Reference Architecture: Canonical Charmed Kubernetes on Lenovo ThinkSystem Server for AI Development

Last update: **16 January 2022**

Version 2.0

Reference Architecture for Kubernetes-based AI development

Validated architecture for deploying and monitoring a Kubernetes environment

Describes hardware and software infrastructure components for installation and deployment

AI workload deployment with Lenovo Intelligent Computing Orchestration (LiCO) and Kubeflow

Miroslav Hodak, Lenovo

Andrey Grebennikov, Canonical

LENOVO PRESS

Table of Contents

List of Figures

List of Tables

1 Introduction

This document describes the Lenovo hardware specifications as well as the tools and services to deploy the solution, including the foundation cluster, the Kubernetes cluster and tools used for cluster monitoring and management.

This guide also provides the deployment steps and references to configuration and automation scripts developed by Lenovo and Canonical for the deployment process. Examples for validating the deployed solution and the expected results are provided as well to ensure a successful implementation of Canonical Charmed Kubernetes for AI development on Lenovo ThinkSystem.

Lenovo and Canonical have worked together to build a jointly engineered and validated architecture that details software, hardware, and integration points of all solution components. The architecture provides prescriptive guidance and recommendations for:

- Hardware design
    - Infrastructure nodes
    - Cluster hyperconverged nodes
- Network hardware and design
- Software layout
- System configurations

The intended audience for this document is technical IT architects, system administrators, and managers who are interested in deploying a Canonical Charmed Kubernetes solution on Lenovo ThinkSystem to support AI model development.

2 Business problem and business value

Organizations are being driven to support on-premises Kubernetes clusters to facilitate containerized workload deployment and need to deploy upstream community versions to maximize compatibility with user applications. Enterprises will require formal support for these environments to ensure maximum uptime, as well as simplified tools to deploy, monitor, and manage the cluster lifecycle.

2.1 Business problem

Kubernetes is quickly becoming a requirement in many organizations, to support the dynamic nature of modern cloud-native applications. This modern approach for deploying containerized workloads can drive greater utilization from both hardware infrastructure and improve development and operational efficiency.

The complexity of architecting, deploying and managing Kubernetes environments has placed a large burden on IT to support the open-source components required to deploy this dynamic infrastructure, manage and monitor the cluster resources, and deploy the tools that users of the system need to fulfil their business objectives.

2.2 Business value

The Canonical Charmed Kubernetes solution delivers a supported upstream Kubernetes solution on Lenovo ThinkSystem infrastructure, that is optimized and validated for Machine Learning and Artificial Intelligence workloads. Software deployment and lifecycle management are simplified through the use of automation via Juju Charms, reducing the effort required of system administrators to maintain the environment.

The combined solution based on Lenovo ThinkSystem servers delivers flexible pools of compute, storage and networking resources which are managed through a single point of rack management, with Canonical’s commercially distributed and supported pure upstream version of Kubernetes based on Ubuntu.

3 Requirements

The functional and non-functional requirements for this reference architecture are described below.

3.1 Functional requirements

This architecture guide is based on upstream Kubernetes release 1.21 deployed on Lenovo ThinkSystem infrastructure.

With Charmed Kubernetes cluster deployed on Lenovo ThinkSystem compute, storage and networking infrastructure, administrators can provide flexible pools of resources for containerized workloads, including tools for AI development such as Kubeflow and Lenovo Intelligent Computing Orchestration (LiCO). The Kubernetes cluster abstracts the need to individually provision systems for deploying workloads, providing more flexibility to AI users in how they manage their data science workflow.

Figure 1 shows the two examples of tools AI users can use to deploy AI workloads on the cluster.

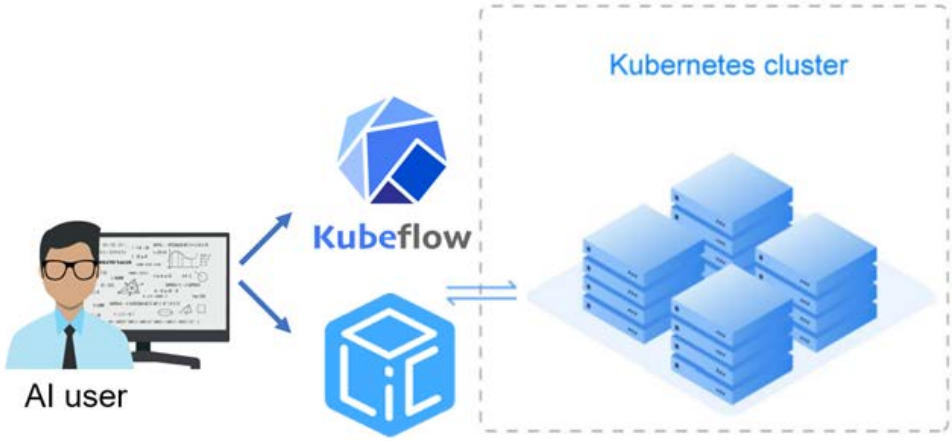


Figure 1. Kubernetes-based AI workload deployment tools

3.2 Non-functional requirements

In addition to appropriately architected physical infrastructure, administrators will need software tools to deploy, provision, scale, and manage the software infrastructure effectively. Juju is an open source application modelling tool that allows you to deploy, configure, scale, and operate cloud infrastructures quickly and efficiently. Metal as a Service (MAAS) enables the physical infrastructure to be automated and managed as a single resource pool. Capabilities of MAAS include:

Table 1: MAAS benefits

Requirement	Description
Automation	Automatic discovery and registration of every device on the network. BMC (IPMI, AMT and more) and PXE (IPv4and IPv6) automation.
Fast deployment	Zero-touch deployment of Ubuntu, CentOS, Windows, RHEL and

	SUSE. Deploys Linux distributions in less than 5 minutes.
Machine configuration	Configures the machine's network interfaces with bridges, VLANs, bonds and more. Creates advanced file system layouts with RAID, bcache, LVM and more.
DevOps integration	Integration with DevOps automation tools like conjure-up, Juju, Chef, Puppet, SALT, Ansible and more.
Pod management	Turns bare-metal servers into hypervisors, allowing automated creation of virtual machines and present them as new servers available for the deployment.
Network management	Observes and catalogs every IP address on the network (IPAM). Built-in highly available DHCP (active-passive) and DNS (active-active).
Service tracking	Monitors and tracks critical services to ensure proper operations.
Manage	Comes with a REST API, Web UI and CLI.

4 Architectural overview

The software and hardware architecture for this reference architecture are described below.

4.1 Software Architecture

4.1.1 Kubernetes

This architecture guide is based on upstream Kubernetes release 1.21. Charmed Kubernetes solution always includes the current upstream version of Kubernetes that is evolving at a very rapid pace, and the focus is to have an easily upgradeable solution to the next version once it is released.

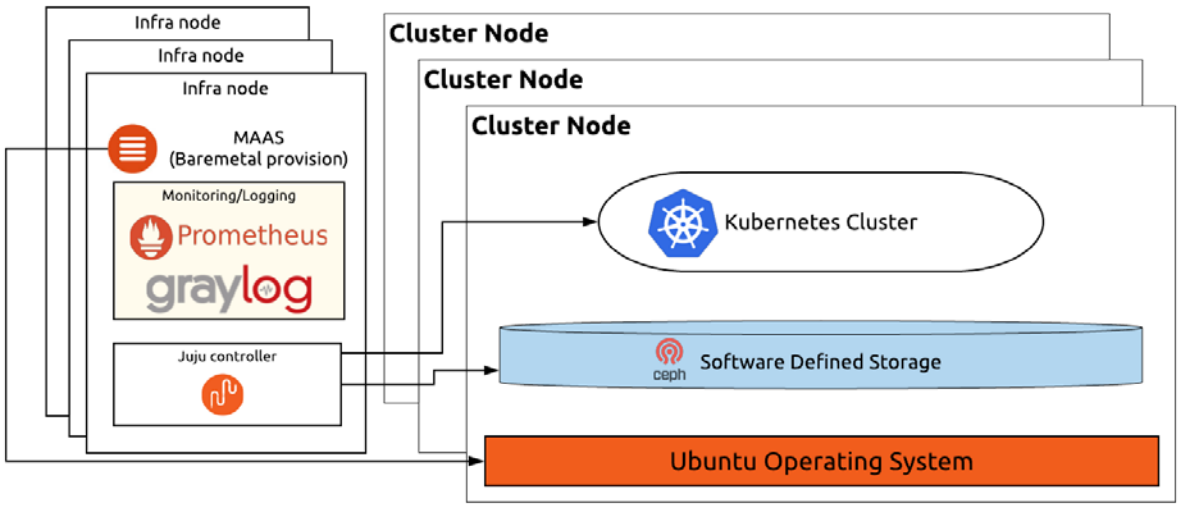
Table 2: Core solution components

Component	**Codename**
Persistent Storage	Ceph RBD, CephFS
Compute	Kubernetes Worker (Containerd based)
Networking	Calico
Logging	Graylog
Monitoring	Prometheus

The standards-based APIs are the same between all Kubernetes deployments, and they enable customer and vendor ecosystems to operate across multiple clouds. The site-specific infrastructure combines open and proprietary software, Lenovo hardware, and operational processes to deliver cloud resources as a service.

The diagram below represents software components of the solution laid out across physical nodes.

Figure 2. Reference software layout



The implementation choices for each cloud infrastructure are highly specific to the requirements of each site. Many of these choices can be standardized and automated using the tools in this reference architecture. Conforming to best practices helps reduce operational risk by leveraging the accumulated experience of Lenovo and Canonical.

Canonical's Metal as a Service (MAAS) is used as a bare metal and VM provisioning tool. The foundation cluster is composed of MAAS and other services (running in highly available (HA) mode) that are used to deploy, manage and update the Kubernetes cluster nodes.

4.1.2 Kubernetes and Canonical

This reference architecture is based on the Canonical's Charmed Kubernetes. Canonical commercially distributes and supports pure upstream version of Kubernetes. Ubuntu is the reference operating system for Kubernetes deployments, making it an easy way to build Kubernetes clusters. In Ubuntu, Kubernetes is delivered in the form of snaps - the universal Linux app packaging format - which dramatically simplifies the installation and upgrades of components.

Canonical Discoverer family of services provides the service to design, deploy, manage and support customer clouds in POC, development, pre-production and production environments.

Canonical reference architectures are delivered on a converged infrastructure approach, where any of the servers can accommodate more than one specific Kubernetes role or service simultaneously. This converged approach has many benefits, including simplicity of operation and management overhead. Canonical can also deploy Kubernetes in a more traditional manner, grouping servers per role - controllers, storage, and container pods.

4.1.3 MAAS (Metal as a Service) physical cloud

Metal as a Service (MAAS) is a complete automation of physical servers for data center operation efficiency on premises. It's open source and supported by Canonical.

MAAS treats physical servers like virtual machines, or instances in the cloud. Rather than having to manage

each server individually, MAAS turns bare metal into an elastic cloud-like resource.

MAAS provides management of a large number of physical machines by creating a single resource pool out of them. Participating machines can then be provisioned automatically and used as normal. When those machines are no longer required, they are "released" back into the pool. MAAS integrates all the tools you require in one smooth experience. It includes:

• Web UI, optimized for mobile devices

• Ubuntu, CentOS, Windows, RHEL and SUSE installation support open source IP Address Management (IPAM)

• Full API/CLI support

• High availability

• IPv6 support

• Inventory of components

• DHCP and DNS for other devices on the network

• DHCP relay integration

• VLAN and fabric support

• NTP for the entire infrastructure

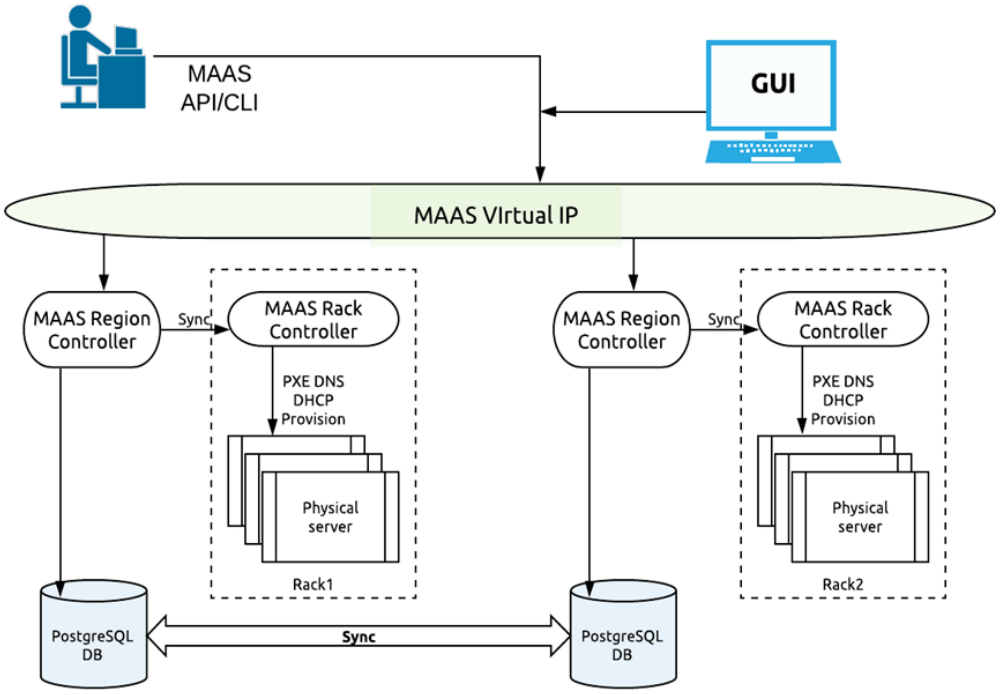
• Hardware testing

• Composable hardware support

MAAS works with any system configuration and is recommended by the teams behind both Chef and Juju as a physical provisioning system.

The Figure 3 below represents the logical architecture of MAAS and high availability of its components.

Figure 3. MAAS deployment logical design



4.1.4 Juju modelling tool

Juju is an open source application modelling tool that allows you to deploy, configure, scale, and operate cloud infrastructures quickly and efficiently on public clouds such as AWS, GCE, and Azure; along with private clouds such as Metal as a Service (MAAS), OpenStack, and VMware vSphere.

The Juju store allows access to a wide range of best practice solutions which you can deploy with a single command. You can use Juju from the command line or through its powerful graphical representation of the model in the GUI.

4.1.5 Why use Juju?

Whether it involves deep learning, container orchestration, real-time big data or stream processing, big software needs operations to be open source and automated.

Juju is the best way to encapsulate all the ops knowledge required to automate the behavior of the application.

4.1.6 Software versions

Table 3 lists the versions of software used as part of this reference architecture:

Table 3: Software stack versions

Component	Version

Ubuntu	20.04.3 LTS (Linux Kernel 5.4)
Kubernetes	1.21
MAAS	2.9
Juju	2.9.21
Kubernetes charms	latest

4.2 Hardware Architecture

The base reference architecture solution for Charmed Kubernetes on Lenovo ThinkSystem servers utilizes three (3) ThinkSystem SR630 v2 as infrastructure (management) nodes, and six (6) ThinkSystem SR650 v2 servers which serve as Cluster (compute/storage) nodes handling Kubernetes components and Storage functions in the cluster. Interconnect within the cluster is achieved with Mellanox SN2410 10/25GbE switches with redundant 8x 40GbE connections, and system BMC management access is via a Mellanox AS4610 1GbE switch.

Table 4: Lenovo ThinkSystem specifications

Component type	**Component description**	**Quantity**
Rack	Standard data center 42U rack	1
Management	Lenovo ThinkSystem SR630 v2 (Infrastructure nodes)	3
Compute	Lenovo ThinkSystem SR650 v2 (Cluster nodes) with 2x Nvidia A100 (Optional)	6
Data switches	Mellanox SN2410 25GbE	2
Provisioning switch	Mellanox AS4610-54T 1GbE	1

4.2.1 Rack layout

The reference deployment of Canonical Kubernetes on the Lenovo ThinkSystem utilizes three nodes as infrastructure nodes. The reference deployment uses the following nodes and their purpose:

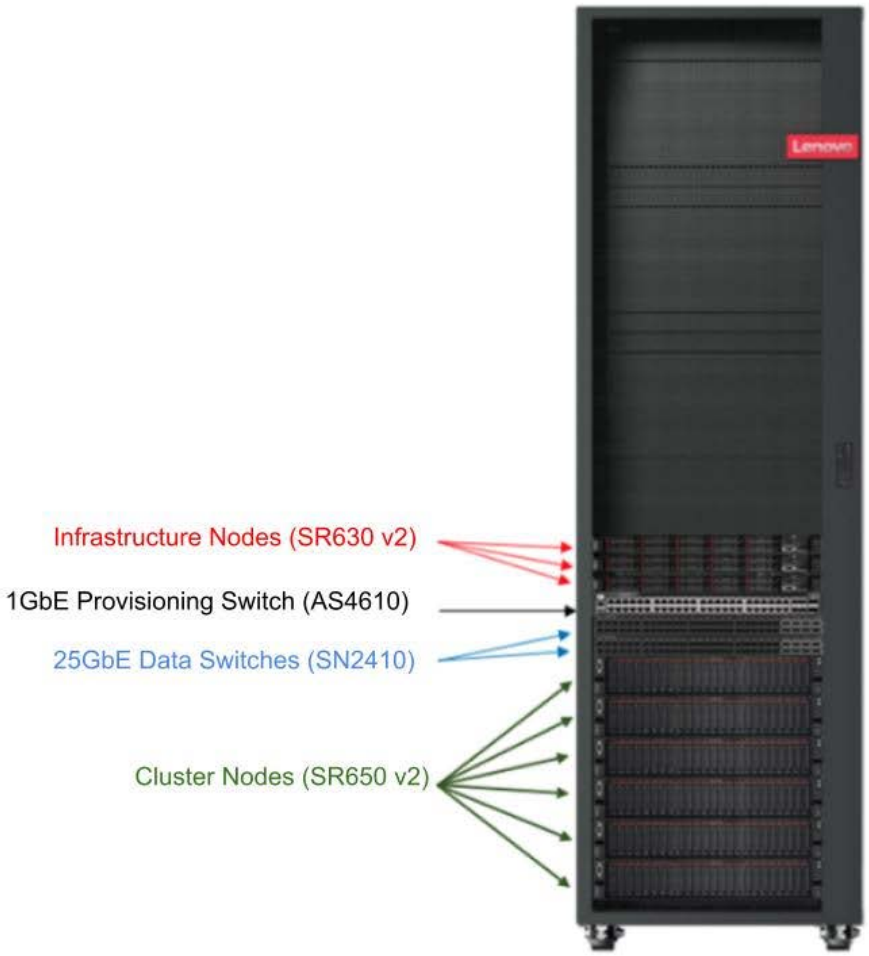
Infrastructure nodes:

Node Name	**Purpose**
Rack1-MAAS1	Infra #1 (MAAS, LMA)
Rack1-MAAS2	Infra #1 (MAAS, LMA)
Rack1-MAAS3	Infra #1 (MAAS, LMA)

Cloud nodes:

Node Name	Purpose
Rack1-cluster1	Converged node handling Kubernetes components + Storage functions
Rack1-cluster2	Converged node handling Kubernetes components + Storage functions
Rack1-cluster3	Converged node handling Kubernetes components + Storage functions
Rack1-cluster4	Converged node handling Kubernetes components + Storage functions
Rack1-cluster5	Converged node handling Kubernetes components + Storage functions
Rack1-cluster6	Converged node handling Kubernetes components + Storage functions

Figure 4: Cluster Hardware Rack Diagram



4.2.2 Server components firmware versions

The firmware versions of the components listed in Table 3 are the versions that were available at the time this Reference Architecture was developed. Ensure that the firmware on all servers, storage devices, and switches are up to date.

Table 5: Firmware versions

Component	Version
BMC	1.15
XClarity Provisioning Manager	3.09
UEFI	1.11

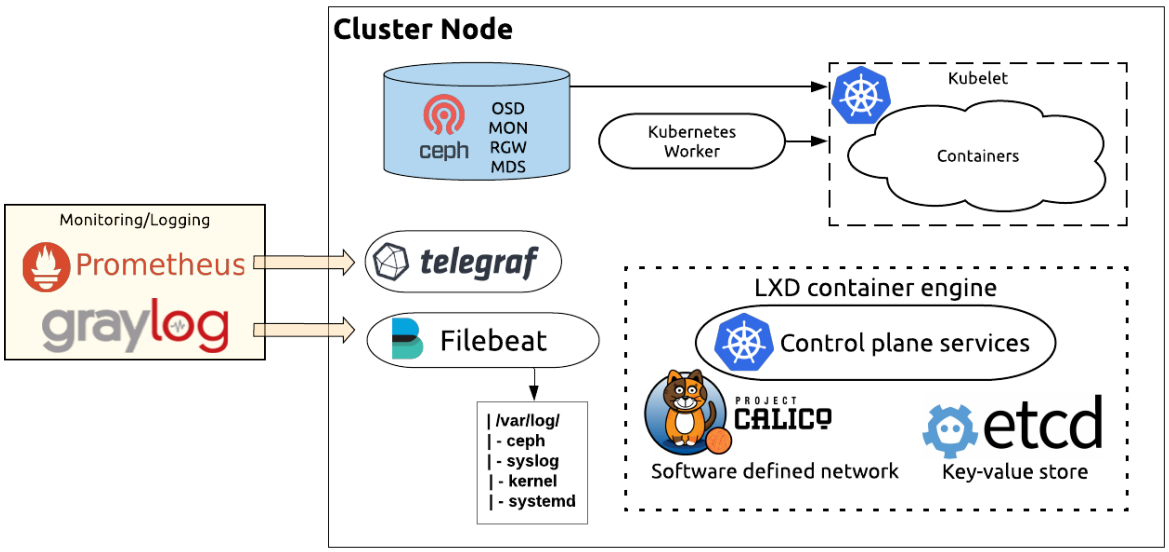
5 Component model

5.1 Charmed Kubernetes components

This chapter presents detailed information about the Kubernetes components included as charms in Charmed Kubernetes.

Below is the high-level representation of logical components in the Kubernetes cluster delivered by Canonical.

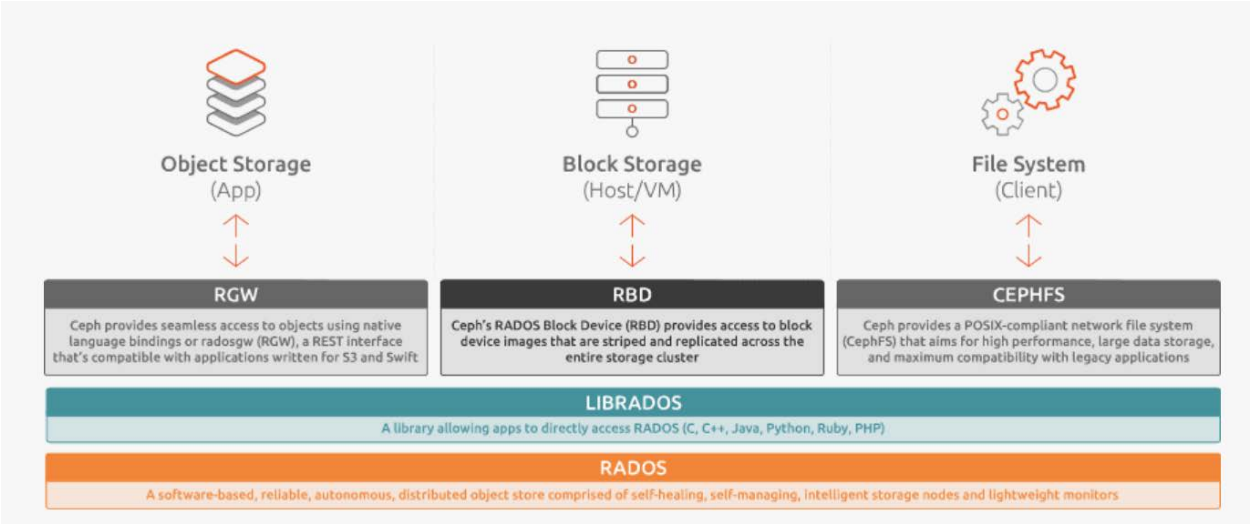
Figure 5. Charmed Kubernetes Software components



5.1.1 Storage charms

Ceph is a distributed storage and network file system designed to provide excellent performance, reliability, and scalability. Canonical uses Ceph by default for providing block/object/file storage functionality to the private cloud, however this can be replaced by, or complemented with, another storage solution such as dedicated storage appliances.

Figure 6. Ceph logical architecture



The following charms allow to deploy and manage production-grade Ceph cluster.

ceph-monitor

Ceph monitors are the endpoints of the storage cluster and store the map of the data placement across Ceph OSDs.

By default Ceph cluster does not bootstrap until three service units have been deployed and started. This is to ensure that a quorum is achieved prior to adding storage devices.

After the initialization of the monitor cluster a quorum forms quickly, and OSD bring-up proceeds.

ceph-osd

Ceph OSDs are managing underlay storage devices that contain user's data and represent the capacity of the cluster.

This charm provides the Ceph OSD personality for expanding storage capacity within a Ceph deployment. It manages the entire set of block devices on the physical node, allowing to add/remove block devices if needed, as well as change their parameters on the fly.

ceph-radosgateway

This charm provides an API endpoint for Swift or S3 clients, supporting OpenStack Keystone-based RBAC and storing objects in the Ceph cluster underneath.

ceph-fs

Deploys the metadata server daemon (MDS) for the Ceph distributed file system (CephFS). This allows to dynamically create file systems on top of Ceph block storage and mount remote file system on the client with help of the kernel driver.

5.1.2 Kubernetes charms

EasyRSA

EasyRSA is the PKI used by Charmed Kubernetes to build and deploy x509 certificates used to secure the communication between the various layers of the solution: etcd cluster, Kubernetes API and others.

It is a simple CLI tool that Juju leverages via SSH implemented over the charm relation protocol. Charms consuming certificates can query the relation to get access to the CA cert, or ask for the creation of separate server or client certificates.

Optionally this functionality can be extended by Hashicorp Vault component that is used for securely storing the certificates.

Kubernetes-master

The Kubernetes Master is in-charge of managing the Kubernetes cluster. It is made of 3 underlying components:

- The API Server is in-charge of being the interface between the administrators and users of the cluster and the cluster itself, but also provides services within the cluster
- The Scheduler is in-charge of allocating pods to their nodes
- The Controller Manager is in-charge of maintaining the cluster in a desired state

All these applications are stateless and can be scaled in and out very easily. The state of the cluster is saved in the etcd database.

Kubernetes-worker

The Worker is the component that runs the compute tasks in Kubernetes. It is based on 2 core elements:

- The kubelet is the Kubernetes Agent that executes the plan and drives container runtime and manipulates the core objects of the Kubernetes API
- The kube-proxy is a service that drives iptables to make sure that the translation between services and pods is efficiently rendered.

Etcd

Etcd is a key-value store, that is acting as the control plane database. It is used to store the state of the cluster, capturing details such as deployment status of containers, pod metrics and logs, network configuration, cluster certificates etc.

Calico

Calico is a virtual network that gives a subnet to each host for use with container runtimes.

This charm will deploy Calico as a background service, and configure CNI for use with Calico, on any principal charm that implements the kubernetes-cni interface.

Docker/Containerd In Charmed Kubernetes, the container runtime is managed by a charm that is separate to the Kubernetes Worker. Charmed Kubernetes supports and has different Juju chams for Docker, containerd and Kata container runtimes.

5.1.3 Resource charms

This topic describes the resource charms used by Charmed Kubernetes.

API Load Balancer The API load balancer is a simple nginx service that exposes a proxied endpoint to the Kubernetes API. It converts the API default port 6443 to run on the more classic 443.

HAcluster There is an option available for defining the Virtual IP in order to provide highly available access to Kubernetes API load balancer.

To use virtual IP address(es) the clustered nodes must be on the same subnet, such that:

- The VIP is a valid IP address on the subnet for one of the node's interfaces;
- Each node has an interface in said subnet.

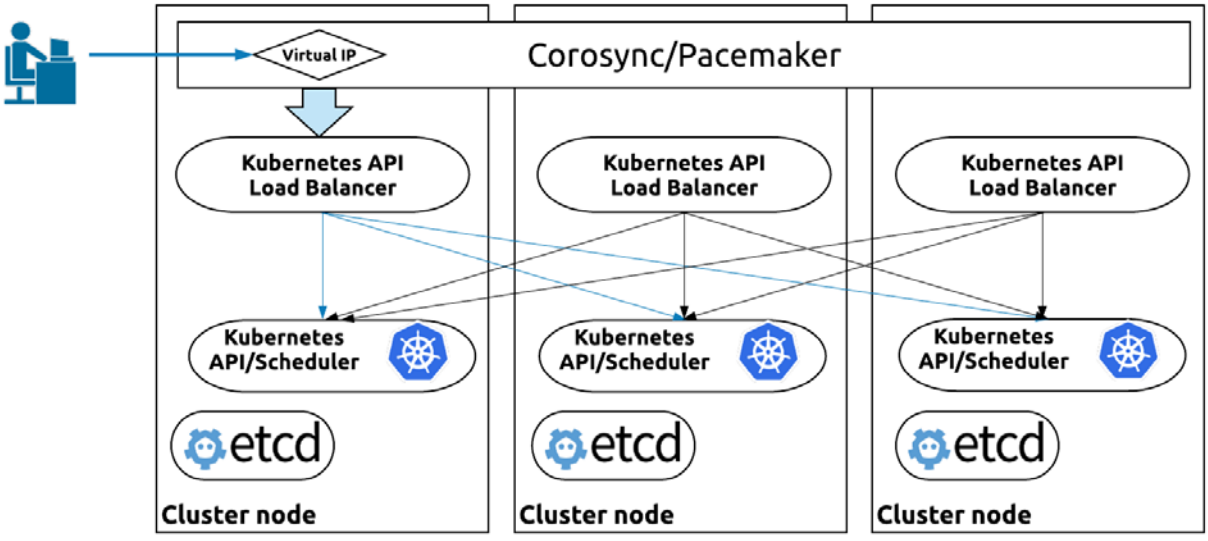
The VIP becomes a highly-available API endpoint.

In this architecture, the Kubernetes control plane is deployed as two or three units onto different physical hosts with their own IP addresses.

Kubernetes-API load balancer is also deployed as three units, while one of them will have a Virtual IP associated with it, providing the endpoint to Kubernetes API and balancing the traffic between available units of Kubernetes-master.

The following diagram explains HA of Kubernetes-API in Charmed Kubernetes:

Figure 7. Kubernetes HA Architecture



5.1.4 Network space support

Kubernetes charms support the use of Juju Network Spaces, allowing the charm to be bound to network space configurations managed directly by Juju. API endpoints can be bound to distinct network spaces supporting the network separation of all existing endpoints.

Network spaces are accordingly mapped to different VLANs managed by MAAS, making networking management transparent and flexible.

5.1.5 Monitoring and Logging tools

This chapter describes the services which have been deployed to manage and monitor your Kubernetes cluster. Canonical has a specially designed architecture for their customers to manage and monitor Kubernetes clusters. Those services are an optional part of Canonical Kubernetes Discoverer services. If customers have different requirements as part of Canonical Kubernetes Discoverer, we design the architecture for customers and do the deployments.

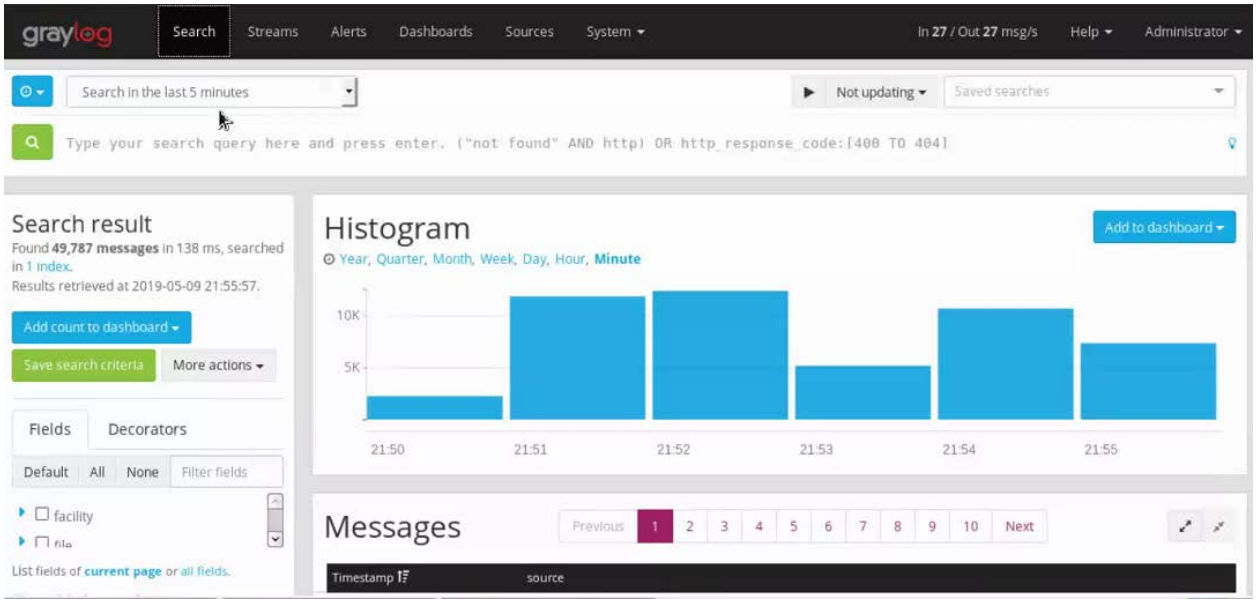
5.2 Cluster logging tools

5.2.1 Graylog

Graylog is the solution for aggregating and managing the logs from various components of the cluster, as well as for providing visualization of the logs.

Figure 8 shows the sample of Graylog aggregation dashboard.

Figure 8. Graylog Dashboard



5.2.2 Elasticsearch

Elasticsearch is a distributed database used for storing indexed logs and act as the backend for Graylog.

5.2.3 Filebeat

As a log forwarder, Filebeat tails various log files on the client side and quickly sends this information to Graylog for further parsing and enrichment, or to Elasticsearch for centralized storage and analysis.

5.3 Monitoring the cluster

From an architectural standpoint, monitoring suite consists of Telegraf (host metric collection), Prometheus (monitoring server) and Grafana (monitoring dashboard).

5.3.1 Prometheus

Prometheus is a systems and services monitoring system. It collects metrics from configured targets at given intervals, evaluates rule expressions, displays the results, and can trigger alerts if some condition is observed to be true.

5.3.2 Grafana

Grafana is the leading graph and dashboard builder for visualizing time series metrics.

Figure below displays the monitoring tool dashboard.

Figure 9. Grafana Monitoring Dashboard

5.3.3 Telegraf

Telegraf is a client-side system that collects information about the status of the host services and makes them available for pulling by any monitoring solutions (in this architecture Prometheus).

6 Operational model

6.1 The node lifecycle

Each machine or “node” managed by MAAS goes through a lifecycle — from new to “enlistment”, then to “commissioned”, and in the end to “ready” state. There are also special statuses as Broken and Testing.

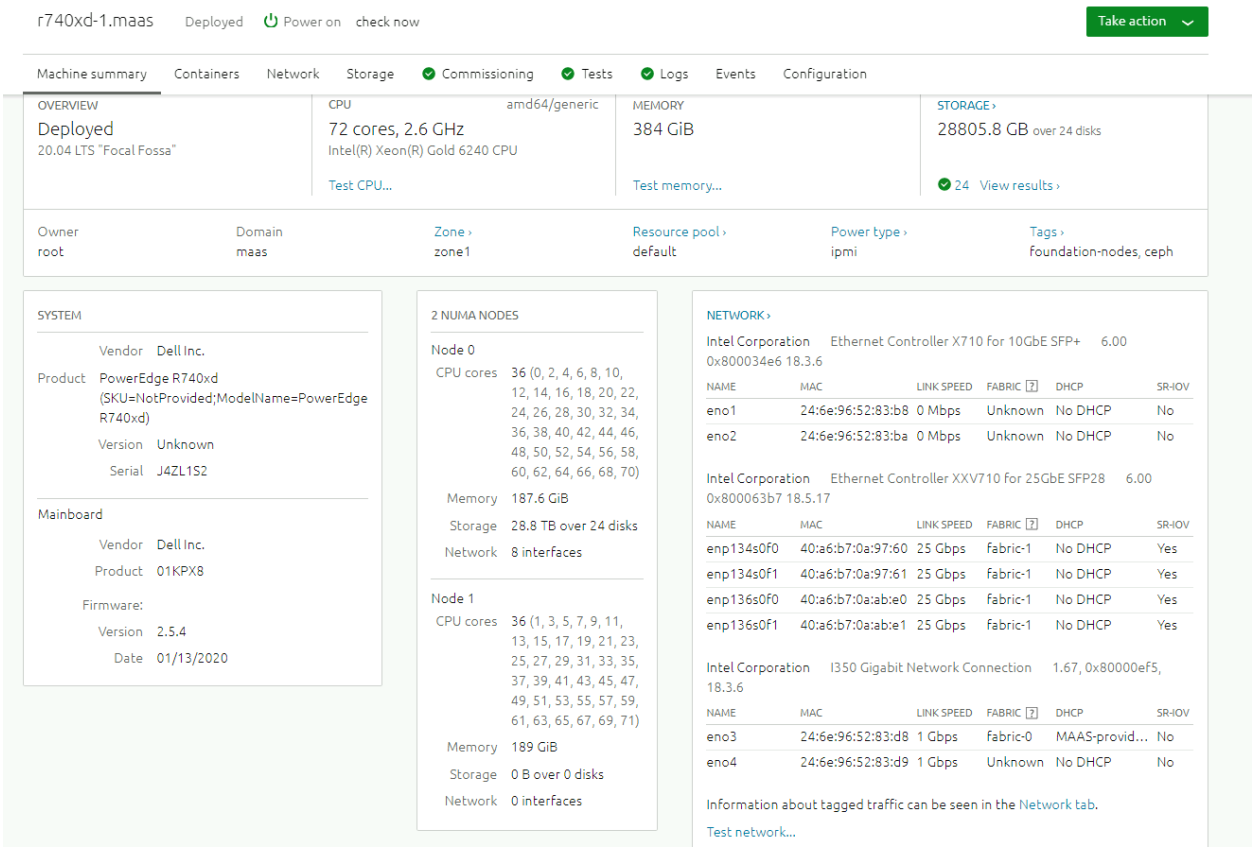
6.1.1 New

New machines that PXE-boot on a MAAS network will be enlisted automatically if MAAS can detect their BMC parameters. During the Enlistment phase MAAS will ensure that it can control the power status of the machine through its BMC. Another option is to add machines through the API by supplying BMC credentials.

6.1.2 Commissioning

In the Commissioning phase, MAAS collects all data about the machine, which includes detailed hardware inventory like CPU model, memory setup, disks, and chipsets. It also collects information about network connectivity. This information can later be used in deployments. In this phase, you can apply custom commissioning scripts that can update firmware, configure hardware RAID, etc.
Example node’s representation by MAAS at the end of commissioning will look like the following diagram.

Figure 10. Node commissioned by MAAS



6.1.3 Ready

A machine that is successfully commissioned is considered “Ready”. A “ready” machine has configured BMC credentials (on IPMI based BMCs) for ongoing power control and ensures that MAAS can start or stop the machine and allocate or redeploy it with a fresh operating system.

6.1.4 Allocated

Ready machines can be Allocated to users, who can configure network interface bonding and addressing, and disks, such as LVM, RAID, bcache or partitioning.

6.1.5 Deploying

Users can request that MAAS to turn the machine on and install a complete operating system from scratch without any manual intervention, configuring network interfaces, disk partitions, and more.

6.1.6 Releasing

When a user has finished with the machine, they can release it back to the shared pool of capacity. You can request MAAS to verify that there is a full disk-wipe of the machine when that happens.

6.2 Install MAAS

In this Reference Architecture MAAS is installed in Highly Available fashion using a set of open source tools including but not limited to: MAAS, PostgreSQL, Corosync/Pacemaker.

For detailed configuration procedure please contact Canonical representatives.

6.2.1 Configuring Hardware

MAAS requires one small server and at least one server that can be managed with a BMC. Lenovo recommends that you have the MAAS server provide DHCP and DNS on a network to which the managed machines are connected.

6.2.2 Installing Ubuntu Server

Download Ubuntu Server 20.04 LTS and follow the step-by-step installation instructions on your MAAS server.

6.2.3 MAAS Installation

This section describes the following MAAS installation topics:

• Prerequisites

• Infrastructure nodes requirements

Prerequisites

Three infrastructure nodes for fully HA, pre-installed with the latest Ubuntu 20.04-LTS, must be available to host MAAS, the Juju controllers and other runtime and monitoring tools. The nodes must have SSH access to the root user enabled through authorized_keys.

Infrastructure nodes requirements

Three infrastructure nodes must be already preinstalled, and they host multiple services intended to support building and operating the OpenStack solution, including:

• MAAS and its dependencies, including PostgreSQL

• Juju controllers

• Monitoring and alerting systems

• Log aggregation and analysis systems

Infrastructure nodes host these services either on the bare metal or in KVM virtual machines.

Infrastructure nodes must have network access to:

• The PXE and BMC networks in order to commission and provision machines.

• The various APIs which must be monitored. In order to monitor Kubernetes, the nodes must have access to the Kubernetes API.

• Externally, to the Ubuntu archives and other online services, in order to obtain images, packages, and other reference data.

To provide HA, infrastructure nodes must:

• Be placed in separate hardware availability zones

• MAAS has a concept zone where server hardware can be placed into different rack and each rack can be placed in single zone. Or within same rack hardware can be divided based on the power redundancy or the slots within rack. It would be helpful to place different services in different hardware zone.

• Have bonded network interfaces in order to provide resiliency from switch or NIC failures.

• Have the MTU on the bonded interfaces set to 9000B (jumbo frames).

• Have a bridge (`broam`) interface active which has the primary bond (typically `bond0`) as its only member. The bridge inherits the MTU of the underlying device, so there is no need to set its MTU explicitly.

6.3 MAAS initial configurations

This section describes the following MAAS initial configurations:

- MAAS credentials
- Enlist and commission servers

6.3.1 MAAS Credentials

For initial installation of MAAS follow official procedure.

All Region controllers should point to the Virtual IP of PostgreSQL database. More info on MAAS HA configuration can be found in MAAS documentation

After the packages installation it is required to create a set of credentials for admin user with **“maas init”** command.

6.3.2 Enlist and commission servers

Now MAAS is ready to enlist and commission machines. To perform that task:

1 Set all the servers to PXE boot from the first 10Gbe network interface.

2 Boot each machine once. You should see these machines appear in MAAS.

3 Select all of the machines and commission them by clicking on the Take action button.

4 When machines have a Ready status you can deploy the services.

6.3.3 Set up MAAS KVM pods

Once MAAS is completely set up, all infrastructure nodes should be turned into KVM hosts managed by MAAS. Once done, MAAS will be able to dynamically provision Virtual Machines on the nodes and present them as available servers to the users. Follow the guide for turning Infrastructure nodes into KVM Pods and creating a set of VMs for setting up Juju controllers and LMA components.

6.4 Juju components

For an overview of Juju, refer to the Juju modelling tool section above. This section discusses the working of different components of Juju.

6.4.1 Juju controller - the heart of Juju

The Juju controller manages all the machines in the running models and responds to the events that are triggered throughout the system. It also manages scale-out, configuration, and placement of all models and applications.

Juju controller has to be located in the same physical segment of the network Kubernetes cluster, be able to execute calls to the MAAS API and connect to Kubernetes Cluster nodes.

Highly available distributed Juju controller is supposed to be created using a set of KVM Virtual machines mentioned in the previous steps.

6.4.2 Charms

Charms are a collection of scripts that contain all of the operations necessary to deploy, configure, scale, and maintain cloud applications with Juju. Charms encapsulate a single application and all the code and know-how that it takes to operate it, such us how to combine and work with other related applications, or how to upgrade it.

Charms also allow a hierarchy, with subordinate charms to complement a main service.

6.4.3 Bundles

Bundles are ready-to-run collections of applications that are modelled to work together and can include particular configurations and relations between the software to be deployed.

Bundles may also be optimized for different deployment scenarios of the same software. For example, a scale-out, production-ready version like the Canonical Distribution of Kubernetes, or a development-friendly test version like Kubernetes Core.

Bundles perform the following functions:

• Install

• Configure

• Connect

• Upgrade and update

• Scale-out and scale-back

• Perform health checks

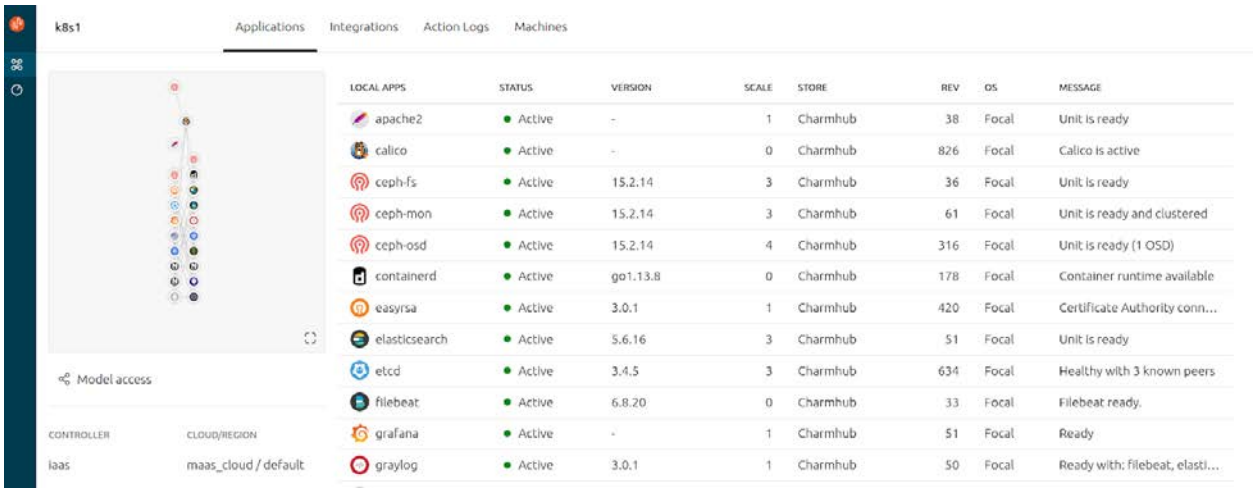
• Undertake operational actions

• Benchmark

Juju supports UI representation of deployed bundle and allow to dynamically manipulate with cluster's configuration options and layout prior to the bundle deployment and during the lifetime.

Figure below shows modelling with Juju.

Figure 11. Juju modelling



6.4.4 Provision

Specify the number of machines you want and how you want them to be deployed or let Juju do it automatically.

6.4.5 Deploy

Deploy your services, or (re)deploy your entire application infrastructure to another cloud, with a few clicks of your mouse.

6.4.6 Monitor and manage

The Juju controller manages:

• Multiple models

• All VMs in all your running models

• Scale out, configure and placement

• User accounts and identification

• Sharing and access

6.4.7 Comparing Juju to other configuration management tools

Juju provides a higher level of abstraction, and supplies the tools needed to manage the full scope of operations beyond deployment and configuration management, regardless of the machine on which it runs.

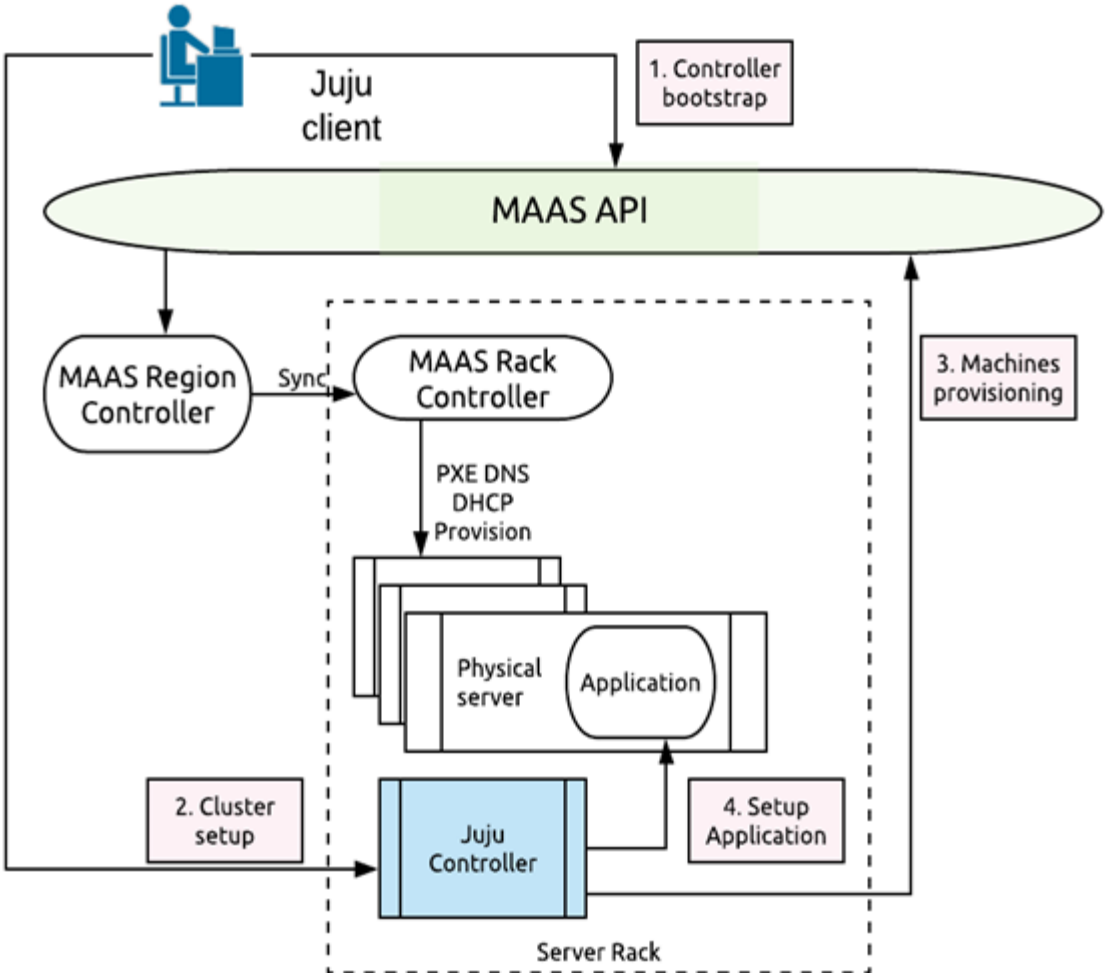
One of the main advantages of Juju is the dynamic configuration ability, which enables you to:

• Reconfigure services on the fly.

• Add, remove, or change relationships between services.

• Scale in or out with ease, shares the operational knowledge and makes the most of the wider community.

Figure 12. Juju Configuration with MAAS



6.5 Monitoring

Charmed Kubernetes solution includes a monitoring suite based on the best open source tools available.

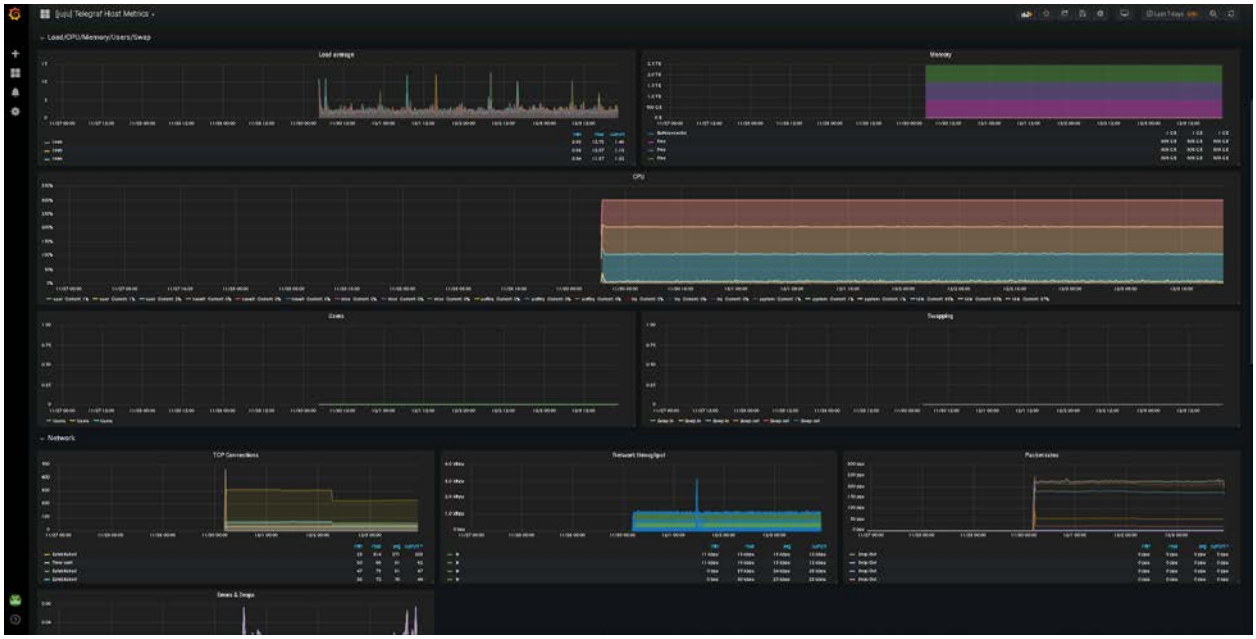
6.5.1 Observability Tools

The Canonical monitoring suite retrieves information from the Kubernetes components and infrastructure monitors, and combines it in a configurable portal, giving the customer visibility to all the different metrics.

The portal aggregates the relevant information from an operational perspective, and differentiates various components, such as compute, network, and storage.

The Canonical observability tool allows both customers and operators to zoom in to on the details of any of the higher-level graphs to obtain further information. The portal also includes an efficient time series database that allows tracking of the evolution of the cloud metrics and health status over time.

Figure 13. Canonical Observability Tool



6.6 Log Aggregation

The solution also implements the Graylog suite for log aggregation, which makes it easy for customers to have visibility on the different logs from their cloud services without accessing them directly.

These services are integrated with Charmed Kubernetes solution as part of the charms, fulfilling the same requirements around upgradeability and operation.

7 Deployment considerations

Machine Learning platforms

The following section describes platforms that can be utilized with the Reference Architecture for deploying AI-based model development and training workloads.

7.1 LiCO as a machine learning/deep learning platform

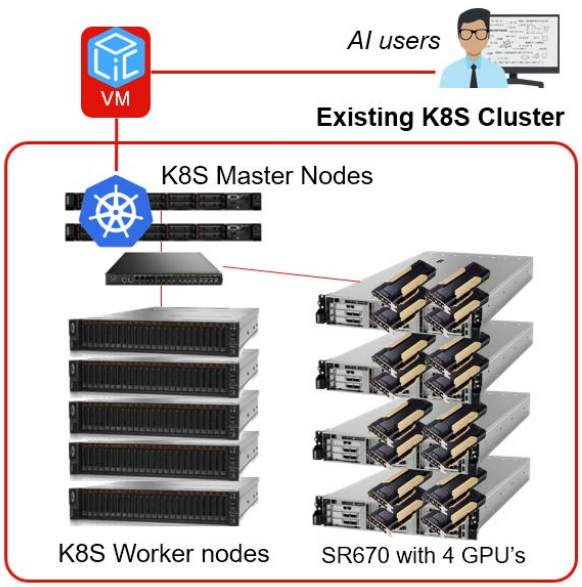
Lenovo Intelligent Computing Orchestration (LiCO) K8S/AI is a software solution developed and supported by Lenovo to facilitate ML/DL model development and production training on a Kubernetes cluster. LiCO K8S/AI provides users an abstracted browser-based interface to deploy and manage their AI workloads on the cluster through a single pane-of-glass, with minimal inputs required to deploy.

At a high level LiCO K8S/AI provides users the following functionalities:

- AI Framework templates allowing the user to specify the containerized framework version and compute resources needed to deploy with python or shell scripts.
- Lenovo Accelerated AI templates with pre-defined models for popular use-cases, allowing users to quickly train and test labelled data.
- LiCO Workflow to define a series of job submissions to automate multi-step processes
- AI Studio workflow for end-to-end image labelling, tuning, training and inference for Image Classification, Instance Segmentation and Object Detection use cases.
- Easy creation, management and deployment of multiple Jupyter notebook environments from the LiCO interface on the Kubernetes cluster, that can be used directly from the web browser.
- Common templates and user-creatable custom templates to accommodate additional workload deployments to the Kubernetes cluster.
- Integrated graphical drag-and-drop access to cluster PVC storage space to facilitate easy transfer, management and inspection of files and directories.

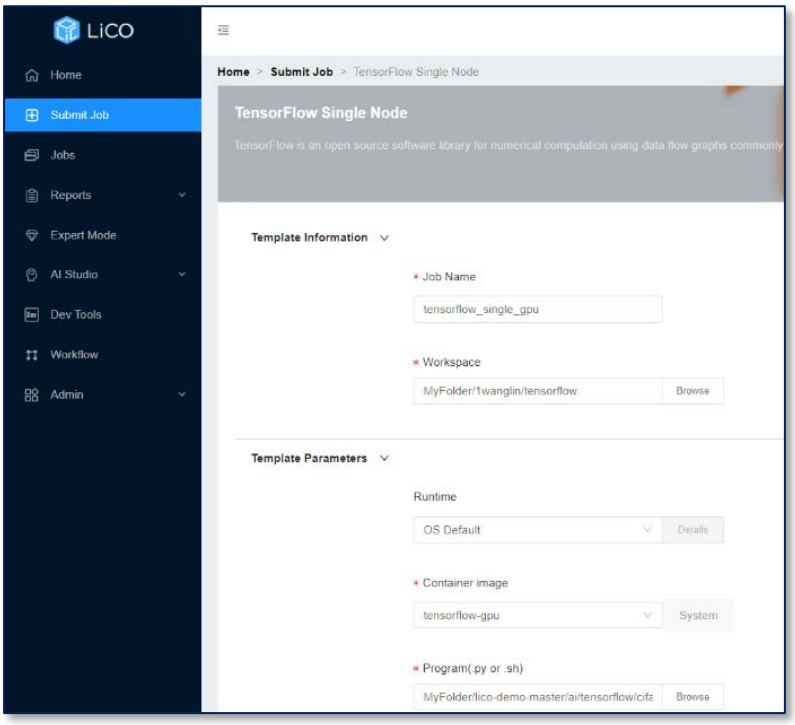
LiCO K8S/AI is deployed outside the Kubernetes cluster to maintain persistence and interfaces with an existing cluster’s master nodes via the Kubernetes API. The implementation can be on a bare metal or virtual machine as shown in Figure 13.

Figure 14: LiCO implementation design



LiCO creates the relevant YAML files needed to deploy workloads based on the inputs the user provides in the job submission template as shown in Figure 14.

Figure 15: LiCO Workload deployment YAML



For more information on the capabilities of LiCO, reference the Lenovo Intelligent Computing Orchestration (LiCO) Product Guide https://lenovopress.com/lp0858-lenovo-intelligent-computing-orchestration-lico.

7.2 Kubeflow as a machine learning platform

Kubeflow is a composition of open source tools for developing, orchestrating, deploying, and running scalable and portable machine learning workloads on top of Kubernetes.

At the high level Kubeflow is comprised of the following components and functionalities:

- Jupyter notebooks - an open-source application that allows users to blend code, equation-style notation, free text and dynamic visualizations to give data scientists a single point of access to their experiment setup and notes.
- Katib (hyper-parameter tuning) - Hyper-parameters are set before the machine learning process takes place. These parameters (e.g. topology or number of layers in a neural network) can be tuned with Katib. Katib supports various AI/ML tools such as TensorFlow, PyTorch and MXNet, making it easy to reuse previous experiments results.
- Pipelines - facilitate end-to-end orchestration of ML workflows, management of multiple experiments and approaches, as well as easier re-use of previously successful solutions into a new workflow. This helps developers and data scientists save time and effort.
- Serving - Kubeflow makes two service systems available: TensorFlowServing and Seldon Core. These allow multi-framework model serving and the choice should be made based on the needs of each project.
- Training - model training is possible with various frameworks, in particular TensorFlow, XGBoost, MXNet and PyTorch. These cover the most popular frameworks in the data science and AI/ML space, ensuring that developers are able to use Kubeflow's MLOps features with their favourite tools.

Utilizing Kubernetes as the infrastructure for Kubeflow installation unlocks the possibility for dynamic scaling of platform components, as well as addition of new or upgrading existing components at any time.

7.2.1 Kubeflow and charms

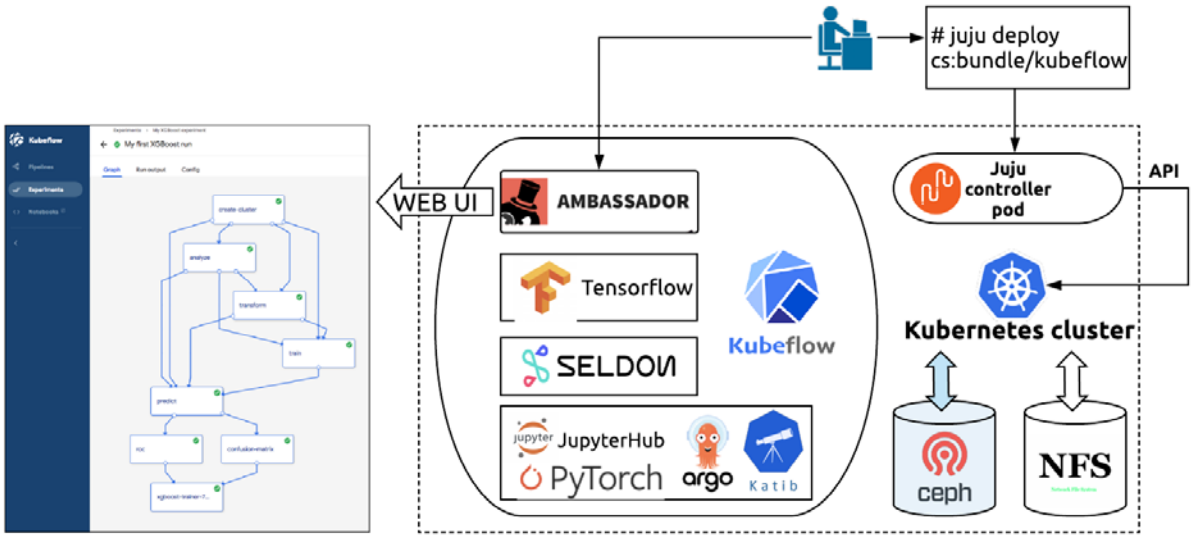
Canonical delivers Kubeflow components in an automated fashion, using the same approach and toolset as for deploying the infrastructure and Kubernetes cluster - with help of Juju.

The current version of Juju is able to register an existing Kubernetes cluster as the target and deploy software on top of that using operator approach with help of Juju Kubernetes charms.

It either utilizes the existing Juju controller that is used for deploying the infrastructure, or bootstraps new controller on top of Kubernetes as the pod. In this reference architecture, the latter approach is used.

Once deployed, Juju registers the model of the Kubeflow cluster that describes the desired components layout, configuration options if necessary, optionally the details of images and scaling options.

Figure 16: Deploying Juju model for Kubeflow cluster



The end user experience when deploying and operating the deployment on top of bare metal infrastructure and Kubernetes is identical, however the same charms cannot be used for both deployment substrates.

Canonical has developed and tested the set of charms automating the delivery of the most demanded components of Kubeflow, providing the reference bundle that can be re-used when deploying to a Kubernetes cluster.

7.3 Server / Compute Nodes

7.3.1 Lenovo ThinkSystem SR630 v2 server specifications

Table 6: **Lenovo ThinkSystem SR630 v2 server specifications for Kubernetes Infrastructure nodes**

Component type	Component Description	Quantity
Processor	Intel Xeon Gold 6330 28C 205W 2.0GHz Processor	2
Memory	ThinkSystem 16GB TruDDR4 3200 MHz (2Rx8 1.2V) RDIMM	16
Drive controller	ThinkSystem RAID 940-16i 8GB Flash PCIe 12Gb Adapter	1
External Network card	ThinkSystem Mellanox ConnectX-4 Lx 10/25GbE SFP28 2-port PCIe Ethernet Adapter	1

Boot/Data drives	ThinkSystem 2.5" PM893 7.68TB Read Intensive SATA 6Gb HS SSD	2

7.3.2 Lenovo ThinkSystem SR650 v2 server specifications

Table 7: **Lenovo ThinkSystem SR650 v2 server specifications for converged Kubernetes components + Storage**

Component type	Component Description	Quantity
Processor	Intel Xeon Gold 6330 28C 205W 2.0GHz Processor	2
Memory	ThinkSystem 32GB TruDDR4 3200 MHz (2Rx8 1.2V) RDIMM	16
Drive controller	ThinkSystem RAID 940-16i 8GB Flash PCIe 12Gb Adapter	1
External Network card	ThinkSystem Mellanox ConnectX-4 Lx 10/25GbE SFP28 2-port PCIe Ethernet Adapter	1
Boot system	ThinkSystem M.2 with Mirroring Enablement Kit (configured as RAID1) with 2x ThinkSystem M.2 240GB SATA 6Gbps Non-Hot Swap SSD	1
Data drives	12TB 7.2K SAS 12Gb Hot Swap 512e 3.5in Hot-plug Hard Drive	6
GPU (optional)	Nvidia A100	2

7.3.3 Hardware Configuration Notes

The Lenovo ThinkSystem configurations are used with 25GbE networking. To ensure that the network is HA ready the network card offering 2 x 25GbE ports, is required for each node.

The ThinkSystem servers need to be configured for the Lenovo Charmed Kubernetes solution. Following are the configurations that need to be taken care of:

- BIOS

- XClarity Controller
- RAID
- Network

Verify that the physical, and virtual disks are in ready state, and that the virtual disks are auto-configured to RAID-0. The IPMI over LAN option must be enabled in each ThinkSystem SR650 v2 server through the BIOS.

For detailed hardware configurations of the ThinkSystem solution for the Charmed OpenStack platform, consult a Lenovo sales and services representative.

Caution: Please ensure that the firmware on hardware is up to date or match the versions from the table above.

7.4 Networking

A Lenovo ThinkSystem solution is agnostic to the top of rack (ToR) switch a customer may choose. For management network role reference implementation in this document uses the Mellanox AS4610-54T switch. Two of the Mellanox SN2410 switches are used at the leaf-layer of the standard leaf-spine topology. The two switches are used to implement high availability on the data network. A pair of switches of similar or better capacity may be added at the spine-layer of the topology, if desired.

7.4.1 Mellanox SN2410 25 GbE Switch

The Mellanox SN2410 is an ultra-low-latency 10/25/40 GbE top-of-rack (ToR) switch built for applications in high performance data center and computing environments. Leveraging a non-blocking switching architecture, the SN2410 delivers line-rate L2 and L3 forwarding capacity with ultra-low-latency to maximize network performance.

Table 8: **10GbE Switch Specification**

Variable	Description
SFP28 ports	48 x 25GbE SFP28 ports
QSFP28 ports	8 x 100GbE QSFP28 ports
RJ45 ports	1 x Console/Management Port
Operating System	Cumulus NOS

Refer to the Mellanox SN2410 specification sheet for more information.

7.4.2 Mellanox AS4610 1 GbE Switch

The Mellanox AS4610 1GbE switch delivers 48 ports of wire-speed, Gigabit Ethernet. With 48 built-in copper Gigabit Ethernet ports in a 1U form factor, the switches offer flexibility and capacity for growth.

Table 9: **1GbE Switch Specification**

Variable	Description
SFP+ ports	4 x 10GbE SFP+ ports
QSFP+ ports	2 x 20GbE QFSP+ stacking ports
RJ45 ports	48 x 1 Gigabit Ethernet Ports
Operating System	Cumulus NOS

Refer to the Mellanox AS4610 specification sheet for more information.

7.4.3 Network Infrastructure layout

The network consists of the following major network infrastructure layouts:

- Data network infrastructure: The server NICs and the leaf switch pair. The leaf switches are connected to the data center user networks and carry the main service traffic in / out of the reference architecture.
- Management network infrastructure: The BMC management network, which consists of IPMI ports and the OOB management ports of the switches, are aggregated into a 1-rack unit (RU) Mellanox AS4610 1GbE switch. This 1-RU switch in turn can connect to the data center management network.
- MAAS Services: The MAAS Rack Controllers (see below) provide DHCP, IPMI, PXE, TFTP and other local services on the provisioning and IPMI network. Ensure that the MAAS DCHP server is isolated from the data center DHCP server.

Network components

The following component blocks make up this network:

• Server nodes

• Leaf switches and networks

• VLANs

• Out-of-Band Management switch and network

Server nodes

To create a highly available solution, the network must be resilient to the loss of a single network switch, network interface card (NIC) or bad cable. To achieve this, the network configuration uses channel bonding across the servers and switches.

There are several types (or modes) of channel bonding, however only one is recommended and supported for this solution:

- 802.3ad or LACP (mode=4)

The endpoints for all nodes are terminated to switch ports that have been configured for LACP bonding mode,

across two Mellanox SN2410's configured with LAG across them. For details regarding network configuration on the servers, please contact your Lenovo services and sales representative.

Table 10: **Recommended channel bonding modes**

Node type	Channel Bonding type
Infrastructure nodes	802.3ad (LACP mode 4, channel fast)
Cluster nodes	802.3ad (LACP mode 4, channel fast)

Multiple bonds may be created on the servers for separating critical types of traffic from each other and allocate them on different physical interfaces.

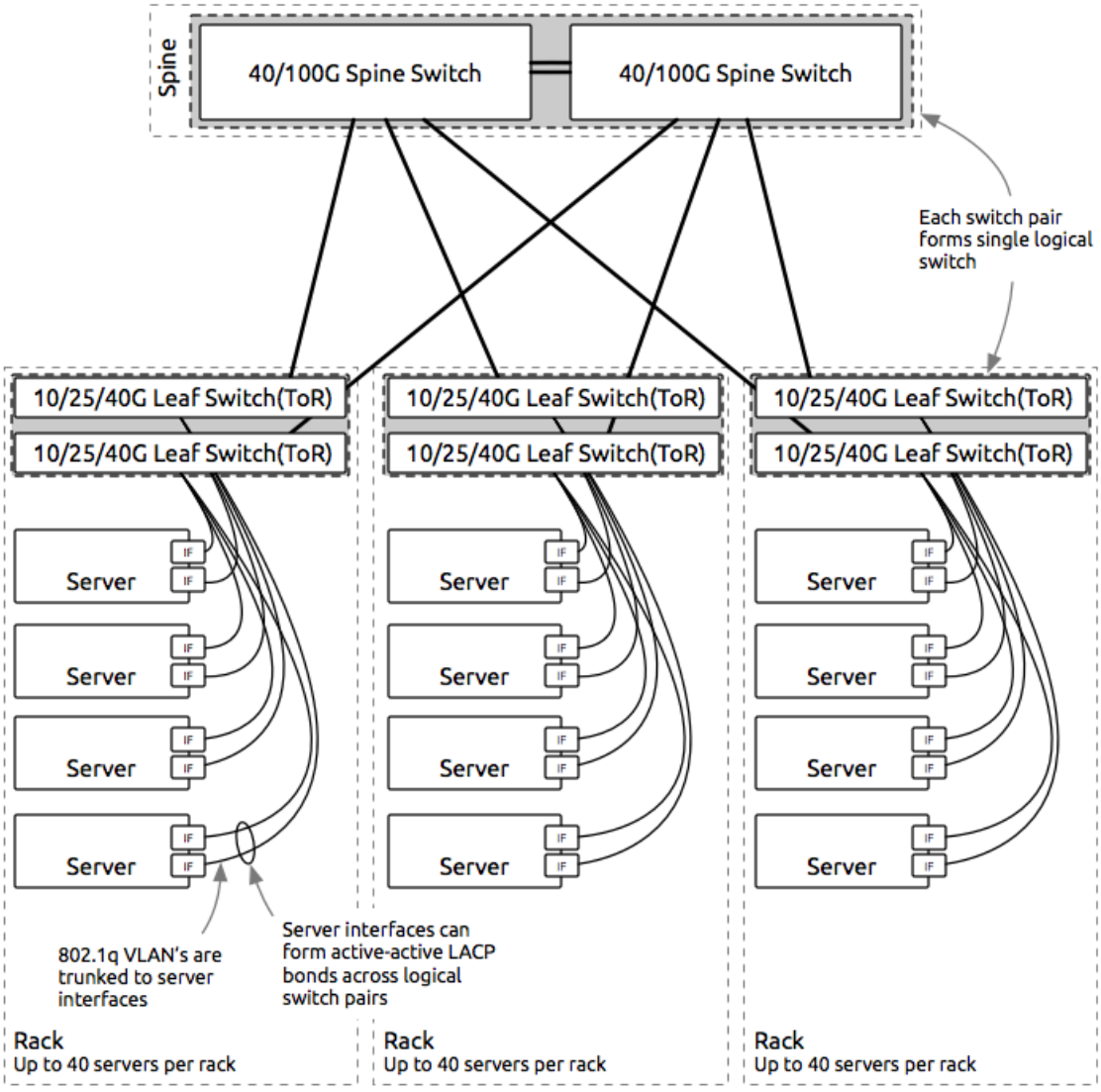
Actual layout depends on the particular cluster configuration and is out of scope of the Reference Architecture.

Leaf switches

This reference implementation uses two Mellanox SN2410 switches. There is a redundant physical 2x 40GbE connection between the two switches. The recommended architecture uses LAG between the switches in the leaf pair.

Figure 16 illustrates the sample physical connections, representing bonding setup of servers' interfaces and switches LAG setup:

Figure 17: Network Architecture



VLANs

This reference architecture implements at a minimum four separate networks through Layer-2 VLANs. Multiple networks below can be combined into single subnet based on end user requirements.

Table 11: Network traffic types

VLAN	Description
OOB Management	Used for the IPMI network.
Internal	Used for cluster provisioning, monitoring and management
External	Used for communication between cluster components, as well as external access to the

	workloads, also for consuming persistent storage resources by the workloads.
Storage (cluster)	Used for replicating persistent storage data between units of Ceph.

Out-of-Band management network

The Management network of all the servers is aggregated into the Mellanox AS4610 1GbE switch in the reference architecture. One interface on the Out-of-Band (OOB) switch provides an uplink to a router/jumphost.

The OOB management network is used for several functions:

• The highly available software uses it to reboot and partition servers.

• When an uplink to a router is added and the XClarity controllers are configured to use it as a gateway, there are tools for monitoring the servers and gather metrics on them. Discussion of this topic is beyond the scope of this document. Contact your Lenovo sales representative for additional information.

7.5 Performance considerations

The systems validated in this Reference Architecture are suitable for many Machine Learning and Deep Learning deployments. Accelerators can be beneficial for Deep Learning development and production training workloads, especially when leveraging frameworks and toolkits that are optimized for acceleration.

Where acceleration is desired, accelerator cards such as NVIDIA A100 or T4 may be used in the ThinkSystem SR650 v2 for those prescribed in the Reference Architecture as Cluster Nodes. Lenovo also offers additional accelerator-enabled systems such as the ThinkSystem SR670 v2 that may be more suited to unique data center requirements.

8 Appendix: Lenovo Bill of materials

This appendix contains the bill of materials (BOMs) for different configurations of hardware for Canonical Charmed Kubernetes on Lenovo ThinkSystem Servers for AI Development deployments. There are sections for compute/storage servers, management servers, and networking.

The BOM lists in this appendix are not meant to be exhaustive and must always be double-checked with the configuration tools. Any discussion of pricing, support, and maintenance options is outside the scope of this document.

8.1 BOM for compute/storage servers

8.1.1 Lenovo ThinkSystem SR650 v2

Part	Description	Qty
7Z73CTO1WW	Server : ThinkSystem SR650 v2 - 3yr Warranty	1
BH8G	ThinkSystem 2U 3.5" Chassis with 8 or 12 Bays	1
BFYE	Operating mode selection for: "Efficiency - Favoring Performance Mode"	1
BB3H	Intel Xeon Gold 6330 28C 205W 2.0GHz Processor	2
B965	ThinkSystem 32GB TruDDR4 3200 MHz (2Rx8 1.2V) RDIMM	16
5977	Select Storage devices - no configured RAID required	1
BFY5	ThinkSystem RAID 530-16i PCIe 12Gb Adapter	1
B117	ThinkSystem 3.5" 12TB 7.2K SAS 12Gb Hot Swap 512e HDD	6
BCFR	ThinkSystem 2.5" U.2 P5600 3.2TB Mixed Use NVMe PCIe 4.0 x4 HS SSD	2
B5XJ	ThinkSystem M.2 SATA/NVMe 2-Bay Enablement Kit	1
B8HS	ThinkSystem M.2 5300 240GB SATA 6Gbps Non-Hot Swap SSD	2
B8LT	ThinkSystem 2U 12x3.5" SAS/SATA Backplane	1
BDY7	ThinkSystem 2U 4x2.5" Middle NVMe Backplane	2
B5SS	ThinkSystem Broadcom 57416 10GBASE-T 2-port + 5720 1GbE 2-port OCP Ethernet Adapter	1
B653	ThinkSystem Mellanox ConnectX-4 Lx 10/25GbE SFP28 2-port PCIe Ethernet Adapter	1
B8LJ	ThinkSystem 2U PCIe Gen4 x16/x8/x8 Riser 1 or 2	1
B8LJ	ThinkSystem 2U PCIe Gen4 x16/x8/x8 Riser 1 or 2	1
B8QB	ThinkSystem V2 1800W (230V) Platinum Hot-Swap Power Supply	2
6311	2.8m, 10A/100-250V, C13 to IEC 320-C14 Rack Power Cable	2
BH8E	ThinkSystem 2U Performance Fan	6
B8LA	ThinkSystem Toolless Slide Rail Kit v2	1
B8ME	ThinkSystem 2U EIA Latch w/ VGA and External Diagnostics Port Upgrade Kit	1
B0MK	Enable TPM 2.0	1
B7XZ	Disable IPMI-over-LAN	1

8.2 BOM for infrastructure servers

8.2.1 Lenovo ThinkSystem SR630 v2

Part	Description	Qty
7Z71CTO1WW	Server : ThinkSystem SR630 v2 - 3yr Warranty	1
BH9Q	ThinkSystem 1U 2.5" Chassis with 8 or 10 Bays	1
BFYE	Operating mode selection for: "Efficiency - Favoring Performance Mode"	1
BB3H	Intel Xeon Gold 6330 28C 205W 2.0GHz Processor	2
B963	ThinkSystem 16GB TruDDR4 3200 MHz (2Rx8 1.2V) RDIMM	16
5977	Select Storage devices - no configured RAID required	1
BDY4	ThinkSystem Raid 940-16i 8GB Flash PCIe Gen4 12Gb Adapter for U.3	1
BM87	ThinkSystem 2.5" PM893 7.68TB Read Intensive SATA 6Gb HS SSD	2
BB3T	ThinkSystem 1U 10x2.5" AnyBay Backplane	1
B8N2	ThinkSystem 1U PCIe Gen4 x16/x16 Riser 1	1
B8NC	ThinkSystem 1U LP+LP BF Riser Cage Riser 1	1
B5SS	ThinkSystem Broadcom 57416 10GBASE-T 2-port + 5720 1GbE 2-port OCP Ethernet Adapter	1
B653	ThinkSystem Mellanox ConnectX-4 Lx 10/25GbE SFP28 2-port PCIe Ethernet Adapter	1
BHS9	ThinkSystem 1100W (230V/115V) v2 Platinum Hot-Swap Power Supply	2
A3SY	2.8m, 16A/100-250V, 2 Short C13s to Long C20 Rack Power Cable	1
BH9M	ThinkSystem 1U Performance Fan Option Kit	8
B8LA	ThinkSystem Toolless Slide Rail Kit v2	1
BA2Y	ThinkSystem 1U Front VGA Cable	1
B0MK	Enable TPM 2.0	1
B7XZ	Disable IPMI-over-LAN	1

Resources

- Lenovo ThinkSystem SR630 v2
  https://lenovopress.com/lp1391-thinksystem-sr630-v2-server
- Lenovo ThinkSystem SR650 v2
  https://lenovopress.com/lp1392-thinksystem-sr650-v2-server
- Lenovo Intelligent Computing Orchestration (LiCO)
  https://lenovopress.com/lp0858-lenovo-intelligent-computing-orchestration-lico
- Canonical's Ubuntu
  https://ubuntu.com
- Charmed Kubernetes
  https://ubuntu.com/kubernetes
  Kubernetes deployment services
  datasheet
- MAAS documentation
  https://maas.io
- Juju documentation
  https://juju.is/docs
  Charmed Kubeflow
  https://ubuntu.com/ai
- Ubuntu Advantage Support
  https://ubuntu.com/support
- NVIDIA A100
  https://www.nvidia.com/en-us/data-center/a100/
- Mellanox SN2410
  https://www.mellanox.com/related-docs/prod_eth_switches/PB_SN2410.pdf
- Mellanox AS4610
  https://www.mellanox.com/products/ethernet-switches/as4610

Document history

Version 1.0	15 December 2020	Initial Version.
Version 2.0	16 January 2022	Update to latest hardware and software versions as of 01/13/2022 Software: 20.04.3 LTS (Linux Kernel 5.4), Kubernetes 1.21, MAAS 2.9, Juju 2.9.21 Hardware: Lenovo ThinkSystem SR630 V2, Lenovo ThinkSystem SR650 V2, NVIDIA A100 GPUs Updated procedure and tools to reflect the latest hardware and software

Trademarks and special notices

References in this document to Lenovo products or services do not imply that Lenovo intends to make them available in every country.

The following terms are trademarks of Lenovo in the United States, other countries, or both:
Lenovo®
AnyBay®
RackSwitch
ThinkSystem
TruDDR4
XClarity®

The following terms are trademarks of other companies:

Intel® and Xeon® are trademarks of Intel Corporation or its subsidiaries.

Linux® is the trademark of Linus Torvalds in the U.S. and other countries.

Azure® and Windows® are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Information is provided "AS IS" without warranty of any kind.

All customer examples described are presented as illustrations of how those customers have used Lenovo products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

Information concerning non-Lenovo products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by Lenovo. Sources for non-Lenovo list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. Lenovo has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-Lenovo products. Questions on the capability of non-Lenovo products should be addressed to the supplier of those products.

All statements regarding Lenovo future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. Contact your local Lenovo office or Lenovo authorized reseller for the full text of the specific Statement of Direction.

Some information addresses anticipated future capabilities. Such information is not intended as a definitive statement of a commitment to specific levels of performance, function or delivery schedules with respect to any future products. Such commitments are only made in Lenovo product announcements. The information is presented here to communicate Lenovo’s current investment and development activities as a good faith effort to help with our customers' future planning.

Performance is based on measurements and projections using standard Lenovo benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Photographs shown are of engineering prototypes. Changes may be incorporated in production models.

Any references in this information to non-Lenovo websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this Lenovo product and use of those websites is at your own risk.