



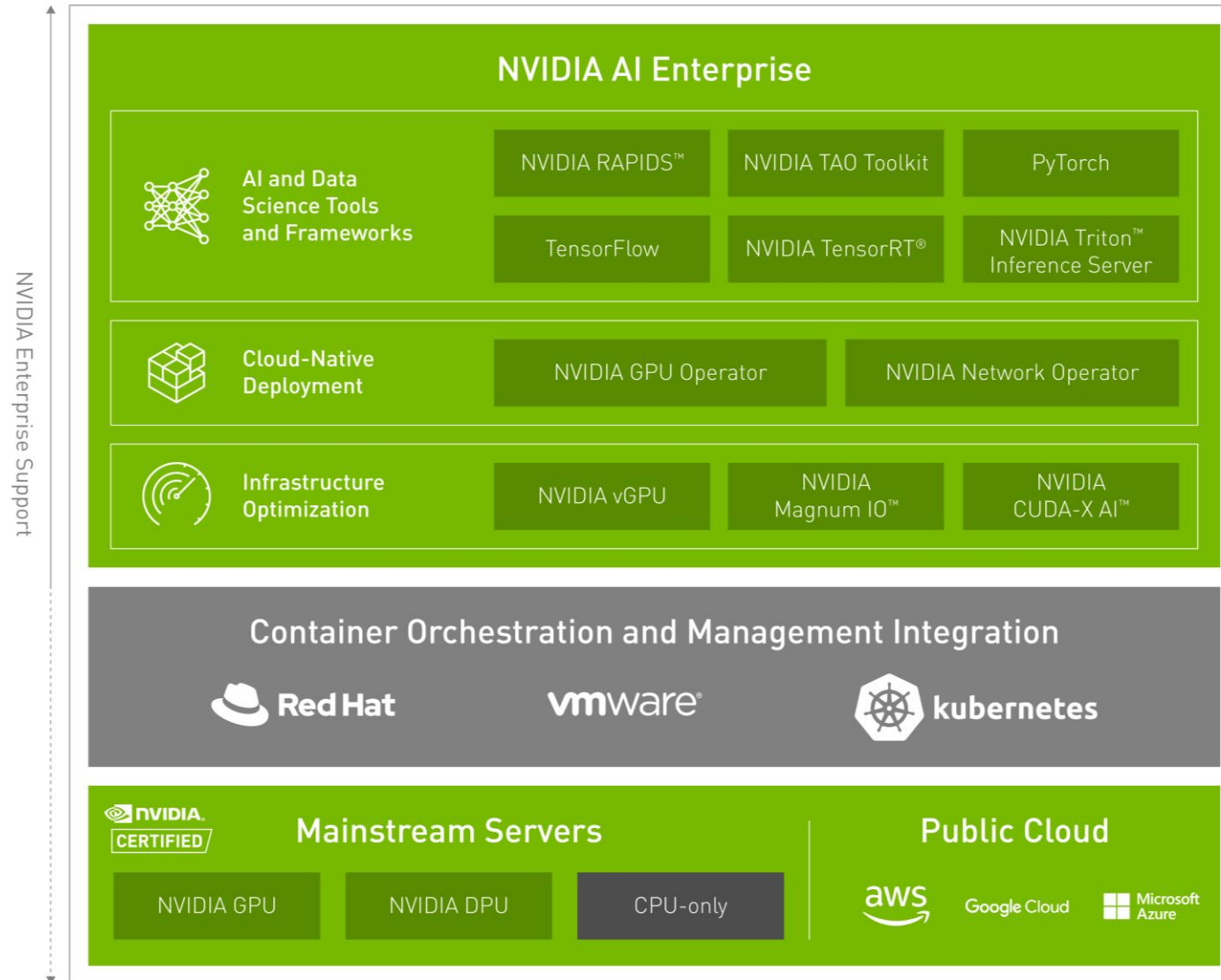
Running Cloud-native Apps in NVIDIA AI Enterprise

Joe Cullen | Technical Marketing Engineer | NVIDIA
Vinay Bagade | Technical Marketing Engineer | NVIDIA



- **NVIDIA AI Enterprise Overview**
 - Delivering AI to the Enterprise
 - NVIDIA Operators
- **Orchestration Methods**
- **Red Hat OpenShift Deployment**
- **Orchestration with OCP and vSphere**
- **Machine Learning Pipeline on Kubernetes**
 - Preprocessing
 - Training
 - Inference

NVIDIA AI ENTERPRISE SOFTWARE SUITE



NVIDIA OPERATORS

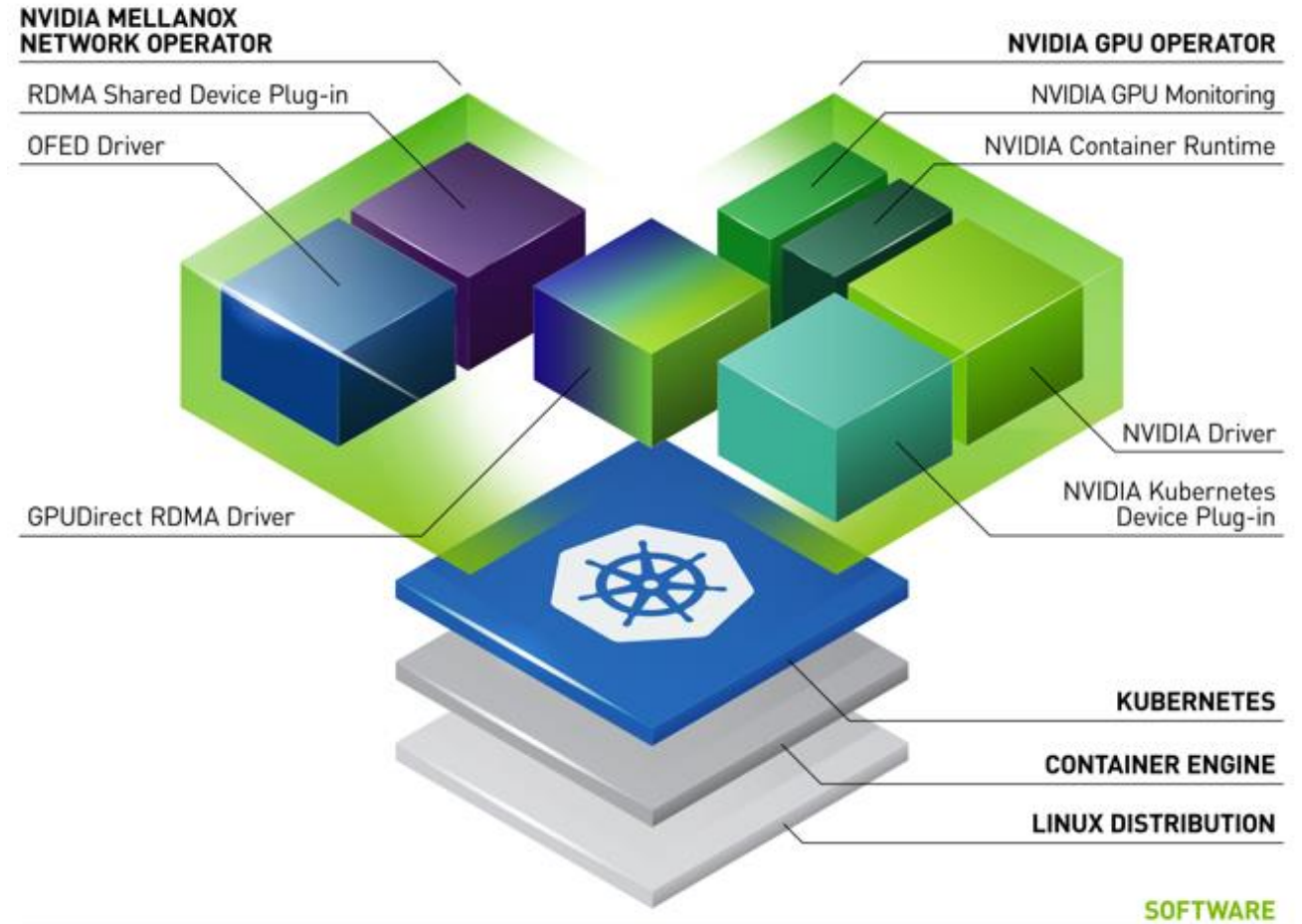
Only NVIDIA AI Enterprise customers have access to containerized vGPU drivers.

GPU Operator installs all software to make GPUs usable by applications running on VMware vSphere with Tanzu.

- Automates the installation of the vGPU Guest Driver, NVIDIA Container Toolkit, Device Plugin, DCGM, etc.
- Automatically scales to newly added GPU accelerated Tanzu nodes.

NVIDIA AI Enterprise customers have access to prebuild vGPU driver images.

- GPU Operator installs a compatible vGPU Guest Driver.
- Only NVIDIA AI Enterprise customers have access to containerized vGPU drivers.

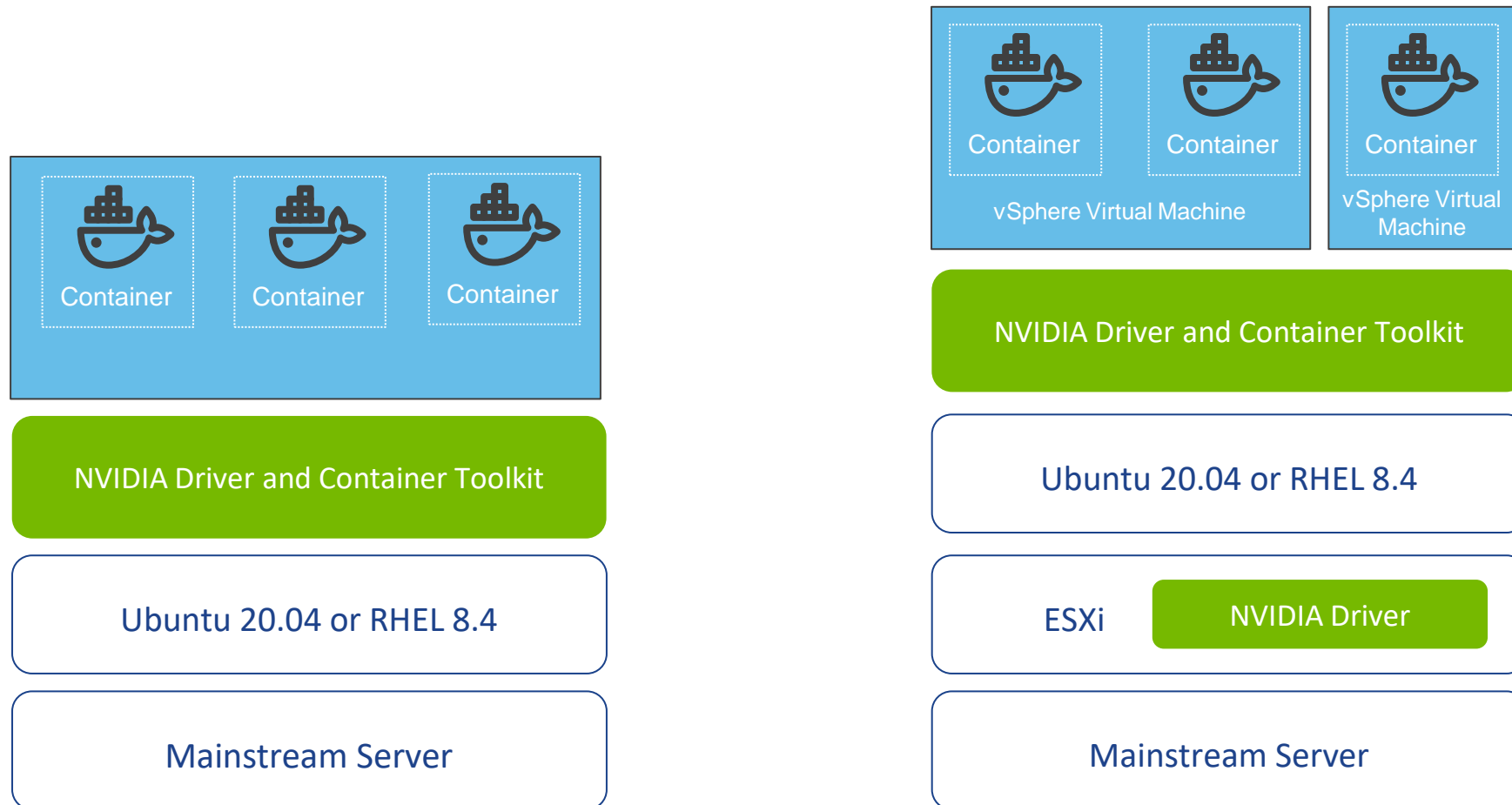


Orchestration Methods



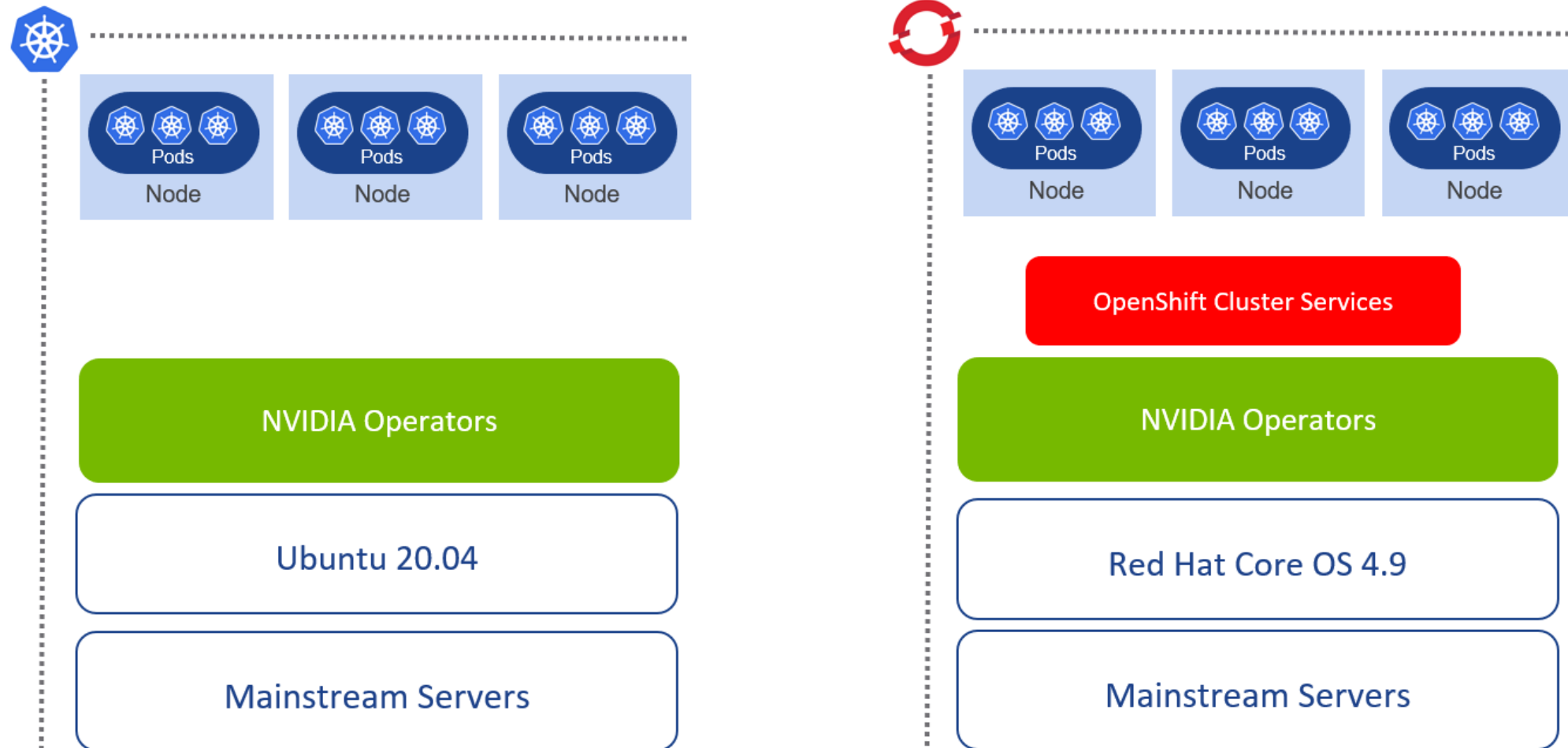
DELIVERING AI WORKLOADS WITH NVIDIA AI ENTERPRISE

Orchestration with Containers



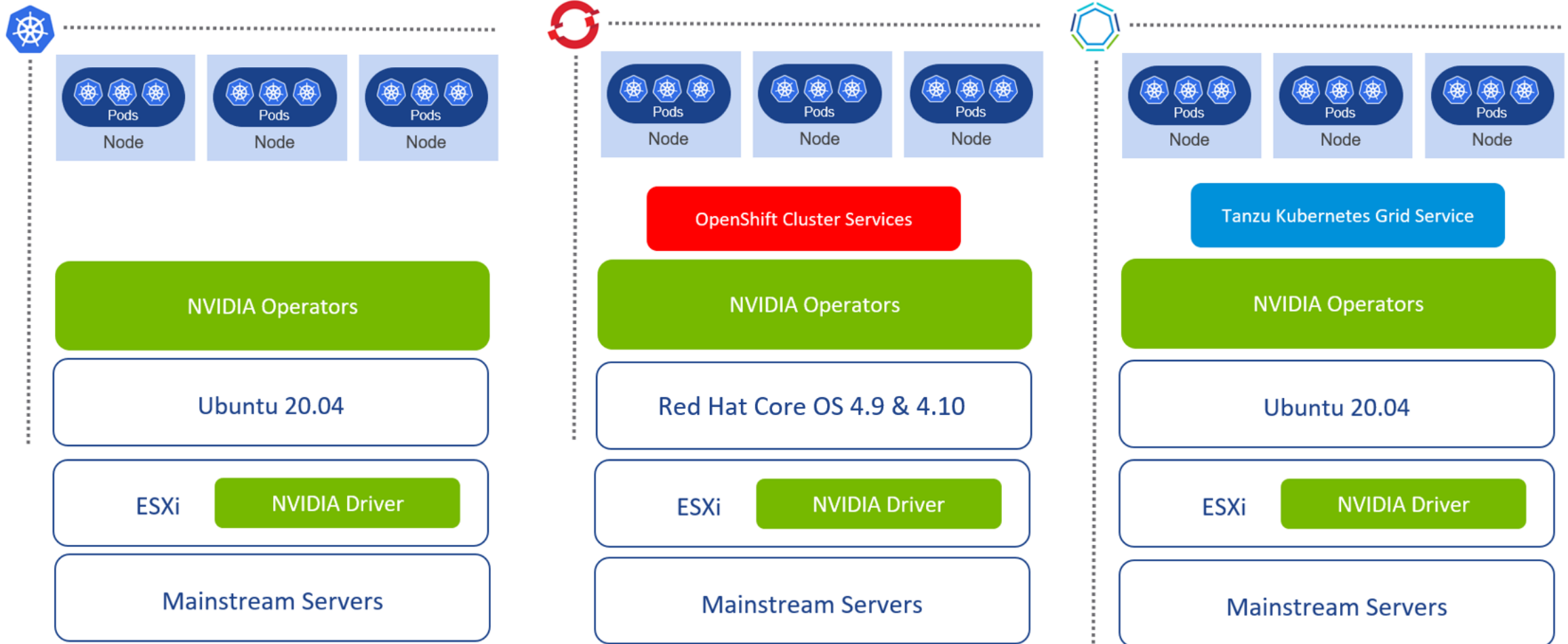
DELIVERING AI WORKLOADS WITH NVIDIA AI ENTERPRISE

Orchestration with Kubernetes on **Bare Metal**

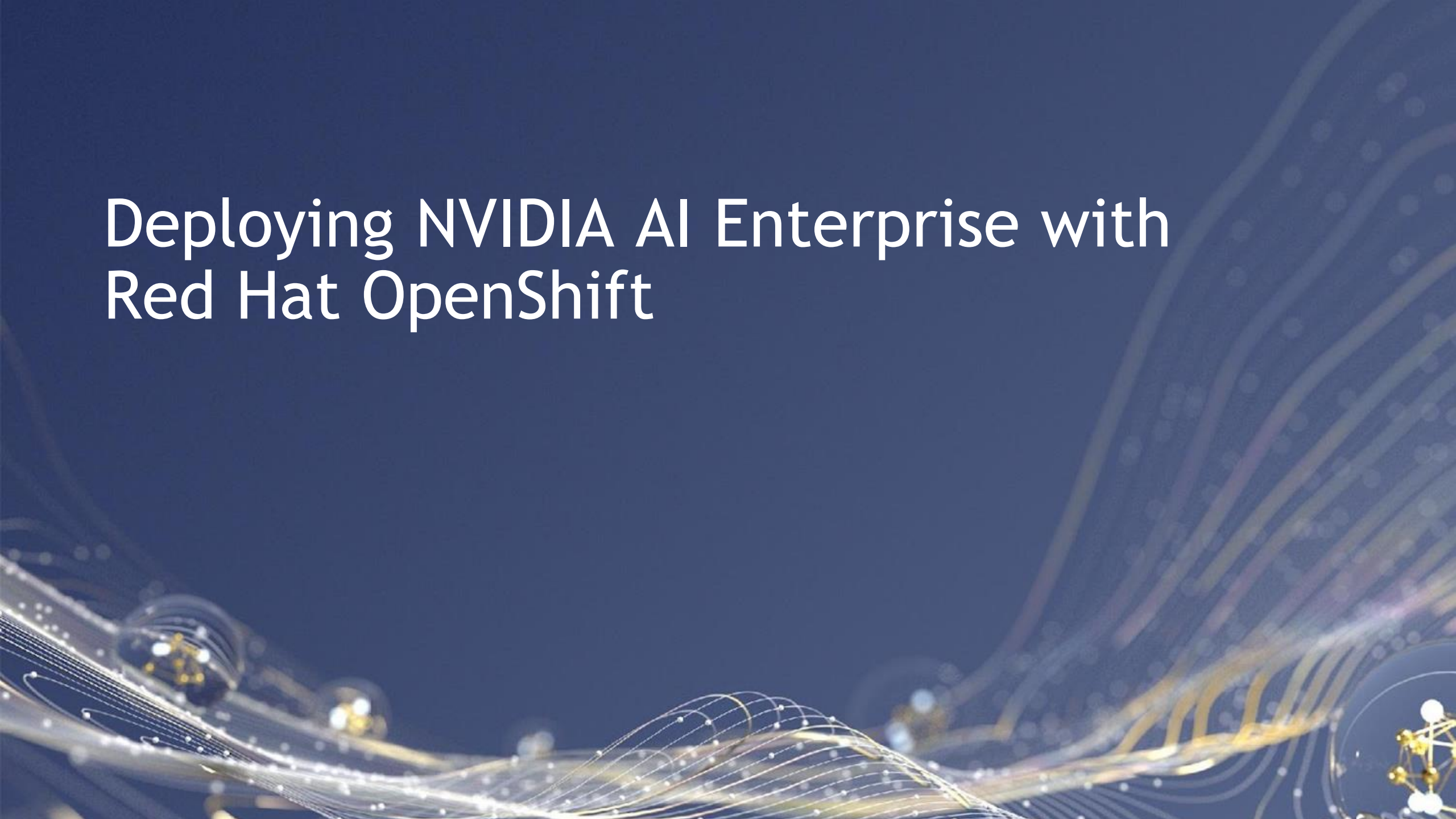


DELIVERING AI WORKLOADS WITH NVIDIA AI ENTERPRISE

Orchestration with Kubernetes and Virtualization

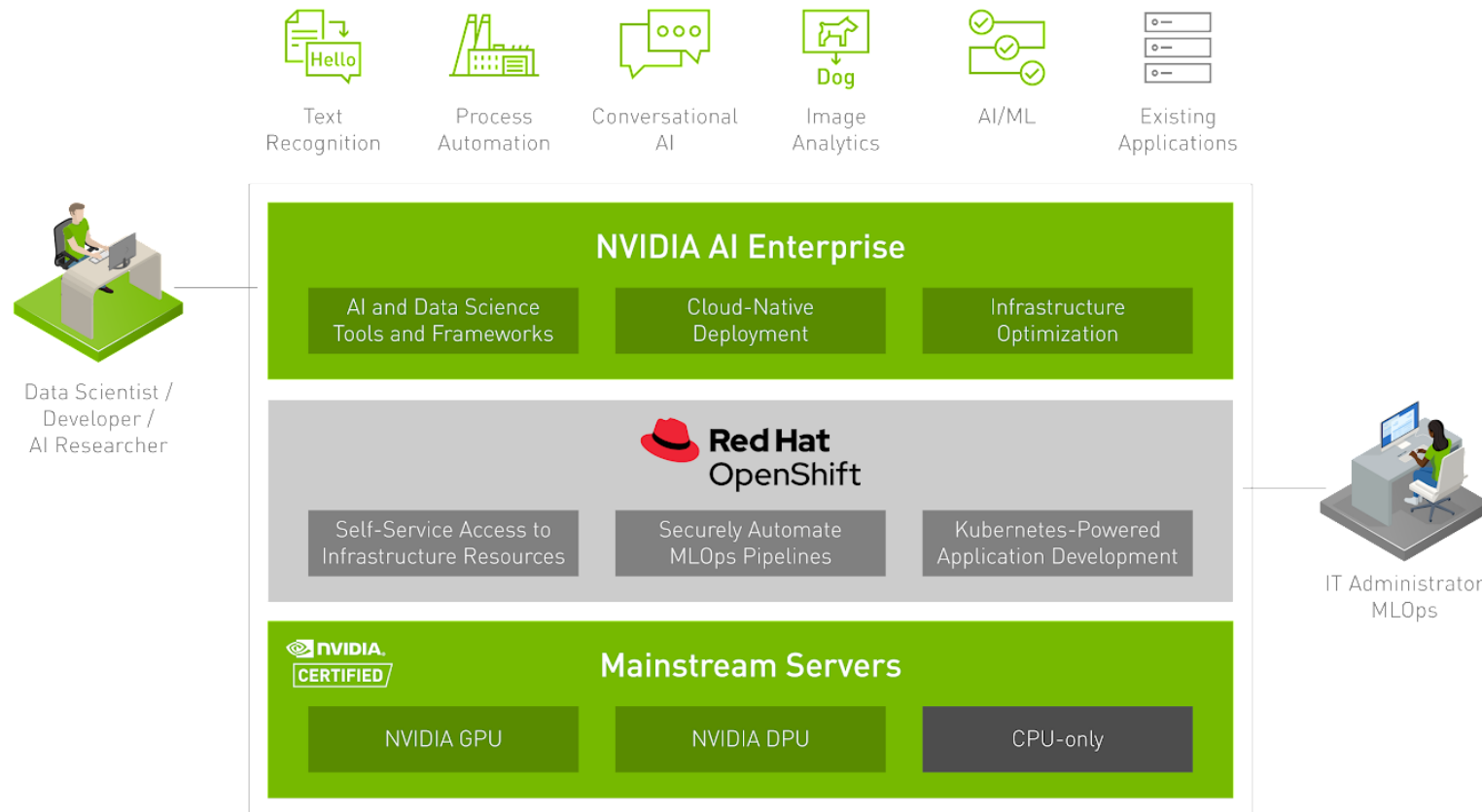


Deploying NVIDIA AI Enterprise with Red Hat OpenShift



NVIDIA AI ENTERPRISE WITH RED HAT OPENSIFT

Supported on Bare Metal or with VMware vSphere



AI ENTERPRISE WITH OPENSIFT

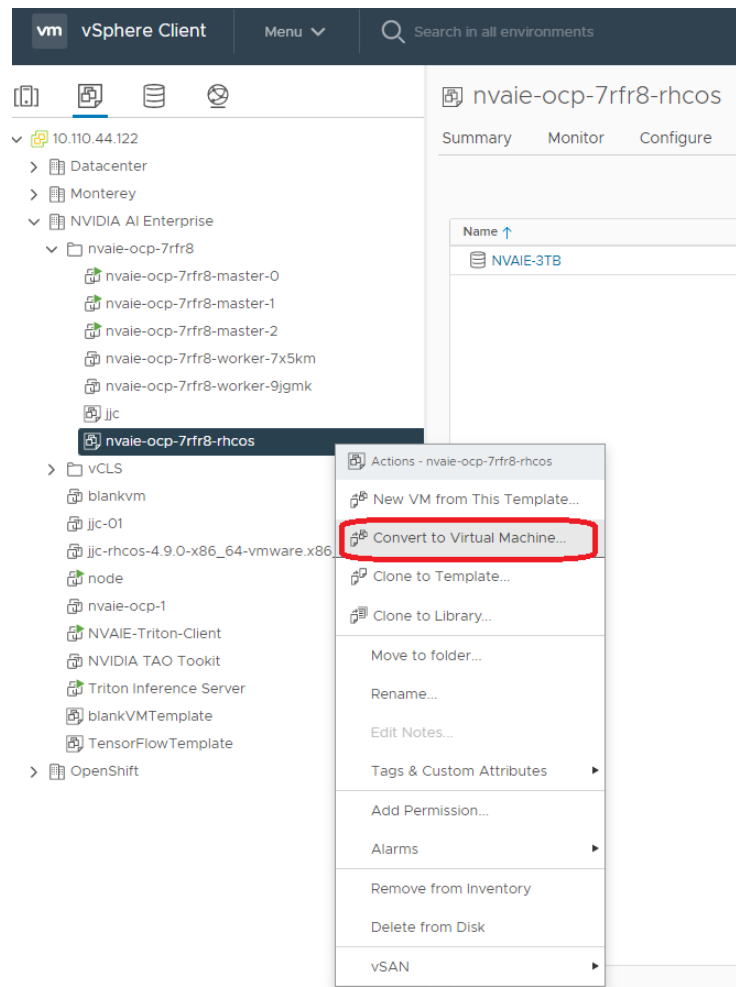
Requirements and Prerequisites for a Virtualized Deployment

- OpenShift CLI
 - Interact with cluster using oc
- NGC CLI
 - Pull/Push drivers, operators, containers, resources
- NVIDIA License
 - CLS or DLS instance
- NVIDIA AI Enterprise VIB on Hosts
 - Needed for virtualization
- Deployed OpenShift Cluster with **EFI Boot**
 - User Provisioned Infrastructure (UPI)
 - Installer Provisioned Infrastructure (IPI)

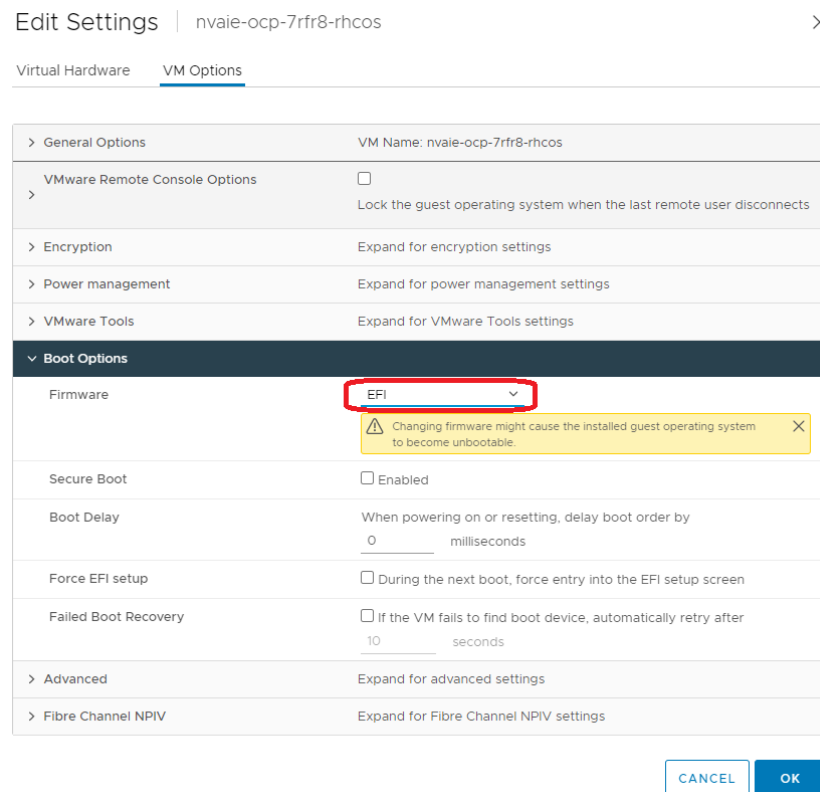
AI ENTERPRISE WITH OPENSIFT

EFI Boot with IPI (Installer Provisioned Infrastructure)

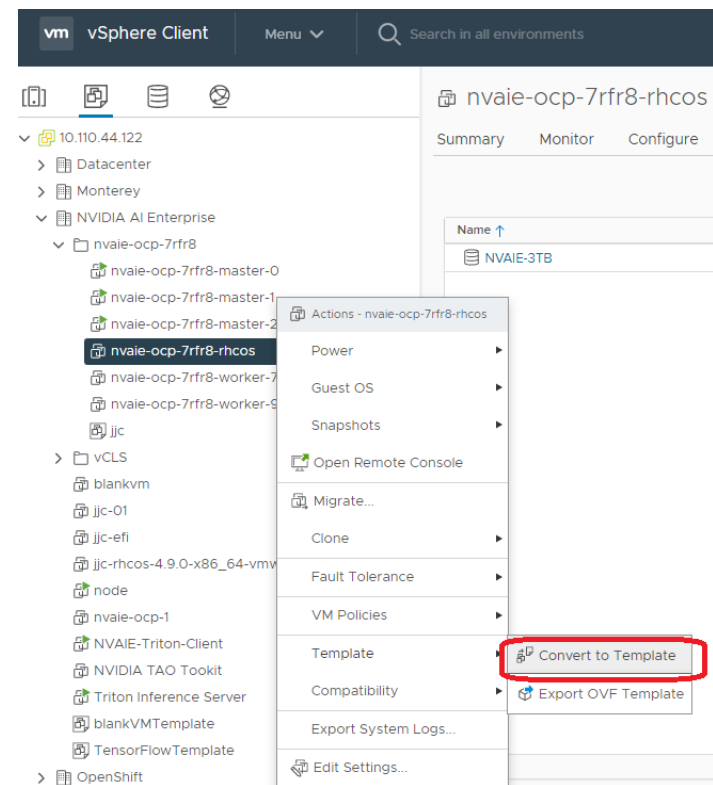
Convert to Virtual Machine



Change boot mode



Convert back to template



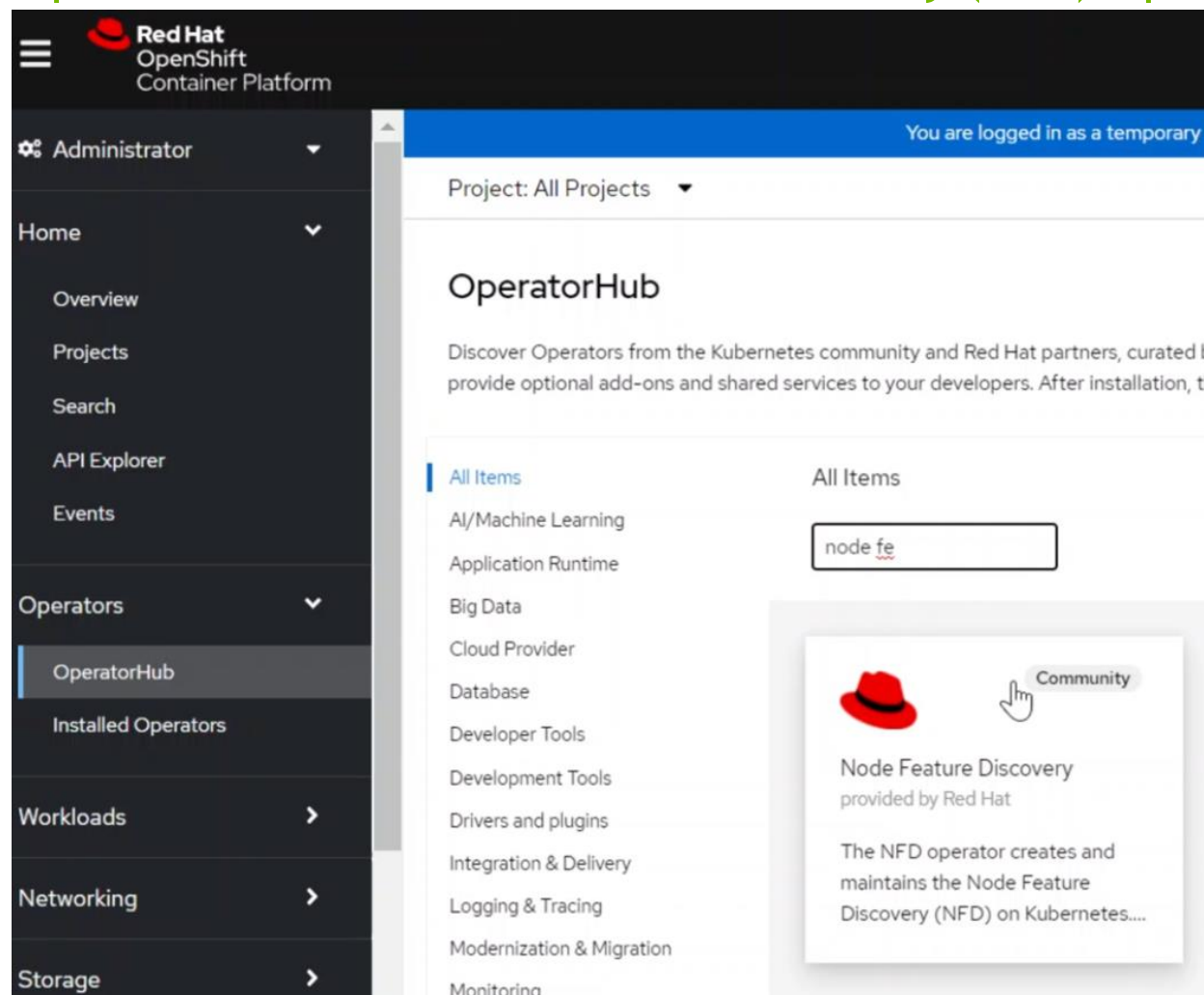
AI ENTERPRISE WITH OPENSIFT CONTAINER PLATFORM

Orchestration on OCP

- Step 1: Install the Node Feature Discovery (NFD) Operator
- Step 2: Create NLS License Config Map
- Step 3: Import NGC Secret
- Step 4: Install the NVIDIA Network Operator (optional)
- Step 5: Install the NVIDIA GPU Operator
- Step 6: Create the Cluster Policy Instance

AI ENTERPRISE WITH OPENSIFT

Step 1: Install the Node Feature Discovery (NFD) Operator



```
$ oc get pods -n openshift-nfd
```

NAME	READY	STATUS	RESTARTS	AGE
nfd-controller-manager-7f86ccfb58-nqgxm	2/2	Running	0	11m

AI ENTERPRISE WITH OPENSHIFT

Step 1: Install the Node Feature Discovery (NFD) Operator

Administrator ▾

Home ▾

- Overview
- Projects
- Search
- API Explorer
- Events

Operators ▾

- OperatorHub
- Installed Operators**

Workloads >

Networking >

Storage >

Project: openshift-nfd ▾

[Installed Operators](#) > Operator details

Node Feature Discovery
4.9.0 provided by Red Hat

[Details](#) [YAML](#) [Subscription](#) [Events](#)

Provided APIs

NFD NodeFeatureDiscovery

The NodeFeatureDiscovery instance is the CustomResource being watched by the NFD-Operator, and holds all the needed information to setup the behaviour of the master and worker pods

[Create instance](#)

Home ▾

- Overview
- Projects
- Search
- API Explorer
- Events

Operators ▾

- OperatorHub
- Installed Operators**

Workloads >

Installed Operators are represented by ClusterServiceVersions within this Namespace. For more information, see the [Understanding Operators documentation](#). Or create an Operator and ClusterServiceVersion using the

Name ▾ Search by name... [?](#)

Name	Namespace	Managed Namespaces	Status	Last updated	Provided APIs
Node Feature Discovery 4.9.0 provided by Red Hat	NS openshift-nfd	All Namespaces	✔ Succeeded Up to date	⌚ Feb 17, 2022, 12:57 PM	NodeFeatureDiscovery
Package Server 0.18.3 provided by Red Hat	NS openshift-operator-lifecycle-manager	NS openshift-operator-lifecycle-manager	✔ Succeeded	⌚ Feb 2, 2022, 11:52 PM	PackageManifest

Home ▾

- Overview
- Projects
- Search
- API Explorer
- Events

Operators ▾

- OperatorHub
- Installed Operators**

Workloads >

[Installed Operators](#) >

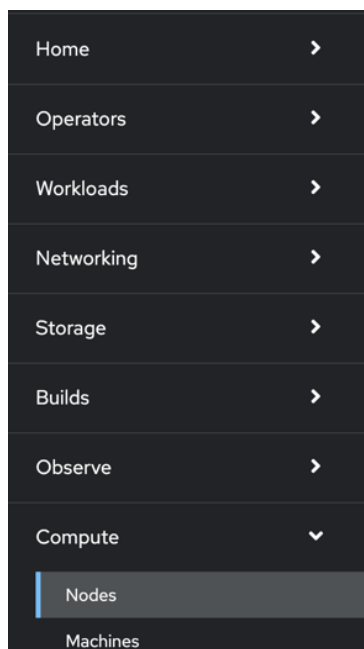
4.9.0 provided by Red Hat

[Details](#) [YAML](#) [Subscription](#) [Events](#) [NodeFeatureDiscovery](#)

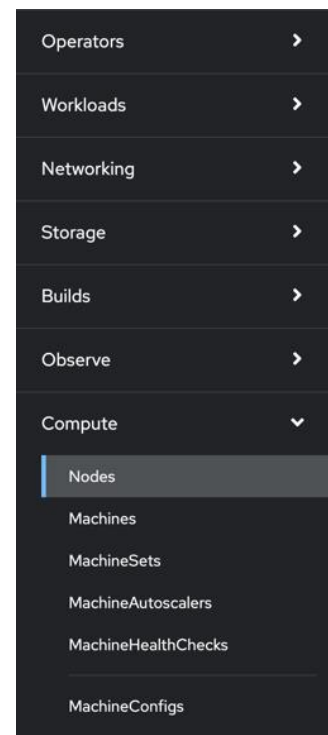
[Create NodeFeatureDiscovery](#)

AI ENTERPRISE WITH OPENSIFT

Step 1: Install the Node Feature Discovery (NFD) Operator



Filter	Name	Status	Role	Pods
	Search by name...			
	nvaie-ocp-7rfr8-master-0	Ready	master	35
	nvaie-ocp-7rfr8-master-1	Ready	master	60
	nvaie-ocp-7rfr8-master-2	Ready	master	35
	nvaie-ocp-7rfr8-worker-7x5km	Ready	worker	29
	nvaie-ocp-7rfr8-worker-9jgmk	Ready	worker	29
	nvaie-ocp-7rfr8-worker-jntsp	Ready	worker	15



AI ENTERPRISE WITH OPENSIFT

Step 1: Install the Node Feature Discovery (NFD) Operator

```
$ oc get nodes -l feature.node.kubernetes.io/pci-10de.present
NAME                                STATUS  ROLES  AGE  VERSION
nvaie-ocp-7rfr8-worker-7x5km      Ready  worker  20d  v1.22.3+e790d7f
nvaie-ocp-7rfr8-worker-jntsp      Ready  worker  11d  v1.22.3+e790d7f
```

AI ENTERPRISE WITH OPENSHIFT

Step 2: Create NLS License Config Map

Red Hat OpenShift Container Platform

You are logged in as a temporary administrative user. Update the [cluster OAuth configuration](#) to allow others to log in.

Projects

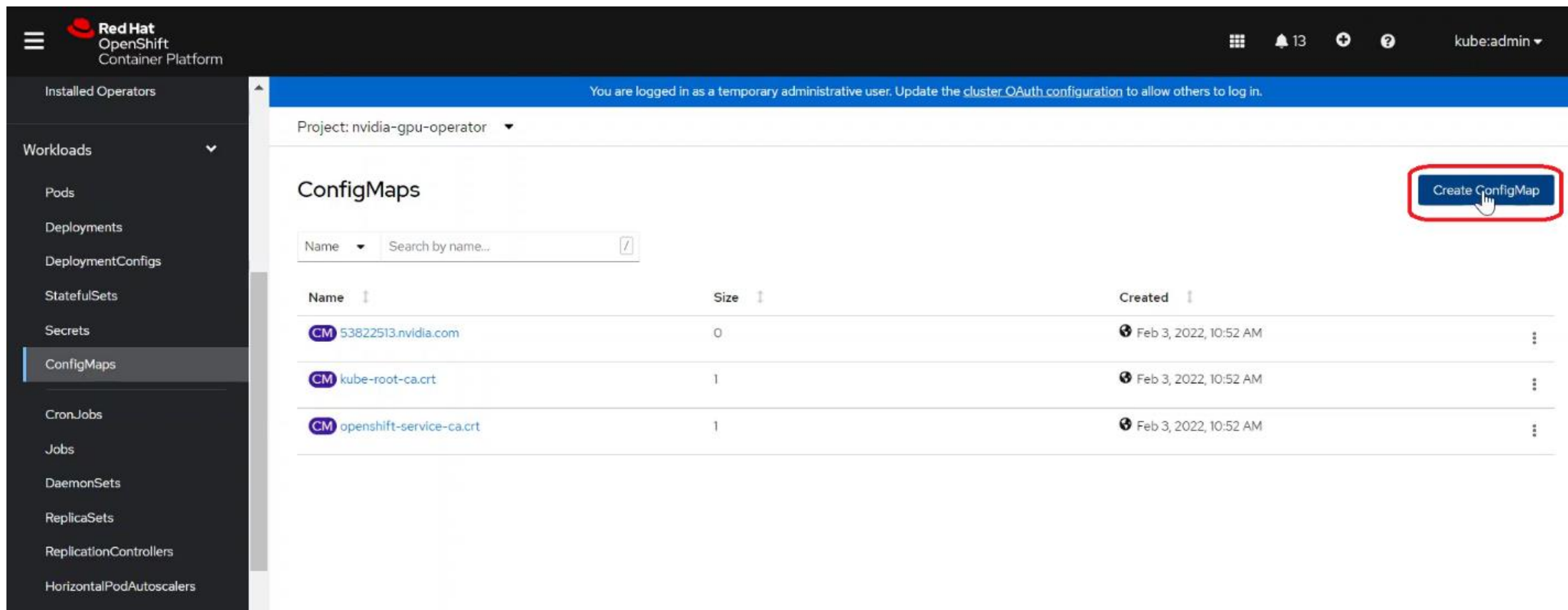
Filter Name Search by name...

Name	Display name	Status	Requester	Memory	CPU	Created
PR default	No display name	Active	No requester	2.1 MiB	0.001 cores	Jan 28, 2022, 11:04 AM
PR kube-node-lease	No display name	Active	No requester	-	-	Jan 28, 2022, 11:04 AM
PR kube-public	No display name	Active	No requester	-	-	Jan 28, 2022, 11:04 AM
PR kube-system	No display name	Active	No requester	-	-	Jan 28, 2022, 11:04 AM

Create Project

AI ENTERPRISE WITH OPENSIFT

Step 2: Create NLS License Config Map



The screenshot shows the Red Hat OpenShift Container Platform console interface. The left sidebar contains a navigation menu with the following items: Installed Operators, Workloads (expanded), Pods, Deployments, DeploymentConfigs, StatefulSets, Secrets, ConfigMaps (selected), CronJobs, Jobs, DaemonSets, ReplicaSets, ReplicationControllers, and HorizontalPodAutoscalers. The main content area displays the 'ConfigMaps' page for the 'Project: nvidia-gpu-operator'. A blue banner at the top of the main area states: 'You are logged in as a temporary administrative user. Update the [cluster OAuth configuration](#) to allow others to log in.' Below this, the 'Create ConfigMap' button is highlighted with a red rectangular box and a hand cursor. The ConfigMaps table lists three entries:

Name	Size	Created
CM 53822513.nvidia.com	0	Feb 3, 2022, 10:52 AM
CM kube-root-ca.crt	1	Feb 3, 2022, 10:52 AM
CM openshift-service-ca.crt	1	Feb 3, 2022, 10:52 AM

AI ENTERPRISE WITH OPENS SHIFT

Step 2: Create NLS License Config Map

The screenshot displays the Red Hat OpenShift Container Platform console interface. The left sidebar shows the navigation menu with categories like Projects, Operators, and Workloads. The 'ConfigMaps' option under the 'Workloads' section is selected. The main panel shows the 'licensing-config' ConfigMap details in the 'nvidia-gpu-operator' namespace. The 'YAML' tab is active, displaying the following configuration:

```
1 kind: ConfigMap
2 apiVersion: v1
3 metadata:
4   name: licensing-config
5   namespace: nvidia-gpu-operator
6   uid: 91aba1dd-ed1d-4544-884c-52f28231239b
7   resourceVersion: '29165345'
8   creationTimestamp: '2022-02-11T17:30:58Z'
9   managedFields: ...
20 data:
21   client_configuration_token.tok: >-
22     eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJ
23     gridd.conf: '# empty file'
```

At the bottom of the console, there are three buttons: 'Save', 'Reload', and 'Cancel'.

AI ENTERPRISE WITH OPENSSHIFT

Step 3: Import NGC Secret

The screenshot shows the Red Hat OpenShift Container Platform console. The left sidebar contains the following menu items: Events, Operators (with a dropdown arrow), OperatorHub, Installed Operators, Workloads (with a dropdown arrow), Pods, Deployments, DeploymentConfigs, StatefulSets, Secrets (highlighted with a red box), and ConfigMaps. The main content area is titled 'Secrets' and shows a table of secrets. The table has columns for Name, Type, Size, and Created. The secrets listed are: builder-dockercfg-n5gkf, builder-token-5p9kv, builder-token-kbrbj, default-dockercfg-4bg4f, and default-token-cblj6. The 'Create' button in the top right corner is highlighted with a red box, and its dropdown menu is open, showing options: Key/value secret, Image pull secret (highlighted with a red box), Source secret, Webhook secret, and From YAML.

Project: nvidia-gpu-operator

You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Secrets

Filter Name Search by name...

Name	Type	Size	Created
builder-dockercfg-n5gkf	kubernetes.io/dockercfg	1	10 minutes ago
builder-token-5p9kv	kubernetes.io/service-account-token	4	10 minutes ago
builder-token-kbrbj	kubernetes.io/service-account-token	4	10 minutes ago
default-dockercfg-4bg4f	kubernetes.io/dockercfg	1	10 minutes ago
default-token-cblj6	kubernetes.io/service-account-token	4	10 minutes ago

Create

- Key/value secret
- Image pull secret
- Source secret
- Webhook secret
- From YAML

AI ENTERPRISE WITH OPENSIFT

Step 3: Import NGC Secret

Secret name: `gpu-operator-secret`

Authentication type: `Image registry credentials`

Registry server address: `nvcr.io/nvaie`

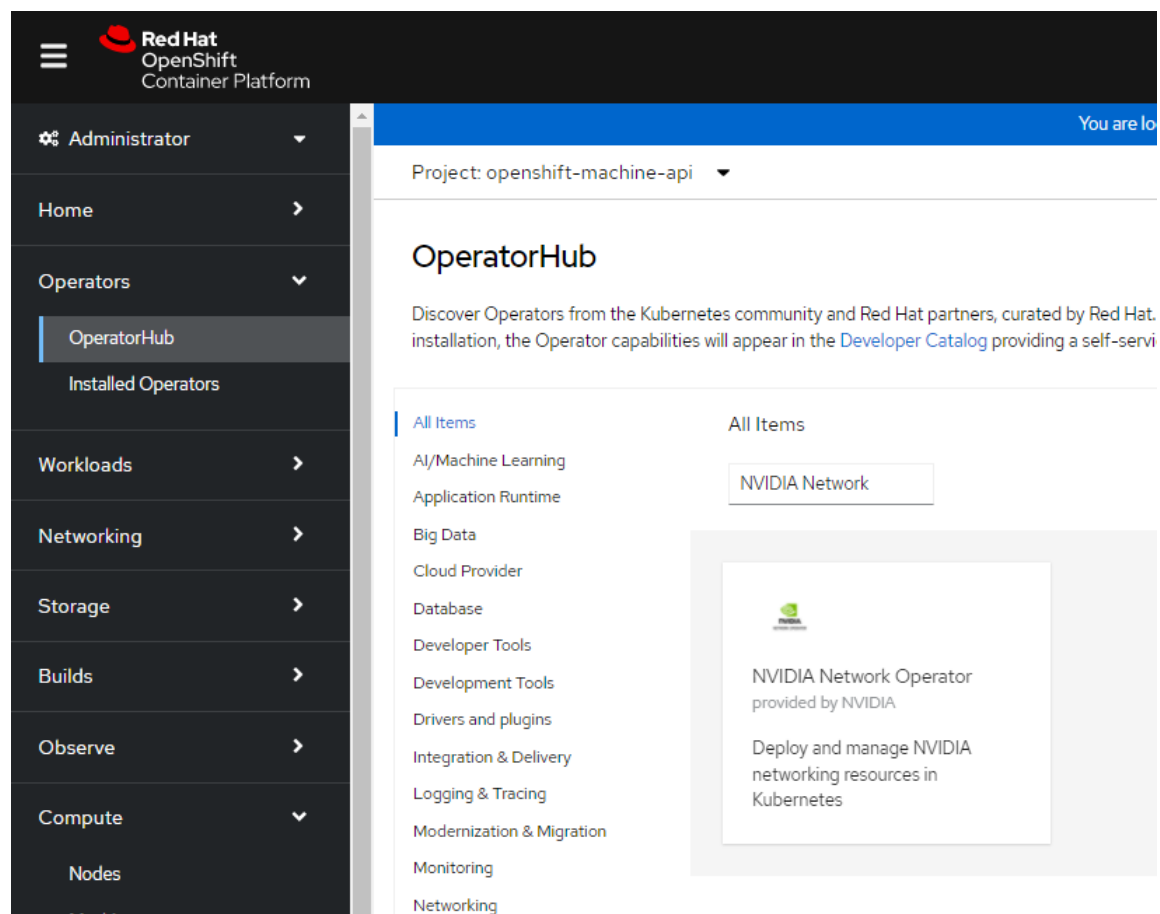
Username: `$oauthtoken`

Password: `<API-KEY>`

Email: `<YOUR-EMAIL>`


AI ENTERPRISE WITH OPENSIFT

Step 4: Install the NVIDIA Network Operator (Optional)



AI ENTERPRISE WITH OPENSIFT

Step 4: Install the NVIDIA Network Operator (Optional)

 **NVIDIA Network Operator** 1.1.0 provided by NVIDIA ✕

[Install](#)

Latest version
1.1.0

Capability level
☒ Basic Install
☐ Seamless Upgrades
☐ Full Lifecycle
☐ Deep Insights
☐ Auto Pilot

Source
Certified

Provider
NVIDIA

Repository
<https://github.com/Mellanox/network-operator/>

Container image
N/A

Created at
N/A

Support
NVIDIA

NVIDIA Network Operator

The NVIDIA Network Operator simplifies the provisioning and management of NVIDIA networking resources in a Kubernetes cluster. The operator automatically installs the required host networking software - bringing together all the needed components to provide high-speed network connectivity. These components include the NVIDIA networking driver, Kubernetes device plugin, CNI plugins, IP address management (IPAM) plugin and others. The NVIDIA Network Operator works in conjunction with the NVIDIA GPU Operator to deliver high-throughput, low-latency networking for scale-out, GPU computing clusters. The Network Operator is open-source. For more information on contributions and release artifacts, see the [GitHub repo](#)

AI ENTERPRISE WITH OPENSSHIFT

Step 4: Install the NVIDIA Network Operator (Optional)

The screenshot displays the Red Hat OpenShift Container Platform OperatorHub interface. The left sidebar contains navigation links: Administrator, Home, Operators (with a sub-link for OperatorHub), Installed Operators, Workloads, Networking, Storage, Builds, Observe, and Compute (with sub-links for Nodes, Machines, MachineSets, MachineAutoscalers, and MachineHealthChecks). The main content area is titled 'Install Operator' and includes a breadcrumb 'OperatorHub > Operator Installation'. A blue banner at the top of the main area states: 'You are logged in as a temporary administrative user. Update the [cluster OAuth configuration](#) to allow others to log in.' Below the title, a paragraph explains that the operator is installed by subscribing to an update channel. The configuration section includes: 'Update channel *' with a radio button for '1.1.0'; 'Installation mode *' with two radio buttons, 'All namespaces on the cluster (default)' (selected) and 'A specific namespace on the cluster'; 'Installed Namespace *' with a dropdown menu showing 'PR openshift-operators'; and 'Update approval *' with radio buttons for 'Automatic' (selected) and 'Manual'. At the bottom left are 'Install' and 'Cancel' buttons. On the right, the 'NVIDIA Network Operator' is listed as 'provided by NVIDIA'. Below this, a section titled 'Provided APIs' shows four API types in a 2x2 grid: 'NAD NetworkAttachmentDefinition', 'HDN HostDeviceNetwork', 'MN MacvlanNetwork', and 'NCP NicClusterPolicy'. Each API type is marked as 'Not available'.

Red Hat OpenShift Container Platform

You are logged in as a temporary administrative user. Update the [cluster OAuth configuration](#) to allow others to log in.

OperatorHub > Operator Installation

Install Operator

Install your Operator by subscribing to one of the update channels to keep the Operator up to date. The strategy determines either manual or automatic updates.

Update channel *

☒ 1.1.0

Installation mode *

☒ All namespaces on the cluster (default)
Operator will be available in all Namespaces.

☐ A specific namespace on the cluster
Operator will be available in a single Namespace only.

Installed Namespace *

PR openshift-operators

Update approval *

☒ Automatic

☐ Manual

[Install](#) [Cancel](#)

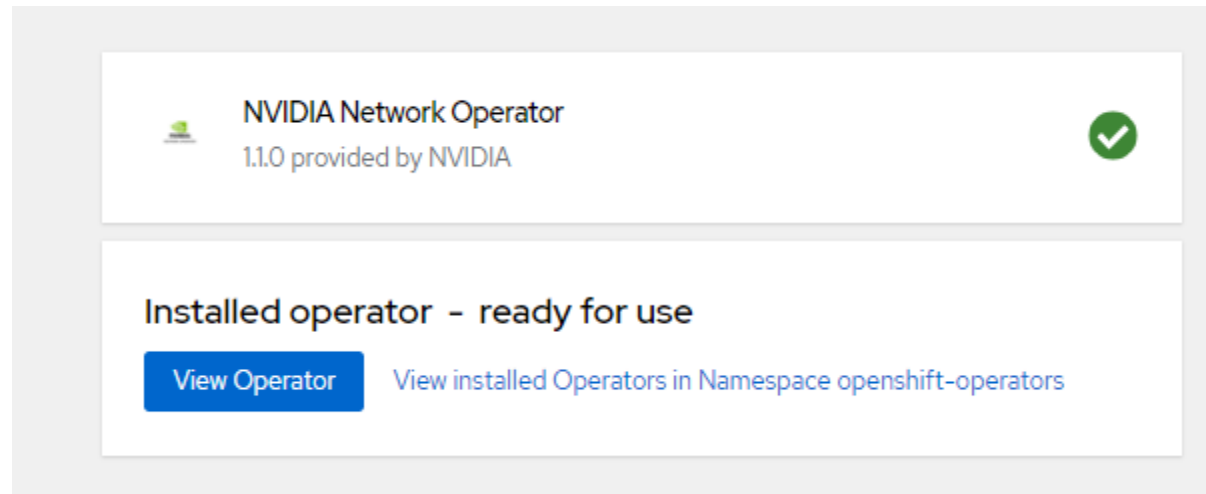
NVIDIA Network Operator
provided by NVIDIA

Provided APIs

NAD NetworkAttachmentDefinition Not available	HDN HostDeviceNetwork Not available
MN MacvlanNetwork Not available	NCP NicClusterPolicy Not available

AI ENTERPRISE WITH OPENSIFT

Step 4: Install the NVIDIA Network Operator (Optional)



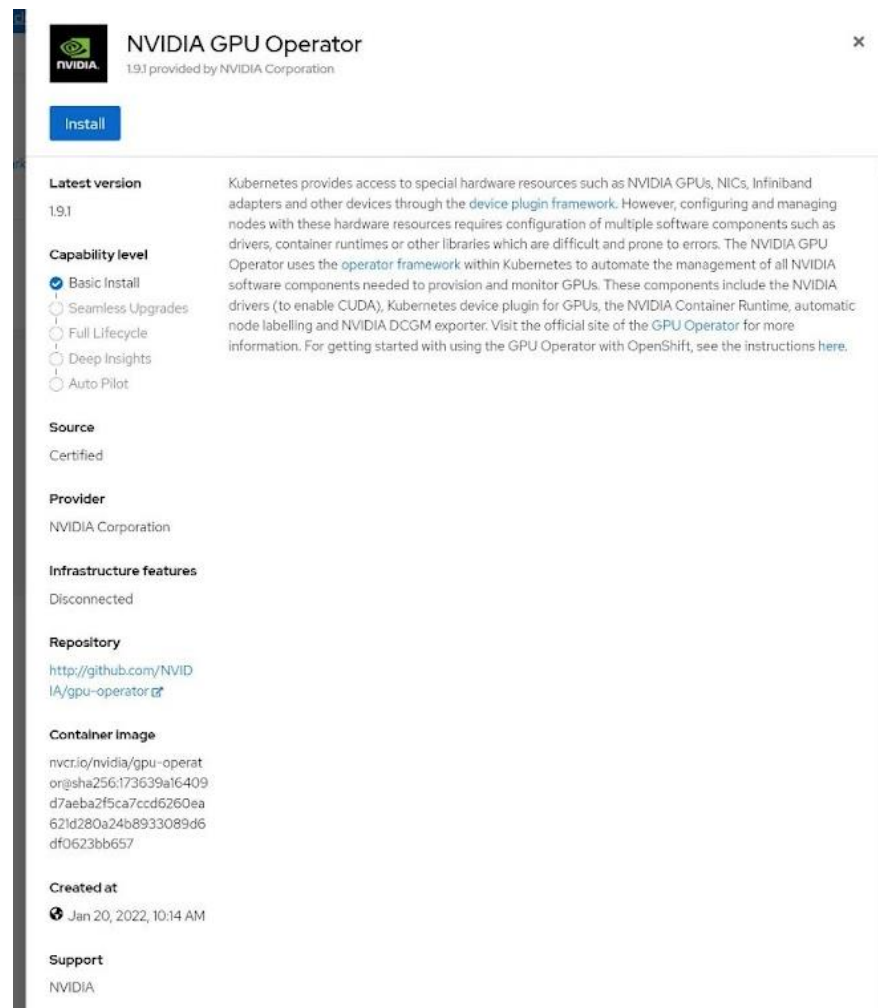
AI ENTERPRISE WITH OPENSIFT

Step 5: Install GPU Operator

The screenshot displays the OperatorHub web interface. On the left is a dark sidebar with navigation links: Home (with a dropdown arrow), Overview, Projects, Search, API Explorer, Events, Operators (with a dropdown arrow), OperatorHub (highlighted with a blue bar), Installed Operators, Workloads (with a right arrow), and Networking (with a right arrow). The main content area is titled 'OperatorHub' and includes a descriptive paragraph: 'Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat and shared services to your developers. After installation, the Operator capabilities will appear in the console.' Below this is a list of categories under the heading 'All Items': AI/Machine Learning, Application Runtime, Big Data, Cloud Provider, Database, Developer Tools, Development Tools, Drivers and plugins, Integration & Delivery, and Logging & Tracing. A search bar is present, containing the text 'GPU Operator', which is highlighted with a green border and a red rectangle. Below the search bar, a card for the 'NVIDIA GPU Operator' is displayed, featuring the NVIDIA logo, the title 'NVIDIA GPU Operator', the subtitle 'provided by NVIDIA Corporation', and the description 'Automate the management and monitoring of NVIDIA GPUs.'

AI ENTERPRISE WITH OPENSIFT

Step 5: Install GPU Operator



AI ENTERPRISE WITH OPENSIFT

Step 5: Install GPU Operator

You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

OperatorHub > Operator Installation

Install Operator

Install your Operator by subscribing to one of the update channels to keep the Operator up to date. The strategy determines either manual or automatic updates.

Update channel *

- ☐ beta
- ☐ stable
- ☐ v1.7
- ☐ v1.8
- ☒ v1.9.0

Installation mode *

- ☐ All namespaces on the cluster (default)
This mode is not supported by this Operator
- ☒ A specific namespace on the cluster
Operator will be available in a single Namespace only.

Installed Namespace *

- ☒ Operator recommended Namespace: **nvidia-gpu-operator**


Namespace creation
Namespace **nvidia-gpu-operator** does not exist and will be created.


☐ Select a Namespace

Update approval *

- ☒ Automatic
- ☐ Manual

Install **Cancel**

**NVIDIA GPU Operator**
provided by NVIDIA Corporation
Provided APIs

**ClusterPolicy**
ClusterPolicy allows you to configure the GPU Operator

```
$ oc label ns/$NAMESPACE_NAME openshift.io/cluster-monitoring=true
```

AI ENTERPRISE WITH OPENSIFT

Step 6: Create the Cluster Policy Instance

The screenshot displays the Red Hat OpenShift Container Platform console interface. The left sidebar contains navigation links: Administrator, Home, Operators (with sub-links for OperatorHub and Installed Operators), Workloads (with sub-links for Pods, Deployments, DeploymentConfigs, StatefulSets, Secrets, ConfigMaps, CronJobs, Jobs, DaemonSets, ReplicaSets, ReplicationControllers, and HorizontalPodAutoscalers), and Networking. The main content area shows the configuration for the 'nvidia-gpu-operator' project. It includes sections for 'Device Plugin config', 'Driver config', 'ImagePullPolicy' (with radio buttons for Always, Never, and IfNotPresent), 'Licensing Config' (with a 'Config Map Name' field set to 'licensing-config'), 'Nls Enabled' (with a checked checkbox for 'nlsEnabled'), 'Enabled' (with a checked checkbox for 'enabled'), 'Use Ocp Driver Toolkit' (with a checked checkbox for 'use_ocp_driver_toolkit'), and 'Cert Config'. A blue banner at the top of the main area states: 'You are logged in as a temporary administrative user. Update the [cluster OAuth configuration](#) to allow others to log in.'

Activate Windows
Go to Settings to activate Windows.

AI ENTERPRISE WITH OPENS SHIFT

Step 6: Create the Cluster Policy Instance

The screenshot shows the Red Hat OpenShift Container Platform console. The left sidebar contains navigation links: Administrator, Home, Operators (selected), Workloads, Pods, Deployments, DeploymentConfigs, StatefulSets, Secrets, ConfigMaps, CronJobs, Jobs, DaemonSets, ReplicaSets, ReplicationControllers, HorizontalPodAutoscalers, Networking, and Storage. The main content area is titled 'Project: nvidia-gpu-operator' and displays a configuration form for the 'nvidia-gpu-operator' project. The form includes sections for Repository, Env, Manager, Security Context, Repo Config, Version, Virtual Topology, Image, and Args. The 'Repository' section has a text input for '<private-repository-path>' and a label 'Driver image repository'. The 'Env' section has a label 'Optional: List of environment variables'. The 'Manager' section has a label 'Manager represents configuration for driver manager initContainer'. The 'Security Context' section has a label 'Optional: Security Context'. The 'Repo Config' section has a label 'Optional: Custom repo configuration for driver container'. The 'Version' section has a text input for '470.82.01' and a label 'Driver image tag'. The 'Virtual Topology' section has a label 'Optional: Virtual Topology Daemon configuration for vGPU drivers'. The 'Image' section has a text input for 'driver' and a label 'Driver image name'. The 'Args' section has a label 'Optional: List of arguments'. At the bottom, there is a section for 'Advanced configuration' with a dropdown menu for 'Image pull secrets'.

```
nlsEnabled: true
repository: nvcr.io/nvaie
version: 510.47.03
image: vgpu-guest-driver
```

Advanced configuration menu and specify the `imagePullSecret`. (eg: `gpu-operator-secret`)



AI ENTERPRISE WITH OPENSIFT

Step 6: Create the Cluster Policy Instance

Project: nvidia-gpu-operator ▼

Installed Operators

Installed Operators are represented by ClusterServiceVersions within this Namespace. For more information, see the [Understanding Operators documentation](#). Or create an Operator and ClusterServiceVersion using the [Operator SDK](#).

Name ▼	Search by name...	
Name ↑	Managed Namespaces ↑	Status
 NVIDIA GPU Operator 1.9.0 provided by NVIDIA Corporation	 nvidia-gpu-operator	✓ Succeeded Up to date
		Last updated
		1 minute ago
		Provided APIs
		ClusterPolicy

```
$ oc get nodes -o=custom-  
columns='Node:metadata.name,GPU:status.capacity.nvidia\.com/gpu'  
Node                                GPU  
nvaie-ocp-7rfr8-master-0           <none>  
nvaie-ocp-7rfr8-master-1           <none>  
nvaie-ocp-7rfr8-master-2           <none>  
nvaie-ocp-7rfr8-worker-7x5km       1  
nvaie-ocp-7rfr8-worker-9jgmk       <none>  
nvaie-ocp-7rfr8-worker-jntsp       1  
nvaie-ocp-7rfr8-worker-zkggt       <none>
```

Orchestration with OCP and vSphere



AI ENTERPRISE WITH OPENSIFT AND VSPHERE

Scaling OpenShift with MachineSets

The screenshot displays the vSphere Web Client interface for configuring a virtual machine (VM) named `nvaie-ocp-7rfr8-worker-jntsp`. The left sidebar shows a tree view with the following structure:

- 10.110.44.122
 - Datacenter
 - Monterey
 - NVIDIA AI Enterprise
 - OpenShift
 - 10.136.145.49
 - 10.136.145.50
 - nvaie-ocp-7rfr8-master-0
 - nvaie-ocp-7rfr8-master-1
 - nvaie-ocp-7rfr8-master-2
 - nvaie-ocp-7rfr8-rhcos
 - nvaie-ocp-7rfr8-worker-jntsp**
 - nvaie-ocp-7rfr8-worker-rlrxh

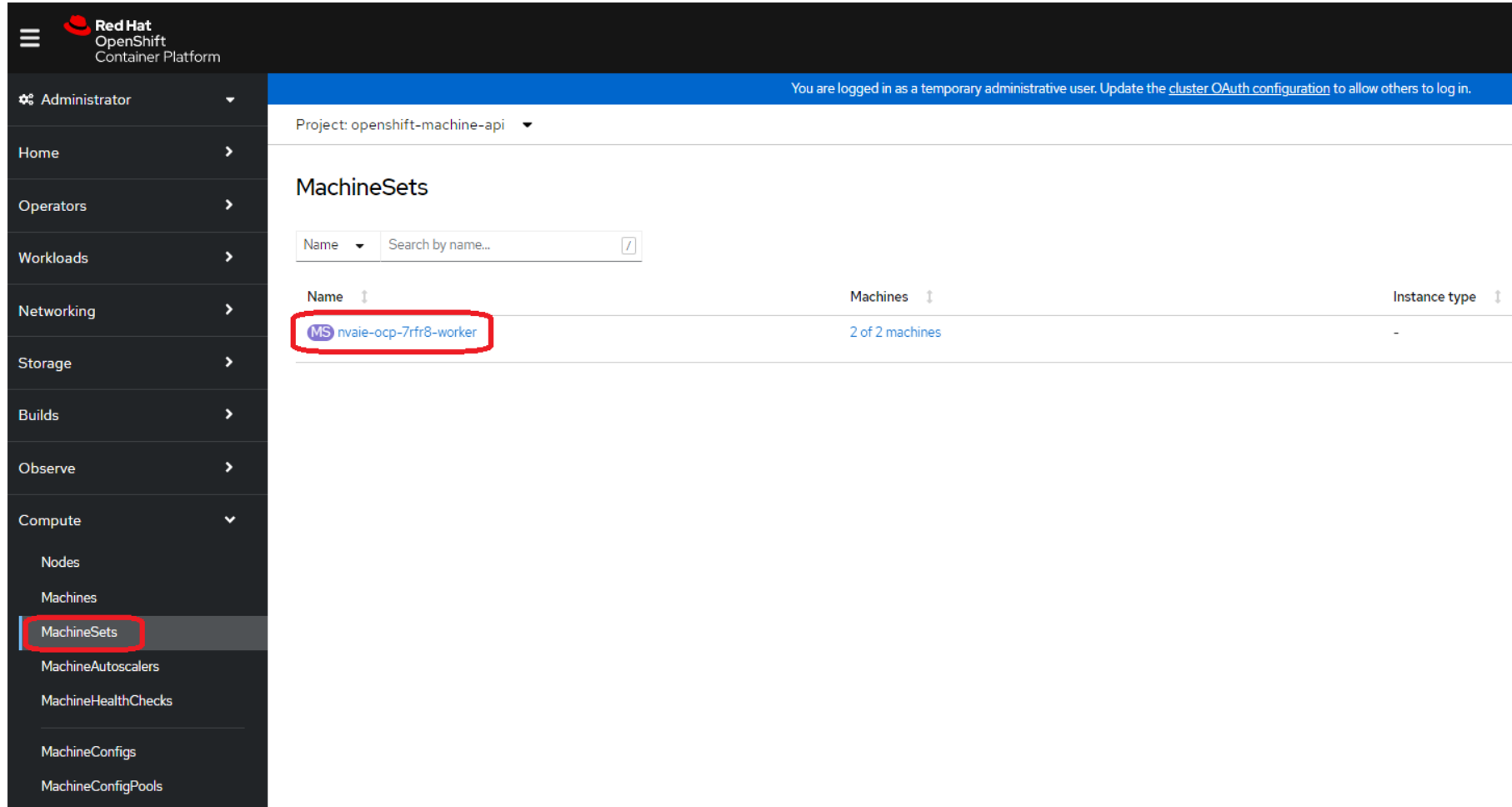
The main panel shows the 'Summary' tab for the selected VM. The 'VM Hardware' section is expanded, displaying the following configuration:

Component	Configuration
CPU	32 CPU(s)
Memory	96 GB, 72 GB memory active
Hard disk 1	160 GB
Network adapter 1	VM Network (connected)
Video card	8 MB
PCI device 0	NVIDIA GRID vGPU nvidia_a30-24c
VMCI device	Device on the virtual machine PCI bus that provides support for the virtual machine communication interface
Other	Additional Hardware
Compatibility	ESXi 6.7 U2 and later (VM version 15)

The 'PCI device 0' row is highlighted with a red box, indicating the NVIDIA GRID vGPU configuration. An 'Edit Settings...' link is visible at the bottom of the hardware configuration section.

AI ENTERPRISE WITH OPENSIFT AND VSPHERE

Scaling OpenShift with MachineSets



The screenshot displays the Red Hat OpenShift Container Platform console interface. The left sidebar contains a navigation menu with the following items: Administrator, Home, Operators, Workloads, Networking, Storage, Builds, Observe, Compute (expanded), Nodes, Machines, MachineSets (highlighted with a red box), MachineAutoscalers, MachineHealthChecks, MachineConfigs, and MachineConfigPools. The main content area shows the 'MachineSets' page for the 'openshift-machine-api' project. A search bar is present with the text 'Search by name...'. Below the search bar, a table lists the MachineSets. The first entry is 'MS nvaie-ocp-7rfr8-worker', which is highlighted with a red box. The table has columns for 'Name', 'Machines', and 'Instance type'. The 'Machines' column for the highlighted entry shows '2 of 2 machines'.

Name	Machines	Instance type
MS nvaie-ocp-7rfr8-worker	2 of 2 machines	-

AI ENTERPRISE WITH OPENS SHIFT AND VSPHERE

Scaling OpenShift with MachineSets

The screenshot displays the Red Hat OpenShift Container Platform console interface. On the left is a dark sidebar with navigation links: Administrator, Home, Operators, Workloads, Networking, Storage, Builds, Observe, Compute (expanded), Nodes, Machines, MachineSets (selected), MachineAutoscalers, MachineHealthChecks, MachineConfigs, and MachineConfigPools. The main content area shows the 'MachineSet details' for 'nvaie-ocp-7rfr8-worker' in the 'openshift-machine-api' namespace. A table at the top shows counts: Desired count (2 machines, highlighted with a red box and an edit icon), Current count (2 machines), Ready count (2 machines), and Available count (2 machines). Below the table, the 'Labels' section shows 'machine.openshift.io/cluster-api-cluster=nvaie-ocp-7rfr8'. The 'Annotations' section shows '2 annotations'. The 'Selector' section shows the selector 'machine.openshift.io/cluster-api-cluster=nvaie-ocp-7rfr8, machine.openshift.io/cluster-api-machineset=nvaie-ocp-7rfr8-worker'.

Red Hat OpenShift Container Platform

You are logged in as a temporary administrative user. Update

Project: openshift-machine-api

MachineSet > MachineSet details

MS nvaie-ocp-7rfr8-worker

Details YAML Machines Events

MachineSet details

Desired count	Current count	Ready count	Available count
2 machines	2 machines	2 machines	2 machines

Name
nvaie-ocp-7rfr8-worker

Namespace
NS openshift-machine-api

Labels

machine.openshift.io/cluster-api-cluster=nvaie-ocp-7rfr8

Annotations
2 annotations

Selector
machine.openshift.io/cluster-api-cluster=nvaie-ocp-7rfr8, machine.openshift.io/cluster-api-machineset=nvaie-ocp-7rfr8-worker

AI ENTERPRISE WITH OPENSIFT AND VSPHERE

Scaling OpenShift with MachineSets

The screenshot shows the Red Hat OpenShift Container Platform console. The left sidebar contains navigation links: Administrator, Home, Operators, Workloads, Networking, Storage, Builds, Observe, Compute, Nodes, Machines, MachineSets (selected), MachineAutoscalers, MachineHealthChecks, MachineConfigs, and MachineConfigPools. The main content area shows the MachineSet details for 'nvaie-ocp-7rfr8-worker' in the 'openshift-machine-api' namespace. The 'Desired count' is highlighted with a red box and shows '2 machines' with an edit icon. Below this, there are fields for Name, Namespace, Labels, Annotations, and Selector.

Desired count	Current count	Ready count	Available count
2 machines	2 machines	2 machines	2 machines

MachineSet details

Name
nvaie-ocp-7rfr8-worker

Namespace
 openshift-machine-api

Labels

machine.openshift.io/cluster-api-cluster=nvaie-ocp-7rfr8

Annotations
2 annotations

Selector
 machine.openshift.io/cluster-api-cluster=nvaie-ocp-7rfr8, machine.openshift.io/cluster-api-machineset=nvaie-ocp-7rfr8-worker

Edit Machine count

MachineSets maintain the proper number of healthy machines.

Cancel

Save

AI ENTERPRISE WITH OPENSIFT AND VSPHERE

Scaling OpenShift with MachineSets

The screenshot shows the Red Hat OpenShift Container Platform console. The left sidebar contains navigation links: Administrator, Home, Operators, Workloads, Networking, Storage, Builds, Observe, Compute, Nodes, Machines, MachineSets (selected), MachineAutoscalers, MachineHealthChecks, MachineConfigs, and MachineConfigPools. The main content area displays the details for the MachineSet 'nvaie-ocp-7rfr8-worker'. A table shows the following counts:

Desired count	Current count	Ready count	Available count
2 machines	2 machines	2 machines	2 machines

The 'Desired count' cell is highlighted with a red rectangle. Below the table, the Name is 'nvaie-ocp-7rfr8-worker', the Namespace is 'openshift-machine-api', and the Labels are 'machine.openshift.io/cluster-api-cluster=nvaie-ocp-7rfr8'. There are 2 annotations and a selector 'machine.openshift.io/cluster-api-cluster=nvaie-ocp-7rfr8, machine.openshift.io/cluster-api-machineset=nvaie-ocp-7rfr8-worker'.

Edit Machine count

MachineSets maintain the proper number of healthy machines.

Cancel

Save

AI ENTERPRISE WITH OPENSIFT AND VSPHERE

Scaling OpenShift with MachineSets

The screenshot displays the Red Hat OpenShift Container Platform console. The left sidebar shows the navigation menu with 'MachineSets' selected under the 'Compute' section. The main content area shows the 'MachineSet details' for 'nvaie-ocp-7rfr8-worker'. A red box highlights the 'MachineSet details' section, which includes a table with the following data:

Desired count	Current count	Ready count	Available count
3 machines	3 machines	3 machines	3 machines

Below the table, the 'Name' is 'nvaie-ocp-7rfr8-worker' and the 'Namespace' is 'openshift-machine-api'. The 'Labels' section shows 'machine.openshift.io/cluster-api-cluster=nvaie-ocp-7rfr8'. The 'Annotations' section shows '2 annotations '. The 'Selector' section shows 'machine.openshift.io/cluster-api-cluster=nvaie-ocp-7rfr8, machine.openshift.io/cluster-api-machineset=nvaie-ocp-7rfr8-worker'.

AI ENTERPRISE WITH OPENSIFT AND VSPHERE

Scaling OpenShift with MachineSets

The screenshot displays the vSphere Web Client interface. On the left, a navigation pane shows a tree structure: 10.110.44.122 > Datacenter > Monterey > NVIDIA AI Enterprise > OpenShift. The VM 'nvaie-ocp-7rfr8-worker-z72rt' is selected and highlighted. The main panel shows the 'Summary' tab for this VM. At the top of the main panel, a red box highlights the 'Edit' icon (a square with a pencil). Below the tabs, the 'VM Hardware' section is expanded, showing a table of hardware specifications. At the bottom of this section, a red box highlights the 'Edit Settings...' link.

VM Hardware	
> CPU	32 CPU(s)
> Memory	96 GB, 72 GB memory active
> Hard disk 1	160 GB
> Network adapter 1	VM Network (connected)
> Video card	8 MB
VMCI device	Device on the virtual machine PCI bus that provides support for the virtual machine communication interface
> Other	Additional Hardware
Compatibility	ESXi 6.7 U2 and later (VM version 15)

[Edit Settings...](#)

AI ENTERPRISE WITH OPENSIFT AND VSPHERE

Scaling OpenShift with MachineSets

Edit Settings | nvaie-ocp-7rfr8-worker-z72rt ×

Virtual Hardware VM Options

[ADD NEW DEVICE ▾](#)

> CPU	32 ▾
> Memory	96 ▾ GB ▾
> Hard disk 1	160 GB ▾
> SCSI controller 0	VMware Paravirtual
> Network adapter 1	VM Network ▾
> Video card	Specify custom settings ▾
VMCI device	
> Other	Additional Hardware

Disks, Drives and Storage

- Hard Disk
- Existing Hard Disk
- RDM Disk
- Host USB Device
- NVDIMM
- CD/DVD Drive

Controllers

- NVMe Controller
- SATA Controller
- SCSI Controller
- USB Controller

Other Devices

- PCI Device**
- Serial Port

Network

- Network Adapter

CANCEL OK

AI ENTERPRISE WITH OPENSIFT AND VSPHERE

Scaling OpenShift with MachineSets

Edit Settings | nvaie-ocp-7rfr8-worker-z72rt ×

Virtual Hardware VM Options ADD NEW DEVICE ▾

> CPU	32 ▾	ⓘ
> Memory	96 ▾ GB ▾	
> Hard disk 1	160 GB ▾	
> SCSI controller 0	VMware Paravirtual	
> Network adapter 1	VM Network ▾	<input checked="" type="checkbox"/> Connected
▼ PCI device 0	NVIDIA GRID vGPU nvidia_a30-24c	
NVIDIA GRID vGPU Profile	<div>nvidia_a30-24c ▾ nvidia_a30-4c nvidia_a30-6c nvidia_a30-8c nvidia_a30-12c nvidia_a30-24c</div>	
> Video card	Specify custom settings ▾	
VMCI device		
> Other	Additional Hardware	

CANCEL OK

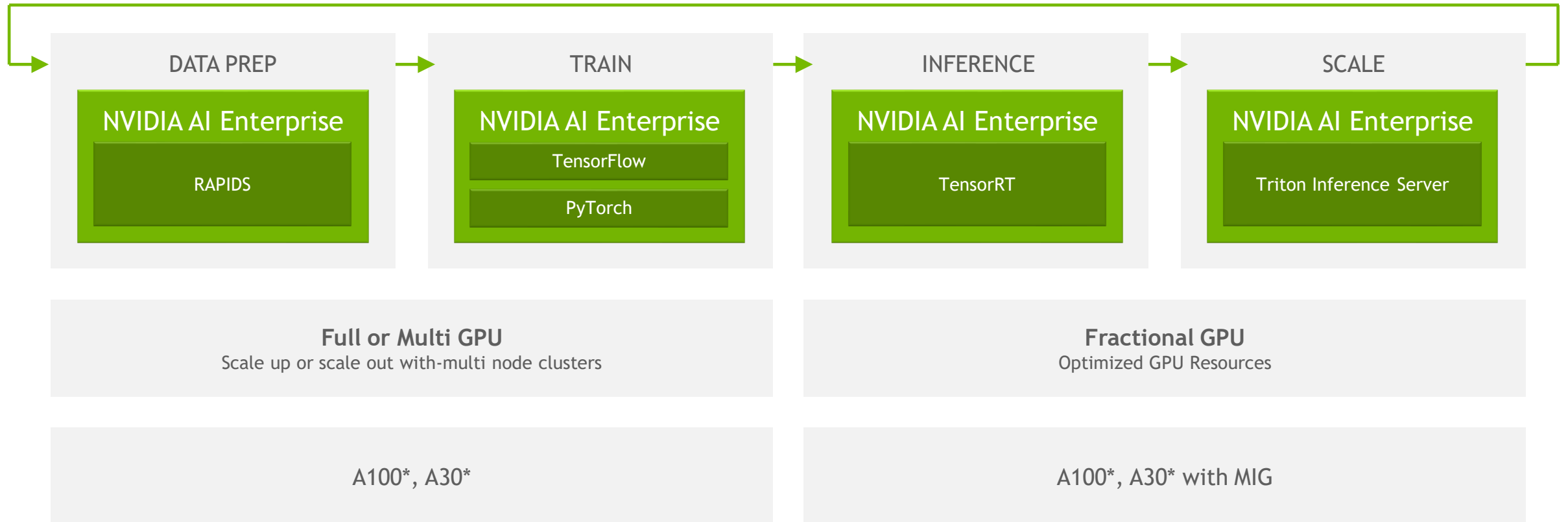
Virtual machine operations are unavailable when high devices are present. Consult user guide for operation limitations with PCI/PCIe passthrough devices.

Machine Learning Workflows in Practice



NVIDIA END-TO-END AI SOFTWARE SUITE

Typical AI Workflow | How They're Deployed



<https://docs.nvidia.com/datacenter/tesla/mig-user-guide/>

** Recommended GPU for NVIDIA AI Enterprise. Other NVIDIA GPUs are supported as well.*

DATA PREPROCESSING

In a GPU machine learning pipeline, the data never leaves the GPU.

cuDF has pandas like APIs for data wrangling.

Additional helper functions for Tokenization (for language models) on GPU which is 30X faster than preprocessing on CPU. Can be scaled on multiple GPUs with Dask.

Types of data

Static

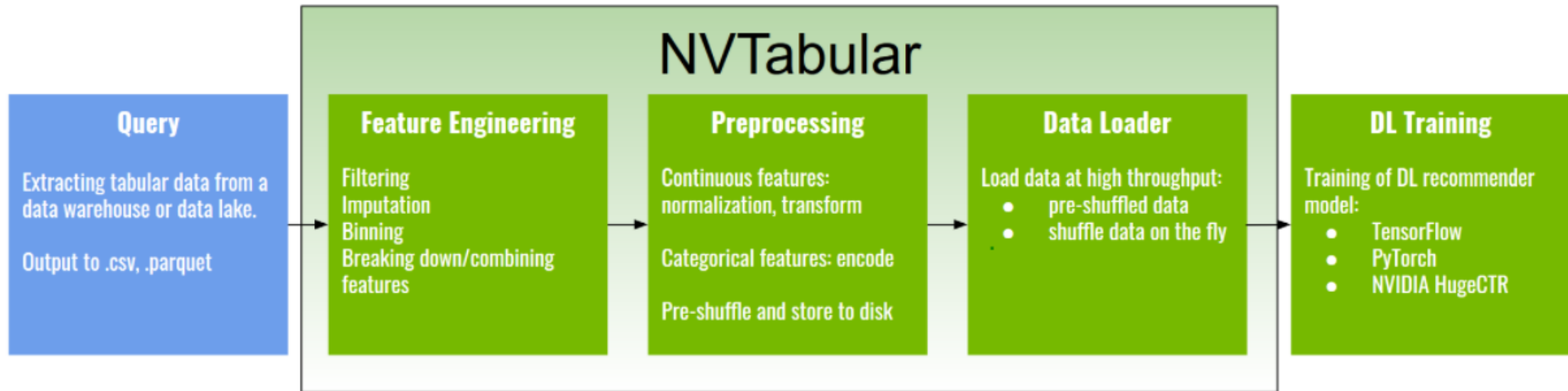
NV Tabular

Streaming

cuStreamz

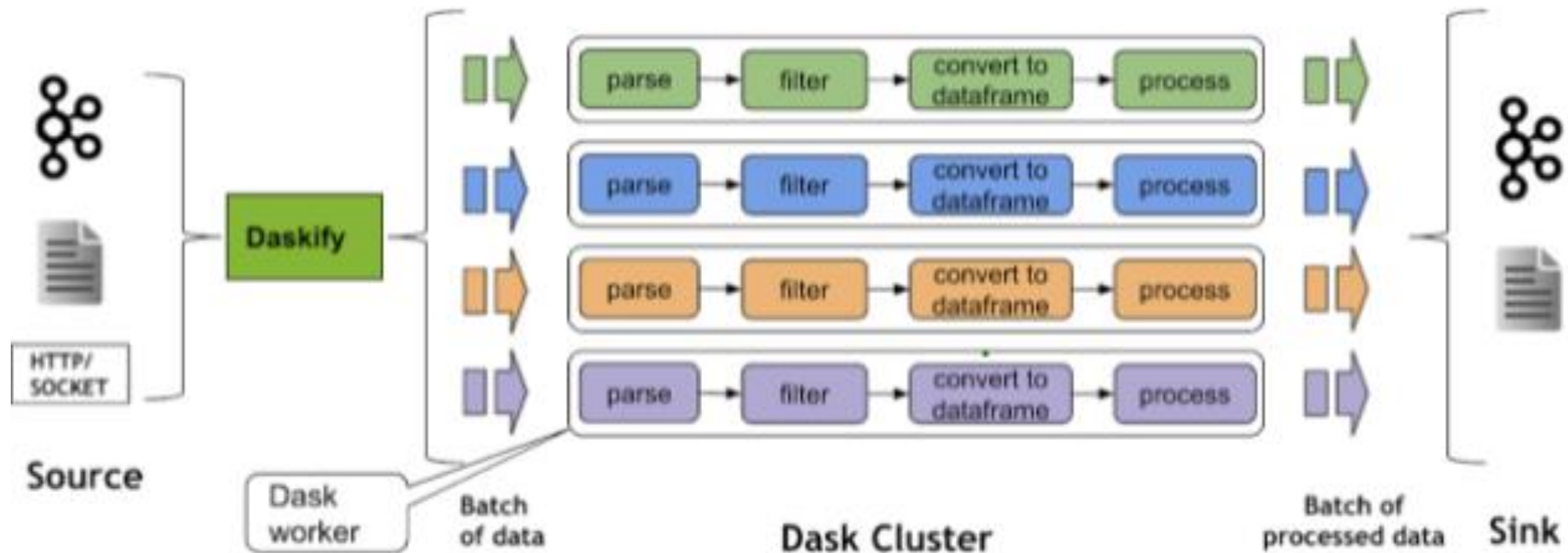
Available with support as part of the NVIDIA AI Enterprise software suite

NVTABULAR



Available with support as part of the NVIDIA AI Enterprise software suite

CUSTREAMZ

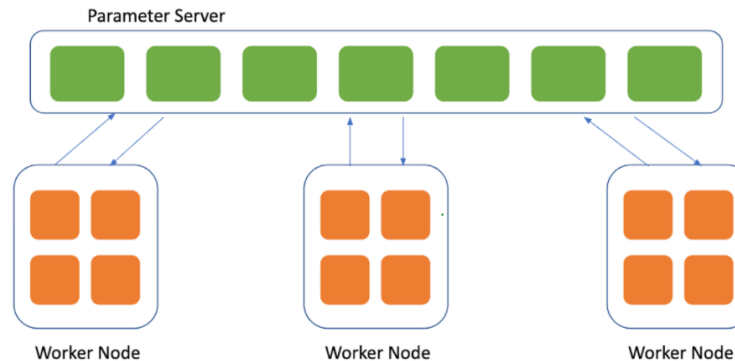


Distributed cuStreamz workflow using Dask

MODEL TRAINING

Horovod and MPI

- Horovod is a distributed deep learning training framework for TensorFlow, Keras, PyTorch, and Apache MXNet. The goal of Horovod is to make distributed deep learning fast and easy to use.
- MPI with NCCL can launch process on remote machines which can communicate to each other over TCP sockets or infiniband using NCCL. Horovod is the application layer on top of Tensorflow that works with MPI to make distributed training possible.



MODEL TRAINING

MultiGPU and Multinode Training

Magnum IO: The NVIDIA MAGNUM IO software development kit (SDK) enables developers to remove input/output (IO) bottlenecks in AI training, high performance computing (HPC), data science.

Components

- NVIDIA Collective Communication Library (NCCL)
 - Implements multi-GPU and multi-node communication primitives optimized for NVIDIA GPUs and Networking.
- MOFED
 - Drivers to enable Infiniband and RoCE for Multinode communications.
- GPU Direct RDMA (GDRDMA) and GPU Direct Storage (GDS)
 - Read data directly from Disk to GPU (GDS) and access the address space of a remote GPU (GDRDMA) without CPU Intervention.

MODEL TRAINING

Kubernetes operator support for MPI

- The MPI Operator makes it easy to run allreduce-style distributed training on Kubernetes.
- Different from Tensorflow or Pytorch operator. It is decoupled from the underlying machine learning framework and has support for Horovod.
- Has a CRD to specify the the master and worker pods. Mpi commands with NCCL flags can be specified in the command spec of the CRD.

```
apiVersion: kubeflow.org/v1alpha2
kind: MPIJob
metadata:
  name: tensorflow-benchmarks
spec:
  slotsPerWorker: 1
  cleanPodPolicy: Running
  mpiReplicaSpecs:
    Launcher:
      replicas: 1
      template:
        spec:
          containers:
            - image: mpioperator/tensorflow-benchmarks:latest
              name: tensorflow-benchmarks
              command:
                - mpirun
                - python
                - scripts/tf_cnn_benchmarks/tf_cnn_benchmarks.py
                - --model=resnet101
                - --batch_size=64
                - --variable_update=horovod
    Worker:
      replicas: 2
      template:
        spec:
          containers:
            - image: mpioperator/tensorflow-benchmarks:latest
              name: tensorflow-benchmarks
          resources:
            limits:
              nvidia.com/gpu: 1
```

MODEL TRAINING

NVIDIA AI Enterprise with MPI Operator

- Start with the base Dockerfile available at <https://github.com/kubeflow/mpi-operator/blob/master/build/base/Dockerfile>
- Add NVIDIA AI Enterprise Tensorflow container as the base container.
- Create the new Docker image and upload to the private registry.
- Create a Persistent volume to hold the dataset
- Install MPI operator

git clone <https://github.com/kubeflow/mpi-operator>

cd mpi-operator

kubectl apply -f deploy/v2beta1/mpi-operator.yaml

```
FROM nvcr.io/nvae/tensorflow-1-1:21.08-nvae1.1-tfl-py3

ARG port=2222

RUN apt update && apt install -y --no-install-recommends \
    openssh-server \
    openssh-client \
    libcap2-bin \
    && rm -rf /var/lib/apt/lists/*
# Add privilege separation directory to run sshd as root.
RUN mkdir -p /var/run/sshd
# Add capability to run sshd as non-root.
RUN setcap CAP_NET_BIND_SERVICE=+eip /usr/sbin/sshd

# Allow OpenSSH to talk to containers without asking for confirmation
# by disabling StrictHostKeyChecking.
# mpi-operator mounts the .ssh folder from a Secret. For that to work, we need
# to disable UserKnownHostsFile to avoid write permissions.
# Disabling StrictModes avoids directory and files read permission checks.
RUN sed -i "s/[ #]\(.*StrictHostKeyChecking \).* / \lno/g" /etc/ssh/ssh_config \
    && echo "    UserKnownHostsFile /dev/null" >> /etc/ssh/ssh_config \
    && sed -i "s/[ #]\(.*Port \).* / \l$port/g" /etc/ssh/ssh_config \
    && sed -i "s/[ #]\(StrictModes \).* / \lno/g" /etc/ssh/sshd_config \
    && sed -i "s/[ #]\(Port \).* / \l$port/g" /etc/ssh/sshd_config

RUN useradd -m mpiuser
WORKDIR /home/mpiuser
# Configurations for running sshd as non-root.
COPY --chown=mpiuser sshd_config .sshd_config
RUN echo "Port $port" >> /home/mpiuser/.sshd_config
```

MODEL TRAINING

MPI Operator with Network Operator and GPU Direct

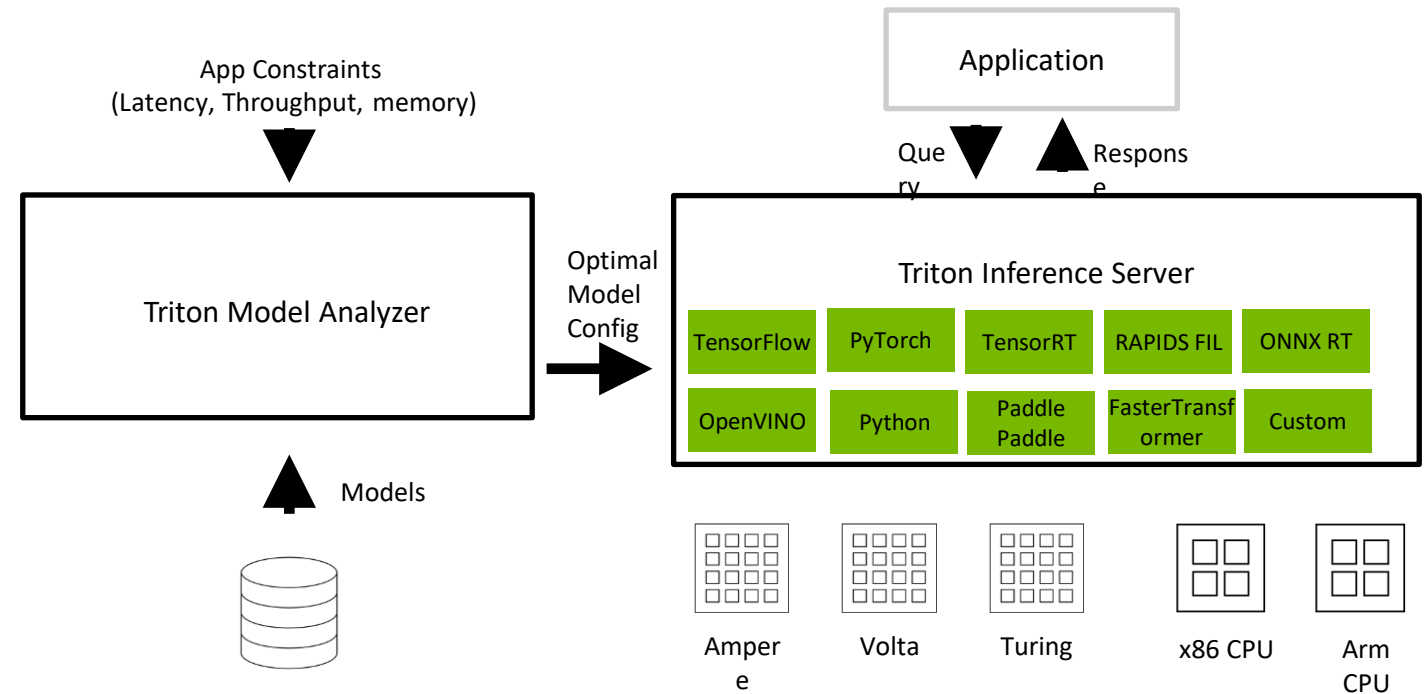
- Setup your Ethernet Switch 100/200G to enable RoCE on a separate VLAN
- Install Network Operator (as covered in previous slides)
 - Make sure to specify the VLAN ID in the NetworkAttachmentDefinition
- Specify the overlay network in the MPI operator CRD (mlnxdma) in the graphic on right
- Add Mellanox resource to the CRD (nvidia.com/sriov_rdma:1) in the graphic on the right
- The Network operator comes with nv_peer_mem pod for GPUdirect and NCCL should make use of it by default

```
Worker:
  replicas: 2
  template:
    metadata:
      annotations:
        k8s.v1.cni.cncf.io/networks: mlnxdma
    spec:
      volumes:
        - name: task-pv-storage
          persistentVolumeClaim:
            claimName: dataset-pv-claim
      containers:
        - image: nvcr.io/nvaie-tme/mpl-operator:latest
          name: tensorflow-benchmarks
          volumeMounts:
            - mountPath: "/data"
              name: task-pv-storage
          securityContext:
            capabilities:
              add: [ "IPC_LOCK" ]
          resources:
            limits:
              nvidia.com/gpu: 1
              nvidia.com/sriov_rdma: 1
```


MODEL INFERENCE WITH TRITON INFERENC

Bringing Fast and Scalable AI to Applications

All Major Frameworks, Major Clouds, AI Platforms
Diverse query types - Real time, Offline batch, Video/Audio streaming, Ensembles
Model Analyzer Optimizes For App Constraints
Distributed Multi-GPU Multi-Node Inference



DEPLOY AT SCALE



TRITON INFERENC

Available with support as part of the NVIDIA AI Enterprise software suite
<https://developer.nvidia.com/nvidia-triton-inference-server>

MODEL INFERENCE

Model Repository

The Triton Inference Server serves models from one or more model repositories that are specified when the server is started. While Triton is running, the models being served can be modified .

The corresponding repository layout must be:

```
<model-repository-path>/  
  <model-name>/  
    [config.pbtxt]  
    [<output-labels-file> ...]  
    <version>/  
      <model-definition-file>  
    <version>/  
      <model-definition-file>
```

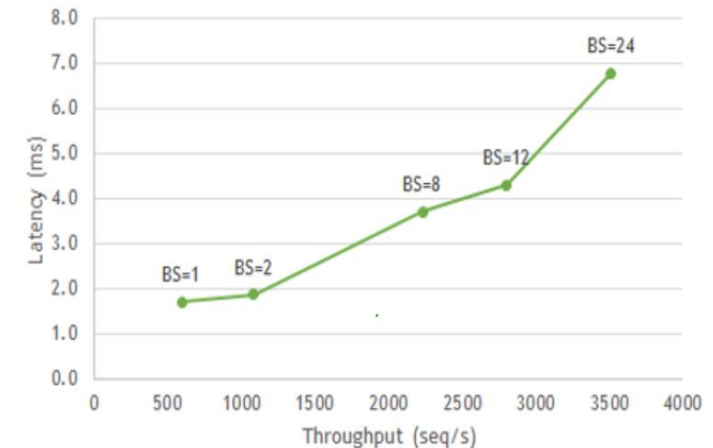
Start the server by pulling the NVIDIA AI Enterprise Triton Inference server container and pointing it to the model repository

```
$ tritonserver --model-repository=/path/to/model/repository
```

MODEL INFERENCE

Triton Inference Server on Kubernetes

- Triton Inference server can be deployed as a Kubernetes service inside the cluster.
- It is preferable to setup the model repository on a volume mount with a Persistent Volume Claim.
- **Latency vs Batching**
 - Triton supports batch inferencing by allowing individual inference requests to specify a batch of inputs.
 - The inferencing for a batch of inputs is performed at the same time which is especially important for GPUs since it can greatly increase inferencing throughput.
 - In many use cases the individual inference requests are not batched, therefore, they do not benefit from the throughput benefits of batching.
 - The inference server contains multiple scheduling and batching algorithms that support many different model types and use-cases.
 - A balance between the latency and throughput requirements must be maintained and the correct value depends on the individual use case.
 - Dynamic batching is a feature of Triton that allows inference requests to be combined by the server, so that a batch is created dynamically.



MODEL INFERENCE

Triton Forest Inference Library (FIL)

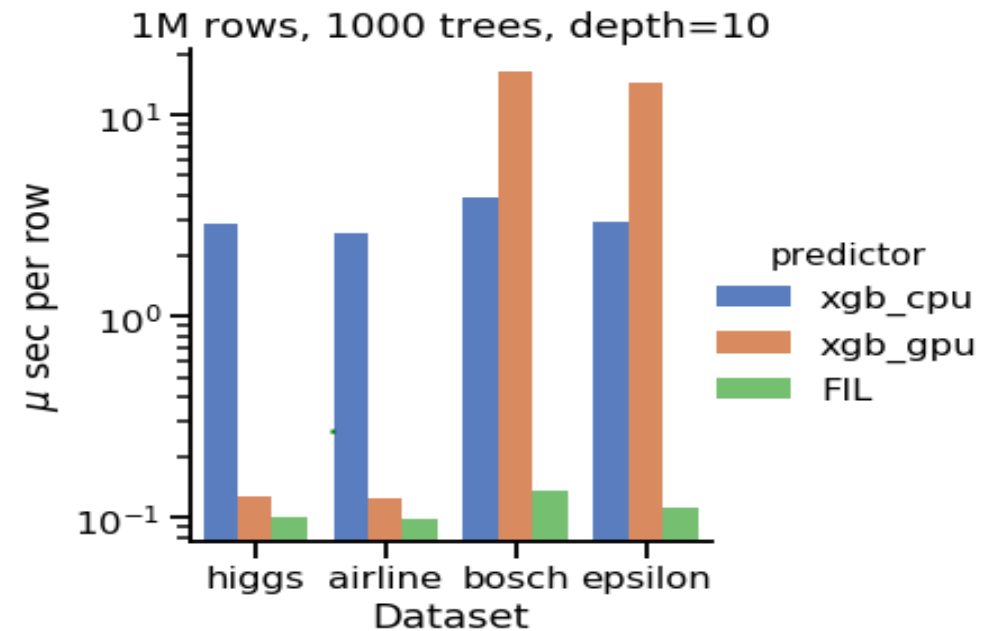
Triton Inference server in addition to serving Deep learning models also has libraries to host XGBoost and Random Forest Models through the FIL backend.

Models are served in a similar manner as regular deep learning models in a model repository.

```
model_repository/  
  |-- fil  
  |   |-- 1  
  |   |-- xgboost.model  
  |-- config.pbtxt
```

Model Configuration file (config.pbtxt) needs to be specified which has information like Model batch size, input and output shapes and threads per tree etc.

The Latency is much better than using XGBoost Inference directly performed on Python.



CPU and GPU performance across datasets

MODEL INFERENCE

Autoscaling Triton Inference Server on Kubernetes

As your service starts becoming more popular over time, the number of inference requests increase. It then becomes important for the service to make use of more compute(GPU) power.

Traditionally, the devops admin gauges server load (requests per second) and then adds additional resources depending on the load and scales down the resources when the load decreases.

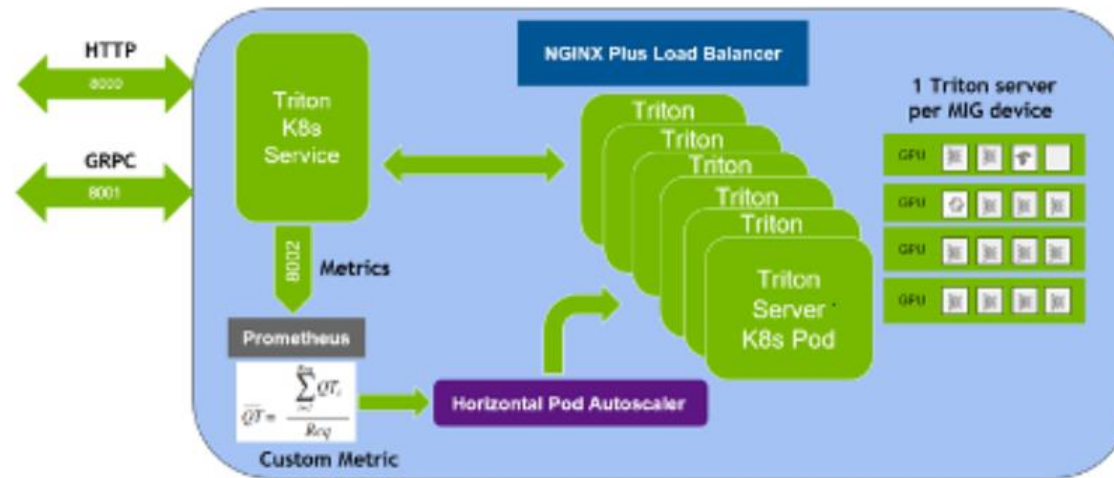
Kubernetes automates this workflow using Horizontal Pod autoscaler (HPA).

The Kubernetes Horizontal Pod auto scaler automatically scales the number of Pods in a Deployment, replication controller, or replica set based on that metrics like CPU utilization.

By providing custom metrics like GPU Utilization, Duty cycle etc to the HPA, the Triton Inference server pods can autoscale on demand based on these Metrics

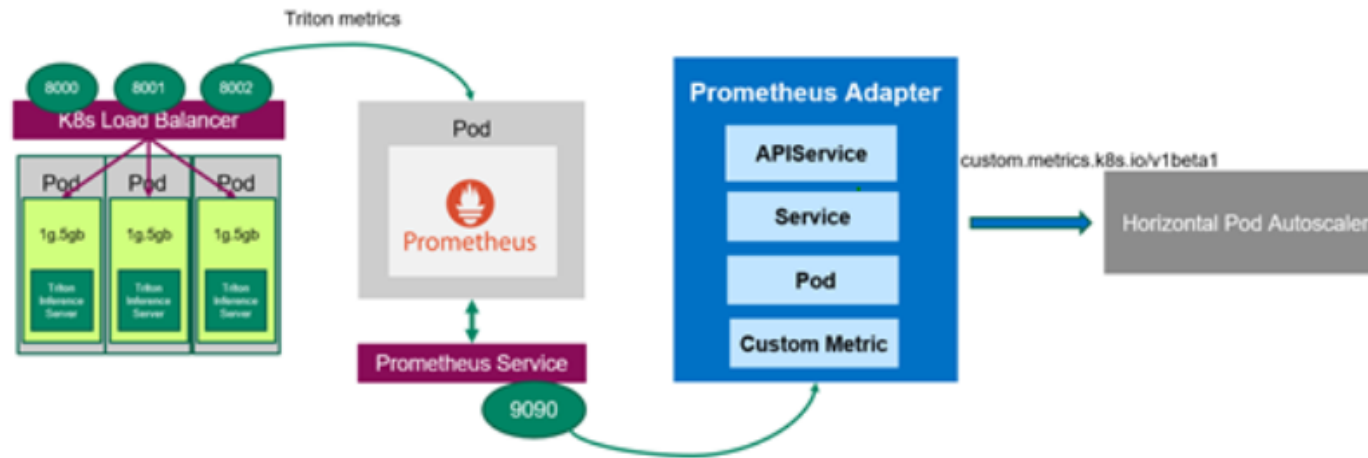
MODEL INFERENCE

Autoscaling with Kubernetes



MODEL INFERENCE

Autoscaling with Kubernetes



MODEL INFERENCE

Steps to Autoscale Triton Inference Server

Custom Metrics Server

The custom metrics server exposes custom metrics for Horizontal Pod auto scaler to the API server. Custom metrics server can be deployed on Kubernetes as follows.

```
kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
```

NVIDIA DCGM Exporter Service

To gather GPU telemetry in Kubernetes, the Nvidia Data Center GPU Manager (DCGM) is used. This suite of data center management tools allows you to manage and monitor GPU resources in an accelerated data center. Since the DevOps Engineer already installed the [GPU Operator](#), the NVIDIA DCGM exporter service is already installed onto the cluster.

MODEL INFERENCE

Steps to Autoscale Triton Inference Server

Prometheus Server

To expose cluster-level and node-level metrics, Prometheus is used. Prometheus, a [Cloud Native Computing Foundation](#) project, is a systems and service monitoring system. It collects metrics from configured targets at given intervals, evaluates rule expressions, displays the results, and can trigger alerts when specific conditions are observed. Refer to the [guide](#) on the GPU Operator website to set up Prometheus on your cluster.

Install Prometheus Adapter

The Prometheus adapter exposes the Prometheus metrics from the DCGM exporter to the custom metrics server we deployed. Therefore, this adapter is suitable for use with the autoscaling/v2 Horizontal Pod auto scaler in Kubernetes 1.16+. It can also replace the [metrics server](#) on clusters that already run Prometheus and collect the appropriate metrics.

MODEL INFERENCE

Steps to Autoscale Triton Inference Server

Verify if the Custom Metrics are Available to the Metrics Server

```
nvidia@node1:~/yaml$ kubectl get --raw /apis/custom.metrics.k8s.io/v1beta1 | jq -r . |  
grep DCGM_FI_DEV_MEM_COPY_UTIL
```

```
"name": "pods/DCGM_FI_DEV_MEM_COPY_UTIL",  
"name": "jobs.batch/DCGM_FI_DEV_MEM_COPY_UTIL",  
"name": "namespaces/DCGM_FI_DEV_MEM_COPY_UTIL",
```

MODEL INFERENCE

Steps to Autoscale Triton Inference Server

Create a yaml file for Horizontal Pod Autoscaler.

```
kind: HorizontalPodAutoscaler
apiVersion: autoscaling/v2beta1
metadata:
  name: gpu-hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: bert-qa
  minReplicas: 1
  maxReplicas: 3
  metrics:
  - type: Pods
    pods:
      metricName: DCGM_FI_DEV_GPU_UTIL # Average GPU usage of the pod.
      targetAverageValue: 40
```

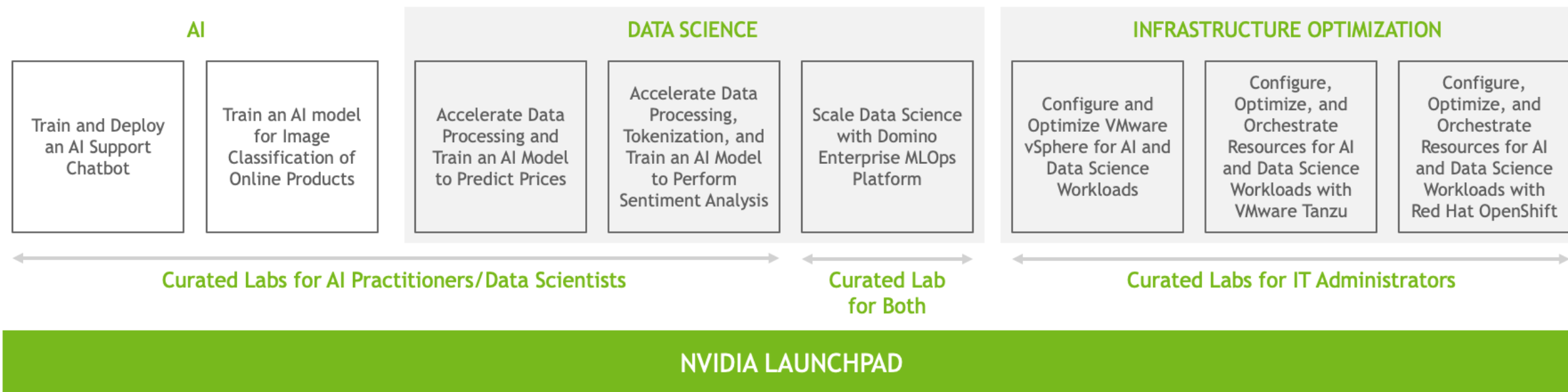
minReplicas is the lower bound to the number of pods to scale down to in the pod auto scaler deployment, with **maxReplicas** being the upper bound. The custom metric on which the pods are to be auto scaled is DCGM_FI_DEV_GPU_UTIL (which is the average GPU Utilization). If it exceeds the average target value of 40 percent, a new pod is scheduled.

FAST TRACK YOUR AI JOURNEY WITH NVIDIA LAUNCHPAD

With Curated Labs for NVIDIA AI Enterprise

- Immediate, short-term, remote access to accelerated compute infrastructure
- Hands-on experience tailored for AI Practitioners, Data Scientists, and IT Administrators
- Available globally

► **APPLY NOW TO GET STARTED: nvidia.com/try-ai**



NVIDIA AI ENTERPRISE SESSIONS TO CHECK OUT AT GTC SPRING 2022

Democratizing AI for the Enterprise

SESSIONS (Data Center & Virtualization)

Session ID	Title
S41858	What Every Business Leader Needs to Know to be Successful with AI
S41894	Containers or VMs: Deploy AI Workloads with Ease
S41871	NVIDIA AI Enterprise 101: Technology Session
S41864	Developing AI with Enterprise-Ready Kubernetes
S41876	Running Cloud Native Apps in NVIDIA AI Enterprise
S41867	Virtualize GPU-accelerated Data Science and AI Workflows in Your Data Center with Enterprise MLOps
S41877	How to Implement AI Across the Enterprise
S41551	Architecting the Next Generation Accelerated Data Center
S42061	Medical Image Reconstruction with Memory-Efficient Neural Networks
S41308	Scaling Remote Healthcare & Wellness: Virtualized GPU Acceleration of Mixed AI Workloads
S41382	Start Your AI Journey in a VMware Data Center
S41838	Tuning Virtualized GPUs for Optimal Performance on ML/AI Workloads
S41883	Manage hyper-converged and accelerated workloads in edge virtual data centers
S41307	A Modern Approach to End-to-End AI/ML: Learn How to Deliver Self-Service MLOps
S42535	AI Your Way: Solutions for Every Organization and VMware