



Virtual GPU Software R430 for Linux with KVM

Release Notes

Table of Contents

Chapter 1. Release Notes.....	1
1.1. Updates in Release 9.0.....	1
1.2. Updates in Release 9.1.....	2
1.3. Updates in Release 9.2.....	2
1.4. Updates in Release 9.3.....	2
1.5. Updates in Release 9.4.....	2
Chapter 2. Validated Platforms.....	4
2.1. Supported NVIDIA GPUs and Validated Server Platforms.....	4
2.2. Hypervisor Software Releases.....	5
2.3. Guest OS Support.....	5
2.4. NVIDIA CUDA Toolkit Version Support.....	5
2.5. Multiple vGPU Support.....	6
2.6. Peer-to-Peer CUDA Transfers over NVLink Support.....	7
Chapter 3. Known Product Limitations.....	9
3.1. Issues occur when the channels allocated to a vGPU are exhausted.....	9
3.2. Virtual GPU hot plugging is not supported.....	10
3.3. Total frame buffer for vGPUs is less than the total frame buffer on the physical GPU....	10
3.4. Issues may occur with graphics-intensive OpenCL applications on vGPU types with limited frame buffer.....	11
3.5. In pass through mode, all GPUs connected to each other through NVLink must be assigned to the same VM.....	12
3.6. vGPU profiles with 512 Mbytes or less of frame buffer support only 1 virtual display head on Windows 10.....	12
3.7. NVENC requires at least 1 Gbyte of frame buffer.....	13
3.8. VM running older NVIDIA vGPU drivers fails to initialize vGPU when booted.....	13
3.9. Single vGPU benchmark scores are lower than pass-through GPU.....	14
3.10. nvidia-smi fails to operate when all GPUs are assigned to GPU pass-through mode....	15
Chapter 4. Resolved Issues.....	16
Chapter 5. Known Issues.....	19
5.1. NVIDIA Control Panel fails to start if launched too soon from a VM without licensing information.....	19
5.2. DWM crashes randomly occur in Windows VMs.....	19
5.3. 9.0, 9.1 Only: VMs fail to boot with failed assertions.....	20
5.4. 9.0, 9.1 Only: ECC memory with NVIDIA vGPU is not supported on Tesla M60 and Tesla M6.....	21

5.5. 9.0, 9.1 Only: Virtual GPU fails to start if ECC is enabled.....	22
5.6. 9.0 only: Hypervisor host hangs when persistence mode is set.....	23
5.7. RAPIDS cuDF merge fails on NVIDIA vGPU.....	24
5.8. 9.0 only: Incorrect ECC error counts are reported for vGPUs on some GPUs.....	25
5.9. ECC memory settings for a vGPU cannot be changed by using NVIDIA X Server Settings	25
5.10. Changes to ECC memory settings for a Linux vGPU VM by nvidia-smi might be ignored	26
5.11. 9.0 only: VM crashes after the volatile ECC error count is reset.....	27
5.12. 9.0 only: No vCS option available in NVIDIA X Server Settings.....	27
5.13. 9.0 only: On Linux VMs, the license directory is not deleted when the guest driver is uninstalled.....	29
5.14. Vulkan applications crash in Windows 7 guest VMs configured with NVIDIA vGPU.....	29
5.15. Host core CPU utilization is higher than expected for moderate workloads.....	30
5.16. Frame capture while the interactive logon message is displayed returns blank screen	31
5.17. RDS sessions do not use the GPU with some Microsoft Windows Server releases.....	31
5.18. Even when the scheduling policy is equal share, unequal GPU utilization is reported...	32
5.19. When the scheduling policy is fixed share, GPU utilization is reported as higher than expected.....	33
5.20. License is not acquired in Windows VMs.....	34
5.21. nvidia-smi reports that vGPU migration is supported on all hypervisors.....	35
5.22. Hot plugging and unplugging vCPUs causes a blue-screen crash in Windows VMs.....	35
5.23. Luxmark causes a segmentation fault on an unlicensed Linux client.....	36
5.24. Resolution is not updated after a VM acquires a license and is restarted.....	36
5.25. A segmentation fault in Dbus code causes nvidia-gridd to exit on Red Hat Enterprise Linux and CentOS.....	37
5.26. No Manage License option available in NVIDIA X Server Settings by default.....	37
5.27. Licenses remain checked out when VMs are forcibly powered off.....	38
5.28. VM bug checks after the guest VM driver for Windows 10 RS2 is installed.....	39
5.29. GNOME Display Manager (GDM) fails to start on Red Hat Enterprise Linux 7.2 and CentOS 7.0.....	40

Chapter 1. Release Notes

These *Release Notes* summarize current status, information on validated platforms, and known issues with NVIDIA vGPU software and associated hardware on Linux with KVM.



Note: The most current version of the documentation for this release of NVIDIA vGPU software can be found online at [NVIDIA Virtual GPU Software Documentation](#).

The releases in this release family of NVIDIA vGPU software include the software listed in the following table:

Software	9.0	9.1	9.2	9.3	9.4
NVIDIA Virtual GPU Manager for the Linux with KVM releases listed in Hypervisor Software Releases	Not supported	430.46	430.67	Not supported	Not supported
NVIDIA Windows driver	Not supported	431.79	432.08	Not supported	Not supported
NVIDIA Linux driver	Not supported	430.46	430.63	Not supported	Not supported

All releases of NVIDIA vGPU software are compatible with **all** releases of the license server.

1.1. Updates in Release 9.0

New Features in Release 9.0

- ▶ NVIDIA Virtual Compute Server (V) vGPUs for artificial intelligence, deep learning, and high-performance computing workloads
- ▶ Error correcting code (ECC) memory support
- ▶ Page retirement support
- ▶ Peer-to-peer CUDA transfers over NVLink support
- ▶ Configurable times slices for equal share schedulers and fixed share schedulers
- ▶ New configuration parameter to specify host ID of a licensed client

- ▶ Miscellaneous bug fixes

Hardware and Software Support Introduced in Release 9.0

- ▶ Support for Windows 10 May 2019 Update (1903) as a guest OS

1.2. Updates in Release 9.1

New Features in Release 9.1

- ▶ Support for NVIDIA Virtual Compute Server vGPUs on the following GPUs:
 - ▶ Quadro RTX 6000
 - ▶ Quadro RTX 8000
- ▶ Security updates
- ▶ Miscellaneous bug fixes

1.3. Updates in Release 9.2

New Features in Release 9.2

- ▶ Miscellaneous bug fixes
- ▶ Security updates
- ▶ Limitation on the maximum number of NVIDIA Virtual Compute Server vGPUs to eight vGPUs per physical GPU, irrespective of the available hardware resources of the physical GPU

1.4. Updates in Release 9.3

New Features in Release 9.3

- ▶ Miscellaneous bug fixes
- ▶ Security updates (see [Security Bulletin: NVIDIA GPU Display Driver - February 2020](#))

1.5. Updates in Release 9.4

New Features in Release 9.4

- ▶ Miscellaneous bug fixes

- ▶ Security updates - see [Security Bulletin: NVIDIA GPU Display Driver - June 2020](#)

Chapter 2. Validated Platforms

This release family of NVIDIA vGPU software provides support for several NVIDIA GPUs on validated server hardware platforms, Linux with KVM hypervisor software versions, and guest operating systems. It also supports the version of NVIDIA CUDA Toolkit that is compatible with R430 drivers.

2.1. Supported NVIDIA GPUs and Validated Server Platforms

For information about supported NVIDIA GPUs and the validated server hardware platforms on which they run, consult the documentation from your hypervisor vendor.



Note:

Tesla M60 and M6 GPUs support compute mode and graphics mode. NVIDIA vGPU requires GPUs that support both modes to operate in graphics mode.

Recent Tesla M60 GPUs and M6 GPUs are supplied in graphics mode. However, your GPU might be in compute mode if it is an older Tesla M60 GPU or M6 GPU, or if its mode has previously been changed.

To configure the mode of Tesla M60 and M6 GPUs, use the `gpumodeswitch` tool provided with NVIDIA vGPU software releases.

Even in compute mode, Tesla M60 and M6 GPUs do **not** support NVIDIA Virtual Compute Server vGPU types.

2.2. Hypervisor Software Releases

NVIDIA vGPU software supports **only** Linux with KVM hypervisor software from the vendors listed in the table.



Note: If a specific NVIDIA vGPU software release, even an update release, is not listed, it's **not** supported. For information about which hypervisor software releases are supported, consult the documentation from your hypervisor vendor.

Hypervisor Vendor	9.0	9.1	9.2	9.3	9.4
Easted	-	#	-	-	-
H3C	-	#	#	-	-
Sangfor	-	#	-	-	-
ZTE	-	#	#	-	-

Release is supported.

- Release is **not** supported.

2.3. Guest OS Support

For information about Windows releases and Linux distributions supported as a guest OS, consult the documentation from your hypervisor vendor.



Note:

Use only a guest OS release that is listed as supported by NVIDIA vGPU software with your virtualization software. To be listed as supported, a guest OS release must be supported not only by NVIDIA vGPU software, but also by your virtualization software. NVIDIA **cannot** support guest OS releases that your virtualization software does not support.

NVIDIA vGPU software supports **only** 64-bit guest operating systems. No 32-bit guest operating systems are supported.

2.4. NVIDIA CUDA Toolkit Version Support

The releases in this release family of NVIDIA vGPU software support NVIDIA CUDA Toolkit 10.1 Update 1.

For more information about NVIDIA CUDA Toolkit, see [CUDA Toolkit 10.1 Documentation](#).



Note:

If you are using NVIDIA vGPU software with CUDA on Linux, avoid conflicting installation methods by installing CUDA from a distribution-independent runfile package. Do not install CUDA from distribution-specific RPM or Deb package.

To ensure that the NVIDIA vGPU software graphics driver is not overwritten when CUDA is installed, deselect the CUDA driver when selecting the CUDA components to install.

For more information, see [NVIDIA CUDA Installation Guide for Linux](#).

2.5. Multiple vGPU Support

To support applications and workloads that are compute or graphics intensive, multiple vGPUs can be added to a single VM. The assignment of more than one vGPU to a VM is supported only on a subset of vGPUs and Linux with KVM releases.

Supported vGPUs

Only Q-series and C-series vGPUs that are allocated all of the physical GPU's frame buffer are supported.

GPU Architecture	Board	vGPU
Turing	Tesla T4	T4-16Q
		T4-16C
	Quadro RTX 6000	RTX6000-24Q
	Quadro RTX 8000	RTX8000-48Q
Volta	Tesla V100 SXM2 32GB	V100DX-32Q
		V100D-32C
	Tesla V100 PCIe 32GB	V100D-32Q
		V100D-32C
	Tesla V100 SXM2	V100X-16Q
		V100X-16C
	Tesla V100 PCIe	V100-16Q
		V100-16C
	Tesla V100 FHHL	V100L-16Q
		V100L-16C
Pascal	Tesla P100 SXM2	P100X-16Q
		P100X-16C
	Tesla P100 PCIe 16GB	P100-16Q
		P100-16C
	Tesla P100 PCIe 12GB	P100C-12Q

GPU Architecture	Board	vGPU
		P100C-12C
	Tesla P40	P40-24Q
		P40-24C
	Tesla P6	P6-16Q
		P6-16C
	Tesla P4	P4-8Q
P4-8C		
Maxwell	Tesla M60	M60-8Q
	Tesla M10	M10-8Q
	Tesla M6	M6-8Q

Maximum vGPUs per VM

NVIDIA vGPU software supports up to a maximum of 16 vGPUs per VM on Linux with KVM.

Supported Hypervisor Releases

For information about which hypervisor software releases support the assignment of more than one vGPU device to a VM, consult the documentation from your hypervisor vendor.

2.6. Peer-to-Peer CUDA Transfers over NVLink Support

Peer-to-peer CUDA transfers enable device memory between vGPUs on different GPUs that are assigned to the same VM to be accessed from within the CUDA kernels. NVLink is a high-bandwidth interconnect that enables fast communication between such vGPUs. Peer-to-Peer CUDA Transfers over NVLink is supported only on a subset of vGPUs, Linux with KVM releases, and guest OS releases.

Supported vGPUs

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

GPU Architecture	Board	vGPU
Turing	Quadro RTX 6000	RTX6000-24Q
	Quadro RTX 8000	RTX8000-48Q
Volta	Tesla V100 SXM2 32GB	V100DX-32Q
		V100DX-32C

GPU Architecture	Board	vGPU
	Tesla V100 SXM2	V100X-16Q
		V100X-16C
Pascal	Tesla P100 SXM2	P100X-16Q
		P100X-16C

Supported Hypervisor Releases

Peer-to-Peer CUDA Transfers over NVLink are supported on all hypervisor releases that support the assignment of more than one vGPU to a VM. For details, see [Multiple vGPU Support](#).

Supported Guest OS Releases

Linux only. Peer-to-Peer CUDA Transfers over NVLink are **not** supported on Windows.

Limitations

- ▶ Only direct connections are supported. NVSwitch is not supported.
- ▶ PCIe is not supported.
- ▶ SLI is not supported.

Chapter 3. Known Product Limitations

Known product limitations for this release of NVIDIA vGPU software are described in the following sections.

3.1. Issues occur when the channels allocated to a vGPU are exhausted

Description

Issues occur when the channels allocated to a vGPU are exhausted and the guest VM to which the vGPU is assigned fails to allocate a channel to the vGPU. A physical GPU has a fixed number of channels and the number of channels allocated to each vGPU is inversely proportional to the maximum number of vGPUs allowed on the physical GPU.

When the channels allocated to a vGPU are exhausted and the guest VM fails to allocate a channel, the following errors are reported on the hypervisor host or in an NVIDIA bug report:

```
Jun 26 08:01:25 srvxen06f vgpu-3[14276]: error: vmiop_log: (0x0): Guest attempted to
allocate channel above its max channel limit 0xfb
Jun 26 08:01:25 srvxen06f vgpu-3[14276]: error: vmiop_log: (0x0): vGPU message 6
failed, result code: 0x1a
Jun 26 08:01:25 srvxen06f vgpu-3[14276]: error: vmiop_log: (0x0):
0xc1d004a1, 0xff0e0000, 0xff0400fb, 0xc36f,
Jun 26 08:01:25 srvxen06f vgpu-3[14276]: error: vmiop_log: (0x0):          0x1,
0xff1fe314, 0xff1fe038, 0x100b6f000, 0x1000,
Jun 26 08:01:25 srvxen06f vgpu-3[14276]: error: vmiop_log: (0x0):
0x80000000, 0xff0e0200, 0x0, 0x0, (Not logged),
Jun 26 08:01:25 srvxen06f vgpu-3[14276]: error: vmiop_log: (0x0):          0x1, 0x0
Jun 26 08:01:25 srvxen06f vgpu-3[14276]: error: vmiop_log: (0x0): , 0x0
```

Workaround

Use a vGPU type with more frame buffer, thereby reducing the maximum number of vGPUs allowed on the physical GPU. As a result, the number of channels allocated to each vGPU is increased.

3.2. Virtual GPU hot plugging is not supported

NVIDIA vGPU software does not support the addition of virtual function I/O (VFIO) mediated device (`mdev`) devices after the VM has been started by QEMU. All `mdev` devices must be added before the VM is started.

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

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

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

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

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

$$\text{max-reserved-fb} = \text{vgpu-profile-size-in-mb} \div 16 + 16 + \text{ecc-adjustments} + \text{page-retirement-allocation}$$

max-reserved-fb

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

vgpu-profile-size-in-mb

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

ecc-adjustments

The amount of frame buffer in Mbytes that is not usable by vGPUs when ECC is enabled on a physical GPU that does not have HBM2 memory. If ECC is disabled or the GPU has HBM2 memory, `ecc-adjustments` is 0.

page-retirement-allocation

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

- ▶ On GPUs based on the NVIDIA Maxwell GPU architecture, *page-retirement-allocation* = $4 \div \text{max-vgpus-per-gpu}$.
- ▶ On GPUs based on NVIDIA GPU architectures **after** the Maxwell architecture, *page-retirement-allocation* = $128 \div \text{max-vgpus-per-gpu}$

max-vgpus-per-gpu

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



Note: In VMs running a Windows guest OS that supports Windows Display Driver Model (WDDM) 1.x, namely, Windows 7, Windows 8.1, Windows Server 2008, and Windows Server 2012, an additional 48 Mbytes of frame buffer are reserved and not available for vGPUs.

3.4. Issues may occur with graphics-intensive OpenCL applications on vGPU types with limited frame buffer

Description

Issues may occur when graphics-intensive OpenCL applications are used with vGPU types that have limited frame buffer. These issues occur when the applications demand more frame buffer than is allocated to the vGPU.

For example, these issues may occur with the Adobe Photoshop and LuxMark OpenCL Benchmark applications:

- ▶ When the image resolution and size are changed in Adobe Photoshop, a program error may occur or Photoshop may display a message about a problem with the graphics hardware and a suggestion to disable OpenCL.
- ▶ When the LuxMark OpenCL Benchmark application is run, XID error 31 may occur.

Workaround

For graphics-intensive OpenCL applications, use a vGPU type with more frame buffer.

3.5. In pass through mode, all GPUs connected to each other through NVLink must be assigned to the same VM

Description

In pass through mode, all GPUs connected to each other through NVLink must be assigned to the same VM. If a subset of GPUs connected to each other through NVLink is passed through to a VM, unrecoverable error `XID 74` occurs when the VM is booted. This error corrupts the NVLink state on the physical GPUs and, as a result, the NVLink bridge between the GPUs is unusable.

Workaround

Restore the NVLink state on the physical GPUs by resetting the GPUs or rebooting the hypervisor host.

3.6. vGPU profiles with 512 Mbytes or less of frame buffer support only 1 virtual display head on Windows 10

Description

To reduce the possibility of memory exhaustion, vGPU profiles with 512 Mbytes or less of frame buffer support only 1 virtual display head on a Windows 10 guest OS.

The following vGPU profiles have 512 Mbytes or less of frame buffer:

- ▶ Tesla M6-0B, M6-0Q
- ▶ Tesla M10-0B, M10-0Q
- ▶ Tesla M60-0B, M60-0Q

Workaround

Use a profile that supports more than 1 virtual display head and has at least 1 Gbyte of frame buffer.

3.7. NVENC requires at least 1 Gbyte of frame buffer

Description

Using the frame buffer for the NVIDIA hardware-based H.264/HEVC video encoder (NVENC) may cause memory exhaustion with vGPU profiles that have 512 Mbytes or less of frame buffer. To reduce the possibility of memory exhaustion, NVENC is disabled on profiles that have 512 Mbytes or less of frame buffer. Application GPU acceleration remains fully supported and available for all profiles, including profiles with 512 Mbytes or less of frame buffer. NVENC support from both Citrix and VMware is a recent feature and, if you are using an older version, you should experience no change in functionality.

The following vGPU profiles have 512 Mbytes or less of frame buffer:

- ▶ Tesla M6-0B, M6-0Q
- ▶ Tesla M10-0B, M10-0Q
- ▶ Tesla M60-0B, M60-0Q

Workaround

If you require NVENC to be enabled, use a profile that has at least 1 Gbyte of frame buffer.

3.8. VM running older NVIDIA vGPU drivers fails to initialize vGPU when booted

Description

A VM running a version of the NVIDIA guest VM drivers from a previous main release branch, for example release 4.4, will fail to initialize vGPU when booted on a Linux with KVM platform running the current release of Virtual GPU Manager.

In this scenario, the VM boots in standard VGA mode with reduced resolution and color depth. The NVIDIA virtual GPU is present in **Windows Device Manager** but displays a warning sign, and the following device status:

```
Windows has stopped this device because it has reported problems. (Code 43)
```

Depending on the versions of drivers in use, the Linux with KVM VM's `/var/log/messages` log file reports one of the following errors:

- ▶ An error message:

```
vmiop_log: error: Unable to fetch Guest NVIDIA driver information
```

- ▶ A version mismatch between guest and host drivers:

```
vmiop_log: error: Guest VGX version(1.1) and Host VGX version(1.2) do not match
```

- ▶ A signature mismatch:

```
vmiop_log: error: VGPU message signature mismatch.
```

Resolution

Install the current NVIDIA guest VM driver in the VM.

3.9. Single vGPU benchmark scores are lower than pass-through GPU

Description

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

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

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

Resolution

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

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

For example:

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

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

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

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

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

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



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

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

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

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

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

3.10. `nvidia-smi` fails to operate when all GPUs are assigned to GPU pass-through mode

Description

If all GPUs in the platform are assigned to VMs in pass-through mode, `nvidia-smi` will return an error:

```
[root@vgx-test ~]# nvidia-smi
Failed to initialize NVML: Unknown Error
```

This is because GPUs operating in pass-through mode are not visible to `nvidia-smi` and the NVIDIA kernel driver operating in the Linux with KVM host.

To confirm that all GPUs are operating in pass-through mode, confirm that the `vfio-pci` kernel driver is handling each device.

```
# lspci -s 05:00.0 -k
05:00.0 VGA compatible controller: NVIDIA Corporation GM204GL [Tesla M60] (rev a1)
Subsystem: NVIDIA Corporation Device 113a
Kernel driver in use: vfio-pci
```

Resolution

N/A

Chapter 4. Resolved Issues

Only resolved issues that have been previously noted as known issues or had a noticeable user impact are listed. The summary and description for each resolved issue indicate the effect of the issue on NVIDIA vGPU software **before the issue was resolved**.

Issues Resolved in Release 9.0

Bug ID	Summary and Description
-	<p>Virtual GPU fails to start if ECC is enabled</p> <p>NVIDIA vGPU does not support error correcting code (ECC) memory. If ECC memory is enabled, NVIDIA vGPU fails to start.</p> <p>Starting with NVIDIA vGPU software release 9.0, NVIDIA vGPU supports ECC memory on GPUs and hypervisor software versions that support ECC.</p>

Issues Resolved in Release 9.1

Bug ID	Summary and Description
200534988	<p>Error XID 47 followed by multiple XID 32 errors</p> <p>After disconnecting Citrix Virtual Apps and Desktops and clicking the power button in the VM, error XID 47 occurs followed by multiple XID 32 errors. When these errors occur, the hypervisor host becomes unusable.</p>
200537078	<p><u>9.0 only: Hypervisor host hangs when persistence mode is set</u></p> <p>When persistence mode is set through the <code>nvidia-smi</code> command, a very large number of <code>nv_queue</code> worker threads is spawned, causing the hypervisor host to hang.</p>
200526633	<p><u>9.0 only: VM crashes after the volatile ECC error count is reset</u></p> <p>After the command <code>nvidia-smi -p 0</code> is run from a guest VM to reset the volatile ECC error count, the VM crashes.</p>
200525006	<p><u>9.0 only: Incorrect ECC error counts are reported for vGPUs on some GPUs</u></p>

Bug ID	Summary and Description
	Incorrect ECC error counts are reported for vGPUs on some GPUs when the command <code>nvidia-smi -q</code> is run from a guest VM.
200524555	<p><u>9.0 only: On Linux VMs, the license directory is not deleted when the guest driver is uninstalled</u></p> <p>On Linux guest VMs, the license directory <code>/etc/nvidia/license</code> is not deleted when the NVIDIA vGPU software graphics driver is uninstalled.</p>
200522255	<p><u>9.0 only: No vCS option available in NVIDIA X Server Settings</u></p> <p>The vCS option is missing from the Manage License section in the NVIDIA X Server Settings window.</p>

Issues Resolved in Release 9.2

Bug ID	Summary and Description
2644858	<p><u>9.0, 9.1 Only: VMs fail to boot with failed assertions</u></p> <p>In some scenarios with heavy workloads running on multiple VMs configured with NVIDIA vGPUs on a single physical GPU, additional VMs configured with NVIDIA vGPU on the same GPU fail to boot. The failure of the VM to boot is followed by failed assertions. This issue affects GPUs based on the NVIDIA Volta GPU architecture and later architectures.</p>
-	<p><u>9.0, 9.1 Only: ECC memory with NVIDIA vGPU is not supported on Tesla M60 and Tesla M6</u></p> <p>Error-correcting code (ECC) memory with NVIDIA vGPU is not supported on Tesla M60 and Tesla M6 GPUs. The effect of starting NVIDIA vGPU when it is configured on a Tesla M60 or Tesla M6 GPU on which ECC memory is enabled depends on your NVIDIA vGPU software release.</p> <ul style="list-style-type: none"> ▶ 9.0 only: The hypervisor host fails. ▶ 9.1 only: The VM fails to start.
-	<p><u>9.0, 9.1 Only: Virtual GPU fails to start if ECC is enabled</u></p> <p>NVIDIA vGPU does not support ECC memory with the following GPUs:</p> <ul style="list-style-type: none"> ▶ Tesla M60 GPUs ▶ Tesla M6 GPUs <p>If ECC memory is enabled and your GPU does not support ECC, NVIDIA vGPU fails to start.</p>

Issues Resolved in Release 9.3

No resolved issues are reported in this release for Linux with KVM.

Issues Resolved in Release 9.4

No resolved issues are reported in this release for Linux with KVM.

Chapter 5. Known Issues

5.1. NVIDIA Control Panel fails to start if launched too soon from a VM without licensing information

Description

If NVIDIA licensing information is not configured on the system, any attempt to start **NVIDIA Control Panel** by right-clicking on the desktop within 30 seconds of the VM being started fails.

Workaround

Wait at least 30 seconds before trying to launch **NVIDIA Control Panel**.

Status

Open

Ref.

200623179

5.2. DWM crashes randomly occur in Windows VMs

Description

Desktop Windows Manager (DWM) crashes randomly occur in Windows VMs, causing a blue-screen crash and the bug check `CRITICAL_PROCESS_DIED`. Computer Management shows problems with the primary display device.

Version

This issue affects Windows 10 1809, 1903 and 1909 VMs.

Status

Not an NVIDIA bug

Ref.

2730037

5.3. 9.0, 9.1 Only: VMs fail to boot with failed assertions

Description

In some scenarios with heavy workloads running on multiple VMs configured with NVIDIA vGPUs on a single physical GPU, additional VMs configured with NVIDIA vGPU on the same GPU fail to boot. The failure of the VM to boot is followed by failed assertions. This issue affects GPUs based on the NVIDIA Volta GPU architecture and later architectures.

When this error occurs, error messages similar to the following examples are logged to the Linux with KVM log file:

```
nvidia-vgpu-mgr[31526]: error: vmiop_log: NVOS status 0x1e
nvidia-vgpu-mgr[31526]: error: vmiop_log: Assertion Failed at 0xb2d3e4d7:96
nvidia-vgpu-mgr[31526]: error: vmiop_log: 12 frames returned by backtrace
nvidia-vgpu-mgr[31526]: error: vmiop_log: /usr/lib64/libnvidia-vgpu.so(_nv003956vgpu
+0x18) [0x7f4bb2cfb338] vmiop_dump_stack
nvidia-vgpu-mgr[31526]: error: vmiop_log: /usr/lib64/libnvidia-vgpu.so(_nv004018vgpu
+0xd4) [0x7f4bb2d09ce4] vmiopd_alloc_pb_channel
nvidia-vgpu-mgr[31526]: error: vmiop_log: /usr/lib64/libnvidia-vgpu.so(_nv002878vgpu
+0x137) [0x7f4bb2d3e4d7] vgpufceInitCopyEngine_GK104
nvidia-vgpu-mgr[31526]: error: vmiop_log: /usr/lib64/libnvidia-vgpu.so(+0x80e27)
[0x7f4bb2cd0e27]
nvidia-vgpu-mgr[31526]: error: vmiop_log: /usr/lib64/libnvidia-vgpu.so(+0x816a7)
[0x7f4bb2cd16a7]
nvidia-vgpu-mgr[31526]: error: vmiop_log: vgpu() [0x413820]
nvidia-vgpu-mgr[31526]: error: vmiop_log: vgpu() [0x413a8d]
nvidia-vgpu-mgr[31526]: error: vmiop_log: vgpu() [0x40e11f]
nvidia-vgpu-mgr[31526]: error: vmiop_log: vgpu() [0x40bb69]
nvidia-vgpu-mgr[31526]: error: vmiop_log: vgpu() [0x40b51c]
nvidia-vgpu-mgr[31526]: error: vmiop_log: /lib64/libc.so.6(__libc_start_main+0x100)
[0x7f4bb2feed20]
nvidia-vgpu-mgr[31526]: error: vmiop_log: vgpu() [0x4033ea]
nvidia-vgpu-mgr[31526]: error: vmiop_log: (0x0): Alloc Channel(Gpfifo) for device
failed error: 0x1e
nvidia-vgpu-mgr[31526]: error: vmiop_log: (0x0): Failed to allocate FCE channel
nvidia-vgpu-mgr[31526]: error: vmiop_log: (0x0): init_device_instance failed for
inst 0 with error 2 (init frame copy engine)
nvidia-vgpu-mgr[31526]: error: vmiop_log: (0x0): Initialization:
init_device_instance failed error 2
nvidia-vgpu-mgr[31526]: error: vmiop_log: display_init failed for inst: 0
```

```
nvidia-vgpu-mgr[31526]: error: vmiop_env_log: (0x0): vmiope_process_configuration:
  plugin registration error
nvidia-vgpu-mgr[31526]: error: vmiop_env_log: (0x0): vmiope_process_configuration
  failed with 0x1a
kernel: [858113.083773] [nvidia-vgpu-vfio] ace3f3bb-17d8-4587-920e-199b8fed532d:
  start failed. status: 0x1
```

Status

Resolved in NVIDIA vGPU software 9.2.

Ref.

2644858

5.4. 9.0, 9.1 Only: ECC memory with NVIDIA vGPU is not supported on Tesla M60 and Tesla M6

Description

Error-correcting code (ECC) memory with NVIDIA vGPU is not supported on Tesla M60 and Tesla M6 GPUs. The effect of starting NVIDIA vGPU when it is configured on a Tesla M60 or Tesla M6 GPU on which ECC memory is enabled depends on your NVIDIA vGPU software release.

- ▶ **9.0 only:** The hypervisor host fails.
- ▶ **9.1 only:** The VM fails to start.

Workaround

Ensure that ECC memory is disabled on Tesla M60 and Tesla M6 GPUs. For more information, see [9.0, 9.1 Only: Virtual GPU fails to start if ECC is enabled](#).

Status

Resolved in NVIDIA vGPU software 9.2

5.5. 9.0, 9.1 Only: Virtual GPU fails to start if ECC is enabled

Description

Tesla M60, Tesla M6, and GPUs based on the Pascal GPU architecture, for example Tesla P100 or Tesla P4, support error correcting code (ECC) memory for improved data integrity. Tesla M60 and M6 GPUs in graphics mode are supplied with ECC memory disabled by default, but it may subsequently be enabled using `nvidia-smi`. GPUs based on the Pascal GPU architecture are supplied with ECC memory enabled.

NVIDIA vGPU does not support ECC memory with the following GPUs:

- ▶ Tesla M60 GPUs
- ▶ Tesla M6 GPUs

If ECC memory is enabled and your GPU does not support ECC, NVIDIA vGPU fails to start.

The following error is logged in the Linux with KVM host's `/var/log/messages` log file:

```
vmiop_log: error: Initialization: VGX not supported with ECC Enabled.
```

Workaround

If you are using Tesla M60 or Tesla M6 GPUs, ensure that ECC is disabled on all GPUs.

Before you begin, ensure that NVIDIA Virtual GPU Manager is installed on your hypervisor.

1. Use `nvidia-smi` to list the status of all GPUs, and check for ECC noted as enabled on GPUs.

```
# nvidia-smi -q
=====NVSMI LOG=====

Timestamp                : Tue Dec 19 18:36:45 2017
Driver Version           : 384.99

Attached GPUs            : 1
GPU 0000:02:00.0

[...]

  Ecc Mode
    Current                : Enabled
    Pending                 : Enabled

[...]
```

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

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

```
# nvidia-smi -e 0
```

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

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

id is the index of the GPU 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.
4. Confirm that ECC is now disabled for the GPU.

```
# nvidia-smi -q
=====NVSMI LOG=====

Timestamp                : Tue Dec 19 18:37:53 2017
Driver Version           : 384.99

Attached GPUs            : 1
GPU 0000:02:00.0
[...]

  Ecc Mode
  Current                : Disabled
  Pending                : Disabled

[...]
```

If you later need to enable ECC on your GPUs, run one of the following commands:

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

```
# nvidia-smi -e 1
```

- ▶ If you want to change the ECC status to on for a specific GPU, run this command:

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

id is the index of the GPU 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
```

After changing the ECC status to on, reboot the host.

Status

Resolved in NVIDIA vGPU software 9.2

5.6. 9.0 only: Hypervisor host hangs when persistence mode is set

Description

When persistence mode is set through the `nvidia-smi` command, a very large number of `nv_queue` worker threads is spawned, causing the hypervisor host to hang.

When this issue occurs, the hypervisor host creates a kernel stack trace similar to the following example.

```
Jun 24 04:42:16 example-02 kernel: kernel BUG at /build/linux-Ue9GXV/linux-4.15.0/
drivers/pci/msi.c:352!
...
Jun 24 04:42:16 example-02 kernel: Call Trace:
Jun 24 04:42:16 example-02 kernel: pci_disable_msi+0x100/0x120
Jun 24 04:42:16 example-02 kernel: nv_open_device+0x411/0x8e0 [nvidia]
Jun 24 04:42:16 example-02 kernel: ? kmem_cache_alloc+0x158/0x1b0
Jun 24 04:42:16 example-02 kernel: nvidia_open+0x339/0x550 [nvidia]
Jun 24 04:42:16 example-02 kernel: ? lookup_fast+0xcc/0x320
Jun 24 04:42:16 example-02 kernel: nvidia_frontend_open+0x58/0xa0 [nvidia]
Jun 24 04:42:16 example-02 kernel: chrdev_open+0xc4/0x1b0
```

Status

Resolved in NVIDIA vGPU software 9.1

Ref.

200537078

5.7. RAPIDS cuDF merge fails on NVIDIA vGPU

Description

The merge function of the RAPIDS cuDF GPU data frame library fails on NVIDIA vGPU. This function fails because RAPIDS uses the Unified Memory feature of CUDA, which NVIDIA vGPU does not support.

Status

Open

Ref.

2642134

5.8. 9.0 only: Incorrect ECC error counts are reported for vGPUs on some GPUs

Description

Incorrect ECC error counts are reported for vGPUs on some GPUs when the command `nvidia-smi -q` is run from a guest VM.

This issue affects only vGPUs that reside on physical GPUs based on the NVIDIA Volta GPU architecture. For vGPUs on GPUs based on other architectures, the ECC error count is correct.

Status

Resolved in NVIDIA vGPU software 9.1

Ref.

200525006

5.9. ECC memory settings for a vGPU cannot be changed by using NVIDIA X Server Settings

Description

The ECC memory settings for a vGPU cannot be changed from a Linux guest VM by using **NVIDIA X Server Settings**. After the ECC memory state has been changed on the **ECC Settings** page and the VM has been rebooted, the ECC memory state remains unchanged.

Workaround

Use the `nvidia-smi` command in the guest VM to enable or disable ECC memory for the vGPU as explained in [Virtual GPU Software User Guide](#).

If the ECC memory state remains unchanged even after you use the `nvidia-smi` command to change it, use the workaround in [Changes to ECC memory settings for a Linux vGPU VM by nvidia-smi might be ignored](#).

Status

Open

Ref.

200523086

5.10. Changes to ECC memory settings for a Linux vGPU VM by `nvidia-smi` might be ignored

Description

After the ECC memory state for a Linux vGPU VM has been changed by using the `nvidia-smi` command and the VM has been rebooted, the ECC memory state might remain unchanged.

This issue occurs when multiple NVIDIA configuration files in the system cause the kernel module option for setting the ECC memory state `RMGuestECCState` in `/etc/modprobe.d/nvidia.conf` to be ignored.

When the `nvidia-smi` command is used to enable ECC memory, the file `/etc/modprobe.d/nvidia.conf` is created or updated to set the kernel module option `RMGuestECCState`. Another configuration file in `/etc/modprobe.d/` that contains the keyword `NVreg_RegistryDwordsPerDevice` might cause the kernel module option `RMGuestECCState` to be ignored.

Workaround

This workaround requires administrator privileges.

1. Move the entry containing the keyword `NVreg_RegistryDwordsPerDevice` from the other configuration file to `/etc/modprobe.d/nvidia.conf`.
2. Reboot the VM.

Status

Open

Ref.

200505777

5.11. 9.0 only: VM crashes after the volatile ECC error count is reset

Description

After the command `nvidia-smi -p 0` is run from a guest VM to reset the volatile ECC error count, the VM crashes.

This issue does not occur if the EEC state in the VM is set to off.

Status

Resolved in NVIDIA vGPU software 9.1

Ref.

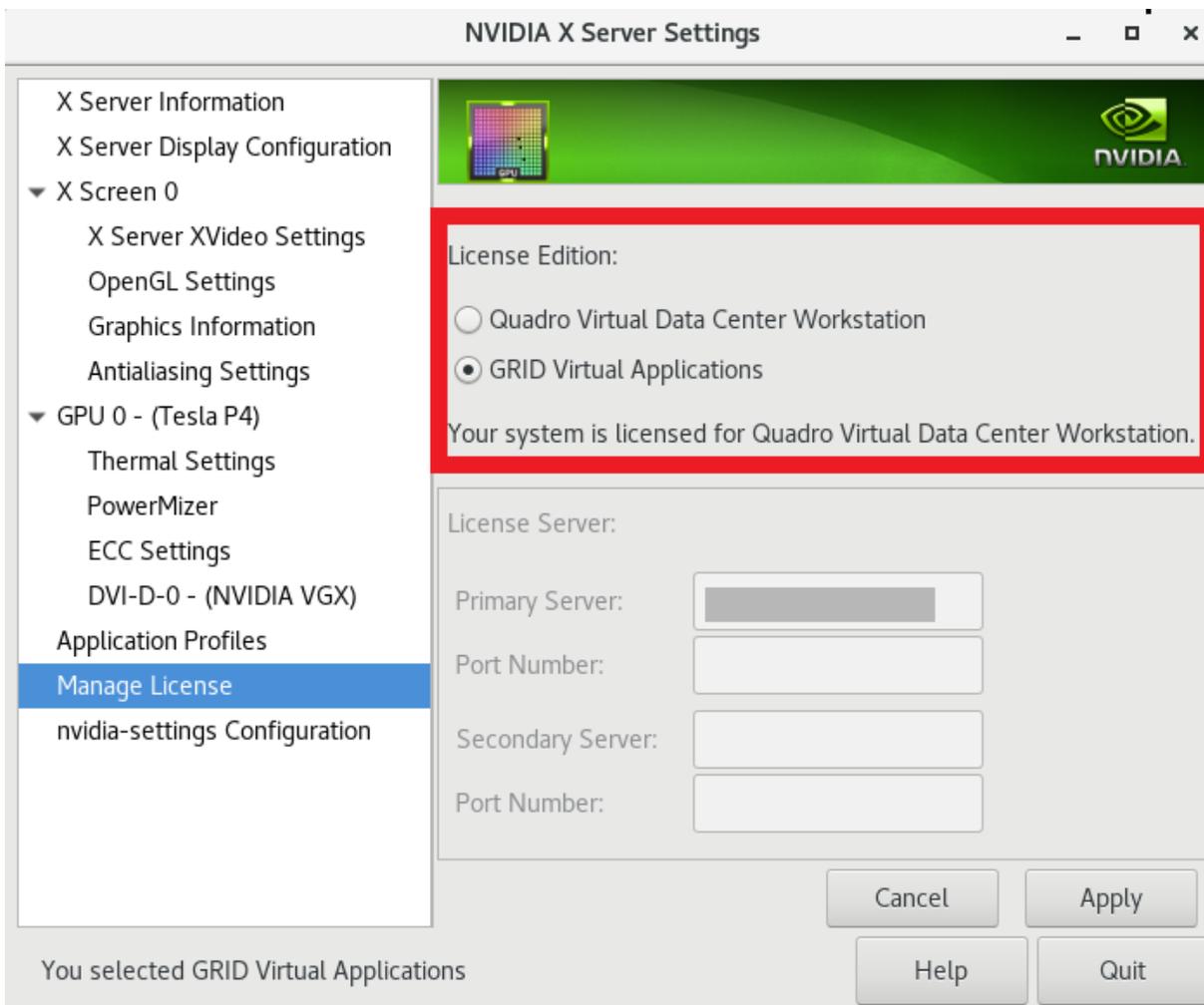
200526633

5.12. 9.0 only: No vCS option available in NVIDIA X Server Settings

Description

The **vCS** option is missing from the **Manage License** section in the **NVIDIA X Server Settings** window.

As a result of this missing option, the **NVIDIA X Server Settings** window incorrectly states that the system is licensed for Quadro vDWS when, in fact, the system is licensed for vCS.



Workaround

If you are licensing a physical GPU for vCS, you **must** use the configuration file `/etc/nvidia/gridd.conf`. See [Virtual GPU Client Licensing User Guide](#).

Status

Resolved in NVIDIA vGPU software 9.1

Ref.

200522255

5.13. 9.0 only: On Linux VMs, the license directory is not deleted when the guest driver is uninstalled

Description

On Linux guest VMs, the license directory `/etc/nvidia/license` is not deleted when the NVIDIA vGPU software graphics driver is uninstalled.

The following error message is written to the `nvidia-uninstaller` log file:

```
Failed to delete the directory '/etc/nvidia' (Directory not empty).
```

Workaround

As root, remove the `/etc/nvidia/license` directory after the NVIDIA vGPU software graphics driver is uninstalled.

Status

Resolved in NVIDIA vGPU software 9.1

Ref.

200524555

5.14. Vulkan applications crash in Windows 7 guest VMs configured with NVIDIA vGPU

Description

In Windows 7 guest VMs configured with NVIDIA vGPU, applications developed with Vulkan APIs crash or throw errors when they are launched. Vulkan APIs require sparse texture support, but in Windows 7 guest VMs configured with NVIDIA vGPU, sparse textures are not enabled.

In Windows 10 guest VMs configured with NVIDIA vGPU, sparse textures are enabled and applications developed with Vulkan APIs run correctly in these VMs.

Status

Open

Ref.

200381348

5.15. Host core CPU utilization is higher than expected for moderate workloads

Description

When GPU performance is being monitored, host core CPU utilization is higher than expected for moderate workloads. For example, host CPU utilization when only a small number of VMs are running is as high as when several times as many VMs are running.

Workaround

Disable monitoring of the following GPU performance statistics:

- ▶ vGPU engine usage by applications across multiple vGPUs
- ▶ Encoder session statistics
- ▶ Frame buffer capture (FBC) session statistics
- ▶ Statistics gathered by performance counters in guest VMs

Status

Open

Ref.

2414897

5.16. Frame capture while the interactive logon message is displayed returns blank screen

Description

Because of a known limitation with NvFBC, a frame capture while the interactive logon message is displayed returns a blank screen.

An NvFBC session can capture screen updates that occur after the session is created. Before the logon message appears, there is no screen update after the message is shown and, therefore, a black screen is returned instead. If the NvFBC session is created after this update has occurred, NvFBC cannot get a frame to capture.

Workaround

Press **Enter** or wait for the screen to update for NvFBC to capture the frame.

Status

Not a bug

Ref.

2115733

5.17. RDS sessions do not use the GPU with some Microsoft Windows Server releases

Description

When some releases of Windows Server are used as a guest OS, Remote Desktop Services (RDS) sessions do not use the GPU. With these releases, the RDS sessions by default use the Microsoft Basic Render Driver instead of the GPU. This default setting enables 2D DirectX applications such as Microsoft Office to use software rendering, which can be more efficient than using the GPU for rendering. However, as a result, 3D applications that use DirectX are prevented from using the GPU.

Version

- ▶ Windows Server 2016
- ▶ Windows Server 2012

Solution

Change the local computer policy to use the hardware graphics adapter for all RDS sessions.

1. Choose **Local Computer Policy > Computer Configuration > Administrative Templates > Windows Components > Remote Desktop Services > Remote Desktop Session Host > Remote Session Environment**.
2. Set the **Use the hardware default graphics adapter for all Remote Desktop Services sessions** option.

5.18. Even when the scheduling policy is equal share, unequal GPU utilization is reported

Description

When the scheduling policy is equal share, unequal GPU engine utilization can be reported for the vGPUs on the same physical GPU.

For example, GPU engine usage for three P40-8Q vGPUs on a Tesla P40 GPU might be reported as follows:

```
[root@localhost:~] nvidia-smi vgpu
Wed Jun 27 10:33:18 2018
+-----+-----+-----+-----+
| NVIDIA-SMI 390.59 | Driver Version: 390.59 |
+-----+-----+-----+-----+
| GPU  Name          | Bus-Id                  | GPU-Util  |
| vGPU ID   Name     | VM ID   VM Name           | vGPU-Util |
+-----+-----+-----+-----+
| 0  Tesla P40      | 00000000:81:00.0       | 52%       |
| 2122661  GRID P40-8Q | 2122682 centos7.4-xmpl-211... | 19%       |
| 2122663  GRID P40-8Q | 2122692 centos7.4-xmpl-211... | 0%        |
| 2122659  GRID P40-8Q | 2122664 centos7.4-xmpl-211... | 25%       |
+-----+-----+-----+-----+
| 1  Tesla P40      | 00000000:85:00.0       | 58%       |
| 2122662  GRID P40-8Q | 2122689 centos7.4-xmpl-211... | 0%        |
| 2122658  GRID P40-8Q | 2122667 centos7.4-xmpl-211... | 59%     |
| 2122660  GRID P40-8Q | 2122670 centos7.4-xmpl-211... | 0%        |
+-----+-----+-----+-----+
```

The vGPU utilization of the vGPU 2122658 is reported as 59%. However, the expected vGPU utilization should not exceed 33%.

This behavior is a result of the mechanism that is used to measure GPU engine utilization.

Status

Open

Ref.

2175888

5.19. When the scheduling policy is fixed share, GPU utilization is reported as higher than expected

Description

When the scheduling policy is fixed share, GPU engine utilization can be reported as higher than expected for a vGPU.

For example, GPU engine usage for six P40-4Q vGPUs on a Tesla P40 GPU might be reported as follows:

```
[root@localhost:~] nvidia-smi vgpu
Mon Aug 20 10:33:18 2018
```

NVIDIA-SMI 390.42		Driver Version: 390.42	
GPU	Name	Bus-Id	GPU-Util
vGPU ID	Name	VM ID	vGPU-Util
0	Tesla P40	00000000:81:00.0	99%
85109	GRID P40-4Q	85110	32%
87195	GRID P40-4Q	87196	39%
88095	GRID P40-4Q	88096	26%
89170	GRID P40-4Q	89171	0%
90475	GRID P40-4Q	90476	0%
93363	GRID P40-4Q	93364	0%
1	Tesla P40	00000000:85:00.0	0%

The vGPU utilization of vGPU 85109 is reported as 32%. For vGPU 87195, vGPU utilization is reported as 39%. And for 88095, it is reported as 26%. However, the expected vGPU utilization of any vGPU should not exceed approximately 16.7%.

This behavior is a result of the mechanism that is used to measure GPU engine utilization.

Status

Open

Ref.

2227591

5.20. License is not acquired in Windows VMs

Description

When a windows VM configured with a licensed vGPU is started, the VM fails to acquire a license.

Error messages in the following format are written to the NVIDIA service logs:

```
[000000020.860152600 sec] - [Logging.lib] ERROR: [nvGridLicensing.FlexUtility]
353@FlexUtility::LogFneError : Error: Failed to add trusted storage. Server
URL : license-server-url -
[1,7E2,2,1[7000003F,0,9B00A7]]
```

System machine type does not match expected machine type..

Workaround

This workaround requires administrator privileges.

1. Stop the **NVIDIA Display Container LS** service.
2. Delete the contents of the folder %SystemDrive%\Program Files\NVIDIA Corporation\Grid Licensing.
3. Start the **NVIDIA Display Container LS** service.

Status

Closed

Ref.

200407287

5.21. `nvidia-smi` reports that vGPU migration is supported on all hypervisors

Description

The command `nvidia-smi vgpu -m` shows that vGPU migration is supported on all hypervisors, even hypervisors or hypervisor versions that do not support vGPU migration.

Status

Closed

Ref.

200407230

5.22. Hot plugging and unplugging vCPUs causes a blue-screen crash in Windows VMs

Description

Hot plugging or unplugging vCPUs causes a blue-screen crash in Windows VMs that are running NVIDIA vGPU software graphics drivers.

When the blue-screen crash occurs, one of the following error messages may also be seen:

- ▶ `SYSTEM_SERVICE_EXCEPTION (nvlddmkm.sys)`
- ▶ `DRIVER_IRQL_NOT_LESS_OR_EQUAL (nvlddmkm.sys)`

NVIDIA vGPU software graphics drivers do not support hot plugging and unplugging of vCPUs.

Status

Closed

Ref.

2101499

5.23. Luxmark causes a segmentation fault on an unlicensed Linux client

Description

If the Luxmark application is run on a Linux guest VM configured with NVIDIA vGPU that is booted without acquiring a license, a segmentation fault occurs and the application core dumps. The fault occurs when the application cannot allocate a CUDA object on NVIDIA vGPUs where CUDA is disabled. On NVIDIA vGPUs that can support CUDA, CUDA is disabled in unlicensed mode.

Status

Not an NVIDIA bug.

Ref.

200330956

5.24. Resolution is not updated after a VM acquires a license and is restarted

Description

In a Red Enterprise Linux 7.3 guest VM, an increase in resolution from 1024×768 to 2560×1600 is not applied after a license is acquired and the `gridd` service is restarted. This issue occurs if the `multimonitor` parameter is added to the `xorg.conf` file.

Version

Red Enterprise Linux 7.3

Status

Open

Ref.

200275925

5.25. A segmentation fault in DBus code causes `nvidia-gridd` to exit on Red Hat Enterprise Linux and CentOS

Description

On Red Hat Enterprise Linux 6.8 and 6.9, and CentOS 6.8 and 6.9, a segmentation fault in DBus code causes the `nvidia-gridd` service to exit.

The `nvidia-gridd` service uses DBus for communication with **NVIDIA X Server Settings** to display licensing information through the **Manage License** page. Disabling the GUI for licensing resolves this issue.

To prevent this issue, the GUI for licensing is disabled by default. You might encounter this issue if you have enabled the GUI for licensing and are using Red Hat Enterprise Linux 6.8 or 6.9, or CentOS 6.8 and 6.9.

Version

Red Hat Enterprise Linux 6.8 and 6.9

CentOS 6.8 and 6.9

Status

Open

Ref.

- ▶ 200358191
- ▶ 200319854
- ▶ 1895945

5.26. No Manage License option available in NVIDIA X Server Settings by default

Description

By default, the **Manage License** option is not available in **NVIDIA X Server Settings**. This option is missing because the GUI for licensing on Linux is disabled by default to work around the

issue that is described in [A segmentation fault in Dbus code causes nvidia-gridd to exit on Red Hat Enterprise Linux and CentOS](#).

Workaround

This workaround requires sudo privileges.



Note: Do not use this workaround with Red Hat Enterprise Linux 6.8 and 6.9 or CentOS 6.8 and 6.9. To prevent a segmentation fault in Dbus code from causing the `nvidia-gridd` service from exiting, the GUI for licensing must be disabled with these OS versions.

If you are licensing a physical GPU for vCS, you **must** use the configuration file `/etc/nvidia/gridd.conf`.

1. If **NVIDIA X Server Settings** is running, shut it down.
2. If the `/etc/nvidia/gridd.conf` file does not already exist, create it by copying the supplied template file `/etc/nvidia/gridd.conf.template`.
3. As root, edit the `/etc/nvidia/gridd.conf` file to set the `EnableUI` option to `TRUE`.
4. Start the `nvidia-gridd` service.

```
# sudo service nvidia-gridd start
```

When **NVIDIA X Server Settings** is restarted, the **Manage License** option is now available.

Status

Open

5.27. Licenses remain checked out when VMs are forcibly powered off

Description

NVIDIA vGPU software licenses remain checked out on the license server when non-persistent VMs are forcibly powered off.

The NVIDIA service running in a VM returns checked out licenses when the VM is shut down. In environments where non-persistent licensed VMs are not cleanly shut down, licenses on the license server can become exhausted. For example, this issue can occur in automated test environments where VMs are frequently changing and are not guaranteed to be cleanly shut down. The licenses from such VMs remain checked out against their MAC address for seven days before they time out and become available to other VMs.

Resolution

If VMs are routinely being powered off without clean shutdown in your environment, you can avoid this issue by shortening the license borrow period. To shorten the license borrow period, set the `LicenseInterval` configuration setting in your VM image. For details, refer to [Virtual GPU Client Licensing User Guide](#).

Status

Closed

Ref.

1694975

5.28. VM bug checks after the guest VM driver for Windows 10 RS2 is installed

Description

When the VM is rebooted after the guest VM driver for Windows 10 RS2 is installed, the VM bug checks. When Windows boots, it selects one of the standard supported video modes. If Windows is booted directly with a display that is driven by an NVIDIA driver, for example a vGPU on Citrix Hypervisor, a blue screen crash occurs.

This issue occurs when the screen resolution is switched from VGA mode to a resolution that is higher than 1920×1200.

Fix

Download and install [Microsoft Windows Update KB4020102](#) from the Microsoft Update Catalog.

Workaround

If you have applied the fix, ignore this workaround.

Otherwise, you can work around this issue until you are able to apply the fix by not using resolutions higher than 1920×1200.

1. Choose a GPU profile in Citrix XenCenter that does not allow resolutions higher than 1920×1200.
2. Before rebooting the VM, set the display resolution to 1920×1200 or lower.

Status

Not an NVIDIA bug

Ref.

200310861

5.29. GNOME Display Manager (GDM) fails to start on Red Hat Enterprise Linux 7.2 and CentOS 7.0

Description

GDM fails to start on Red Hat Enterprise Linux 7.2 and CentOS 7.0 with the following error:

```
Oh no! Something has gone wrong!
```

Workaround

Permanently enable permissive mode for Security Enhanced Linux (SELinux).

1. As root, edit the `/etc/selinux/config` file to set `SELINUX` to `permissive`.

```
SELINUX=permissive
```

2. Reboot the system.

```
~]# reboot
```

For more information, see [Permissive Mode](#) in *Red Hat Enterprise Linux 7 SELinux User's and Administrator's Guide*.

Status

Not an NVIDIA bug

Ref.

200167868

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 NON-INFRINGEMENT, 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.

VESA DisplayPort

DisplayPort and DisplayPort Compliance Logo, DisplayPort Compliance Logo for Dual-mode Sources, and DisplayPort Compliance Logo for Active Cables are trademarks owned by the Video Electronics Standards Association in the United States and other countries.

HDMI

HDMI, the HDMI logo, and High-Definition Multimedia Interface are trademarks or registered trademarks of HDMI Licensing LLC.

OpenCL

OpenCL is a trademark of Apple Inc. used under license to the Khronos Group Inc.

Trademarks

NVIDIA, the NVIDIA logo, NVIDIA GRID, NVIDIA GRID vGPU, NVIDIA Maxwell, NVIDIA Pascal, NVIDIA Turing, NVIDIA Volta, Quadro, and Tesla are trademarks 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

© 2013-2020 NVIDIA Corporation. All rights reserved.