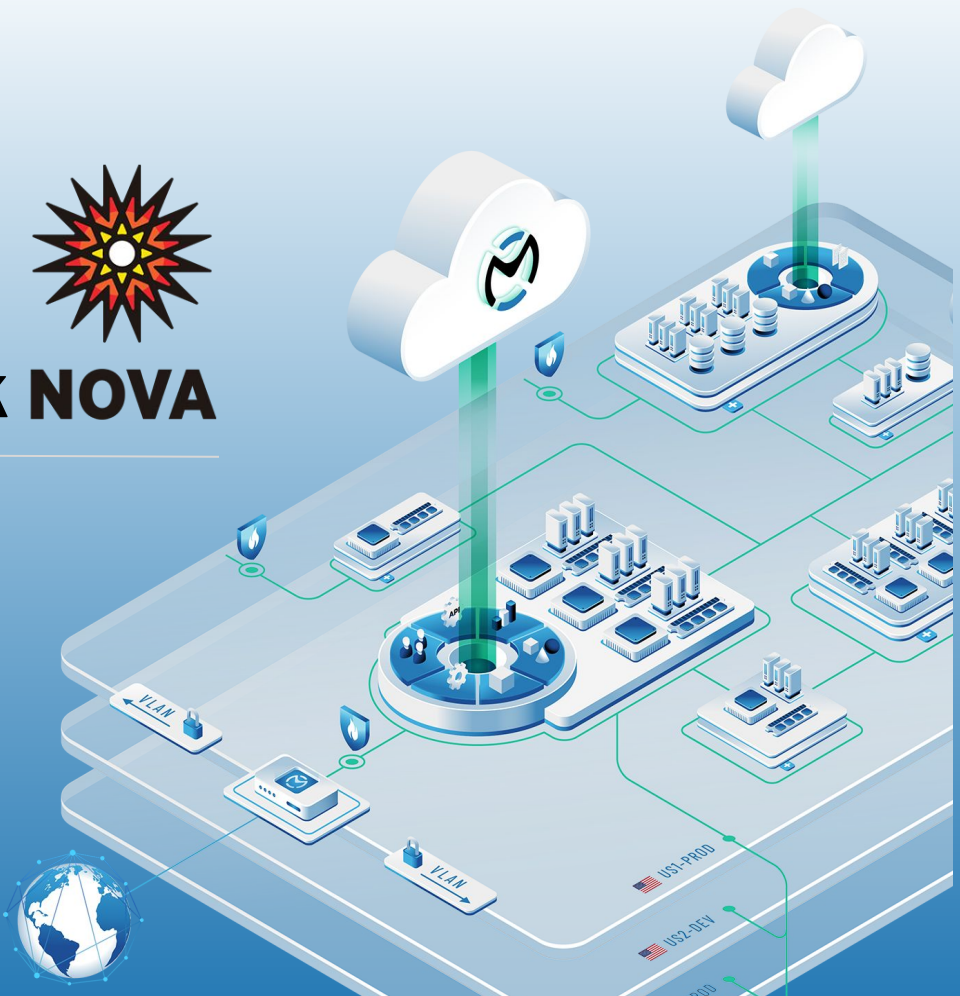




# vGPUs with OpenStack NOVA

 openmetal





# Jacob Hipps


Principal Engineer  
OpenMetal

[jacobh@openmetal.io](mailto:jacobh@openmetal.io)  
[@yellowcrescent](https://twitter.com/yellowcrescent)



- **OMI OpenStack Private Cloud configuration and deployment system**
- **Monitoring and instrumentation for hardware & deployment systems**
- **Research & development of new technologies and platform improvements**



 Open Infrastructure  
FOUNDATION

SILVER MEMBER



INFRASTRUCTURE DONOR



OpenMetal is an Infrastructure as a Service Company (IaaS) that believes in the collective and fundamental good of open source in the information technology world.

**Our Mission: Make highly complex open source systems available on-demand to increase accessibility for smaller teams.**



# What & Why

- vGPUs are virtualized GPUs, shared from one or more physical GPUs
- Use cases (and when not to use)
- Hardware setup
- Configuring the host node
- Spinning up a VM with a vGPU

# Use Case #1

CI jobs or transient tasks

- **Spin up a VM with all of your required tools to run a job; destroy when completed**
- **Only uses the vGPU resource while the VM is active**
- **Scheduling vGPUs via Kubernetes pods/resources**

## Use Case #2

Rendering or Graphical VMs

- **Isolated rendering instance with dedicated GPU resources**
- **Capabilities depend on physical GPU and licensing**
  - ◆ A100 only supports “compute” workloads (eg. CUDA, OptiX)
  - ◆ RTX-6000+ has RT cores for faster OptiX raytracing and allows OpenGL, Vulkan & DirectX display
- **Virtual desktop/workstation**

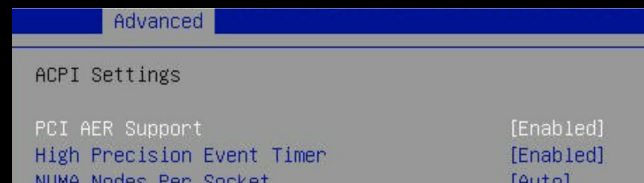
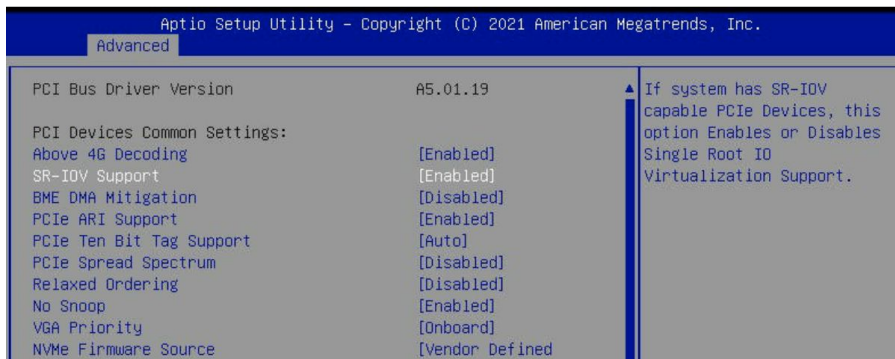
# When NOT to use vGPUs?

Use PCI passthrough instead

- **Workloads requiring high amount of VRAM (such as AI or ML training)**
- **Long-lived VMs or applications that can handle task delegation on their own**
- **Video transcoding**

# Hardware Setup

- GPU with vGPU support
  - ◆ NVIDIA A-series, L-series, GRID, RTX-6000, etc.
- Correct BIOS options are **critical!**
- Need to ensure options for SR-IOV and IOMMU are enabled, along with all of their dependencies



## BIOS Settings (from Supermicro BH12 AMD series)

- Virtualization Support (VMX or SVM)
- VT-d (Directed I/O) or IOMMU
  - ◆ Required for vfio-pci and vGPU isolation, also used for PCI passthrough
- SR-IOV
  - ◆ Required to create virtual functions for vGPUs
- Above 4GB Decoding (64-bit addressing)
  - ◆ Allows PCIe devices to map BAR memory above 4GB
- PCIe ARI (Advanced Routing ID Interpretation)
  - ◆ Required for SR-IOV
- PCIe ACS (Access Control Services)
  - ◆ Required to ensure IOMMU groups are created for SR-IOV VFs
- PCIe AER (Advanced Error Recovery)
  - ◆ Required for ACS to function



# Configuring the Host GPU Setup

- Ensure IOMMU is enabled in the kernel
  - ◆ intel\_iommu=on or amd\_iommu=on
- Install prereqs
  - ◆ kernel headers; optional: dkms, mdevctl
- Install Host GPU driver
  - ◆ Enterprise NVIDIA "Linux KVM" GRID Host driver
  - ◆ Blacklist nouveau if necessary
  - ◆ May need to use `-no-drm` for golden image builds
  - ◆ Ensure `nvidia-vgpubd` & `nvidia-vgpu-mgr` services are enabled in `systemd`
- Reboot!
- Enable VFs & Register MDEVs
  - ◆ `/usr/lib/nvidia/sriov-manage -e ALL`
- Check your work
  - ◆ `lspci -nn | grep 10de #` should show 16+1 entries
  - ◆ `mdevctl types`
    - `ls /sys/class/mdev_bus/*/mdev_supported_types`
  - ◆ `nvidia-smi`

```
[root@magnificent-zebu ~]# /usr/lib/nvidia/sriov-manage -e ALL
Enabling VFs on 0000:01:00.0
```

```
[ 47.350362] pci 0000:01:02.1: Enabling HDA controller
[ 47.356266] pci 0000:01:02.1: Adding to iommu group 87
[ 47.362162] NVRM: Aborting probe for VF 0000:01:02.1 since PF is not bound to nvidia driver.
[ 47.371422] nvidia: probe of 0000:01:02.1 failed with error -1
[ 47.378043] pci-pf-stub 0000:01:02.1: claimed by pci-pf-stub
[ 47.384524] pci 0000:01:02.2: [10de:20f1] type 00 class 0x030200
[ 47.391340] pci 0000:01:02.2: enabling Extended Tags
[ 47.397092] pci 0000:01:02.2: Enabling HDA controller
[ 47.402944] pci 0000:01:02.2: Adding to iommu group 88
[ 47.408899] NVRM: Aborting probe for VF 0000:01:02.2 since PF is not bound to nvidia driver.
[ 47.418157] nvidia: probe of 0000:01:02.2 failed with error -1
[ 47.424793] pci-pf-stub 0000:01:02.2: claimed by pci-pf-stub
[ 47.431239] pci 0000:01:02.3: [10de:20f1] type 00 class 0x030200
[ 47.438045] pci 0000:01:02.3: enabling Extended Tags
[ 47.443824] pci 0000:01:02.3: Enabling HDA controller
[ 47.449737] pci 0000:01:02.3: Adding to iommu group 89
[ 47.455690] NVRM: Aborting probe for VF 0000:01:02.3 since PF is not bound to nvidia driver.
[ 47.464937] nvidia: probe of 0000:01:02.3 failed with error -1
[ 47.471568] pci-pf-stub 0000:01:02.3: claimed by pci-pf-stub
[ 47.478173] pci-pf-stub 0000:01:00.0: driver left SR-IOV enabled after remove
[ 47.849370] NVRM: GPU at 0000:01:00.0 has software scheduler DISABLED with policy BEST_EFFORT.
[ 48.154785] nvidia 0000:01:00.4: enabling device (0000 -> 0002)
[ 48.161612] nvidia 0000:01:00.4: MDEV: Registered
[ 48.174734] nvidia 0000:01:00.5: enabling device (0000 -> 0002)
[ 48.182241] nvidia 0000:01:00.5: MDEV: Registered
[ 48.200565] nvidia 0000:01:00.6: enabling device (0000 -> 0002)
[ 48.207134] nvidia 0000:01:00.6: MDEV: Registered
[ 48.212680] nvidia 0000:01:00.7: enabling device (0000 -> 0002)
[ 48.219432] nvidia 0000:01:00.7: MDEV: Registered
[ 48.225069] nvidia 0000:01:00.0: enabling device (0000 -> 0002)
```

```
[root@timely-lamprey ~]# ls /sys/class/mdev_bus/*/mdev_supported_types
'/sys/class/mdev_bus/0000:01:00.4/mdev_supported_types':
nvidia-468 nvidia-469 nvidia-470 nvidia-471 nvidia-472 nvidia-473 nvi
'/sys/class/mdev_bus/0000:01:00.5/mdev_supported_types':
nvidia-468 nvidia-469 nvidia-470 nvidia-471 nvidia-472 nvidia-473 nvi
'/sys/class/mdev_bus/0000:01:00.6/mdev_supported_types':
nvidia-468 nvidia-469 nvidia-470 nvidia-471 nvidia-472 nvidia-473 nvi
```

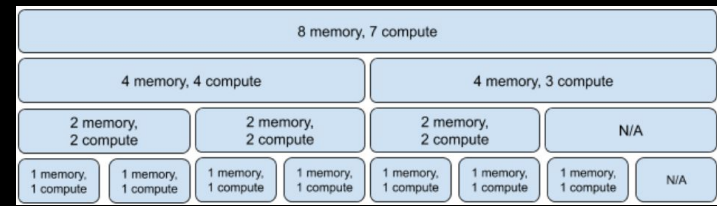
# Configuring the Host vGPU Profiles

- vGPU Mode: Time-shared versus MIG
  - ◆ MIG only supports Compute; max 7 instances
- Possible to mix and match non-MIG profiles, but may not be efficient
- Example A100 profile: A100-3-20C
  - ◆ [NVIDIA vGPU Profile Matrix](#)
- MIG Instance Type: MIG 3g.20gb
- Each GPU model has their own profiles

```
# Enable MIG mode
nvidia-smi -mig 1
```

```
# List available profiles
nvidia-smi mig -lgip
```

```
# Create vGPU instances with our profiles
nvidia-smi mig -cgi $PROFILE_ID,...
```



Config	GPC Slice #0	GPC Slice #1	GPC Slice #2	GPC Slice #3	GPC Slice #4	GPC Slice #5	GPC Slice #6	OFA	NVDEC	NVJPG	P2P	GPU Direct RDMA
1								1	5	1	No	
2		4		7		2	1	0	2+1+0	0	No	
3		4			1	1	1	0	2+0+0	0	No	
4		3				3		0	2+2	0	No	
5					2		1	0	2+1+0	0	No	
6		3		1	1	1		0	2+0+0	0	No	
7		2		2			3	0	1+1+2	0	No	
8		2		1	2		3	0	1+0+2	0	No	
9		1		2	1		3	0	0+0+1+2	0	No	
10		1		1	1		3	0	0+0+0+2	0	No	
11		2		2			1	0	1+1+1+0	0	No	
12		2		1	1		2	1	1+0+0+1+0	0	No	
13		1		2		2	1	0	0+0+1+1+0	0	No	
14		2		1	1		1	1	1+0+0+0+0	0	No	
15		1		2	1		1	0	0+0+1+0+0+0	0	No	
16		1		1	1		2	1	0+0+0+1+1+0	0	No	
17		1		1	1		2	1	0+0+0+0+0+1	0	No	
18		1		1	1		1	1	0+0+0+0+0+0+0	0	No	

Supported MemBW proportional to size of the instance

```
[root@magnificent-zebu ~]# nvidia-smi mig -lgip
GPU instance profiles:
GPU Name ID Instances Memory Free/Total P2P SM DEC ENC
Free/Total GiB CE JPEG OFA
-----
0 MIG 1g.5gb 19 7/7 4.75 No 14 0 0
1 0
2 0
-----
0 MIG 2g.10gb 14 3/3 9.75 No 28 1 0
2 0
-----
0 MIG 3g.20gb 9 2/2 19.62 No 42 2 0
3 0
-----
0 MIG 4g.20gb 5 1/1 19.62 No 56 2 0
4 0
-----
0 MIG 7g.40gb 0 1/1 39.50 No 98 5 0
7 1 1
-----
[root@magnificent-zebu ~]# nvidia-smi mig -cgi 9,9
Successfully created GPU instance ID 2 on GPU 0 using profile MIG 3g.20gb (ID 9)
Successfully created GPU instance ID 1 on GPU 0 using profile MIG 3g.20gb (ID 9)
[root@magnificent-zebu ~]#
```

# Configuring the Host

## NOVA

- Recommend Yoga or above
- After creating your vGPUs in the last step, check to see their MDEV name [verify]
  - ◆ `ls /sys/class/mdev_bus/*/mdev_supported_types`
- Update nova-compute's nova.conf

```
[devices]
enabled_vgpu_types = nvidia-476 # Victoria and below
enabled_mdev_types = nvidia-476 # Wallaby and above
```

- Restart nova-compute
- Check Placement to verify that our VGPU resources are available

```
openstack allocation candidate list --resource VGPU=1
```

- Bonus: Custom Traits
  - ◆ CUSTOM\_NVIDIA\_A100\_3\_20C - Create & set it on the "local\_pci" resource providers, up to the max instances allowed (2)
  - ◆ Restricts max instances to correct number instead of 16
  - ◆ Allows having flavors for different vGPU types

```
grep '1' /sys/class/mdev_bus/*/mdev_supported_types/*/available_instances
1:00.4/mdev_supported_types/nvidia-476/available_instances:1
1:00.5/mdev_supported_types/nvidia-476/available_instances:1
1:00.6/mdev_supported_types/nvidia-476/available_instances:1
1:00.7/mdev_supported_types/nvidia-476/available_instances:1
```

```
(.venv) [root@affectionate-hornet ~]# openstack allocation candidate list
+-----+-----+-----+-----+
| # | allocation | resource provider | explicitly | inventory used |
+-----+-----+-----+-----+
| 1 | VGPU=1     | 4b6a7367-d4d0-49fa-9b9d-0b010f5b855d |           | VGPU=0/1      |
| 2 | VGPU=1     | 31a418c2-4158-4176-a13e-72e18aa77f0e |           | VGPU=0/1      |
| 3 | VGPU=1     | 10935d4a-fb72-4229-b5b9-5df814d37d68 |           | VGPU=0/1      |
| 4 | VGPU=1     | 88b4c5c4-874b-4352-92fc-95098a5390f7 |           | VGPU=0/1      |
| 5 | VGPU=1     | 0c250855-0b53-45c3-9c6e-080d705008b5 |           | VGPU=0/1      |
```

```
# Creating & assigning optional traits
openstack trait create CUSTOM_NVIDIA_A100_3_20C
```

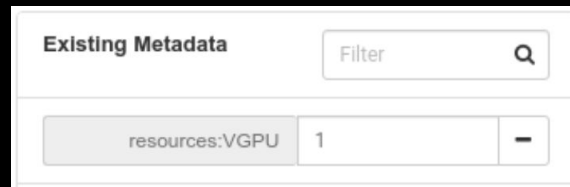
```
openstack resource provider trait set
289b46c2-bba7-4fd7-9f7e-8dc631bc723a --trait
CUSTOM_NVIDIA_A100_3_20C
```

```
./) [root@awesome-crayfish ~]# openstack allocation candidate list --required CUSTOM_NVIDIA_A100_3_20C --resource VGPU=1
+-----+-----+-----+-----+
| allocation | resource provider | inventory used/capacity | traits |
+-----+-----+-----+-----+
| VGPU=1     | 92d4cad8-60c2-4055-96f3-836af6027c69 | VGPU=0/1                | CUSTOM_NVIDIA_A100_3_20C |
| VGPU=1     | 289b46c2-bba7-4fd7-9f7e-8dc631bc723a | VGPU=0/1                | CUSTOM_NVIDIA_A100_3_20C |
```

# Provisioning a VM with a vGPU!

- Create a new flavor with resources:VGPU=1
  - ◆ This can be adjusted to assign multiple vGPUs to a single VM
- Or provision a VM with an existing flavor and set the metadata ad-hoc
- Inside the VM
  - ◆ Install the NVIDIA Guest GRID driver
  - ◆ Ensure nvidia-gridd is running for licensing
  - ◆ Must have a license server running
- Create the local compute instance & verify

```
nvidia-smi mig -cci  
nvidia-smi mig -lci
```



Existing Metadata

resources:VGPU 1

```
root@vgpuimgtest2:~# nvidia-smi mig -lci  
+-----+  
| Compute instances: |  
| GPU   GPU   Name           Profile  Instance  Placement |  
|      Instance ID           ID        ID        Start:Size |  
|-----+-----+-----+-----+-----+-----+  
| 0     0     MIG 1g.5gb       0        0         0:1      |  
+-----+-----+-----+-----+-----+-----+-----+  
+-----+  
+-----+
```



**Thank you!**

