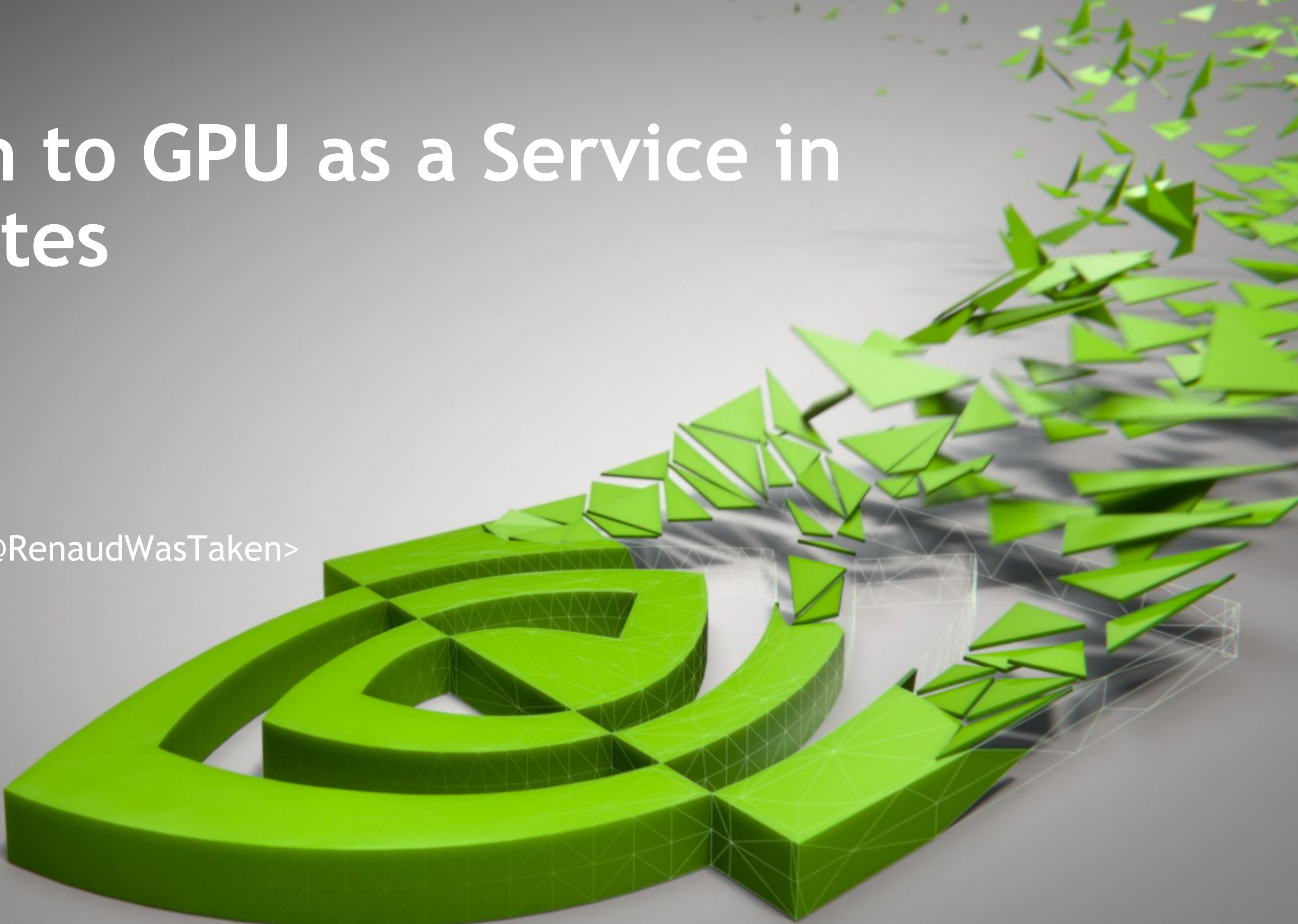
The Path to GPU as a Service in Kubernetes

March 27, 2018

Viraj Chavan
Renaud Gaubert <@RenaudWasTaken>



Kubernetes
The State of GPUs

The State of GPUs in Kubernetes

In 1.7 and Before: --accelerators

- Completely experimental support (1.6 supports one GPU / node)
- Manually mount the volumes in your pod spec
- No GPU Monitoring or Health check
    - Black Hole effect
- Not supported by NVIDIA

The State of GPUs in Kubernetes

In 1.8 and 1.9: Device Plugin

- Pluggable System in alpha state
- GPU Health check
- Official NVIDIA support
  - Through the use of the new NVIDIA container runtime

Limits:

- It's an alpha feature
  - 1.8 Plugins are not compatible with 1.9
  - 1.9 plugins are not compatible with 1.10
- You might get some races when Kubelet restarts
- Init Containers are counted as regular containers in 1.8
- Homogeneous nodes only (e.g: You can't have a 1070 and a 1080 on the same node)

The State of GPUs in Kubernetes

In 1.10

- Graduated to a Beta system
- GPU Metrics are now advertised by cAdvisor
- Complete CRIO support

The State of GPUs in Kubernetes

Going Forward

- Kubernetes is still missing a number of important features for GPUs:
    - NUMA
    - GPU Topology
    - Multi-node
    - GPU sharing
    - GPU attributes
    - More GPU metrics
    - GPU soft quotas?

Kubernetes Optimized For NVIDIA GPUs

KUBERNETES Optimized for NVIDIA GPUs

Mission

- A specialized Kubernetes for specialized computing
- Maximize individual GPU utilization and cluster level GPU occupancy
- Provide Early access to complex GPU features
- Provide Frictionless adoption of Kubernetes for NVIDIA GPUs

KUBERNETES Optimized for NVIDIA GPUs

Why

- Similar to TensorFlow we will upstream features as fast as possible
  - We want to provide these features today, not a year from now
- Some features are specific to GPUs and don’t need to be in core Kubernetes
- Single product offer rather than 10 plugins
- Support for upstream changes

FEATURE OVERVIEW

Full Docker Runtime support

Cluster Admin Facing

- **Use case**
  - I want the minimum amount of setup when provisioning a node
  - I want the NVIDIA runtime to be ran only for NVIDIA images
- **Before**
  - The NVIDIA runtime was ran for all images (default runtime)
  - Images that did not request GPUs might have all GPUs exposed
- **After:** The NVIDIA runtime is selected only for NVIDIA images

Full CRI-O Runtime support

Cluster Admin Facing

- **Use case**
  - For enterprises customers running RHEL, CRI-O is becoming the default runtime
- **Before**
  - Same issues as Docker (default runtime, …)
- **After:** The NVIDIA runtime is selected only for NVIDIA images
- Additionally this will be in upstream 1.10

GPU Attributes

Cluster Admin and User Facing

- **Use case**
  - I want to request 2 different GPUs
  - I want to request N GPUs with a minimum of 16Gb
- **Before:** only homogeneous nodes + manually label nodes with GPU attributes
  - Attributes needed to be exposed automatically
  - Attributes needed an explicit API
- **After:** GPU selection can be done on Memory, Compute Capability, ECC

GPU Sharing

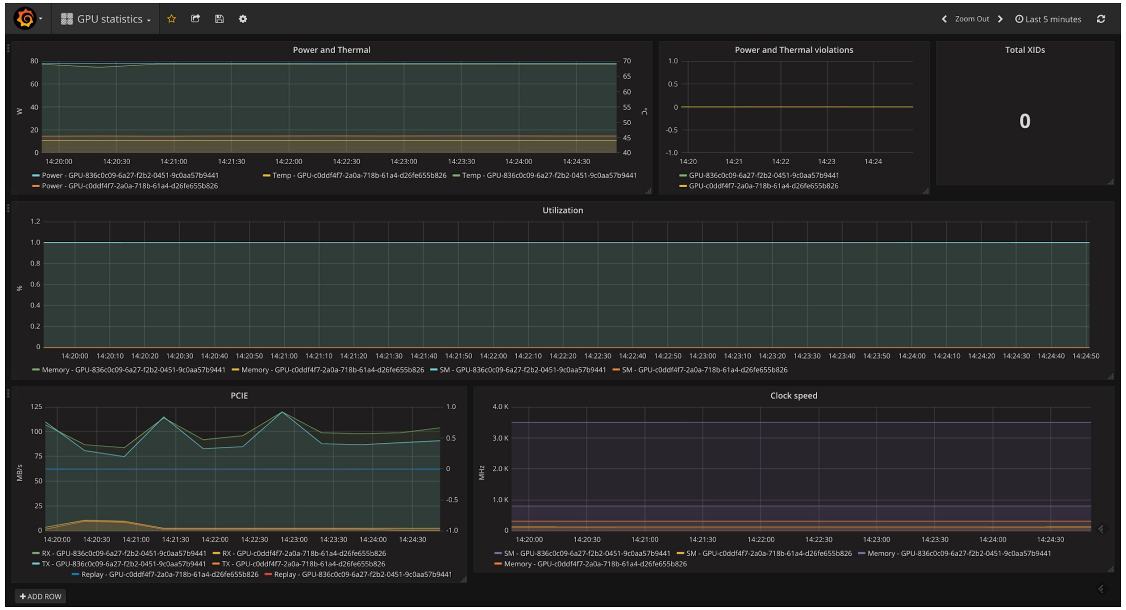
User Facing

- **Use case**
  - Sharing the same GPU between multiple containers
  - Requesting “shares” of any GPU for a container
- **Before: No sharing**

GPU Monitoring

Cluster Admin and User Facing

- **Use case**
  - Monitor GPU usage and Health
  - Prometheus and cadvisor
  - Per-process/container monitoring



- **Before: no or little GPU monitoring (1.10)**

Hard Quotas

- **Use case**
  - Limit the number of GPUs / namespace
- **Before: No quotas**
- Will be upstreamed in 1.10

FEATURES IN THE RACKS

NUMA and Topology

User Facing

- **Use case**
  - As a GPU Software Engineer I want my application to run as fast as possible
    - I want my container to be pinned to the CPU(s) that matches my GPU(s)
    - I want a NIC on the same NUMA node as my GPU(s)
    - I might want to select the minimum interconnection between my GPUs (QPI, Bridge, Switch, NVLINK, 2xNVLINK)
  - As a cluster admin I want to maximize GPU occupancy
    - A common workaround the NUMA issue is to request all the GPUs on a node
    - Even though you might only need 2/3/5/6/…
- **Today: NUMA and Topology not handled**

Batch Scheduling

- **Use case**
  - Run MPI jobs on a Kubernetes cluster
- **Before: No support for batch scheduling**
- We need to sync efforts with the NGC team

Dive into the Architecture

Conclusion

Thank You!

NVIDIA

Viraj Chavan
Renaud Gaubert