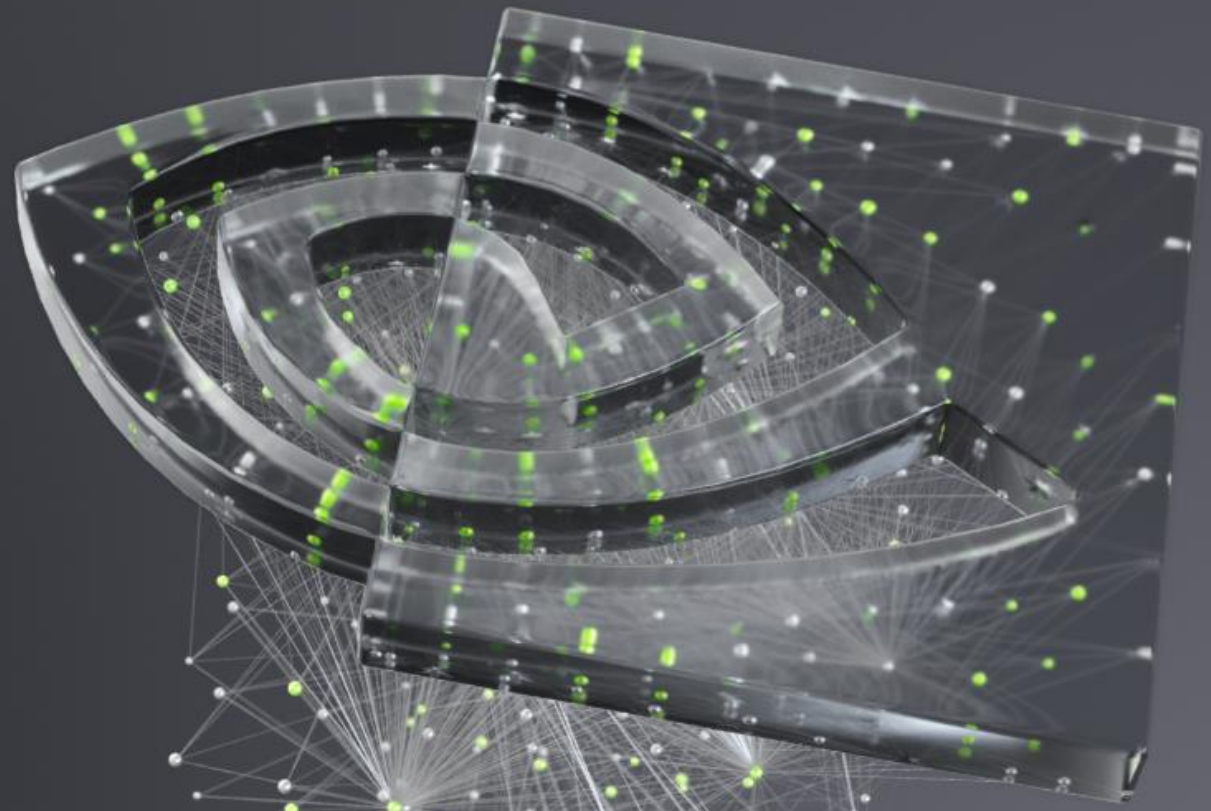


NVIDIA VGPU在LINUX KVM中的新优化和提升

Neo Jia, Dec 19th 2019

AGENDA

NVIDIA vGPU architecture on KVM

Internals of NVIDIA vGPU on KVM

NVIDIA vGPU new features on KVM

What's new

Tuning vGPU on KVM

Best practices for deploying vGPU, and how

What's next

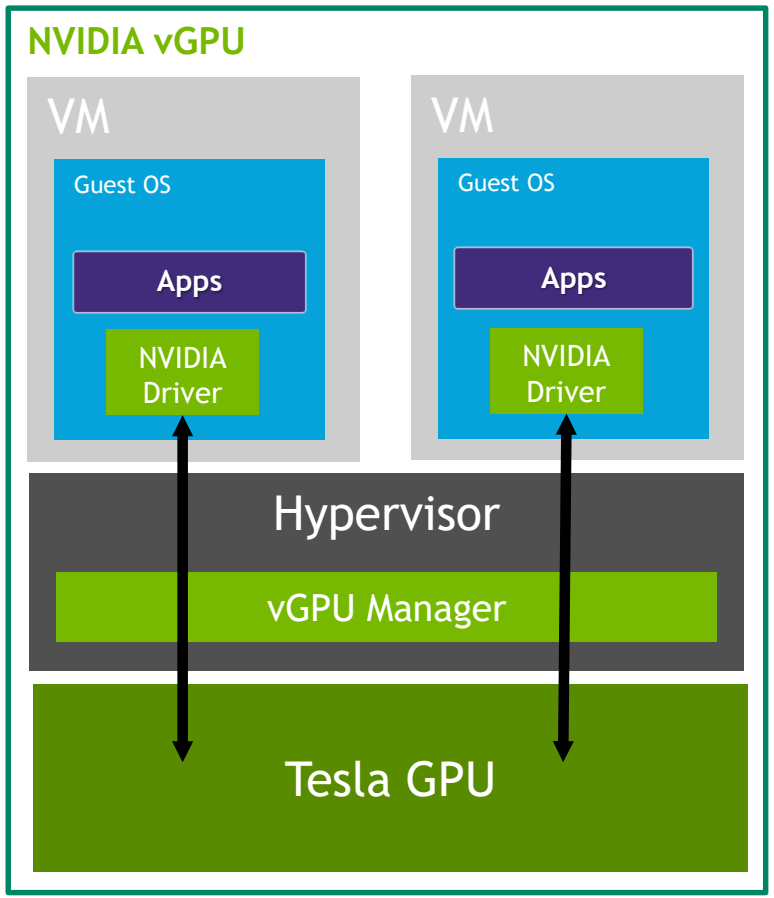
Upcoming features


NVIDIA VGPU ON KVM

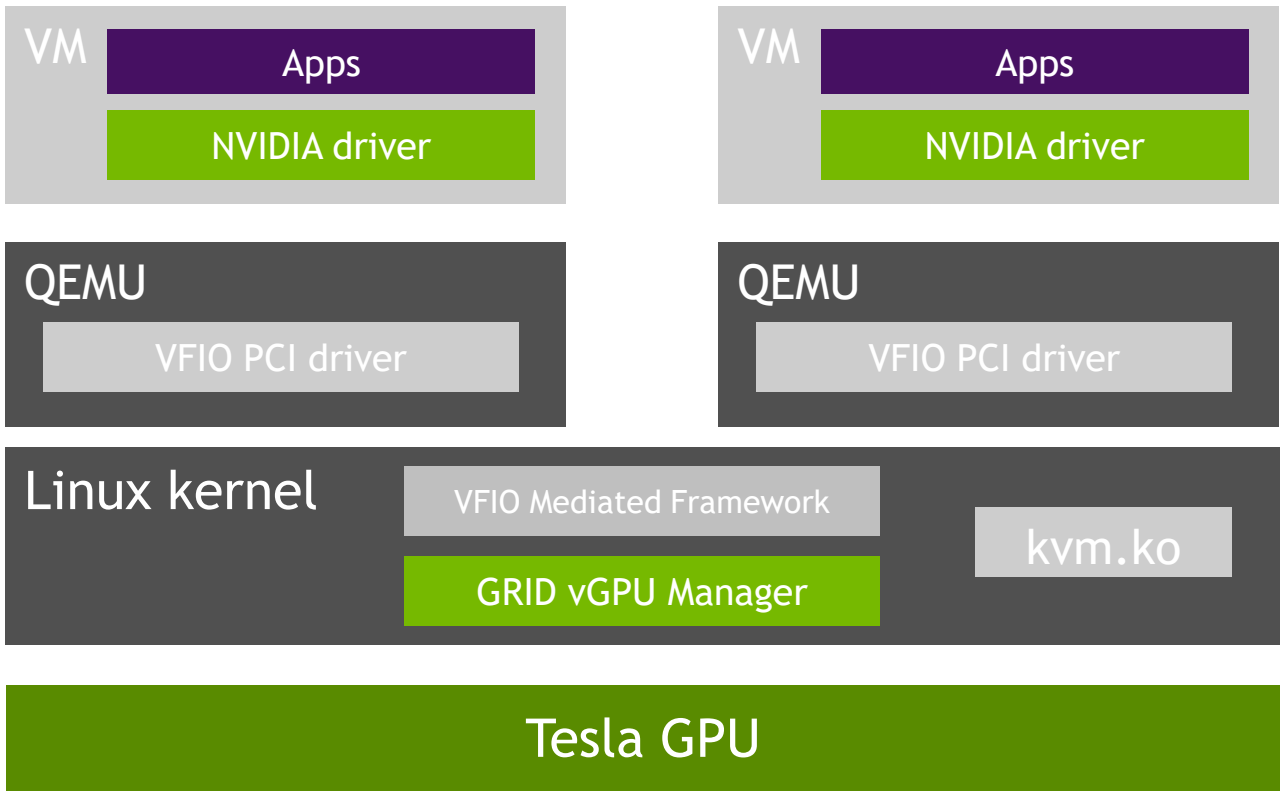
NVIDIA vGPU

Performance, Density, Manageability – for GPU

- Fully enables NVIDIA GPU on virtualized platforms
  - Wide availability - supported by all major hypervisors
  - Great app compatibility – NVIDIA driver inside VM
  - Great performance – VM direct access to GPU hardware
- Improved density
  - Multiple VMs can share one GPU
- Highly manageable
  - NVIDIA host driver, management tools retain full control of the GPU
  - vGPU suspend, resume, live migration enables workloads to be transparently moved between GPUs



NVIDIA vGPU KVM Architecture 101



Based on upstream VFIO-mediated architecture

No VFIO UAPI change

Mediated device managed by generic sysfs interface or libvirt

Mediated Device Framework – VFIO MDEV

A common framework for mediated I/O devices

Present in KVM Forum 2016, upstream since Linux 4.10, kernel maintainer - Kirti Wankhede @ NVIDIA

Mediated core module (new)

- Mediated bus driver, create mediated device
- Physical device interface for vendor driver callbacks
- Generic mediated device management user interface (sysfs)

Mediated device module (new)

- Manage created mediated device, fully compatible with VFIO user API

VFIO IOMMU driver (enhancement)

- VFIO IOMMU API TYPE1 compatible, easy to extend to non-TYPE1

Mediated Device Framework

Mediated Device sysfs

After NVIDIA driver device registration, under physical device sysfs (sys/bus/pci/drivers/nvidia/0000:83:00.0):

```
[root@cjia-vgx-kvm bin]# ls /sys/bus/pci/drivers/nvidia/0000:83:00.0/mdev_supported_types
```

```
nvidia-157  nvidia-243  nvidia-289  nvidia-64  nvidia-66  nvidia-68  nvidia-70
nvidia-214  nvidia-288  nvidia-63   nvidia-65  nvidia-67  nvidia-69  nvidia-71
```

```
[root@cjia-vgx-kvm bin]# cat /sys/bus/pci/drivers/nvidia/0000:83:00.0/mdev_supported_types/nvidia-289/name
```

```
GRID P4-8C
```

```
[root@cjia-vgx-kvm bin]# cat /sys/bus/pci/drivers/nvidia/0000:83:00.0/mdev_supported_types/nvidia-289/description
```

```
num_heads=1, frl_config=60, framebuffer=8192M, max_resolution=4096x2160, max_instance=1
```

Mdev node: /sys/bus/mdev/devices/$mdev_UUID/

https://www.kernel.org/doc/Documentation/ABI/testing/sysfs-bus-vfio-mdev

CREATE AND START VGPU VM

Generate vGPU mdev UUID via *uuid-gen*, for example "98d19132-f8f0-4d19-8743-9efa23e6c493"

Create vGPU device

```
echo $UUID > /sys/bus/pci/drivers/nvidia/0000:05:00.0/mdev_supported_types/nvidia-289/create
```

Start vGPU directly via QEMU command line

```
-sysfsdev /sys/bus/mdev/devices/$UUID
```

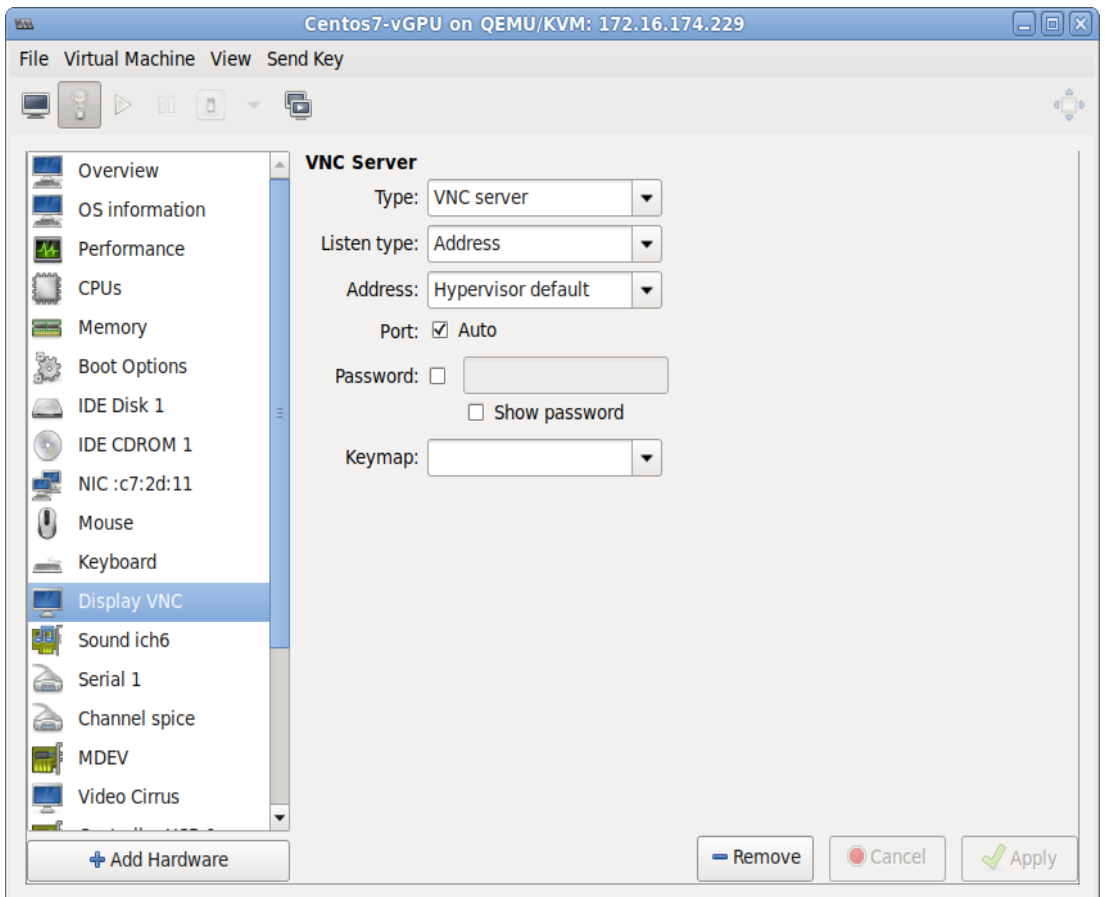
Start vGPU via libvirt

```
<hostdev mode='subsystem' type='mdev' managed='no' model='vfio-pci'>
    <source>
        <address uuid='$UUID'/>
    </source>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x08’ function='0x0'/>
</hostdev>
```

NEW VGPU FEATURES ON KVM

CONSOLE VNC

GRID 7.1



Console VNC - the management interface normally exposed by device model of VMM, VIFO ioctl

VFIO_DEVICE_QUERY_GFX_PLANE

VFIO_DEVICE_GET_GFX_DMABUF

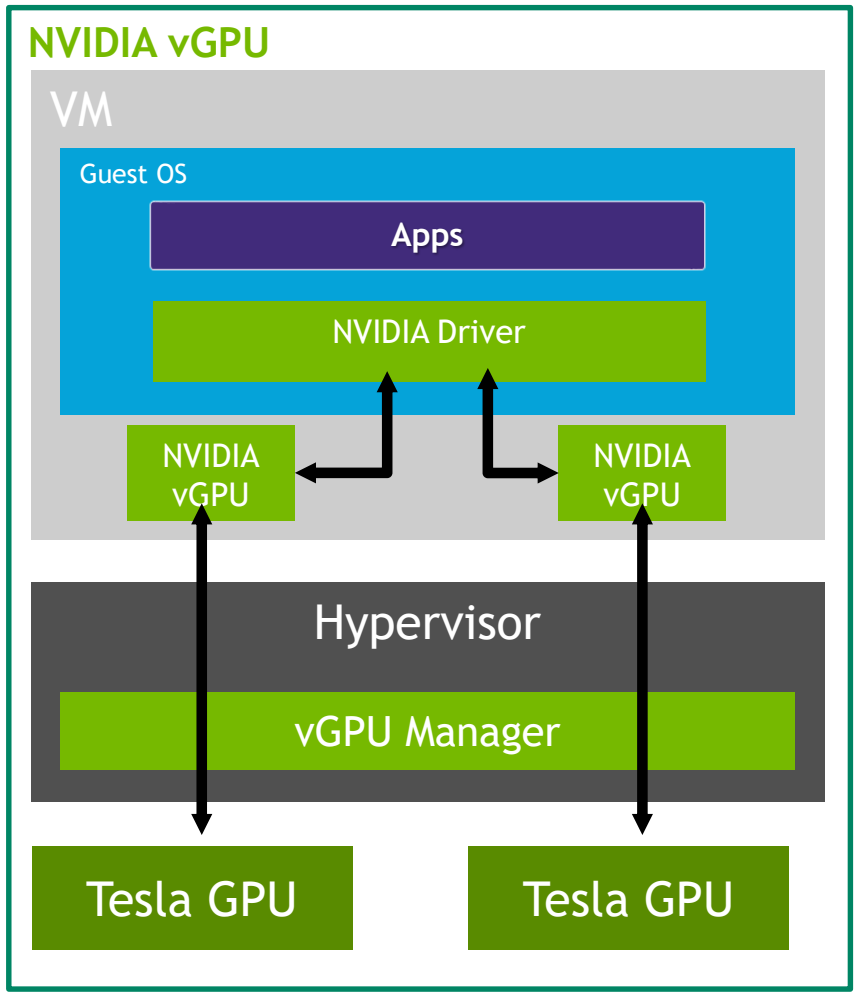
Low FPS interface for management / debugging only

Only expose head 0 for every virtual GPU inside VM

Officially supported by RHEL 8.0 – all changes are upstreamed

MULTI-VGPU

GRID 8.0



Multiple virtual GPUs exposed to the guest OS

Allow applications take advantages of multiple physical GPU

One vGPU manager instance manages multiple virtual device per VM

Only 1:1 vGPU profiles are supported

No additional Linux kernel mdev changes required, supported since GRID 8.0

ERROR-CORRECTING CODE (ECC) MEMORY SUPPORT

GRID 9.0

vGPU startup fails if ECC is enabled on older drivers

> *“nvidia-vgpu-mgr[27029]: error: vmiop_log: (0x0): Initialization: vGPU not supported with ECC Enabled.”*

ECC is a critical feature for service provider especially computing scenarios

> In terms of memory space overhead after turning ECC, no on HBM2, 6.25% overhead on DDR memory

Maintain the behavior as baremetal

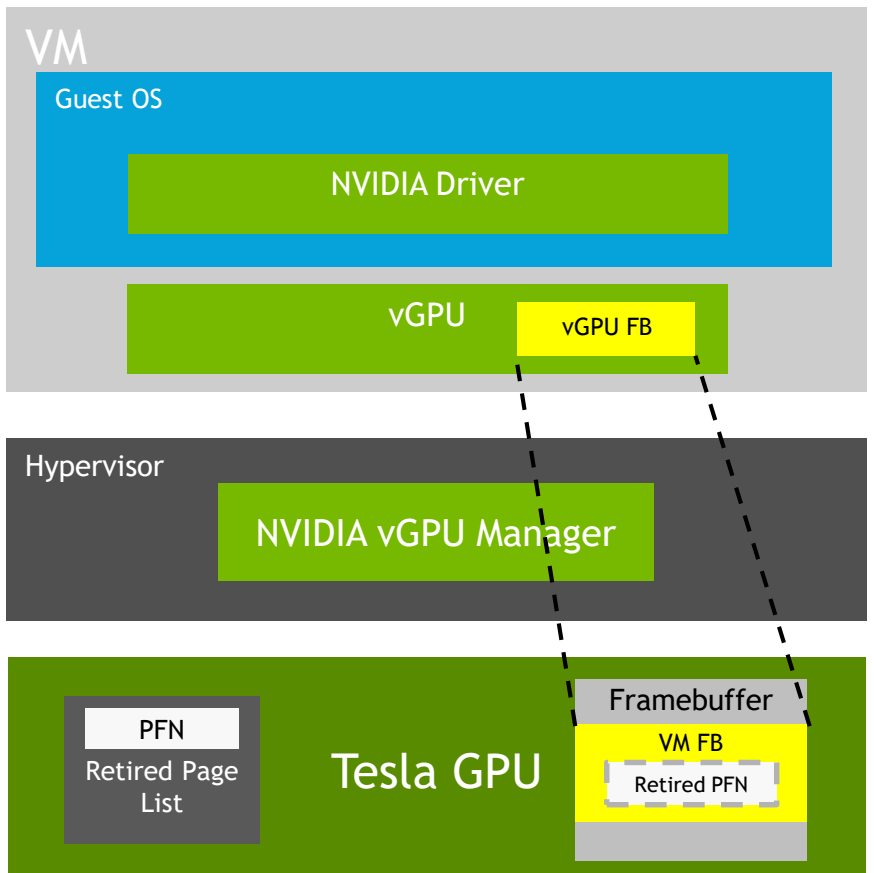
> ECC error on a given VM will kill all its compute tasks, no new compute tasks can be launched until VM reboot.

vGPU guest can opt-out of ECC just like baremetal if ECC is enabled on physical GPU

> Allow the service provide to enable ECC independent of customer (guest) choice

PAGE RETIREMENT SUPPORT

GRID 9.0



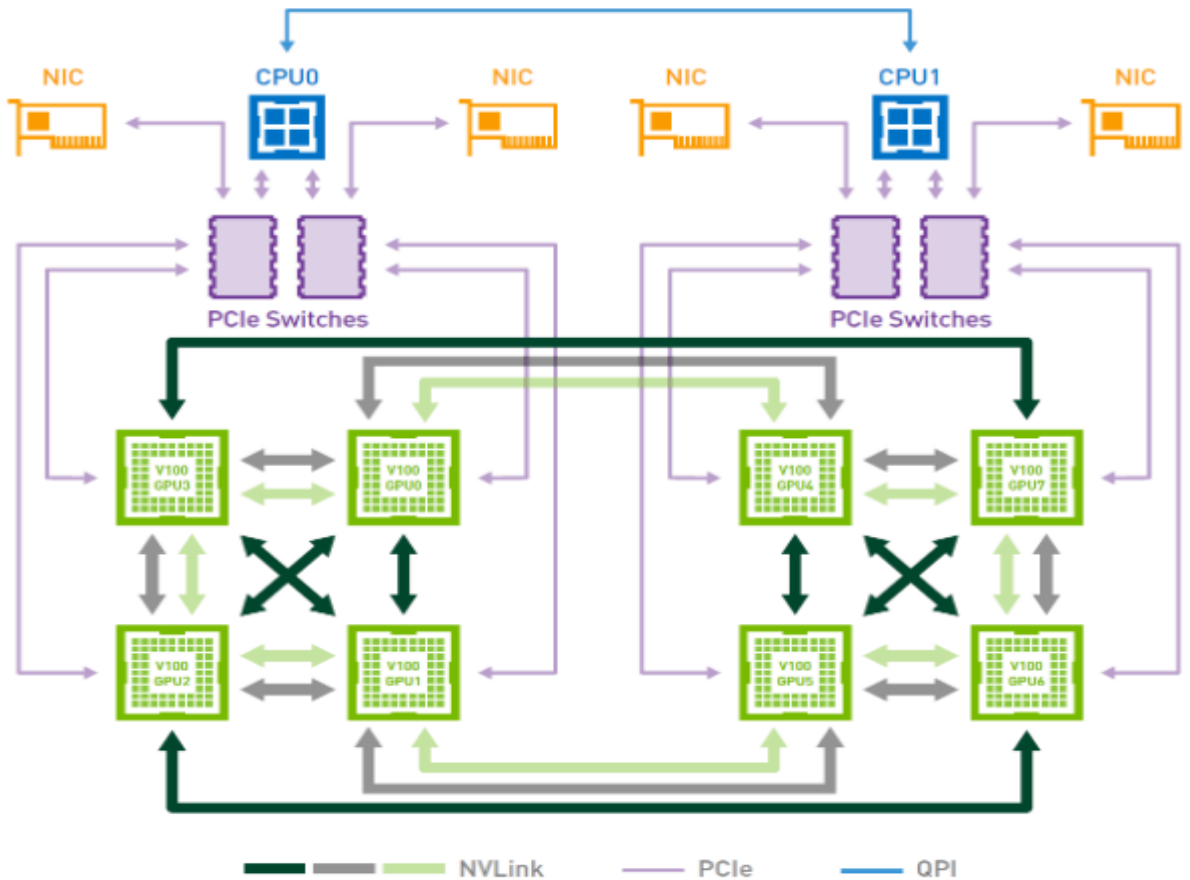
When ECC is enabled, NVIDIA driver retires FB pages that register Double Bit Error or multiple Single Bit Errors

When ECC is disabled, only existing failed pages are retired, no additional pages are retired

Guest driver always see a contiguous framebuffer

VGPU P2P OVER NVLINK

GRID 9.0



NVLINK is high-bandwidth interconnect enabling ultra-fast communication between GPUs or between GPU and CPU

Requires multiple 1:1 vGPU, created on physical GPUs with direct NVLink connections

Linux VM only

VGPU P2P OVER NVLINK - TOPO

GRID 9.0

nvidia-smi topo -m // on DGX V100

```
[root@dhcp-10-24-129-49 ~]# nvidia-smi topo -m
        GPU0    GPU1    GPU2    GPU3    GPU4    GPU5    GPU6    GPU7    mlx5_0  mlx5_1  mlx5_2  mlx5_3  CPU Affinity
GPU0     X      NV1     NV1     NV2     NV2     SYS     SYS     SYS     PIX     PHB     SYS     SYS     0-19,40-59
GPU1    NV1      X      NV2     NV1     SYS     NV2     SYS     SYS     PIX     PHB     SYS     SYS     0-19,40-59
GPU2    NV1     NV2      X      NV2     SYS     SYS     NV1     SYS     PHB     PIX     SYS     SYS     0-19,40-59
GPU3    NV2     NV1     NV2      X      SYS     SYS     SYS     NV1     PHB     PIX     SYS     SYS     0-19,40-59
GPU4    NV2     SYS     SYS     SYS      X      NV1     NV1     NV2     SYS     SYS     PIX     PHB     20-39,60-79
GPU5    SYS     NV2     SYS     SYS     NV1      X      NV2     NV1     SYS     SYS     PIX     PHB     20-39,60-79
GPU6    SYS     SYS     NV1     SYS     NV1     NV2      X      NV2     SYS     SYS     PHB     PIX     20-39,60-79
GPU7    SYS     SYS     SYS     NV1     NV2     NV1     NV2      X      SYS     SYS     PHB     PIX     20-39,60-79
mlx5_0  PIX     PIX     PHB     PHB     SYS     SYS     SYS     SYS      X      PHB     SYS     SYS
mlx5_1  PHB     PHB     PIX     PIX     SYS     SYS     SYS     SYS     PHB      X      SYS     SYS
mlx5_2  SYS     SYS     SYS     SYS     PIX     PIX     PHB     PHB     SYS     SYS      X      PHB
mlx5_3  SYS     SYS     SYS     SYS     PHB     PHB     PIX     PIX     SYS     SYS     PHB      X
```

nvidia-smi topo -m // Creating a VM with first 4 physical GPU above

```
        GPU0    GPU1    GPU2    GPU3    CPU Affinity
GPU0     X      NV1     NV1     NV2     0-3
GPU1    NV1      X      NV2     NV1     0-3
GPU2    NV1     NV2      X      NV2     0-3
GPU3    NV2     NV1     NV2      X      0-3
```

VGPU P2P OVER NVLINK - BW

DGX-1V sample vGPU vs. Passthru

./p2p_bandwidth -t Memcpy_DtoD_Read_CE_Bandwidth // vGPU

```
Device 0: GRID V100X-16C
Device 1: GRID V100X-16C
Device 2: GRID V100X-16C
Device 3: GRID V100X-16C
Peer to peer support matrix:
       0   1   2   3
    0 no  yes yes yes
    1 yes no  yes yes
    2 yes yes no  yes
    3 yes yes yes no
testutils::random seed value: 2942506236
Dispatcher pid: 15242
Running test Memcpy_DtoD_Read_CE_Bandwidth (pid: 15245)
testutils::random seed value: 3663319292
memcpy CE GPU(row) -> GPU(column) bandwidth (GB/s)
          0         1         2         3
0      0.00     24.17     24.18     48.17
1     24.17      0.00     48.17     24.18
2     24.17     48.14      0.00     48.17
3     48.13     24.17     48.18      0.00
&&&& PERF Memcpy_DtoD_Read_CE_Bandwidth_sum 433.9982
+GB/s
^^^^ PASS: Memcpy_DtoD_Read_CE_Bandwidth
1 out of 1 ENABLED tests passed (100%)
&&&& p2p_bandwidth test PASSED
```

./p2p_bandwidth -t Memcpy_DtoD_Read_CE_Bandwidth // Passthru

```
Device 0: Tesla V100-SXM2-16GB
Device 1: Tesla V100-SXM2-16GB
Device 2: Tesla V100-SXM2-16GB
Device 3: Tesla V100-SXM2-16GB
Peer to peer support matrix:
       0   1   2   3
    0 no  yes yes yes
    1 yes no  yes yes
    2 yes yes no  yes
    3 yes yes yes no
testutils::random seed value: 3123139763
Dispatcher pid: 10931
Running test Memcpy_DtoD_Read_CE_Bandwidth (pid: 10944)
testutils::random seed value: 372228203
memcpy CE GPU(row) -> GPU(column) bandwidth (GB/s)
          0         1         2         3
0      0.00     24.17     24.14     48.17
1     24.16      0.00     48.16     24.14
2     24.17     48.11      0.00     48.13
3     48.11     24.19     48.14      0.00
&&&& PERF Memcpy_DtoD_Read_CE_Bandwidth_sum 433.7746 +GB/s
^^^^ PASS: Memcpy_DtoD_Read_CE_Bandwidth
1 out of 1 ENABLED tests passed (100%)
&&&& p2p_bandwidth test PASSED
```

TUNING NVIDIA VGPU ON KVM

VM CONFIGURATION

Hugepage setting of VM XML file

```
<hugepages>
   <page size='1048576' unit='KiB'/>
</hugepages>
```

HyperV enlightenment setting

```
<features>
    <hyperv>
        <relaxed state='on' />
        <vapic status='on' />
        <spinlocks status='on' retires='4096' />
        <vpindex state='on' />
        <runtime status='on' />
    </hyperv>
</features>
<clock offset='localtime'>
    <timer name='hypervclock' present='yes'/>
</clock>
```

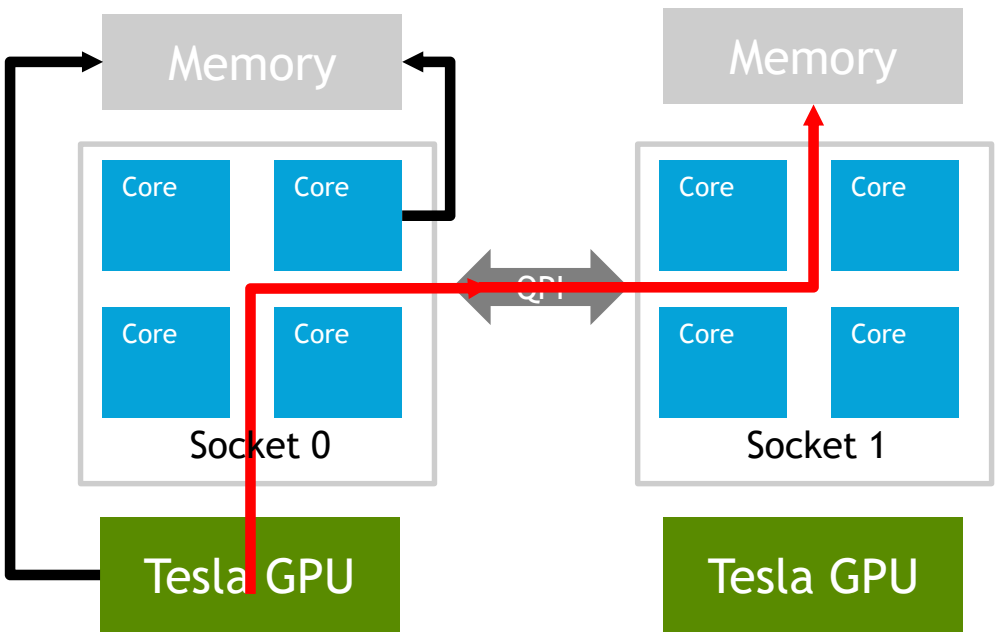
Enable hugepage

For Windows, checkout the following HyperV enlightenments

- relaxed – disable watchdog timeout
- vapic – virtual APIC MSR
- spinlock - yield vCPU to other guests
- vpindex – synthetic MSR returns virtual process index
- runtime – MSR provides time spent in guest/hypervisor
- hypervclock - paravirt timer source

PLATFORM

CPU NUMA and I/O NUMA



Non-Uniform Memory Access (NUMA)

Memory and GPUs connected to each socket

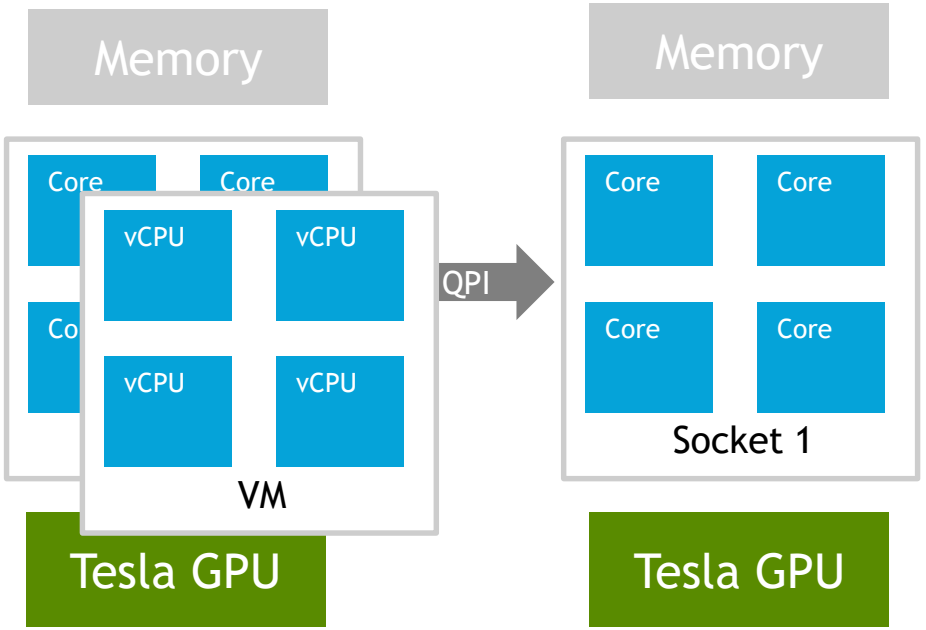
CPUs connected via QPI

CPU/GPU access to memory on same socket is faster

Access to memory on remote socket is slower

PLATFORM

vCPU Pinning



"restrict static" pinning recommended for no oversubscription use case

vCPU Pinning

```
<vcpupin vcpu="0" cpuset="0" />
<vcpupin vcpu="1" cpuset="1" />
<vcpupin vcpu="2" cpuset="2" />
<vcpupin vcpu="3" cpuset="3" />
```

PERFORMANCE DEBUGGING STEP BY STEP

Always start with passthru VM

Then move to vGPU 1:1, with same configuration – CPU (pinning), memory, NUMA

Disable vGPU's frame rate limiter for graphics applications

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

Performance delta between vGPU 1:1 vs. passthru generally < 5%

Multiple vGPU VM sharing single physical GPU

GPU bound application – aggregated perf generally <5% of passthru

CPU bound application – aggregated perf generally >100% passthru

WHAT'S NEXT

VGPU MIGRATION

Upstream in-progress

Bring migration feature to VFIO mdev infrastructure

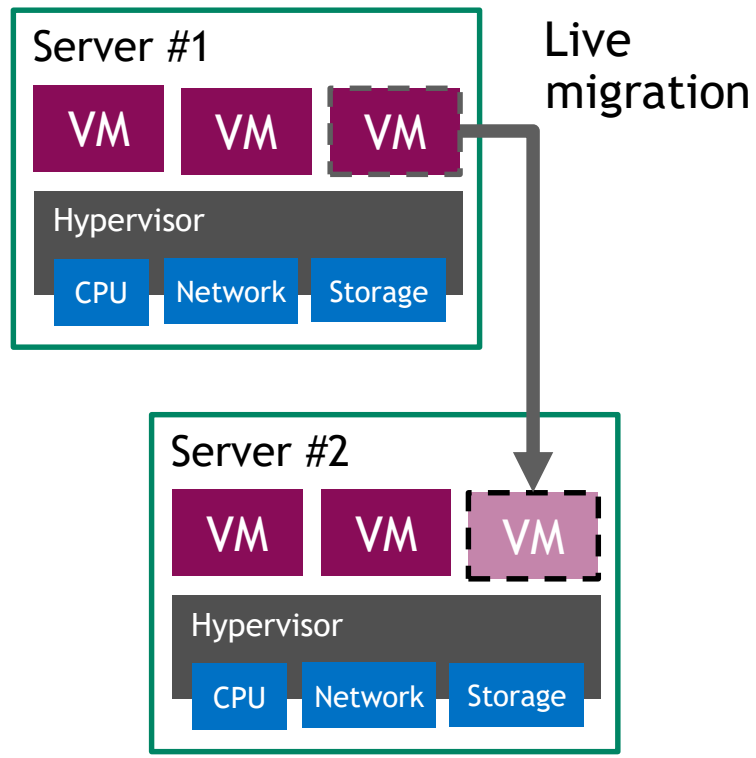
http://patchwork.kernel.org/patch/11239917

KABI upstream estimated around end of 2019

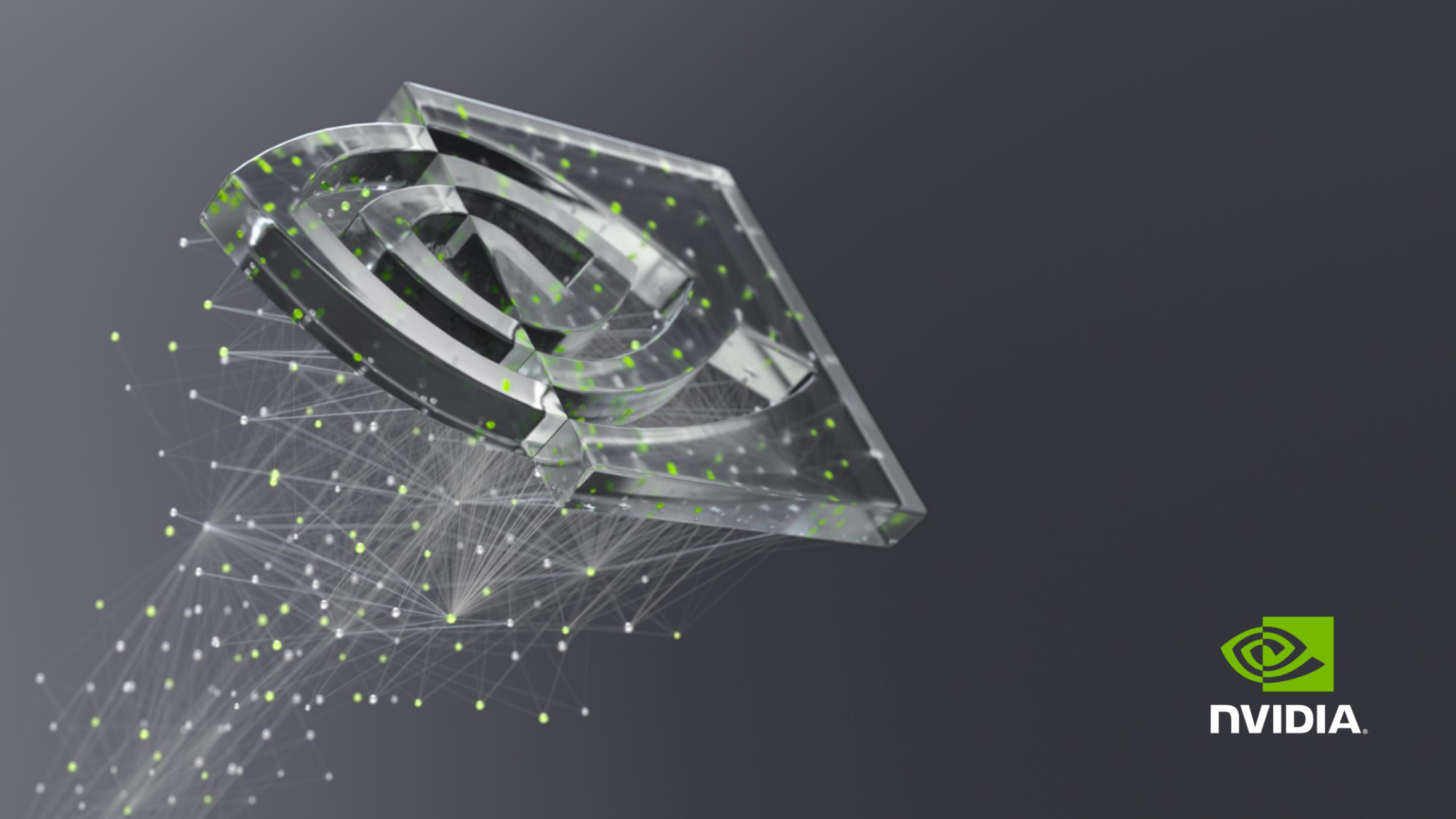
Fully functional with current patchset

Migration compatibility enforced via “migration_version” attribute

Documentation/vfio-mediated-device.txt




QUESTIONS?


NVIDIA

BACKUP SLIDES

ERROR-CORRECTING CODE (ECC) MEMORY SUPPORT

GRID 9.0

nvidia-smi output

```
[cjia@cjia-vgx-kvm ~]$ nvidia-smi -i 3
Mon Dec  2 02:52:07 2019
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 440.33.01    Driver Version: 440.33.01    CUDA Version: 10.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   3  Tesla P4            Off  | 00000000:83:00.0 Off |                    0 |
| N/A   29C    P0    24W /  75W |      0MiB /  7611MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

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

RESOURCES

NVIDIA Virtual GPU (vGPU) Software Documentation

https://docs.nvidia.com/grid/

Public talks

DELIVERING HIGH-PERFORMANCE REMOTE GRAPHICS WITH NVIDIA GRID VIRTUAL GPU – Andy Currid, GTC Silicon Valley 2014

vGPU on KVM - A VFIO Based Framework - Neo Jia & Kirti Wankhede, Toronto, KVM forum 2016

持续助力数据中心虚拟化：KVM 里的虚拟 GPU – Neo Jia, Michael Shen, GTC China 2018