

Canonical OpenStack on Dell EMC PowerEdge Rack Servers

Reference Architecture Guide

Dell EMC Converged Platforms and Solutions
October 2017

Revisions

Date	Description	Authors
October 2017	Initial release	

THIS DOCUMENT IS FOR INFORMATIONAL PURPOSES ONLY, AND MAY CONTAIN TYPOGRAPHICAL ERRORS AND TECHNICAL INACCURACIES. THE CONTENT IS PROVIDED AS IS, WITHOUT EXPRESS OR IMPLIED WARRANTIES OF ANY KIND.

Copyright © 2017 Dell Inc. All rights reserved. Dell and the Dell EMC logo are trademarks of Dell Inc. in the United States and/or other jurisdictions. All other marks and names mentioned herein may be trademarks of their respective companies.

Table of contents

Revisions.....	2
Trademarks.....	7
Glossary of Terms.....	8
Chapter 1 Overview	9
Executive summary	9
Dell EMC PowerEdge R630 overview.....	9
OpenStack Mitaka	9
Introduction.....	10
Core components	10
Optional components	10
OpenStack and Canonical.....	11
MAAS (Metal as a Service) physical cloud	11
Juju modeling tool	12
Why use Juju?	13
Landscape systems management tool.....	13
Firmware versions	14
Software versions	14
Chapter 2 Dell EMC PowerEdge R630 hardware specifications	15
Dell EMC PowerEdge R630 rack specifications	15
Dell EMC PowerEdge R630 server specifications	15
Rack layout.....	16
Hardware Configuration Notes	17
Chapter 3 Network architecture	19
Infrastructure layout.....	19
Network components.....	19
Server nodes	19
Leaf switches.....	20
VLANs	20
Out-of-Band management network	21
Chapter 4: Foundation cloud: infrastructure components	22
How MAAS works.....	22

High availability in MAAS	22
The node lifecycle	22
MAAS logical design	24
Install MAAS	24
Configure Your Hardware.....	24
Install Ubuntu Server.....	24
MAAS Installation	24
MAAS initial configurations.....	25
MAAS Credentials	26
Enlist and commission servers.....	26
Juju components	26
Juju controller - the heart of Juju	26
Charms	26
Bundles.....	26
Provision.....	27
Deploy	27
Monitor and manage	27
Comparing Juju to any configuration management tool.....	27
Monitoring.....	28
Observability Tools.....	28
Alerting and continuous service checks	29
Capacity Planning	29
Log Aggregation	30
Landscape	30
Chapter 5 Foundation cloud: OpenStack components.....	32
Storage charms	32
ceph-monitor.....	32
ceph-osd.....	32
ceph-radosgw	32
cinder	32
glance	33
Compute Charms	33
nova-compute.....	33

Resources Charms	34
HA/Clustering	34
Network space support.....	36
Heat orchestration	36
RabbitMQ-Server	36
OpenStack-dashboard	36
nova-cloud-controller	36
Percona cluster.....	37
Keystone.....	37
aodh Alarming	37
Ceilometer	38
Network charms.....	38
neutron-api	38
Network space support.....	38
neutron-gateway.....	39
neutron-openvswitch	39
Chapter 6 Management and monitoring tools.....	40
Designate	40
MongoDB.....	40
PostgreSQL	41
Kibana	41
Elasticsearch	41
HAproxy	41
Prometheus	41
Grafana.....	42
nrpe Nagios Add-on	42
Filebeat.....	42
Logical OpenStack service architecture.....	42
Chapter 7 OpenStack validation testing.....	43
Test cases	43
Tempest.....	43
Tempest test summary.....	43
Tempest test cases	44

Rally 46	
Rally tests summary	46
Rally test cases:	46
Appendix A Update history	48
Initial release	48
Appendix B References	49
Dell EMC documentation.....	49
Canonical documentation.....	49
Download Ubuntu 16.04 Server	49
Canonical service support	49
OpenStack Documentation	49
To Learn More	49

Trademarks

The information in this publication is provided as is. Dell Inc. makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any software described in this publication requires an applicable software license.

Copyright © 2017 Dell Inc. or its subsidiaries. All Rights Reserved. Dell, EMC, and other trademarks are trademarks of Dell Inc. or its subsidiaries. Intel, the Intel logo, the Intel Inside logo and Xeon are trademarks of Intel Corporation in the U.S. and/or other countries. Other trademarks may be the property of their respective owners. Published in the USA.

Ubuntu and Canonical are registered trademarks of Canonical Ltd. Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

Dell Inc. believes the information in this document is accurate as of its publication date. The information is subject to change without notice.

Glossary of Terms

The following Table provides definitions for some of the terms used in this document:

Table 1 Terminology

Terms	Description
AMT	Active Management Technology
Bcache	Allows SSD to work as cache for one or more slower hard disk drives
BMC	Baseband management controller
CLI	Command Line Interface
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Service
DSS	Dell Scalable Solutions
HA	High Availability
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.
HPC	High Performance Computing
Intel RSD	Intel Rack Scale Design
IPAM	IP address management
IPMI	Intelligent Platform Management Interface
Juju	An open source application modelling tool.
Landscape	Landscape is the leading management tool to deploy, monitor and manage Ubuntu servers.
LVM	Logical Volume Manager
MAAS	Metal as a Service, an open source hardware management tool
MLAG	Multi Chassis Link Aggregation Group
PXE	Preboot Execution Environment
RAID	Redundant Array of Independent Disks
REST API	REpresentational State Transfer Application Program Interface

Chapter 1 Overview

This document provides a complete reference architecture guide for Canonical OpenStack (Mitaka) on Dell EMC Hardware, including Dell EMC PowerEdge servers for compute and storage and Dell EMC Networking.

The document discusses the Dell EMC hardware specifications and the tools and services to set up both the hardware and software, including the foundation cluster and the OpenStack cluster. It also covers other tools used for the monitoring and management of the cluster in detail and how all these components work together in the system. The guide provides the deployment steps and references to configuration and automation scripts developed by Dell EMC and Canonical for the deployment process. Finally, tools for validation of the deployed solution and expected results are provided, and include OPNFV tools targeted for Telco customers.

Executive summary

An OpenStack® based cloud is now a common need by many organizations. Dell EMC 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
 - Compute nodes
 - Controller nodes
 - Storage nodes
- Network hardware and design
- Software layout
- System configurations

Dell EMC is a founding member of OpenStack and Gold member of the OpenStack Foundation since 2011.

Dell EMC PowerEdge R630 overview

The Dell EMC PowerEdge R630 rack level solution is comprised of pools of compute, storage and networking resources which are managed through a single point of rack management. All nodes in the rack are R630 1U servers handling compute, control, and storage functions, as assigned by the Metal as a Service (MAAS) management nodes.

For more information regarding the R630 hardware, refer to the [Chapter 2 Dell EMC PowerEdge R630 hardware specifications](#) section.

OpenStack Mitaka

This architecture guide is based on the OpenStack Mitaka - the 13th release of the most widely deployed open source software for building clouds. The Mitaka release was designed and built by an international community of 2,336 developers, operators, and users from 345 organizations. See [OpenStack Mitaka](#) for information regarding OpenStack Mitaka.

Introduction

This Dell EMC and Canonical designed architecture guide to make it easy for Dell EMC and Canonical customers to build their own operational readiness cluster and design their initial offerings using the current releases. Dell EMC and Canonical provide the support and services that customers need to stand up production-ready OpenStack clusters.

The solution is based on Canonical Mitaka OpenStack Platform that provides long term support, or (LTS). The code base for Canonical OpenStack Platform is evolving at a very rapid pace. See [Ubuntu release end of life](#) for more information.

Core components

Table 2 Core components

Component	Codename
Block Storage	Cinder with Ceph
Image Service	Glance with Ceph
Compute	Nova with KVM
Identity	Keystone
Networking	Neutron

Optional components

Table 3 Optional components

Component	Codename
Telemetry	Ceilometer
Orchestration	Heat
DNS as a Service	Designate
Validation Testing	Tempest
Dashboard	Horizon

Note: Before using Tempest, review the Tempest documentation at <https://docs.openstack.org/tempest/latest/>.

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

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 Dell EMC, Canonical and the broader OpenStack community.

Canonical 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) that are used to deploy, manage and update the OpenStack cluster nodes. In the Dell EMC Solution for Canonical OpenStack, the OpenStack Controllers, Computes and Ceph Storage servers comprise the OpenStack cluster.

OpenStack and Canonical

This reference architecture is based on the Canonical distribution of Ubuntu OpenStack. Canonical, the company behind the Ubuntu, and Platinum member of the OpenStack foundation, was the first company to commercially distribute and support OpenStack. Ubuntu is the reference operating system for OpenStack deployments, making it the easiest way to build an OpenStack cloud, and since 2011 the latest version of OpenStack has been included in every Ubuntu release. The release schedules of the two projects are synchronized and, ensures that OpenStack updates and releases are immediately available on widely deployed releases of Ubuntu.

Canonical Foundation Cloud services provide the service to design, deploy, manage and support customer cloud in POC, development, pre-production and production environments.

The Canonical Reference Architecture is delivered on a hyper-converged infrastructure approach, where any of the servers can accommodate more than one specific OpenStack role or service simultaneously. This hyper-converged approach has many benefits, including simplicity of operation and management overhead. Canonical can also deploy OpenStack in a more traditional manner, grouping servers per role - controllers, storage, and computes.

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, its 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, optimised 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.

Table 4 Key MAAS Features

Feature	Description
Automation	Automatic discovery and registration of every device on the network. BMC (IPMI, AMT and more) and PXE (IPv4 and 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.
Chassis management	Full chassis convergence. Dynamic hardware resource management with Intel RSD.
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.

Juju modeling tool

Juju is an open source application modeling 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.

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 your application.

Landscape systems management tool

The Landscape systems management tool helps you monitor, manage and update your entire Ubuntu infrastructure from a single interface. Part of Canonical's Ubuntu Advantage support service, Landscape brings you intuitive systems management tools combined with world-class support.

Landscape is part of the Canonical support service for Ubuntu Advantage.

Landscape is the most cost-effective way to support and monitor large and growing networks of desktops, servers and clouds. Landscape also reduces day-to-day management by allowing you to control of your infrastructure with:

Table 5 Landscape features

Feature	Description
Systems management	<ul style="list-style-type: none">• Manage desktop, server and cloud deployments• Up to 40,000 machines with a single instance• Create custom profiles for managing different machine classes• Easily install, update, rollback and remove software• Define policies for automated updates and security patches• Configure users and groups
Monitor your machines at scale	<ul style="list-style-type: none">• Set alerts for updates on specific machines• Graph trends of temperature, disk, memory usage and system load• List all processes running on a system and remotely stops rogue processes• Build your own graphs with custom metrics
Maintain security and compliance	<ul style="list-style-type: none">• Patch compliance - keep systems secure and up to date• Role Based Access Control (RBAC)• Automated audit logging and compliance reporting

Feature	Description
	<ul style="list-style-type: none"> Regulatory compliance is significantly simplified with custom reporting
Control your inventory	<ul style="list-style-type: none"> Quickly track full software package information for all registered machines Gather asset information in real time Create dynamic search groups to perform operations on categories of machines Easily access any machine property
Package repository management	<ul style="list-style-type: none"> Mirror and stage internal or external APT repositories Upload and manage custom packages

Firmware versions

Table 6 Firmware versions

Component	Version
iDRAC	2.41.40.40
BIOS	2.4.3

Software versions

The following versions of software are part of this reference architecture:

Table 7 Software versions

Component	Version
Ubuntu	16.04 LTS (kernel: 4.4)
OpenStack	Mitaka LTS (2016.4)
MAAS	2.2.x
Juju	2.2.x
OpenStack Charms	17.02

Chapter 2 Dell EMC PowerEdge R630 hardware specifications

The base validated reference architecture solution is based on the Dell EMC PowerEdge R630. The reference architecture uses the following rack and server specifications.

Dell EMC PowerEdge R630 rack specifications

Table 8 Dell EMC PowerEdge R630 rack specifications

Component Type	Component Description	Quantity
Rack	Standard data center rack (1) with enough capacity to hold 12x 1RU nodes, and 3x 1RU switches	1
Chassis	R630 (3 MAAS nodes, 9 OpenStack nodes)	12
Data switches	Dell S6010-ON 40Gbe switch	2
iDRAC/Provisioning switch	Dell PowerConnect S4048 (or equivalent)	1
Control cables	1G copper cables with RJ45 connector	12
Data cables	40GbE QSFP+ to 4x 10GbE SPF+ breakout cables	24

1 19 inch wide rack enclosure with rack mount rails which are 17 3/4" (450.85 mm) apart and whose height is measured in 1.75" (44.45 mm) unit increments

Dell EMC PowerEdge R630 server specifications

Table 9 Dell EMC PowerEdge R630 server specifications

Component Type	Component Description	Quantity
Processor	Intel® Xeon® E5-2650 v4	2
Memory	128GB total memory (4x 32GB or 8x16G RDIMMs)	4
Drive controller	PERC H730P	1
Hard drive	4TB 7.2K RPM SATA 6Gbps 3.5in hot-plug hard drive	6
Network card	Intel Ethernet X520 2x10Gb SFP+ DCS Mezzanine card	3
Network card	Intel Ethernet X520 2x10Gb SFP+ + 2x 1GbE network daughter card	1
Component Type	Component Description	Quantity

OOB license	iDRAC8 Express, integrated Dell Remote Access Controller, Express	1
Boot disks	Seagate 2x1TB 2.5" disks	2
SSD	Intel® SSD 2x 800G (S3510)	2

Rack layout

The reference deployment of Canonical OpenStack on the Dell EMC PowerEdge R630 server, utilizes three nodes as infrastructure nodes. The reference deployment uses the following purpose:

Table 10 Infrastructure nodes

Node	Purpose
Rack1-MAAS1	Infra # 1 (MAAS)
Rack1-MAAS2	Infra # 2 (MAAS)
Rack1-MAAS3	Infra # 3 (MAAS)

Table 11 OpenStack nodes

Node	Purpose
Rack1-node1	Converged node handling Compute + Control + Storage functions.
Rack1-node2	Converged node handling Compute + Control + Storage functions.
Rack1-node3	Converged node handling Compute + Control + Storage functions.
Rack1-node4	Converged node handling Compute + Control + Storage functions.
Rack1-node5	Converged node handling Compute + Control + Storage functions.
Rack1-node6	Converged node handling Compute + Control + Storage functions.
Rack1-node7	Converged node handling Compute + Control + Storage functions.
Rack1-node8	Converged node handling Compute + Control + Storage functions.
Rack1-node9	Converged node handling Compute + Control + Storage functions.

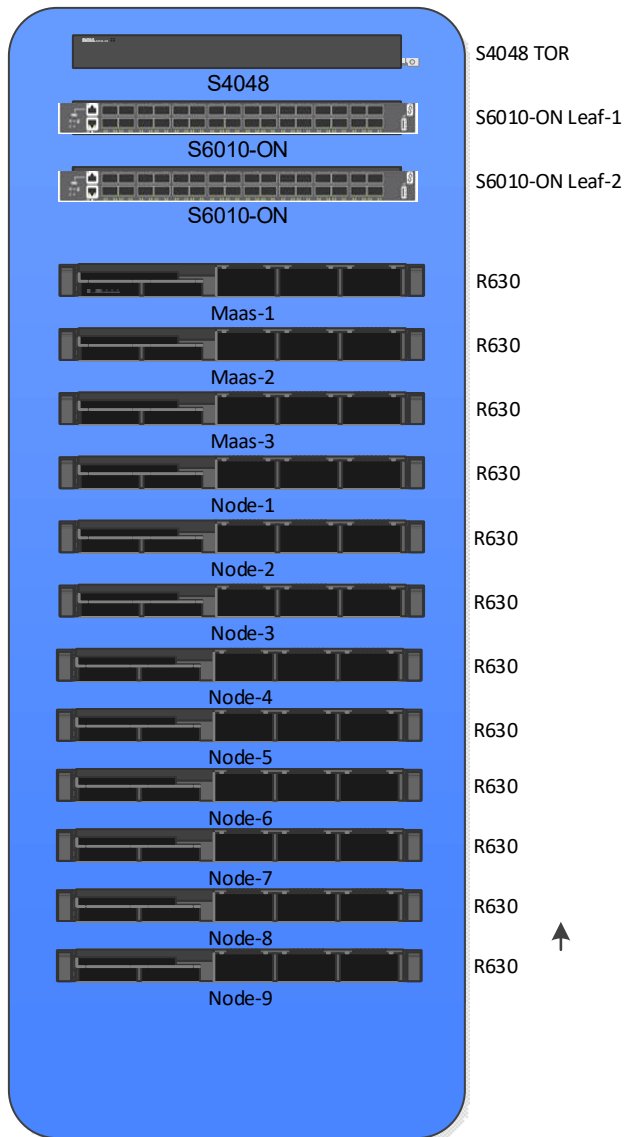


Figure 1 Rack layout of Dell EMC PowerEdge OpenStack

Hardware Configuration Notes

The Dell EMC PowerEdge R630 configurations are used with 10GbE networking. To ensure that the network is HA ready, four network cards each offering 2x 10GbE ports are required for each node.

The R630 servers need to be configured for the Dell EMC Canonical OpenStack Mitaka solution. Following are the configurations that need to be taken care of:

- BIOS
- iDRAC
- 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 must be enabled in the iDRAC.

For detailed hardware configurations of Dell EMC R630 solution for the Canonical OpenStack platform, consult a Dell EMC sales and services representative at openstack@dell.com.

Caution: Please ensure that the firmware on your hardware is up to date.

Chapter 3 Network architecture

A Dell EMC PowerEdge R630 rack solution is agnostic to the top of rack (ToR) switch a customer may choose for out-of-band (OOB) management. The reference implementation in this document uses the Dell EMC S4048 switch for this purpose. Two of the Dell EMC Networking S6010-ON 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.

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 iDRAC ports and the OOB management ports of the switches, are aggregated into a 1-rack unit (RU) Dell EMC PowerConnect S4048 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 iDRAC network. Ensure that the MAAS DHCP 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 high 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 Dell EMC S6010-ON's configured with VLT across them. For details regarding network configuration on the servers, please contact your Dell EMC services and sales representative.

Table 12 Supported channel bonding modes

Node Type	Channel Bonding Type
MAAS nodes	802.3ad (LACP mode 4)
OpenStack converged nodes	802.3ad (LACP mode 4)
OpenStack converged nodes	802.3ad (LACP mode 4)
OpenStack converged nodes	802.3ad (LACP mode 4)

Leaf switches

This reference implementation uses two Dell EMC Networking S6010-ON switches. There is a redundant physical 2x 40GbE connection between the two switches. The recommended architecture uses Dell EMC VLT between the switches in the leaf pair.

VLANs

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

VLAN	Description
admin	Used for admin-level access to services, including for automating administrative tasks.
internal	Used for internal endpoints and communications between most of the services.
public	Used for public service endpoints, such as using the OpenStack CLI to upload images to Glance.
external	Used by neutron to provide outbound access for tenant networks.
data	Used mostly for guest compute traffic between tenants and between tenants and OpenStack services.
storage(data)	Used by clients of the Ceph/Swift storage backend to consume block and object storage contents.
storage(cluster)	Used for replicating persistent storage data between units of Ceph/Swift.
OOB Management	Used for the iDRAC network.

Out-of-Band management network

The Management network of all the servers is aggregated into the Dell EMC PowerConnect S4048 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 iDRACs 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 Dell EMC sales representative for additional information.

Chapter 4: Foundation cloud: infrastructure components

The foundation cluster or the infrastructure nodes are composed of the following services and tools that are configured for high availability:

- MAAS
- Juju
- Landscape
- Monitoring
- Log aggregation

This section provides details about how each of these components work.

How MAAS works

Metal as a Service or MAAS, has a tiered architecture with a central Postgres database backing a region controller (regiond) that deals with operator requests. Distributed rack controllers, or (rackd) provide high-bandwidth services to multiple racks. The controller itself is stateless and horizontally scalable, and only presents a REST API.

Rackd provides DHCP, IPMI, PXE, TFTP and other local services. They cache large items like operating system install images at the rack level for performance but maintain no exclusive state other than credentials to talk to the controller.

High availability in MAAS

MAAS is a mission critical service that provides infrastructure coordination upon which HPC and cloud infrastructures depend. High availability in the region controller is achieved at the database level. The region controller will automatically switches gateways to ensure high availability of services to network segments in the event of a rack failure.

MAAS can scale from a small set of servers to many racks of hardware in a datacenter. High-bandwidth activities (such as the initial operating system installation) are handled by the distributed gateways enabling massively parallel deployments.

The node lifecycle

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

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.

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.

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.

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.

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.

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.

MAAS logical design

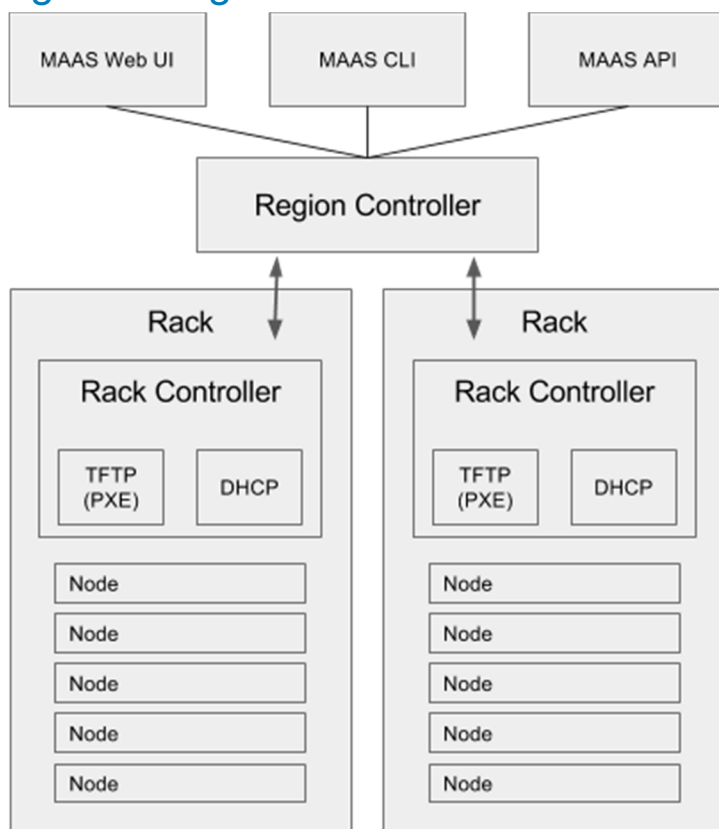


Figure 2 MAAS logical design

Install MAAS

In this Reference Architecture MAAS is installed in HA fashion using the Canonical Foundation Cloud process. Only customers who purchased the Canonical Foundation Cloud SKUs are able to use this document to install and configure MAAS in an automated fashion.

Configure Your Hardware

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

Install Ubuntu Server

Download [Ubuntu Server 16.04 LTS](#), and follow the step-by-step installation instructions on your MAAS server.

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 16.04-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
- Landscape dedicated server
- Monitoring and alerting systems
- Log aggregation and analysis systems
- Capacity planning

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 OpenStack, the nodes must have access to the OpenStack APIs
- 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 (`br0`) 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.

MAAS initial configurations

This section describes the following MAAS initial configurations:

- MAAS credentials
- Enlist and commission servers

MAAS Credentials

The MAAS admin user is created automatically and has a randomly generated password.. For your convenience, password is stored in the `MAAS-pass` file. Username is defined in the `infra.yaml` file.

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.
- 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.

Juju components

For an overview of Juju, refer to the [Juju modeling tool](#). This section discusses the working of different components of Juju.

Juju controller - the heart of Juju

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

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 as 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.

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

Provision

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

Deploy

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

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

Comparing Juju to any configuration management tool

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.

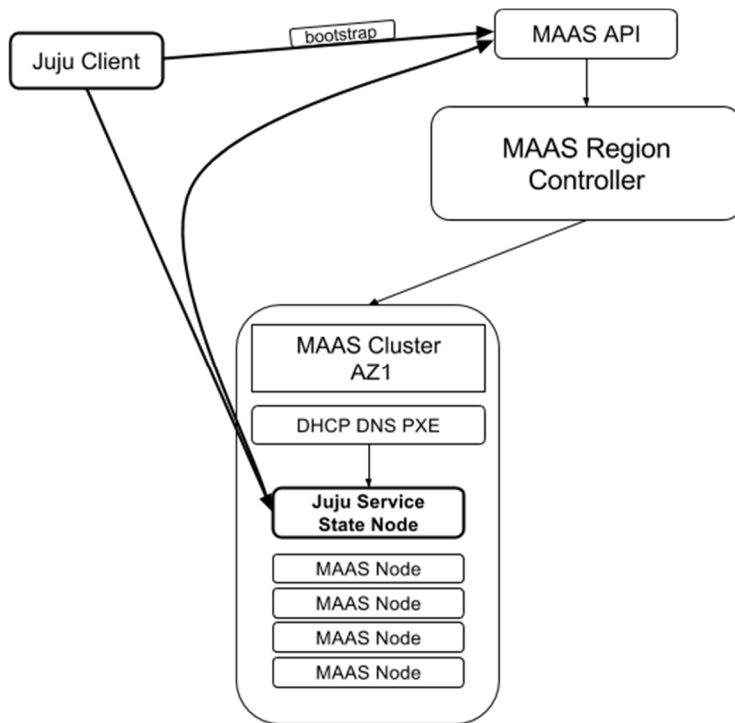


Figure 3 Juju client with MAAS

Monitoring

Canonical OpenStack includes a monitoring suite based on the best open source tools available.

Observability Tools

The Canonical OpenStack monitoring suite retrieves information from the different OpenStack 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.

Below displays the monitoring tool dashboard.



Figure 4 Monitoring tool dashboard

Alerting and continuous service checks

Canonical OpenStack encapsulates different operations as actions in the charms. Some of these operations are related to service tests, and produce alerts when the tests are not successful. The number of tests we add to our monitoring system grows over time. Our customers benefit, not only from our internal CI/CD systems, but also from the different checks that we implement in the many different clouds Canonical OpenStack deploys and manages.

The different tests performed regularly on the customer's clouds are aggregated and accessible to the customer through a [Thruk web interface implementation](#). Canonical can also integrate the underlying Nagios monitoring and alerting system with the existing tools that the customer may be using.

Canonical OpenStack implements Nagios, the industry standard in IT infrastructure monitoring that retrieves information such as the status of network, servers, storage, and services for deployment. Also, Nagios implements an alerting system that can be integrated with the operations center that sends notifications in real-time. This alerting allows the customer to have visibility of any issues related to their clouds.

Capacity Planning

The Canonical OpenStack monitoring suite includes both a capacity threshold alert and capacity forecast based on the consumption pattern for the cloud. This provides our customers with information about when their resources are expected to be exhausted and allows them to plan in advance with their hardware provider.

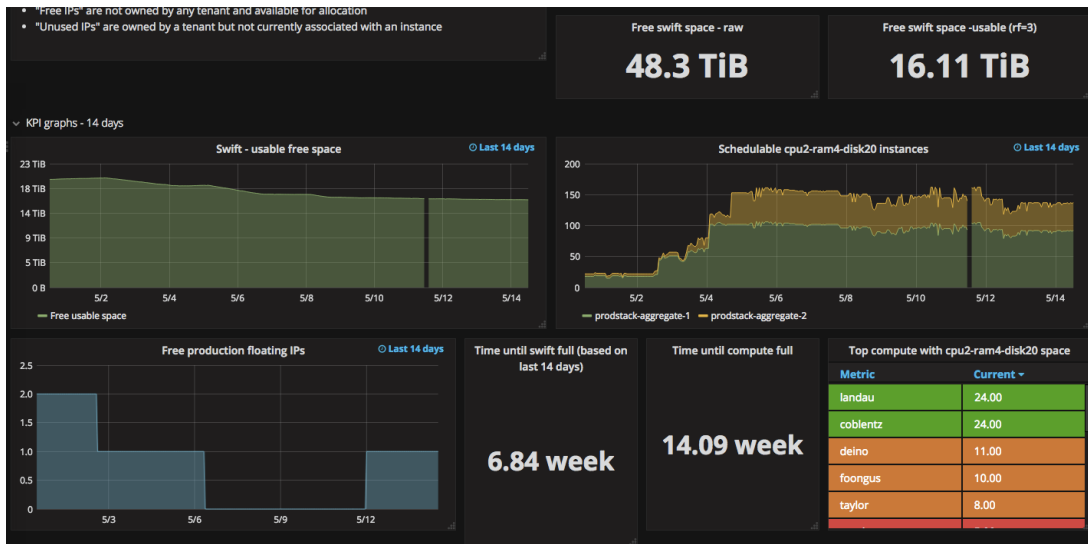


Figure 5 Capacity planning parameters statistics

Log Aggregation

Canonical OpenStack also implements the Elasticsearch 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 Canonical OpenStack as part of the charms, fulfilling the same requirements around upgradeability and operation.

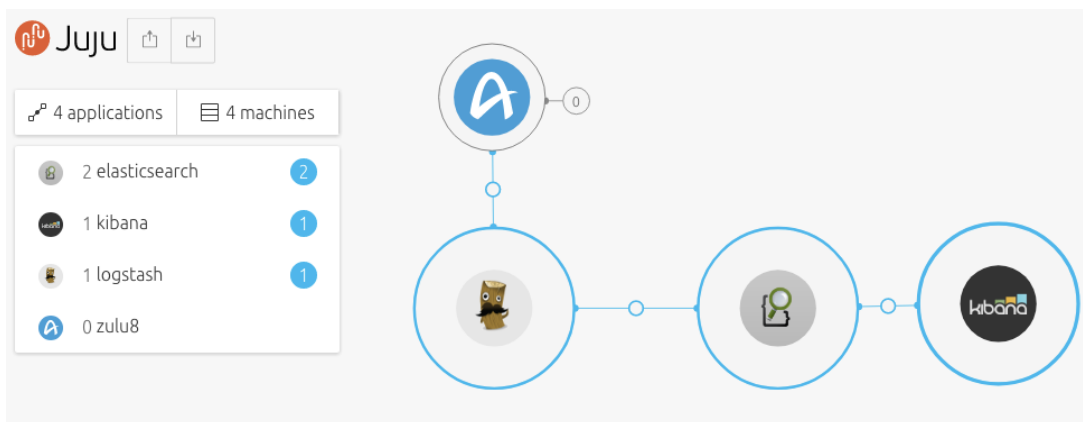


Figure 6 Juju dashboard

Landscape

The Landscape systems management tool helps you monitor, manage, and update your entire Ubuntu infrastructure from a single interface. As part of Canonical's Ubuntu Advantage support service, Landscape includes intuitive system management tools combined with world-class support.

This charm deploys Landscape Dedicated Server (LDS), and must be connected to other charms to be fully functional.

See Landscape systems management tool for more Landscape product information.

Visit <http://www.ubuntu.com/management> for more Landscape product information.

Chapter 5 Foundation cloud: OpenStack components

This chapter presents detailed information about the OpenStack components included as charms in Foundation Cloud.

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 storage however, this can be replaced by, or complemented with, any other storage solution.

ceph-monitor

The Ceph charm has two pieces of mandatory configuration for which no defaults are provided:

- fsid
- monitor-secret

Caution: These two pieces of configuration **must NOT be changed post bootstrap**; attempting to do this will cause a reconfiguration error and new service units will not join the existing ceph cluster.

At a minimum you must provide a Juju config file during the initial deployment with the fsid and monitor-secret options, which are the contents of ceph.yaml listed below:

By default the 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.

This charm uses the new-style Ceph deployment as reverse-engineered from the [Chef cookbook](#), although Dell EMC selected a different strategy to form the monitor cluster. Since we do not know the names or addresses of the machines in advance, Dell EMC uses the `_relation-joined_hook` to wait for all three nodes to come up, and then write their addresses to `ceph.conf` in the `mon` host parameter. After we initialize the monitor cluster a quorum forms quickly, and OSD bring-up proceeds.

ceph-osd

This charm provides the Ceph OSD personality for expanding storage capacity within a Ceph deployment.

ceph-radosgw

This charm provides the RADOS HTTP gateway supporting S3 and Swift protocols for object storage.

cinder

This charm provides the Cinder volume service for OpenStack. It is intended to be used alongside the other OpenStack components, starting with the Folsom release. Cinder is made up of three separate services:

- An API service
- A scheduler
- A volume service

glance

This charm provides the Glance image service for OpenStack. It is intended to be used alongside the other OpenStack components, starting with the Essex release in Ubuntu 12.04.

Glance may be deployed in a number of ways. This charm focuses on three (3) main configurations. All require the existence of the other core OpenStack services deployed via Juju charms, specifically:

- mysql
- keystone
- nova-cloud-controller

The following assumes these services have already been deployed:

- Local storage
- Swift backed storage
- Ceph backed storage

In this reference architecture we use Ceph backed storage.

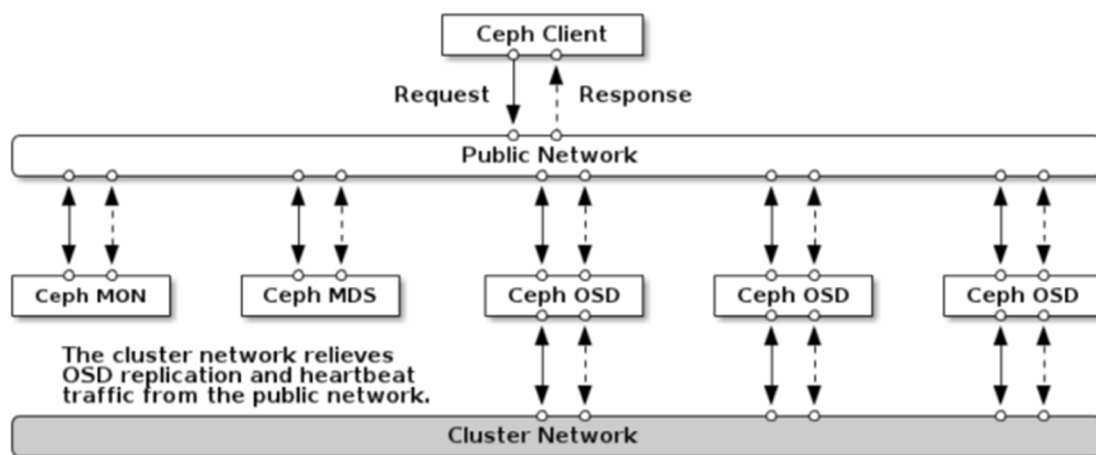


Figure 7 Ceph components

Compute Charms

Foundation Cloud uses nova compute, the OpenStack compute service.

nova-compute

This charm provides Nova Compute. The target platform is Ubuntu (preferably LTS) + OpenStack.

The following interfaces are provided:

- cloud-compute - Used to relate with (at least) one or more of:
 - nova-cloud-controller

- glance
- ceph
- cinder
- mysql
- ceilometer-agent
- rabbitmq-server
- neutron

In this deployment Canonical uses both KVM and LXD as a hypervisor.

Resources Charms

This topic describes the resource charms used by Foundation Cloud.

HA/Clustering

There are two mutually exclusive HA options available:

- Virtual IP address(es)
- DNS

In this reference architecture, deployment and testing the HA option used is the VIP. In both cases a relationship to `hacluster` is required, which provides the Corosync back end HA functionality. 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.

At a minimum, the configuration option, `vip`, must be set in order to use virtual IP HA. If multiple networks are being used, a VIP should be provided for each network, separated by spaces. Optionally, `vip_iface` or `vip_cidr` may be specified.

To use DNS high availability there are several prerequisites. However, DNS HA does not require the clustered nodes to be on the same subnet.

- Currently the DNS HA feature is only available for MAAS 2.0 or greater environments.
- MAAS 2.0 requires Juju 2.0 or greater.
- The clustered nodes must have static or "reserved" IP addresses registered in MAAS.
- The DNS hostname(s) must be pre-registered in MAAS before use with DNS HA.

At a minimum, the configuration option, `dns-ha`, must be set to `true`, and at least one or more of the following hostnames must be set, in order to use DNS HA:

- os-public-hostname
- os-internal-hostname
- os-internal-hostname

The charm will throw an exception in the following circumstances:

- If neither `vip` nor `dns-ha` is set and the charm is related to `hacluster`
- If both `vip` and `dns-ha` are set, as they are mutually exclusive
- If `dns-ha` is set and none of the `os-{admin,internal,public}-hostname(s)` are set

All the OpenStack services will be deployed in HA, and each service will have three units; each one running on a LXC container in a separate hypervisor. The Canonical OpenStack provides high availability for all OpenStack services as well as HA Juju. The following diagram explains the different types of HA used in Canonical OpenStack:

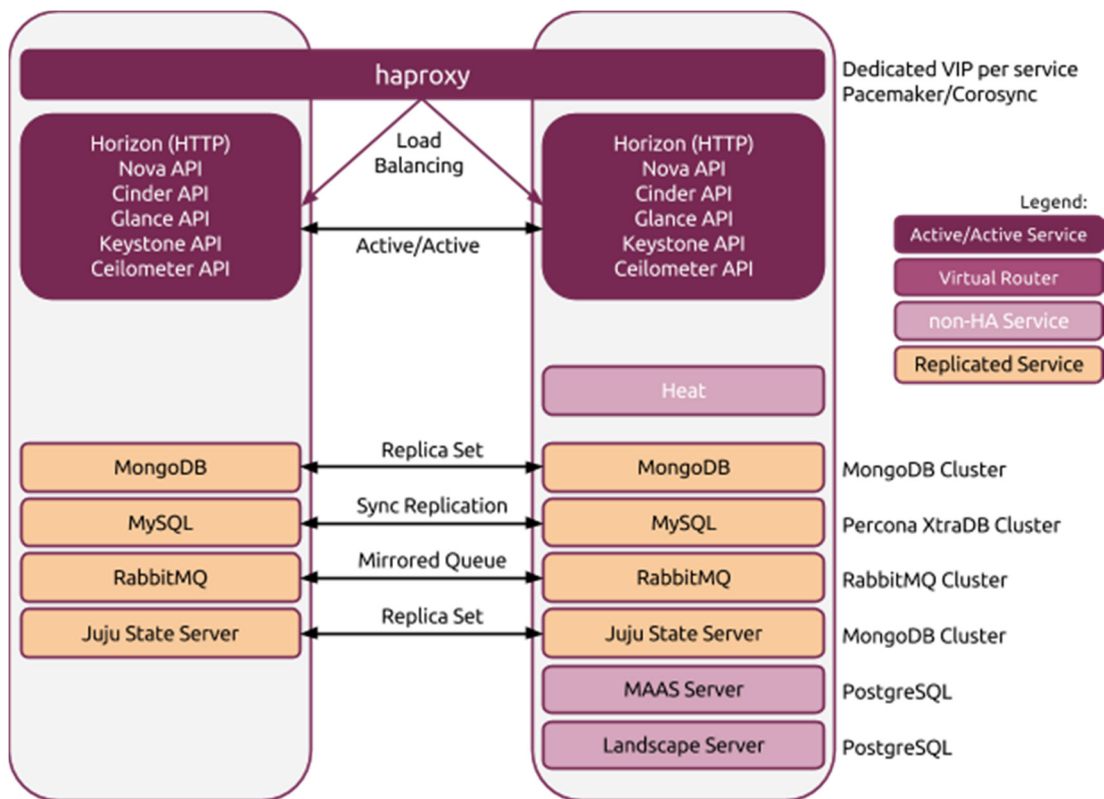


Figure 8 HA clustering of foundation cloud and OpenStack services

Network space support

OpenStack 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 public, internal, and admin endpoints. Access to the underlying MySQL instance can also be bound to a specific space using the `shared-db` relation.

Heat orchestration

Heat is the main project in the OpenStack Orchestration program. It implements an orchestration engine to launch multiple composite cloud applications based on templates, in the form of text files that can be treated like code. Heat requires the existence of the other core OpenStack services deployed via Juju charms; specifically:

- `mysql`
- `rabbitmq-server`
- `keystone`
- `nova-cloud-controller`

RabbitMQ-Server

RabbitMQ is an implementation of AMQP, the emerging standard for high performance enterprise messaging. The RabbitMQ server is a robust and scalable implementation of an AMQP broker. This charm deploys RabbitMQ server and provides AMQP connectivity to clients.

When more than one unit of the charm is deployed the charm will bring up a native RabbitMQ cluster. The process of clustering the units together takes some time.

Note: Due to the nature of asynchronous hook execution, it is possible that client relationship hooks may be executed before the cluster is complete. In some cases, this can lead to client charm errors. Single unit deployments behave as expected.

OpenStack-dashboard

The OpenStack Dashboard provides a Django-based web interface for use by both administrators and users of an OpenStack Cloud. It allows you to manage Nova, Glance, Cinder, and Neutron resources within the cloud.

nova-cloud-controller

`nova-cloud-controller` is the Cloud controller node for OpenStack Nova. It contains:

- `nova-scheduler`
- `nova-api`
- `nova-network`
- `nova-objectstore`

If console access is required, then `console-proxy-ip` should be set to a client accessible IP address that resolves to the `nova-cloud-controller`. If running in HA mode, then the public VIP is used if `console-proxy-ip` is set to `local`.

Note: The console access protocol is configured into a guest when it is created; if you change it then console access for existing guests will stop working.

Percona cluster

Percona XtraDB Cluster is a high availability and high scalability solution for MySQL clustering. Percona XtraDB Cluster integrates Percona Server with the Galera library of MySQL high availability solutions in a single product package, which enables you to create a cost-effective MySQL cluster. This charm deploys Percona XtraDB Cluster onto Ubuntu.

Note: Percona XtraDB Cluster is not a 'scale-out' MySQL solution. Reads and writes are channeled through a single service unit and synchronously replicated to other nodes in the cluster. Reads/writes are as slow as the slowest node you have in your deployment.

Keystone

This charm provides Keystone, the OpenStack identity service. Support for SSL and https endpoint is provided via a set of configuration options on the charm. Two types are supported:

- `use-https` - When enabled, this option tells Keystone to configure the identity endpoint as HTTPS. Under this model the keystone charm will either use the CA as provided by the user, or will generate its own and sync across peers. The certificate will be distributed to all service endpoints, which will be configured to use HTTPS.
- `https-service-endpoints` - When enabled, this option tells Keystone to configure ALL endpoints as HTTPS. Under this model the keystone charm will either use the CA as provided by the user, or will generate its own and synchronize across peers. The certificate will be distributed to all service endpoints, which will be configured to use HTTPS as well as configuring themselves to be used as HTTPS.

aodh Alarming

Ceilometer aims to deliver a single point of contact for billing systems, providing all the counters they need in order to establish customer billing across all current and future OpenStack components:

- The delivery of counters must be traceable and auditable
- The counters must be easily extensible to support new projects
- Agents performing data collections should be independent of the overall system

aodh provides the Alarming service as part of OpenStack telemetry.

Ceilometer

This charm provides the Ceilometer service for OpenStack. It is intended to be used alongside the other OpenStack components, starting with the Folsom release.

Ceilometer is made up of two (2) separate services:

- An API service
- A collector service

This charm enables the services to be deployed in different combination, depending upon user preferences and requirements.

Network charms

This section describes the various network charms included with Foundation Cloud.

neutron-api

This principle charm provides the OpenStack Neutron API service, which was previously provided by the `nova-cloud-controller` charm. When this charm is related to the `nova-cloud-controller` charm, the `nova-cloud-controller` charm will:

- 1 Shut down its API service
- 2 Deregister it from keystone
- 3 Inform the compute nodes of the new neutron URL

Network space support

This charm supports the use of Juju Network Spaces, allowing the charm to be bound to network space configurations managed directly by Juju.

Note: This is only supported with Juju 2.0 and above.

The API endpoints can be bound to distinct network spaces supporting the network separation of public, internal, and admin endpoints. Access to the underlying MySQL instance can also be bound to a specific space using the `shared-db` relation.

To use this feature, use the `--bind` option when deploying the charm:

```
$ juju deploy neutron-api --bind "public=public-space internal=internal-space  
admin=admin-space shared-db=internal-space"
```

Alternatively, these can also be provided as part of a Juju native bundle configuration:

```
neutron-api:
```

```
charm: cs:xenial/neutron-api
num_units: 1
bindings:
  public: public-space
  admin: admin-space
  internal: internal-space
  shared-db: internal-space
```

Note: Spaces must be configured in the underlying provider prior to attempting to use them.

Note: Existing deployments using `os-*-network` configuration options will continue to function; these options are preferred over any network space binding provided if set.

neutron-gateway

Neutron provides flexible software defined networking (SDN) for OpenStack.

This charm is designed to be used in conjunction with the rest of the OpenStack related charms in the charm store, to virtualize the network that Nova Compute instances plug into. It is designed as a replacement for `nova-network`. However, it does not yet support all of the features of `nova-network` (such as multihost), so it may not be suitable for all environments.

Neutron supports a rich plugin/extension framework for proprietary networking solutions, and supports:

By default this architecture uses the Open vSwitch (OVS).

See the [upstream Neutron documentation](#) for more details.

The gateway provides two key services:

- L3 network routing
- DHCP services

These are both required in a fully-functional Neutron OpenStack deployment. All internal and external network types are configured with `bridge-mappings` and `data-port`, and the `flat-network-providers` configuration option of the `neutron-api` charm. Once deployed, you can configure the network specifics using `neutron net-create`.

neutron-openvswitch

This subordinate charm provides the Neutron Open vSwitch configuration for a compute node. Once deployed it takes over the management of the Neutron base and plugin configuration on the compute node. This charm supports DPDK fast packet processing as well.

Chapter 6 Management and monitoring tools

This chapter describes the services which have been deployed to manage and monitor your OpenStack cloud. Canonical has a specially designed architecture for their customers to manage and monitor OpenStack clouds. Those services are part of Canonical Foundation Cloud Build (FCB) services. If customers have different requirements as part of FCB, we design the architecture for customers and do the deployments.

Designate

Designate provides DNSaaS services for OpenStack:

- A REST API for domain/record management Multi-tenant
- Integrated with Keystone for authentication
- Framework in place to integrate with Nova and Neutron notifications

MongoDB

MongoDB is a high-performance, open source, schema-free, document-oriented data store that is easy to deploy, manage, and use. MongoDB is network-accessible, written in C++, and offers the following features:

- Collection oriented storage - easy storage of object-style data
- Full index support, including on inner objects
- Query profiling
- Replication and fail-over support
- Efficient storage of binary data including large objects (e.g., videos)
- Auto-sharing for cloud-level scalability (Q209)
- High performance, scalability, and reasonable depth of functionality are the goals for the project

This charm deploys MongoDB in three configurations:

- Single node
- Replica set
- Shared clusters

In this Reference Architecture we deploy as a three node Replica set configuration.

Note: By default, the MongoDB application is installed from the Ubuntu archive, except for arm64 platforms. The version of MongoDB in the archive is known to have issues on arm64, so by default this charm will use `ppa:mongodb-arm64/ppa`, which contains backported fixes for this architecture.

PostgreSQL

PostgreSQL is a powerful, open source, object-relational database system. It has more than 15 years of active development and a proven architecture that has earned it a strong reputation for reliability, data integrity, and correctness. It is fully ACID-compliant, has full support for foreign keys, joins, views, triggers, and stored procedures (in multiple languages). It includes most SQL: 2008 data types, including:

- INTEGER
- NUMERIC
- BOOLEAN
- CHAR
- VARCHAR
- DATE
- INTERVAL
- TIMESTAMP

PostgreSQL also supports storage of binary large objects, including pictures, sounds, or video. It has native programming interfaces for C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, among others, and exceptional documentation [PostgreSQL Manuals](https://www.postgresql.org/docs/).

Kibana

Kibana is a search front end for Logstash. This charm provides Kibana from <https://www.elastic.co/products/kibana>

Elasticsearch

Elasticsearch is a flexible and powerful open source, distributed, real-time search and analytics engine. Elasticsearch was architected from the ground up for use in distributed environments where reliability and scalability are must haves, Elasticsearch gives you the ability to move easily beyond simple full-text search. Through its robust set of APIs and query DSLs, plus clients for the most popular programming languages, Elasticsearch delivers on the near limitless promises of search technology.

HAproxy

HAproxy is a TCP/HTTP reverse proxy that is particularly suited for high availability environments. It features connection persistence through HTTP cookies, load balancing, header addition, modification, and deletion both ways. It has request-blocking capabilities, and provides an interface to display server status.

Prometheus

Prometheus 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 if some condition is observed to be true.

Grafana

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

nrpe Nagios Add-on

Nagios is a host/service/network monitoring and management system. The purpose of this add-on is to allow you to execute Nagios plugins on a remote host in as transparent a manner as possible. This program runs as a background process on the remote host and processes command execution requests from the `check_nrpe` plugin on the Nagios host.

Filebeat

As the next-generation Logstash Forwarder, Filebeat tails logs and quickly sends this information to Logstash for further parsing and enrichment, or to Elasticsearch for centralized storage and analysis.

Logical OpenStack service architecture

The figure below explains how and where services are deployed in the Canonical OpenStack reference architecture. Canonical supports multiple custom architectures based on customer requirement of their production OpenStack.

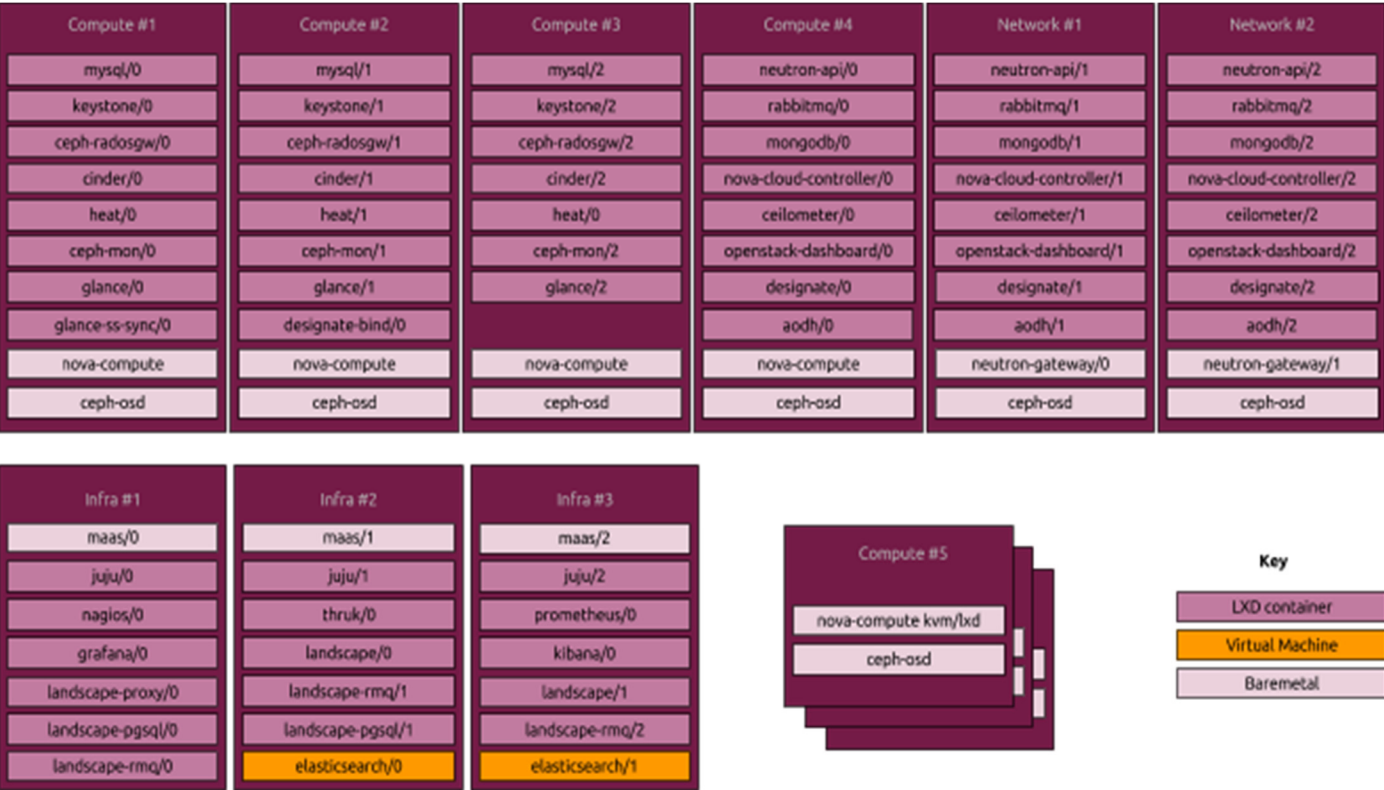


Figure 9 Infrastructure and OpenStack components distributed service architecture

Chapter 7 OpenStack validation testing

This chapter describes validating an OpenStack deployment with:

- Test Cases
- Tempest
- Rally

Test cases

Canonical uses OPNFV `functest` to run tempest and rally test cases on the above deployed environment for test and verification purpose. Canonical used only functional test cases for this environment. Performance test cases are out of scope of this document, but can be delivered as part of a professional services engagement.

Tempest

Tempest is designed to be useful for a large number of different environments. This is useful for:

- Gating commits to OpenStack core projects
- Validating OpenStack cloud implementations for correctness
- Performing as a burn-in tool for OpenStack clouds

As such Tempest tests come in many flavors, each with their own rules and guidelines. The Tempest repository is structured as follows:

- **tempest/** - main directory
- **api/** - API tests
- **scenario/** - complex scenario tests
- **tests/** - unit tests for Tempest internals

Each of these directories contains different types of tests. What belongs in each directory, the rules and examples for good tests, are documented in a *README.rst* file in the directory.

Run the following command to start the `functest` test cases:

```
$ functest run test case tempest-sanity
```

Tempest test summary

The Tempest test summary results were:

Tempest - INFO -	
Tempest - INFO - Verification	
Tempest - INFO - Status	finished
Tempest - INFO - Duration	0:21:30

Tempest - INFO -	Run arguments	concurrency: 1
Tempest - INFO -	Verifier type	tempest (namespace: openstack)
Tempest - INFO -	Tests count	105
Tempest - INFO -	Tests duration, sec	1276.104
Tempest - INFO -	Success	104
Tempest - INFO -	Skipped	1
Tempest - INFO -	Expected failures	0
Tempest - INFO -	Failures	0

Tempest test cases

The following test cases were performed after the deployment on this architecture:

```

api.compute.flavors.test_flavors.FlavorsV2TestJSON.test_get_flavor
api.compute.flavors.test_flavors.FlavorsV2TestJSON.test_list_flavors
api.compute.security_groups.test_security_group_rules.SecurityGroupRulesTestJSON.test_security_group_rules_create
api.compute.security_groups.test_security_group_rules.SecurityGroupRulesTestJSON.test_security_group_rules_list
api.compute.security_groups.test_security_groups.SecurityGroupsTestJSON.test_security_groups_create_list_delete
api.compute.servers.test_attach_interfaces.AttachInterfacesTestJSON.test_add_remove_fixed_ip
api.compute.servers.test_create_server.ServersTestJSON.test_list_servers
api.compute.servers.test_create_server.ServersTestJSON.test_verify_server_details
api.compute.servers.test_create_server.ServersTestManualDisk.test_list_servers
api.compute.servers.test_create_server.ServersTestManualDisk.test_verify_server_details
api.compute.servers.test_server_actions.ServerActionsTestJSON.test_reboot_server_hard
api.compute.servers.test_server_addresses.ServerAddressesTestJSON.test_list_server_addresses
api.compute.servers.test_server_addresses.ServerAddressesTestJSON.test_list_server_addresses_by_network
api.identity.admin.v2.test_services.ServicesTestJSON.test_list_services
api.identity.admin.v2.test_users.UsersTestJSON.test_create_user
api.identity.admin.v3.test_credentials.CredentialsTestJSON.test_credentials_create_get_update_delete
api.identity.admin.v3.test_domains.DefaultDomainTestJSON.test_default_domain_exists
api.identity.admin.v3.test_domains.DomainsTestJSON.test_create_update_delete_domain
api.identity.admin.v3.test_endpoints.EndPointsTestJSON.test_update_endpoint
api.identity.admin.v3.test_groups.GroupsV3TestJSON.test_group_users_add_list_delete
api.identity.admin.v3.test_policies.PoliciesTestJSON.test_create_update_delete_policy
api.identity.admin.v3.test_regions.RegionsTestJSON.test_create_region_with_specific_id
api.identity.admin.v3.test_roles.RolesV3TestJSON.test_role_create_update_show_list
api.identity.admin.v3.test_services.ServicesTestJSON.test_create_update_get_service
api.identity.admin.v3.test_trusts.TrustsV3TestJSON.test_get_trusts_all
api.identity.v2.test_api_discovery.TestApiDiscovery.test_api_media_types
api.identity.v2.test_api_discovery.TestApiDiscovery.test_api_version_resources
api.identity.v2.test_api_discovery.TestApiDiscovery.test_api_version_statuses
api.identity.v3.test_api_discovery.TestApiDiscovery.test_api_media_types
api.identity.v3.test_api_discovery.TestApiDiscovery.test_api_version_resources
api.identity.v3.test_api_discovery.TestApiDiscovery.test_api_version_statuses
api.image.v2.test_images.BasicOperationsImagesTest.test_delete_image
api.image.v2.test_images.BasicOperationsImagesTest.test_register_upload_get_image_file
api.image.v2.test_images.BasicOperationsImagesTest.test_update_image
api.network.test_extensions.ExtensionsTestJSON.test_list_show_extensions
api.network.test_floating_ips.FloatingIPTestJSON.test_create_floating_ip_specifying_a_fixed_ip_address
api.network.test_floating_ips.FloatingIPTestJSON.test_create_list_show_update_delete_floating_ip
api.network.test_networks.BulkNetworkOpsIpv6Test.test_bulk_create_delete_network
api.network.test_networks.BulkNetworkOpsIpv6Test.test_bulk_create_delete_port
api.network.test_networks.BulkNetworkOpsIpv6Test.test_bulk_create_delete_subnet

```

api.network.test_networks.BulkNetworkOpsTest.test_bulk_create_delete_network
 api.network.test_networks.BulkNetworkOpsTest.test_bulk_create_delete_port
 api.network.test_networks.BulkNetworkOpsTest.test_bulk_create_delete_subnet
 api.network.test_networks.NetworkslpV6Test.test_create_update_delete_network_subnet
 api.network.test_networks.NetworkslpV6Test.test_external_network_visibility
 api.network.test_networks.NetworkslpV6Test.test_list_networks
 api.network.test_networks.NetworkslpV6Test.test_list_subnets
 api.network.test_networks.NetworkslpV6Test.test_show_network
 api.network.test_networks.NetworkslpV6Test.test_show_subnet
 api.network.test_networks.NetworksTest.test_create_update_delete_network_subnet
 api.network.test_networks.NetworksTest.test_external_network_visibility
 api.network.test_networks.NetworksTest.test_list_networks
 api.network.test_networks.NetworksTest.test_list_subnets
 api.network.test_networks.NetworksTest.test_show_network
 api.network.test_networks.NetworksTest.test_show_subnet
 api.network.test_ports.PortslpV6TestJSON.test_create_port_in_allowed_allocation_pools
 api.network.test_ports.PortslpV6TestJSON.test_create_port_with_no_securitygroups
 api.network.test_ports.PortslpV6TestJSON.test_create_update_delete_port
 api.network.test_ports.PortslpV6TestJSON.test_list_ports
 api.network.test_ports.PortslpV6TestJSON.test_show_port
 api.network.test_ports.PortsTestJSON.test_create_port_in_allowed_allocation_pools
 api.network.test_ports.PortsTestJSON.test_create_port_with_no_securitygroups
 api.network.test_ports.PortsTestJSON.test_create_update_delete_port
 api.network.test_ports.PortsTestJSON.test_list_ports
 api.network.test_ports.PortsTestJSON.test_show_port
 api.network.test_routers.RoutersIpV6Test.test_add_multiple_router_interfaces
 api.network.test_routers.RoutersIpV6Test.test_add_remove_router_interface_with_port_id
 api.network.test_routers.RoutersIpV6Test.test_add_remove_router_interface_with_subnet_id
 api.network.test_routers.RoutersIpV6Test.test_create_show_list_update_delete_router
 api.network.test_routers.RoutersTest.test_add_multiple_router_interfaces
 api.network.test_routers.RoutersTest.test_add_remove_router_interface_with_port_id
 api.network.test_routers.RoutersTest.test_add_remove_router_interface_with_subnet_id
 api.network.test_routers.RoutersTest.test_create_show_list_update_delete_router
 api.network.test_security_groups.SecGroupIPv6Test.test_create_list_update_show_delete_security_group
 api.network.test_security_groups.SecGroupIPv6Test.test_create_show_delete_security_group_rule
 api.network.test_security_groups.SecGroupIPv6Test.test_list_security_groups
 api.network.test_security_groups.SecGroupTest.test_create_list_update_show_delete_security_group
 api.network.test_security_groups.SecGroupTest.test_create_show_delete_security_group_rule
 api.network.test_security_groups.SecGroupTest.test_list_security_groups
 api.network.test_subnetpools_extensions.SubnetPoolsTestJSON.test_create_list_show_update_delete_subnetpools
 api.network.test_versions.NetworksApiDiscovery.test_api_version_resources
 api.orchestration.stacks.test_resource_types.ResourceTypesTest.test_resource_type_list
 api.orchestration.stacks.test_resource_types.ResourceTypesTest.test_resource_type_show
 api.orchestration.stacks.test_resource_types.ResourceTypesTest.test_resource_type_template
 api.orchestration.stacks.test_soft_conf.TestSoftwareConfig.test_get_deployment_list
 api.orchestration.stacks.test_soft_conf.TestSoftwareConfig.test_get_deployment_metadata
 api.orchestration.stacks.test_soft_conf.TestSoftwareConfig.test_get_software_config
 api.orchestration.stacks.test_soft_conf.TestSoftwareConfig.test_software_deployment_create_validate
 api.orchestration.stacks.test_soft_conf.TestSoftwareConfig.test_software_deployment_update_no_metadata_change
 api.orchestration.stacks.test_soft_conf.TestSoftwareConfig.test_software_deployment_update_with_metadata_change
 api.orchestration.stacks.test_stacks.StacksTestJSON.test_stack_crud_no_resources
 api.orchestration.stacks.test_stacks.StacksTestJSON.test_stack_list_responds
 api.volume.test_volumes_actions.VolumesV1ActionsTest.test_attach_detach_volume_to_instance

```

api.volume.test_volumes_actions.VolumesV2ActionsTest.test_attach_detach_volume_to_instance
api.volume.test_volumes_get.VolumesV1GetTest.test_volume_create_get_update_delete
api.volume.test_volumes_get.VolumesV1GetTest.test_volume_create_get_update_delete_from_image
api.volume.test_volumes_get.VolumesV2GetTest.test_volume_create_get_update_delete
api.volume.test_volumes_get.VolumesV2GetTest.test_volume_create_get_update_delete_from_image
api.volume.test_volumes_list.VolumesV1ListTestJSON.test_volume_list
api.volume.test_volumes_list.VolumesV2ListTestJSON.test_volume_list
tempest.scenario.test_network_basic_ops.TestNetworkBasicOps.test_network_basic_ops
tempest.scenario.test_server_basic_ops.TestServerBasicOps.test_server_basic_ops
tempest.scenario.test_server_multinode.TestServerMultinode.test_schedule_to_all_nodes
tempest.scenario.test_volume_boot_pattern.TestVolumeBootPattern.test_volume_boot_pattern
tempest.scenario.test_volume_boot_pattern.TestVolumeBootPatternV2.test_volume_boot_pattern

```

Rally

OpenStack is a large ecosystem of cooperative services. Rally is a benchmarking tool that answers the question, "How does OpenStack work at scale?" To make this possible, Rally automates and unifies multi-node OpenStack deployment, cloud verification, benchmarking, and profiling. Rally does this in a generic way, which makes it possible to verify if OpenStack is going to work well on, for example, a 1k-servers installation under high load. Thus it can be used as a basic tool for an OpenStack CI/CD system that continuously improves SLA, performance and stability.

Rally tests summary

The table below displays the Rally test results run on this architecture.

Table 13 Rally test results

Module	Test Cases	Results
authenticate	12	Pass (100%)
cinder	11	Pass (100%)
glance	19	Pass (100%)
heat	16	Pass (100%)
keystone	50	Pass (100%)
neutron	30	Pass (100%)
nova	26	Pass (100%)
quotas	12	Pass (100%)
requests	2	Pass (100%)

Rally test cases:

Rally has multiple test scenarios against each service. Below are the test scenarios listed which has been tested as part of Rally tests. Each scenario will have multiple repeated tests related to service tested.

test scenario Authenticate.validate_glance
 test scenario Authenticate.keystone
 test scenario Authenticate.validate_heat
 test scenario Authenticate.validate_nova
 test scenario Authenticate.validate_cinder
 test scenario Authenticate.validate_neutron
 test scenario GlanceImages.list_images
 test scenario GlanceImages.create_image_and_boot_instances
 test scenario CinderVolumes.create_and_extend_volume
 test scenario CinderVolumes.create_and_delete_volume
 test scenario CinderVolumes.create_from_volume_and_delete_volume
 test scenario CinderVolumes.create_and_delete_snapshot
 test scenario HeatStacks.create_suspend_resume_delete_stack
 test scenario KeystoneBasic.create_tenant_with_users
 test scenario KeystoneBasic.create_add_and_list_user_roles
 test scenario KeystoneBasic.add_and_remove_user_role
 test scenario KeystoneBasic.create_update_and_delete_tenant
 test scenario KeystoneBasic.create_and_delete_service
 test scenario KeystoneBasic.create_tenant
 test scenario KeystoneBasic.create_user
 test scenario KeystoneBasic.create_and_list_tenants
 test scenario KeystoneBasic.create_and_delete_role
 test scenario KeystoneBasic.get_entities
 test scenario KeystoneBasic.create_and_list_users
 test scenario NeutronNetworks.create_and_delete_ports
 test scenario NeutronNetworks.create_and_list_routers
 test scenario NeutronNetworks.create_and_delete_routers
 test scenario NeutronNetworks.create_and_list_ports
 test scenario NeutronNetworks.create_and_delete_subnets
 test scenario NeutronNetworks.create_and_delete_networks
 test scenario NeutronNetworks.create_and_list_networks
 test scenario NeutronNetworks.create_and_list_subnets
 test scenario NovaKeypair.boot_and_delete_server_with_keypair
 test scenario Quotas.cinder_update
 test scenario Quotas.neutron_update
 test scenario Quotas.cinder_update_and_delete
 test scenario Quotas.nova_update_and_delete
 test scenario Quotas.nova_update
 test scenario HttpRequests.check_request

In Canonical deployment, all of the rally test cases passed and worked 100%. In the event that you have any issues with OpenStack, consult your Canonical service representative.

Appendix A Update history

This appendix lists changes to this document, per major release.

Initial release

First Architecture Guide for the Dell EMC Solution on Canonical OpenStack Platform.

Appendix B References

Please see the following resources for more information.

Dell EMC documentation

- <http://en.community.dell.com/techcenter/cloud/w/wiki/12401.dell-emc-canonical-openstack-cloud-solutions>
- <http://www.dell.com/en-us/work/learn/rack-scale-infrastructure>

Canonical documentation

- <https://jujucharms.com/>
- <https://MAAS.io/>
- <https://wiki.ubuntu.com/>
- <https://www.ubuntu.com/>
- [BootStack: Fully managed operations OpenStack and Kubernetes](#)

Download Ubuntu 16.04 Server

- <http://www.ubuntu.com/download/server>

Canonical service support

- [Canonical Foundation Cloud Services](#)
- [OpenStack UA Support](#)

OpenStack Documentation

- https://wiki.openstack.org/wiki/Main_Page
- <https://wiki.opnfv.org/display/functest/Functest+1.+Getting+Started>
- <https://docs.openstack.org/charm-guide/latest/>

To Learn More

If you need additional services or implementation help, please contact your Dell EMC sales representative, or email openstack@dell.com.

Copyright © 2017 Dell Inc. or its subsidiaries. All rights reserved. Trademarks and trade names may be used in this document to refer to either the entities claiming the marks and names or their products. Specifications are correct at date of publication but are subject to availability or change without notice at any time. Dell EMC and its affiliates cannot be responsible for errors or omissions in typography or photography. Dell EMC's Terms and Conditions of Sales and Service apply and are available on request. Dell EMC service offerings do not affect consumer's statutory rights.

Dell EMC, the DELL EMC logo, the DELL EMC badge, and PowerEdge are trademarks of Dell Inc.