIA vGPU (C-Series) VMs run at a reduced capability until a license is acquired.

Refer to the NVIDIA Licensing Quick Start Guide for instructions on configuring an express Cloud License Service (CLS) instance and verifying the license status of a licensed vGPU VM client.

Refer to the NVIDIA License System User Guide if you use Delegated License Service (DLS) instances to serve licenses. Detailed instructions on how to install, configure, and manage the NVIDIA License System can be found here.

4.6. Obtaining NVIDIA Base Command Manager Essentials

NVIDIA Base Command Manager Essentials streamlines cluster provisioning, workload management, and infrastructure monitoring in the data center. In bare-metal deployments, NVIDIA Base Command Manager Essentials simplifies the installation of operating systems supported by NVIDIA Base Command Manager Essentials.

Before obtaining NVIDIA Base Command Manager Essentials, ensure that you have activated the accounts for getting NVIDIA AI Enterprise, as explained in *Activating the Accounts for Getting NVIDIA AI Enterprise*.

1. Request your NVIDIA Base Command Manager Essentials product keys by sending an email with your entitlement certificate to sw-bright-sales-ops@NVIDIA.onmicrosoft.com. After your entitlement certificate has been reviewed, you will receive a product key from which you can generate a license key for the number of licenses that you purchased.
2. Go to this page to download NVIDIA Base Command Manager Essentials for your operating system.

For detailed instructions on deploying and using Base Command Manager Essentials, refer to the Base Command Manager Essentials Product Manuals.

After obtaining NVIDIA Base Command Manager Essentials, follow the steps in the NVIDIA Base Command Manager Essentials Installation Manual to create and license your head node.

Chapter 5. Deployment Guide

5.1. NVIDIA AI Enterprise Deployment Guides

Table 1: NVIDIA AI Enterprise Deployment Guides

Deployment	Deployment Guide	Description
Public Cloud	NVIDIA AI Enterprise Cloud Deployment Guide	Use this guide to deploy and run NVIDIA AI Enterprise in the Cloud.
On-Premises: Virtualized Environment	NVIDIA AI Enterprise VMware vSphere Deployment Guide	This document provides insights into deploying NVIDIA AI Enterprise for VMware vSphere.
	NVIDIA AI Enterprise Red Hat Enterprise Linux With KVM Deployment Guide	This document provides insights into deploying NVIDIA AI Enterprise on Red Hat Enterprise Linux with KVM Virtualization.
	NVIDIA AI Enterprise OpenShift on VMware vSphere Deployment Guide	This document provides insights into deploying NVIDIA AI Enterprise with Red Hat OpenShift on VMware vSphere.
On-Premises: Bare Metal Environment	NVIDIA AI Enterprise Bare Metal Deployment Guide	This document provides insights into deploying NVIDIA AI Enterprise on Bare Metal Servers.
	NVIDIA AI Enterprise OpenShift on Bare Metal Deployment Guide	This document provides insights into deploying NVIDIA AI Enterprise with Red Hat OpenShift on Bare Metal Servers.
Containerized Environments	Run:ai & NVIDIA AI Enterprise Deployment Guide	This document provides a validated deployment guide for deploying Run:ai Atlas Platform on NVIDIA AI Enterprise leveraging a VMware vSphere Tanzu cluster.
	Domino & NVIDIA AI Enterprise Deployment Guide	This document describes the Domino Data Lab's Enterprise MLOps Platform for NVIDIA AI Enterprise deployed into a Kubernetes cluster hosted by VMware vSphere and using VMware vSAN storage.
	UbiOps & NVIDIA AI Enterprise Deployment Guide	This document provides a validated deployment guide for deploying the UbiOps MLOps Platform.
	NVIDIA AI Enterprise and Canonical Charmed Kubernetes Deployment Guide	This document provides a comprehensive guide for installing Charmed Kubernetes with NVIDIA GPU Operator.
	HPE ML Data Management Deployment Guide	Learn about HPE ML Data Management (MLDM) basics and how to install the platform within a Kubernetes cluster.
CPU-Only Deployment	NVIDIA AI Enterprise CPU Only Deployment Guide	This document provides insights into CPU-only deployments of NVIDIA AI Enterprise.
Multi-Node Deployment	NVIDIA AI Enterprise Multi-Node Deep Learning Training with TensorFlow	This guide aims to guide how to set up a high-performance multi-node cluster as a virtual machine using Docker.

5.2. Use Cases and Examples

Table 2: Use Cases and Examples

Deployment Guide	Description
NVIDIA AI Enterprise with RAPIDS Accelerator Deployment Guide	NVIDIA RAPIDS Accelerator for Apache Spark enables data engineers to speed up Apache Spark 3 data science pipelines and AI model training while lowering infrastructure costs.
NVIDIA AI Enterprise Natural Language Processing with Triton Inference Server	AI pipeline on NVIDIA AI Enterprise by leveraging a Natural Language Processing (NLP) use case example.

Chapter 6. NVIDIA vGPU (C-Series) Documentation

6.1. Release Notes

6.1.1. Supported Hardware

6.1.1.1 Microsoft Windows Guest Operating Systems Supported

NVIDIA AI Enterprise supports only the Tesla Compute Cluster (TCC) driver model for Windows guest drivers.

Windows guest OS support is limited to running applications natively in Windows VMs without containers. NVIDIA AI Enterprise features that depend on the containerization of applications are not supported on Windows guest operating systems.

If you are using a supported generic Linux with KVM hypervisor, consult the documentation from your hypervisor vendor for information about Windows releases supported as a guest OS.

Table 1: Microsoft Windows Guest Operating Systems Supported

Guest OS	Red Hat Enterprise Linux KVM	Ubuntu	VMware vSphere
Microsoft Windows Server 2022	▶ 8.8 ▶ 8.10 ▶ 9.2 ▶ 9.4	Not Supported	▶ 7.0 update 3 ▶ 8.0
Microsoft Windows 11	Not Supported	▶ 20.04 LTS ▶ 22.04 LTS	▶ 7.0 update 3 ▶ 8.0
Microsoft Windows 10	Not Supported	▶ 20.04 LTS ▶ 22.04 LTS	▶ 7.0 update 3 ▶ 8.0

6.1.2. Requirements for Using NVIDIA vGPU (C-Series)

Because NVIDIA vGPU (C-Series) have large BAR memory settings, using these vGPUs has some restrictions on VMware ESXi.

- The guest OS must be a 64-bit OS.
- 64-bit MMIO and EFI boot must be enabled for the VM.
- The guest OS must be able to be installed in EFI boot mode.
- The VM's MMIO space must be increased to 64 GB as explained in VMware Knowledge Base Article: VMware vSphere VMDirectPath I/O: Requirements for Platforms and Devices (2142307).

6.1.3. Requirements for Using NVIDIA vGPU (C-Series) on GPUs Requiring 64 GB or More of MMIO Space with Large-Memory VMs

Some GPUs require 64 GB or more of MMIO space. When a vGPU on a GPU that requires 64 GB or more of MMIO space is assigned to a VM with 32 GB or more of memory on ESXi, the VM's MMIO space must be increased to the amount of MMIO space that the GPU requires.

For more information, refer to the VMware Knowledge Base Article: VMware vSphere VMDirectPath I/O: Requirements for Platforms and Devices (2142307).

No extra configuration is needed.

The following table lists the GPUs that require 64 GB or more of MMIO space and the amount of MMIO space that each GPU requires.

Table 2: Requirements for Using NVIDIA vGPU (C-Series) on GPUs Requiring 64 GB or More of MMIO Space with Large-Memory VMs

GPU	MMIO Space Required
NVIDIA A10	64 GB
NVIDIA A30	64 GB
NVIDIA A40	128 GB
NVIDIA A100 40GB (all variants)	128 GB
NVIDIA A100 80GB (all variants)	256 GB
NVIDIA RTX A5000	64 GB
NVIDIA RTX A5500	64 GB
NVIDIA RTX A6000	128 GB
Quadro RTX 6000 Passive	64 GB
Quadro RTX 8000 Passive	64 GB
Tesla V100 (all variants)	64 GB

6.1.4. NVIDIA CUDA Toolkit Version Support

The releases in this release family of NVIDIA AI Enterprise support NVIDIA CUDA Toolkit 12.3.

To build a CUDA application, the system must have the NVIDIA CUDA Toolkit and the libraries required for linking. For details on the components of the NVIDIA CUDA Toolkit, refer to NVIDIA CUDA Toolkit Release Notes for CUDA 12.3.

To run a CUDA application, the system must have a CUDA-enabled GPU and an NVIDIA display driver that is compatible with the NVIDIA CUDA Toolkit release that was used to build the application. If the application relies on dynamic linking for libraries, the system must also have the correct version of these libraries.

For more information about the NVIDIA CUDA Toolkit, refer to CUDA Toolkit 12.3 Documentation.

6.1.5. NVIDIA vGPU (C-Series) Migration Support

NVIDIA vGPU (C-Series) Migration, which includes vMotion and suspend-resume, is supported for both time-sliced and MIG-backed vGPUs on all supported GPUs and guest operating systems but only on a subset of supported hypervisor software releases.

Limitations with NVIDIA vGPU (C-Series) Migration Support

Red Hat Enterprise Linux with KVM: Migration between hosts that are running different versions of the NVIDIA Virtual GPU Manager driver is not supported, even within the same NVIDIA Virtual GPU Manager driver branch.

NVIDIA vGPU (C-Series) migration is disabled for a VM for which any of the following NVIDIA CUDA Toolkit features is enabled:

- Unified memory
- Debuggers
- Profilers

Supported Hypervisor Software Releases

Since Red Hat Enterprise Linux with KVM 9.4

Not supported on Ubuntu

All supported releases of VMware vSphere

Known Issues with NVIDIA vGPU (C-Series) Migration Support

Table 3: Requirements for Using NVIDIA vGPU (C-Series) on GPUs Requiring 64 GB or More of MMIO Space with Large-Memory VMs

Use Case	Affected GPUs	Issue
Migration between hosts with different ECC memory configuration	All GPUs that support vGPU migration	Migration of VMs configured with vGPU stops before the migration is complete

6.1.5.1 vGPUs that Support Multiple vGPUs Assigned to a VM

The supported vGPUs depend on the hypervisor:

- For generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu, **all** NVIDIA vGPU (C-Series) are supported. On GPUs that support the Multi-Instance GPU (MIG) feature, both time-sliced and MIG-backed vGPUs are supported.
- For VMware vSphere, the supported vGPUs depend on the hypervisor release:
  - **Since VMware vSphere 8.0**: All Q-series and NVIDIA vGPU (C-Series) are supported. On GPUs that support the Multi-Instance GPU (MIG) feature, both time-sliced and MIG-backed vGPUs are supported.
  - **VMware vSphere 7.x releases**: Only Q-series and NVIDIA vGPU (C-Series) that are allocated all of the physical GPU's frame buffer are supported.

You can assign multiple vGPUs with differing amounts of frame buffer to a single VM, provided the board type and the series of all the vGPUs are the same. For example, you can assign an A40-48C vGPU and an A40-16C vGPU to the same VM. However, you cannot assign an A30-8C vGPU and an A16-8C vGPU to the same VM.

Table 4: Multiple vGPU Support on the NVIDIA Ada Lovelace Architecture

Board	vGPU
NVIDIA L40S	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) ▶ All Q-series vGPUs Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) ▶ All Q-series vGPUs VMware vSphere 7.x releases: ▶ L40S-48C ▶ L40S-48Q
NVIDIA L40	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) ▶ All Q-series vGPUs Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) ▶ All Q-series vGPUs VMware vSphere 7.x releases: ▶ L40-48C ▶ L40-48Q
▶ NVIDIA L20 ▶ NVIDIA L20 liquid cooled	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) ▶ All Q-series vGPUs Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) ▶ All Q-series vGPUs VMware vSphere 7.x releases: ▶ L20-48C ▶ L20-48Q
NVIDIA L4	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) ▶ All Q-series vGPUs Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) ▶ All Q-series vGPUs VMware vSphere 7.x releases: ▶ L4-24C ▶ L4-24Q
NVIDIA L2	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) ▶ All Q-series vGPUs Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) ▶ All Q-series vGPUs VMware vSphere 7.x releases: ▶ L2-24C

Table 5: Multiple vGPU Support on the NVIDIA Hopper GPU Architecture

Board	vGPU[?]
NVIDIA H800 PCIe 94GB	All NVIDIA vGPU (C-Series)
NVIDIA H800 PCIe 80GB	All NVIDIA vGPU (C-Series)
NVIDIA H800 SXM5 80GB	All NVIDIA vGPU (C-Series)
NVIDIA H100 PCIe 94GB (H100 NVL)	All NVIDIA vGPU (C-Series)
NVIDIA H100 SXM5 94GB	All NVIDIA vGPU (C-Series)
NVIDIA H100 PCIe 80GB	All NVIDIA vGPU (C-Series)
NVIDIA H100 SXM5 80GB	All NVIDIA vGPU (C-Series)
NVIDIA H100 SXM5 64GB	All NVIDIA vGPU (C-Series)

[1] This type of vGPU cannot be assigned with other types of vGPU to the same VM.

Table 6: Multiple vGPU Support on the NVIDIA Ampere GPU Architecture

Board	vGPU[?]
▶ NVIDIA A800 PCIe 80GB ▶ NVIDIA A800 PCIe 80GB liquid-cooled ▶ NVIDIA AX800	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ A800D-80C
NVIDIA A800 HGX 80GB	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ A800DX-80C
NVIDIA A800 PCIe 40GB active-cooled	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ A800-40C
▶ NVIDIA A100 PCIe 80GB ▶ NVIDIA A100 PCIe 80GB liquid-cooled	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ A100D-80C
NVIDIA A100 HGX 80GB	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ A100DX-80C
NVIDIA A100 PCIe 40GB	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ A100-40C
NVIDIA A100 HGX 40GB	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0:

Table 7: Multiple vGPU Support on the NVIDIA Turing GPU Architecture

Board	vGPU
Tesla T4	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) ▶ All Q-series vGPUs Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ T4-16C
Quadro RTX 6000 passive	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ RTX6000P-24C
Quadro RTX 8000 passive	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ RTX8000P-48C

Table 8: Multiple vGPU Support on the NVIDIA Volta GPU Architecture

Board	vGPU
Tesla V100 SXM2 32GB	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ V100D-32C
Tesla V100 PCIe 32GB	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ V100D-32C
Tesla V100S PCIe 32GB	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ V100S-32C
Tesla V100 SXM2	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ V100X-16C
Tesla V100 PCIe	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ V100-16C
Tesla V100 FHHL	Generic Linux with KVM hypervisors, Red Hat Enterprise Linux KVM, and Ubuntu: ▶ All NVIDIA vGPU (C-Series) Since VMware vSphere 8.0: ▶ All NVIDIA vGPU (C-Series) VMware vSphere 7.x releases: ▶ V100L-16C

6.1.5.2 vGPUs that Support Peer-to-Peer CUDA Transfers

Only Q-series and C-series time-sliced vGPUs that are allocated all of the physical GPU's frame buffer on physical GPUs that support NVLink are supported.

Table 9: Peer-to-Peer CUDA Transfer Support on the NVIDIA Hopper GPU Architecture

Board	vGPU
NVIDIA H800 PCIe 94GB	H800L-94C
NVIDIA H800 PCIe 80GB	H800-80C
NVIDIA H100 PCIe 94GB (H100 NVL)	H100L-94C
NVIDIA H100 SXM5 94GB	H100XL-94C
NVIDIA H100 PCIe 80GB	H100-80C
NVIDIA H100 SXM5 80GB	H100XM-80C
NVIDIA H100 SXM5 64GB	H100XS-64C

Table 10: Peer-to-Peer CUDA Transfer Support on the NVIDIA Ampere GPU Architecture

Board	vGPU
▶ NVIDIA A800 PCIe 80GB ▶ NVIDIA A800 PCIe 80GB liquid-cooled ▶ NVIDIA AX800	A800D-80C
NVIDIA A800 HGX 80GB	A800DX-80C[2]
NVIDIA A800 PCIe 40GB active-cooled	A800-40C
▶ NVIDIA A100 PCIe 80GB ▶ NVIDIA A100 PCIe 80GB liquid-cooled ▶ NVIDIA A100X	A100D-80C
NVIDIA A100 HGX 80GB	A100DX-80C?
NVIDIA A100 PCIe 40GB	A100-40C
NVIDIA A100 HGX 40GB	A100X-40C?
NVIDIA A40	▶ A40-48Q ▶ A40-48C
▶ NVIDIA A30 ▶ NVIDIA A30X ▶ NVIDIA A30 liquid-cooled	A30-24C
NVIDIA A10	▶ A10-24Q ▶ A10-24C
NVIDIA RTX A6000	▶ A6000-48Q ▶ A6000-48C
NVIDIA RTX A5500	▶ A5500-24Q ▶ A5500-24C
NVIDIA RTX A5000	▶ A5000-24Q ▶ A5000-24C

[2] Supported only on the following hardware:

- NVIDIA HGX A100 4-GPU baseboard with four fully connected GPUs
- NVIDIA HGX A100 8-GPU baseboards with eight fully connected GPUs

Fully connected means that each GPU is connected to every other GPU on the baseboard.

Table 11: Peer-to-Peer CUDA Transfer Support on the NVIDIA Turing GPU Architecture

Board	vGPU
Quadro RTX 6000 passive	▶ RTX6000P-24Q ▶ RTX6000P-24C
Quadro RTX 8000 passive	▶ RTX8000P-48Q ▶ RTX8000P-48C

Table 12: Peer-to-Peer CUDA Transfer Support on the NVIDIA Volta GPU Architecture

Board	vGPU
Tesla V100 SXM2 32GB	▶ V100DX-32Q ▶ V100DX-32C
Tesla V100 SXM2	▶ V100X-16Q ▶ V100X-16C

6.1.6. GPUDirect Technology Support

NVIDIA GPUDirect Remote Direct Memory Access (RDMA) technology enables network devices to directly access the vGPU frame buffer, bypassing CPU host memory altogether. GPUDirect Storage technology enables a direct data path for direct memory access (DMA) transfers between GPU memory and storage. GPUDirect technology is supported only on a subset of vGPUs and guest OS releases.

Supported vGPUs

GPUDirect RDMA and GPUDirect Storage technology are supported on all time-sliced and MIG-backed NVIDIA vGPU (C-Series) on physical GPUs that support single root I/O virtualization (SR-IOV).

- GPUs based on the NVIDIA Ada Lovelace GPU architecture:
  - NVIDIA L40
  - NVIDIA L40S
  - NVIDIA L20
  - NVIDIA L20 liquid-cooled
  - NVIDIA L4
  - NVIDIA L2
  - NVIDIA RTX 6000 Ada

- NVIDIA RTX 5880 Ada
- NVIDIA RTX 5000 Ada

- GPUs based on the NVIDIA Hopper GPU architecture:
  - NVIDIA H800 PCIe 94GB
  - NVIDIA H800 PCIe 80GB
  - NVIDIA H800 SXM5 80GB
  - NVIDIA H100 PCIe 94GB (H100 NVL)
  - NVIDIA H100 SXM5 94GB
  - NVIDIA H100 PCIe 80GB
  - NVIDIA H100 SXM5 80GB
  - NVIDIA H100 SXM5 64GB
- GPUs based on the NVIDIA Ampere GPU architecture:
  - NVIDIA A800 PCIe 80GB
  - NVIDIA A800 PCIe 80GB liquid-cooled
  - NVIDIA A800 HGX 80GB
  - NVIDIA AX800
  - NVIDIA A800 PCIe 40GB active-cooled
  - NVIDIA A100 PCIe 80GB
  - NVIDIA A100 PCIe 80GB liquid-cooled
  - NVIDIA A100 HGX 80GB
  - NVIDIA A100 PCIe 40GB
  - NVIDIA A100 HGX 40GB
  - NVIDIA A100X
  - NVIDIA A30
  - NVIDIA A30 liquid-cooled
  - NVIDIA A30X
  - NVIDIA A40
  - NVIDIA A16
  - NVIDIA A10
  - NVIDIA A2
  - NVIDIA RTX A6000
  - NVIDIA RTX A5500
  - NVIDIA RTX A5000

Supported Guest OS Releases

Linux only. GPUDirect technology is **not** supported on Windows.

Supported Network Interface Cards

GPUDirect technology is supported on the following network interface cards:

- NVIDIA ConnectX- 7 SmartNIC
- Mellanox Connect-X 6 SmartNIC
- Mellanox Connect-X 5 Ethernet adapter card

Limitations

Starting with GPUDirect Storage technology release 1.7.2, the following limitations apply:

- GPUDirect Storage technology is not supported on GPUs based on the NVIDIA Ampere GPU architecture.
- On GPUs based on the NVIDIA Hopper GPU architecture and the NVIDIA Ada Lovelace GPU architecture, GPUDirect Storage technology is supported only with the guest driver for Linux that is based on NVIDIA Linux open GPU kernel modules.

GPUDirect Storage technology releases before 1.7.2 are supported only with guest drivers with Linux kernel versions earlier than 6.6.

GPUDirect Storage technology is supported only on the following guest OS releases:

- Ubuntu 22.04 LTS
- Ubuntu 20.04 LTS

6.1.7. NVIDIA NVSwitch On-Chip Memory Fabric Support

NVIDIA NVSwitch on-chip memory fabric enables peer-to-peer vGPU communication within a single node over the NVLink fabric. NVSwitch on-chip memory fabric is supported only on a subset of hardware platforms, vGPUs, hypervisor software releases, and guest OS releases.

For information about how to use the NVSwitch on-chip memory fabric, see Fabric Manager for NVIDIA NVSwitch Systems User Guide (PDF).

6.1.7.1 Hardware Platforms that Support NVIDIA NVSwitch On-Chip Memory Fabric

- NVIDIA HGX H800 8-GPU baseboard
- NVIDIA HGX H100 8-GPU baseboard
- NVIDIA HGX A100 8-GPU baseboard

6.1.7.2 vGPUs that Support NVIDIA NVSwitch On-Chip Memory Fabric

Only C-series time-sliced vGPUs that are allocated all of the physical GPU's frame buffer on NVIDIA H800 and NVIDIA H100 SXM5 physical GPUs, and NVIDIA A800 and NVIDIA A100 HGX physical GPUs are supported.

Table 13: NVIDIA NVSwitch On-Chip Memory Fabric Support on the NVIDIA Hopper GPU Architecture

Board	vGPU
NVIDIA H800 SXM5 80GB	H800XM-80C
NVIDIA H100 SXM5 80GB	H100XM-80C

Table 14: NVIDIA NVSwitch On-Chip Memory Fabric Support on the NVIDIA Ampere GPU Architecture

Board	vGPU
NVIDIA A800 HGX 80GB	A800DX-80C
NVIDIA A100 HGX 80GB	A100DX-80C
NVIDIA A100 HGX 40GB	A100X-40C

6.1.7.3 Hypervisor Releases that Support NVIDIA NVSwitch On-Chip Memory Fabric

For information about which generic Linux with KVM hypervisor software releases support NVIDIA NVSwitch on-chip memory fabric, consult the documentation from your hypervisor vendor.

All supported Red Hat Enterprise Linux KVM releases support NVIDIA NVSwitch on-chip memory fabric.

On the Ubuntu hypervisor, NVSwitch is not supported.

The earliest VMware vSphere Hypervisor (ESXi) release that supports NVIDIA NVSwitch on-chip memory fabric depends on the GPU architecture.

Table 15: Hypervisor Releases that Support NVIDIA NVSwitch On-Chip Memory Fabric

GPU Architecture	Earliest Supported VMware vSphere Hypervisor (ESXi) Release
NVIDIA Hopper	VMware vSphere Hypervisor (ESXi) 8 update 2
NVIDIA Ampere	VMware vSphere Hypervisor (ESXi) 8 update 1

6.1.7.4 Guest OS Releases that Support NVIDIA NVSwitch On-Chip Memory Fabric

Linux only. NVIDIA NVSwitch on-chip memory fabric is **not** supported on Windows.

6.1.7.5 Limitations on Support for NVIDIA NVSwitch On-Chip Memory Fabric

- Only time-sliced vGPUs are supported. MIG-backed vGPUs are **not** supported.
- On the Ubuntu hypervisor, NVSwitch is not supported.
- GPU passthrough is **not** supported.
- SLI is not supported.
- All vGPUs that are communicating peer-to-peer must be assigned to the same VM.
- On GPUs that are based on the NVIDIA Hopper GPU architecture, multicast is **not** supported.

6.1.8. vGPUs that Support Unified Memory

On GPUs that support the Multi-Instance GPU (MIG) feature, all MIG-backed vGPUs are supported. Only time-sliced Q-series and NVIDIA vGPU (C-Series) that are allocated all of the physical GPU's frame buffer on physical GPUs that support unified memory are supported.

Table 16: Unified Memory Support on the NVIDIA Ada Lovelace GPU Architecture

Board	vGPU
NVIDIA L40	▶ L40-48Q ▶ L40-48C
NVIDIA L40S	▶ L40S-48Q ▶ L40S-48C
▶ NVIDIA L20 ▶ NVIDIA L20 liquid-cooled	▶ L20-48Q ▶ L20-48C
NVIDIA L4	▶ L4-24Q ▶ L4-24C
NVIDIA L2	▶ L2-24Q ▶ L2-24C
NVIDIA RTX 6000 Ada	▶ RTX 6000 Ada-48Q ▶ RTX 6000 Ada-48C
NVIDIA RTX 5880 Ada	▶ RTX 5880 Ada-48Q ▶ RTX 5880 Ada-48C
NVIDIA RTX 5000 Ada	▶ RTX 5000 Ada-32Q ▶ RTX 6000 Ada-32C

Table 17: Unified Memory Support on the NVIDIA Hopper GPU Architecture

Board	vGPU
NVIDIA H800 PCIe 94GB	▶ H800L-94C ▶ All MIG-backed vGPUs
NVIDIA H800 PCIe 80GB	▶ H800-80C ▶ All MIG-backed vGPUs
NVIDIA H800 SXM5 80GB	▶ H800XM-80C ▶ All MIG-backed vGPUs
NVIDIA H100 PCIe 94GB (H100 NVL)	▶ H100L-94C ▶ All MIG-backed vGPUs
NVIDIA H100 SXM5 94GB	▶ H100XL-94C ▶ All MIG-backed vGPUs
NVIDIA H100 PCIe 80GB	▶ H100-80C ▶ All MIG-backed vGPUs
NVIDIA H100 SXM5 80GB	▶ H100XM-80C ▶ All MIG-backed vGPUs
NVIDIA H100 SXM5 64GB	▶ H100XS-64C ▶ All MIG-backed vGPUs

Table 18: Unified Memory Support on the NVIDIA Ampere GPU Architecture

Board	vGPU
▶ NVIDIA A800 PCIe 80GB ▶ NVIDIA A800 PCIe 80GB liquid-cooled ▶ NVIDIA AX800	▶ A800D-80C ▶ All MIG-backed vGPUs
NVIDIA A800 HGX 80GB	▶ A800DX-80C ▶ All MIG-backed vGPUs
NVIDIA A800 PCIe 40GB active-cooled	▶ A800-40C ▶ All MIG-backed vGPUs
▶ NVIDIA A100 PCIe 80GB ▶ NVIDIA A100 PCIe 80GB liquid-cooled ▶ NVIDIA A100X	▶ A100D-80C ▶ All MIG-backed vGPUs
NVIDIA A100 HGX 80GB	▶ A100DX-80C ▶ All MIG-backed vGPUs
NVIDIA A100 PCIe 40GB	▶ A100-40C ▶ All MIG-backed vGPUs
NVIDIA A100 HGX 40GB	▶ A100X-40C ▶ All MIG-backed vGPUs
NVIDIA A40	A40-48C
▶ NVIDIA A30 ▶ NVIDIA A30X ▶ NVIDIA A30 liquid-cooled	▶ A30-24C ▶ All MIG-backed vGPUs
NVIDIA A16	▶ A16-16Q ▶ A16-16C
NVIDIA A10	▶ A10-24Q ▶ A10-24C
NVIDIA RTX A6000	▶ A6000-48Q ▶ A6000-48C
NVIDIA RTX A5500	▶ A5500-24Q ▶ A5500-24C
NVIDIA RTX A5000	

6.1.9. nvidia-smi cannot report GPU utilization for MIG instances

When Multi-Instance GPU (MIG) mode is enabled for a GPU, the `nvidia-smi` command cannot report any GPU engine utilization for MIG instances. To monitor GPU engine utilization for MIG instances, run the `nvidia-smi vgpu` command with the `--gpm-metrics ID-list` option.

The following example shows the output from the nvidia-smi for a GPU for which MIG mode is enabled.

```
[root@host ~]# nvidia-smi
Fri Jun 14 11:45:28 2024
+-----------------------------------------------------------------------------
↪-+
| NVIDIA-SMI 550.90.05    Driver Version: 550.90.05   CUDA Version:  12.4
↪ |
|-------------------------------+----------------------+----------------------
↪-+
|-------------------------------+----------------------+----------------------
↪-+
| 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 GRID A100-2-10C On          | 00000000:02:02.0 Off | On
↪ |
| N/A N/A P0 N/A / N/A          | 2556MiB / 10235MiB   | N/A Default
↪ |
|                               |                      | Enabled
↪ |
+-------------------------------+----------------------+----------------------
↪-+

+-----------------------------------------------------------------------------
↪-+
| MIG devices:
↪ |
+------------------+----------------------+-----------+----------------------
↪-+
| GPU GI CI MIG    | Memory-Usage         | Vol       | Shared
↪ |
| ID ID Dev        | BAR1-Usage           | SM Unc    | CE ENC DEC OFA JPG
↪ |
|                  |                      | ECC       |
↪ |
|==================+======================+===========+======================|
| 0 0 0 0          | 2556MiB / 10235MiB   | 28 0      | 2 0 1 0 0
↪ |
|                  | 5MiB / 4096MiB       |           |
↪ |
+------------------+----------------------+-----------+----------------------
↪-+

+-----------------------------------------------------------------------------
```

(continues on next page)

(continued from previous page)

```
↪-+
| Processes:
↪ |
| GPU GI CI PID Type Process name GPU Memory
↪ |
| ID ID Usage
↪ |
|=============================================================================|
| 0 0 0 2843 C python3 1516MiB |
```

6.1.10. Total frame buffer for vGPUs is less than the total frame buffer on the physical GPU

Some of the physical GPU's frame buffer is used by the hypervisor on behalf of the VM for allocations that the guest OS would otherwise have made in its frame buffer. The frame buffer used by the hypervisor is not available for vGPUs on the physical GPU. In NVIDIA vGPU deployments, the frame buffer for the guest OS is reserved in advance, whereas in bare-metal deployments, the frame buffer for the guest OS is reserved based on the runtime needs of applications.

If error-correcting code (ECC) memory is enabled on a physical GPU that does not have HBM2 memory, the amount of frame buffer that is usable by vGPUs is further reduced. All types of vGPUs are affected, not just vGPUs that support ECC memory.

On all GPUs that support ECC memory and, therefore, dynamic page retirement, an additional frame buffer is allocated for dynamic page retirement. The amount that is allocated is inversely proportional to the maximum number of vGPUs per physical GPU. All GPUs that support ECC memory are affected, even GPUs that have HBM2 memory or for which ECC memory is disabled.

The approximate amount of frame buffer that NVIDIA AI Enterprise reserves can be calculated from the following formula:

`max-reserved-fb = vgpu-profile-size-in-mb÷16 + 16 + ecc-adjustments + page-retirement-allocation + compression-adjustment`

`max-reserved-fb` - The maximum total amount of reserved frame buffer in Mbytes that is not available for vGPUs.

`vgpu-profile-size-in-mb` - The amount of frame buffer in Mbytes allocated to a single vGPU. This amount depends on the vGPU type. For example, for the T4-16Q vGPU type, `vgpu-profile-size-in-mb` is 16384.

`ecc-adjustments` - The amount of frame buffer in Mbytes that is not usable by vGPUs when ECC is enabled on a physical GPU that does not have HBM2 memory.

- If ECC is enabled on a physical GPU that does not have HBM2 memory `ecc-adjustments` is `fb-without-ecc/16`, which is equivalent to 64 Mbytes for every Gbyte of frame buffer assigned to the vGPU. `fb-without-ecc` is the total amount of frame buffer with ECC disabled.
- If ECC is disabled or the GPU has HBM2 memory, `ecc-adjustments` is 0.

`page-retirement-allocation` - The amount of frame buffer in Mbytes that is reserved for dynamic page retirement.

  - On GPUs based on the NVIDIA Maxwell GPU architecture, `page-retirement-allocation = 4÷max-vgpus-per-gpu`.

- On GPUs based on NVIDIA GPU architectures after the Maxwell architecture, `page-retirement-allocation = 128÷max-vgpus-per-gpu`.

`max-vgpus-per-gpu` - The maximum number of vGPUs that can be created simultaneously on a physical GPU. This number varies according to the vGPU type. For example, for the T4-16Q vGPU type, `max-vgpus-per-gpu` is 1.

`compression-adjustment` - The amount of frame buffer in Mbytes that is reserved for the higher compression overhead in vGPU types with 12 Gbytes or more of frame buffer on GPUs based on the Turing architecture.

`compression-adjustment` depends on the vGPU type as shown in the following table.

Table 19: Total frame buffer for vGPUs is less than the total frame buffer on the physical GPU

vGPU Type	Compression Adjustment (MB)
▶ T4-16Q ▶ T4-16C ▶ T4-16A	28
▶ RTX6000-12Q ▶ RTX6000-12C ▶ RTX6000-12A	32
▶ RTX6000-24Q ▶ RTX6000-24C ▶ RTX6000-24A	104
▶ RTX6000P-12Q ▶ RTX6000P-12C ▶ RTX6000P-12A	32
▶ RTX6000P-24Q ▶ RTX6000P-24C ▶ RTX6000P-24A	104
▶ RTX8000-12Q ▶ RTX8000-12C ▶ RTX8000-12A	32
▶ RTX8000-16Q ▶ RTX8000-16C ▶ RTX8000-16A	64
▶ RTX8000-24Q ▶ RTX8000-24C ▶ RTX8000-24A	96
▶ RTX8000-48Q ▶ RTX8000-48C ▶ RTX8000-48A	238
▶ RTX8000P-12Q ▶ RTX8000P-12C ▶ RTX8000P-12A	32
▶ RTX8000P-16Q ▶ RTX8000P-16C ▶ RTX8000P-16A	64

For all other vGPU types, `compression-adjustment` is 0.

6.1.11. Single vGPU benchmark scores are lower than passthrough GPU

Description

A single vGPU configured on a physical GPU produces lower benchmark scores than the physical GPU run in passthrough mode.

Aside from performance differences that may be attributed to a vGPU's smaller frame buffer size, vGPU incorporates a performance balancing feature known as a Frame Rate Limiter (FRL). On vGPUs that use the best-effort scheduler, FRL is enabled. On vGPUs that use the fixed share or equal share scheduler, FRL is disabled.

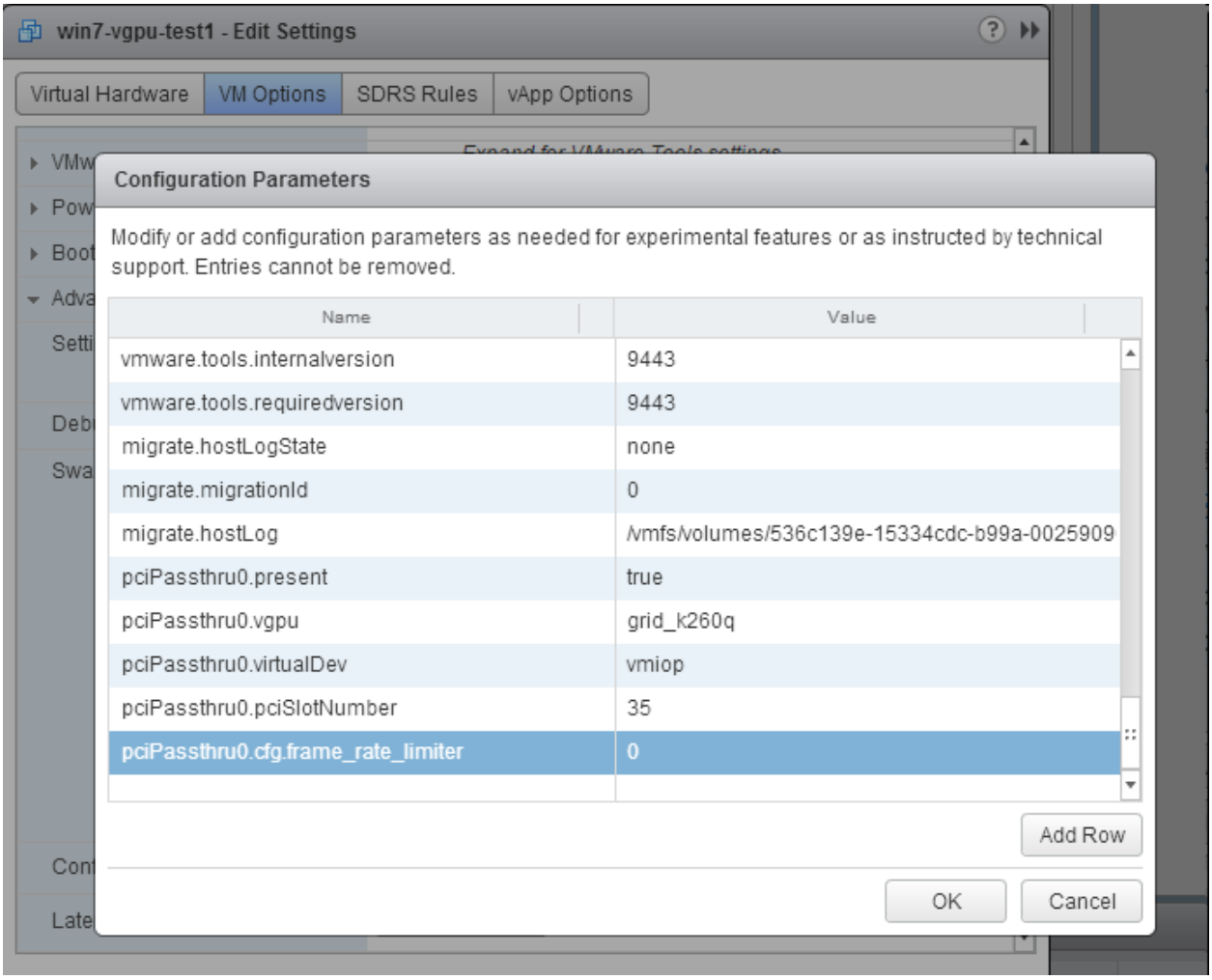
FRL is used to ensure balanced performance across multiple vGPUs that are resident on the same physical GPU. The FRL setting is designed to give a good interactive remote graphics experience but may reduce scores in benchmarks that depend on measuring frame rendering rates, as compared to the same benchmarks running on a passthrough GPU.

Resolution

FRL is controlled by an internal vGPU setting. On vGPUs that use the best-effort scheduler, NVIDIA does not validate vGPU with FRL disabled, but for validation of benchmark performance, FRL can be temporarily disabled by adding the configuration parameter `pciPassthru0.cfg.frame_rate_limiter` in the VM's advanced configuration options.

> **Note**
>
> This setting can only be changed when the VM is powered off.

1. Select **Edit Settings**.
2. In the Edit Settings window, select the **VM Options** tab.
3. From the **Advanced** drop-down list, select **Edit Configuration**.
4. In the Configuration Parameters dialog box, click **Add Row**.
5. In the **Name** field, type the parameter name `pciPassthru0.cfg.frame_rate_limiter`, in the **Value** field type `0`, and click **OK**.



With this setting in place, the VM's vGPU will run without any frame rate limit. The FRL can be reverted to its default setting by setting `pciPassthru0.cfg.frame_rate_limiter` to 1 or by removing the parameter from the advanced settings.

Resolution

FRL is controlled by an internal vGPU setting. On vGPUs that use the best-effort scheduler, NVIDIA does not validate vGPU with FRL disabled, but for validation of benchmark performance, FRL can be temporarily disabled by setting `frame_rate_limiter=0` in the vGPU configuration file.

```
# echo "frame_rate_limiter=0" > /sys/bus/mdev/devices/vgpu-id/nvidia/vgpu_
↪params
```

For example:

```
# echo "frame_rate_limiter=0" > /sys/bus/mdev/devices/aa618089-8b16-4d01-a136-
↪25a0f3c73123/nvidia/vgpu_params
```

The setting takes effect the next time any VM using the given vGPU type is started.

With this setting in place, the VM's vGPU will run without any frame rate limit.

The FRL can be reverted to its default setting as follows:

1. Clear all parameter settings in the vGPU configuration file.

```
# echo " " > /sys/bus/mdev/devices/vgpu-id/nvidia/vgpu_params
```

Note

You cannot clear specific parameter settings. If your vGPU configuration file contains other parameter settings that you want to keep, you must reinstate them in the next step.

2. Set `frame_rate_limiter=1` in the vGPU configuration file.

```
# echo "frame_rate_limiter=1" > /sys/bus/mdev/devices/vgpu-id/nvidia/vgpu_
↪params
```

If you need to reinstate other parameter settings, include them in the command to set `frame_rate_limiter=1`. For example:

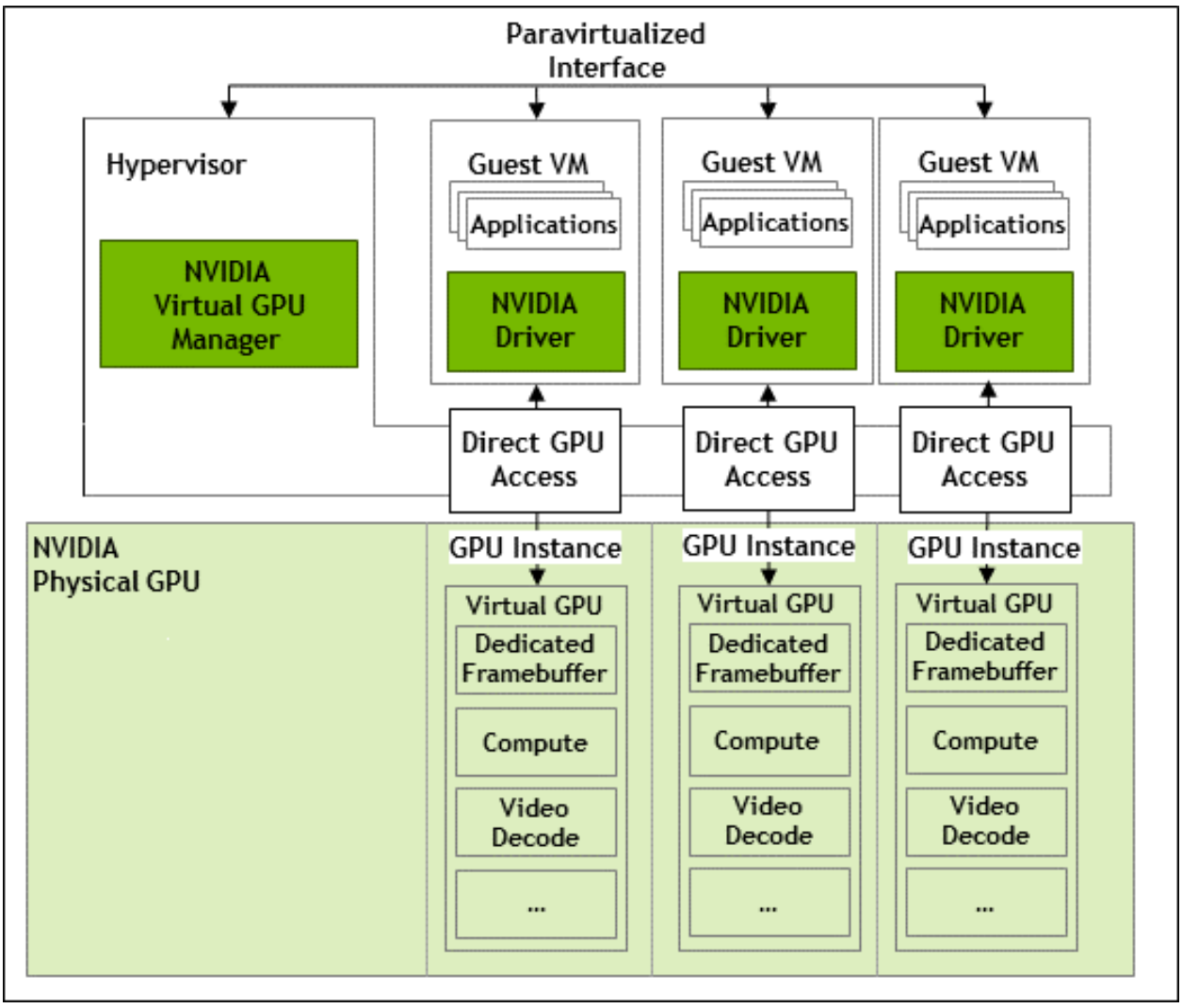
```
# echo "frame_rate_limiter=1 disable_vnc=1" > /sys/bus/mdev/devices/
↪aa618089-8b16-4d01-a136-25a0f3c73123/nvidia/vgpu_params
```

6.2. User Guide

6.2.1. MIG-Backed NVIDIA vGPU Internal Architecture

A MIG-backed vGPU is a vGPU that resides on a GPU instance in a MIG-capable physical GPU. Each MIG-backed vGPU resident on a GPU has exclusive access to the GPU instance's engines, including the compute and video decode engines.

In a MIG-backed vGPU, processes that run on the vGPU run in parallel with processes running on other vGPUs on the GPU. Process run on all vGPUs resident on a physical GPU simultaneously.

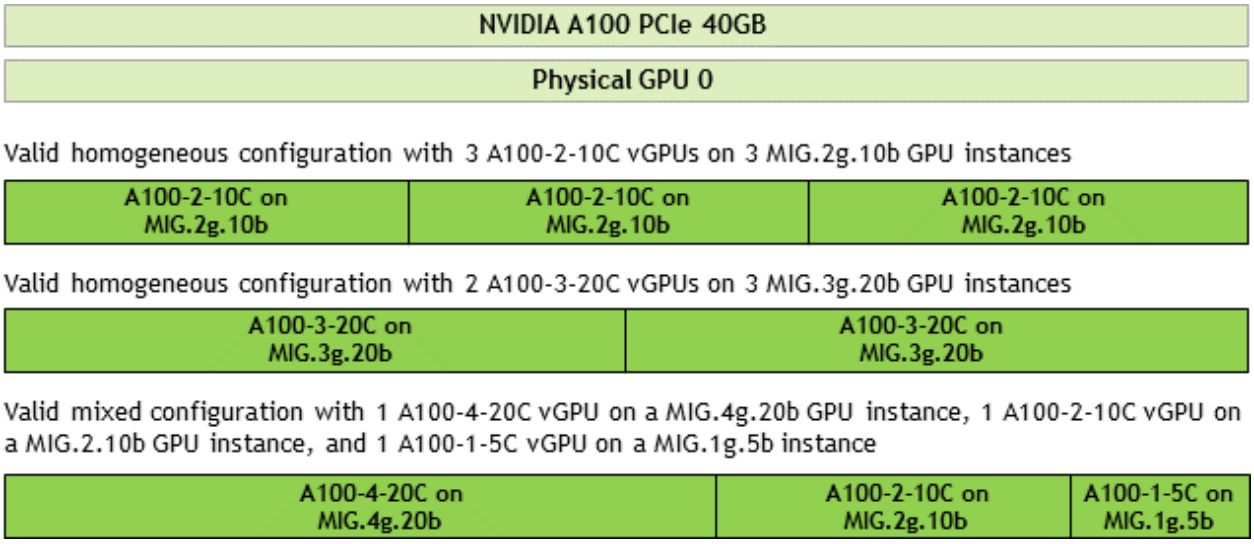


6.2.2. Valid MIG-Backed Virtual GPU Configurations on a Single GPU

This release of NVIDIA vGPU supports both homogeneous and mixed MIG-backed virtual GPUs based on the underlying GPU instance configuration.

For example, an NVIDIA A100 PCIe 40GB card has one physical GPU and can support several types of virtual GPU. Figure 4 shows the following examples of valid homogeneous and mixed MIG-backed virtual GPU configurations on NVIDIA A100 PCIe 40GB.

- A valid homogeneous configuration with 3 A100-2-10C vGPUs on 3 MIG.2g.10b GPU instances
- A valid homogeneous configuration with 2 A100-3-20C vGPUs on 3 MIG.3g.20b GPU instances
- A valid mixed configuration with 1 A100-4-20C vGPU on a MIG.4g.20b GPU instance, 1 A100-2-10C vGPU on a MIG.2.10b GPU instance, and 1 A100-1-5C vGPU on a MIG.1g.5b instance



6.2.3. Installing and Configuring the NVIDIA Virtual GPU Manager for Red Hat Enterprise Linux KVM

The following topics step you through the process of setting up a single Red Hat Enterprise Linux Kernel-based Virtual Machine (KVM) VM to use NVIDIA vGPU.

> **Caution**
>
> Output from the VM console is not available for VMs that are running vGPU. Make sure that you have installed an alternate means of accessing the VM (such as a VNC server) before you configure vGPU.

Follow this sequence of instructions:

1. Installing the Virtual GPU Manager Package for Red Hat Enterprise Linux KVM
2. Verifying the Installation of the NVIDIA AI Enterprise for Red Hat Enterprise Linux KVM
3. **MIG-backed vGPUs only**: Configuring a GPU for MIG-Backed vGPUs
4. **vGPUs that support SR-IOV only**: Preparing the Virtual Function for an NVIDIA vGPU that Supports SR-IOV on a Linux with KVM Hypervisor
5. **Optional**: Putting a GPU Into Mixed-Size Mode
6. Getting the BDF and Domain of a GPU on a Linux with KVM Hypervisor
7. Creating an NVIDIA vGPU on a Linux with KVM Hypervisor
8. Adding One or More vGPUs to a Linux with KVM Hypervisor VM
9. **Optional**: Placing a vGPU on a Physical GPU in Mixed-Size Mode
10. Setting vGPU Plugin Parameters on a Linux with KVM Hypervisor

After the process is complete, you can install the graphics driver for your guest OS and license any NVIDIA AI Enterprise-licensed products that you are using.

6.2.4. Installing and Configuring the NVIDIA Virtual GPU Manager for Ubuntu

Caution

Output from the VM console is not available for VMs that are running vGPU. Make sure that you have installed an alternate means of accessing the VM (such as a VNC server) before you configure vGPU.

Follow this sequence of instructions to set up a single Ubuntu VM to use NVIDIA vGPU.

1. Installing the NVIDIA Virtual GPU Manager for Ubuntu
2. **MIG-backed vGPUs only**: Configuring a GPU for MIG-Backed vGPUs
3. Getting the BDF and Domain of a GPU on a Linux with KVM Hypervisor
4. **vGPUs that support SR-IOV only**: Preparing the Virtual Function for an NVIDIA vGPU that Supports SR-IOV on a Linux with KVM Hypervisor
5. **Optional**: Putting a GPU Into Mixed-Size Mode
6. Creating an NVIDIA vGPU on a Linux with KVM Hypervisor
7. Adding One or More vGPUs to a Linux with KVM Hypervisor VM
8. **Optional**: Placing a vGPU on a Physical GPU in Mixed-Size Mode
9. Setting vGPU Plugin Parameters on a Linux with KVM Hypervisor

After the process is complete, you can install the graphics driver for your guest OS and license any NVIDIA AI Enterprise-licensed products that you are using.

6.2.5. Installing and Configuring the NVIDIA Virtual GPU Manager for VMware vSphere

You can use the NVIDIA Virtual GPU Manager for VMware vSphere to set up a VMware vSphere VM to use NVIDIA vGPU.

Note

Some servers, for example, the Dell R740, do not configure SR-IOV capability if the SR-IOV SBIOS setting is disabled on the server. If you are using the Tesla T4 GPU with VMware vSphere on such a server, you must ensure that the SR-IOV SBIOS setting is enabled on the server.

However, with any server hardware, do not enable SR-IOV in VMware vCenter Server for the Tesla T4 GPU. If SR-IOV is enabled in the VMware vCenter Server for T4, VMware vCenter Server lists the status of the GPU as needing a reboot. You can ignore this status message.

Requirements for Configuring NVIDIA vGPU in a DRS Cluster

You can configure a VM with NVIDIA vGPU on an ESXi host in a VMware Distributed Resource Scheduler (DRS) cluster. However, to ensure that the automation level of the cluster supports VMs configured with NVIDIA vGPU, you must set the automation level to **Partially Automated** or **Manual**.

For more information about these settings, see Edit Cluster Settings in the VMware documentation.

6.2.6. Configuring a GPU for MIG-Backed vGPUs

To support GPU instances with NVIDIA vGPU, a GPU must be configured with MIG mode enabled, and GPU instances must be created and configured on the physical GPU. Optionally, you can create compute instances within the GPU instances. If you don't create compute instances within the GPU instances, they can be added later for individual vGPUs from within the guest VMs.

Ensure that the following prerequisites are met:

- The NVIDIA Virtual GPU Manager is installed on the hypervisor host.
- You have root user privileges on your hypervisor host machine.
- You have determined which GPU instances correspond to the vGPU types of the MIG-backed vGPUs that you will create.
- The GPU is not being used by any other processes, such as CUDA applications, monitoring applications, or the `nvidia-smi` command.

To configure a GPU for MIG-backed vGPUs, follow these instructions:

1. Enable MIG mode for a GPU.

> **Note**
>
> For VMware vSphere, only enabling MIG mode is required because VMware vSphere creates the GPU instances and, after the VM is booted and the guest driver is installed, one compute instance is automatically created in the VM.

2. Create a GPU instance on a MIG-enabled GPU.
3. **Optional**: Create a compute instance in a GPU instance.

After configuring a GPU for MIG-backed vGPUs, create the vGPUs that you need and add them to their VMs.

6.2.6.1 Enabling MIG Mode for a GPU

Perform this task in your hypervisor command shell.

1. Open a command shell as the root user on your hypervisor host machine. On all supported hypervisors, you can use a secure shell (SSH) for this purpose. Individual hypervisors may provide additional means for logging in. For details, refer to the documentation for your hypervisor.
2. Determine whether MIG mode is enabled. Use the `nvidia-smi` command for this purpose. By default, MIG mode is disabled. This example shows that MIG mode is disabled on GPU 0.

Note

In the output from `nvidia-smi`, the NVIDIA A100 HGX 40GB GPU is referred to as A100-SXM4-40GB.

```
$ nvidia-smi -i 0
+-----------------------------------------------------------------------------
↪----+
| NVIDIA-SMI 550.54.16   Driver Version: 550.54.16    CUDA Version:  12.3
↪    |
|-------------------------------+----------------------+----------------------
↪----+
| 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  A100-SXM4-40GB      On   | 00000000:36:00.0 Off |
↪  0 |
| N/A   29C    P0    62W / 400W |      0MiB / 40537MiB |      6%
↪Default |
|                               |                      |
↪Disabled |
+-------------------------------+----------------------+----------------------
↪----+
```

3. If MIG mode is disabled, enable it.

```
$ nvidia-smi -i [gpu-ids] -mig 1
```

`gpu-ids` - A comma-separated list of GPU indexes, PCI bus IDs, or UUIDs that specifies the GPUs on which you want to enable MIG mode. If `gpu-ids` are omitted, MIG mode is enabled on all GPUs on the system.

This example enables MIG mode on GPU 0.

```
$ nvidia-smi -i 0 -mig 1
Enabled MIG Mode for GPU 00000000:36:00.0
All done.
```

Note

If the GPU is being used by another process, this command fails and displays a warning message that MIG mode for the GPU is in the pending enable state. In this situation, stop all processes that are using the GPU and retry the command.

4. VMware vSphere ESXi with GPUs based on the NVIDIA Ampere architecture only: Reboot the hypervisor host. If you are using any other hypervisor or GPUs that are based on the NVIDIA Hopper GPU architecture or a later architecture, omit this step.
5. Query the GPUs on which you enabled MIG mode to confirm that MIG mode is enabled. This example queries GPU 0 for the PCI bus ID and MIG mode in comma-separated values (CSV) format.

```
$ nvidia-smi -i 0 --query-gpu=pci.bus_id,mig.mode.current --format=csv
pci.bus_id, mig.mode.current
00000000:36:00.0, Enabled
```

6.2.6.2 Creating GPU Instances on a MIG-Enabled GPU

Note

If you are using VMware vSphere, omit this task. VMware vSphere creates the GPU instances automatically.

Perform this task in your hypervisor command shell.

1. If necessary, open a command shell as the root user on your hypervisor host machine.
2. List the GPU instance profiles that are available on your GPU. You will need to specify the profiles by their IDs, not their names when you create them.

```
$ nvidia-smi mig -lgip
+--------------------------------------------------------------------------
↪-+
| GPU instance profiles:
↪ |
| GPU   Name          ID    Instances   Memory     P2P    SM    DEC   ENC
↪ |
|                           Free/Total   GiB              CE    JPEG  OFA
↪ |
|=========================================================================|
|   0  MIG 1g.5gb     19     7/7         4.95       No     14     0     0
↪ |
|                                                          1     0     0
↪ |
+--------------------------------------------------------------------------
↪-+
|   0  MIG 2g.10gb    14     3/3         9.90       No     28     1     0
↪ |
|                                                          2     0     0
↪ |
+--------------------------------------------------------------------------
↪-+
|   0  MIG 3g.20gb     9     2/2         19.79      No     42     2     0
↪ |
|                                                          3     0     0
↪ |
+--------------------------------------------------------------------------
↪-+
|   0  MIG 4g.20gb     5     1/1         19.79      No     56     2     0
↪ |
|                                                          4     0     0
↪ |
+--------------------------------------------------------------------------
↪-+
|   0  MIG 7g.40gb     0     1/1         39.59      No     98     5     0
↪ |
```

(continues on next page)

(continued from previous page)

```
|                                                                         7      1      1
↪ |
+-------------------------------------------------------------------------------
↪-+
```

3. Create the GPU instances that correspond to the vGPU types of the MIG-backed vGPUs that you will create.

```
$ nvidia-smi mig -cgi gpu-instance-profile-ids
```

`gpu-instance-profile-ids` - A comma-separated list of GPU instance profile IDs that specifies the GPU instances that you want to create.

This example creates two GPU instances of type 2g.10gb, which has profile ID 14.

```
$ nvidia-smi mig -cgi 14,14
Successfully created GPU instance ID  5 on GPU  2 using profile
↪MIG 2g.10gb (ID 14)
Successfully created GPU instance ID  3 on GPU  2 using profile
↪MIG 2g.10gb (ID 14)
```

6.2.6.3 Optional: Creating Compute Instances in a GPU instance

Creating compute instances within GPU instances is optional. If you don't create compute instances within the GPU instances, they can be added later for individual vGPUs from within the guest VMs.

> **Note**
>
> If you are using VMware vSphere, omit this task. After the VM is booted and the guest driver is installed, one compute instance is automatically created in the VM.

Perform this task in your hypervisor command shell.

1. If necessary, open a command shell as the root user on your hypervisor host machine.
2. List the available GPU instances.

```
$ nvidia-smi mig -lgi
+-------------------------------------------------------+
| GPU instances:                                        |
| GPU   Name          Profile  Instance   Placement  |
|                       ID       ID       Start:Size |
|=======================================================|
|   2  MIG 2g.10gb      14        3          0:2     |
+-------------------------------------------------------+
|   2  MIG 2g.10gb      14        5          4:2     |
+-------------------------------------------------------+
```

3. Create the compute instances that you need within each GPU instance

```
$ nvidia-smi mig -cci -gi gpu-instance-ids
```

`gpu-instance-ids` - A comma-separated list of GPU instance IDs that specifies the GPU instances within which you want to create the compute instances.

Caution

To avoid an inconsistent state between a guest VM and the hypervisor host, do not create compute instances from the hypervisor on a GPU instance on which an active guest VM is running. Instead, create the compute instances from within the guest VM as explained in Modifying a MIG-Backed vGPU's Configuration.

This example creates a compute instance on each of GPU instances 3 and 5.

```
$ nvidia-smi mig -cci -gi 3,5
Successfully created compute instance on GPU  0 GPU instance ID  1 using
↪profile ID  2
Successfully created compute instance on GPU  0 GPU instance ID  2 using
↪profile ID  2
```

4. Verify that the compute instances were created within each GPU instance.

```
$ nvidia-smi
+-----------------------------------------------------------------------------
↪----+
| MIG devices:
↪    |
+------------------+----------------------+-----------+----------------------
↪----+
| GPU  GI  CI  MIG |         Memory-Usage |        Vol|         Shared
↪    |
|      ID  ID  Dev |           BAR1-Usage | SM     Unc| CE  ENC  DEC  OFA
↪ JPG|
|                  |                      |        ECC|
↪    |
|==================+======================+===========+=======================|
|  2    3   0   0  |      0MiB /  9984MiB | 28      0 |  2   0    1    0
↪  0 |
|                  |      0MiB / 16383MiB |           |
↪    |
+------------------+----------------------+-----------+----------------------
↪----+
|  2    5   0   1  |      0MiB /  9984MiB | 28      0 |  2   0    1    0
↪  0 |
|                  |      0MiB / 16383MiB |           |
↪    |
+------------------+----------------------+-----------+----------------------
↪----+

+-----------------------------------------------------------------------------
↪----+
| Processes:
↪    |
|  GPU   GI   CI        PID   Type   Process name                  GPU
↪Memory |
|        ID   ID                                                   Usage
↪    |
|=============================================================================|
```

Note

Additional compute instances that have been created in a VM are destroyed when the VM is shut down or rebooted. After the shutdown or reboot, only one compute instance remains in the VM. This compute instance is created automatically after the NVIDIA AI Enterprise graphics driver is installed.

6.2.7. Disabling MIG Mode for One or More GPUs

If a GPU that you want to use for time-sliced vGPUs or GPU passthrough has previously been configured for MIG-backed vGPUs, disable MIG mode on the GPU.

Ensure that the following prerequisites are met:

- The NVIDIA Virtual GPU Manager is installed on the hypervisor host.
- You have root user privileges on your hypervisor host machine.
- The GPU is not being used by any other processes, such as CUDA applications, monitoring applications, or the `nvidia-smi` command.

Perform this task in your hypervisor command shell.

1. Open a command shell as the root user on your hypervisor host machine. On all supported hypervisors, you can use a secure shell (SSH) for this purpose. Individual hypervisors may provide additional means for logging in. For details, refer to the documentation for your hypervisor.
2. Determine whether MIG mode is disabled. Use the `nvidia-smi` command for this purpose. By default, MIG mode is disabled but might have previously been enabled. This example shows that MIG mode is enabled on GPU 0.

Note

In the output from `nvidia-smi`, the NVIDIA A100 HGX 40GB GPU is referred to as A100-SXM4-40GB.

```
$ nvidia-smi -i 0
+-----------------------------------------------------------------------------
↪----+
| NVIDIA-SMI 550.54.16    Driver Version: 550.54.16   CUDA Version:  12.3
↪    |
|-------------------------------+----------------------+------------------
↪----+
| 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  A100-SXM4-40GB      Off  | 00000000:36:00.0 Off |
↪  0 |
| N/A   29C    P0    62W / 400W |      0MiB / 40537MiB |      6%
↪Default |
```

(continues on next page)

(continued from previous page)

```
|                                 |                      |
↪Enabled |
+---------------------------------+----------------------+-----------------
↪----+
```

3. If MIG mode is enabled, disable it.

```
$ nvidia-smi -i [gpu-ids] -mig 0
```

`gpu-ids` - A comma-separated list of GPU indexes, PCI bus IDs, or UUIDs that specifies the GPUs on which you want to disable MIG mode. If `gpu-ids` are omitted, MIG mode is disabled on all GPUs on the system.

This example disables MIG mode on GPU 0.

```
$ sudo nvidia-smi -i 0 -mig 0
Disabled MIG Mode for GPU 00000000:36:00.0
All done.
```

4. Confirm that MIG mode was disabled. Use the `nvidia-smi` command for this purpose. This example shows that MIG mode is disabled on GPU 0.

```
$ nvidia-smi -i 0
+-----------------------------------------------------------------------------
↪----+
| NVIDIA-SMI 550.54.16    Driver Version: 550.54.16   CUDA Version:  12.3
↪    |
|---------------------------------+----------------------+-----------------
↪----+
| 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  A100-SXM4-40GB      Off  | 00000000:36:00.0 Off |
↪  0 |
| N/A   29C    P0    62W / 400W |      0MiB / 40537MiB |      6%
↪Default |
|                                 |                      |
↪Disabled |
+---------------------------------+----------------------+-----------------
↪----+
```

6.2.8. Installing NVIDIA AI Enterprise Software Components by Using Kubernetes

Perform this task if you are using one of the following combinations of guest operating system and container platform:

- Ubuntu with Kubernetes

Ensure that the following prerequisites are met:

1. If you are using Kubernetes, ensure that:
    1. Kubernetes is installed in the VM.
    2. NVIDIA vGPU Manager is installed.
    3. NVIDIA vGPU License Server with licenses is installed.
2. Helm is installed.
3. You have generated your NGC API key for accessing the NVIDIA Enterprise Collection at the URL provided to you by NVIDIA.

6.2.8.1 Transforming Container Images for AI and Data Science Applications and Frameworks into Kubernetes Pods

The AI and data science applications and frameworks are distributed as NGC container images through the NGC private registry. If you are using Kubernetes or Red Hat OpenShift, you must transform each image that you want to use into a Kubernetes pod. Each container image contains the entire user-space software stack that is required to run the application or framework, namely, the CUDA libraries, cuDNN, any required Magnum IO components, TensorRT, and the framework.

6.2.9. Installing NVIDIA AI Enterprise Software, Applications, and Deep Learning Framework Components by Using Docker

NVIDIA AI Enterprise software components in the infrastructure optimization and cloud-native deployment layers are distributed through the NVIDIA AI Enterprise Infra Release 5 collection on NVIDIA NGC. Applications and deep learning framework components for NVIDIA AI Enterprise are distributed exclusively through the NGC Public Catalog.

The container image for each application or framework contains the entire user-space software stack that is required to run the application or framework, namely, the CUDA libraries, cuDNN, any required Magnum IO components, TensorRT, and the framework.

Ensure that you have completed the following tasks in the NGC Private Registry User Guide:

- Generating Your NGC API Key
- Accessing the NGC Container Registry

Perform this task from the VM.

Obtain the Docker `pull` command for downloading each of the following applications and deep learning framework components from the listing for the application or component in the NGC Public Catalog.

- Applications:
    - NVIDIA Clara Parabricks
    - NVIDIA DeepStream
    - NVIDIA Riva
    - MONAI - Medical Open Network for Artificial Intelligence
    - RAPIDS

- RAPIDS Accelerator for Apache Spark
- TAO

- Deep learning framework components:
  - NVIDIA TensorRT
  - NVIDIA Triton Inference Server
  - PyTorch
  - TensorFlow 2

Obtain the command for downloading each of the following NVIDIA AI Enterprise software components from the listing for the component in the NVIDIA AI Enterprise Infra Release 5 collection on NVIDIA NGC.

- GPU Operator
- Network Operator
- NVIDIA Base Command Manager Essentials
- vGPU Guest Driver, Ubuntu 22.04

6.2.10. Installing NVIDIA GPU Operator by Using a Bash Shell Script

A bash shell script for installing an NVIDIA GPU Operator with the NVIDIA vGPU guest driver is available for download from NVIDIA NGC.

Before performing this task, ensure that the following prerequisites are met:

- A client configuration token has been generated for the client on which the script will install the vGPU guest driver.
- The API key of the NVIDIA NGC user to be used for creating the image pull secret has been generated.
- The following environment variables are set:

  `NGC_API_KEY` - The API key of the NVIDIA NGC user to be used for creating the image pull secret. For example:

  ```
  export NGC_API_KEY=
  ↪"RLh1zerCiG4wPGWWt4Tyj2VMyd7T8MnDyCT95pygP5VJFv8en4eLvdXVZzjm"
  ```

  `NGC_USER_EMAIL` - The email address of the NVIDIA NGC user to be used for creating the image pull secret. For example:

  ```
  export NGC_USER_EMAIL="ada.lovelace@example.com"
  ```

1. Download the NVIDIA GPU Operator - Deploy Installer Script from NVIDIA NGC.
2. Ensure that the file access modes of the script allow the owner to execute the script.
   1. Change to the directory that contains the script.

      ```
      # cd script-directory
      ```

`script-directory` - The directory to which you downloaded the script in the previous step.

2. Determine the current file access modes of the script.

```
# ls -l gpu-operator-nvaie.sh
```

3. If necessary, grant execute permission to the owner of the script.

```
# chmod u+x gpu-operator-nvaie.sh
```

3. Copy the client configuration token to the directory that contains the script.
4. Rename the client configuration token to `client_configuration_token.tok`. The client configuration token is generated with a file name that includes a time stamp, namely: `client_configuration_token_mm-dd-yyyy-hh-mm-ss.tok`.
5. From the directory that contains the script, start the script, specifying the option to install the NVIDIA vGPU guest driver.

```
# bash gpu-operator-nvaie.sh install
```

6.2.11. Modifying a MIG-Backed vGPU's Configuration

If compute instances weren't created within the GPU instances when the GPU was configured for MIG-backed vGPUs, you can add the compute instances for an individual vGPU from within the guest VM. If you want to replace the compute instances that were created when the GPU was configured for MIG-backed vGPUs, you can delete them before adding the compute instances from within the guest VM.

Ensure that the following prerequisites are met:

- You have root user privileges in the guest VM.
- The GPU instance is not being used by any other processes, such as CUDA applications, monitoring applications, or the `nvidia-smi` command.

Perform this task in a guest VM command shell.

1. Open a command shell as the root user in the guest VM. On all supported hypervisors, you can use a secure shell (SSH) for this purpose. Individual hypervisors may provide additional means for logging in. For details, refer to the documentation for your hypervisor.
2. List the available GPU instances.

```
$ nvidia-smi mig -lgi
+-------------------------------------------------------+
| GPU instances:                                        |
| GPU   Name          Profile  Instance   Placement  |
|                       ID       ID       Start:Size |
|=======================================================|
|   0  MIG 2g.10gb       0        0          0:8     |
+-------------------------------------------------------+
```

3. **Optional**: If compute instances were created when the GPU was configured for MIG-backed vGPUs that you no longer require, delete them.

```
$ nvidia-smi mig -dci -ci compute-instance-id -gi gpu-instance-id
```

`compute-instance-id` - The ID of the compute instance that you want to delete.

`gpu-instance-id` - The ID of the GPU instance from which you want to delete the compute instance.

> **Note**
>
> If the GPU instance is being used by another process, this command fails. In this situation, stop all processes that are using the GPU instance and retry the command.

This example deletes compute instance 0 from GPU instance 0 on GPU 0.

```
$ nvidia-smi mig -dci -ci 0 -gi 0
Successfully destroyed compute instance ID  0 from GPU  0 GPU instance ID
↪ 0
```

4. List the compute instance profiles that are available for your GPU instance.

```
$ nvidia-smi mig -lcip
```

This example shows that one `MIG 2g.10gb` compute instance or two `MIG 1c.2g.10gb` compute instances can be created within the GPU instance.

```
$ nvidia-smi mig -lcip
+-----------------------------------------------------------------------------
↪------+
| Compute instance profiles:
↪       |
| GPU    GPU      Name          Profile  Instances   Exclusive
↪Shared      |
|      Instance                   ID     Free/Total     SM      DEC   ENC
↪  OFA |
|        ID                                                     CE
↪JPEG      |
|=============================================================================|
|   0     0       MIG 1c.2g.10gb   0      2/2           14       1     0
↪   0  |
|                                                               2     0
↪      |
+-----------------------------------------------------------------------------
↪------+
|   0     0       MIG 2g.10gb      1*     1/1           28       1     0
↪   0  |
|                                                               2     0
↪      |
+-----------------------------------------------------------------------------
↪------+
```

5. Create the compute instances that you need within the available GPU instance. Create each compute instance individually by running the following command.

```
$ nvidia-smi mig -cci compute-instance-profile-id -gi gpu-instance-id
```

`compute-instance-profile-id` - The compute instance profile ID that specifies the compute instance.

`gpu-instance-id` - The GPU instance ID that specifies the GPU instance within which you want to create the compute instance.

> **Note**
>
> If the GPU instance is being used by another process, this command fails. In this situation, stop all processes that are using the GPU and retry the command.

This example creates a `MIG 2g.10gb` compute instance on GPU instance 0.

```
$ nvidia-smi mig -cci 1 -gi 0
Successfully created compute instance ID  0 on GPU  0 GPU instance ID  0
↪using profile MIG 2g.10gb (ID  1)
```

This example creates two `MIG 1c.2g.10gb` compute instances on GPU instance 0 by running the same command twice.

```
$ nvidia-smi mig -cci 0 -gi 0
Successfully created compute instance ID  0 on GPU  0 GPU instance ID  0
↪using profile MIG 1c.2g.10gb (ID  0)
$ nvidia-smi mig -cci 0 -gi 0
Successfully created compute instance ID  1 on GPU  0 GPU instance ID  0
↪using profile MIG 1c.2g.10gb (ID  0)
```

6. Verify that the compute instances were created within the GPU instance. Use the `nvidia-smi` command for this purpose. This example confirms that a `MIG 2g.10gb` compute instance was created on GPU instance 0.

```
nvidia-smi
Mon Mar 25 19:01:24 2024
+-----------------------------------------------------------------------------
↪----+
| NVIDIA-SMI 550.54.16    Driver Version: 550.54.16   CUDA Version:  12.3
↪    |
|-------------------------------+----------------------+------------------
↪----+
| 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  GRID A100X-2-10C     On  | 00000000:00:08.0 Off |
↪ On |
| N/A   N/A    P0    N/A /  N/A |   1058MiB / 10235MiB |     N/A
↪Default |
|                               |                      |
↪Enabled |
+-------------------------------+----------------------+------------------
↪----+

+-----------------------------------------------------------------------------
↪----+
| MIG devices:
↪    |
+------------------+----------------------+-----------+------------------
↪----+
| GPU  GI  CI  MIG |         Memory-Usage |        Vol|         Shared
```

(continues on next page)

(continued from previous page)

```
↪    |
|      ID  ID  Dev |           BAR1-Usage | SM     Unc| CE  ENC  DEC  OFA
↪ JPG|
|                  |                      |        ECC|
↪    |
|==================+======================+===========+=======================|
|  0    0   0   0  |   1058MiB / 10235MiB | 28      0 |  2   0    1    0
↪  0 |
|                  |      0MiB /  4096MiB |           |
↪    |
+------------------+----------------------+-----------+-----------------------
↪----+

+-----------------------------------------------------------------------------
↪----+
| Processes:
↪    |
|  GPU   GI   CI        PID   Type   Process name                  GPU
↪Memory |
|        ID   ID                                                   Usage
↪    |
|=============================================================================|
|  No running processes found
↪    |
+-----------------------------------------------------------------------------
↪----+
```

This example confirms that two `MIG 1c.2g.10gb` compute instances were created on GPU instance 0.

```
$ nvidia-smi
Mon Mar 25 19:01:24 2024
+-----------------------------------------------------------------------------
↪----+
| NVIDIA-SMI 550.54.16    Driver Version: 550.54.16   CUDA Version:  12.3
↪    |
|-------------------------------+----------------------+-----------------
↪----+
| 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  GRID A100X-2-10C     On  | 00000000:00:08.0 Off |
↪ On |
| N/A   N/A    P0    N/A /  N/A |   1058MiB / 10235MiB |     N/A
↪Default |
|                               |                      |
↪Enabled |
+-------------------------------+----------------------+-----------------
↪----+

+-----------------------------------------------------------------------------
↪----+
```

(continues on next page)

(continued from previous page)

```
| MIG devices:
↪    |
+------------------+----------------------+-----------+------------------
↪----+
| GPU  GI  CI  MIG |         Memory-Usage |        Vol|         Shared
↪    |
|      ID  ID  Dev |           BAR1-Usage | SM     Unc| CE  ENC  DEC  OFA
↪ JPG|
|                  |                      |        ECC|
↪    |
|==================+======================+===========+======================|
|  0    0   0   0  |   1058MiB / 10235MiB | 14      0 |  2   0    1    0
↪  0 |
|                  |      0MiB /  4096MiB |           |
↪    |
+------------------+                      +-----------+------------------
↪----+
|  0    0   1   1  |                      | 14      0 |  2   0    1    0
↪  0 |
|                  |                      |           |
↪    |
+------------------+----------------------+-----------+------------------
↪----+

+-----------------------------------------------------------------------------
↪----+
| Processes:
↪    |
|  GPU   GI   CI        PID   Type   Process name                  GPU
↪Memory |
|        ID   ID                                                   Usage
↪    |
|=============================================================================|
|  No running processes found
↪    |
+-----------------------------------------------------------------------------
↪----+
```

6.2.12. Monitoring MIG-backed vGPU Activity

Note

MIG-backed vGPU activity cannot be monitored on GPUs based on the NVIDIA Ampere GPU architecture because the required hardware feature is not present on these GPUs.

To monitor MIG-backed vGPU activity across multiple vGPUs, run `nvidia-smi vgpu` with the `--gpm-metrics ID-list` option.

`ID-list` - A comma-separated list of integer IDs that specify the statistics to monitor as shown in the following table. The table also shows the name of the column in the command output under which the statistic is reported.

Table 20: Monitoring MIG-backed vGPU Activity

Statistic	ID	Column
Graphics activity	1	`gract`
Streaming multiprocessor (SM) activity	2	`smutil`
SM occupancy	3	`smocc`
Integer activity	4	`intutil`
Tensor activity	5	`mmaact`
Double-precision fused multiply-add (DFMA) tensor activity	6	`dfmat`
Half matrix multiplication and accumulation (HMMA) tensor activity	7	`hmmat`
Integer matrix multiplication and accumulation (IMMA) tensor activity	9	`immat`
Dynamic random-access memory (DRAM) activity	10	`dram`
Double-precision 64-bit floating-point (FP64) activity	11	`fp64`
Single-precision 32-bit floating-point (FP32) activity	12	`fp32`
Half-precision 16-bit FP16 activity	13	`fp16`

Each reported percentage is the percentage of the physical GPU's capacity that a vGPU is using. For example, a vGPU that uses 20% of the GPU's DRAM capacity will report 20%.

For each vGPU, the specified statistics are reported once every second.

To modify the reporting frequency, use the `-l` or `--loop` option.

To limit monitoring to a subset of the GPUs on the platform, use the `-i` or `--id` option to select one or more GPUs.

The following example reports graphics activity, SM activity, SM occupancy, and integer activity for one vGPU VM that is powered on and within which one application is running.

```
[root@vgpu ~]# nvidia-smi vgpu --gpm-metrics 1,2,3,4
# gpu        vgpu    mig_id       gi_id        ci_id       gract    smutil
↪ smocc    intutil
# Idx          Id       Idx         Idx          Idx           %         %
↪     %          %
    0  3251634249          0           2            0           -         -
↪         -          -
    0  3251634249          0           2            0          99        97
↪        26         13
    0  3251634249          0           2            0          99        96
↪        23         13
    0  3251634249          0           2            0          99        97
↪        27         13
```

When no vGPUs are active on the hypervisor host, no activity is reported.

```
[root@vgpu ~]# nvidia-smi vgpu --gpm-metrics 1,2,3,4
# gpu        vgpu    mig_id       gi_id        ci_id       gract    smutil
↪ smocc    intutil
# Idx          Id       Idx         Idx          Idx           %         %
```

(continues on next page)

(continued from previous page)

```
↪      %           %
    0              -         -         -         -         -         -
↪           -            -
    0              -         -         -         -         -         -
↪           -            -
    0              -         -         -         -         -         -
↪           -            -
```

6.2.13. Virtual GPU Types for Supported GPUs

6.2.13.1 NVIDIA A800 PCIe 80GB, NVIDIA A800 PCIe 80GB Liquid Cooled, and NVIDIA AX800 Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

The virtual GPU types for the NVIDIA A800 PCIe 80GB, NVIDIA A800 PCIe 80GB liquid-cooled, and NVIDIA AX800 GPUs are identical.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A800 PCIe 80GB, NVIDIA A800 PCIe 80GB Liquid-Cooled, and NVIDIA AX800

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 21: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A800 PCIe 80GB, NVIDIA A800 PCIe 80GB Liquid-Cooled, and NVIDIA AX800

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
A800D-7-80C	81920	1	7	7	MIG 7g.80gb
A800D-4-40C	40960	1	4	4	MIG 4g.40gb
A800D-3-40C	40960	2	3	3	MIG 3g.40gb
A800D-2-20C	20480	3	2	2	MIG 2g.20gb
A800D-1-20C[3]	20480	4	1	1	MIG 1g.20gb
A800D-1-10C	10240	7	1	1	MIG 1g.10gb
A800D-1-10CME[?]	10240	1	1	1	MIG 1g.10gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A800 PCIe 80GB, NVIDIA A800 PCIe 80GB Liquid-Cooled, and NVIDIA AX800

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

[3] These vGPU types are supported on ESXi starting with vSphere 8.0 update 3.

Table 22: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A800 PCIe 80GB, NVIDIA A800 PCIe 80GB Liquid-Cooled, and NVIDIA AX800

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[4]	Virtual Displays per vGPU
A800D-80C	81920	1	1	3840x2400	1
A800D-40C	40960	2	2	3840x2400	1
A800D-20C	20480	4	4	3840x2400	1
A800D-16C	16384	5	4	3840x2400	1
A800D-10C	10240	8	8	3840x2400	1
A800D-8C	8192	10	8	3840x2400	1
A800D-4C	4096	20	16	3840x2400	1

6.2.13.2 NVIDIA A800 PCIe 40GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A800 PCIe 40GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

[4] NVIDIA vGPU (C-Series) is optimized for compute-intensive workloads. As a result, they support only a single display head and do not provide Quadro graphics acceleration.

Table 23: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A800 PCIe 40GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
A800-7-40C	40960	1	7	7	MIG 7g.40gb
A800-4-20C	20480	1	4	4	MIG 4g.20gb
A800-3-20C	20480	2	3	3	MIG 3g.20gb
A800-2-10C	10240	3	2	2	MIG 2g.10gb
A800-1-10C?	10240	4	1	1	MIG 1g.10gb
A800-1-5C	5120	7	1	1	MIG 1g.5gb
A800-1-5CME?	5120	1	1	1	MIG 1g.5gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A800 PCIe 40GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 24: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A800 PCIe 40GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
A800-40C	40960	1	1	3840x2400	1
A800-20C	20480	2	2	3840x2400	1
A800-10C	10240	4	4	3840x2400	1
A800-8C	8192	5	4	3840x2400	1
A800-5C	5120	8	8	3840x2400	1
A800-4C	4096	10	8	3840x2400	1

6.2.13.3 NVIDIA A800 HGX Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A800 HGX 80GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 25: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A800 HGX 80GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
A800DX-7-80C	81920	1	7	7	MIG 7g.80gb
A800DX-4-40C	40960	1	4	4	MIG 4g.40gb
A800DX-3-40C	40960	2	3	3	MIG 3g.40gb
A800DX-2-20C	20480	3	2	2	MIG 2g.20gb
A800DX-1-20C[?]	20480	4	1	1	MIG 1g.20gb
A800DX-1-10C	10240	7	1	1	MIG 1g.10gb
A800DX-1-10CME[?]	10240	1	1	1	MIG 1g.10gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A800 HGX 80GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 26: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A800 HGX 80GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[7]	Virtual Displays per vGPU
A800DX-80C	81920	1	1	3840x2400	1
A800DX-40C	40960	2	2	3840x2400	1
A800DX-20C	20480	4	4	3840x2400	1
A800DX-16C	16384	5	4	3840x2400	1
A800DX-10C	10240	8	8	3840x2400	1
A800DX-8C	8192	10	8	3840x2400	1
A800DX-4C	4096	20	16	3840x2400	1

6.2.13.4 NVIDIA A100 PCIe 40GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A100 PCIe 40GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 27: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A100 PCIe 40GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
A100-7-40C	40960	1	7	7	MIG 7g.40gb
A100-4-20C	20480	1	4	4	MIG 4g.20gb
A100-3-20C	20480	2	3	3	MIG 3g.20gb
A100-2-10C	10240	3	2	2	MIG 2g.10gb
A100-1-10C?	10240	4	1	1	MIG 1g.10gb
A100-1-5C	5120	7	1	1	MIG 1g.5gb
A100-1-5CME?	5120	1	1	1	MIG 1g.5gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A100 PCIe 40GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 28: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A100 PCIe 40GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[7]	Virtual Displays per vGPU
A100-40C	40960	1	1	3840x2400	1
A100-20C	20480	2	2	3840x2400	1
A100-10C	10240	4	4	3840x2400	1
A100-8C	8192	5	4	3840x2400	1
A100-5C	5120	8	8	3840x2400	1
A100-4C	4096	10	8	3840x2400	1

6.2.13.5 NVIDIA A100 HGX 40GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A100 HGX 40GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 29: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A100 HGX 40GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
A100X-7-40C	40960	1	7	7	MIG 7g.40gb
A100X-4-20C	20480	1	4	4	MIG 4g.20gb
A100X-3-20C	20480	2	3	3	MIG 3g.20gb
A100X-2-10C	10240	3	2	2	MIG 2g.10gb
A100X-1-10C?	10240	4	1	1	MIG 1g.10gb
A100X-1-5C	5120	7	1	1	MIG 1g.5gb
A100X-1-5CME?	5120	1	1	1	MIG 1g.5gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A100 HGX 40GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 30: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A100 HGX 40GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[7]	Virtual Displays per vGPU
A100X-40C	40960	1	1	3840x2400	1
A100X-20C	20480	2	2	3840x2400	1
A100X-10C	10240	4	4	3840x2400	1
A100X-8C	8192	5	4	3840x2400	1
A100X-5C	5120	8	8	3840x2400	1
A100X-4C	4096	10	8	3840x2400	1

6.2.13.6 NVIDIA A100 PCIe 80GB, NVIDIA A100 PCIe 80GB Liquid-Cooled and NVIDIA A100X Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

The virtual GPU types for the NVIDIA A100 PCIe 80GB, NVIDIA A100 PCIe 80GB liquid-cooled and NVIDIA A100X GPUs are identical.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A100 PCIe 80GB, NVIDIA A100 PCIe 80GB Liquid-Cooled, and NVIDIA A100X

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 31: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A100 PCIe 80GB, NVIDIA A100 PCIe 80GB Liquid-Cooled, and NVIDIA A100X

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
A100D-7-80C	81920	1	7	7	MIG 7g.80gb
A100D-4-40C	40960	1	4	4	MIG 4g.40gb
A100D-3-40C	40960	2	3	3	MIG 3g.40gb
A100D-2-20C	20480	3	2	2	MIG 2g.20gb
A100D-1-20C[?]	20480	4	1	1	MIG 1g.20gb
A100D-1-10C	10240	7	1	1	MIG 1g.10gb
A100D-1-10CME[?]	10240	1	1	1	MIG 1g.10gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A100 PCIe 80GB, NVIDIA A100 PCIe 80GB Liquid-Cooled, and NVIDIA A100X

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 32: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A100 PCIe 80GB, NVIDIA A100 PCIe 80GB Liquid-Cooled, and NVIDIA A100X

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
A100D-80C	81920	1	1	3840x2400	1
A100D-40C	40960	2	2	3840x2400	1
A100D-20C	20480	4	4	3840x2400	1
A100D-16C	16384	5	4	3840x2400	1
A100D-10C	10240	8	8	3840x2400	1
A100D-8C	8192	10	8	3840x2400	1
A100D-4C	4096	20	16	3840x2400	1

6.2.13.7 NVIDIA A100 HGX 80GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A100 HGX 80GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 33: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A100 HGX 80GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
A100DX-7-80C	81920	1	7	7	MIG 7g.80gb
A100DX-4-40C	40960	1	4	4	MIG 4g.40gb
A100DX-3-40C	40960	2	3	3	MIG 3g.40gb
A100DX-2-20C	20480	3	2	2	MIG 2g.20gb
A100DX-1-20C[?]	20480	4	1	1	MIG 1g.20gb
A100DX-1-10C	10240	7	1	1	MIG 1g.10gb
A100DX-1-10CME[?]	10240	1	1	1	MIG 1g.10gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A100 HGX 80GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 34: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A100 HGX 80GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[7]	Virtual Displays per vGPU
A100DX-80C	81920	1	1	3840x2400	1
A100DX-40C	40960	2	2	3840x2400	1
A100DX-20C	20480	4	4	3840x2400	1
A100DX-16C	16384	5	4	3840x2400	1
A100DX-10C	10240	8	8	3840x2400	1
A100DX-8C	8192	10	8	3840x2400	1
A100DX-4C	4096	20	16	3840x2400	1

6.2.13.8 NVIDIA A40 Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

NVIDIA vGPU (C-Series) for NVIDIA A40

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 35: NVIDIA vGPU (C-Series) for NVIDIA A40

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[2]	Virtual Displays per vGPU
A40-48C	49152	1	1	3840x2400	1
A40-24C	24576	2	2	3840x2400	1
A40-16C	16384	3	2	3840x2400	1
A40-12C	12288	4	4	3840x2400	1
A40-8C	8192	6	4	3840x2400	1
A40-6C	6144	8	8	3840x2400	1
A40-4C	4096	12[5]	8	3840x2400	1

6.2.13.9 NVIDIA A30, NVIDIA A30X, and NVIDIA A30 Liquid-Cooled Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

The virtual GPU types for the NVIDIA A30, NVIDIA A30X, and NVIDIA A30 Liquid Cooled GPUs are identical.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A30, NVIDIA A30X, and NVIDIA A30 Liquid-Cooled

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

[5] The maximum number of NVIDIA Virtual Compute Server vGPUs is limited to 12 vGPUs per physical GPU, irrespective of the available hardware resources of the physical GPU.

Table 36: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA A30, NVIDIA A30X, and NVIDIA A30 Liquid-Cooled

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
A30-4-24C	24576	1	4	4	MIG 4g.24gb
A30-2-12C	12288	2	2	2	MIG 2g.12gb
A30-2-12CME[?]	12288	1	2	2	MIG 2g.12gb+me
A30-1-6C	6144	4	1	1	MIG 1g.6gb
A30-1-6CME[?]	6144	1	1	1	MIG 1g.6gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A30, NVIDIA A30X, and NVIDIA A30 Liquid-Cooled

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 37: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA A30, NVIDIA A30X, and NVIDIA A30 Liquid-Cooled

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
A30-24C	24576	1	1	3840x2400	1
A30-12C	12288	2	2	3840x2400	1
A30-8C	8192	3	2	3840x2400	1
A30-6C	6144	4	4	3840x2400	1
A30-4C	4096	6	4	3840x2400	1

6.2.13.10 NVIDIA A16 Virtual GPU Types

Physical GPUs per board: 4

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

NVIDIA vGPU (C-Series) for NVIDIA A16

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 38: NVIDIA vGPU (C-Series) for NVIDIA A16

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
A16-16C	16384	1	1	3840x2400	1
A16-8C	8192	2	2	3840x2400	1
A16-4C	4096	4	4	3840x2400	1

6.2.13.11 NVIDIA A10 Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

NVIDIA vGPU (C-Series) for NVIDIA A10

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 39: NVIDIA vGPU (C-Series) for NVIDIA A10

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[2]	Virtual Displays per vGPU
A10-24C	24576	1	1	3840x2400	1
A10-12C	12288	2	2	3840x2400	1
A10-8C	8192	3	2	3840x2400	1
A10-6C	6144	4	4	3840x2400	1
A10-4C	4096	6	4	3840x2400	1

6.2.13.12 NVIDIA H100 PCIe 94GB (H100 NVL) Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H100 PCIe 94GB (H100 NVL)

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 40: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H100 PCIe 94GB (H100 NVL)

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
H100L-7-94C	96246	1	7	7	MIG 7g.94gb
H100L-4-47C	48128	1	4	4	MIG 4g.47gb
H100L-3-47C	48128	2	3	3	MIG 3g.47gb
H100L-2-24C	24672	3	2	2	MIG 2g.24gb
H100L-1-24C[?]	24672	4	1	1	MIG 1g.24gb
H100L-1-12C	12288	7	1	1	MIG 1g.12gb
H100L-1-12CME[?]	12288	1	1	1	MIG 1g.12gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H100 PCIe 94GB (H100 NVL)

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 41: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H100 PCIe 94GB (H100 NVL)

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
H100L-94C	96246	1	1	3840x2400	1
H100L-47C	48128	2	2	3840x2400	1
H100L-23C	23552	4	4	3840x2400	1
H100L-15C	15360	6	4	3840x2400	1
H100L-11C	11264	8	8	3840x2400	1
H100L-6C	6144	15	8	3840x2400	1
H100L-4C	4096	23	16	3840x2400	1

6.2.13.13 NVIDIA H100 SXM5 94GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H100 SXM5 94GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 42: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H100 SXM5 94GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
H100XL-7-94C	96246	1	7	7	MIG 7g.94gb
H100XL-4-47C	48128	1	4	4	MIG 4g.47gb
H100XL-3-47C	48128	2	3	3	MIG 3g.47gb
H100XL-2-24C	24672	3	2	2	MIG 2g.24gb
H100XL-1-24C[?]	24672	4	1	1	MIG 1g.24gb
H100XL-1-12C	12288	7	1	1	MIG 1g.12gb
H100XL-1-12CME[?]	12288	1	1	1	MIG 1g.12gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H100 SXM5 94GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 43: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H100 SXM5 94GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[7]	Virtual Displays per vGPU
H100XL-94C	96246	1	1	3840x2400	1
H100XL-47C	48128	2	2	3840x2400	1
H100XL-23C	23552	4	4	3840x2400	1
H100XL-15C	15360	6	4	3840x2400	1
H100XL-11C	11264	8	8	3840x2400	1
H100XL-6C	6144	15	8	3840x2400	1
H100XL-4C	4096	23	16	3840x2400	1

6.2.13.14 NVIDIA H100 PCIe 80GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H100 PCIe 80GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 44: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H100 PCIe 80GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
H100-7-80C	81920	1	7	7	MIG 7g.80gb
H100-4-40C	40960	1	4	4	MIG 4g.40gb
H100-3-40C	40960	2	3	3	MIG 3g.40gb
H100-2-20C	20480	3	2	2	MIG 2g.20gb
H100-1-20C?	20480	4	1	1	MIG 1g.20gb
H100-1-10C	10240	7	1	1	MIG 1g.10gb
H100-1-10CME?	10240	1	1	1	MIG 1g.10gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H100 PCIe 80GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 45: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H100 PCIe 80GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[7]	Virtual Displays per vGPU
H100-80C	81920	1	1	3840x2400	1
H100-40C	40960	2	2	3840x2400	1
H100-20C	20480	4	4	3840x2400	1
H100-16C	16384	5	4	3840x2400	1
H100-10C	10240	8	8	3840x2400	1
H100-8C	8192	10	8	3840x2400	1
H100-5C	5120	16	16	3840x2400	1
H100-4C	4096	20	16	3840x2400	1

6.2.13.15 NVIDIA H100 SXM5 80GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H100 SXM5 80GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 46: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H100 SXM5 80GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
H100XM-7-80C	81920	1	7	7	MIG 7g.80gb
H100XM-4-40C	40960	1	4	4	MIG 4g.40gb
H100XM-3-40C	40960	2	3	3	MIG 3g.40gb
H100XM-2-20C	20480	3	2	2	MIG 2g.20gb
H100XM-1-20C[?]	20480	4	1	1	MIG 1g.20gb
H100XM-1-10C	10240	7	1	1	MIG 1g.10gb
H100XM-1-10CME[?]	10240	1	1	1	MIG 1g.10gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H100 SXM5 80GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 47: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H100 SXM5 80GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[7]	Virtual Displays per vGPU
H100XM-80C	81920	1	1	3840x2400	1
H100XM-40C	40960	2	2	3840x2400	1
H100XM-20C	20480	4	4	3840x2400	1
H100XM-16C	16384	5	4	3840x2400	1
H100XM-10C	10240	8	8	3840x2400	1
H100XM-8C	8192	10	8	3840x2400	1
H100XM-5C	5120	16	16	3840x2400	1
H100XM-4C	4096	20	16	3840x2400	1

6.2.13.16 NVIDIA H100 SXM5 64GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H100 SXM5 64GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 48: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H100 SXM5 64GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
H100XS-7-64C	65536	1	7	7	MIG 7g.64gb
H100XS-4-32C	32768	1	4	4	MIG 4g.32gb
H100XS-3-32C	32768	2	3	3	MIG 3g.32gb
H100XS-2-16C	16384	3	2	2	MIG 2g.16gb
H100XS-1-16C[?]	16384	4	1	1	MIG 1g.16gb
H100XS-1-8C	8192	7	1	1	MIG 1g.8gb
H100XS-1-8CME[?]	8192	1	1	1	MIG 1g.8gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H100 SXM5 64GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 49: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H100 SXM5 64GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
H100XS-64C	65536	1	1	3840x2400	1
H100XS-32C	32768	2	2	3840x2400	1
H100XS-16C	16384	4	4	3840x2400	1
H100XS-8C	8192	8	8	3840x2400	1
H100XS-4C	4096	16	16	3840x2400	1

6.2.13.17 NVIDIA H800 PCIe 94GB (H800 NVL) Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H800 PCIe 94GB (H800 NVL)

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 50: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H800 PCIe 94GB (H800 NVL)

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
H800L-7-94C	96246	1	7	7	MIG 7g.94gb
H800L-4-47C	48128	1	4	4	MIG 4g.47gb
H800L-3-47C	48128	2	3	3	MIG 3g.47gb
H800L-2-24C	24672	3	2	2	MIG 2g.24gb
H800L-1-24C[?]	24672	4	1	1	MIG 1g.24gb
H800L-1-12C	12288	7	1	1	MIG 1g.12gb
H800L-1-12CME[?]	12288	1	1	1	MIG 1g.12gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H800 PCIe 94GB (H800 NVL)

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 51: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H800 PCIe 94GB (H800 NVL)

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
H800L-94C	96246	1	1	3840x2400	1
H800L-47C	48128	2	2	3840x2400	1
H800L-23C	23552	4	4	3840x2400	1
H800L-15C	15360	6	4	3840x2400	1
H800L-11C	11264	8	8	3840x2400	1
H800L-6C	6144	15	8	3840x2400	1
H800L-4C	4096	23	16	3840x2400	1

6.2.13.18 NVIDIA H800 PCIe 80GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H800 PCIe 80GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 52: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H800 PCIe 80GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
H800-7-80C	81920	1	7	7	MIG 7g.80gb
H800-4-40C	40960	1	4	4	MIG 4g.40gb
H800-3-40C	40960	2	3	3	MIG 3g.40gb
H800-2-20C	20480	3	2	2	MIG 2g.20gb
H800-1-20C?	20480	4	1	1	MIG 1g.20gb
H800-1-10C	10240	7	1	1	MIG 1g.10gb
H800-1-10CME?	10240	1	1	1	MIG 1g.10gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H800 PCIe 80GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 53: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H800 PCIe 80GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
H800-80C	81920	1	1	3840x2400	1
H800-40C	40960	2	2	3840x2400	1
H800-20C	20480	4	4	3840x2400	1
H800-16C	16384	5	4	3840x2400	1
H800-10C	10240	8	8	3840x2400	1
H800-8C	8192	10	8	3840x2400	1
H800-5C	5120	16	16	3840x2400	1
H800-4C	4096	20	16	3840x2400	1

6.2.13.19 NVIDIA H800 SXM5 80GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU supports MIG-backed virtual GPUs and time-sliced virtual GPUs.

MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H800 SXM5 80GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

For details on GPU instance profiles, see the NVIDIA Multi-Instance GPU User Guide.

Table 54: MIG-Backed NVIDIA vGPU (C-Series) for NVIDIA H800 SXM5 80GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Slices per vGPU	Compute Instances per vGPU	Corresponding GPU Instance Profile
H800XM-7-80C	81920	1	7	7	MIG 7g.80gb
H800XM-4-40C	40960	1	4	4	MIG 4g.40gb
H800XM-3-40C	40960	2	3	3	MIG 3g.40gb
H800XM-2-20C	20480	3	2	2	MIG 2g.20gb
H800XM-1-20C[?]	20480	4	1	1	MIG 1g.20gb
H800XM-1-10C	10240	7	1	1	MIG 1g.10gb
H800XM-1-10CME[?]	10240	1	1	1	MIG 1g.10gb+me

Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H800 SXM5 80GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 55: Time-Sliced NVIDIA vGPU (C-Series) for NVIDIA H800 SXM5 80GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
H800XM-80C	81920	1	1	3840x2400	1
H800XM-40C	40960	2	2	3840x2400	1
H800XM-20C	20480	4	4	3840x2400	1
H800XM-16C	16384	5	4	3840x2400	1
H800XM-10C	10240	8	8	3840x2400	1
H800XM-8C	8192	10	8	3840x2400	1
H800XM-5C	5120	16	16	3840x2400	1
H800XM-4C	4096	20	16	3840x2400	1

6.2.13.20 NVIDIA L40 Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

NVIDIA vGPU (C-Series) for NVIDIA L40

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 56: NVIDIA vGPU (C-Series) for NVIDIA L40

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
L40-48C	49152	1	1	3840x2400	1
L40-24C	24576	2	2	3840x2400	1
L40-16C	16384	3	2	3840x2400	1
L40-12C	12288	4	4	3840x2400	1
L40-8C	8192	6	4	3840x2400	1
L40-6C	6144	8	8	3840x2400	1
L40-4C	4096	12[?]	8	3840x2400	1

6.2.13.21 NVIDIA L40S Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

NVIDIA vGPU (C-Series) for NVIDIA L40S

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 57: NVIDIA vGPU (C-Series) for NVIDIA L40S

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
L40S-48C	49152	1	1	3840x2400	1
L40S-24C	24576	2	2	3840x2400	1
L40S-16C	16384	3	2	3840x2400	1
L40S-12C	12288	4	4	3840x2400	1
L40S-8C	8192	6	4	3840x2400	1
L40S-6C	6144	8	8	3840x2400	1
L40S-4C	4096	12[?]	8	3840x2400	1

6.2.13.22 NVIDIA L20 and NVIDIA L20 Liquid-Cooled Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

The virtual GPU types for the NVIDIA L20 and NVIDIA L20 liquid-cooled GPUs are identical.

NVIDIA vGPU (C-Series) for NVIDIA L20 and NVIDIA L20 Liquid-Cooled

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 58: NVIDIA vGPU (C-Series) for NVIDIA L20 and NVIDIA L20 Liquid-Cooled

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
L20-48C	49152	1	1	3840x2400	1
L20-24C	24576	2	2	3840x2400	1
L20-16C	16384	3	2	3840x2400	1
L20-12C	12288	4	4	3840x2400	1
L20-8C	8192	6	4	3840x2400	1
L20-6C	6144	8	8	3840x2400	1
L20-4C	4096	12[?]	8	3840x2400	1

6.2.13.23 NVIDIA L4 Virtual GPU Types

Physical GPUs per board: 1

NVIDIA vGPU (C-Series) for NVIDIA L4

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 59: NVIDIA vGPU (C-Series) for NVIDIA L4

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
L4-24C	24576	1	1	3840x2400	1
L4-12C	12288	2	2	3840x2400	1
L4-8C	8192	3	2	3840x2400	1
L4-6C	6144	4	4	3840x2400	1
L4-4C	4096	6	4	3840x2400	1

6.2.13.24 NVIDIA L2 Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

NVIDIA vGPU (C-Series) for NVIDIA L2

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 60: NVIDIA vGPU (C-Series) for NVIDIA L2

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
L2-24C	24576	1	1	3840x2400	1
L2-12C	12288	2	2	3840x2400	1
L2-8C	8192	3	2	3840x2400	1
L2-6C	6144	4	4	3840x2400	1
L2-4C	4096	6	4	3840x2400	1

6.2.13.25 NVIDIA RTX 6000 Ada Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

NVIDIA vGPU (C-Series) for NVIDIA RTX 6000 Ada

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 61: NVIDIA vGPU (C-Series) for NVIDIA RTX 6000 Ada

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
RTX 6000 Ada-48C	49152	1	1	3840x2400	1
RTX 6000 Ada-24C	24576	2	2	3840x2400	1
RTX 6000 Ada-16C	16384	3	2	3840x2400	1
RTX 6000 Ada-12C	12288	4	4	3840x2400	1
RTX 6000 Ada-8C	8192	6	4	3840x2400	1
RTX 6000 Ada-6C	6144	8	8	3840x2400	1
RTX 6000 Ada-4C	4096	12[?]	8	3840x2400	1

6.2.13.26 NVIDIA RTX 5880 Ada Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

NVIDIA vGPU (C-Series) for NVIDIA RTX 5880 Ada

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 62: NVIDIA vGPU (C-Series) for NVIDIA RTX 5880 Ada

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
RTX 5880 Ada-48C	49152	1	1	3840x2400	1
RTX 5880 Ada-24C	24576	2	2	3840x2400	1
RTX 5880 Ada-16C	16384	3	2	3840x2400	1
RTX 5880 Ada-12C	12288	4	4	3840x2400	1
RTX 5880 Ada-8C	8192	6	4	3840x2400	1
RTX 5880 Ada-6C	6144	8	8	3840x2400	1
RTX 5880 Ada-4C	4096	12[?]	8	3840x2400	1

6.2.13.27 NVIDIA RTX 5000 Ada Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

NVIDIA vGPU (C-Series) for NVIDIA RTX 5000 Ada

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 63: NVIDIA vGPU (C-Series) for NVIDIA RTX 5000 Ada

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[7]	Virtual Displays per vGPU
RTX 5000 Ada-32C	32768	1	1	3840x2400	1
RTX 5000 Ada-16C	16384	2	2	3840x2400	1
RTX 5000 Ada-8C	8192	4	4	3840x2400	1
RTX 5000 Ada-4C	4096	8	8	3840x2400	1

6.2.13.28 NVIDIA RTX A6000 Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

NVIDIA vGPU (C-Series) for NVIDIA RTX A6000

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 64: NVIDIA vGPU (C-Series) for NVIDIA RTX A6000

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
RTXA6000-48C	49152	1	1	3840x2400	1
RTXA6000-24C	24576	2	2	3840x2400	1
RTXA6000-16C	16384	3	2	3840x2400	1
RTXA6000-12C	12288	4	4	3840x2400	1
RTXA6000-8C	8192	6	4	3840x2400	1
RTXA6000-6C	6144	8	8	3840x2400	1
RTXA6000-4C	4096	12[?]	8	3840x2400	1

6.2.13.29 NVIDIA RTX A5500 Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

NVIDIA vGPU (C-Series) for NVIDIA RTX A5500

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 65: NVIDIA vGPU (C-Series) for NVIDIA RTX A5500

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[2]	Virtual Displays per vGPU
RTXA5500-24C	24576	1	1	3840x2400	1
RTXA5500-12C	12288	2	2	3840x2400	1
RTXA5500-8C	8192	3	2	3840x2400	1
RTXA5500-6C	6144	4	4	3840x2400	1
RTXA5500-4C	4096	6	4	3840x2400	1

6.2.13.30 NVIDIA RTX A5000 Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

NVIDIA vGPU (C-Series) for NVIDIA RTX A5000

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 66: NVIDIA vGPU (C-Series) for NVIDIA RTX A5000

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU in Equal-Size Mode	Maximum vGPUs per GPU in Mixed-Size Mode	Maximum Display Resolution[?]	Virtual Displays per vGPU
RTXA5000-24C	24576	1	1	3840x2400	1
RTXA5000-12C	12288	2	2	3840x2400	1
RTXA5000-8C	8192	3	2	3840x2400	1
RTXA5000-6C	6144	4	4	3840x2400	1
RTXA5000-4C	4096	6	4	3840x2400	1

6.2.13.31 Tesla T4 Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU does **not** support mixed-size mode.

NVIDIA vGPU (C-Series) for Tesla T4

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 67: NVIDIA vGPU (C-Series) for Tesla T4

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Maximum Display Resolution[?]	Virtual Displays per vGPU
T4-16C	16384	1	3840x2400	1
T4-8C	8192	2	3840x2400	1
T4-4C	4096	4	3840x2400	1

6.2.13.32 Tesla V100 SXM2 Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU does **not** support mixed-size mode.

NVIDIA vGPU (C-Series) for Tesla V100 SXM2

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 68: NVIDIA vGPU (C-Series) for Tesla V100 SXM2

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Maximum Display Resolution[?]	Virtual Displays per vGPU
V100X-16C	16384	1	3840x2400	1
V100X-8C	8192	2	3840x2400	1
V100X-4C	4096	4	3840x2400	1

6.2.13.33 Tesla V100 SXM2 32GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU does **not** support mixed-size mode.

NVIDIA vGPU (C-Series) for Tesla V100 SXM2 32GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 69: NVIDIA vGPU (C-Series) for Tesla V100 SXM2 32GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Maximum Display Resolution[?]	Virtual Displays per vGPU
V100DX-32C	32768	1	3840x2400	1
V100DX-16C	16384	2	3840x2400	1
V100DX-8C	8192	4	3840x2400	1
V100DX-4C	4096	8	3840x2400	1

6.2.13.34 Tesla V100 PCIe Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU does **not** support mixed-size mode.

NVIDIA vGPU (C-Series) for Tesla V100 PCIe

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 70: NVIDIA vGPU (C-Series) for Tesla V100 PCIe

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Maximum Display Resolution[?]	Virtual Displays per vGPU
V100-16C	16384	1	3840x2400	1
V100-8C	8192	2	3840x2400	1
V100-4C	4096	4	3840x2400	1

6.2.13.35 Tesla V100 PCIe 32GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU does **not** support mixed-size mode.

NVIDIA vGPU (C-Series) for Tesla V100 PCIe 32GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads

- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 71: NVIDIA vGPU (C-Series) for Tesla V100 PCIe 32GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Maximum Display Resolution[?]	Virtual Displays per vGPU
V100D-32C	32768	1	3840x2400	1
V100D-16C	16384	2	3840x2400	1
V100D-8C	8192	4	3840x2400	1
V100D-4C	4096	8	3840x2400	1

6.2.13.36 Tesla V100S PCIe 32GB Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU does **not** support mixed-size mode.

NVIDIA vGPU (C-Series) for Tesla V100S PCIe 32GB

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 72: NVIDIA vGPU (C-Series) for Tesla V100S PCIe 32GB

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Maximum Display Resolution[?]	Virtual Displays per vGPU
V100S-32C	32768	1	3840x2400	1
V100S-16C	16384	2	3840x2400	1
V100S-8C	8192	4	3840x2400	1
V100S-4C	4096	8	3840x2400	1

6.2.13.37 Tesla V100 FHHL Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU does **not** support mixed-size mode.

NVIDIA vGPU (C-Series) for Tesla V100 FHHL

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 73: NVIDIA vGPU (C-Series) for Tesla V100 FHHL

Virtual GPU Type	Intended Use Case	Frame Buffer (MB)	Maximum vGPUs per GPU	Maximum vGPUs per Board	Maximum Display Resolution[7]	Virtual Displays per vGPU
V100L-16C	Training Workloads	16384	1	1	3840x2400	1
V100L-8C	Training Workloads	8192	2	2	3840x2400	1
V100L-4C	Inference Workloads	4096	4	4	3840x2400	1

6.2.13.38 Quadro RTX 8000 Passive Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU does **not** support mixed-size mode.

NVIDIA vGPU (C-Series) for Quadro RTX 8000 Passive

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 74: NVIDIA vGPU (C-Series) for Quadro RTX 8000 Passive

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Maximum Display Resolution[?]	Virtual Displays per vGPU
RTX8000P-48C	49152	1	3840x2400	1
RTX8000P-24C	24576	2	3840x2400	1
RTX8000P-16C	16384	3	3840x2400	1
RTX8000P-12C	12288	4	3840x2400	1
RTX8000P-8C	8192	6	3840x2400	1
RTX8000P-6C	6144	8	3840x2400	1
RTX8000P-4C	4096	8[?]	3840x2400	1

6.2.13.39 Quadro RTX 6000 Passive Virtual GPU Types

Physical GPUs per board: 1

The maximum number of vGPUs per board is the product of the maximum number of vGPUs per GPU and the number of physical GPUs per board.

This GPU does **not** support mixed-size mode.

NVIDIA vGPU (C-Series) for Quadro RTX 6000 Passive

Intended use cases:

- vGPUs with more than 4096 MB of frame buffer: Training Workloads
- vGPUs with 4096 MB of frame buffer: Inference Workloads

Required license edition: vCS or vWS

These vGPU types support a single display with a fixed maximum resolution.

Table 75: NVIDIA vGPU (C-Series) for Quadro RTX 6000 Passive

Virtual GPU Type	Frame Buffer (MB)	Maximum vGPUs per GPU	Maximum Display Resolution[?]	Virtual Displays per vGPU
RTX6000P-24C	24576	1	3840x2400	1
RTX6000P-12C	12288	2	3840x2400	1
RTX6000P-8C	8192	3	3840x2400	1
RTX6000P-6C	6144	4	3840x2400	1
RTX6000P-4C	4096	6	3840x2400	1

Copyright