NGC on Google Cloud Platform Virtual Machines

Release Notes and Setup Guide

DU-08962-001 _v31 | January 2025

Table of Contents

Chapter 1. Using NGC with Google Cloud Platform

NVIDIA makes available on Google Cloud Platform (GCP) GPU-optimized VMIs for GCP VM instances with NVIDIA A100, V100 or T4 GPUs.

For those familiar with GCP, the process of launching the instance is as simple as logging into GCP and creating a deployment solution using the Google Cloud Launcher. After deploying the NVIDIA GPU-Optimized Image of choice,you can SSH into your VM and start building a host of AI applications in deep learning, machine learning and data science by leveraging the wide range of GPU-accelerated containers, pre-trained models and resources available from the NGC Catalog.

This document provides step-by-step instructions for accomplishing this, including how to use the gcloud CLI.

1.1. Security Best Practices

Cloud security starts with the security policies of your CSP account. Refer to the following link for how to configure your security policies for your CSP:

- Google Cloud security best practices center

Users must follow the security guidelines and best practices of their CSP to secure their VM and account.

1.2. Prerequisites

- You have a Google Cloud account - https://console.cloud.google.com/ .
- Browse the NGC website and identify an available NGC container and to run on the Virtual Machine Instance (VMI).
- You have installed the gcloud SDK if you plan to use the CLI. See setup instructions below.

- Windows Users: The CLI code snippets are for bash on Linux or Mac OS X. If you are using Windows and want to use the snippets as-is, you can use the Windows Subsystem for Linux and use the bash shell (you will be in Ubuntu Linux).

1.3. Before You Get Started

Be sure you are familiar with the information in this chapter before starting to use the NVIDIA GPU Cloud Image on the Google Cloud Platform (GCP).

1.3.1. Set Up Your SSH Key

The Google Compute Engine generates and manages an SSH key automatically for logging into your instance (see the Google Cloud documentation Connecting to Instances.). However, to facilitate logging into the NGC container registry upon the initial connection to the VM instance, you need to -

1. Generate your own SSH keys (see Creating a new SSH key for instructions), and then
2. Add them to the metadata for your project (see Adding or Removing Project-Wide Public SSH Keys for instructions).

If you do not prepare your SSH keys before launching and connecting to your VM instance, you will not be able to access the NGC initially. In that case you will need to

1. Add yourself to the docker group after connecting to the instance.

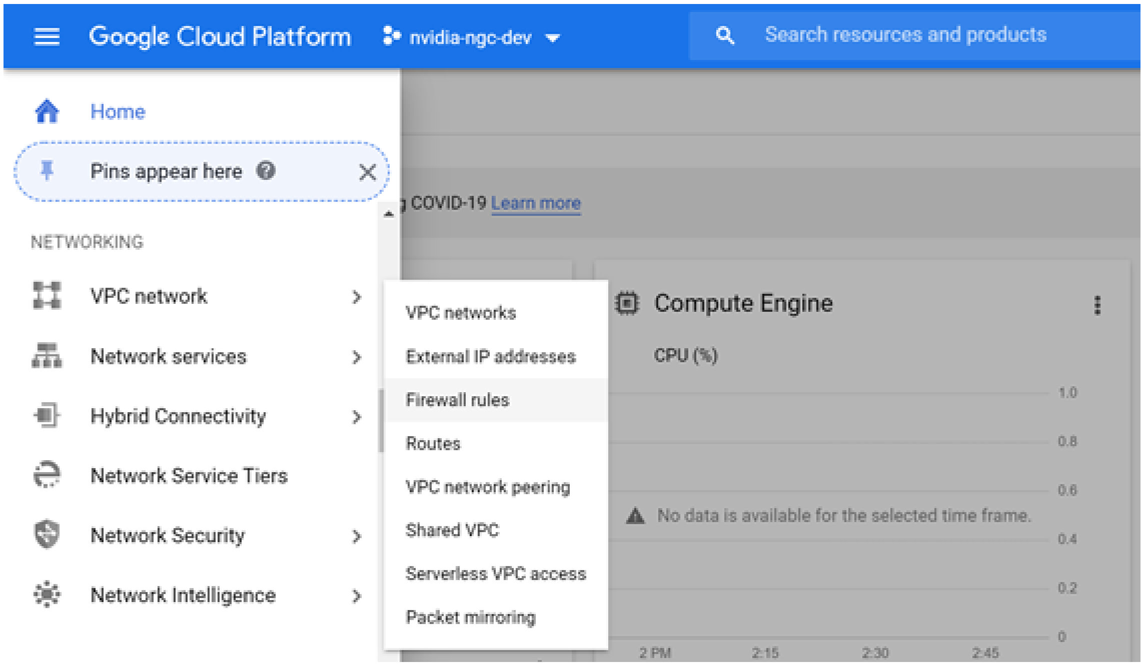
```
sudo usermod -aG docker $USER
```

2. Restart the session.

1.3.2. Set Up Firewall Rules

NVIDIA recommends setting firewall rules to allow external access to ports 443 (HTTPS), 8888 (DIGITS), and any other ports that may be needed. This should be done before launching an instance to avoid having to stop the instance when setting any firewall rules later.

1. Log in to https://console.cloud.google.com
2. Verify you are in the correct Project.
3. Click the Products and Services menu icon, then scroll down to the Networking section and click VPC Network->Firewall Rules.



4. Click Create Firewall Rule.
5. Enter the following information to specify the firewall rule you want to create.
   - Name: NVIDIA recommends the following naming format

     For HTTPS: “default-allow-https”

     For DIGITS: “default-allow-digits”

     You can also create rules for other DIGITS versions, such as DIGITS4
   - Direction of traffic: "Ingress"
   - Action on match: "Allow"
   - Targets: "All instances in the network"
   - Source filter: "IP ranges"
   - Source IP ranges: "0.0.0.0/0"
   - Protocols and ports: "Specified protocols and ports", then enter

     For HTTPS: “tcp:443”

     For DIGITS: “tcp:8888”

     You can enter ports for other DIGITS versions as well

   Security Warning

   It is important to use proper precautions and security safeguards prior to granting access, or sharing your AMI over the internet. By default, internet connectivity to the AMI instance is blocked. You are solely responsible for enabling and securing access to your AMI. Please refer to Google Cloud Platform guides for managing security groups.
6. Click Create.

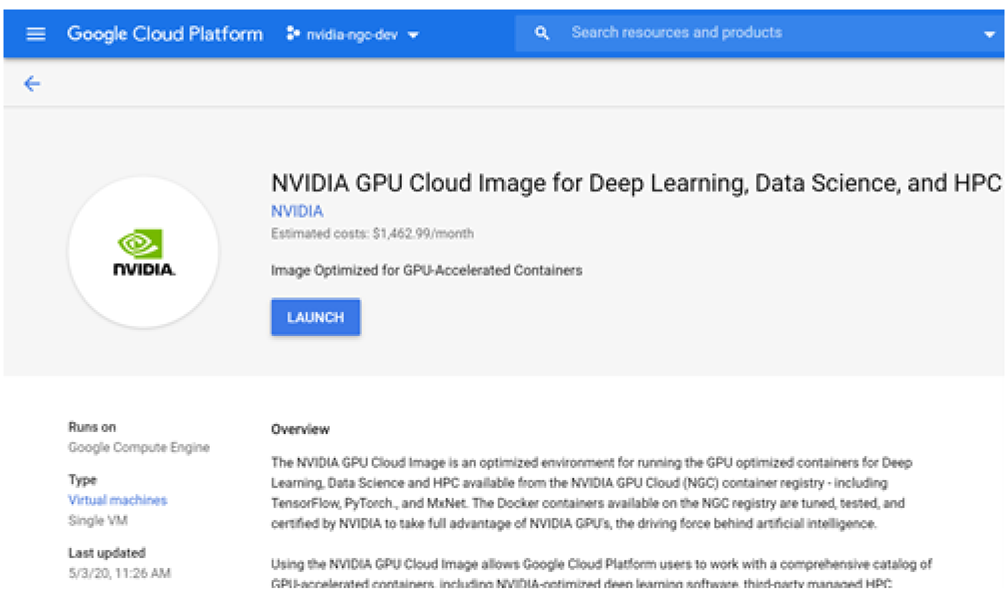
Your new firewall rules should appear on the Firewall Rules page.

1.4. Creating an NGC Certified Virtual Machine Using the GCP Console

1.4.1. Log In and Create VM

1. Log in to https://console.cloud.google.com.
2. Verify you are in the correct project.
3. Open the Google Cloud Platform Marketplace page.

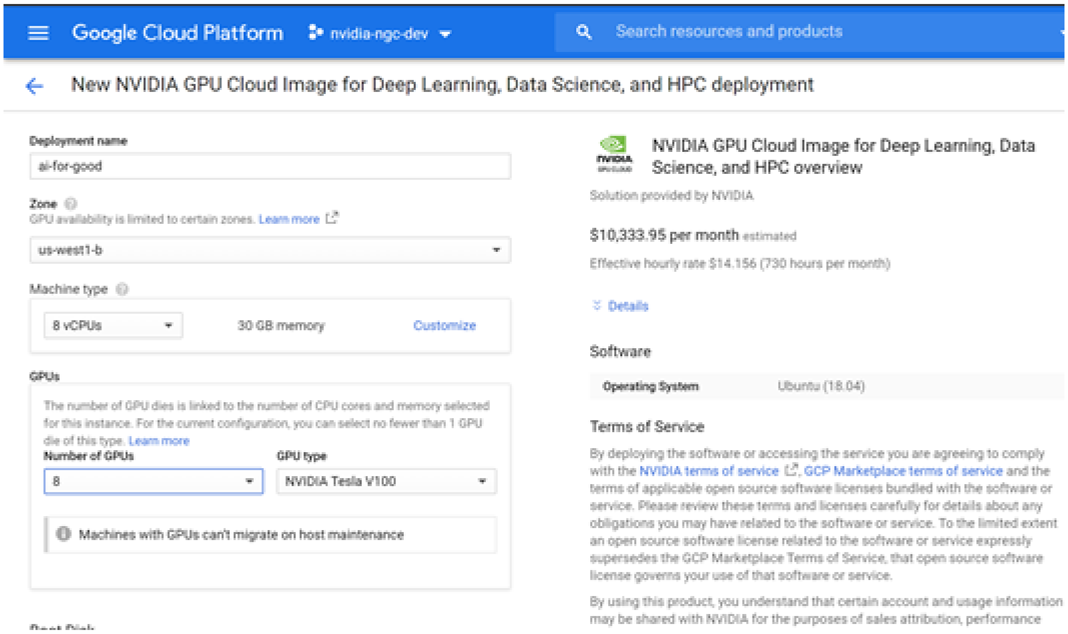
   Click the Products and Services menu icon and select Marketplace.
4. Search for "nvidia".
5. Select your choice of the three flavors of NVIDIA GPU-optimized images published by NVIDIA.
6. From the image information page, click Launch.



7. Configure the NVIDIA GPU Cloud Image deployment.
   a). In "Name", enter your new deployment name.
   b). In "Zone", select the zone to create the instance (select one that features the appropriate GPU).
   c). In the "Machine Type" section, click Customize to open the customize view.
   d). Under the GPU section, select the GPU type and Number of GPUs.

   Assign the Cores (vCPUs) and Memory. The following ratio is recommended: 1x GPU : 10x vCPU: 60 GB mem
   e). In the "Boot disk" section, select Standard Persistent Disk.

f). Make other changes as needed for Networking, Firewall and IP.

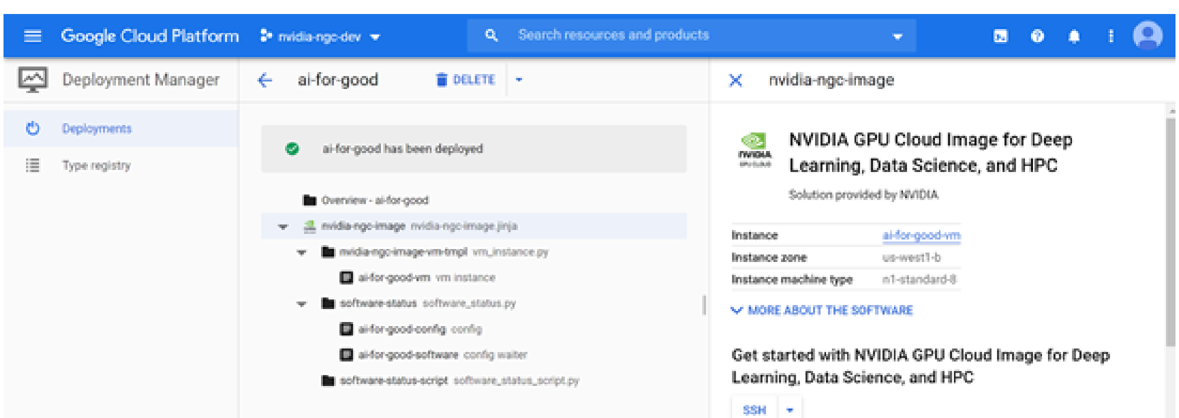


8. Click Deploy from the bottom of the page.
   It may take a few minutes for the deployment process to complete.

1.4.2. Connect to Your VM Instance

- If you are still on the Deployment page, you can click SSH to connect to your instance.
- If you are no longer on the Deployment page, you can return to your instance and connect as follows.

1. Click the Products and Services menu icon, then scroll down to the Compute Engine section and click VM Instances.
2. Either click SSH by your listed deployed instance, or click your deployed instance and then click SSH from the VM instance details page.



1.4.3. Start/Stop Your VM Instance

Select your GPU instance, either from the Deployment Manager->your deployment page or from the Compute Engine->VM Instances page.

The top menu lets you edit, stop a running instance, or start a stopped instance.

Delete VM and Assocated Resources

Select your GPU instance, either from the Deployment Manager->your deployment page or from the Compute Engine->VM Instances page.and then click Delete.

1.5. Create an NGC Certified Virtual Machine Using the gcloud CLI

This section explains how to create a GPU Cloud instance using the gcloud CLI.

This flow and the code snippets in this section are for Linux or Mac OS X. If you are using Windows, you can use the Windows Subsystem for Linux and use the bash shell (where you will be in Ubuntu Linux). Many of these CLI commands can have significant delays.

For more information about creating a deployment using gcloud CLI, see Creating a Deployment using gcloud or the API.

1.5.1. Install and Set Up gcloud CLI

Follow the instructions at https://cloud.google.com/sdk/docs/quickstarts. These include instructions for Linux, Mac, and Windows.

The instructions walk you through the platform specific install and initial gcloud login.

For at least the Mac, you will be given a large list of additional gcloud components to install such as extensions for GO, Python and Java. You can use the defaults for now, and use the `gcloud components` command later to list, install, or remove them.

Once the setup is complete, start a new shell since your environment has been updated.

1.5.2. Set Up Instance Options

You will need to specify the following options when creating the custom GPU instance.

OPTION[1]	VALUE	NOTES
<instance-name>	Name of your choosing. Ex. “my-ngc-instance”	Must be all lowercase, with no spaces. Hyphens and numbers are allowed.

`--project`	"<my-project-id>"	This is the project in which the VM will be created. Use `gcloud projects list` to view PROJECT ID to use for this field.
`--zone`	One of the following zones that contain GPUs: "us-west1-b" "us-east1-c" "us-east1-d" "europe-west1-b" "europe-west1-d" "asia-east1-a" "asia-east1-b"	Pick one nearest you and with the GPUs you want to use.
`--machine-type`	One of the following: "custom-10-61440" (for 1x P100 or V100) "custom-20-122880" (for 2x P100) "custom-40-212992" (for 4x P100) "custom-80-491520" (for 8x V100)	vCPU/Memory configuration of the VM in "custom-<#vCPUs>-<memory MB>" format. Recommended ratio is 1 GPU : 10 vCPUs : 60 GB memory
`--subnet`	“default”, or the name of the VPC network to use	
`--metadata`	"ssh-keys=<user-id>:ssh-rsa <ssh-key> <user-email>"	
`--maintenance-policy`	"TERMINATE"	What to do with your instance when Google performs maintenance on the host
`--service-account`		Compute Engine identity attached to the instance.

		Use `gcloud iam service-accounts list` to view the email for your account.
`--scope`	"https://www.googleapis.com/auth/devstorage.read_only", "https://www.googleapis.com/auth/logging.write", "https://www.googleapis.com/auth/monitoring.write", "https://www.googleapis.com/auth/servicecontrol", "https://www.googleapis.com/auth/service.management.readonly", "https://www.googleapis.com/auth/trace.append"	Default values (recommended). Specifies the permissions for your instance.
`--accelerator`	nvidia-tesla-p100,count=[1,2,4]	Which GPU to attach, and how many
`--min-cpu-platform`	"Intel Broadwell" (for P100 instances)	
`--image`		Name of the latest NVIDIA GPU Cloud Image (See the NGC GCP VMI Release Notes for the current name.)
`--image-project`	"nvidia-ngc-public"	Project name in which the NVIDIA GPU Cloud Image is located
`--boot-disk-size`	32	
`--boot-disk-type`	"pd-standard"	
`--boot-disk-device-name`	Name of your choosing	Recommend using the same name as your VM instance for easy correlation

1.5.3. Launch Your VM Instance

Use the Python scripts provided at https://github.com/nvidia/ngc-examples/tree/master/ncsp to create your custom GPU instance. You can also enter the following, using the information gathered in the previous section:

```
gcloud compute \
--project "<project-id>" \
instances create "<instance-name>" \
--zone "<zone>" \
--machine-type "<vCPU-mem-config>" \
--subnet "<subnet-name>" \
--metadata "<your-public-ssh-key>" \
--maintenance-policy "<maintenance-policy>" \
--service-account "<service-account-email>" \
--scopes "https://www.googleapis.com/auth/devstorage.read_only","https://
www.googleapis.com/auth/logging.write","https://www.googleapis.com/auth/
monitoring.write","https://www.googleapis.com/auth/servicecontrol","https://
www.googleapis.com/auth/service.management.readonly","https://www.googleapis.com/
auth/trace.append" \
--accelerator type=<accelerator-type> \
--min-cpu-platform "<CPU-platform>" \
--image "<nvidia-gpu-cloud-image>" \
--image-project "<project-name>" \
--boot-disk-size "32" \
--boot-disk-type "pd-standard" \
--boot-disk-device-name "<boot-disk-name>"
```

1.5.4. Connect to Your VM Instance

(Use a CLI on Mac or Linux. Windows users: use OpenSSH on Windows PowerShell or use the Windows Subsystem for Linux)

If you ran the scripts from https://github.com/nvidia/ngc-examples/tree/master/ncsp you should be connected to your instance. Otherwise, run `ssh` to connect to your GPU instance, or enter the following gcloud command.

Command syntax:

```
gcloud compute --project "<project-id>" ssh --zone "<zone>" "<instance-name>"
```

See https://cloud.google.com/compute/docs/instances/connecting-to-instance for more information about connecting to your GPU instance.

1.5.5. Stop/Stop Your VM Instance

Once an instance is running, you can stop and (re)start your instance.

Stop:

```
gcloud compute instances stop <instance-name>
```

Start or Restart:

```
gcloud compute instances start <instance-name> <zone>
```

1.6. Access Jupyter Notebooks in Your GPU Virtual Machine

Accessing Jupyter notebooks you create or download from the NGC Catalog in your Google Cloud virtual machine is simple and straightforward.

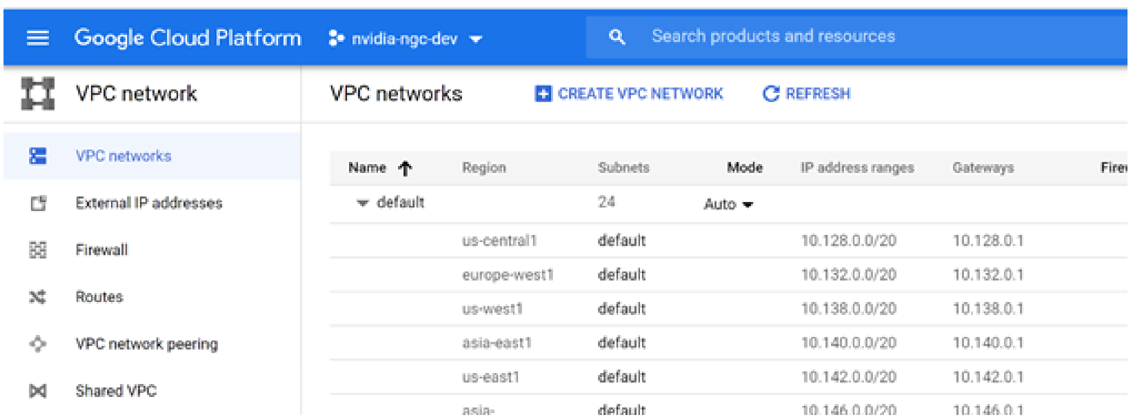
Follow the below steps to configure your virtual machine instance with the right network settings to be able to run a JupyterLab server on your VM and access Jupyter notebooks via a browser on your local machine.

1. Create a NGC certified virtual machine instance on Google Cloud by following the steps listed above. (LINK TO ‘CREATING AN NGC CERTIFIED VIRTUAL MACHINE USING THE GOOGLE CLOUD CONSOLE’ SECTION ABOVE)

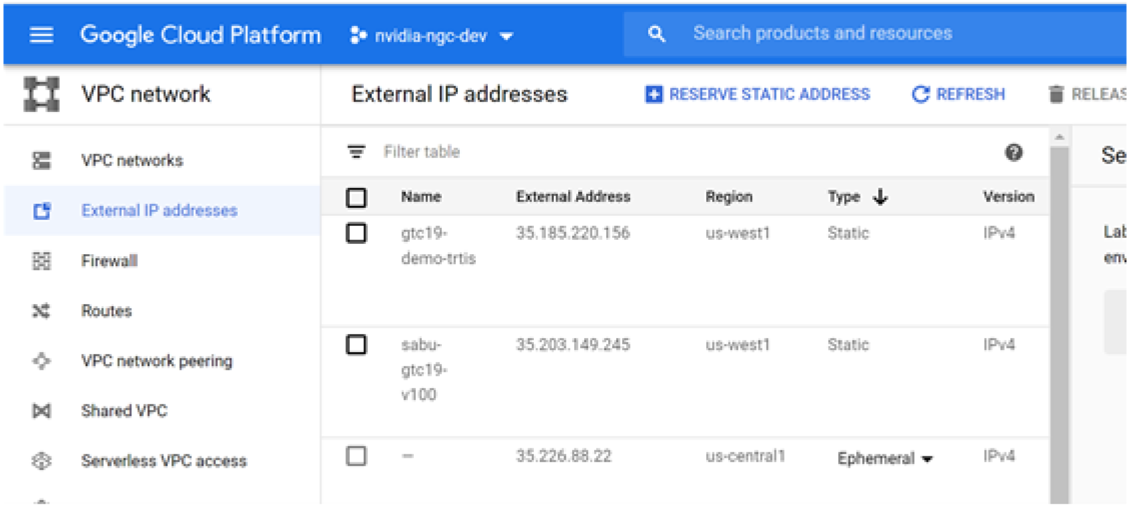
   Refer to Creating an NGC Certified Virtual Machine Using the GCP Console
2. Create a static external IP address.

   This static external IP address will be used as hostname when you access the Jupyter notebook from your local browser. (example: http://<external IP address>:8080)

   a). Navigate to Networking (from menu) → VPC network → External IP addresses.



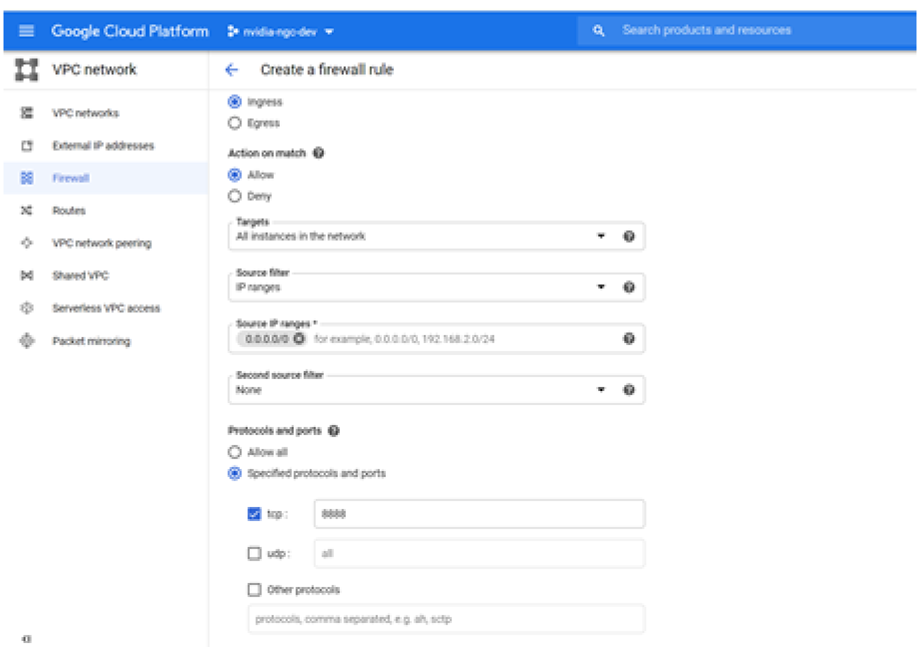
   b). Identify the virtual machine instance you created and change the the type from “Ephemeral” to “Static”



c). Copy the external IP in the corresponding External Address column for use later.

3. Navigate to Networking (from menu) → VPC network → Firewall rules to create a new firewall rule with the following parameters (or add a new rule to an existing VPC if you already created one).

- Name: <Enter a firewall name>
- Targets: All instances in the network
- Source IP ranges: 0.0.0.0/0
- Protocols and ports: Select "Specified protocols and ports" option. tcp: 8080 <You can change any other port number>



That's it! Now you're all set to create and edit Jupyter notebooks that are in your virtual machine instance.

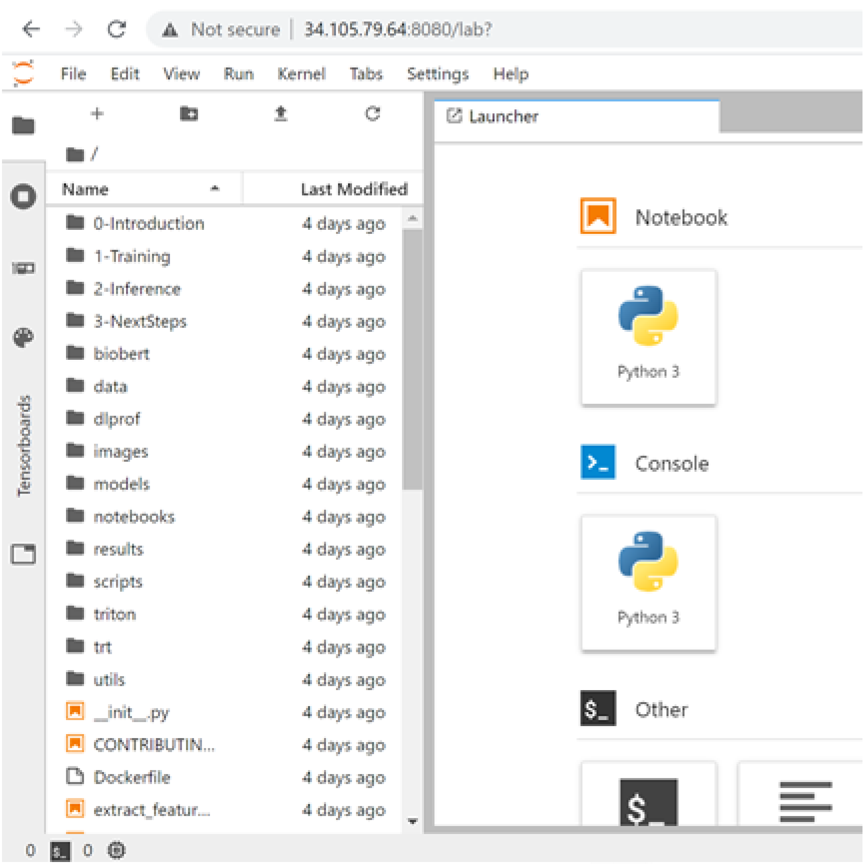
Now you can pull any container from NGC and access Jupyter notebooks from within the container as well.

While running the container, make sure to include the port you configured for the JupyterLab while creating the VPC (in this example, the port used was 8080)

For example:

```
$ docker run --gpus '"device=1"' --rm -it \
    -p 8080:8080 \
    -p 6006:6006 \
    --shm-size=1g \
    --ulimit memlock=-1 \
    --ulimit stack=67108864 \
    --name bert_gcp \
$ docker pull nvcr.io/nvidia/tensorflow:20.08-tf1-py3
```

You can now access the Jupyter notebooks in your Google Cloud virtual machine by simply navigating to `https://<externalip>:8080` on any browser on your local machine. (External IP to be included in the URL is the same as the external IP you made a note of in step 2c)



1.7. Persistent Data Storage for GCP Virtual Machines

GCP recommends using Persistent SSD Disks for Compute Engine storage. A minimum of 1 TB of storage is recommended for storing deep learning datasets. However, a much larger disk or a software RAID, using mdadm, can be used to create a volume with multiple SSD Persistent Disks for achieving the the maximum performance supported by GCP on a Compute Engine instance. See instructions on how to set up software RAID on local disks. Persistent SSD disks can also be set up for software RAID using the same instructions.

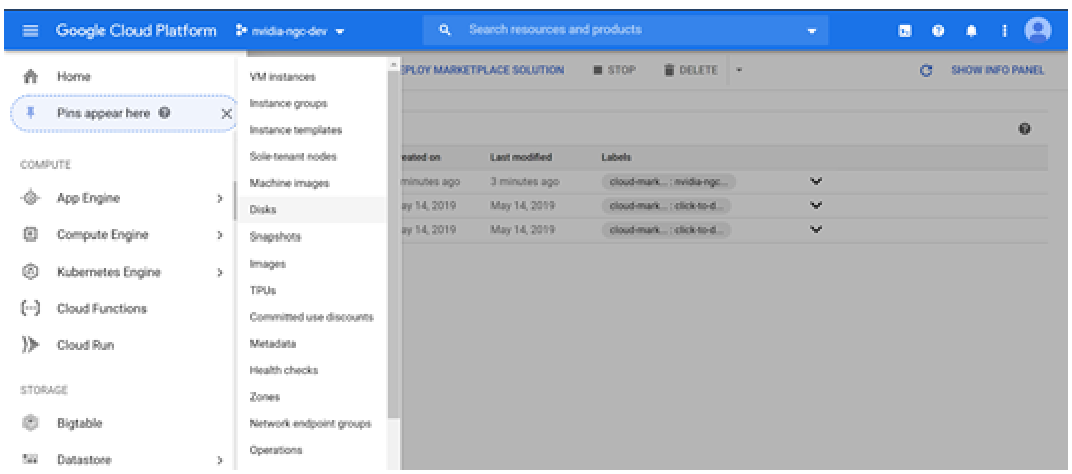
1.7.1. Create a Data Disk Using the GCP Console

You can create a persistent SSD dataset disk from the GCP console as follows.

1. Log on to the Google Cloud Platform.
2. Create the SSD disk.

   a). Click Compute Engine-> Disks in the left-side navigation pane.

   b). Click Create Disk from the top of the page.



   c). Specify the following and then click Create when done:

   - Zone: Select the same zone as the VM instance you created.
   - Disk Type: SSD persistent disk
   - Source type: None (blank disk)
   - Size: At least 1024 GB

   If you choose to provide your own encryption key, You must provide a key that is a 256-bit string encoded in RFC 4648 standard base64 to Compute Engine. See Customer-Supplied-Encryption-Keys for details on how to provide a custom Encryption Key globally for all your operations.

3. Attach the disk to the VM instance.
   a). Go to the Compute Engine->VM Instance page.
   b). Click your VM instance from the list.
   c). Click Stop.

      You must stop a running VM Instance as changes cannot be performed when the instance is running.
   d). Click Edit.
   e). Scroll down to the Additional Disks and click + Add Item.
   f). Under Name, select the disk that you created and want to attach to the VM instance.
   g). Click Save.
   h). Start the VM instance.

1.7.2. Create a Data DIsk Using the gcloud CLI

1. Create the disk using the following command.

```
$ gcloud compute disks create ngc-ssd --zone <zone> --description "<your-
description>" --type=pd-ssd --size=1000GM
```

2. Attach the disk to a VM instance using the following command..

```
$ gcloud compute instances attach-disk <instance-name> --disk ngc-ssd --zone
 <zone>
```

1.7.3. Delete a Data Disk

Be aware that once you delete a Persistent SSD Disk, you cannot undelete it.

gcloud CLI

```
$ gcloud compute instances detach-disk <instance-name> --disk ngc-ssd --zone <zone>
```

GCP Console

1. Click the disk to delete from the Compute Engine->Disks page.
2. On the top of the page, click Delete.
3. Click Delete at the Delete a disk confirmation dialog. .

Chapter 2. NVIDIA Virtual Machine Images on the Google Cloud Platform

NVIDIA makes available on the Google Cloud Platform a customized NVIDIA virtual machine image optimized for the NVIDIA® Volta™ GPU. Running NVIDIA GPU Cloud containers on this instance provides optimum performance for deep learning jobs.

See the NGC with Google Cloud Platform Setup Guide for instructions on setting up and using the VMI.

2.1. NVIDIA AI Enterprise

2.1.1. Information

NVIDIA AI Enterprise is a secure, end-to-end, cloud-native suite of AI software enabling organizations to solve new challenges while increasing operational efficiency. It accelerates the data science pipeline and streamlines the development, deployment, and management of predictive AI models to automate essential processes and gain rapid insights from data. With an extensive library of full-stack software, including AI solution workflows, frameworks, pre-trained models, and infrastructure optimization. Global enterprise support and regular security reviews ensure business continuity and AI projects are successful and stay on track.

With NVIDIA AI Enterprise, customers get support and access to the following:

- NVIDIA AI Workflows, prepackaged reference applications that include Helm Charts, Jupyter Notebooks, and documentation to enable fast time to production for contact center intelligent virtual assistants, audio transcription, and cybersecurity digital fingerprinting to detect anomalies. Only available with an NVIDIA AI Enterprise subscription.
- Unencrypted pre-trained models for AI explainability, understanding model weights and biases, and faster debugging and customization. Only available with an NVIDIA AI Enterprise subscription.

- Frameworks and tools to accelerate AI development (PyTorch, TensorFlow, NVIDIA RAPIDS, TAO Toolkit, TensorRT, and Triton Inference Server).
- Healthcare-specific frameworks and applications including NVIDIA Clara MONAI and NVIDIA Clara Parabricks.
- NVIDIA AI Enterprise includes support for all NVIDIA AI software published on the NGC public catalog labeled "NVIDIA AI Enterprise Supported." Over 50 pre-trained models, frameworks, and development tools.
- The NVIDIA AI Enterprise marketplace offer also includes a VMI, which provides a standard, optimized run time for easy access to the NVIDIA AI Enterprise software mentioned above and ensures development compatibility between clouds and on-premises infrastructure. Develop once, run anywhere.

Contact NVIDIA to learn more about NVIDIA AI Enterprise on Google Cloud and for private pricing by filling out the form here.

To get started, refer to the NVIDIA AI Enterprise on Google Cloud Marketplace Quick Start Guide.

2.1.2. Release Notes

Version 24.12.1-NVAIE 5.2

- Ubuntu Server 22.04 LTS (x86)
- NVIDIA vGPU Driver 550.127.05
- Docker CE 27.4.0
- NVIDIA Container Toolkit 1.17.3-1
- GCP CLI (latest version)
- Miniconda
- JupyterLab (latest version) and core Jupyter packages
- NGC CLI 3.56.0
- Git, Python3, and pip

Version 24.11.2-NVAIE 5.2

- Ubuntu Server 22.04 LTS (x86)
- NVIDIA vGPU Driver 550.127.05
- Docker CE 27.3.1
- NVIDIA Container Toolkit 1.17.1-1
- GCP CLI (latest version)
- Miniconda
- JupyterLab (latest version) and core Jupyter packages

- NGC CLI 3.54.0
- Git, Python3, and pip

Version 24.07.3-NVAIE 5.1

- Ubuntu Server 22.04 (x86)
- NVIDIA vGPU Driver 550.90.07
- Docker-CE 26.1.4
- NVIDIA Container Toolkit 1.15.0-1
- Latest GCP CLI
- Miniconda
- JupyterLab latest and other Jupyter core packages
- NGC CLI 3.43.0
- Git, Python3-PIP

Version 24.03-NVAIE 5.0

- Ubuntu Server 22.04 (x86)
- NVIDIA vGPU Driver 550.54.14
- Docker-CE 26.0.0
- NVIDIA Container Toolkit 1.14.6-1
- Latest GCP CLI
- Miniconda
- JupyterLab latest and other Jupyter core packages
- NGC CLI 3.41.1
- Git, Python3-PIP

Version 24.01-NVAIE 4.2

- Ubuntu Server 22.04 (x86)
- NVIDIA vGPU Driver 535.154.05
- Docker-CE 25.0.3
- NVIDIA Container Toolkit 1.14.5-1
- Latest GCP CLI
- Miniconda
- JupyterLab latest and other Jupyter core packages
- NGC CLI 3.38
- Git, Python3-PIP

Version 23.11-NVAIE 4.1

- Ubuntu Server 22.04 (x86)
- NVIDIA vGPU Driver 535.129.03
- Docker-CE 24.0.7
- NVIDIA Container Toolkit 1.14.3-1
- Latest GCP CLI
- Miniconda
- JupyterLab latest and other Jupyter core packages
- NGC CLI 3.22.0
- Git, Python3-PIP

Version 23.09.2-NVAIE 4.0

- Ubuntu Server 22.04
- NVIDIA AI Enterprise Catalog access script
- NVIDIA vGPU Driver 535.54.03 (v16.0)
- Docker-CE 24.0.5
- NVIDIA Container Toolkit 1.13.5
- NGC CLI 3.22.0
- Miniconda
- JupyterLab (within miniconda)

Version 22.12-NVAIE 3.0

- Ubuntu Server 20.04
- NVIDIA vGPU Driver 525.60.13
- Docker-ce 20.10.21
- NVIDIA Container Toolkit 1.11.0-1
- Google Cloud CLI 411.0.0
- NGC CLI 3.10.0
- Miniconda 22.9.0
- JupyterLab (within conda base env) 3.5.01
- Git 2.25.1
- NVIDIA AI Enterprise access script

2.2. NVIDIA Riva VMI

2.2.1. Information

NVIDIA® Riva is a GPU-accelerated software development kit (SDK) for building and deploying fully customizable, real-time multilingual speech and translation AI pipelines —including automatic speech recognition (ASR), neural machine translation (NMT), and text-to-speech (TTS). Riva delivers world-class, out-of-the-box, accurate transcriptions and translations and engaging professional voices. It can be deployed on-premises, in any cloud environment, in embedded systems, and at the edge.

With NVIDIA Riva, customers can customize for different languages, accents, domains, vocabulary, and contexts to achieve the best possible accuracy for their use cases, as well as the desired brand voice and intonation. It provides consistent experiences for hundreds of thousands of input streams with higher inference performance compared to existing technologies. To accelerate building speech-AI-based AI solutions, Riva offers pre-packaged AI workflows for audio transcription and intelligent virtual assistants that include pretrained models and resources such as Helm Charts, Jupyter Notebooks, and documentation.

Riva, a premium edition of the NVIDIA AI Enterprise software platform, is available for $60 per GPU hour for high-performance GPUs such as NVIDIA A100. Contact NVIDIA for special pricing for alternative GPU options or private offers.

2.2.2. Release Notes

Version 24.05

- Ubuntu Server 22.04 (x86)
- NVIDIA GRID Driver 550.54.14 (vGPU 17.0)
- Docker-ce 26.1.1
- NVIDIA Container Toolkit 1.15.0-1
- Latest Google Cloud SDK
- Latest Miniconda and JupyterLab
- NGC CLI 3.41.4

Version 2023.06.2-riva

- Ubuntu Server 22.04
- NVIDIA vGPU Driver 525.60.13
- Docker-CE 24.0.2
- NVIDIA Container Toolkit 1.11.0-1
- GCP CLI
- NGC CLI 3.22.0

- Miniconda 23.5.0
- JupyterLab (within conda base env) 34.0.2
- Git 2.34.1

2.3. NVIDIA GPU-Optimized VMI

2.3.1. Information

The NVIDIA GPU-Optimized VMI is a virtual machine image for accelerating your Machine Learning, Deep Learning, Data Science and HPC workloads. Using this AMI, you can spin up a GPU-accelerated Compute Engine VM instance in minutes with a pre-installed Ubuntu OS, GPU driver, Docker and NVIDIA container toolkit.

Moreover, this VMI provides easy access to NVIDIA's NGC Catalog, a hub for GPU-optimized software, for pulling & running performance-tuned, tested, and NVIDIA certified docker containers. NGC provides free access to containerized AI, Data Science, and HPC applications, pre-trained models, AI SDKs and other resources to enable data scientists, developers, and researchers to focus on building solutions, gathering insights, and delivering business value.

This GPU-optimized VMI is provided free of charge for developers with an enterprise support option. For more information on enterprise support, please visit NVIDIA AI Enterprise

2.3.2. Release Notes

Version 24.10.1

- Ubuntu Server 22.04 LTS (x86)
- NVIDIA TRD Driver 550.127.05
- Docker CE 27.3.1
- NVIDIA Container Toolkit 1.16.2-1
- GCP CLI (latest version)
- Miniconda (latest version)
- JupyterLab (latest version) and core Jupyter packages
- NGC CLI 3.53.0
- Git, Python3, and pip

Version 24.05

- Ubuntu Server 22.04 (x86)

- NVIDIA TRD Driver 550.54.15
- Docker-ce 26.1.2
- NVIDIA Container Toolkit 1.15.0
- Latest GCP CLI
- Miniconda
- JupyterLab latest and other Jupyter core packages
- NGC CLI 3.41.4
- Git, Python3-PIP

Version 24.03.4

- Ubuntu Server 22.04 (x86)
- NVIDIA TRD Driver 535.161.07
- Docker-ce 26.0.0
- NVIDIA Container Toolkit 1.14.6
- Latest GCP CLI
- Miniconda
- JupyterLab latest and other Jupyter core packages
- NGC CLI 3.40.0
- Git, Python3-PIP

Version 23.09.1

- Ubuntu Server 22.04 (x86)
- NVIDIA TRD Driver 535.54.03
- Docker-ce 24.0.6
- NVIDIA Container Toolkit 1.13.5
- Latest GCP CLI
- Miniconda
- JupyterLab latest and other Jupyter core packages
- NGC CLI 3.22.0
- Git, Python3-PIP

Version 23.02.0

- Ubuntu Server 20.04
- NVIDIA Driver 525.85.12
- Docker-ce 20.10.23
- NVIDIA Container Toolkit 1.12.0-1

- NVIDIA Container Runtime 3.12.0-1
- GCP Command Line Interface (CLI)
- Miniconda (latest)
- JupyterLab 3.5.4 and other Jupyter core packages
- NGC-CLI 3.14.0
- Git, Python3-PIP

Key Changes

- Updated NVIDIA Driver to 525.85.12
- Updated Docker-ce to 20.10.23
- Updated NVIDIA Container Toolkit to Version 1.12.0-1
- Updated NVIDIA Container Runtime to Version 3.12.0-1
- Updated NGC-CLI to 3.14.0
- Fixed CVE-2022-3515

Version 22.06.0

- Ubuntu Server 20.04
- NVIDIA Driver 515.48.07
- Docker-ce 20.10.17
- NVIDIA Container Toolkit 1.10.0-1
- NVIDIA Container Runtime 3.10.0-1
- GCP Command Line Interface (CLI)
- Miniconda 4.13.0
- JupyterLab 3.4.3 and other Jupyter core packages
- NGC-CLI 3.0.0
- Git, Python3-PIP

Key Changes

- Updated NVIDIA Driver to 515.48.07
- Updated Docker-ce to 20.10.17
- Updated Nvidia Container Toolkit to Version 1.10.0-1
- Updated Nvidia Container Runtime to Version 3.10.0-1
- Packaged additional tools: Miniconda, JupyterLab, NGC-CLI, Git, Python3-PIP

2.4. NVIDIA HPC SDK GPU-Optimized Image

2.4.1. Information

The NVIDIA HPC SDK C, C++, and Fortran compilers support GPU acceleration of HPC modeling and simulation applications with standard C++ and Fortran, OpenACC directives, and CUDA. GPU-accelerated math libraries maximize performance on common HPC algorithms, and optimized communications libraries enable standards-based multi-GPU and scalable systems programming. Performance profiling and debugging tools simplify porting and optimization of HPC applications, and containerization tools enable easy deployment on-premises or in the cloud.

Key features of the NVIDIA HPC SDK for Linux include:

- Support for NVIDIA Ampere Architecture GPUs with FP16, TF32 and FP64 tensor cores
- NVC++ ISO C++17 compiler with Parallel Algorithms acceleration on GPUs, OpenACC and OpenMP
- NVFORTRAN ISO Fortran 2003 compiler with array intrinsics acceleration on GPUs, CUDA Fortran, OpenACC and OpenMP
- NVC ISO C11 compiler with OpenACC and OpenMP
- NVCC NVIDIA CUDA C++ compiler
- NVIDIA Math Libraries including cuBLAS, cuSOLVER, cuSPARSE, cuFFT, cuTENSOR and cuRAND
- Thrust, CUB, and libcu++ GPU-accelerated libraries of C++ parallel algorithms and data structures
- NCCL, NVSHMEM and Open MPI libraries for fast multi-GPU/multi-node communications
- NVIDIA Nsight Systems/Compute for interactive HPC applications performance profiler

2.4.2. Release Notes

Version 23.11

- Ubuntu Server 22.04 (x86)
- NVIDIA Driver 535.129.03
- Docker-ce 24.0.7
- NVIDIA Container Toolkit Version: 1.14.3-1
- Latest GCP CLI
- Miniconda latest
- JupyterLab latest and other Jupyter core packages
- NGC-CLI 3.35.0

- Git
- Updated HPC SDK 23.11

Version 23.03.0

- Ubuntu Server 20.04
- NVIDIA Driver 525.85.12
- Docker-ce 23.0.1
- NVIDIA Container Toolkit Version: 1.12.1-1
- AWS CLI, NGC-CLI 3.16.0
- Miniconda latest
- JupyterLab and other Jupyter core packages
- Git, Python3-PIP
- HPC SDK 23.1
- NVIDIA Peer Memory: 1.3
- MOFED: 5.8-1.0.1.1

Key Changes

- Updated NVIDIA Driver to 525.85.12
- Updated Docker-ce to 23.0.1
- Updated Nvidia Container Toolkit to Version 1.12.1-1
- Updated Nvidia Container Runtime to Version 3.12.0-1
- Updated NGC-CLI to 3.16.0
- Updated HPC SDK to 23.1

Version 23.02.0

- Ubuntu Server 20.04
- NVIDIA Driver 525.85.12
- Docker-ce 20.10.23
- NVIDIA Container Toolkit Version: 1.11.1-1
- NVIDIA Container Runtime Version: 3.12.0-1
- GCP Command Line Interface (CLI)
- Miniconda (latest)
- JupyterLab 3.5.4 and other Jupyter core packages
- NGC-CLI 3.14.0
- Git, Python3-PIP
- HPC SDK 23.1
- NVIDIA Peer Memory

- MOFED 5.8-1.0.1.1

Key Changes

- Updated NVIDIA Driver to 525.85.12
- Updated Docker-ce to 20.10.23
- Updated NVIDIA Container Toolkit to Version 1.20.0-1
- Updated NVIDIA Container Runtime to Version 3.12.0-1
- Updated NGC-CLI to 3.14.0
- Fixed CVE-2022-3515
- Updated HPC SDK to 23.1

Version 22.08.0

- Ubuntu Server 20.04
- NVIDIA Driver 515.65.01
- Docker-ce 20.10.17
- NVIDIA Container Toolkit Version: 1.10.1-1
- NVIDIA Container Runtime Version: 3.10.0-1
- GCP Command Line Interface (CLI)

Key Changes

- Updated NVIDIA Driver to 515.48.07
- Updated Docker-ce to 20.10.17
- Updated NVIDIA Container Toolkit to Version 1.10.0-1
- Updated NVIDIA Container Runtime to Version 3.10.0-1

Known Issues

- The version of Nsight Systems bundled with the HPC SDK 22.7 fails with the error 'Agent launcher failed' on some instance types. The issue is fixed in Nsight Systems version 2022.3.4 and later, which can be installed separately from the Nsight Systems downloads page. For more information, refer to the Nsight Systems documentation.

2.5. NVIDIA Cloud Native Stack VM Image

2.5.1. Information

NVIDIA Cloud Native Stack VMI is a GPU-accelerated VMI that is pre-installed with Cloud Native Stack, which is a reference architecture that includes upstream Kubernetes and the NVIDIA GPU and Network Operator. NVIDIA Cloud Native Stack VMI allows developers to build, test and run GPU-accelerated containerized applications that are orchestrated by Kubernetes.

2.5.2. Release Notes

Version 6.2

- Ubuntu Server 20.04
- Containerd 1.6.5
- Kubernetes 1.23.8
- Helm 3.8.2
- GPU Operator 1.11.0
- NVIDIA Driver 515.65.01

2.6. NVIDIA cuQuantum Appliance VMI

2.6.1. Information

The NVIDIA cuQuantum Appliance is a highly performant multi-GPU multi-node solution for quantum circuit simulation. It contains NVIDIA cuStateVec and cuTensorNet libraries which optimize state vector and tensor network simulation, respectively. The cuTensorNet library functionality is accessible through Python for Tensor Network operations. NVIDIA provides the following simulators with the cuStateVec libraries:

- IBM Qiskit Aer frontend via cusvaer, NVIDIA distributed state vector backend solver.
- Multi-GPU-optimized Google Cirq frontend via qsim, Google state vector simulator.

2.6.2. Release Notes

Version 23.03

- Ubuntu Server 22.04
- NVIDIA Driver 525.105.17
- NVIDIA cuQuantum Appliance Docker Container 23.03
- Docker-ce 24.0.1
- NVIDIA Container Toolkit 1.13.0-1
- GCP CLI, NGC CLI
- Miniconda, JupyterLab (within conda base env), Git: latest

Version 22.11

- Ubuntu Server 20.04
- NVIDIA Driver 525.85.12
- NVIDIA cuQuantum Appliance Docker Container 22.11
- Docker-ce 23.0.1
- NVIDIA Container Toolkit 1.12.0-1
- GCP CLI, NGC CLI
- Miniconda, JupyterLab (within conda base env), Git: latest

Known Issues

- Some instances on Azure, (specifically ND40rs v2) may emit an "unsupported instance type" warning at log-in. This is a known issue and will be addressed in the next release.

Chapter 3. Known Security Vulnerabilities

The NVIDIA GPU-Optimized VMI includes conda by default in order to use jupyter-lab notebooks. The internal Python dependencies may be patched in newer Python versions, but conda must use the specific versions in the VMI. These vulnerabilities are not directly exploitable unless there is a vulnerability in conda itself. An attacker would need to obtain access to a VM running conda, so it is important that VM access must be protected. See the security best practices section.

The following releases are affected by the vulnerabilities:

- NVIDIA GPU-Optimized VMI 22.06
- NVIDIA GPU-Optimized VMI (ARM64) 22.06

The list of vulnerabilities are:

- GHSA-3gh2-xw74-jmcw: High; Django 2.1; SQL injection
- GHSA-6r97-cj55-9hrq: Critical; Django 2.1; SQL injection
- GHSA-c4qh-4vgv-qc6g: High; Django 2.1; Uncontrolled resource consumption
- GHSA-h5jv-4p7w-64jg: High; Django 2.1; Uncontrolled resource consumption
- GHSA-hmr4-m2h5-33qx: Critical; Django 2.1; SQL injection
- GHSA-v6rh-hp5x-86rv: High; Django 2.1; Access control bypass
- GHSA-v9qg-3j8p-r63v: High; Django 2.1; Uncontrolled recursion
- GHSA-vfq6-hq5r-27r6: Critical; Django 2.1; Account hijack via password reset form
- GHSA-wh4h-v3f2-r2pp: High; Django 2.1; Uncontrolled memory consumption
- GHSA-32gv-6cf3-wcmq: Critical; Twisted 18.7.0; HTTP/2 DoS attack
- GHSA-65rm-h285-5cc5: High; Twisted 18.7.0; Improper certificate validation
- GHSA-92x2-jw7w-xvvx: High; Twisted 18.7.0; Cookie and header exposure
- GHSA-c2jg-hw38-jrqq: High; Twisted 18.7.0; HTTP request smuggling
- GHSA-h96w-mmrf-2h6v: Critical; Twisted 18.7.0; Improper input validation
- GHSA-p5xh-vx83-mxcj: Critical; Twisted 18.7.0; HTTP request smuggling
- GHSA-5545-2q6w-2gh6: High; numpy 1.15.1; NULL pointer dereference

- CVE-2019-6446: Critical; numpy 1.15.1; Deserialization of untrusted data
- GHSA-h4m5-qpfp-3mpv: High; Babel 2.6.0; Arbitrary code execution
- GHSA-ffqj-6fqr-9h24: High; PyJWT 1.6.4; Key confusion through non-blocklisted public key formats
- GHSA-h7wm-ph43-c39p: High; Scrapy 1.5.1; Uncontrolled memory consumption
- CVE-2022-39286: High; jupyter_core 4.11.2; Arbitrary code execution
- GHSA-55x5-fj6c-h6m8: High; lxml 4.2.4; Crafted code allowed through lxml HTML cleaner
- GHSA-wrxv-2j5q-m38w: High; lxml 4.2.4; NULL pointer dereference
- GHSA-gpvv-69j7-gwj8: High; pip 8.1.2; Path traversal
- GHSA-hj5v-574p-mj7c: High; py 1.6.0; Regular expression DoS
- GHSA-x84v-xcm2-53pg: High; requests 2.19.1; Insufficiently protected credentials
- GHSA-mh33-7rrq-662w: High; urllib3 1.23; Improper certificate validation
- CVE-2021-33503: High; urllib3 1.23; Denial of service attack
- GHSA-2m34-jcjv-45xf: Medium; Django 2.1; XSS in Django
- GHSA-337x-4q8g-prc5: Medium; Django 2.1; Improper input validation
- GHSA-68w8-qjq3-2gfm: Medium; Django 2.1; Path traversal
- GHSA-6c7v-2f49-8h26: Medium; Django 2.1; Cleartext transmission of sensitive information
- GHSA-6mx3-3vqg-hpp2: Medium; Django 2.1; Django allows unprivileged users can read the password hashes of arbitrary accounts
- GHSA-7rp2-fm2h-wchj: Medium; Django 2.1; XSS in Django
- GHSA-hvmf-r92r-27hr: Medium; Django 2.1; Django allows unintended model editing
- GHSA-wpjr-j57x-wxfw: Medium; Django 2.1; Data leakage via cache key collision in Django
- GHSA-9x8m-2xpf-crp3: Medium; Scrapy 1.5.1; Credentials leakage when using HTTP proxy
- GHSA-cjvr-mfj7-j4j8: Medium; Scrapy 1.5.1; Incorrect authorization and information exposure
- GHSA-jwqp-28gf-p498: Medium; Scrapy 1.5.1; Credential leakage
- GHSA-mfjm-vh54-3f96: Medium; Scrapy 1.5.1; Cookie-setting not restricted
- GHSA-6cc5-2vg4-cc7m: Medium; Twisted 18.7.0; Injection of invalid characters in URI/method
- GHSA-8r99-h8j2-rw64: Medium; Twisted 18.7.0; HTTP Request Smuggling
- GHSA-vg46-2rrj-3647: Medium; Twisted 18.7.0; NameVirtualHost Host header injection
- GHSA-39hc-v87j-747x: Medium; cryptography 37.0.2; Vulnerable OpenSSL included in cryptography wheels

- GHSA-hggm-jpg3-v476: Medium; cryptography 2.3.1; RSA decryption vulnerable to Bleichenbacher timing vulnerability
- GHSA-jq4v-f5q6-mjqq: Medium; lxml 4.2.4; XSS
- GHSA-pgww-xf46-h92r: Medium; lxml 4.2.4; XSS
- GHSA-xp26-p53h-6h2p: Medium; lxml 4.2.4; Improper Neutralization of Input During Web Page Generation in LXML
- GHSA-6p56-wp2h-9hxr: Medium; numpy 1.15.1; NumPy Buffer Overflow, very unlikely to be exploited by an unprivileged user
- GHSA-f7c7-j99h-c22f: Medium; numpy 1.15.1; Buffer Copy without Checking Size of Input in NumPy
- GHSA-fpfv-jqm9-f5jm: Medium; numpy 1.15.1; Incorrect Comparison in NumPy
- GHSA-5xp3-jfq3-5q8x: Medium; pip 8.1.2; Improper Input Validation in pip
- GHSA-w596-4wvx-j9j6: Medium; py 1.6.0; ReDoS in py library when used with subversion
- GHSA-hwfp-hg2m-9vr2: Medium; pywin32 223; Integer overflow in pywin32
- GHSA-r64q-w8jr-g9qp: Medium; urllib3 1.23; Improper Neutralization of CRLF Sequences
- GHSA-wqvq-5m8c-6g24: Medium; urllib3 1.23; CRLF injection

NVIDIA Corporation | 2788 San Tomas Expressway, Santa Clara, CA 95051
www.nvidia.com