Condivisione di una singola GPU verso multiple macchine virtuali attraverso la virtualizzazione delle GPU

C. Pisa¹, D. Passalacqua¹, A. Colla¹, A. Barchiesi¹, F. Galeazzi¹, M. Di Fazio¹, M. Lorini¹, F. Lombardo¹, E. Petagna¹, S. Rabellino², I. Colonnelli², M. Aldinucci², S. Donetti²

¹ Consortium GARR, Italy
² University of Turin, Italy
GARR Cloud Services

- Federated GARR Cloud Platform (IaaS)
- GARR Container Platform (CaaS)
- Deployment as a Service (DaaS)
- GARR Workplace (SaaS)
7600 cores
13.5 PB
333 TFlop

... 11 rack/CSD-modules
Team CSD in real life
Users vs GPU workloads

- Increasing number of users
- Increasing popularity of GPU workloads
- GPU hardware scarcity
- GPU virtualization!
Making GPUs available to users
GPU virtualization can be challenging.

• Slicing GPUs can be non-trivial
• e.g. For NVIDIA V100 GPUs, several steps need to be taken and several technical issues need to be tackled
GPU virtualization in OpenStack

- Get Nvidia GRID license and install license manager
- Download and install drivers on the hypervisor servers
- Configure OpenStack nova
  - Choose one of the supported vGPU types, depending on your use case

<table>
<thead>
<tr>
<th>Virtual GPU Type</th>
<th>Intended Use Case</th>
<th>Frame Buffer (MB)</th>
<th>Maximum vGPUs per GPU</th>
<th>Maximum vGPUs per Board</th>
<th>Maximum Display Resolution</th>
<th>Virtual Displays per vGPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>V100DX-32C</td>
<td>Training Workloads</td>
<td>32768</td>
<td>1</td>
<td>1</td>
<td>4096x2160</td>
<td>1</td>
</tr>
<tr>
<td>V100DX-16C</td>
<td>Training Workloads</td>
<td>16384</td>
<td>2</td>
<td>2</td>
<td>4096x2160</td>
<td>1</td>
</tr>
<tr>
<td>V100DX-8C</td>
<td>Training Workloads</td>
<td>8192</td>
<td>4</td>
<td>4</td>
<td>4096x2160</td>
<td>1</td>
</tr>
<tr>
<td>V100DX-4C</td>
<td>Inference Workloads</td>
<td>4096</td>
<td>8</td>
<td>8</td>
<td>4096x2160</td>
<td>1</td>
</tr>
</tbody>
</table>

https://docs.nvidia.com/grid/13.0/grid-vgpu-user-guide/index.html#installing-configuring-grid-vgpu

```
$ cat /sys/class/mdev_bus/0000:1a:00.0/mdev_supported_types/nvidia-315/name
GRID V100DX-4C
```

- Add configuration to nova-compute.conf

  ```
  [devices]
  enabled_vgpu_types = nvidia-315
  ```
GPU Virtualization in OpenStack

- Create a new flavor
- Download the vGPU drivers
- Create a new VM and install vGPU drivers

Nice but... there is an increasing demand from our users for GPU powered Kubernetes workloads
Enable GPU powered Kubernetes workloads

- Create GPU powered Kubernetes workers using the vGPU OpenStack flavor created previously*
- Enable Nvidia device plugin on Kubernetes Workers*
- Enable Kubernetes API PodTolerationRestriction admission controller and “flag” GPU enabled Kubernetes worker nodes*

- Now users can create vGPU powered Kubernetes workloads!

* these operations can be performed automatically through Juju
But…

- **Issue:** Linux devices associated with virtual GPUs are not persistent across reboots
  - **Solution:**
    - save the list of devices during operation
    - restore them at boot time
  - **Our scripts available at**
    https://github.com/ConsortiumGARR/gpu_mdev_scripts
Wrap-Up

We have increased the availability of GPU resources for our users by leveraging virtualization

- GPUs are scarce compared to the demand from users
- Several use cases don’t need all the resources of a physical GPU
- => More Happy Users!

Future work:

- Benchmark vGPU vs GPU Passthrough
- Improve automatic deployment of new GPU servers
- Improve GPU reservation and accounting
- Integrate A30/A100 GPU hardware
- Mix different vGPU types (/2, /4, /8)
Thank you for your attention!