redhat.

Red Hat and the NVIDIA DGX: Tried, Tested, Trusted

NVIDIA GTC 2019

Jeremy Eder, Andre Beausoleil, Red Hat

Agenda

redhat

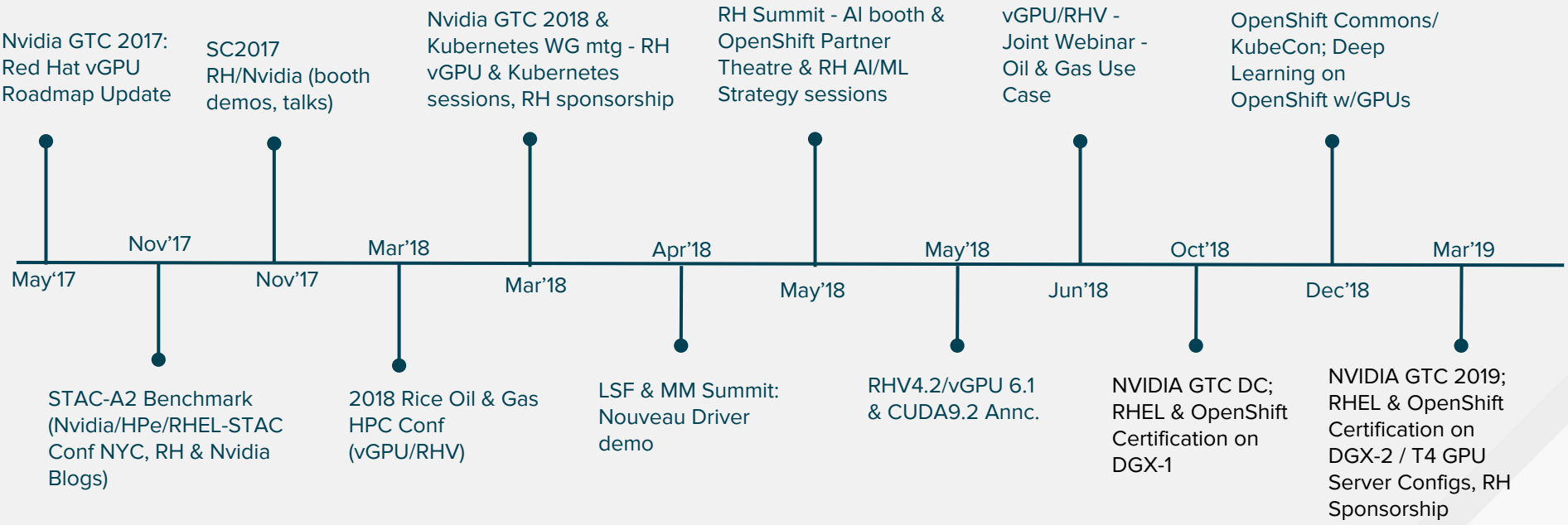
- Red Hat + NVIDIA Partnership Overview
- Announcements / What's New
- OpenShift + GPU Integration Details

redhat.

Where Red Hat Partners with NVIDIA

- GPU accelerated workloads in the enterprise
  - AI/ML and HPC
- Deploy and manage NGC containers
  - On-prem or public cloud
- Managing virtualized resources in the data center
  - vGPU for technical workstation
- Fast deployment of GPU resources with Red Hat
  - Easy to use driver framework

Red Hat/NVIDIA Technology Partnership Timeline



redhat.

Red Hat + NVIDIA: What's New?

- Red Hat Enterprise Linux Certification on DGX-1 & DGX-2 systems
  - Support for Kubernetes-based, OpenShift Container Platform
  - NVIDIA GPU Cloud (NGC) containers to run on RHEL and OpenShift
- Red Hat's OpenShift provides advanced ways of managing hardware to best leverage GPUs in container environments
- NVIDIA developed precompiled driver packages to simplify GPU deployments on Red Hat products
- NVIDIA's latest T4 GPUs are available on Red Hat Enterprise Linux
  - T4 Server with RHEL support from most major OEM server vendors
  - T4 servers are "NGC-Ready" to run GPU containers

Red Hat + NVIDIA: Open Source Collaboration

Open Source Projects

- Heterogeneous Memory Management (HMM)
  - Memory management between device and CPU
- Nouveau Driver
  - Graphics device driver for NVIDIA GPU
- Mediated Devices (mdev)
  - Enabling vGPU through the Linux kernel framework
- Kubernetes Device Plugins
  - Fast and direct access to GPU hardware
  - Run GPU enabled containers in Kubernetes cluster

redhat.

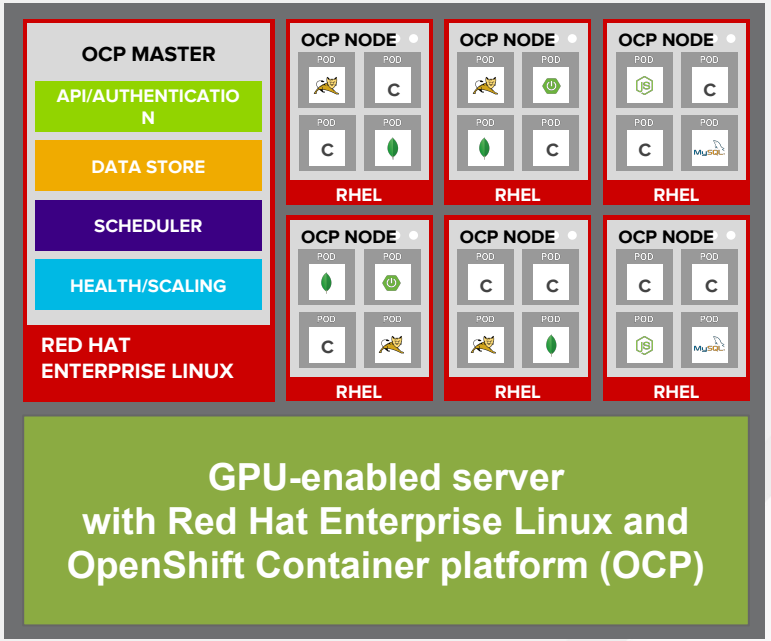
Red Hat OpenShift Container Platform

OPENSHIFT - CONTAINER PLATFORM FOR AI

Enable Kubernetes clusters to seamlessly run accelerated AI workloads in containers

Red Hat is delivering required functionality to efficiently run AI/ML workloads on OpenShift

- 3.10, 3.11
  - Device plugins provide access to FPGAs, GPGPUs, SoC and other specialized HW to applications running in containers
  - CPU Manager provides containers with exclusive access to compute resources, like CPU cores, for better utilization
  - Huge Pages Support enables containers with large memory requirements to run more efficiently
- 4.0
  - Multi-network feature allows more than one network interface per container for better traffic management



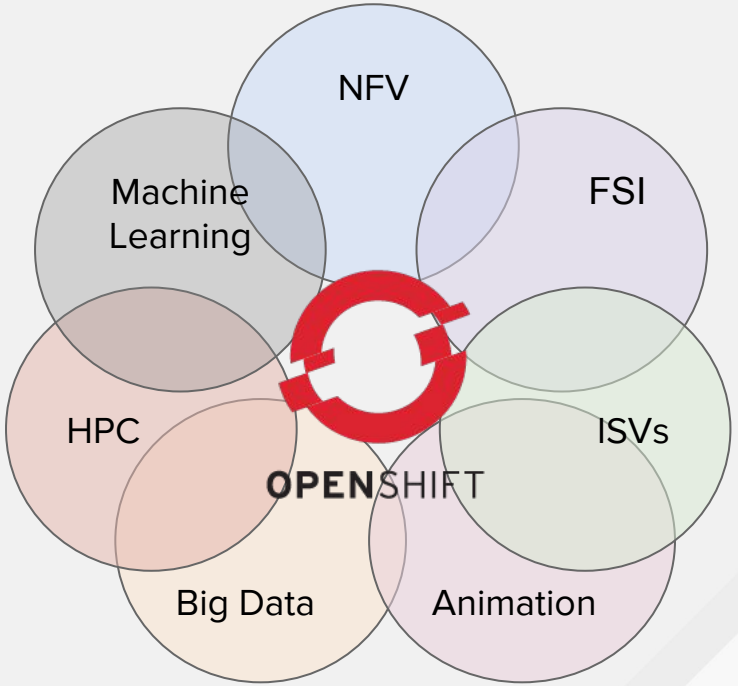
redhat.

One Platform to...

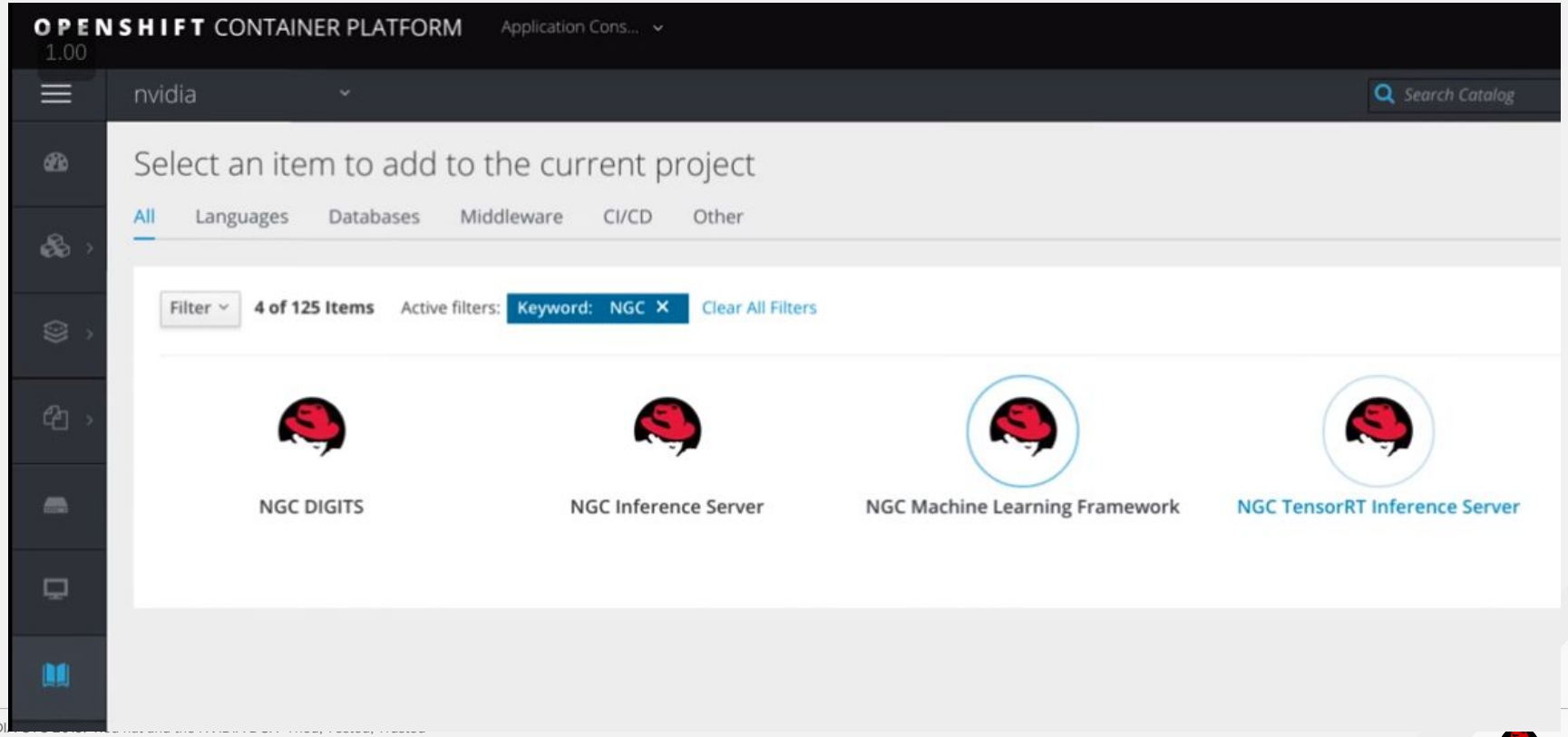
OpenShift is the **single platform**

to run any application:

- Old or new
- Monolithic/Microservice



Data Scientist User Experience (Service Catalog)



redhat.

Upstream First: Kubernetes Working Groups

- Resource Management Working Group
  - Features Delivered
    - Device Plugins (GPU/Bypass/FPGA)
    - CPU Manager (exclusive cores)
    - Huge Pages Support
  - Extensive Roadmap
- Intel, IBM, Google, NVIDIA, Red Hat, many more...

Upstream First: Kubernetes Working Groups

- Network Plumbing Working Group
  - Formalized Dec 2017
- Implemented a multi-network specification:

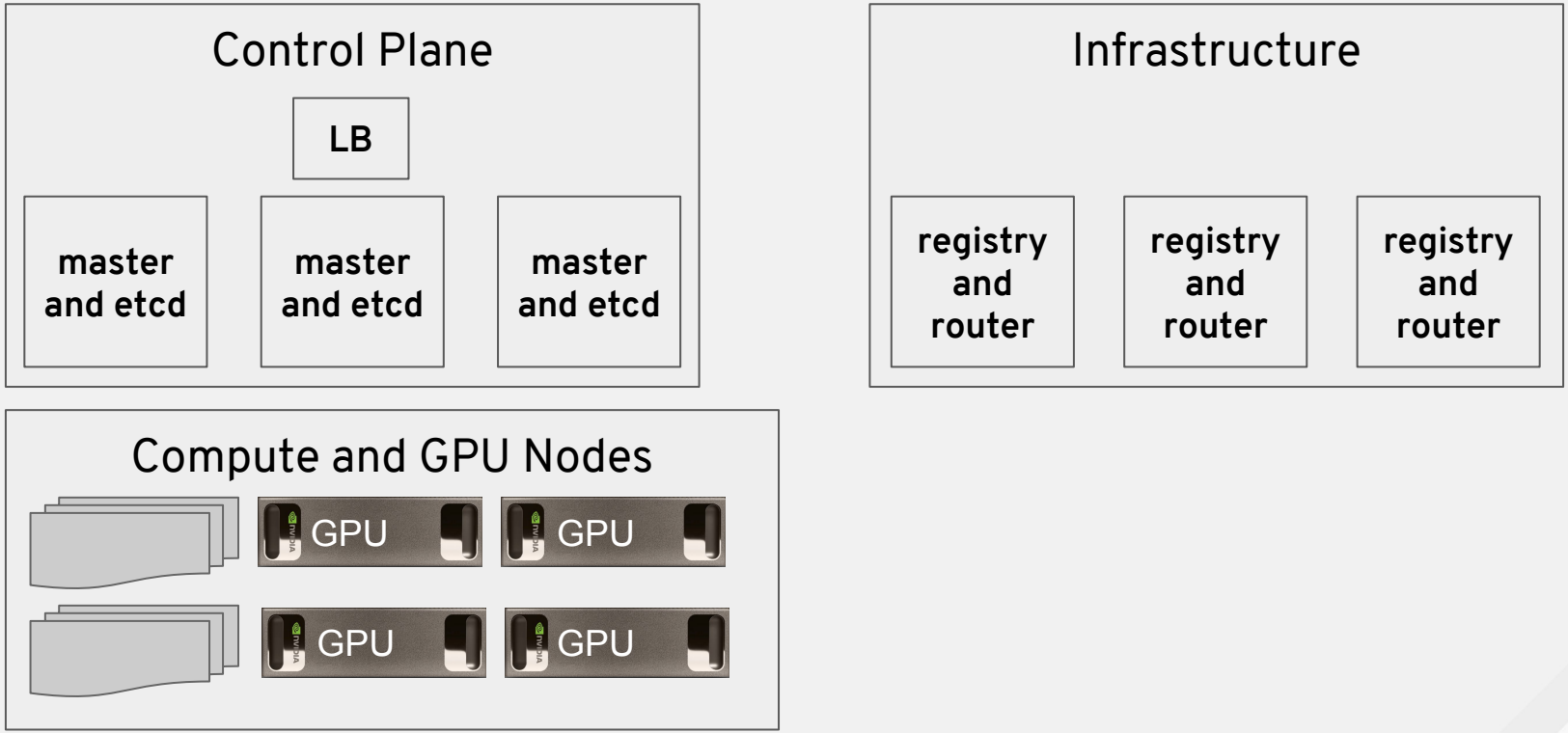
  https://github.com/K8sNetworkPlumbingWG/multi-net-spec

  (collection of CRDs for multiple networks, owned by sig-network)
- Reference Design implemented in Multus CNI by Red Hat
- Separate control- and data-plane, Overlapping IPs, Fast Data-plane
- IBM, Intel, Red Hat, Huawei, Cisco, Tigera...at least.

redhat.

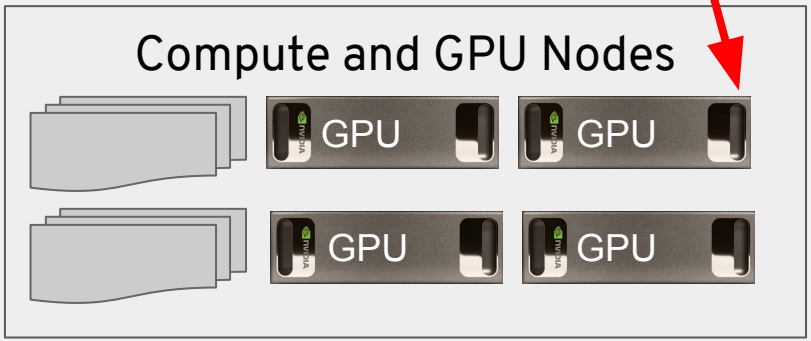
GPU Cluster Topology

What does an OpenShift (OCP) Cluster look like?



OpenShift Cluster Topology

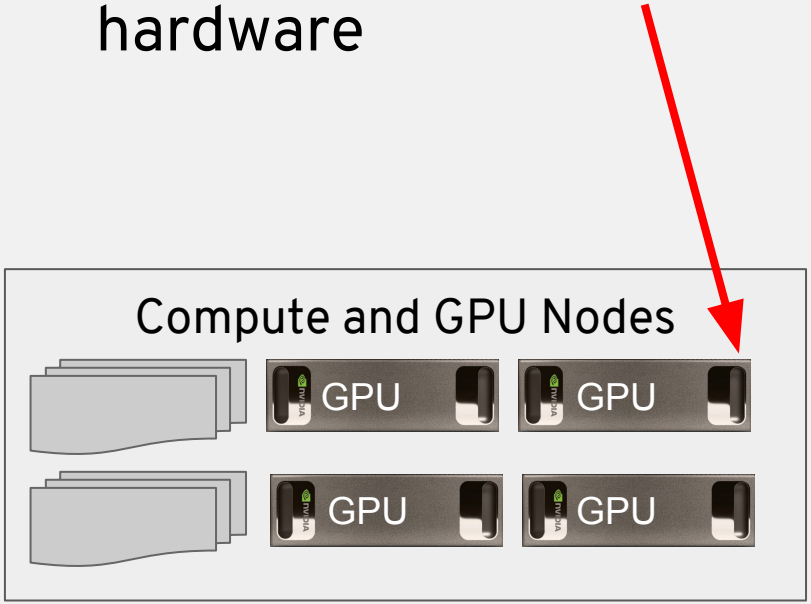
- How to enable software to take advantage of “special” hardware



- Create Node Pools
  - MachineSets
  - Mark them as “special”
  - Taints/Tolerations
  - Priority/Preemption
  - ExtendedResourceToleration

OpenShift Cluster Topology

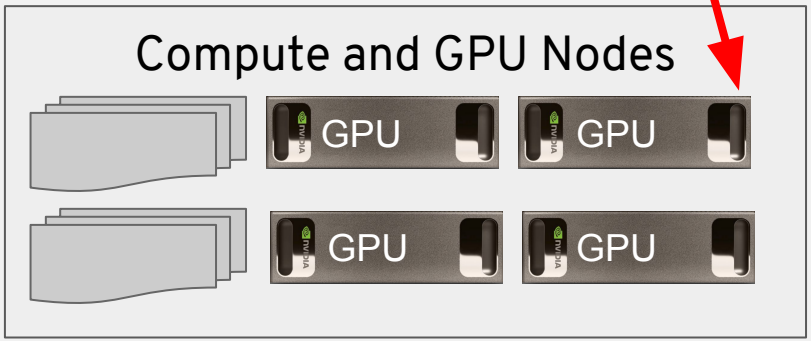
- How to enable software to take advantage of “special” hardware



- Tune/Configure the OS
  - Tuned Profiles
  - CPU Isolation
  - sysctls

OpenShift Cluster Topology

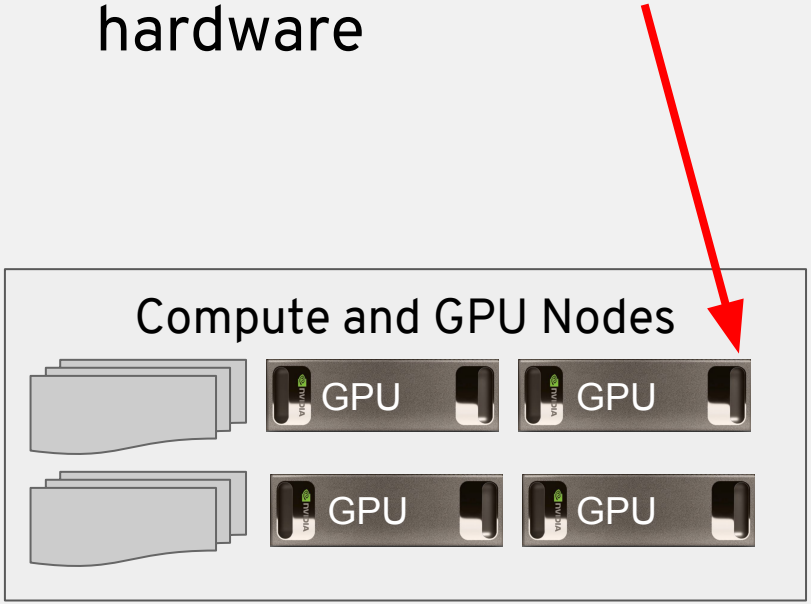
- How to enable software to take advantage of “special” hardware



- Optimize your workload
  - Dedicate CPU cores
  - Consume hugepages

OpenShift Cluster Topology

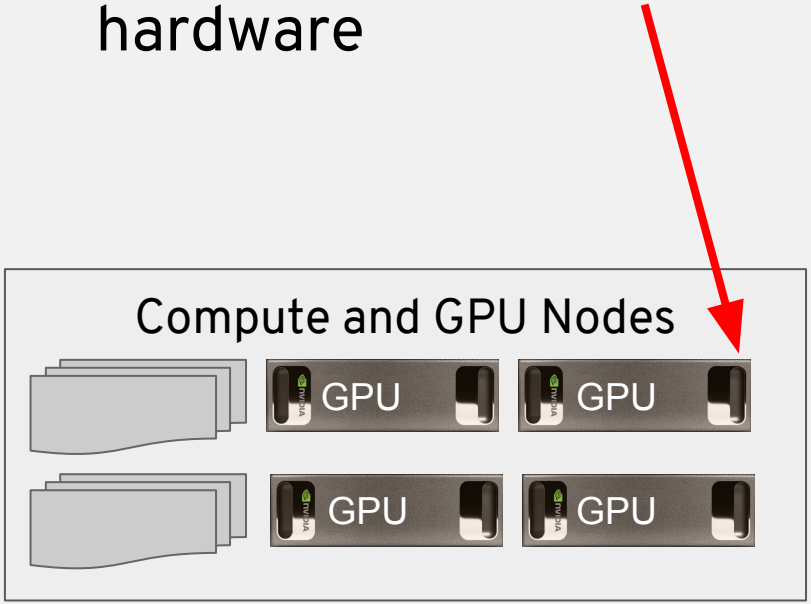
- How to enable software to take advantage of “special” hardware



- Enable the Hardware
  - Install drivers
  - Deploy Device Plugin
  - Deploy monitoring

OpenShift Cluster Topology

- How to enable software to take advantage of “special” hardware



- Consume the Device
  - KubeFlow Template deployment

redhat.

Support Components

Cluster Node Tuning Operator (tuned)

OpenShift node-level tuning operator

- Consolidate/Centralize node-level tuning (openshift-ansible)
- Set tunings for Elastic/Router/SDN
- Add more flexibility to add custom tuning specified by customers
- NVIDIA DGX-1 & DGX-2 Tuned Profiles

Node Feature Discovery Operator (NFD)

Steer workloads based on infrastructure capabilities

- Git Repos:
  - Upstream
  - Downstream
- Client/Server model
- Customize with "hooks"

Labels:
feature.node.kubernetes.io/cpu-hardware_multithreading=true
feature.node.kubernetes.io/cpuid-AVX2=true
feature.node.kubernetes.io/cpuid-SSE4.2=true
feature.node.kubernetes.io/kernel-selinux.enabled=true
feature.node.kubernetes.io/kernel-version.full=3.10.0-957.5.1.el7.x86_64
feature.node.kubernetes.io/pci-0300_10de.present=true
feature.node.kubernetes.io/storage-nonrotationaldisk=true
feature.node.kubernetes.io/system-os_release.ID=rhcos
feature.node.kubernetes.io/system-os_release.VERSION_ID=4.0
feature.node.kubernetes.io/system-os_release.VERSION_ID.major=4
feature.node.kubernetes.io/system-os_release.VERSION_ID.minor=0

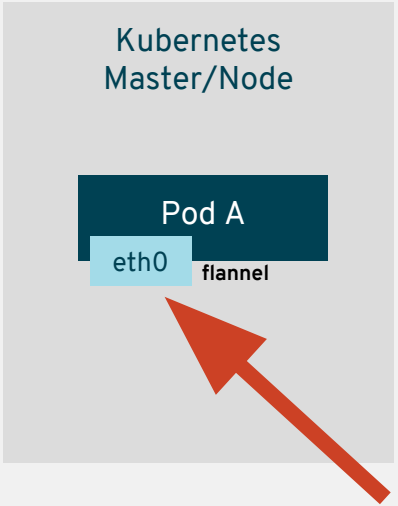
https://github.com/intel/multus-cni

NFV Partner Engineering along with the Network Plumbing Working Group is using Multus as part of a reference implementation.

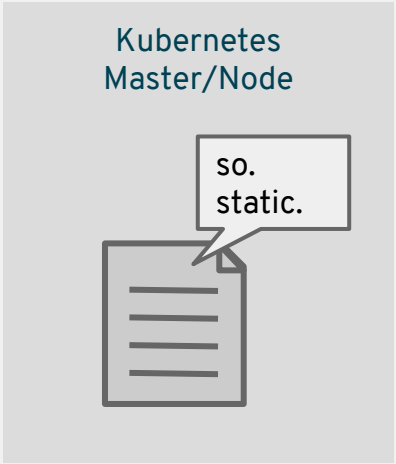
Multus CNI is a “meta plugin” for Kubernetes CNI which enables one to attach multiple network interfaces on each pod. It allows one to assign a CNI plugin to each interface created in the pod.

THE PROBLEM (Today)

#1 Each pod only has one network interface

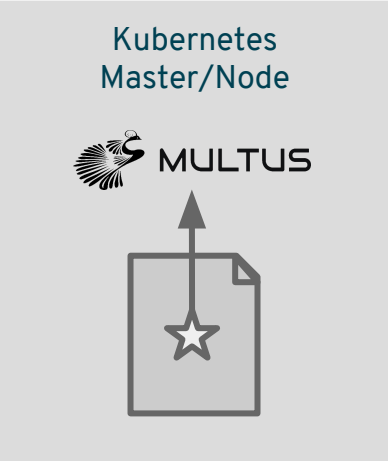


#2 Each master/node has only one static CNI configuration

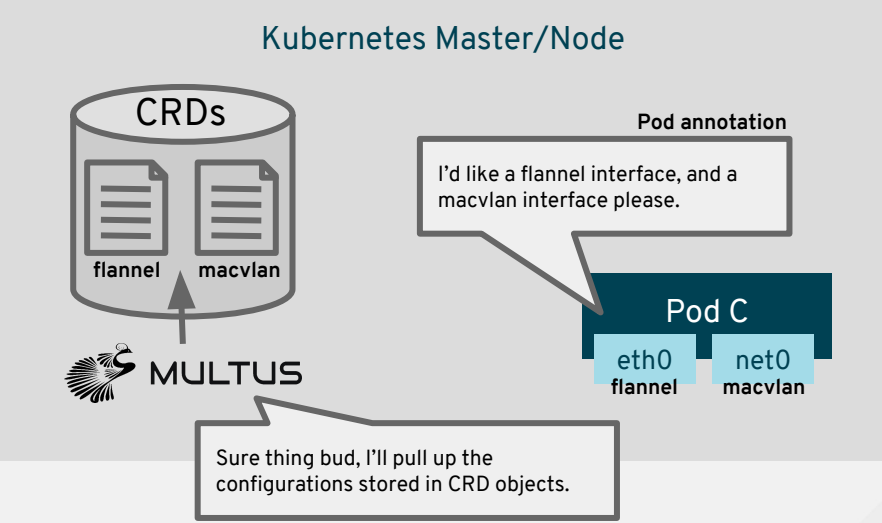


THE SOLUTION (Today)

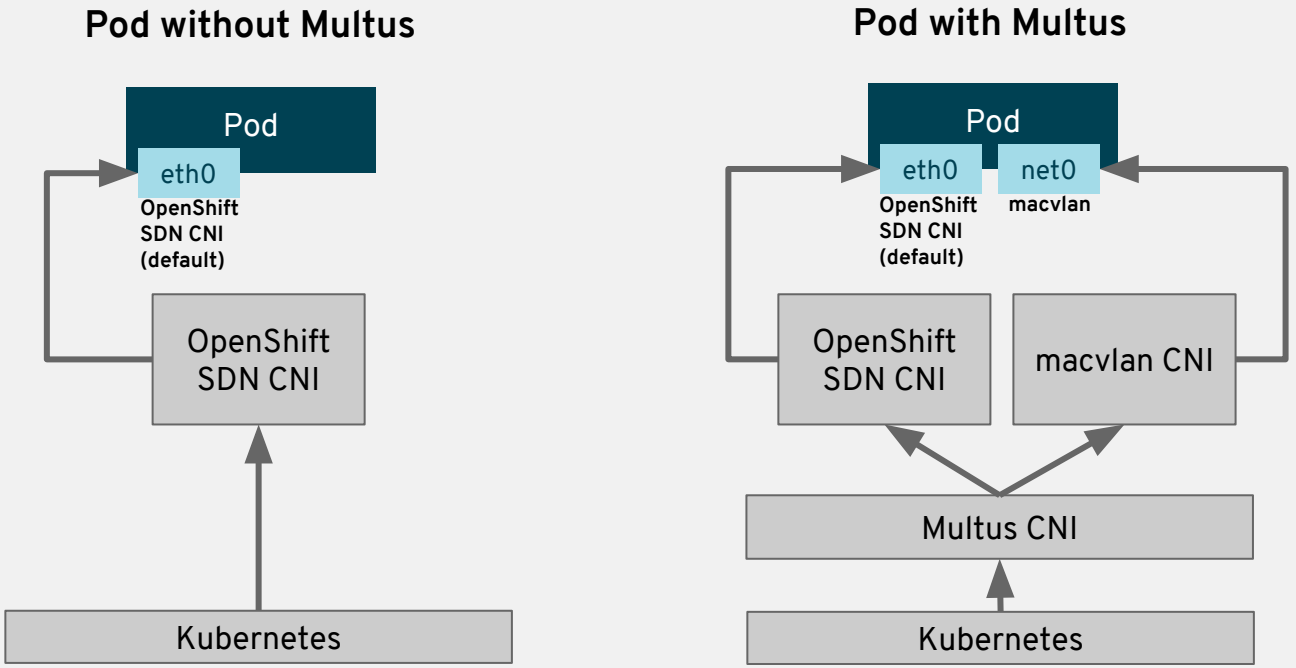
Static CNI configuration points to Multus



Each subsequent CNI plugin, as called by Multus, has configurations which are defined in CRD objects



WHAT MULTUS DOES



Standardized CRD

As currently proposed by Network Plumbing Working Group.

Pod annotations

```
apiVersion: v1
kind: Pod
metadata:
  name: pod_c
  annotations:
    kubernetes.cni.cncf.io/networks: '[
        { "name": "flannel-conf" },
        { "name": "macvlan-conf" }
    ]'
spec:
  containers: [...]
```

Maps to...

CRD Object

```
Name:         macvlan-conf
Namespace:    default
Labels:       <none>
Annotations:  <none>
API Version:  cni.cncf.io/v1
Args:         [ { "master": "eth0", "mode": "bridge", ...
Kind:         Network
Plugin:       macvlan
Metadata: [...]
```

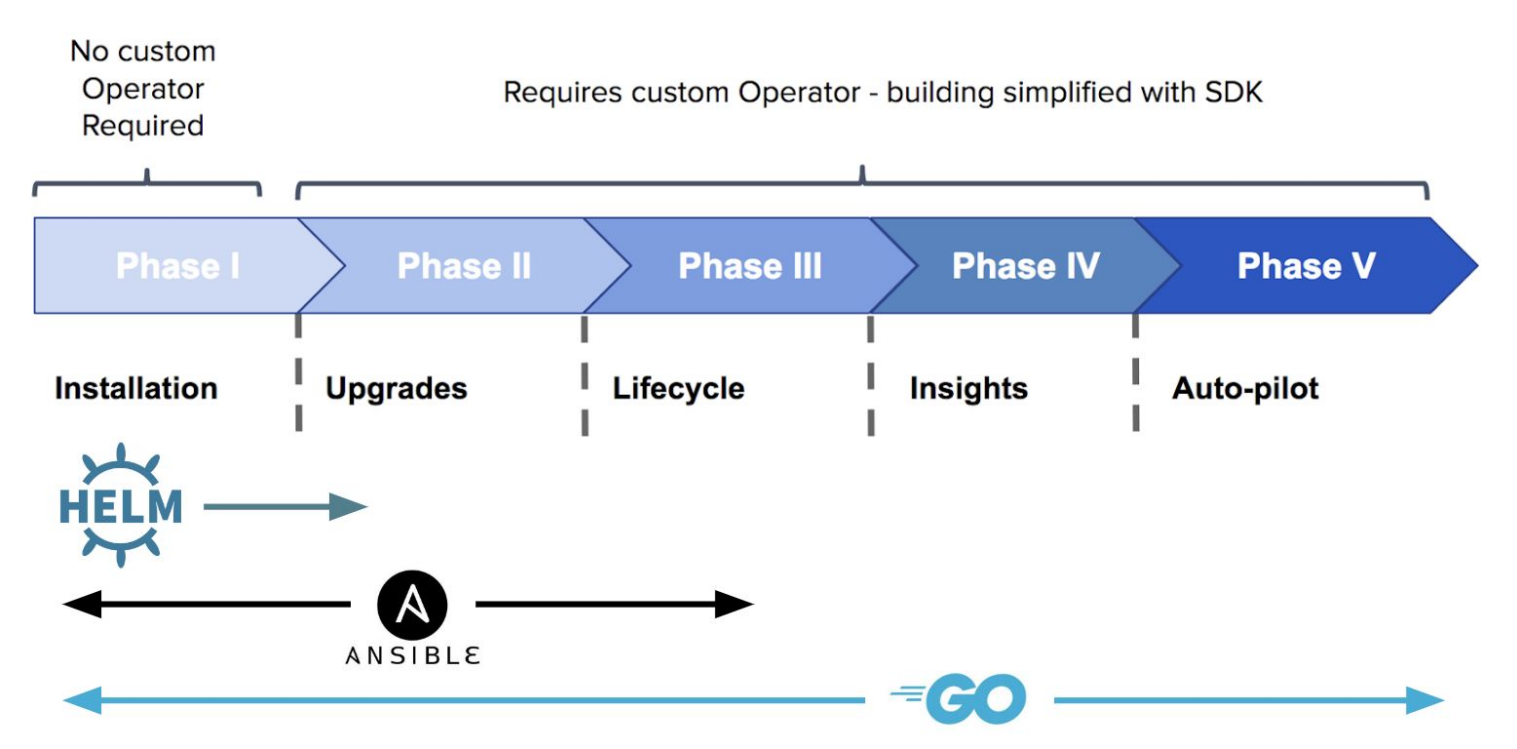
The specification uses annotations to call out a list of intended network attachments as “sidecar networks”

CNI network configurations are packed inside CRD objects.

redhat.

Installation and Day 2 Management of NVIDIA GPUs in OpenShift 4

Roadmap: Operationalizing GPUs on OpenShift 4



Roadmap: Specialized Hardware in OpenShift 4

machine-api-operator

machine-config-operator

cluster-network-operator

openshift-multus daemonset

cluster-nfd-operator

cluster-node-tuning-operator

special-resource-operator (GPU)

special-resource-operator (NIC)

prometheus/grafana dashboards

Roadmap: Special Resource Operator

OpenShift Node | GPU | FPGA | NIC | OTHER

Blue Boxes: owned, supported, shipped by **Red Hat**

Green Boxes: owned, supported, shipped by **Partner**

Cluster Node Tuning Operator (next gen tuned)

Node Feature Discovery Operator

Special Resource Operator

Daemonset

- Driver Container (optional)
- Device Plugin Container
- Monitoring Container (Prometheus endpoint)

OpenShift Node | GPU | FPGA | NIC | OTHER

Node Object:

```
Labels:
feature.node.kubernetes.io/pci-0300_10de.present=true

Capacity:
     example.com/gpu: 4
```

Soft or Hard Shared Cluster Partitioning?

Priority and Preemption

- Create PriorityClasses based on business goals
- Annotate pod specs with priorityClassName
- If all GPUs are used
  - A high prio pod is queued
  - A low prio pod is running
  - Kube will preempt low prio pod
    - And schedule high prio pod

- Ensures optimal density

Taints and Toleration

- Taints are “node labels with policies”
  - You can taint a node like
  - nvidia.com/gpu=value:NoSchedule
- Then a pod will have to “tolerate” the nvidia.com/gpu taint, otherwise it won’t run on that node.
- This allows you to create “node pools”

- Could lead to under-utilized resources
- Might make sense for security or business rules

Enforcing Quota on GPUs (per namespace)

Create a quota on a namespace

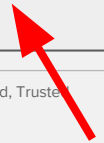
```
# cat gpu-quota.yaml
apiVersion: v1
kind: ResourceQuota
metadata:
  name: gpu-quota
  namespace: nvidia
spec:
  hard:
    requests.nvidia.com/gpu: 1
```

Verify the quota is set

```
# oc describe quota gpu-quota -n nvidia
Name:                    gpu-quota
Namespace:               nvidia
Resource                 Used  Hard
--------                 ----  ----
requests.nvidia.com/gpu  0     1
```

Expected message when exceeding quota

```
# oc create -f gpu-pod.yaml
Error from server (Forbidden): error when creating "gpu-pod.yaml": pods "gpu-pod-f7z2w" is forbidden: exceeded
quota: gpu-quota, requested: requests.nvidia.com/gpu=1, used: requests.nvidia.com/gpu=1, limited:
requests.nvidia.com/gpu=1
```



NVIDIA Driver Packaging

- Red Hat and NVIDIA are collaborating to improve the user experience of NVIDIA's drivers and CUDA Toolkit on RHEL and OpenShift
- Easier install/upgrade through upcoming changes to the driver packaging (e.g., no DKMS required anymore)
  - Let us know if you are interested in a tech preview!
- Improved coordination between NVIDIA and Red Hat regarding testing, release processes, and support
- High-level goal is to make NVIDIA's driver feel more like a normal in-box driver

Roadmap: Special Resource Operator

OpenShift Node | GPU | FPGA | NIC | OTHER

Blue Boxes: owned, supported, shipped by **Red Hat**

Green Boxes: owned, supported, shipped by **Partner**

Cluster Node Tuning Operator (next gen tuned)

Node Feature Discovery Operator

Special Resource Operator

Daemonset

- Driver Container (optional)
- Device Plugin Container
- Monitoring Container (Prometheus endpoint)

OpenShift Node | GPU | FPGA | NIC | OTHER

Node Object:

```
Labels:
feature.node.kubernetes.io/pci-0300_10de.present=true

Capacity:
     example.com/gpu: 4
```

Thank You!

- Come see us @ Booth 716
- Jobs for training / with Priority/Preemption
- Deployments for Inference
- TensorRT on OpenShift

redhat.

THANK YOU

plus.google.com/+RedHat

facebook.com/redhatinc

linkedin.com/company/red-hat

twitter.com/RedHat

youtube.com/user/RedHatVideos

Demo

Link

1. Show no driver
2. Show node labels
3. Show nfd operator and node label differences, focus on PCI row (CPU node and GPU node)
4. Show GPU operator create and tail operator logs
5. Show oc describe node (nvidia.com/gpu=X)
6. Taints and Tolerations? Show oc describe node focus on taints (nvidia.com/gpu:NoSchedule)
7. Priority/Preemption Show oc get priorityclasses
8. GPU workload demo…
9. Send Jeremy the kubeconfig for running cluster