NVIDIA AI Enterprise and Charmed Kubernetes Deployment Guide

Contents

Overview

This document provides a comprehensive guide for installing Charmed Kubernetes with NVIDIA GPU Operator providing the ideal platform to run NVIDIA AI Enterprise Software.

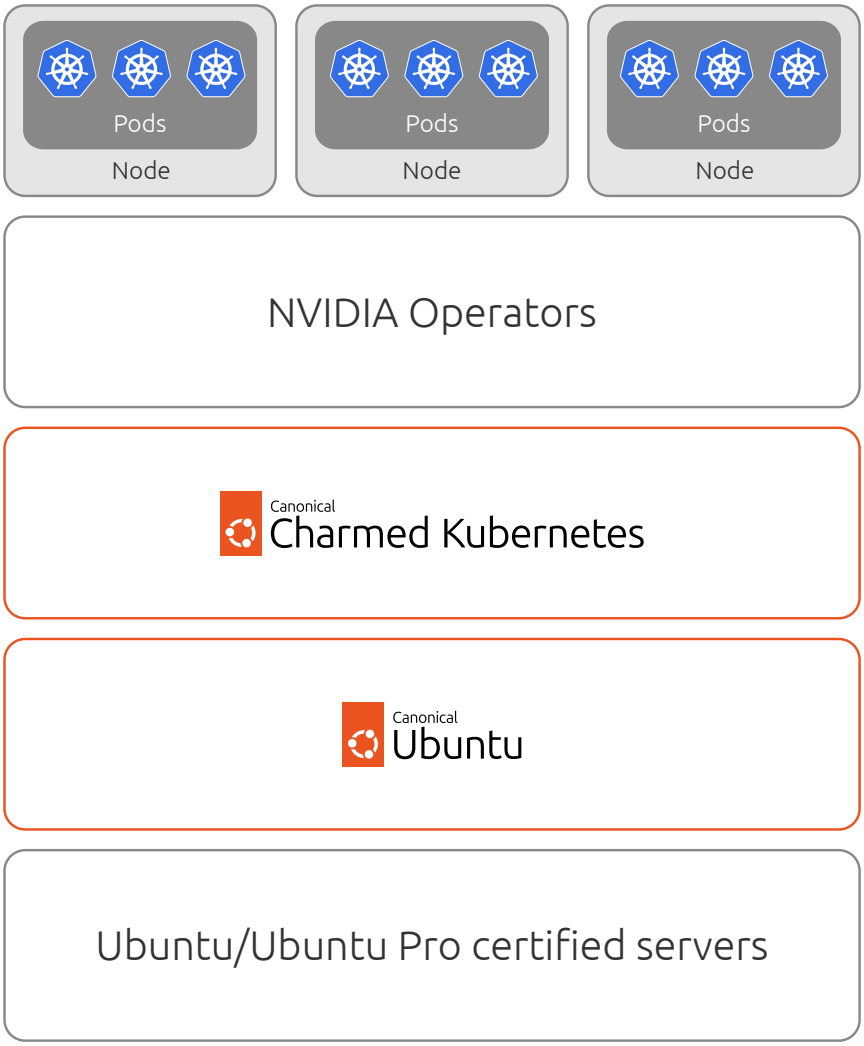

Fig 1. Ideal stack for running Charmed Kubernetes with NVIDIA AI Enterprise on Ubuntu/Ubuntu Pro certified servers.

Introduction

Ubuntu Server

Ubuntu Server is a version of the Ubuntu operating system designed specifically for server and data center use. Ubuntu Pro is a subscription-based offering that extends the standard Ubuntu distribution with additional features and support for enterprise environments.

With Ubuntu Pro, organizations gain access to an expanded security maintenance coverage that spans more than 30,000 packages for a duration of 10 years, and optional enterprise-grade phone and ticket support by Canonical.

Canonical Juju

Juju is an open-source application modeling tool that enables rapid deployment, configuration, scaling, and management of cloud infrastructures. It works seamlessly on various cloud platforms, including public clouds such as AWS, GCE, and Azure, as well as private clouds such as MAAS, OpenStack, and VMware vSphere®.

A Juju charm is a modular and reusable encapsulation of application deployment instructions within the Juju orchestration framework.

Serving as an abstraction layer, charms simplify the deployment and configuration of services, promoting consistency across environments.

They include configurable parameters, support relations between different services, and can be easily shared via the Juju Charmhub. By using charms, Juju streamlines the complex task of managing applications, reducing manual errors and providing a standardized approach to service orchestration. Juju uses charms to define, deploy, and manage services across diverse cloud and on-premises environments.

Charmed Kubernetes

Kubernetes is an open-source container orchestration platform that simplifies the deployment, scaling, and management of containerized applications. Canonical offers commercial distribution and support for a pure upstream version of Kubernetes. Charmed Kubernetes leverages the Juju model-driven approach to build Kubernetes clusters. Charms are used within the model to deploy, configure, and manage cloud services, encapsulating all the necessary code and expertise, including integration with other related applications and upgrade processes. This approach enables the rapid and consistent deployment and management of Kubernetes clusters.

Charmed Kubernetes is typically deployed using a charm bundle, which includes multiple charms, configuration options, and optional elements such as hardware and network constraints.

Canonical and NVIDIA have joined forces to harness the potential of AI across all sectors by providing a comprehensive enterprise platform tailored for AI tasks. This unified platform offers top-notch AI software, the NVIDIA AI Enterprise suite, meticulously fine-tuned for Charmed Kubernetes, the leading platform for

Containers and Kubernetes in the industry. Utilizing NVIDIA-Certified Systems™, which are renowned for their accelerated servers, this platform expedites the AI and high-performance data analytics development process, empowering businesses to expand modern workloads on their existing infrastructure investments while ensuring enterprise-grade management, security, and availability. Moreover, with Charmed Kubernetes, enterprises have the versatility to deploy in either bare-metal or virtualized environments.

Charmed Kubernetes utilizes the NVIDIA Operators to enhance the performance of each node. Its capabilities in load balancing, autoscaling inferencing requests, and streamlining the scaling out of training jobs make it well-suited for production environments and the concluding phases of the AI development lifecycle. Moreover, both Charmed Kubernetes and the NVIDIA GPU Operator provide metrics to Prometheus, an open-source monitoring and alerting toolkit. This facilitates the effortless visualization of GPU metrics via Grafana dashboards.

The advantages of this collaborative NVIDIA and Canonical solution include:

- **Streamlined Deployment and Scalability:** The ease of deployment and scaling is a significant advantage for enterprises opting for an end-to-end AI solution certified by both NVIDIA and Canonical on NVIDIA-Certified Systems. This offers the flexibility to deploy the NVIDIA AI Enterprise and data analytics software consistently on Charmed Kubernetes, whether on bare metal or virtualized infrastructure across data centers and edge locations. The integration of Charmed Kubernetes with NVIDIA GPUs via the certified Kubernetes Operator and containerized AI software mitigates deployment risks and facilitates smooth scaling operations.

- **Self-service access to AI tools and infrastructure:** NVIDIA AI Enterprise on Charmed Kubernetes provides data scientists, ML engineers, and developers with a self-service, uniform, cloud-like experience. This setup offers flexibility and portability, allowing users to utilize containerized AI tools and infrastructure resources efficiently. As a result, they can quickly develop, scale, replicate, and collaborate on models before deploying them into production. Additionally, users have access to pre-validated AI tools out of the box, enhancing productivity and accelerating time to value.

- **Enhanced Security in Application Delivery with the integration of MLOps:** Expanding the DevOps automation capabilities of Charmed Kubernetes throughout the AI lifecycle facilitates enhanced collaboration among data scientists, ML engineers, software developers, and IT operations teams. This integration empowers organizations to automate and streamline the iterative process of incorporating models into software development workflows, deploying them into production, monitoring their performance, retraining them as necessary, and redeploying them to maintain prediction accuracy over time.

NVIDIA Kubernetes Operators

NVIDIA Kubernetes Operators are designed to enhance the management and optimization of GPU and network resources within Kubernetes clusters. These operators leverage the Kubernetes custom resources and Operator framework to streamline various tasks, from GPU management to networking configurations. NVIDIA Kubernetes Operators are deployed as charms in the Charmed Kubernetes environment with Juju, further simplifying the deployment and operation of GPU-accelerated workloads in Kubernetes environments.

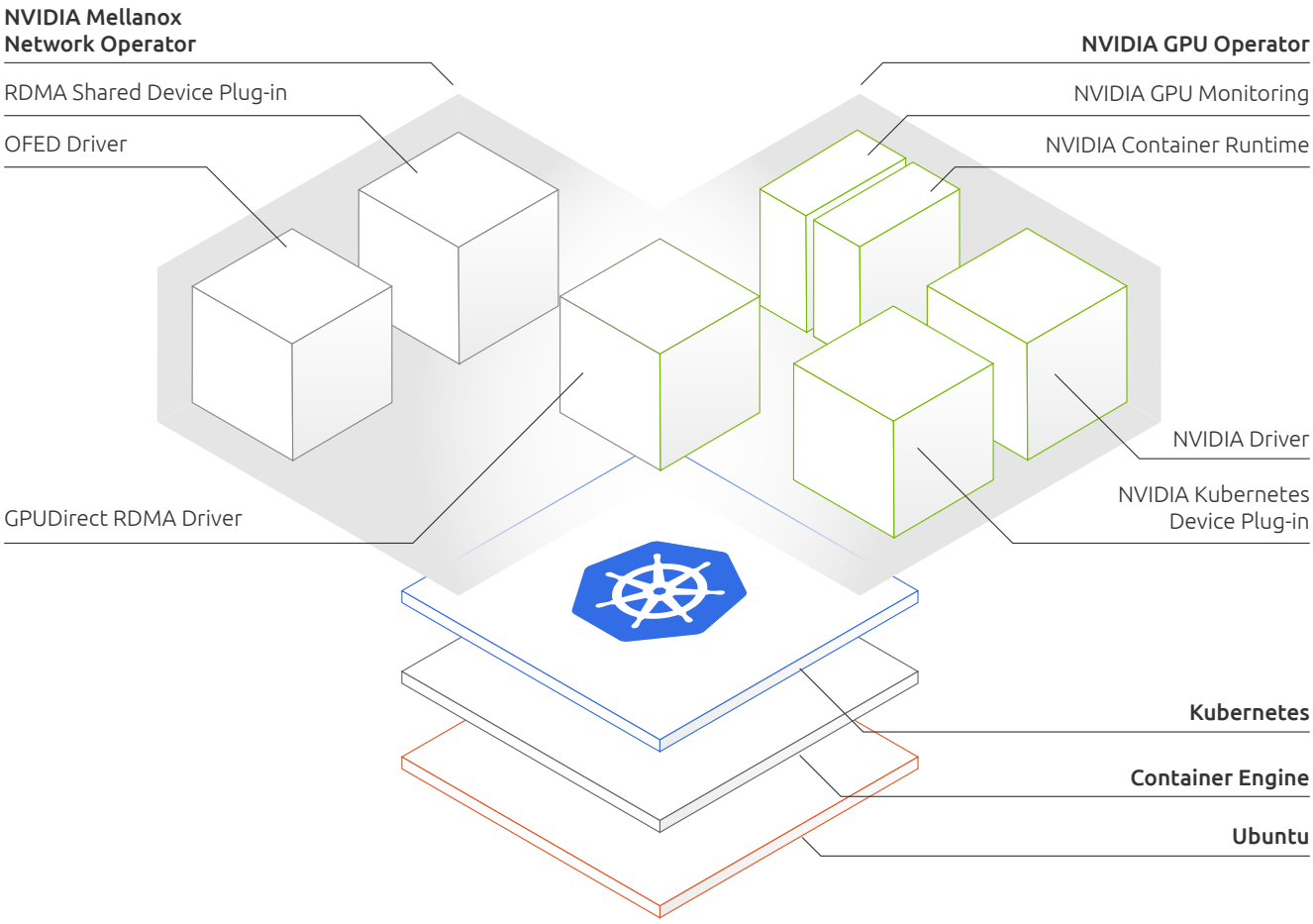

Fig 2.NVIDIA Kubernetes Operators

NVIDIA GPU Operator

When integrated with Charmed Kubernetes, the GPU Operator enables DevOps teams to efficiently manage GPU lifecycles at a cluster level, eliminating the need for individual node management. With its capability to detect newly added GPU-accelerated Kubernetes worker nodes, the GPU Operator enables faster node provisioning by automatically installing all necessary software components for running GPU-accelerated applications. The GPU Operator serves as a unified tool for managing all Kubernetes components, including the GPU Device Plugin, GPU Feature Discovery, GPU Monitoring Tools, NVIDIA Runtime, and importantly, it installs the NVIDIA Datacentre Driver or NVIDIA vGPU Guest Drivers.

The components include:

- **GPU Feature Discovery:** This component labels the worker node according to the GPU specifications. It provides customers with the ability to selectively choose the GPU resources necessary for their applications.

- **NVIDIA vGPU Guest Drivers:** This driver is deployed in a containerized format, facilitating the operation of NVIDIA AI Enterprise within the Kubernetes environment.

- **Kubernetes Device Plugin:** Responsible for advertising the GPU to the Kubernetes scheduler, enabling efficient allocation of GPU resources.

- **NVIDIA Container Toolkit:** This toolkit enables the building and execution of GPU-accelerated containers. It comprises a container runtime library and utilities to automatically configure containers for optimal utilization of NVIDIA GPUs.

- **Data Center GPU Manager (DCGM) Monitoring:** Allows for the monitoring of GPUs within the Kubernetes environment, ensuring efficient resource management and performance optimization.

NVIDIA Network Operator

The goal of the Network Operator is to manage the networking related components, while enabling execution of RDMA and GPUDirect RDMA workloads in a Kubernetes cluster. This includes:

- NVIDIA Networking drivers to enable advanced features

- Kubernetes device plugins to provide hardware resources required for a fast network

- Kubernetes secondary network components for network intensive workloads

Prerequisites

Hardware Requirements

NVIDIA AI Enterprise with Charmed Kubernetes has the following prerequisites:

- At least 3 NVIDIA AI Enterprise Compatible and NVIDIA-Certified servers:
  - One server for the Juju Controller.
  - One server for the Kubernetes control plane.
  - One server with an NVIDIA AI Enterprise supported NVIDIA GPU for the Kubernetes worker node.

- The recommended GPUs are H100 for training and L40S for inference.
  - Single Root I/O Virtualization (SR-IOV) Enabled (Optional)
  - VT-d/IOMMU Enabled (Optional)

- NVIDIA AI Enterprise Software must be installed on the GPU-accelerated server(s).

Deployment

Install Ubuntu Server 22.04 on each server. Please refer to the installation guide for further information.

Update each server with this command:

```
sudo apt update && sudo apt upgrade -y
```

Install Juju

In Juju, a controller is the initial cloud instance created during bootstrapping. It serves as the bridge between the Juju CLI and the cloud API, running an API server and maintaining a database to track the state of deployed models, applications, and machines within the environment. The controller is the central management node in a Juju cloud environment, responsible for executing changes defined by the Juju user.

Run the following commands on the server that will be the Juju Controller. Make sure that the Juju Controller can SSH into the manually provisioned servers.

Install the Juju snap:

```
sudo snap install juju
mkdir -p ~/.local/share/juju
```

Bootstrap the local server as the Juju Controller, named kubernetes:

```
juju bootstrap manual/ubuntu@localhost kubernetes
```

Add a new Juju model, named kubernetes-model:

```
juju add-model kubernetes-model
```

Add manually provisioned servers to Juju:

```
juju add-machine ssh:ubuntu@<IP address of the server that will be
the kubernetes control plane>
juju add-machine ssh:ubuntu@<IP address of the server with the
GPU>
```

Run the following command to ensure machine 0 is the server that will be the kubernetes control plane, and machine 1 is the server with the GPU.

```
juju machines
```

Download and extract the kubernetes-core Juju bundle:

```
juju download kubernetes-core
sudo apt install unzip
unzip *.bundle
```

Install Charmed Kubernetes

To install Charmed Kubernetes, follow these steps:

Edit bundle.yaml:

```
description: A minimal two-machine Kubernetes cluster, appropriate
for development.
issues: https://bugs.launchpad.net/charmed-kubernetes-bundles
series: jammy
source: https://github.com/charmed-kubernetes/bundle
website: https://ubuntu.com/kubernetes/charmed-k8s
name: kubernetes-core
machines:
  ‘0’:
  ‘1’:
applications:
  calico:
    channel: stable
    charm: calico
    options:
      vxlan: Always
  containerd:
    channel: stable
    charm: containerd
    options:
      gpu_driver: none
  easyrsa:
    channel: stable
    charm: easyrsa
    num_units: 1
    to:
    - ‘0’
  etcd:
    channel: stable
    charm: etcd
    num_units: 1
    options:
      channel: 3.4/stable
    to:
    - ‘0’
  kubernetes-control-plane:
    channel: stable
    charm: kubernetes-control-plane
    expose: true
    num_units: 1
    options:
      channel: 1.28/stable
    to:
    - ‘0’
  kubernetes-worker:
    channel: stable
    charm: kubernetes-worker
    expose: true
    num_units: 1
```

```
    options:
      channel: 1.28/stable
    to:
    - '1'
relations:
- - kubernetes-control-plane:kube-control
  - kubernetes-worker:kube-control
- - kubernetes-control-plane:certificates
  - easyrsa:client
- - kubernetes-control-plane:etcd
  - etcd:db
- - kubernetes-worker:certificates
  - easyrsa:client
- - etcd:certificates
  - easyrsa:client
- - calico:etcd
  - etcd:db
- - calico:cni
  - kubernetes-control-plane:cni
- - calico:cni
  - kubernetes-worker:cni
- - containerd:containerd
  - kubernetes-worker:container-runtime
- - containerd:containerd
  - kubernetes-control-plane:container-runtime
```

Deploy the bundle:

```
juju deploy ./bundle.yaml --map-machines=existing
```

Monitor the deployment until all statuses become active:

```
watch -c juju status --color
```

Install NVIDIA GPU Operator

The GPU Operator enables DevOps Engineers of Kubernetes clusters to manage GPU nodes seamlessly alongside CPU nodes within the cluster. It handles the installation and lifecycle management of software components, facilitating the execution of GPU-accelerated applications on Kubernetes.

Follow this procedure to deploy NVIDIA GPU Operator:
https://docs.nvidia.com/datacenter/cloud-native/gpu-operator/latest/install-gpu-operator-nvaie.html

Running a Sample GPU Application from the NVIDIA AI Enterprise Catalog

NVAIE containers are provided on https://catalog.ngc.nvidia.com. In this example we will run a simple CUDA sample, which adds two vectors together.

1. Log into the Kubernetes Control Plane node:

```
juju ssh kubernetes-control-plane
```

2. Create a file, such as cuda-vectoradd.yaml, with contents like the following:

```
apiVersion: v1
kind: Pod
metadata:
  name: cuda-vectoradd
spec:
  restartPolicy: OnFailure
  containers:
  - name: cuda-vectoradd
    image: “nvcr.io/nvidia/k8s/cuda-sample:vectoradd-cuda10.2”
    resources:
      limits:
        nvidia.com/gpu: 1
```

3. Run the pod:

```
kubectl apply -f cuda-vectoradd.yaml
```

The pod starts, runs the vectorAdd command, and then exits.

4. View the logs from the container:

```
kubectl logs pod/cuda-vectoradd
```

Example Output

```
[Vector addition of 50000 elements]
Copy input data from the host memory to the CUDA device
CUDA kernel launch with 196 blocks of 256 threads
Copy output data from the CUDA device to the host memory
Test PASSED
Done
```

5. Remove the stopped pod:

```
kubectl delete -f cuda-vectoradd.yaml
```

Example Output

```
pod “cuda-vectoradd” deleted
```

Conclusion

The deployment of Charmed Kubernetes with the NVIDIA GPU Operator and NVIDIA Network Operator represents a powerful solution for organizations seeking to harness the potential of AI and GPU-accelerated workloads. By leveraging Canonical's Juju and Charmed Kubernetes alongside NVIDIA's advanced GPU capabilities, enterprises can achieve streamlined deployment, enhanced scalability, and efficient management of AI workloads. This comprehensive platform provides a unified and secure environment for running NVIDIA AI Enterprise Software, empowering data scientists, ML engineers, and developers with self-service access to AI tools and infrastructure. The joint solution from NVIDIA and Canonical offers a robust foundation for organizations looking to accelerate their AI initiatives and drive innovation in their respective industries.

Canonical