NVIDIA Virtual Compute Server for vSphere

Deployment Guide
## Document History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Authors</th>
<th>Description of Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>September 4, 2020</td>
<td>AS, EA</td>
<td>Initial Release</td>
</tr>
</tbody>
</table>
# Table of Contents

Chapter 1. **Executive Summary** ................................................................. 1
  1.1 What is NVIDIA Virtual Compute Server ........................................... 1
  1.2 Why NVIDIA vGPU? ............................................................................ 1
  1.3 NVIDIA vGPU Architecture ............................................................... 2
  1.4 Supported GPUS ................................................................................ 4
  1.5 Virtual GPU Types ............................................................................. 5
  1.6 General Prerequisites ....................................................................... 5
    1.6.1 Server Configuration .................................................................... 6

Chapter 2. **Installing VMware ESXi** ..................................................... 7
  2.1 Choosing the Installation method ...................................................... 7
  2.2 Preparing USB Boot Media ............................................................... 7
  2.3 Installing VMware ESXi .................................................................... 9
  2.4 Initial Host Configuration ................................................................. 13

Chapter 3. **Installing VMware vCenter Server** ................................... 17
  3.1 Installing VCenter Server Appliance ................................................ 17
    3.1.1 About VCSA ............................................................................... 17
    3.1.2 vCenter Server Appliance (VCSA) Installation .......................... 18
  3.2 Post Installation ................................................................................ 28
    3.2.1 Adding Licenses to Your vCenter Server ................................. 29
    3.2.2 Adding a Host ........................................................................... 32
    3.2.3 Setting the NTP Service on a Host ............................................. 35
    3.2.4 Setting a vCenter Appliance to Auto-Start .............................. 36
    3.2.5 Mounting an NFS ISO Data Store ............................................ 38

Chapter 4. **Installing and Configuring the NVIDIA vGPU** .................. 41
  4.1 Uploading VIB in vSphere Web Client .............................................. 41
  4.2 Installing the VIB ............................................................................. 43
  4.3 Updating the VIB ............................................................................. 44
  4.4 Verifying the Installation of the VIB ............................................... 45
  4.5 Uninstalling VIB ............................................................................. 46
  4.6 Changing the Default Graphics Type in VMware vSphere 6.5 and Later ....................... 46
  4.7 Changing the vGPU Scheduling Policy ........................................... 48
    4.7.1 vGPU Scheduling Policies ....................................................... 49
    4.7.2 RmPVMRL Registry Key ......................................................... 49
    4.7.3 Changing the vGPU Scheduling Policy for All GPUs ............... 50
    4.7.4 Changing the vGPU Scheduling Policy for Select GPUs ........... 51
    4.7.5 Restoring Default vGPU Scheduler Settings ............................ 52
4.8 Disabling and Enabling ECC Memory ...................................................................................... 52
4.8.1 Disabling ECC Memory ...................................................................................................... 53
4.8.2 Enabling ECC Memory ....................................................................................................... 55

Chapter 5. Deploying the NVIDIA vGPU Software License Server ........................................ 57
5.1 Platform Requirements ........................................................................................................... 57
5.1.1 Hardware and Software Requirements ............................................................................. 57
5.1.2 Platform Configuration Requirements ............................................................................. 57
5.1.3 Network Ports and Management Interface ...................................................................... 58
5.2 Installing the NVIDIA vGPU Software License Server on Windows ..................................... 58
5.2.1 Installing the Java Runtime Environment on Windows .................................................... 58
5.2.2 Installing the License Server Software on Windows ......................................................... 60
5.2.3 Obtaining the License Server’s MAC Address .................................................................... 63
5.2.4 Managing your License Server and Getting your License Files ......................................... 63
5.2.4.1 Creating a Licenser Server on the NVIDIA Licensing Portal ....................................... 63
5.2.4.2 Downloading a License File ....................................................................................... 65
5.2.5 Installing a License ............................................................................................................ 66

Chapter 6. Creating Your First NVIDIA Virtual Compute Server VM ......................................... 69
6.1 Creating a Virtual Machine ...................................................................................................... 69
6.2 Installing Ubuntu Server 18.04.4 LTS ...................................................................................... 74
6.3 Enabling the NVIDIA vGPU ................................................................................................... 79
6.4 Installing the NVIDIA Driver in the Ubuntu Virtual Machine .............................................. 82
6.5 Licensing an NVIDIA vGPU ................................................................................................... 82

Chapter 7. Selecting the Correct vGPU Profiles ............................................................................. 84
7.1 The Role of the vGPU Manager ............................................................................................... 84
7.2 VGPU Profiles for NVIDIA Virtual Compute Server .......................................................... 84

Chapter 8. GPU Aggregation for NVIDIA Virtual Compute Server ........................................... 86
8.1 Multi vGPU .............................................................................................................................. 86
8.2 Peer-to-Peer NVIDIA NVLINK ............................................................................................... 86

Chapter 9. Page Retirement and ECC .......................................................................................... 89

Chapter 10. Installing Docker and The Docker Utility Engine for NVIDIA GPUs .................. 90
10.1 Enabling the Docker Repository and Installing the NVIDIA Container Toolkit ................ 91
10.2 Testing Docker and NVIDIA Container Run Time ................................................................. 91

Chapter 11. Testing and Benchmarking ......................................................................................... 92
11.1 TensorRT RN50 Inference ...................................................................................................... 92
11.1.1 Commands to the Run Test .............................................................................................. 92
11.1.2 Interpreting the Results ................................................................................................... 93
11.2 TensorFlow RN50 Mixed Training ....................................................................................... 93
11.2.1 Commands to Run the Test .............................................................................................. 93
11.2.2 Interpreting the Results ................................................................................................... 93
Chapter 12. Troubleshooting ................................................................................................................. 94
  12.1 Forums........................................................................................................................................ 94
  12.2 Filing a Bug Report ........................................................................................................................ 94
Appendix A. Using WINSCP to Upload the vGPU Manager VIB to Server Host ......................... 96
Chapter 1. Executive Summary

This document provides insights into how to deploy NVIDIA Virtual Compute Server on VMWare vSphere and serves as a technical resource for understanding system pre-requisites, installation, and configuration.

1.1 What is NVIDIA Virtual Compute Server

NVIDIA Virtual Compute Server enables the benefits of hypervisor-based server virtualization for GPU accelerated servers. Data center admins are now able to power any compute-intensive workload with GPUs in a virtual machine (VM).

NVIDIA Virtual Compute Server software virtualizes NVIDIA GPUs to accelerate large workloads, including more than 600 GPU accelerated applications for AI, deep learning, and HPC. With GPU sharing, multiple VMs can be powered by a single GPU, maximizing utilization and affordability, or a single VM can be powered by multiple virtual GPUs, making even the most intensive workloads possible. With support for all major hypervisor virtualization platforms, including VMWare vSphere, data center admins can use the same management tools for their GPU-accelerated servers as they do for the rest of their data center.

NVIDIA Virtual Compute Server supports NVIDIA NGC GPU-optimized software for deep learning, machine learning, and HPC. NGC software includes containers for the top AI and data science software, tuned, tested, and optimized by NVIDIA, as well as fully tested containers for HPC applications and data analytics. NVIDIA Virtual Compute Server is not tied to a user with a display. It is licensed per GPU as a 1-year subscription with NVIDIA enterprise support included. This allows a number of compute workloads in multiple VMs to be run on a single GPU, maximizing utilization of resources and ROI.

For more information regarding NVIDIA Virtual Compute Server please refer to the NVIDIA Virtual Compute Server Solution Overview.

1.2 Why NVIDIA vGPU?

NVIDIA Virtual Compute Server (NVIDIA vCS) can power the most compute-intensive workloads with virtual GPUs. NVIDIA vCS software is based upon NVIDIA virtual GPU (vGPU) technology and includes the NVIDIA compute driver that is required by compute intensive operations. NVIDIA vGPU enables multiple virtual machines (VMs) to have simultaneous, direct access to a single physical GPU or GPUs can be aggregated within a single VM. vGPU uses the same NVIDIA drivers that are deployed on non-virtualized operating systems. By doing so, NVIDIA vGPU provides VMs with high performance
compute and application compatibility, as well as cost-effectiveness and scalability since multiple VMs can be customized to specific tasks that may demand more or less GPU compute or memory.

With NVIDIA vCS you can gain access to the most powerful GPUs in a virtualized environment and gain vGPU software features such as:

- Management and monitoring – streamline data center manageability by leveraging hypervisor-based tools.
- Live Migration – Live migrate GPU-accelerated VMs without disruption, easing maintenance and upgrades.
- Security – Extend the benefits of server virtualization to GPU workloads.
- Multi-Tenant – Isolate workloads and securely support multiple users.

1.3 NVIDIA vGPU Architecture

The high-level architecture of an NVIDIA virtual GPU enabled VDI environment is illustrated below in Figure 1-1. Here, we have GPUs in the server, and the NVIDIA vGPU manager software (vib) is installed on the host server. This software enables multiple VMs to share a single GPU or if there are multiple GPU’s in the server, they can be aggregated so that a single VM can access multiple GPUs. This GPU enabled environment, provides not only unprecedented performance, it also enables support for more users on a server because work that was typically done by the CPU, can be offloaded to the GPU. Physical NVIDIA GPUs can support multiple virtual GPUs (vGPUs) and be assigned directly to guest VMs under the control of NVIDIA’s Virtual GPU Manager running in a hypervisor.

Guest VMs use the NVIDIA vGPUs in the same manner as a physical GPU that has been passed through by the hypervisor. For NVIDIA vGPU deployments, the NVIDIA vGPU software automatically selects the correct type of license based on the vGPU type assigned.
NVIDIA vGPUs are comparable to conventional GPUs in that they have a fixed amount of GPU-Memory and one or more virtual display outputs or heads. Multiple heads support multiple displays. Managed by the NVIDIA vGPU Manager installed in the hypervisor, the vGPU Memory is allocated out of the physical GPU frame buffer at the time the vGPU is created. The vGPU retains exclusive use of that GPU Memory until it is destroyed.

Note: These are virtual heads, meaning on GPUs there is no physical connection point for external physical displays.

All vGPUs resident on a physical GPU share access to the GPU’s engines, including the graphics (3D) and video decode and encode engines. Figure 1-2 shows the vGPU internal architecture. VM’s guest OS leverages direct access to the GPU for performance and critical fast paths. Non-critical performance management operations use a para-virtualized interface to the NVIDIA Virtual GPU Manager.
1.4 Supported GPUs

NVIDIA virtual GPU software is supported with NVIDIA GPUs. Determine the NVIDIA GPU best suited for your environment based on whether you are optimizing for performance or density, and whether the GPUs will be installed in rack servers or blade servers. Please refer to the NVIDIA vCS solution brief for a full list of recommended and supported GPUs. For a list of certified servers with NVIDIA GPUs, consult the NVIDIA vGPU Certified Servers page. Cross-reference the NVIDIA certified server list with the VMware vSphere HCL to find servers best suited for your NVIDIA vGPU and VMware vSphere environment. Each card requires auxiliary power cables connected to it (except NVIDIA P4 & T4). Most industry standard servers require an enablement kit for proper mounting the of the NVIDIA cards. Check with your server OEM of choice for more specific requirements.

The maximum number of vGPUs that can be created simultaneously on a physical GPU is defined by the amount of GPU memory per VM, and thus how many VMs can share that physical GPU. For example, an NVIDIA GPU which has 24GB of GPU Memory, can support up to six 4C profiles (24 GB total with 4GB per VM). You cannot oversubscribe GPU memory and it must be shared equally for...
each physical GPU. If you have multiple GPUs inserted within the server, you have the flexibility to carved up each physical GPU appropriately to meet your users demands.

1.5 Virtual GPU Types

vGPU types have a fixed amount of GPU memory, number of supported display heads, and maximum resolutions. They are grouped into different series according to the different classes of workload for which they are optimized. Each series is identified by the last letter of the vGPU type name.

<table>
<thead>
<tr>
<th>Series</th>
<th>Optimal Workload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q-series</td>
<td>Virtual workstations for creative and technical professionals who require the performance and features of Quadro technology</td>
</tr>
<tr>
<td>C-series</td>
<td>Compute-intensive server workloads, such as artificial intelligence (AI), deep learning, or high-performance computing (HPC)</td>
</tr>
<tr>
<td>B-series</td>
<td>Virtual desktops for business professionals and knowledge workers</td>
</tr>
<tr>
<td>A-series</td>
<td>App streaming or session-based solutions for virtual applications users</td>
</tr>
</tbody>
</table>

NVIDIA vCS use the C-Series vGPU profiles. Please refer to the NVIDIA vCS solution brief for more information regarding the available profiles.

1.6 General Prerequisites

Prior to installing and configuring vGPU software for NVIDIA vCS it is important to document an evaluation plan. This can consist of all the following:

- List of your business drivers and goals
- List of all the user groups, their workloads, and applications with current, and future projections in consideration
- Current end-user experience measurements and analysis
- ROI / Density goals

NVIDIA vGPU technical documentation contains vGPU sizing guides that can also assist you in understanding how deploy to best practices, run a proof of concept, as well as leverage management and monitoring tools.

If you are new to virtualization it is also recommended to review VMware’s ESXi Getting Started which includes courses and guidance on potentially any current configuration that you may already have.

The following elements are required to install and configure vGPU software on VMware ESXi.

- NVIDIA certified servers with NVIDIA GPUs (2.6GHz CPU or faster (Intel Xeon E5-2600 v4, Intel Xeon Scalable Processor Family)
  - High-speed RAM
  - Fast networking
  - If using local storage IOPS plays a major role in performance. If using VMware for Virtual SAN, see the VMware Virtual SAN requirements website for more details.
• Intel Xeon E5-2600 v4, Intel Xeon Scalable Processor Family Higher-performance end points for testing access

- Select the appropriate NVIDIA GPU for your use case. Please refer to the NVIDIA vCS solution brief for a full list of recommended and supported GPUs.

  - vGPU license (free evaluation is available here)
  - VMware ESXi and vCenter Server. For a list of supported VMware vSphere versions, please refer to the vGPU software documentation.

You may deploy vCenter Server on a Windows server or as an OVA Appliance.

- VMware Horizon software (free evaluation is available here)

- NVIDIA vGPU software:
  - NVIDIA vGPU manager VIB
  - NVIDIA WDDM guest driver

  Note: The vGPU Manager VIB is loaded like a driver in the vSphere hypervisor, and is then managed by the vCenter Server.

For testing and benchmarking you may leverage the NVIDIA System Management interface (NV-SMI) management and monitoring tool.

### 1.6.1 Server Configuration

The following server configuration details are considered best practices:

- Hyperthreading – Enabled
- Power Setting or System Profile— High Performance
- CPU Performance (if applicable) – Enterprise or High Throughput
- Memory Mapped I/O above 4-GB - Enabled (if applicable)

  Note: If NVIDIA card detection does not include all the installed GPUs, set this option to Enabled.
Chapter 2. Installing VMware ESXi

This chapter covers the following VMware ESXi installation topics:

- Choosing the Installation method
- Preparing USB Boot Media
- Installing VMware ESXi
- Initial Host Configuration

Note: This deployment guide assumes you are building an environment as a proof of concept and is not meant to be a production deployment, as a result, choices made are meant to speed up and ease the process. See the corresponding guides for each technology, and make choices appropriate for your needs, before building your production environment.

For the purpose of this guide, ESXi 6.7 U3 is used as the hypervisor version.

2.1 Choosing the Installation method

With the ability to install from and onto a SD card or USB memory stick, ESXi offers flexibility verses local hard drive install. Please see vSphere documentation regarding best practices for logs when booting from USB or similar. In our main lab we used Supermicro’s IPMI and virtual media to boot from ISO file and install on local storage. In home labs USB was used to quickly move from one version to another.

2.2 Preparing USB Boot Media

For more information, see the VMware knowledgebase article Installing ESXi on a supported USB flash drive or SD flash card (2004784).

Booting ESXi from a USB drive is useful if your host has an existing ESXi Version 6.X or earlier installation that you want to retain.

Use the following procedure to prepare a USB drive for booting:


The Windows version of the application does not include an installer; however, the OSX version is packaged in a .DMG file that you must mount. You must also copy the application to the Applications folder before launching. Alternatively, you can use YUMI, which allows booting multiple installation images on one USB device plus the option to load the entire installation into RAM. The download link is http://www.pendrivelinux.com/yumi-multiboot-usb-creator/.
Start the application, select **Diskimage**, and then click the ... icon to browse for the installation **.ISO** file.

Navigate to the location that contains the installation **.ISO** file and then select **Open**. Select the mounted USB drive on which to perform the installation and then select **OK**.

The copying process begins, and a series of progress bars are displayed.

When the copying process is complete, click **Exit** and then remove the USB drive.

To install from this USB drive, insert into the host using either an internal or on motherboard USB port, then set that as the primary boot source or select from the boot menu on power up.
2.3 Installing VMware ESXi

Use the following procedure to install VMWare ESXi regardless of boot source. Select the boot media with the ESXi ISO on your host’s boot menu.

1. Apply power to start the host.

The following menu displays when the host starts up.

Select the installer using the arrow keys and then press [ENTER] to begin booting the ESXi installer.

A compatibility warning is displayed.

Press [ENTER] to proceed.

The End User License Agreement (EULA) displays.
Installing VMware ESXi

Read the EULA and then press [F11] to accept it and continue the installation.

The installer scans the host to locate a suitable installation drive.

It should display all drives available for install.

Use the arrow keys to select the drive you want to install ESXi and then press [ENTER] to continue.

Note: You can install ESXi to a USB drive and then boot and run the system from that USB drive. This sample installation shows ESXi being installed on a local hard drive.

The installer scans the chosen drive to determine suitability for install.
Installing VMware ESXi

The Confirm Disk Selection window displays.

Press [ENTER] to accept your selection and continue. (For this EA2 release, Upgrade ESXi is not a supported selection.)

The Please select a keyboard layout window displays.

Select your desired keyboard layout using the arrow keys and then press [ENTER].

The **Enter a root password** window displays.
Enter a root password in the Root password field.

**CAUTION:** To prevent unauthorized access, your selected root password should contain at least eight (8) characters and consist of a mix of lowercase and capital letters, digits, and special characters.

Confirm the password in the **Confirm password** field and then press **[ENTER]** to proceed.

The installer rescans the system.

It then displays the **Confirm Install** window.

Press **[F11]** to proceed with the installation.

**CAUTION:** The installer will repartition the selected disk. All data on the selected disk will be destroyed.

The ESXi installation proceeds.

The **Installation Complete** window displays when the installation process is completed.
Press **[ENTER]** to reboot the system. (Make sure your installation media has been ejected and your bios set to the boot disk.)

The installation is now complete.

### 2.4 Initial Host Configuration

A countdown timer displays when you first boot ESXi. You can wait for the countdown to expire or press **[ENTER]** to proceed with booting. A series of notifications displays during boot.

The VMware ESXi screen displays when the boot completes.
Use the following procedure to configure the host:

1. Press [F2].

   The Authentication Required window displays.

   ![Authentication Required Window]

   Enter the root account credentials that you created during the installation process and then press [ENTER].

   The System Customization screen displays.
Scroll down to select **Configure Management Network** and then press [ENTER].

The **Network Adapters** window appears.

Use the arrow keys to select the adapter to use as the default management network and then press [ENTER].

The IPv4 Configuration window displays.

Use the arrow keys to select **Set static IPv4 address and network configuration** and then enter the IPv4 address, subnet mask, and default gateway in the respective fields.

Press [ENTER] when finished to apply the new management network settings.

The Confirm Management Network popup displays.

Press [Y] to confirm your selection.

The **DNS Configuration** window displays.
Add the primary and (if available) secondary DNS server address(es) in the respective fields.
Set the host name for this ESXi host in the **Hostname** field.
Press **[ENTER]** when finished.

Select **Test Management Network** on the main ESXi screen to open the **Test Management Network** window.

Perform the following tests:
- Ping the default gateway.
- Ping the DNS server.
- Resolve a known address.

Return to the main ESXi screen when you have completed testing, and then select **Troubleshooting Options**.

The Troubleshooting Mode Options window displays.

To install the NVIDIA VIB in a later step, you will need to enable the ESXi shell. This can be accomplished by selecting **Enable ESXi Shell**.

Press **[ENTER]** to toggle **Enable ESXi Shell** on.

The window on the right displays the status: **Enable ESXi Shell Disabled**.

Enable SSH by selecting **Enable SSH** and press **[ENTER]** to toggle this option on.

The window on the right displays the status: **SSH is Enabled**.
Chapter 3. Installing VMware vCenter Server

This chapter covers installing VMware vCenter Server, including:

- Installing VCenter Server Appliance
- Adding Licenses to Your vCenter Server
- Adding a Host
- Setting the NTP Service on a Host
- Setting a vCenter Appliance to Auto-Start
- Mounting an NFS ISO Data Store

Review the prerequisites in General Prerequisites on page 5 before proceeding with these installations.

Note: This deployment guide assumes you are building an environment for a proof of concept. Refer to VMware best practice guides before building your production environment.

3.1 Installing VCenter Server Appliance

3.1.1 About VCSA

The VCSA is a pre-configured virtual appliance built on Project Photon OS. Since the OS has been developed by VMware it benefits from enhanced performance and boot times over the previous Linux based appliance. Furthermore, the embedded vPostgres database means VMware have full control of the software stack, resulting in significant optimization for ESXi environments and quicker release of security patches and bug fixes. The VCSA scales up to 2000 hosts and 35,000 virtual machines. A couple of releases ago the VCSA reached feature parity with its Windows counterpart and is now the preferred deployment method for vCenter Server. Features such as Update Manager are bundled into the VCSA, as well as file-based backup and restore, and vCenter High Availability. The appliance also saves operating system license costs and is quicker and easier to deploy and patch.

Software Considerations

- VCSA must be deployed to an ESXi host or vCenter running v5.5 or above. However, all hosts you intend to connect to vCenter Server should be running ESXi 6.0 or above, hosts running 5.5 and earlier cannot be managed by vCenter and do not have a direct upgrade path to.
- You must check compatibility of any third-party products and plugins that might be used for backups, anti-virus, monitoring, etc. as these may need upgrading for ESXi compatibility.
To check version compatibility with another VMware products, see the Product Interoperability Matrix.

Architectural Considerations

When implementing a new ESXi environment you should plan your topology in accordance with the VMware vCenter Server and Platform Services Controller Deployment Types.

Most deployments will include the vCenter Server and PSC in one appliance, following the embedded deployment model, which is used in this guide.

Other Considerations

The VCSA with embedded PSC requires the following hardware resources (disk can be thin provisioned)

- Tiny (up to 10 hosts, 100 VMs) – 2 CPUs, 10 GB RAM.
- Small (up to 100 hosts, 1000 VMs) – 4 CPUs, 16 GB RAM.
- Medium (up to 400 hosts, 4000 VMs) – 8 CPUs, 24 GB RAM.
- Large (up to 1000 hosts, 10,000 VMs) – 16 CPUs, 32 GB RAM.
- X-Large (up to 2000 hosts, 35,000 VMs) – 24 CPUs, 48 GB RAM – new to v6.5.

Storage requirements for the smallest environments start at 250 GB and increase depending on your specific database requirements. See the Storage Requirements document for further details.

Where the PSC is deployed as a separate appliance this requires 2 CPUs, 4 GB RAM, 60 GB disk.

Environments with ESXi host(s) with more than 512 LUNs and 2048 paths should be sized large or x large.

The ESXi host on which you deploy the VCSA should not be in lockdown or maintenance mode.

All ESXi components should be configured to use an NTP server. The installation can fail or the vCenter Server Appliance vpxd service may not be able to start if the clocks are unsynchronized.

FQDN resolution should be in place when deploying vCenter Server.

A list of Required Ports for vCenter Server and PSC can be found here.

The configuration maximums for ESXi can be found here.

3.1.2 vCenter Server Appliance (VCSA) Installation

Download the VMware vCenter Server Appliance ISO from VMware downloads: v6.7.0.

1. Mount the ISO on your computer. The VCSA installer is compatible with Mac, Linux, and Windows.
2. Browse to the corresponding directory for your operating system, e.g. \vcsa-ui-installer\win32. Right click Installer and select Run as administrator.
3. As we are installing a new instance click **Install**.

4. The installation is split into 2 stages, we begin with deploying the appliance. Click **Next**.
5. Read and accept the EULA, and then click [Next] to continue.

6. Select the deployment model. In this example, we will be using an embedded deployment combining the vCenter Server and Platform Services Controller in one appliance. Click Next.
7. In this step you are selecting the ESXi host to install the VCSA on as a guest, which can be on a host running ESXi 5.5 or later. It is recommended that the vCenter server (Windows or appliance based) run on a separate management cluster from the one designated for VDI workloads. Enter the IP address or Fully Qualified Domain Name (FQDN) of the chosen host, then its root username and password and click [Next].
8. If your desktop can reach the host, you should see a certificate warning as it connects. This warning is due to the use of a self-signed certificate. If you are using signed certificate, you will not see this warning. Click [Yes] to continue.

![Certificate Warning]

The credentials you provided are validated:

![Validation]

9. When prompted after a successful connection, name the appliance, enter a root password for the appliance, enter the root password again, and click **Next**.
10. Select the deployment size in line with the number of hosts and virtual machines that will be managed and click Next.

11. Select the datastore where the VCSA will be deployed, select thin provisioning if required, and click Next.
12. **Configure the network settings for the appliance and click Next.**

The Configure Network Settings page is a long page and will require scrolling down to see all settings. Before configuring these settings, choose an appropriate static IP address and enter it into local DNS (for example, on the Domain Controller). After you can resolve the address, enter that IP address, host name, and then scroll down for remaining entries:

13. **On the summary page click Finish.**

The appliance will now be deployed.
14. With the VCSA now deployed move on to stage 2 by clicking Continue.

15. Click Next to begin the VCSA setup.
16. Configure the NTP servers, enable SSH access if required, and click **Next**.

17. Enter a unique SSO domain name, configure a password for the SSO administrator, click **Next**. The default SSO domain name is vSphere.local. The SSO domain name should not be the same as your Active Directory Domain.
18. Select or deselect the customer experience improvement program box and click **Next**.

19. Review the details on the summary page and click **Finish**.
20. Click **OK** to acknowledge that the VCSA setup cannot be paused or stopped after it is started. When the installer is complete click **Close** to close the wizard.

3.2 Post Installation

This section describes post install and configure vCenter Server.
3.2.1 Adding Licenses to Your vCenter Server

Use the following procedure to configure vCenter:

1. Connect to the vCenter post install using the IP or FQDN of the vCenter. Access vSphere by clicking either **Launch vSphere Client (HTML5)** or **Launch vSphere Web Client (FLEX)**. As the web client will be depreciated in future versions, and the HTML5 client is now nearly at full feature parity, we will use the HTML5 vSphere client.

   ![VMware Single Sign-On page](image)

   The VMware Single Sign-On page displays.

2. Enter the username and password that you specified during installation, and then click the Login button.

   The VMware vSphere Web Client page displays.
3. You must apply a new vCenter license key within 60 days. If you have purchased vCenter Server then log into your licensing portal here. If the license key does not appear then check with your VMware account manager. Log in to the vSphere Web Client using the SSO administrator login. From the Menu drop-down click Administration.

4. Select Licenses from the left-hand menu and then select the Licenses tab to open the Licenses tab. Click Add New Licenses to open the New Licenses popup.
5. Enter the vCenter Server Standard license key provided at the vSphere beta program website.

6. Enter a unique name for the license in the License Name field and then click **Next**.

7. Review your selections and then click **Finish** to close the Enter New License popup and return to the VMware vSphere Web Client page.
3.2.2 Adding a Host

Use the following procedure to add a host in vCenter:

1. Select the **Home** icon (house) on the **VMware vSphere Web Client** page.
2. Select **Hosts** and **Clusters**.
3. From the **ACTIONS** drop-down list, select **New Datacenter**.

The New Datacenter popup displays.
4. Enter a name for the datacenter in the **Datacenter name** field and click **OK**. The new datacenter is visible in the left panel of the **vSphere Web Client**.

5. Drop down **ACTIONS** and select **Add a Host**. The **Name and location** dialog box opens.
6. Enter the host name or IP address of the vSphere host and click **Next**.  
   The **Connection settings** dialog box displays.

7. Enter the administrator account credentials in the **Username** and **Password** fields and click **Next**.  
   The **Security Alert** popup displays.

8. Click **Yes** to replace the host certificate.  
   The **Host summary** dialog displays.

9. Review the settings and click **Next** to proceed.  
   The **Assign license** dialog displays.

10. Confirm the license selection and click **Next**.  
    The **Lockdown mode** dialog displays.

11. Accept the default setting (Disabled) and click **Next**.  
    The **VM location** dialog displays.

12. Select a cluster or accept the default option and click **Next** to proceed.  
    The **Ready to complete** dialog displays.
13. Click **Finish** to complete adding the new host.

The new host is now visible in the left panel when you click the datacenter name.

3.2.3 Setting the NTP Service on a Host

Set the NTP service on each host to ensure time is accurate for all guests.

1. Click a host object in the menu on the left, click **Configure > System > Time Configuration > Edit**.

   Enter a valid time server and click **OK**.
3.2.4 Setting a vCenter Appliance to Auto-Start

Use the following procedure to set a vCenter Appliance to start automatically:

1. In the vSphere Web Client, select the host then select **Configure > Virtual Machines > VM Startup/Shutdown**.

   ![Edit VM Startup and Shutdown window](image)

   Click the **Edit** button.

   The Edit VM Startup and Shutdown window displays.
Select the **vCenter Appliance** and click the **Up** arrow to move that virtual machine up to the **Automatic Startup** section. Click the **Edit** button.
Select the following options:

- Set Startup Behavior to Use specified settings and select Continue immediately if VMware Tools starts
- Set Startup Delay to 0
- Set Shutdown Behavior to Use specified settings
- Set Shutdown Delay to 0
- Select Guest Shutdown

Click OK to apply the configuration.

Note: The vCenter Web Client may not reflect these configuration changes immediately. Either click the Refresh icon or different configuration group and return to the current setting.

3.2.5 Mounting an NFS ISO Data Store

Use the following procedure to mount an NFS ISO data store:

1. In the main vSphere Web Client window, select Hosts and Clusters and select the host.
   Select Storage -> New Datastore from the Actions drop-down menu.

   The New Datastore window displays with the Type tab selected.
Select **NFS** and click **Next** to proceed.

The **Select NFS version** tab displays.
Select the correct NFS version and click **Next** to proceed.

The **Name and configuration** tab displays.
Enter the NFS exported folder path and the NFS server address in the **Folder** and **Address** fields, respectively.

Because the data store is an ISO data store, consider mounting it as read-only by checking the **Mount NFS** as read-only checkbox.
Click **Next** to proceed.

The **Host accessibility** tab displays.
Select the host that will use the new data store.
Select **Next** to proceed.

The **Ready to complete** tab displays.
Review the settings.

Click Finish to complete adding the NFS ISO data store.

This data store is now accessible as an installation source for virtual machine CD drives.
Chapter 4. Installing and Configuring the NVIDIA vGPU

This chapter covers installing and configuring the NVIDIA vGPU Manager:

- Uploading VIB in vSphere Web Client
- Installing the VIB
- Updating the VIB
- Verifying the Installation of the VIB
- Uninstalling VIB
- Changing the Default Graphics Type in VMware vSphere 6.5 and Later
- Changing the vGPU Scheduling Policy

4.1 Uploading VIB in vSphere Web Client

For demonstration purposes, these steps use the VMWare vSphere web interface for uploading the VIB to the server host.

Before you begin, download the archive containing the VIB file and extract the contents of the archive to a folder. The file ending with VIB is the file that you must upload to the host data store for installation.

To upload the file to the data store using vSphere Web Client:

1. Click the Related Objects tab for the desired server.
2. Select Datastores.
3. Either right click the data store and then select Browse Files or click the icon in the toolbar.
   The Datastore Browser window displays.
4. Click the New Folder icon.

The Create a new folder window displays.

5. Name the new folder vib and then click Create.

6. Select the vib folder in the Datastore Browser window.

Click the Upload icon.

The Client Integration Access Control window displays.

Select Allow.

The .VIB file is uploaded to the data store on the host.

Note: If you do not click Allow before the timer runs out, then further attempts to upload a file will silently fail. If this happens, exit and restart vSphere Web Client. Repeat this procedure and be sure to click Allow before the timer runs out.
## 4.2 Installing the VIB

The NVIDIA Virtual GPU Manager runs on the ESXi host. It is provided in the following formats:

- As a VIB file, which must be copied to the ESXi host and then installed
- As an offline bundle that you can import manually as explained in [Import Patches Manually](#) in the VMware vSphere documentation

**CAUTION:** Prior to vGPU software release 11, NVIDIA Virtual GPU Manager and Guest VM drivers must be matched from the same main driver branch. If you update vGPU Manager to a release from another driver branch, guest VMs will boot with vGPU disabled until their guest vGPU driver is updated to match the vGPU Manager version. Consult Virtual GPU Software for VMware vSphere Release Notes for further details.

To install the vGPU Manager VIB you need to access the ESXi host via the ESXi Shell or SSH. Refer to VMware’s documentation on how to enable ESXi Shell or SSH for an ESXi host.

**Note:** Before proceeding with the vGPU Manager installation make sure that all VMs are powered off and the ESXi host is placed in maintenance mode. Refer to VMware’s documentation on how to place an ESXi host in maintenance mode.

1. Place the host into Maintenance mode by right-clicking it and then selecting **Maintenance Mode - Enter Maintenance Mode**.

```
Note: Alternatively, you can place the host into Maintenance mode using the command prompt by entering
$ esxcli system maintenanceMode set -- enable=true
This command will not return a response. Making this change using the command prompt will not refresh the vSphere Web Client UI. Click the Refresh icon in the upper right corner of the vSphere Web Client window.

**CAUTION:** Placing the host into maintenance mode disables any vcenter appliance running on this host until you exit maintenance mode and then restart that vcenter appliance.
```

2. Click **OK** to confirm your selection.

3. Use the esxcli command to install the vGPU Manager package:
Installing and Configuring the NVIDIA vGPU

[root@esxi:~] esxcli software vib install -v directory/NVIDIA-vGPU-
VMware_ESXi_6.0_Host_Driver_390.72-1OEM.600.0.0.2159203.vib

Installation Result  Message: Operation finished successfully.
   Reboot Required: false
   VIBs Installed: NVIDIA-vGPU-VMware_ESXi_6.0_Host_Driver_390.72-
1OEM.600.0.0.2159203
   VIBs Removed:
   VIBs Skipped:

The directory is the absolute path to the directory that contains the VIB file. You must specify the absolute path even if the VIB file is in the current working directory.

4. Reboot the ESXi host and remove it from maintenance mode.

   Note: Although the display states “Reboot Required: false”, a reboot is necessary for the vib to load and xorg to start.

5. From the vSphere Web Client, exit Maintenance Mode by right clicking the host and selecting Exit Maintenance Mode.

   Note: Alternatively, you may exit from Maintenance mode via the command prompt by entering:
   $ esxcli system maintenanceMode set -- enable=false
   This command will not return a response.
   Making this change via the command prompt will not refresh the vSphere Web Client UI. Click the Refresh icon in the upper right corner of the vSphere Web Client window.

6. Reboot the host from the vSphere Web Client by right clicking the host and then selecting Reboot.

   Note: You can reboot the host by entering the following at the command prompt:
   $ reboot
   This command will not return a response. The Reboot Host window displays.

7. When rebooting from the vSphere Web Client, enter a descriptive reason for the reboot in the Log a reason for this reboot operation field, and then click OK to proceed.

4.3 Updating the VIB

Update the vGPU Manager VIB package if you want to install a new version of NVIDIA Virtual GPU Manager on a system where an existing version is already installed.

To update the vGPU Manager VIB you need to access the ESXi host via the ESXi Shell or SSH. Refer to VMware’s documentation on how to enable ESXi Shell or SSH for an ESXi host.

   Note: Before proceeding with the vGPU Manager update, make sure that all VMs are powered off and the ESXi host is placed in maintenance mode. Refer to VMware’s documentation on how to place an ESXi host in maintenance mode.

Use the esxcli command to update the vGPU Manager package:

   [root@esxi:~] esxcli software vib update -v directory/NVIDIA-vGPU-
VMware_ESXi_6.0_Host_Driver_390.72-1OEM.600.0.0.2159203.vib

Installation Result  Message: Operation finished successfully.
   Reboot Required: false
   VIBs Installed: NVIDIA-vGPU-
8. Reboot the ESXi host and remove it from maintenance mode.

### 4.4 Verifying the Installation of the VIB

After the ESXi host has rebooted, verify the installation of the NVIDIA vGPU software package.

1. Verify that the NVIDIA vGPU software package installed and loaded correctly by checking for the NVIDIA kernel driver in the list of kernels loaded modules.

   ```
   [root@esxi:~] vmkload_mod -l | grep nvidia
   nvidia                   5 8420
   8420
   ```

2. If the NVIDIA driver is not listed in the output, check dmesg for any load-time errors reported by the driver.

3. Verify that the NVIDIA kernel driver can successfully communicate with the NVIDIA physical GPUs in your system by running the `nvidia-smi` command.

   The `nvidia-smi` command is described in more detail in NVIDIA System Management Interface `nvidia-smi`.

   Running the `nvidia-smi` command should produce a listing of the GPUs in your platform.

   ```
   [root@esxi:~] nvidia-smi
   Fri Jul 20 17:56:22 2018
   +-----------------------------------------------------------------------------+
   | NVIDIA-SMI 390.72     Driver Version: 390.75     |-------------------------------+----------------------+----------------------+
   | GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |-------------------------------+----------------------+----------------------+
   | Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |-------------------------------+----------------------+----------------------+
   |===============================+======================+======================|===============================+----------------------+----------------------+
   |   0  M60           On         | 0000:85:00.0     Off |                  Off |                           +-------------------------------+----------------------+----------------------+
   | N/A   23C    P8    23W / 150W |     13MiB /  8191MiB |      0%      Default |                           +-------------------------------+----------------------+----------------------+
   +-----------------------------------------------------------------------------+
   | Processes:                                                       GPU Memory |
   | GPU       PID  Type  Process name                               Usage      |
   |=============================================================================|
   | No running processes found                                                 |
   +-----------------------------------------------------------------------------+
   ```
If nvidia-smi fails to report the expected output for all the NVIDIA GPUs in your system, see NVIDIA Virtual GPU Software User Guide for troubleshooting steps.

The NVIDIA System Management Interface nvidia-smi also allows GPU monitoring using the following command:

$ nvidia-smi -l

This command switch adds a loop, automatically refreshing the display. The default refresh interval is 1 second.

### 4.5 Uninstalling VIB

1. Determine the name of the vGPU driver bundle.

   $ esxcli software vib list | grep -i nvidia

   This command returns output similar to the following:

   NVIDIA-VMware_ESXi_6.7_Host_Driver 390.72-1OEM.600.0.0.2159203
   NVIDIA VMwareAccepted 2018-07-20

2. Run the following command to uninstall the driver package:

   $ esxcli software vib remove -n NVIDIA-VMware_ESXi_6.7_Host_Driver
   --maintenance-mode

   The following message displays when installation is successful:

   Removal Result
   - Message: Operation finished successfully.
   - Reboot Required: false
   - VIBs Installed:
     - NVIDIA_Bootbank_NVIDIA-VMware_ESXi_6.7_Host_Driver_390.72-1OEM.600.0.0.2159203
   - VIBs Skipped:

3. Reboot the host to complete the uninstallation process.

### 4.6 Changing the Default Graphics Type in VMware vSphere 6.5 and Later

The vGPU Manager VIBs for VMware vSphere 6.5 and later provide vSGA and vGPU functionality in a single VIB. After this VIB is installed, the default graphics type is Shared, which provides vSGA functionality. To enable vGPU support for VMs in VMware vSphere 6.5, you must change the default graphics type to Shared Direct. If you do not change the default graphics type, VMs to which a vGPU is assigned fail to start and the following error message is displayed:

The amount of graphics resource available in the parent resource pool is insufficient for the operation.

Note: If you are using a supported version of VMware vSphere earlier than 6.5, or are configuring a VM to use vSGA, omit this task.
Change the default graphics type before configuring vGPU. Output from the VM console in the VMware vSphere Web Client is not available for VMs that are running vGPU.

Before changing the default graphics type, ensure that the ESXi host is running and that all VMs on the host are powered off.

1. Log in to vCenter Server by using the vSphere Web Client.
2. In the navigation tree, select your ESXi host and click the Configure tab.
3. From the menu, choose Graphics and then click the Host Graphics tab.

5. In the Edit Host Graphics Settings dialog box that opens, select Shared Direct and click OK.

Note: In this dialog box, you can also change the allocation scheme for vGPU-enabled VMs. For more information, see Modifying GPU Allocation Policy on VMware vSphere.
After you click **OK**, the default graphics type changes to Shared Direct.

6. Restart the ESXi host or stop and restart the Xorg service and nv-hostengine on the ESXi host.

To stop and restart the Xorg service and nv-hostengine, perform these steps:

a). Stop the Xorg service.

```bash
[root@esxi:~] /etc/init.d/xorg stop
```

b). Stop nv-hostengine.

```bash
[root@esxi:~] nv-hostengine -t
```

c). Wait for 1 second to allow nv-hostengine to stop.


```bash
[root@esxi:~] nv-hostengine -d
```

e). Start the Xorg service.

```bash
[root@esxi:~] /etc/init.d/xorg start
```

After changing the default graphics type, configure vGPU as explained in [Configuring a vSphere VM with Virtual GPU](https://example.com).

See also the following topics in the VMware vSphere documentation:

- [Log in to vCenter Server by Using the vSphere Web Client](https://example.com)
- [Configuring Host Graphics](https://example.com)

### 4.7 Changing the vGPU Scheduling Policy

GPUs, starting with the NVIDIA Maxwell™ graphic architecture, implement a best effort vGPU scheduler that aims to balance performance across vGPUs. The best effort scheduler allows a vGPU to use GPU processing cycles that are not being used by other vGPUs. Under some circumstances, a VM running a graphics-intensive application may adversely affect the performance of graphics-light applications running in other VMs.

GPUs, starting with the NVIDIA Pascal™ architecture, also support equal share and fixed share vGPU schedulers. These schedulers impose a limit on GPU processing cycles used by a vGPU which prevents graphics-intensive applications running in one VM from affecting the performance of graphics-light applications running in other VMs. The best effort scheduler is the default scheduler for all supported GPU architectures.

The GPUs that are based on the Pascal architecture are the NVIDIA P4, NVIDIA P6, NVIDIA P40, and NVIDIA P100.

The GPUs that are based on the Volta™ architecture are the NVIDIA V100 SXM2, NVIDIA V100 PCIe, NVIDIA V100 FHHL, and NVIDIA V100s.

The GPUs that are based on the Turing™ architecture are the NVIDIA T4, RTX6000 and RTX8000.

The GPU that is based on the Ampere™ architecture is the NVIDIA A100.

1 Support is coming in an upcoming release.
4.7.1 vGPU Scheduling Policies

In addition to the default best effort scheduler, GPUs based on the Pascal and Volta architectures support equal share and fixed share vGPU schedulers.

Equal Share Scheduler

The physical GPU is shared equally amongst the running vGPUs that reside on it. As vGPUs are added to or removed from a GPU, the share of the GPU’s processing cycles allocated to each vGPU changes accordingly. As a result, the performance of a vGPU may increase as other vGPUs on the same GPU are stopped or decrease as other vGPUs are started on the same GPU.

Fixed Share Scheduler

Each vGPU is given a fixed share of the physical GPU’s processing cycles, the amount of which depends on the vGPU type. As vGPUs are added to or removed from a GPU, the share of the GPU’s processing cycles allocated to each vGPU remains constant. As a result, the performance of a vGPU remains unchanged as other vGPUs are stopped or started on the same GPU.

4.7.2 RmPVMRL Registry Key

The RmPVMRL registry key sets the scheduling policy for NVIDIA vGPUs.

Note: You can change the vGPU scheduling policy only on GPUs based on the Pascal and Volta architectures.

<table>
<thead>
<tr>
<th>Type</th>
<th>Dword</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Meaning</td>
</tr>
<tr>
<td>0x00 (default)</td>
<td>Best effort scheduler</td>
</tr>
<tr>
<td>0x01</td>
<td>Equal share scheduler with the default time slice length</td>
</tr>
<tr>
<td>0x00TT0001</td>
<td>Equal share scheduler with a user-defined time slice length TT</td>
</tr>
<tr>
<td>0x11</td>
<td>Fixed share scheduler with the default time slice length</td>
</tr>
<tr>
<td>0x00TT0011</td>
<td>Fixed share scheduler with a user-defined time slice length TT</td>
</tr>
</tbody>
</table>

Examples

The default time slice length depends on the maximum number of vGPUs per physical GPU allowed for the vGPU type.
4.7.3 Changing the vGPU Scheduling Policy for All GPUs

Note: You can change the vGPU scheduling policy only on GPUs based on the Pascal, Volta, Turing, and Ampere architectures.

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 secure shell (SSH) for this purpose. Set the RmPVMRL registry key to the value that sets the GPU scheduling policy that you want.

   ```bash
   # esxcli system module parameters set -m nvidia -p "NVreg_RegistryDwords=RmPVMRL=value"
   ```

   **Value** - The value that sets the vGPU scheduling policy that you want, for example:
   - **0x01** - Sets the vGPU scheduling policy to Equal Share Scheduler.
   - **0x11** - Sets the vGPU scheduling policy to Fixed Share Scheduler.
   - For all supported values, see RmPVMRL Registry Key.

2. Reboot your hypervisor host machine.
4.7.4 Changing the vGPU Scheduling Policy for Select GPUs

Note: You can change the vGPU scheduling policy only on GPUs based on the Pascal, Volta, Turing, and Ampere architectures.

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 secure shell (SSH) for this purpose.
2. Use the lspci command to obtain the PCI domain and bus/device/function (BDF) of each GPU for which you want to change the scheduling behavior.

On VMware vSphere, pipe the output of lspci to the grep command to display information only for NVIDIA GPUs.

# lspci | grep NVIDIA

The NVIDIA GPUs listed in this example have the PCI domain 0000 and BDFs 85:00.0 and 86:00.0.

1. 0000:85:00.0 VGA compatible controller: NVIDIA Corporation GM204GL [M60] (rev a1)
2. 0000:86:00.0 VGA compatible controller: NVIDIA Corporation GM204GL [M60] (rev a1)
3. Use the module parameter NVreg_RegistryDwordsPerDevice to set the pci and RmPVMRL registry keys for each GPU.

On VMware vSphere, use the esxcli set command.

# esxcli system module parameters set -m nvidia \
-p "NVreg_RegistoryDwordsPerDevice=pci=pci-domain:pci-bdf;RmPVMRL=value\n[;pci=pci-domain:pci-bdf;RmPVMRL=value...]"

For each GPU, provide the following information:

- **pci-domain**
  - The PCI domain of the GPU.
- **pci-bdf**
  - The PCI device BDF of the GPU.
- **value**
  - The value that sets the vGPU scheduling policy that you want, for example:
    - **0x01** - Sets the GPU scheduling policy to Equal Share Scheduler.
    - **0x11** - Sets the GPU scheduling policy to Fixed Share Scheduler.

For all supported values, see [RmPVMRL Registry Key](#).
Installing and Configuring the NVIDIA vGPU

This example adds an entry to the /etc/modprobe.d/nvidia.conf file to change the scheduling behavior of two GPUs as follows:

- For the GPU at PCI domain 0000 and BDF 85:00.0, the vGPU scheduling policy is set to Equal Share Scheduler.
- For the GPU at PCI domain 0000 and BDF 86:00.0, the vGPU scheduling policy is set to Fixed Share Scheduler.

```
options nvidia NVreg_RegistryDwordsPerDevice=
"pci=0000:85:00.0;RmPVMRL=0x01;pci=0000:86:00.0;RmPVMRL=0x11"
```

Reboot your hypervisor host machine.

### 4.7.5 Restoring Default vGPU Scheduler Settings

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 secure shell (SSH) for this purpose.
2. Unset the RmPVMRL registry key.
3. Set the module parameter to an empty string.

```
# esxcli system module parameters set -m nvidia -p "module-parameter="
```

**module-parameter**

The module parameter to set, which depends on whether the scheduling behavior was changed for all GPUs or select GPUs:

- For all GPUs, set the NVreg_RegistryDwords module parameter.
- For select GPUs, set the NVreg_RegistryDwordsPerDevice module parameter.
- For example, to restore default vGPU scheduler settings after they were changed for all GPUs, enter this command:

```
# esxcli system module parameters set -m nvidia -p "NVreg_RegistryDwords="
```

4. Reboot your hypervisor host machine

### 4.8 Disabling and Enabling ECC Memory

Some GPUs that support NVIDIA vGPU software support error correcting code (ECC) memory with NVIDIA vGPU. ECC memory improves data integrity by detecting and handling double-bit errors. However, not all GPUs, vGPU types, and hypervisor software versions support ECC memory with NVIDIA vGPU.
On GPUs that support ECC memory with NVIDIA vGPU, ECC memory is supported with C-series and Q-series vGPUs, but not with A-series and B-series vGPUs. Although A-series and B-series vGPUs start on physical GPUs on which ECC memory is enabled, enabling ECC with vGPUs that do not support it might incur some costs.

On physical GPUs that do not have HBM2 memory, the amount of frame buffer that is usable by vGPUs is reduced. All types of vGPU are affected, not just vGPUs that support ECC memory.

The effects of enabling ECC memory on a physical GPU are as follows:

- ECC memory is exposed as a feature on all supported vGPUs on the physical GPU.
- In VMs that support ECC memory, ECC memory is enabled, with the option to disable ECC in the VM.
- ECC memory can be enabled or disabled for individual VMs. Enabling or disabling ECC memory in a VM does not affect the amount of frame buffer that is usable by vGPUs.

GPUs based on the Pascal GPU architecture and later GPU architectures support ECC memory with NVIDIA vGPU. These GPUs are supplied with ECC memory enabled.

Tesla M60 and M6 GPUs support ECC memory when used without GPU virtualization, but NVIDIA vGPU does not support ECC memory with these GPUs. In graphics mode, these GPUs are supplied with ECC memory disabled by default.

Some hypervisor software versions do not support ECC memory with NVIDIA vGPU.

If you are using a hypervisor software version or GPU that does not support ECC memory with NVIDIA vGPU and ECC memory is enabled, NVIDIA vGPU fails to start. In this situation, you must ensure that ECC memory is disabled on all GPUs if you are using NVIDIA vGPU.

### 4.8.1 Disabling ECC Memory

If ECC memory is unsuitable for your workloads but is enabled on your GPUs, disable it. You must also ensure that ECC memory is disabled on all GPUs if you are using NVIDIA vGPU with a hypervisor software version or a GPU that does not support ECC memory with NVIDIA vGPU. If your hypervisor software version or GPU does not support ECC memory and ECC memory is enabled, NVIDIA vGPU fails to start.

Where to perform this task from depends on whether you are changing ECC memory settings for a physical GPU or a vGPU.

- For a physical GPU, perform this task from the hypervisor host.
- For a vGPU, perform this task from the VM to which the vGPU is assigned.

**Note:** ECC memory must be enabled on the physical GPU on which the vGPUs reside.

Before you begin, ensure that NVIDIA Virtual GPU Manager is installed on your hypervisor. If you are changing ECC memory settings for a vGPU, also ensure that the NVIDIA vGPU software graphics driver is installed in the VM to which the vGPU is assigned.
1. Use nvidia-smi to list the status of all physical GPUs or vGPUs, and check for ECC noted as enabled.

   # nvidia-smi -q

   ===========NVSMI LOG==============
   Timestamp                           : Mon Jul 13 18:36:45 2020
   Driver Version                      : 450.55
   Attached GPUs                       : 1
   GPU 0000:02:00.0

   [...]  
   
<table>
<thead>
<tr>
<th>Ecc Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current   : Enabled</td>
</tr>
<tr>
<td>Pending   : Enabled</td>
</tr>
</tbody>
</table>
   [...]  

2. Change the ECC status to off for each GPU for which ECC is enabled.

   - If you want to change the ECC status to off for all GPUs on your host machine or vGPUs assigned to the VM, run this command:

   # nvidia-smi -e 0

   - If you want to change the ECC status to off for a specific GPU or vGPU, run this command:

   # nvidia-smi -i \id\ -e 0

   \id\ is the index of the GPU or vGPU as reported by nvidia-smi.

   This example disables ECC for the GPU with index 0000:02:00.0.

   # nvidia-smi -i 0000:02:00.0 -e 0

3. Reboot the host or restart the VM.

4. Confirm that ECC is now disabled for the GPU or vGPU.

   # nvidia-smi -q

   ===========NVSMI LOG==============
   Timestamp                           : Mon Jul 13 18:37:53 2020
   Driver Version                      : 450.55
   Attached GPUs                       : 1
   GPU 0000:02:00.0

   [...]

<table>
<thead>
<tr>
<th>Ecc Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current   : Enabled</td>
</tr>
<tr>
<td>Pending   : Enabled</td>
</tr>
</tbody>
</table>
4.8.2 Enabling ECC Memory

If ECC memory is suitable for your workloads and is supported by your hypervisor software and GPUs, but is disabled on your GPUs or vGPUs, enable it.

Where to perform this task from depends on whether you are changing ECC memory settings for a physical GPU or a vGPU.

- For a physical GPU, perform this task from the hypervisor host.
- For a vGPU, perform this task from the VM to which the vGPU is assigned.

**Note:** ECC memory must be enabled on the physical GPU on which the vGPUs reside.

Before you begin, ensure that NVIDIA Virtual GPU Manager is installed on your hypervisor. If you are changing ECC memory settings for a vGPU, also ensure that the NVIDIA vGPU software graphics driver is installed in the VM to which the vGPU is assigned.

1. Use `nvidia-smi` to list the status of all physical GPUs or vGPUs, and check for ECC noted as disabled.

   ```
   # nvidia-smi -q
   =================NVSMI LOG=================
   Timestamp                           : Mon Jul 13 18:36:45 2020
   Driver Version                      : 450.55
   Attached GPUs                       : 1
   GPU 0000:02:00.0
   ...

   Ecc Mode
   Current                     : Disabled
   Pending                     : Disabled
   ...
   ```

2. Change the ECC status to on for each GPU or vGPU for which ECC is enabled.

   - If you want to change the ECC status to on for all GPUs on your host machine or vGPUs assigned to the VM, run this command:

   ```
   # nvidia-smi -e 1
   ```
- If you want to change the ECC status to on for a specific GPU or vGPU, run this command:

```
# nvidia-smi -i id -e 1
```

`id` is the index of the GPU or vGPU as reported by nvidia-smi.

This example enables ECC for the GPU with index 0000:02:00.0.

```
# nvidia-smi -i 0000:02:00.0 -e 1
```

3. Reboot the host or restart the VM.

4. Confirm that ECC is now enabled for the GPU or vGPU

```
# nvidia-smi -q

==============NVSMI LOG==============
Timestamp                           : Mon Jul 13 18:37:53 2020
Driver Version                      : 450.55
Attached GPUs                       : 1
GPU 0000:02:00.0
[...]

   Ecc Mode
    Current       : Enabled
    Pending       : Enabled
[...]
```
Chapter 5. Deploying the NVIDIA vGPU Software License Server

This chapter covers deployment of the NVIDIA vGPU software license server, including:

- Platform Requirements
- Installing the Java Runtime Environment on Windows
- Installing the License Server Software on Windows

5.1 Platform Requirements

Before proceeding, ensure that you have a platform suitable for hosting the license server.

5.1.1 Hardware and Software Requirements

- The hosting platform may be a physical machine, an on-premises virtual machine (VM), or a VM on a supported cloud service. NVIDIA recommends using a host that is dedicated solely to running the license server.
- The recommended minimum configuration is 2 CPU cores and 4 GB of RAM. A high-end configuration of 4 or more CPU cores with 16 GB of RAM is suitable for handling up to 150,000 licensed clients.
- At least 1 GB of hard drive space is required.
- The hosting platform must run a supported operating system.
- On Windows platforms, .NET Framework 4.5 or later is required.

5.1.2 Platform Configuration Requirements

- The platform must have a fixed (unchanging) IP address. The IP address may be assigned dynamically by DHCP or statically configured but must be constant.
- The platform must have at least one unchanging Ethernet MAC address, to be used as a unique identifier when registering the server and generating licenses in the NVIDIA Licensing Portal.
- The platform’s date and time must be set accurately. NTP is recommended.
5.1.3 Network Ports and Management Interface

The license server requires TCP port 7070 to be open in the platform’s firewall, to serve licenses to clients. By default, the installer will automatically open this port. The license server’s management interface is web-based and uses TCP port 8080. The management interface itself does not implement access control; instead, the installer does not open port 8080 by default, so that the management interface is only available to web browsers running locally on the license server host. Access to the management interface is therefore controlled by limiting remote access (via VNC, RDP, etc.) to the license server platform.

Note: If you choose to open port 8080 during license server installation, or at any time afterwards, the license server’s management interface is unprotected.

5.2 Installing the NVIDIA vGPU Software License Server on Windows

The license server requires a Java runtime environment, which must be installed separately before you install the license server.

5.2.1 Installing the Java Runtime Environment on Windows

If a suitable Java runtime environment (JRE) version is not already installed on your system install a supported JRE before running the NVIDIA license server installer.

1. Download a supported 64-bit Oracle Java SE JRE or OpenJDK JRE.
   - Download Oracle Java SE JRE from the Java Downloads for All Operating Systems page.
   - Download Oracle Java SE JRE from the java.com: Java + You page
   - Download OpenJDK JRE from the Community builds using source code from OpenJDK project on GitHub.

2. Install the JRE that you downloaded.
   - Oracle Java SE JRE installation:
• **OpenJDK JRE installation:**

3. Set the `JAVA_HOME` system variable to the full path to the jre folder of your JRE installation.
   - **For 64-bit Oracle Java SE JRE:** C:\Program Files\Java\jre1.8.0_191
   - **For 64-bit OpenJDK JRE:** C:\Program Files\ojdkbuild\java-1.8.0-openjdk-1.8.0.201-1\jre

   Ensure that the path does not include any trailing characters, such as a slash or a space.

   If you are upgrading to a new version of the JRE, update the value of the `JAVA_HOME` system variable to the full path to the jre folder of your new JRE version.

4. Ensure that the Path system variable contains the path to the java.exe executable file.
   - **For 64-bit Oracle Java SE JRE:** C:\Program Files\Java\jre1.8.0_191\bin
• **For 64-bit OpenJDK JRE**: C:\Program Files\ojdkbuild\java-1.8.0-openjdk-1.8.0.201-1\bin

5.2.2 Installing the License Server Software on Windows

1. Unzip the license server installer and run setup.exe.

5. Accept the EULA for the license server software and the Apache Tomcat software used to support the license server’s management interface.

6. Choose the destination folder where you want the license server software to be installed.
7. In the Choose Firewall Options dialog box, select the ports to be opened in the firewall.

To enable remote clients to access licenses from the server and prevent remote access to the management interface, use the default setting, which sets ports as follows:

- Port 7070 is open to enable remote clients to access licenses from the server.
- Port 8080 is closed to ensure that the management interface is available only through a web browser running locally on the license server host.
8. After installation has completed successfully, click Done to exit the installer.
5.2.3 Obtaining the License Server’s MAC Address

The license server’s Ethernet MAC address uniquely identifies your server to the NVIDIA Licensing Portal. You will need this address to register your license server with the NVIDIA Licensing Portal to generate license files.

1. Open a web browser on the license server host and connect to the URL http://localhost:8080/licserver.
2. In the license server management interface, select Configuration.
3. On the License Server Configuration page that opens, in the Server host ID drop-down list, select the platform’s ETHERNET address.

5.2.4 Managing your License Server and Getting your License Files

To be able to download NVIDIA vGPU software licenses, you must create at least one license server on the NVIDIA Licensing Portal and allocate licenses to the server. After creating a license server and allocating licenses to it, you can download your license file.

5.2.4.1 Creating a License Server on the NVIDIA Licensing Portal

To be able to download NVIDIA vGPU software licenses, you must create at least one license server on the NVIDIA Licensing Portal. Creating a license server on the NVIDIA Licensing Portal registers your license server host with the NVIDIA Licensing Portal through the MAC address of the host.

1. In the NVIDIA Licensing Portal, navigate to the organization or virtual group for which you want to create the license server.
a. If you are not already logged in, log in to the NVIDIA Enterprise Application Hub and click NVIDIA LICENSING PORTAL to go to the NVIDIA Licensing Portal.

b. **Optional:** If your assigned roles give you access to multiple virtual groups, select the virtual group for which you are creating the license server from the list of virtual groups at the top right of the page.

If no license servers have been created for your organization or virtual group, the NVIDIA Licensing Portal dashboard displays a message asking if you want to create a license server.

2. On the NVIDIA Licensing Portal dashboard, click **CREATE LICENSE SERVER.**

The Create License Server pop-up window opens.
3. Provide the details of your license server.
   a. In the **Server Name** field, enter the host name of the license server.
   b. In the **Description** field, enter a text description of the license server. This description is required and will be displayed on the details page for the license server that you are creating.
   c. In the **MAC Address** field, enter the MAC address of your license server.

4. Add the licenses for the products that you want to allocate to this license server. For each product, add the licenses as follows:
   a. From the **Product** drop-down list, select the product for which you want to add licenses.
   b. In the **Licenses** field, enter the number of licenses for the product that you want to add.
   c. Click **ADD**.

5. Leave the **Failover License Server** and **Failover MAC Address** fields unset.

6. Click **CREATE LICENSE SERVER**.

5.2.4.2 Downloading a License File

Each license server that you create has license file associated with it. The license file contains all the licenses that you allocated to the license server. After downloading the license file, you can install it on the license server host associated with the license server on the NVIDIA Licensing Portal.

1. In the NVIDIA Licensing Portal, navigate to the organization or virtual group for which you want to download the license file.
Deploying the NVIDIA vGPU Software License Server

a. If you are not already logged in, log in to the NVIDIA Enterprise Application Hub and click NVIDIA LICENSING PORTAL to go to the NVIDIA Licensing Portal.

b. Optional: If your assigned roles give you access to multiple virtual groups, select the virtual group for which you are downloading the license file from the list of virtual groups at the top right of the page.

2. In the list of license servers on the NVIDIA Licensing Portal dashboard, select the license server whose associated license file you want to download.

3. In the License Server Details page that opens, review the licenses allocated to the license server.

4. Click DOWNLOAD LICENSE FILE and save the .bin license file to your license server for installation.

5.2.5 Installing a License

NVIDIA vGPU software licenses are distributed as .bin files for download from the NVIDIA Licensing Portal.
Before installing a license, ensure that you have downloaded the license file from the NVIDIA Licensing Portal.

1. In the license server management interface, select **License Management**.

2. On the License Management page that opens, click **Choose File**.

3. In the file browser that opens, select the .bin file and click **Open**.

4. Back on the License Management page, click **Upload** to install the license file on the license server. The license server should confirm successful installation of the license file.
Deploying the NVIDIA vGPU Software License Server

Note: For additional configuration options including Linux server deployment, securing your license server, and license provisioning, refer to the Virtual GPU Software License Server User Guide.
Chapter 6. Creating Your First NVIDIA Virtual Compute Server VM

This chapter covers creating an NVIDIA Virtual Compute Server VM, including:

- Creating a Virtual Machine
- Installing Ubuntu Server 18.04.4 LTS
- Enabling the NVIDIA vGPU
- Installing the NVIDIA Driver in the Ubuntu Virtual Machine
- Licensing an NVIDIA vGPU

6.1 Creating a Virtual Machine

These instructions are to assist in making a VM from scratch that will support NVIDIA vGPU. Later the VM will be used as a gold master image. Use the following procedure to configure a vGPU for a single guest desktop:

1. Browse to the host or cluster using the vSphere Web Client.

2. Right-click the desired host or cluster and select New Virtual Machine.
The **New Virtual Machine** wizard begins.

9. **Select Create a new virtual machine** and click **Next**.

10. Enter a name for the virtual machine. Choose the location to host the virtual machine using the **Select a location for the virtual machine** section. Click **Next** to continue.
11. Select a compute resource to run the VM. Click Next to continue.

Note: This compute resource should include an NVIDIA vGPU enabled card installed and be correctly configured.

12. Select the datastore to host the virtual machine. Click Next to continue.
13. Select compatibility for the virtual machine. This allows VMs to run on different versions of ESXi. To run vGPU select ESXi 6.0 and later. Click Next to continue.

14. Select the appropriate Ubuntu Linux OS from the Guest OS Family and Guest OS Version pull-down menus. Click Next to continue.
15. Customize hardware is next. Set the virtual hardware based on your compute workload requirements. Click Next to continue.

16. Review the New Virtual Machine configuration prior to completion. Click **Finish** when ready.
6.2 Installing Ubuntu Server 18.04.4 LTS

1. Download Ubuntu Server OS.
2. Mount the ISO to your VM and make sure to check Connect At Power On. Click Okay to finish.

4. Select your preferred language and press the enter key.
5. **Continue without updating as this guide is built around 18.04.**

6. **On this screen, select your network connection type and modify to fit your internal requirements.**

   We will be using DHCP for our configuration.
7. Format the entire disk.

8. Configure the VM with a user account, name, and password.
9. Select Install OpenSSH server and select Done.

10. Select any server snaps that maybe required for internal use in your environment and select Done.
Installation will now complete. VMware Tools will be installed and managed by Ubuntu server.

6.3 Enabling the NVIDIA vGPU

Use the following procedure to enable vGPU support for your virtual machine (you must edit the virtual machine settings):

1. Power down the virtual machine.

11. Click on the VM in the Navigator window. Right click the VM and Edit Settings.

   The Edit Settings dialog appears.
12. Click on the New Device bar and select Shared PCI device.
13. Click on Add to continue

![Image of Edit Settings window with NVIDIA GRID vGPU selected]

14. Select the desired GPU Profile underneath the New PCI device.

![Image of virtual machine settings with NVIDIA GRID vGPU selected]

15. Click Reserve all memory!

![Warning message about VM not powering on until memory reservation equals memory size]

16. Click OK to complete the configuration.
6.4 Installing the NVIDIA Driver in the Ubuntu Virtual Machine

After you create a Linux VM on the hypervisor and boot the VM, install the NVIDIA vGPU software display driver in the VM to fully enable GPU operation.

Installation of the NVIDIA vGPU software display driver for Linux requires:

- Compiler toolchain
- Kernel headers

1. Log in and shut down the display manager.
   ```
   sudo service lightdm stop
   ```

2. From a console shell, run the driver installer as the root user.
   ```
   sudo sh ./ NVIDIA-Linux_x86_64-440.87-grid.run
   ```
   In some instances, the installer may fail to detect the installed kernel headers and sources. In this situation, re-run the installer, specifying the kernel source path with the --kernel-source-path option:
   ```
   sudo sh ./ NVIDIA-Linux_x86_64-440.87-grid.run \
   --kernel-source-path=/usr/src/kernels/3.10.0-229.11.1.el7.x86_64
   ```

3. When prompted, accept the option to update the X configuration file (xorg.conf).

4. Enable Persistence Mode.
   ```
   sudo systemctl daemon-reload
   sudo systemctl enable nvidia-persistenced.service
   sudo systemctl start nvidia-persistenced.service
   ```

5. Reboot the system.
   ```
   sudo reboot
   ```

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

After you install the NVIDIA vGPU software graphics driver, you can license any NVIDIA vGPU software licensed products that you are using. For instructions, see Licensing an NVIDIA vGPU (update 11.0).

6.5 Licensing an NVIDIA vGPU

NVIDIA vGPU is a licensed product. When booted on a supported GPU, a vGPU initially operates at full capability but its performance is degraded over time if the VM fails to obtain a license. If the performance of a vGPU has been degraded, the full capability of the vGPU is restored when a license is acquired. For complete information about configuring and using NVIDIA vGPU software licensed features, including vGPU, refer to Virtual GPU Client Licensing User Guide.

Perform this task from the guest VM to which the vGPU is assigned.
The NVIDIA X Server Settings tool that you use to perform this task detects that a vGPU is assigned to
the VM and, therefore, provides no options for selecting the license type. After you license the vGPU,
NVIDIA vGPU software automatically selects the correct type of license based on the vGPU type.

1. Start NVIDIA X Server Settings by using the method for launching applications provided by your
Linux distribution.
   For example, on Ubuntu Desktop, open the Dash, search for NVIDIA X Server Settings, and click
the NVIDIA X Server Settings icon.
2. In the NVIDIA X Server Settings window that opens, click Manage GRID License.
   The License Edition section of the NVIDIA X Server Settings window shows that NVIDIA vGPU is
currently unlicensed.
3. In the Primary Server field, enter the address of your primary NVIDIA vGPU software License
   Server.
   The address can be a fully qualified domain name such as gridlicense1.example.com, or an IP
   address such as 10.31.20.45. If you have only one license server configured, enter its address in
   this field.
4. Leave the Port Number field under the Primary Server field unset.
   The port defaults to 7070, which is the default port number used by NVIDIA vGPU
   software License Server.
5. In the Secondary Server field, enter the address of your secondary NVIDIA vGPU software License
   Server.
   If you have only one license server configured, leave this field unset. The address can be a fully
   qualified domain name such as gridlicense2.example.com, or an IP address such as 10.31.20.46.
6. Leave the Port Number field under the Secondary Server field unset.
   The port defaults to 7070, which is the default port number used by NVIDIA vGPU
   software License Server.
7. Click Apply to assign the settings.
   The system requests the appropriate license for the current vGPU from the configured license
   server.

The vGPU within the VM should now exhibit full frame rate, resolution, and display output
capabilities. The VM is now capable of running the full range of DirectX and OpenGL graphics
applications.

If the system fails to obtain a license, see Virtual GPU Client Licensing User Guide for guidance on
troubleshooting.
Choosing the right vGPU profile to maximize your stakeholders experience within the virtual instance is critical for ensuring expected performance and quality of service. Below, you will find guidance through the vGPU Manager and beyond to ensure your deployment is successful.

7.1  The Role of the vGPU Manager

NVIDIA vGPU profiles assign custom amounts of dedicated GPU memory for each user. NVIDIA vGPU Manager assigns the correct amount of memory to meet the specific needs within the workflow for said user. Every virtual machine has dedicated GPU memory and must be assigned accordingly thus ensuring that it has the resources needed to handle the expected compute load.

NVIDIA vGPU Manager allows up to eight users to share each physical GPU by assigning the graphics resources of the available GPUs to virtual machines using a balanced approach. Depending on the number of GPUs within each line card, there can be multiple user types assigned.

7.2  VGPU Profiles for NVIDIA Virtual Compute Server

The profiles represent a very flexible deployment option of virtual GPUs, varying in size of GPU memory. The division of GPU memory defines the number of vGPUs that are possible per GPU.

NVIDIA vCS is supported on the following NVIDIA GPUs: A100 (SXM4), A100 (PCIe), T4, RTX6000, RTX8000, V100 (SXM2), V100S/V100 (PCIe), P40, P100 and P6 for blade form factor.

C-series vGPU types are NVIDIA vCS vGPU types, which are optimized for compute-intensive workloads. As a result, they support only a single display head at a maximum resolution of 4096×2160 and do not provide Quadro graphics acceleration.
The following table illustrates examples of the NVIDIA vCS profiles and how they fractionalize.

<table>
<thead>
<tr>
<th>Virtual GPU Type</th>
<th>Intended Use Case</th>
<th>Frame Buffer (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>48C</td>
<td>Training Workloads</td>
<td>49152</td>
</tr>
<tr>
<td>32C</td>
<td>Training Workloads</td>
<td>32768</td>
</tr>
<tr>
<td>24C</td>
<td>Training Workloads</td>
<td>24576</td>
</tr>
<tr>
<td>16C</td>
<td>Training Workloads</td>
<td>16384</td>
</tr>
<tr>
<td>12C</td>
<td>Training Workloads</td>
<td>12288</td>
</tr>
<tr>
<td>8C</td>
<td>Training Workloads</td>
<td>8192</td>
</tr>
<tr>
<td>6C</td>
<td>Training Workloads</td>
<td>6144</td>
</tr>
<tr>
<td>4C</td>
<td>Inference Workloads</td>
<td>4096</td>
</tr>
</tbody>
</table>
Chapter 8. GPU Aggregation for NVIDIA Virtual Compute Server

NVIDIA vCS supports GPU aggregation where a VM can access more than one GPU, which is often required for compute-intensive workloads. NVIDIA vCS supports both multi-vGPU and peer-to-peer computing. The following sections describe both technologies and how to deploy GPU aggregation within VMware ESXi.

8.1 Multi vGPU

NVIDIA vCS supports multi vGPU workloads which can offer a monumental improvement in virtual GPU performance by aggregating the power of up to four NVIDIA GPUs in a single virtual machine. With multi-vGPU, the GPUs are not directly connected to one another. The following graphic illustrates multi-gpu and how a single VM can be assigned two shared PCIe devices:

![Diagram of multi-vGPU](image)

8.2 Peer-to-Peer NVIDIA NVLINK

NVIDIA vCS supports peer to peer computing where multiple GPU’s are connected through NVIDIA NVLink. This enables a high speed, direct GPU-to-GPU interconnect that provides higher bandwidth for multi-GPU system configurations than traditional PCIe-based solutions. The following graphic illustrates peer-to-peer NVLINK:
This peer-to-peer communication allows access to device memory between GPU’s from within the CUDA kernels and eliminates the system memory allocation and copy overheads. It provides a more convenient multi-GPU programming. Peer-to-Peer CUDA Transfers over NVLink are supported for Linux only and are not supported on Windows. Currently vGPU does not support NVSwitch therefore only direct connections are supported.

Peer-to-Peer CUDA Transfers over NVLink are supported only on a subset of vGPUs, VMware vSphere Hypervisor (ESXi) releases, and guest OS releases. Only C-series full frame buffer (1:1) vGPU profiles are supported with NVLink. Refer to the vGPU latest release notes for a listed of GPU’s which are supported.

1. Connect to the ESXi host over SSH, for example using Putty
2. Type nvidia-smi within the command window.
3. Detect the topology between the GPUs by typing the following command:
   `nvidia-smi topo -m`

   ![Topology Diagram]

   **Legend:**
   - `X` = Self
   - `SYS` = Connection traversing PCIe as well as the SMP interconnect between NUMA nodes (e.g., QPI/UP1)
   - `NODE` = Connection traversing PCIe as well as the interconnect between PCIe Host Bridges within a NUMA node
   - `PHB` = Connection traversing PCIe as well as a PCIe Host Bridge (typically the CPU)
   - `PBX` = Connection traversing multiple PCIe bridges (without traversing the PCIe Host Bridge)
   - `TX` = Connection traversing at most a single PCIe bridge
   - `NVL` = Connection traversing a bonded set of # NVLinks

4. Assign suitable 1:1 vGPU(s) to the VM.

   The CUDA driver in the VM will detect the peer-to-peer capability between the vGPUs and allow the CUDA application to use it.
Chapter 9. Page Retirement and ECC

NVIDIA vCS supports ECC and dynamic page retirement. This feature will "retire" bad framebuffer memory cells, by retiring the page the cell belongs to. Dynamic page retirement is done automatically for cells that are degrading in quality. This feature can improve the longevity of an otherwise good board and is thus an important resiliency feature on supported products, especially in HPC and enterprise environments. Retiring of pages may only occur when ECC is enabled. However, once a page has been retired it will always be blacklisted, even if ECC is later disabled. Refer to the NVIDIA Developer Zone page retirement documentation for more information.

These page retirement and ECC features are offered on all GPUs that are supported on NVIDIA vCS.
Chapter 10. Installing Docker and The Docker Utility Engine for NVIDIA GPUs

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

Make sure you have installed the NVIDIA driver and Docker 19.03 for your Linux distribution. Note that you do not need to install the CUDA toolkit on the host, but the driver needs to be installed.

Note that with the release of Docker 19.03, usage of nvidia-docker2 packages are deprecated since NVIDIA GPUs are now natively supported as devices in the Docker runtime.

For first-time users of Docker 19.03 and GPUs, continue with the instructions for getting started below.

1. Add the package repositories.

   ```bash
distribution=$(./etc/os-release;echo $ID$VERSION_ID)
curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add -
curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list | sudo tee /etc/apt/sources.list.d/nvidia-docker.list
```

2. Download information from all configured sources about the latest versions of the packages and install the `nvidia-container-toolkit` package.

   ```bash
   sudo apt-get update && sudo apt-get install -y nvidia-container-toolkit
   ```

3. Restart the docker service.

   ```bash
   sudo systemctl restart docker
   ```

10.2 Testing Docker and NVIDIA Container Run Time

```bash
### Test nvidia-smi with the latest official CUDA image

docker run --gpus all nvidia/cuda:10.0-base nvidia-smi

# Start a GPU enabled container on two GPUs

docker run --gpus 2 nvidia/cuda:10.0-base nvidia-smi

# Starting a GPU enabled container on specific GPUs

docker run --gpus '"device=1,2"' nvidia/cuda:10.0-base nvidia-smi

docker run --gpus '"device=UUID-ABCDEF,1"' nvidia/cuda:10.0-base nvidia-smi

# Specifying a capability (graphics, compute, ...) for my container

# Note this is rarely if ever used this way

docker run --gpus all,capabilities=utility nvidia/cuda:10.0-base nvidia-smi
```
Chapter 11. Testing and Benchmarking

All deep learning frameworks are found on the NGC container registry. [https://ngc.nvidia.com/container](https://ngc.nvidia.com/container). NVIDIA is using the 19.04-py3 containers for each DL framework. Instructions for installing NVIDIA Docker can be found on the GitHub page [https://github.com/NVIDIA/nvidia-docker](https://github.com/NVIDIA/nvidia-docker).

Note that most of these assume you have the dataset available on your system. NVIDIA is not allowed to distribute ImageNet ([http://image-net.org/download](http://image-net.org/download)) so customers will have to acquire it themselves (needed for all the RN50 training benchmarks).

Following are several examples with GNMT. While the dataset is the same, the preprocessing on the dataset is different for each case. Therefore, you cannot use the same dataset for each run. You must run the specific command to download and process the data to the benchmark example.

The following instructions are intended to be a shortcut to getting started with benchmarking. In the working directory of each benchmark, there is a README file (named either README.md or README.txt) that provides more details of data download, preprocessing, and running the code.

### 11.1 TensorRT RN50 Inference

- The container used in this example nvcr.io/nvidia/tensorrt:19.04-py3.
- Binary needed is included with the container at: /workspace/tensorrt/bin
- The Resnet50 model prototxt and caffemodel files are within the container at:/workspace/tensorrt/data/resnet50
- The command may take several minutes to run because NVIDIA® TensorRT™ is building the optimized plan prior to running. If you wish to see what it is doing, add --verbose to the command.

#### 11.1.1 Commands to the Run Test

```bash
$ docker pull nvcr.io/nvidia/tensorrt:19.04-py3
$ nvidia-docker run -it --rm -v $(pwd):/work nvcr.io/nvidia/tensorrt:19.04-py3
  # cd /workspace/tensorrt/data/resnet50
  # /workspace/tensorrt/bin/trtexec --batch=128 --iterations=400 --workspace=1024 --percentile=99
  deploy=ResNet50_N2.prototxt --model=ResNet50_fp32.caffemodel --output=prob --int8
```
11.1.2 Interpreting the Results

Results are reported in time to infer the given batch size. To convert to images per second compute \( \text{BATCH\_SIZE/AVERAGE\_TIME} \).

11.2 TensorFlow RN50 Mixed Training

- The container used in this example `nvcr.io/nvidia/tensorflow:19.04-py3`.
- The scripts for this test are in `/workspace/nvidia-examples/cnn`.
- The example is a synthetic training example, so no data is needed.
- The file `README.md` describes the functionality of this test.

11.2.1 Commands to Run the Test

```bash
$ docker pull nvcr.io/nvidia/tensorflow:19.04-py3
$ nvidia-docker run -it --rm -v $(pwd):/work nvcr.io/nvidia/tensorflow:19.04-py3
# cd /workspace/nvidia-examples/cnn
# mpirun --allow-run-as-root -np 1 python -u ./resnet.py --batch_size 256 --num_iter 800 --precision fp16 --iter_unit batch --layers 50
```

11.2.2 Interpreting the Results

This benchmark reports images per second training performance at each reporting iteration. Use the last few values reported to represent training performance.
Chapter 12. Troubleshooting

12.1 Forums

NVIDIA forums are a very inclusive source of solutions to many problems that may be faced when deploying a virtualized environment. Search on the NVIDIA forums located at https://gridforums.nvidia.com/ first.

You may also wish to look through the NVIDIA Enterprise Services Knowledgebase to find further support articles and links at https://nvidia-esp.custhelp.com/app/answers/list/autologout/1

Keep in mind that not all issues within your deployment may be answered in the NVIDIA vGPU forums. You may also have to reference forums from the hardware supplier, the hypervisor and application themselves.

Some examples of other key forums to look through are as follows:

- VMware Forums: https://communities.vmware.com/welcome
- HPE ProLiant Server Forums: https://community.hpe.com/t5/ProLiant/ct-p/proliant
- Dell Server Forums: https://www.dell.com/community/Servers/ct-p/ESServers

12.2 Filing a Bug Report

When filing a bug or requesting support assistance, it is critical to include information about the environment, so that the technical staff that can help you resolve the issue. NVIDIA includes the nvidia-bug-report.sh script within the vib installation package to collect and package this critical information. The script collects the following information:

- VMware version
- X.Org log and configuration
- PCI information
- CPU information
- GPU information
- `esxcf` information for PLX devices
- `esxcf` information for GPU devices
- VIB information
- NVRM messages from vmkernel.log
Troubleshooting

- **System dmesg output**
- **Which virtual machines have vGPU or vSGA configured**
- **NSMI output**

When running this script:

- You may specify the output location for the bug report using either the `-o` or `--output` switch followed by the output file name. If you do not specify an output directory, the script will write the bug report to the current directory.
- If you do not specify a file name, the script will use the default name `nvidia-bug-report.log.gz`.
- If the selected directory already contains a bug report file, then the script will change the name of that existing report file to `nvidia-bug-report.log.old.gz` before generating a new `nvidia-bug-report.log.gz` file.

To collect a bug report, issue the command:

```
$ nvidia-bug-report.sh
```

The system displays the following message during the collection process:

```
nvidia-bug-report.sh will now collect information about your system and create the file 'nvidia-bug-report.log.gz' in the current directory. It may take several seconds to run. In some cases, it may hang trying to capture data generated dynamically by the vSphere kernel and/or the NVIDIA kernel module. While the bug report log file will be incomplete if this happens, it may still contain enough data to diagnose your problem.
```

Be sure to include the `nvidia-bug-report.log.gz` log file when reporting problems to NVIDIA.
Appendix A. Using WINSCP to Upload the vGPU Manager VIB to Server Host

This section describes how to upload a .VIB file using WinSCP. Before doing this:

- Ensure that SSH is enabled on ESXi host (see Chapter 2 on page 7).
- Download and install WinSCP on a Windows PC that has network connectivity to your Esxi host.

Use the following procedure to upload a .VIB file using WinSCP:

1. Start WinSCP to display the Login screen.
2. Enter the connection details required to connect to the ESXi host.
3. Select Login to display the WinSCP Login window.

![WinSCP Login window](image)

- Note: If you are connecting to this server for the first time, a warning dialog will appear asking you to confirm the connection.

4. Navigate to the local folder containing the .VIB file that you want to upload to the ESXi host once the connection is established. Use the left pane of the WinSCP interface.
5. Navigate to a data store you want to upload the .VIB file to using the right pane of the WinSCP interface.
6. Right-click the .VIB file and select **Upload** to upload the file to the host.
Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation ("NVIDIA") makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice. Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete. NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer ("Terms of Sale"). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death, or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer’s own risk.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer’s sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer’s product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA’s aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

Trademarks

NVIDIA, the NVIDIA logo, CUDA, NVIDIA OptiX, NVIDIA RTX, NVIDIA Turing, Quadro, Quadro RTX, and TensorRT trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

© 2020 NVIDIA Corporation. All rights reserved.