# MIXED WORKLOADS – VIRTUAL DESKTOPS AND HPC ON COMMON ARCHITECTURE

DG-08567-001_v01 | Oct 2018

**Reference Design Guide**

# DOCUMENT CHANGE HISTORY

DG-08567-001_v01

| Version | Date | Authors | Description of Change |
|---|---|---|---|
| 01 | Oct 2018 | FD, DH, SF | Initial release |
| | | | |

# TABLE OF CONTENTS



# LIST OF FIGURES



# LIST OF TABLES

# MIXED WORKLOADS – VIRTUAL DESKTOPS AND HPC ON COMMON ARCHITECTURE

When considering the implementation of mixed workload virtualized infrastructure in the enterprise, many might require the need for a common platform of universal systems that can be leveraged by several types of workloads. Often these platforms leverage common cloud concepts like hardware virtualization, common software defined storage volumes, and fabric-based software defined networks.

It is critical to determine the workloads that will be deployed into the infrastructure as a prerequisite to designing the environment. Those workloads may range from Virtual Desktop Infrastructure (VDI) to Deep Learning (DL) applications, or High Performance Computing (HPC) clustered applications. It is important to consider the requirements of all potential workloads that may be hosted on the infrastructure when determining the hardware and software configuration requirements.

Additionally, it is also important to consider the orchestration methodologies that will be used to change the deployed topology of workloads in the environment. For instance, if you anticipate a reduction in virtual desktop utilization during off peak hours, having software orchestration tools to shut down the underutilized VDI infrastructure and subsequently turn on an alternative workload virtual infrastructure will be key to maximizing your infrastructure return on investment (ROI).

**Note:** This document is **not** intended to provide complete instructions for designing and building a production system. The system described in this document is suggested as a reference configuration that may be used as a basis for further experimentation, evolution, and refinement of a mixed workload environment concept. NVIDIA **cannot** provide technical support or services to the "do-it-yourself" (DIY) community based on the information contained in this document.

# DESIGN REQUIREMENTS

The primary goal of this experiment was to build a common cluster infrastructure capable of delivering the rich graphics requirements of high-end virtual workstations like those used by engineers and analysts. Users of the VDI would have access to capabilities for compute workloads both on their virtual workstations as well as additional high-performance computing virtual machine nodes. These HPC nodes could be leveraged either in a standalone single-node or as a cluster depending on the application requirements.

The system should appear and function logically identical to a conventional arrangement of multiple networked workstations and servers used for similar functions by the end users. Application requirements should be considered, and the selection of test applications should be conducive to known limitations of the available technology.

A baseline VDI configuration will be leveraged as the primary building block for our experiment. A common virtual desktop graphics infrastructure capability will be used to deliver both graphics and GPU compute resources. The design must meet the following requirements:

- NVIDIA® Tesla® data center GPUs based on the Pascal™ architecture or later which support CUDA code execution in all NVIDIA virtual GPU (vGPU) profile types.
- NVIDIA® Quadro® Virtual Data Center Workstation (Quadro vDWS) licensing to maintain the availability of CUDA code execution within each virtual machine and large framebuffer requirements.
- ECC mode disabled on the NVIDIA Tesla data center GPUs to support vGPU for graphics rendering.

# INFRASTRUCTURE

The choice of hardware should only be limited by the manufacturers support for NVIDIA Pascal-based architecture GPUs or later and 64-bit processors released after September 2006 that support hardware virtualization using either Intel VT-x or AMD RVI enablement. It is recommended to use systems that support Unified Extensible Firmware Interface (UEFI) boot methodologies.

Most hypervisors also require that the NX/XD bit be enabled for the CPU in the BIOS, and a minimum of 4GB of physical RAM, although typical VDI and HPC workloads will require significantly more RAM and therefore the workload constraints should be the primary consideration factors for RAM sizing.

Network interfaces should be allocated based on the storage and client network requirements for the workloads.

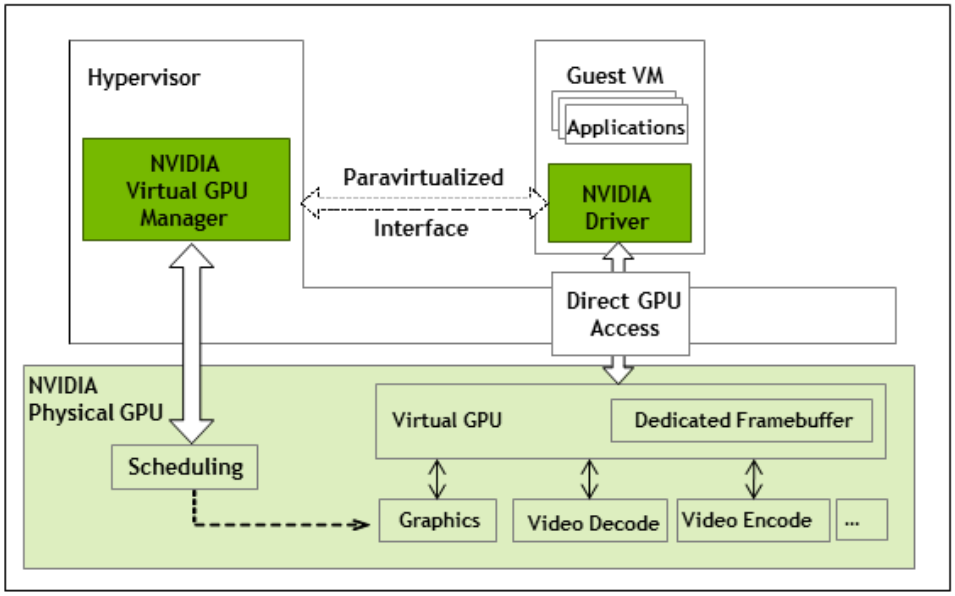
## TEST ENVIRONMENTS

Our test environments were setup with minimal configuration to get things working. These details are not to be considered production ready or deployment best practices. Rather, these are field test notes regarding experimental features or first release software by multiple vendors in addition to NVIDIA, and there may be some unexpected behaviors. These details are therefore provided without warranty or support.

# GPUs and vGPU Manager

NVIDIA Tesla GPUs based on the Pascal architecture or later are preferred because they allow for CUDA processes to be executed on any vGPU profile. Prior to the Pascal-based GPUs, it was required to either utilize vDGA full PCIe passthrough or the largest vGPU profile to execute CUDA.

## Figure 1: NVIDIA vGPU Architecture




All NVIDIA Tesla GPUs based on Pascal and Volta architectures can be used with NVIDIA virtual GPU software which enables the NVIDIA vGPU Manager at the hypervisor level. The NVIDIA Tesla P100 and Tesla V100 SXM2 GPUs can be used with NVIDIA virtual GPU software release 6.x or later. Table 1 outlines the currently available support for vGPU on the currently available hypervisors.

Table 1: NVIDIA vGPU Supported Hypervisors

| Software | Release Supported | Notes |
|---|---|---|
| Citrix Hypervisor 7.4 (formerly XenServer) | RTM build is supported. | All NVIDIA GPUs that support NVIDIA vGPU software are supported.<br>This release supports Citrix XenMotion with NVIDIA vGPU software on suitable GPUs as listed in the NVIDIA Certified Servers Matrix. |
| Citrix Hypervisor 7.3 | RTM build is supported. | All NVIDIA GPUs that support NVIDIA vGPU software are supported.<br>XenMotion with vGPU is **not** supported. |
| Citrix Hypervisor 7.2 | RTM build is supported. | All NVIDIA GPUs that support NVIDIA vGPU software are supported.<br>XenMotion with vGPU is **not** supported. |
| Citrix Hypervisor 7.1 | RTM build is supported. | All NVIDIA GPUs that support NVIDIA vGPU software are supported.<br>XenMotion with vGPU is **not** supported. |
| Nutanix AHV | 5.5 RTM | NVIDIA Tesla M10, M60, P40 supported.<br>Live Migration with vGPU is not supported. |
| Red Hat RHV 4.2 | RHEL with KVM 7.5 | All NVIDIA GPUs that support NVIDIA vGPU software are supported.<br>NVIDIA Virtual GPU Manager for the Red Hat Enterprise Linux with KVM releases listed in Hypervisor Software Releases, NVIDIA vGPU 6.1 software required.<br>Live Migration with vGPU is not supported. |

| Software | Release Supported | Notes |
|---|---|---|
| VMware vSphere Hypervisor (ESXi) 6.7 | 6.7 and compatible updates | All NVIDIA GPUs that support NVIDIA vGPU software are supported.<br>This release supports suspend and resume with NVIDIA vGPU software on suitable GPUs as listed in the NVIDIA Certified Servers Matrix.<br><br>VMware vSphere 6.7 update 1 supported vMotion on suitable GPUs as listed in the NVIDIA Certified Servers Matrix |
| VMware vSphere Hypervisor (ESXi) 6.5 | 6.5 and compatible updates | All NVIDIA GPUs that support NVIDIA vGPU software are supported.<br>Suspend-resume and vMotion with vGPU is **not** supported. |
| VMware vSphere Hypervisor (ESXi) 6.0 | 6.0 and compatible updates | GPUs based on the Pascal and Volta architectures in pass-through mode require 6.0 Update 3 or later.<br>For vGPU, all NVIDIA GPUs that support NVIDIA vGPU software are supported.<br>Suspend-resume and vMotion with vGPU is **not** supported. |

# Test Case 1: VMware vSphere 6.5 + VSAN + VMware Horizon 7

## HW outline:

- 4x Supermicro SYS-2027GR-TRFH
    - Dual-socket 20-core Intel Xeon E5-2690 v2
    - 128GB DDR3 memory
    - 4x NVIDIA® Tesla® P100 PCIe 16GB GPU
    - 3x Samsung SSD 850 EVO 500GB
    - Dual-port Intel X540-AT2 10GBASE-T NIC

## Infrastructure

The VMware mixed-workload infrastructure was built using common OEM systems to validate the lack of requirement for new, expensive or high-end hardware.

The four servers were provisioned with a recent version of VMware ESXi and joined to a 4-node vSphere cluster utilizing VMware vCenter. Networking utilized 10Gb Ethernet over a VMware distributed virtual switch, and storage utilized local SSDs in a VMware virtual SAN.

Virtual desktops were provisioned using VMware Horizon from a single Windows 10 virtual machine template with a P100-8Q vGPU profile. Since all vGPU profiles allocated on a single GPU must be of the same type, careful planning must be done to ensure optimal utilization of GPU resources. The test environment utilized several instances of Tesla P100-8Q profiles, including both VDI and HPC virtual machines.

The HPC portion of the virtual mixed-workload environment was provisioned using Terraform, a tool which allows you to define infrastructure as code and automate the deployment of entire virtual environments. The Terraform VMware module allows you to define networks, storage, and virtual machines to create a virtual HPC infrastructure alongside a VDI deployment. Terraform can also dynamically scale virtual machine counts to meet demands.

The deployed virtual HPC infrastructure was configured with Ansible to define user accounts, shared storage, required software packages, and the Slurm job scheduling system. Shared storage was provisioned using a block device on VMware vSAN, and shared to HPC compute VMs and VDI VMs using NFS and CIFS. Shared storage facilitates easy workload pipelines by reducing or eliminating data movement.

## Workflow

The experiment was designed to showcase a typical engineering HPC workflow. Starting with a three-stage workflow, which included a pre-processing, execution, and then post-processing step.

1. Pre-processing

   Execution of the pre-processing stage was performed on a Tesla P100-8Q vGPU-enabled Microsoft Windows 10 virtual machine. This step sets up the execution phase by allowing for the creation and testing of the HPC job using a common tool, Altair HyperWorks, to import a CAD model and create the CAE workflow solvers. This was done via HyperMesh to prepare the structural analysis job of a turbine blade before execution.

2. Execution

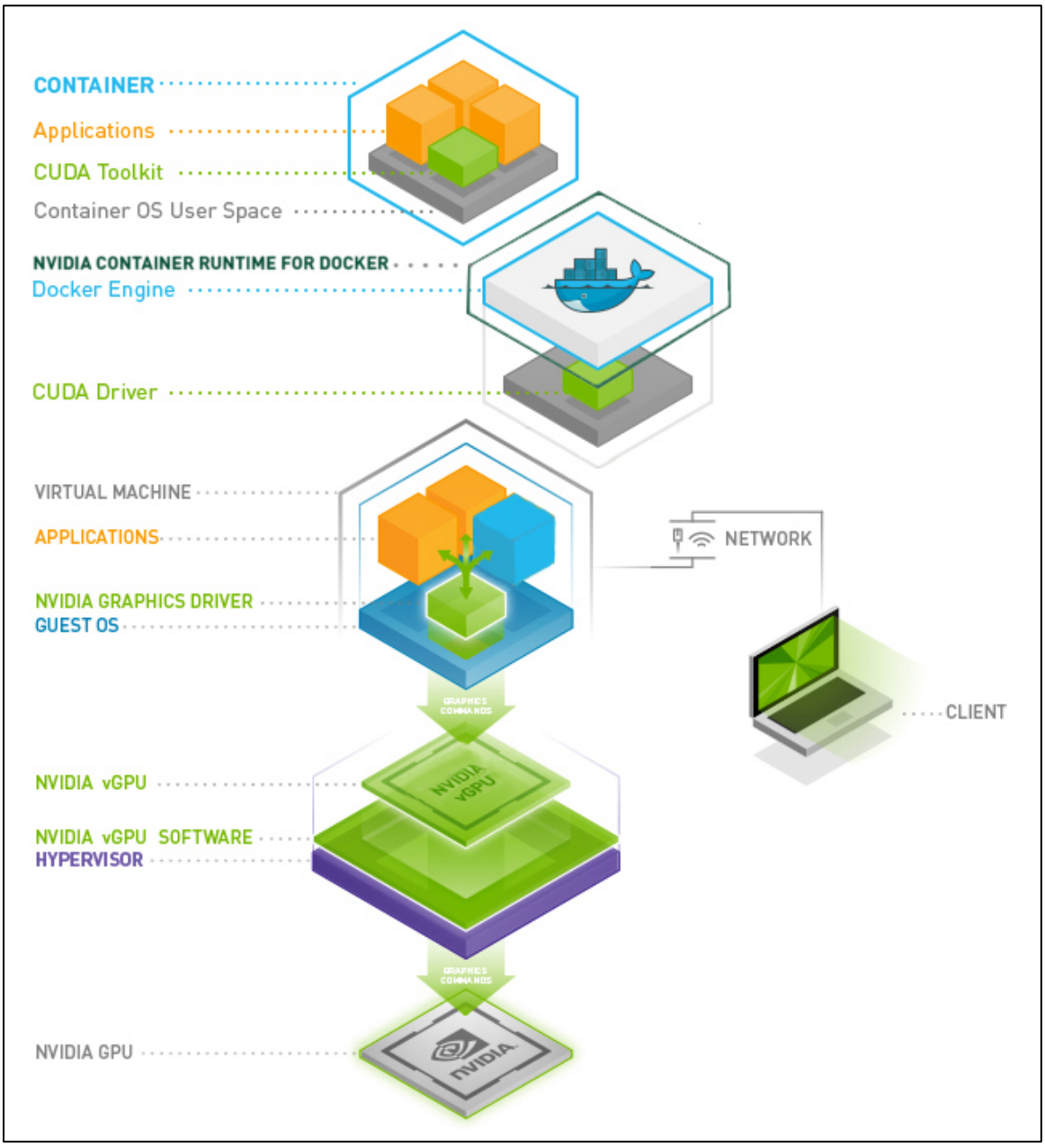
   Before execution could be performed at an HPC level we first did process evaluation testing on a single Tesla P100-8Q vGPU enabled HPC node. This node was Ubuntu 16.0.4, which could execute an NVIDIA Docker container running CentOS 7.4. The CentOS 7.4 was a core requirement of the ANSYS Mechanical HPC engine runtime. Custom Slurm scripts were used to initiate the Docker runtime, which would start the CentOS 7.4 container and then subsequently execute the ANSYS job.

   Once single execution was achieved, additional customization was required to the Slurm script to split the job across multiple nodes. This also required assigning additional IP addresses to each HPC node Ubuntu VM and statically mapping these IP addresses to each worker node Docker CentOS container. A detailed view of the software infrastructure stack can be found in Figure 2.

3. Post-processing

   Once execution on the HPC was completed, any post-process analysis was performed on a Tesla P100-8Q vGPU-enabled Microsoft Windows 10 virtual machine. This specific analysis was performed using Altair HyperWorks HyperView tools. This provides the ability to view the results of the ANSYS Mechanical CAE execution job.

Figure 2: Containers on NVIDIA vGPU Architecture



# Test Case 2: Red Hat RHV 4.2 + Citrix Virtual Desktops 7.15

## HW outline:

- 3x Supermicro SYS-2027GR-TRFH
    - Dual-socket 20-core Intel Xeon E5-2690 v2
    - 512GB DDR3 memory
    - 1x NVIDIA Tesla V100 PCIe 16GB GPU
    - 1x Samsung SSD 850 EVO 500GB
    - Dual-port Intel X540-AT2 10GBASE-T NIC

## Infrastructure

The Red Hat Virtualization mixed-workload infrastructure was built using common OEM systems to validate the lack of requirement for new, expensive or high-end hardware.

Installation details can be found on the *Red Hat Virtualization Self-Hosted Engine Guide*[1] and *Virtual GPU Software R390 for Red Hat Enterprise Linux with KVM Release Notes*[2]. The 3 servers were provisioned with the latest version of Red Hat Virtualization Host 7, a minimal version of Red Hat Enterprise Linux designed as a dedicated hypervisor. The Red Hat Virtualization Manager was bootstrapped via yum to start the cluster deployment on a single host, after which the other hosts were added to form a cluster. Networking utilized 10Gb Ethernet over a virtual switch, and storage utilized iSCSI via a dedicated SAN appliance.

NVIDIA vGPU setup involved installing the NVIDIA vGPU driver on each hypervisor and polling the system to retrieve a list of supported vGPU profiles. vGPU assignments are made by setting the 'mdev_type' custom property of a virtual machine to one of the available vGPU profiles. Each virtual machine assigned a vGPU must also run the NVIDIA vGPU driver and be configured to point to a vGPU license server with a valid license.

[1] https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.2/html-single/self-hosted_engine_guide/

[2] https://docs.nvidia.com/grid/latest/grid-vgpu-release-notes-red-hat-el-kvm/

A single Windows 10 virtual machine was created and cloned to create virtual desktops with a Tesla V100-8Q vGPU profile. The Citrix Virtual Desktop (formerly Citrix XenDesktop) receiver was installed in the VDI instances to connect to a Citrix connection broker and provide remote access via the Citrix Receiver client. An Unmanaged Pool was required because Citrix desktop delivery controller does not have access to the KVM hypervisor to manage power state or deployment of virtual machines. Rather the virtual machines were treated like physical workstations when adding them to Citrix. An additional option could have been to use Citrix Provisioning Services to stream a master image to the KVM virtual machine profiles using a boot image to reference the streaming service config.

**Note:** KVM is not on the supported hypervisor list for Citrix Virtual Desktop. Citrix Virtual Desktop and ICA HDX3DPro as a remoting protocol were used for experimentation purposes only and are not a supported configuration.

To create a mixed-workload Linux cluster, a single virtual machine was manually created and provisioned with Red Hat Enterprise Linux 7.5 and cloned into several instances with Tesla V100-4Q vGPU profiles. This virtual cluster was provisioned using Ansible to install and configure Kubernetes as the workload scheduler. Using the NVIDIA container runtime and NVIDIA Kubernetes GPU device plugin, Kubernetes on virtual machines with vGPU can run mixed workloads on shared virtual GPUs.

## Workflow

The experiment was designed to showcase a typical engineering HPC workflow. This was comprised of a three-stage workflow which included a pre-processing, execution, and then post-processing step. Additionally, a typical analyst's workflow was added to execute an inference job against a multi-node virtualized deep learning cluster.

1. Pre-processing

   Execution of the pre-processing stage was performed on a Tesla P100-8Q vGPU-enabled Microsoft Windows 10 virtual machine. This step setup the execution phase by allowing for the creation and testing of the HPC job using a common tool, Altair HyperWorks, to import a CAD model and create the CAE workflow solvers. This was done via HyperMesh to prepare the structural analysis job of a turbine blade before execution.

2. Execution

   Before execution could be performed at an HPC level we first conducted process evaluation testing on a single Tesla P100-8Q vGPU-enabled HPC node. This node was Ubuntu 16.0.4 which could execute an NVIDIA Docker container running an updated CentOS 7.5. While the CentOS 7.4 is the core requirement of the ANSYS Mechanical HPC engine runtime, we inadvertently updated the container. This did not negatively impact the execution but was not supported by ANSYS. Custom

Kubernetes scripts were used to initiate the Docker runtime which would start the CentOS 7.5 Container and then subsequently execute the ANSYS job.

Once single execution was achieved, additional customization was required to the Kubernetes script to split the job across multiple nodes. This also required assigning additional IP addresses to each HPC node Ubuntu VM and statically mapping these IP addresses to each worker node Docker CentOS container. A detailed view of the software infrastructure stack can be found in Figure 2.

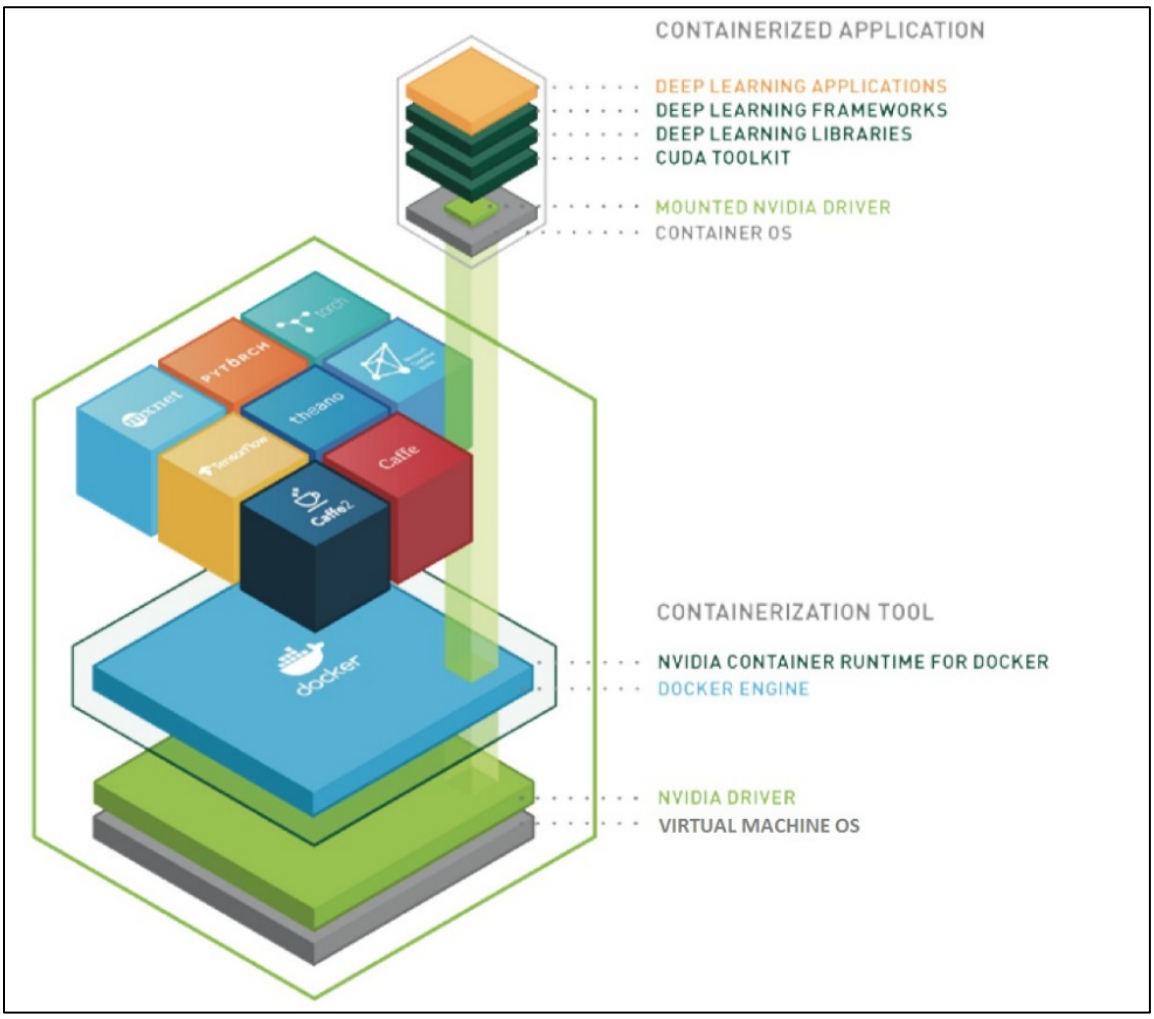
Additionally a multi-node inference cluster was deployed on the hypervisor using Red Hat Ansible and Kubernetes to manage the Ubuntu 16.0.4 virtual machines each with a Tesla P100-8Q vGPU. An Ubuntu 16.0.4 Docker container was deployed on each VM to perform the inference runtime when requested by the client, which was installed on the post-processing virtual machine.

3. Post-processing

Once execution on the HPC was completed, post-process analysis was performed on a Tesla P100-8Q vGPU-enabled Microsoft Windows 10 virtual machine. This specific analysis was performed using Altair HyperWorks HyperView tools. This makes it possible to view the results of the ANSYS Mechanical CAE execution job.

An inference client was installed and used to execute an image classification job against the multi-node inference cluster. This utilized the Tesla P100-8Q vGPU on the Microsoft Windows 10 virtual machine to run an OpenGL based client for graphics rendering. The inference was performed on a multi-node inference cluster enabled with Tesla P100-8Q vGPU to execute CUDA runtimes for Deep Learning processing on TensorFlow. See Figure 3 for a visual explanation of the Deep Learning software infrastructure.

Figure 3: Deep Learning on Virtual Machines



# RESULTS

After detailed testing and experimentation it can be determined that NVIDIA vGPU technology can be utilized as a stable platform for HPC compute, deep learning, and graphics workloads. The relative mix of these workloads simultaneously on a common infrastructure is still subject to the relative needs of the applications being executed on the cluster.

The workloads should always be considered when sizing appropriate resources for their execution. Also contributing factors should be examined such as network and storage resource utilization and their impact on the entire cluster and subsequent user experience.

There is no singularly capable architectural design that can be utilized for all workloads. What can be determined is that there are many HPC, deep learning, and graphics workloads that can be accelerated with NVIDIA virtual GPU technology on hypervisors.

## Notice



## HDMI



## ROVI Compliance Statement

NVIDIA Products that support Rovi Corporation's Revision 7.1.L1 Anti-Copy Process (ACP) encoding technology can only be sold or distributed to buyers with a valid and existing authorization from ROVI to purchase and incorporate the device into buyer's products.

This device is protected by U.S. patent numbers 6,516,132; 5,583,936; 6,836,549; 7,050,698; and 7,492,896 and other intellectual property rights. The use of ROVI Corporation's copy protection technology in the device must be authorized by ROVI Corporation and is intended for home and other limited pay-per-view uses only, unless otherwise authorized in writing by ROVI Corporation. Reverse engineering or disassembly is prohibited.

## Trademarks



## Copyright