Lenovo

Reference Architecture:

Lenovo Open Cloud Reference Architecture

Version 1.0

Feng Xia, Lijun Gu, Jay Bryant, Miro Halas

January 31, 2021

DRAFT COPY

Lenovo Open Cloud (LOC) is an extension based on the Lenovo Open Cloud RHHI Reference Architecture[1]. LOC provides a list of services using commercial and Opensource software. These services support automation of deployment of OpenStack and Ceph, which was traditionally measured in days, even weeks, now in hours. Services continue to manage LOC hardware and software, and maintain their health through full life cycle including system upgrade, fixes, devops, CICD, extending the cluster by adding new hardware resources, workload migration, backup, and restore.

This document provides a comprehensive description of these services. We will demonstrate using LOC to build storage infrastructures such as Red Hat Ceph[2], and cloud infrastructures such as Red Hat OpenStack 13[3].

Contents

List of Tables

List of Figures

1

Introduction

The target audience for this Reference Architecture (RA) is system administrators or system architects who are looking for an integrated environment that makes deployment of an infrastructure such as Ceph and OpenStack, either for a lab environment or for production, easier, quicker, better automated, better supported, than the traditional, manual intensive approach. Especially if bringing new baremetal servers onboard for a cluster has been a time-consuming experience, Lenovo Open Cloud is a solution to change that.

Further, for users who are looking to deploy a virtualized control plane so as to take advantage of snapshot/backup, restoration, and migration, Lenovo Open Cloud will demonstrate how this feature is implemented by Lenovo's hardware and a selected set of software services.

Technology evolves fast. We have seen the wave of hardware virtualization in both server space and in personal computing. Then came cloud computing, in which infrastructure becomes even more abstract and remote to the end user than ever before. Instead of being viewed as brick and mortar, server, storage and networking are **resources** that can be requested, leased for a period of time, paid per use, and released when done — all through nothing but an online account and a credit card. The flexibility of this model and the feeling that resource pools can be extended boundlessly has both lowered the barrier of entry of new applications growing from zero to infinity with little sweat, and elevated requirements on the design, implementation, and operation of such infrastructure

Further more, along the trail of technology evolution, business has been left with an army of legacy systems which were designed and built on a technology stack that was adequate then, but not in trend now. Millions have been invested, millions of users are probably depending on the continuity of service, and many developers and operators were trained and are given the responsibility to maintain such stacks. It is neither feasible to cut the cord just because a new technology becomes the talk of the day, nor advisable to continue as before without taking advantage of what new tools

can bring. Therefore, it is not only desirable, but in our opinion essential, to have an infrastructure that is both flexible and balanced — it must support a broad range of users and applications by providing a platform that has a rich mix of building blocks which, first of all, covers common needs out of the box, such as keeping an operating system up to date via patches, updates and hotfixes, while maintaining an open architecture to extend both horizontally in terms of resource (compute, storage, networking), and vertically (application stack).

It is with this in mind that Lenovo Open Cloud is designed to combine the best of technologies in the market today into a coherent user experience while all the following users will feel at home:

1. **VM users**: Open Cloud supports hardware virtualization in its core. Virtual machine users and applications can be migrated onto the platform while minimizing dependency on the underlying hardware environment.

2. **Cloud users**: Open Cloud provides on-premise cloud computing environments based on OpenStack, the leading cloud operating system.

3. **Container users**: Devops have continuously pushed the boundary to merge development and production into a single, consistent experience that developers use as a `sandbox` should be identical to what can be used in production. By doing so not only will we eliminate the necessity to maintain multiple stacks catering for different environments — a typical setup will have one for development, one for testing, and one for production, but minimize the chance of incompatibility and bugs due to differences between two environments.

This RA describes the system architecture of the Lenovo Open Cloud (LOC) with a focus on its management services, and how users can use them to build infrastructure such as Ceph storage and OpenStack. For deployment mode based on the Lenovo Open Cloud RHHI Reference Architecture[1], knowledge and experience of the LOC RHHI would be highly desirable.

2

Business problem and business value

2.1 Business problem

IT infrastructure is the back bone of all other IT activities. But at any given time there are always competing technologies that require different, if not conflicting, infrastructure to support — the stacks are different (more so on the software side than on the hardware side), the groups of users and developers are different, and the workloads running on top often have different architectural genes, and sometimes completely different design paradigm. How to build an infrastructure such as an OpenStack on demand? and how to lower the barrier of operating these different infrastructures using some common "best practices"?

Further, infrastructure means different things to different people, just as the term "cloud" does. What does not change is that all software, regardless the form, language, design pattern, will live on a set of hardware, or be more precise, hardware resources — servers (also referred to as "compute"), storages, networks. From end user's prospective, these resources are magically appearing whenever needed, always available, always functional, always ready. How to turn hardware to new resources quickly? and how to minimize service interruption by having better failure detection, health monitoring, and less manual steps that are prone to operator errors?

Finally, hardware breaks, and hardware is expansive. Not to mention a broken hardware can damage the integrity of its workloads to the point that it can kill the business. However, in today's IT, hardware is also shared so that its utilization can be improved as a measure to reduce its total cost of ownership (TCO). This makes its more difficult to troubleshoot when things go awry, and at the mean time having more diversified user requirements to meet.

These, are the challenges.

2.2 Business value

Lenovo Open Cloud is a solution to meet these challenges. We recognize the need to support multiple platforms on the same set of hardware so to enable development, deployment, and business that requires or prefers different technology. We believe infrastructure should enable business and application developers to have the freedom to choose the best tools for the job, not hinder them.

Out of box, the Lenovo Open Cloud has fully deployed control plane to build a hyperconverged hypervisor cluster with gluster storage over three physical nodes, OpenStack, Ceph storage, and Kubernetes.

Hyperconverged cluster provides small footprint (3-node as default) and robust functions to satisfy compute and storage need, especially for VM-based workloads. All others are ready to go using Lenovo's just-in-time license so that no cost is incurred until user decides to activate the OpenStack, for example, by creating an instance over a number of compute nodes. The process of adding new hardware resources, and on board process can be fully automated using LOC's management services.

Further, each of these platforms are defined in descriptive YAML. Using them as templates, user can specify different configurations to build new instance with a click of a button — an OpenStack 10 environment for production, another OpenStack 13 environment for dev team, or a k8s for the Northeast region only — the possibilities are wide open.

Last but not the least, by streamlining the deployment process and treat infrastructure as code, LOC can reduce transaction cost of communication, which traditionally requires hardware team, networking team, and so on to coordinate in person to get resources ready for the next step in the workflow, now being modeled and orchestrated in programming scripts and services. An iteration that used to be measured in days, is now being done in hours:

Table 2.1: Example of duration of LOC building a supported infrastructure[1]

Infrastructure (3-node)	Average build (hours)
Hyperconverged virtualization	7
Ceph	3
OpenStack	4

Infrastructure (3-node)	Average build (hours)
Kubernetes	5

[1]Hardware configuration varies from platform to platform. Some requires more physical resources, thus will take longer. Also, each platform has different configuration choices. See Lenovo's product guide on these for details.

3

Requirements

3.1 Functional requirements (use cases)

Lenovo Open Cloud is an infrastructure for infrastructure (aka. iFi). The primary use case of it is to build on-premise infrastructure — OpenStack, Ceph, Kubernetes, and hypervisors. This includes both the upstream versions and commercial offerings, in particular, the Red Hat products. Thus the LOC is catering to four types of users — cloud users, container users, storage users, and virtual machine users. This section describes the functional requirements for such an infrastructure to support these workloads.

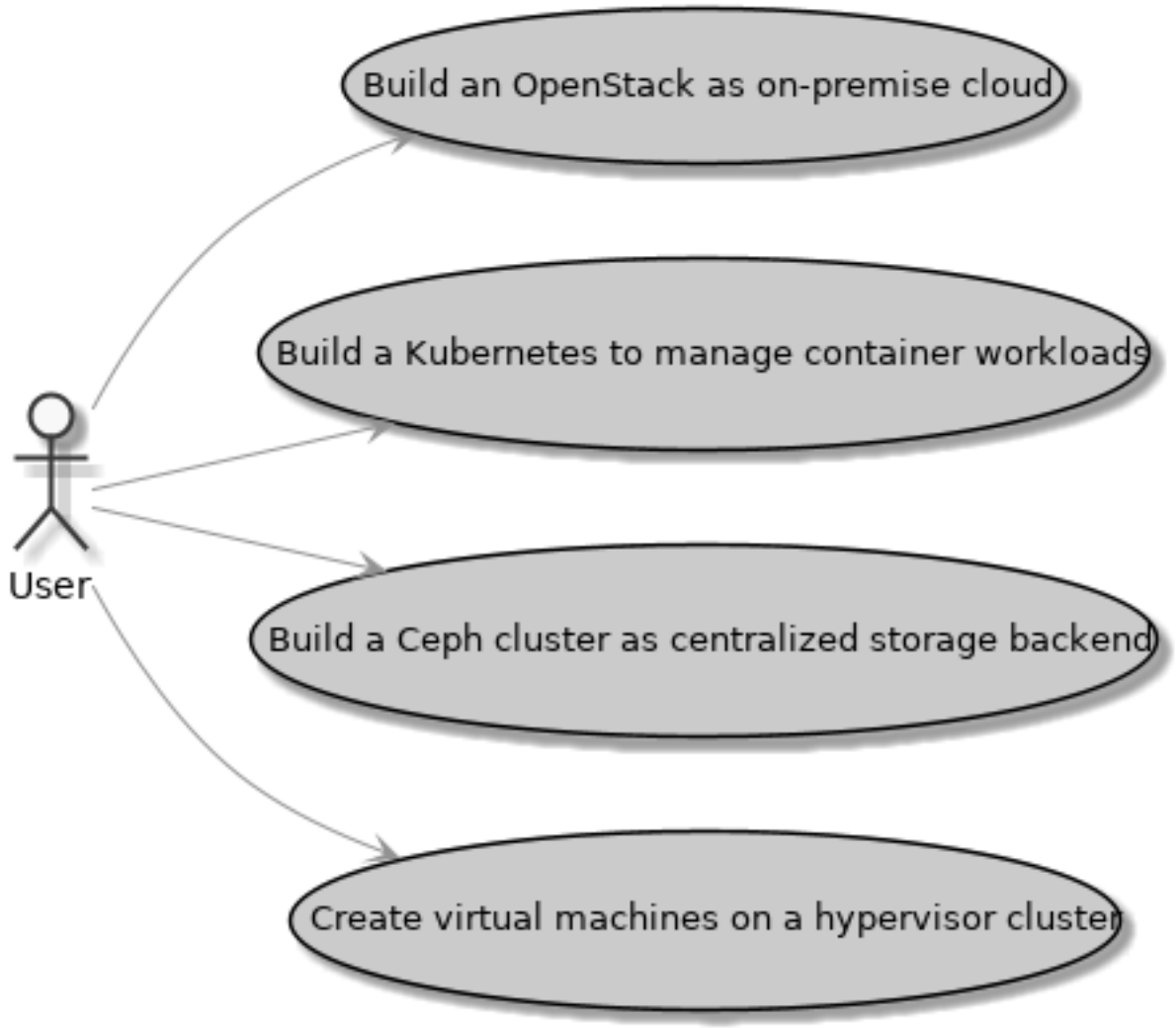

Figure 3.1: LOC RHHI functional requirement use cases

1. As a LOC cloud user, I want to build an OpenStack of minimal three-compute node cluster, so that I can offer an on-premise cloud to my users.
2. As a LOC storage user, I want to build a Ceph cluster of minimal three nodes and use it as storage backend for OpenStack and Kubernetes, so that data persistence and their life cycle will be managed centrally.
3. As a LOC container user, I want to build a Kubernetes over minimal three nodes, so that container workloads will be managed by it.
4. As a LOC virtual machine user, I want to build a hypervisor cluster of minimal three nodes,

so that I can deploy application in virtual machines that is portable to VMware and KVM.

Derived from these high level requirements, additional use cases can then be perceived to facilitate management and implementation:

5. As a LOC user, I want to manage a physical server without installing an operating system, so that I can upgrade/downgrade firmware, select boot device and boot order, turn power on/off, set OOB management IP, manage components including network interface card (NIC), RAID controllers, disks, and monitor these hardware for failures.

6. As a LOC user, I want to provision an operating system on a server with or without using DHCP, and have these OS images managed centrally, so that I can manage versions of these images, add new server to cluster, or re-commission a server for a different use.

7. As a LOC user, I want a service to manage the software packages on provisioned servers, so that I can install, upgrade or downgrade a version of a software knowing the dependencies are being taken care of.

8. As a LOC user, I want to detect a server as soon as it is plugged in on a monitored network, so that I can discover new server and kick off some executions against this server such as OS provisioning.

9. As a LOC user, I want to build hardware inventory of my data center before procurement shipment arrives, so that I can plan for their location, rack position, usage, and configurations that go with these decisions.

10. As a LOC user, I want to orchestrate a workflow that supports serial execution and parallel execution of programming scripts, so that I can develop functions and compose them into a workflow to accomplish complex automation.

11. As a LOC user, I want to simplify administration by having a single portal that aggregates managerial tasks and data of the LOC and infrastructures deployed on top of it, so that routine status report and tasks can be carried out from it.

12. As a LOC user, I want to run tests against the LOC infrastructure and its workloads as CICD, so that I can define and validate the health of the infrastructures, and continuously monitor and improve its soundness.

3.2 Non-functional requirements

The LOC is built with maintenance in mind, in particular, scalability and infrastructure as code (IaC).

3.2.1 Scalability

Infrastructure resources can expand or shrink. This is especially true to a cloud computing platform who, by definition, provides such elasticity.

1. As a LOC cloud/storage/container/VM user, I want to add new server to the OpenStack/-Ceph/Kubernetes/hypervisor cluster, so that the cluster can be scaled up.
2. Similarly, server can be removed from a cluster to scale down.
3. As a LOC user, I want server be re-purposed by software, so that I can choose to use it as an OpenStack compute node at one time, and as a Ceph storage node at another time.

3.2.2 Infrastructure as code (IaC)

Traditionally an infrastructure does not change much once it is built, and rebuilding it is often a daunting task. Changing configurations is always a risk because the infrastructure lacks built-in support of best practices that have been well known in quality control of software development — version control, automated testing, and CICD. LOC is addressing this gap by adopting and advocating the infrastructure as code approach:

1. As a LOC user, I want to define the infrastructure as code, so that I can manage its end state descriptively, and rebuild/re-configure it with deterministic result.

4

Architectural Overview

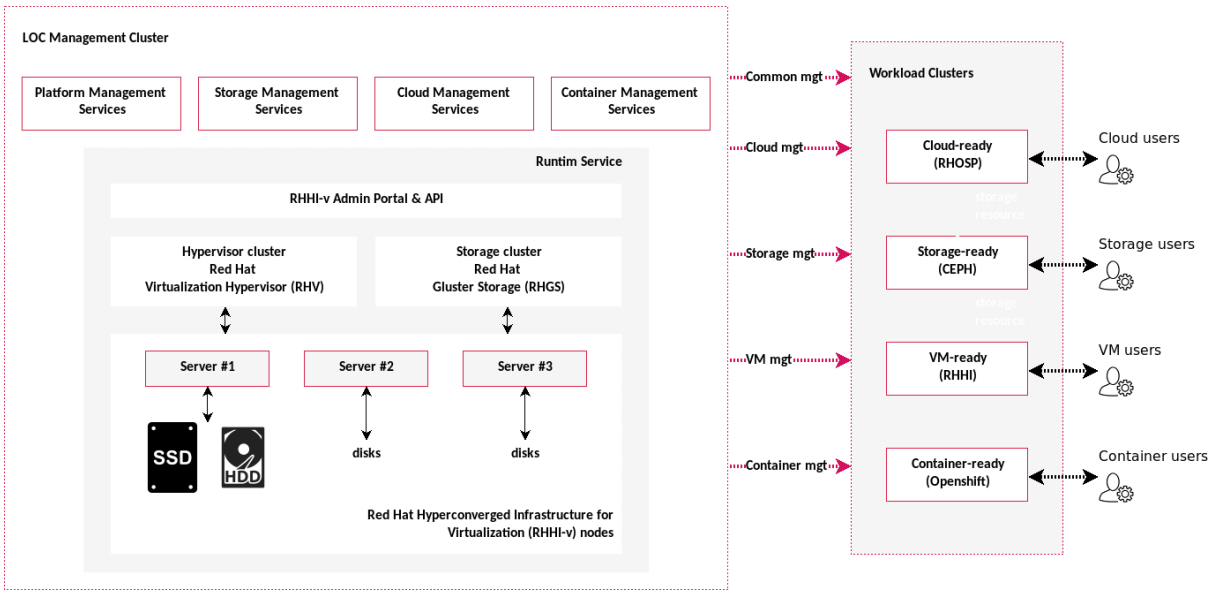

Figure 4.1: Lenovo Open Cloud architecture (on a three-node LOC RHHI)

Overall Lenovo Open Cloud architecture can be viewed in two sets of clusters: `management cluster` and `workload cluster`.

Management cluster includes two groups of management services: platform management and workload management.

Platform management includes a Runtime service that provides virtual machine capability, and a list of services (aka. Platform services) to manage infrastructure hardware and software. They cover a wide range of common use cases including server discovery and provisioning, software package management, configuration driven automation, workflow, inventory planning and tracking, and validation, monitoring and storage performance test of the

LOC and its supported infrastructures.

Workload management are infrastructure dependent services that support and manage the infrastructure deployed by LOC.

1. Storage services: Management functions of storage infrastructure. In this version LOC supports Ceph clusters, and Swift object stores.

2. Cloud services: Management functions of cloud infrastructure. In this version LOC supports both Red Hat Openstack 10 & 13.

3. Container services: Management functions of container infrastructure. In this version LOC supports Red Hat OpenShift 3.11.

Workload cluster refers to software and applications that run either on top of LOC or in parallel to LOC, while their management software, such as OpenStack controllers, are deployed as LOC services.

5

Component model

LOC infrastructure consists of four components: Lenovo Open Cloud RHHI, platform management services, configuration templates and deployed instances, and interfaces.

Figure 5.1: Lenovo Open Cloud component model

5.1 Hypervisor runtime service

This service refers to a virtualization environment. By default LOC uses the Lenovo Open Cloud RHHI Reference Architecture[1] as its runtime service. It is a hyperconverged infrastructure deployed over Lenovo ThinkSystem servers and switches. LOC RHHI offers four configurations:

1. 3-server configuration
2. 6-server configuration
3. 9-server configuration
4. 12-server configuration

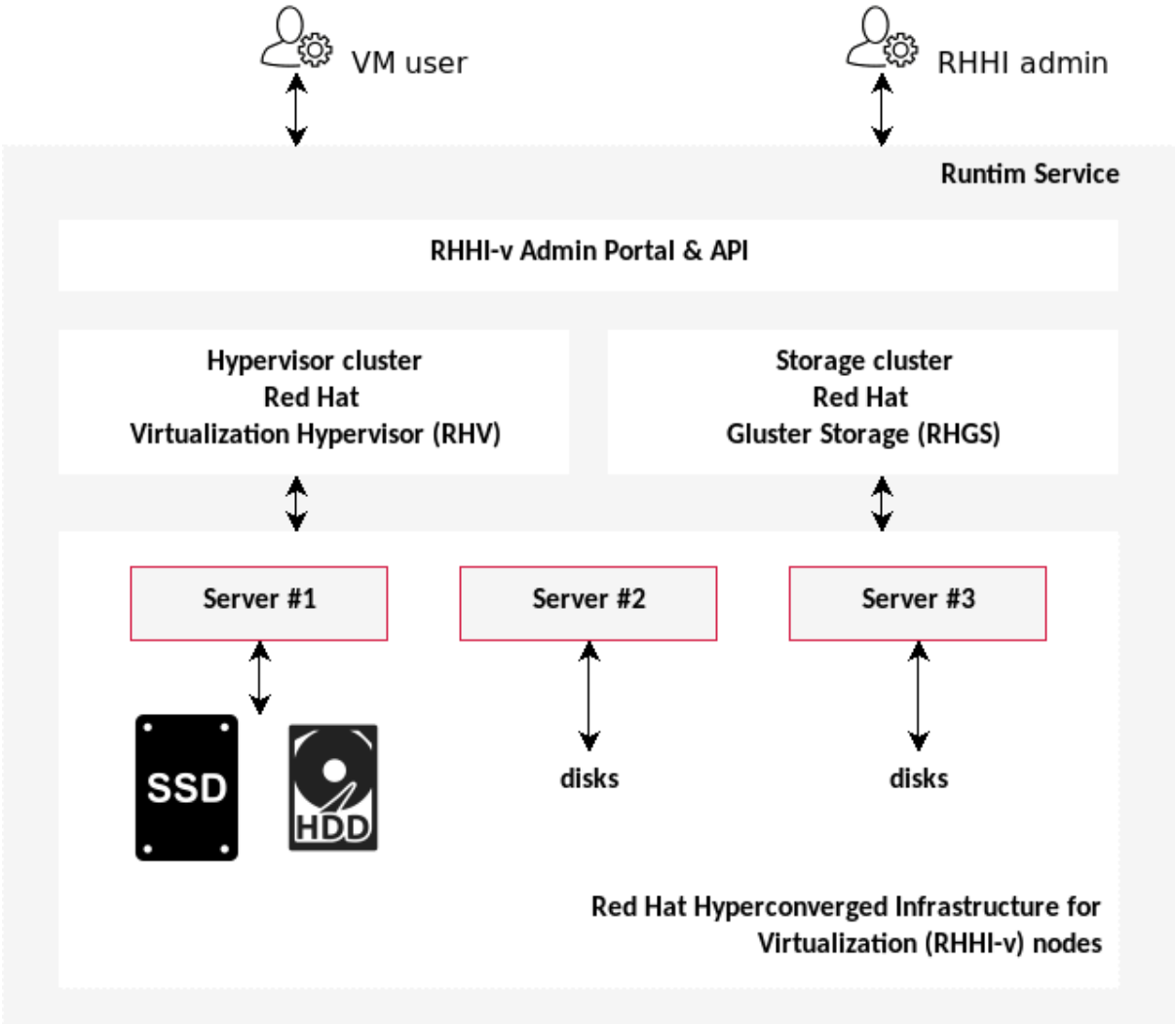

Figure 5.2: Lenovo Open Cloud RHHI architecture of a 3-server configuration

Primary storage is a Gluster cluster that takes servers' internal disks to form a distributed storage. By default data are replicated to all servers in the cluster.

Server's compute resources are virtualized through hypervisor, and workloads are deployed in virtual machines that can then be located on any of the hosts in the cluster. Cluster's management application is also in a virtual machine. It can cut over to another host automatically when its host is experiencing problem, thus providing an uninterrupted management capability. There is also a management CLI application identical on all cluster hosts. Thus accessing any of the host will give admin the administration capability.

For more information, see the Lenovo Open Cloud RHHI Reference Architecture[1].

5.2 Platform management services

There are ten platform management services. They provide administrative functions to support operation of LOC, including building and managing other infrastructure clusters, and repurposing hardware to extend or downsize footprint.

Table 5.1: LOC Platform management services and versions

Index	Service	Implementation Example	Version
1	Software repo	Red Hat Satellite	6.4
2	Automation	Red Hat Ansible Tower	3.4.2
3	Discovery	Lenovo Confluent	2.2.0
4	Inventory planning	Digital Ocean Netbox	Lenovo fork
5	OS deployment	Lenovo xClarity	2.3.0
6	Configuration repo	Gitlab CE	8.1.4
7	Validation	Jenkins, Elasticsearch, Logstash, Kibana	J(2.150.3), ELK (6.5.4)
8	OS Image	Lenovo distribution point	n/a
9	Single pane of glass	Red Hat Cloudforms	5.10.0.18

Index	Service	Implementation Example	Version
10	Launchpad	Red Hat Enterprise Linux	7.5

LOC Platform management services are in place to recognize common tasks we have seen in deploying and managing a cloud environment, such as extending the cluster with new hardware resources, provisioning of baremetal, deploying management software, maintaining configs, and administration with a single view of truth. Together they provide a comprehensive coverage of these needs, and many of them are being covered by some form of automation developed in LOC. Thus, using these services would speed up the onboarding process of a new resource, and centralizing management of cloud resources that admins would inevitably consider one way or another.

Automation service & Config repo service All LOC Platform management services except the Discovery Service and the Launchpad have an admin web portal and a RESTful API. Integration of services are orchestrated through the Automation service. Automation code, template, and default configs are maintained by the Config Repo service. The Automation Service can connect directly to the Config Repo to be in sync, and has the option to execute based on version, branch, or even a specific commit. This makes it possible to fine tune automation and try out new development.

Launchpad Execution of automation is proceeded in launchpad. Launchpad represents an isolated automation environment. Decoupling the two makes it flexible to compose different environments for different automation needs, such as version, library, and network requirement.

Discovery service & Inventory Planning service work together to maintain hardware inventory including procured future inventory and new ones just plugged into the grid for the first time. These hardware resources can then be commissioned by others based on their characteristics. In addition, a pre-determined automation path can be activated as soon as the hardware is detected so that its life cycle starts with little operator intervention.

Software Repo Service and OS Image Service provide the baseline software packages for all other services. OS Image, considering the size and distribution cycle of its contents, is mostly static and has no version control. Software Repo, on the other hand, is highly sophisticated and can integrate with upstream sources including vendor release points and open source projects as long as the package format is compatible.

OS Deployment service All software lands on hardeware somewhere. OS Deployment Ser-

vice makes managing Lenovo hardware easy. Things like firmware, boot order, RAID configuration are so fundamental that getting them right is the only way to build a robust stack. OS Deployment, thus, is here to bring these aspects of management under control.

Validation service monitors the status of LOC hardware and software so to keep LOC in a good shape. It can be customized to launch user tests that development and administration often require. It integrates seamlessly with the Config Repo Service and the Automation Service, bringing writing test code and running test code into a cohesive workflow.

Single pane of glass: takes advantage of the RESTful APIs of these services to provide a unified user experience through a single view point of a Lenovo Open Cloud. Through a flexible plug-in system, all LOC Platform management services with an API endpoint are integrated, and some common actions such as creating a new VM resource can even be directly configured and launched from the service. Itself also has a RESTful API, thus making LOC services ready for integration with other tools a user maybe using.

In this version of the Lenovo Open Cloud, many services are implemented by Red Hat software. The following discussions will be based on some unique concepts applicable to that Red Hat component, for example, the subscription of the Software repo service.

[LOC Platform management services overview][platform service overview]

LOC has built a set of recipes to orchestrate these services to work together to achieve highly sophisticated automation such as building an OpenStack as soon as compute nodes have been powered on even without have been set OOB IP address. Of course certain level of planning is still required. But these services and LOC's recipes have covered many common cases that once planning has filled in a configuration template with necessary information, the configuration will drive these services to carry out the rest of the execution from end to end.

Services can be used separately also as we will demonstrate in the following sections.

5.2.1 Software repository & life cycle management service

This service is the central location of software packages. It is the upstream source for all installers happening next. It handles software licenses, updates, and versioning for all third-party software. In Lenovo Open Cloud, it manages Red Hat subscriptions for all LOC hardware and software. A user can also use it to manage subscriptions for his/her workloads, making it a single source-of-truth of software licenses.

Out-of-box, all native LOC hardware and VMs are registered to this instance, and it provides a matching number of subscriptions and entitlements to support the life cycle of these subscribers.

This service is built upon the Red Hat Satellite[4]:

> Satellite is an on-premise alternative to trying to download all of your content from the Red Hat content delivery network or managing your subscriptions through the Customer Portal. From a performance side, it reduces hits to your network bandwidth because local systems can download everything they need locally; from a security standpoint, it can limit the risks of malicious content or access, even enabling entirely disconnected environments.
>
> Satellite is composed of a centralized Satellite Server. Depending on your data center setup, organization design, and geographic locations, you can have local Capsule Servers, which are proxies that locally manage content and obtain subscriptions, registration, and content from the central Satellite Server. (source: [4])

Key concepts of the software repo service are:

1. **Life cycle environment**: Logical group to make different versions of the same software available to members in the group. The name implies that the environments are defined to represent the *life cycle* of the application, eg. development to QA to production.
2. **Content view**: Logical group to segment software sources in the repo. For example, a view that includes the OpenStack files, but excludes OpenShift.
3. **Subscriptions**: These are software licenses that controls the number of endpoints one license supports, and its expiration date.

5.2.1.1 Life cycle environment

> Application life cycles are divided into life cycle environments, which represent each stage of the application life cycle. Life cycle environments are linked to form an environment path. You can promote content along the environment path to the next life cycle environment when required. (source: [5])

LOC software repo service offers three environments: development, test, and production, and the default environment path is `development to test to production`.

development Usually it maintains the newest version of packages. Once a version satisfies development, this version can be promoted to the `test` environment for more rigorous testing

and broader integration.

test Staging area for QA/CICD. Versions included in this environment are usually promoted from `development`, but are not necessarily the latest version.

production Environment supporting a stable version of applications.

In LOC, all hosts and virtual machines of the management cluster are registered to `production` life cycle environment.

If a new resource, including provisioned new baremetal server and new virtual machine, is created by LOC Automation Service, the resource will be registered to the `development` environment by default. This can be changed by specifying a different environment as an automation parameter.

5.2.1.2 Subscriptions

The following subscriptions are included to support Platform hosts and services. Additional subscriptions are needed for infrastructure such as Ceph storage and OpenStack.

Table 5.2: LOC Software repo and life cycle subscriptions of a 3-server configuration

Subscription	Qty
Red Hat Satellite	1
Red Hat Virtualization	3
Red Hat Gluster Storage	3
Red Hat Ansible Tower	1
Red Hat CloudForms	1
Red Hat Ceph Storage	1
Red Hat Enterprise Linux	13
Red Hat OpenStack Platform	3
OpenShift	1

5.2.1.3 Content views

> Content views are managed selections of content, which contain one or more repositories (yum, puppet, or containers) with optional filtering. These filters can be either inclusive or exclusive, and tailor a system view of content for life cycle management. They are used to customize content to be made available to client systems. (source: [6])

Content view is the way to make subscriptions available to managed hosts and virtual machines. LOC Platform provides the following content views:

Table 5.3: LOC Software repo and life cycle management RH Satellite Content Views

Content View	Subscription
platform	Red Hat Gluster Storage
	Red Hat Virtualization
rhel	Red Hat Enterprise Linux
satellite	Red Hat Satellite
dashboard	Red Hat CloudForms
ansible	Red Hat Enterprise Linux

5.2.1.4 Activation keys

> An activation key ensures that the content host is associated with the correct host group and that the content host consumes the correct subscription. Activation keys store a set of subscriptions that can be associated with content hosts without attaching these subscriptions directly. (source: [7])

LOC Platform names activation keys following the pattern: `<content view><life cycle ↪ > key`. Subscriptions under each key is consistent with what is included in the content view.

Table 5.4: LOC Software repo and life cycle management activation keys

Activation Key	Content View	Environment	Subscription
rhel development key	rhel	development	Red Hat Enterprise Linux

Activation Key	Content View	Environment	Subscription
rhel test key	rhel	test	Red Hat Enterprise Linux
rhel production key	rhel	production	Red Hat Enterprise Linux
satellite development key	satellite	development	Red Hat Satellite
satellite test key	satellite	test	Red Hat Satellite
satellite production key	satellite	production	Red Hat Satellite
dashboard development key	dashboard	development	Red Hat CloudForms
dashboard test key	dashboard	test	Red Hat CloudForms
dashboard production key	dashboard	production	Red Hat CloudForms
ansible development key	ansible	development	Red Hat Enterprise Linux
ansible test key	ansible	test	Red Hat Enterprise Linux
ansible production key	ansible	production	Red Hat Enterprise Linux
platform development key	platform	development	Red Hat Gluster Storage
			Red Hat Virtualization
platform test key	platform	test	Red Hat Gluster Storage
			Red Hat Virtualization
platform production key	platform	production	Red Hat Gluster Storage
			Red Hat Virtualization

5.2.1.5 Docker images

LOC Software repo service is also the central management for Docker images. It can be viewed as a local Docker registry so in a self-contained environment, LOC infrastructure can support container based applications out of the box.

5.2.2 Automation service

Automation service is an execution orchestrator. On one hand, it fully integrates with the Config Repo Service, thus having visibility to its files — code, scripts, templates, template instances — managed by the config repo. By changing where it points to, it can run a version of a file by git branch such as in testing a feature implementation, or git commit such as in debugging a specific code change.

On the other, it supports common workflow pattern — sequential, parallel, mixed, cron — while execution status and outputs are logged and trackable through this service.

This service is built upon [Red Hat Ansible Tower][tower]. Lenovo Open Cloud is shipped with a list of pre-defined automation that makes managing the infrastructure easy and efficient.

See product guide[8] for details.

5.2.3 Discovery service

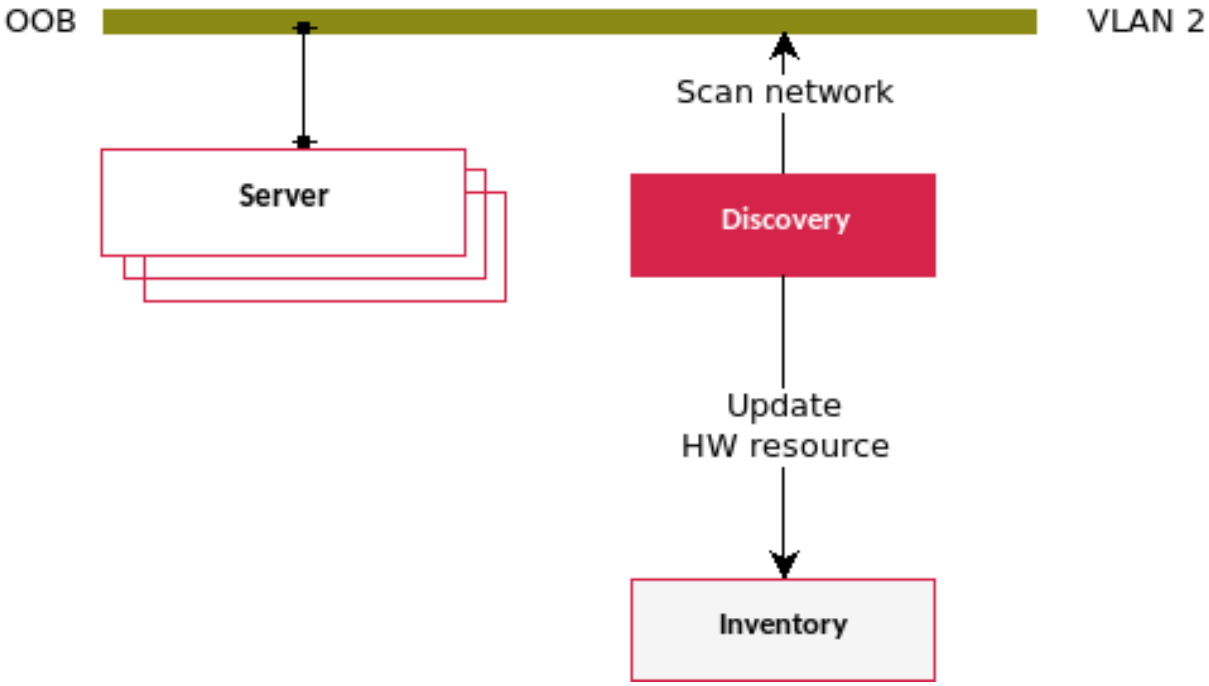

Figure 5.3: LOC Discovery service overview

Discovery service is a helper service. It continuously monitors the BMC network for new Lenovo hardware resources becoming available. When used together with the Inventory Planning Service, it can initialize a device inventory of hundreds even thousands of devices in a matter of seconds.

A role-based discovery can further reduce operator task by matching device UUID w/ inventory record, and kick off a pre-defined server configuration as soon as the device is connected to the network, including setting up the server's BMC IP without ever walking to the rack.

5.2.4 Inventory planning service

This service is the source-of-truth of infrastructure planning that handles the pre-on-board phase of hardware such as tracking device attributes (manufacturer, part number, model, etc.), allocating rack position, assigning it to a tenant, organizing it by site or by organization, designating a role if using role-based LOC discovery automation (see Discovery Service for details).

A unique feature of this service is to be the source-of-truth of the cable schema between LOC servers and switches. It can both take a schema config from the Config Repo Service to apply to the switch, or create a schema config based on what it sees on these hardware.

5.2.5 Configure & Automation repository service

Config repo service is the source-of-truth of configuration templates and scripts. All published LOC automation scripts are saved and version-controlled by this service.

LOC implements a data-driven design that configuration templates are inputs to LOC automation, whether it is to rebuild an LOC service, or even the entire LOC instance, or to create a new VM for your workload, or a new network matching to a VLAN in your environment. Check "LOC User Manual" for full list of use cases.

We call a template filled with a set of values "an instance of that template", and these instances can also be managed by the config repo service.

5.2.6 Validation service

Based on an ELK stack, the Validation service is a versatile service. It can do many things and do them well.

First, it integrates with the Config Repo Service, and runs all tests defined in the repo's `/LOC validation test suite` continuously to monitor the health of LOC instance.

Second, it is pre-configured w/ test cases for storage performance test using PTS that can be launched against any storage solution. An example of using it to test LOC RHHI's performance can be viewed in the Lenovo Open Cloud RHHI Reference Architecture[1] `Appendix: LOC RHHI Storage Performance`.

5.2.7 OS image service

OS image service is the source-of-truth of OS images including virtual machine images.

It is a common task to provision a baremetal server with an operating system, or to create a virtual machine using a cloud image or a snapshot of a previous VM dump. These files are usually large, not changed after their creation, but crucial to be managed properly so their contents are well known, and its deployment is deterministic.

Instead of relying on Internet resources to download on the fly, OS image service brings these artifacts inside LOC so not only using them becomes much more efficient, but admins now have a single point of management to maintain them, whether for licensing purposes or to upgrade to a new release.

In LOC, the following images are preloaded:

Table 5.5: LOC OS Image service preloaded images

Index	Filename	Description
1	rhel-server-7.4-x86_64-dvd.iso	For server provisioning
2	rhel-server-7.5-x86_64-dvd.iso	For server provisioning
3	lnvgy_sw_lxca_204-2.1.0_kvm_x86-64.qcow2	Installer for the OS Deployment service
4	rhel-server-7.5-x86_64-kvm.qcow2	Size SMALL RHV virtual machine base image
5	rhel-server-7.5-x86_64-kvm-500G.qcow2	Size MEDIUM RHV virtual machine base image

Index	Filename	Description
6	cfme-rhevm-5.10.0.18-1.x86_64.qcow2	Installer for the Single Pane of Glass service

5.2.8 Server config & OS deployment service

This service is a configurator of hardware including Lenovo baremetal servers and Lenovo switches.

Common tasks such as firmware updates, boot order, and switch port configuration are managed by this service.

Following LOC's data driven philosophy, this service is integrated with the Config Repo Service in which a config file, called `profile`, defines the state of hardware, and this service will apply this `profile` to make it happen.

5.2.9 Single pane of glass

This service is the LOC admin portal. It centralizes view of:

- ☞ hardware: by integrating with the Server Config & OS Deployment Service.
- ☞ software: by integrating with the Software Repo Service.
- ☞ virtual machines: by integrating with the Runtime Service.
- ☞ automation: by integrating with the Automation Service.
- ☞ inventory: by integrating with the Inventory Service.
- ☞ OS images: by integrating with the OS Image Service.
- ☞ LOC health: by integrating with the Validation Service.

5.3 Ceph management services

5.3.1 Ceph storage monitor

Lenovo Open Cloud comes with a value add monitoring solution, Lenovo Open Cloud Storage Monitor. LOC's Storage Monitor is automatically deployed as a VM in the LOC platform clus-

ter and is configured to monitor any Red Hat Ceph Storage Clusters deployed by Lenovo Open Cloud.

The LOC Storage Monitor provides a number of features to cloud and storage administrators:

- ☞ A dashboard displaying cluster utilization, server and disk status as well as health alerts
- ☞ Infrastructure tools to manage the cluster's hardware
- ☞ A GUI based tool to manage the cluster's pools and volumes
- ☞ Tools for backing up the cluster's data
- ☞ Alert configuration and change auditing

Additional details on Lenovo Open Cloud Storage Monitor may be seen in the Lenovo's Reference Architecture for Red Hat Ceph Storage[2].

5.4 Cloud management services

5.4.1 Red Hat Cloud Director

The Red Hat OpenStack Platform uses Director as the toolset for installing and managing a production OpenStack environment, e.g. overcloud. Each overcloud instance requires a Director instance for installation, management and maintenance operations. In LOC solution, the Red Hat OpenStack Platform Director is deployed as a VM on the LOC platform cluster. Additional information about Director usage can be found in Director Installation and Usage[9].

5.4.2 Cloud controllers

OpenStack Controllers are Nodes that provide administration, networking, and high availability for the OpenStack environment. An ideal OpenStack environment recommends three of these nodes together in a high availability cluster.

The Controller nodes are deployed with a collection of interacting services that control compute, storage, and networking resources, such as OpenStack nova, horizon, neutron, glance, etc..and other software components like MariaDB, HAproxy and pacemaker. The three controller nodes can be deployed as VM on LOC platform and work in HA mode.

6

Operational Model

A LOC deployment is shown in fig. 6.1. Initial deployment can start with three HCI ndoes. User can then choose to activate Ceph and/or OpenStack control plane to build Ceph or OpenStack infrastructure with added servers. Three Ceph nodes and three OSP compute nodes are default configuration for illustration purpose. In reality, there can be much more servers in a Ceph or OpenStack cluster than three.

Further, the value of LOC is to construct these infrastructure quickly and with flexible configuration. Thus, it is not uncommon to have LOC build multiple OpenStack instances with different number of compute resources so to cater for different user groups — a three-node OpenStack with virtual Ceph for development, and a larger deployment with hundreds of compute nodes and an external Ceph for production.

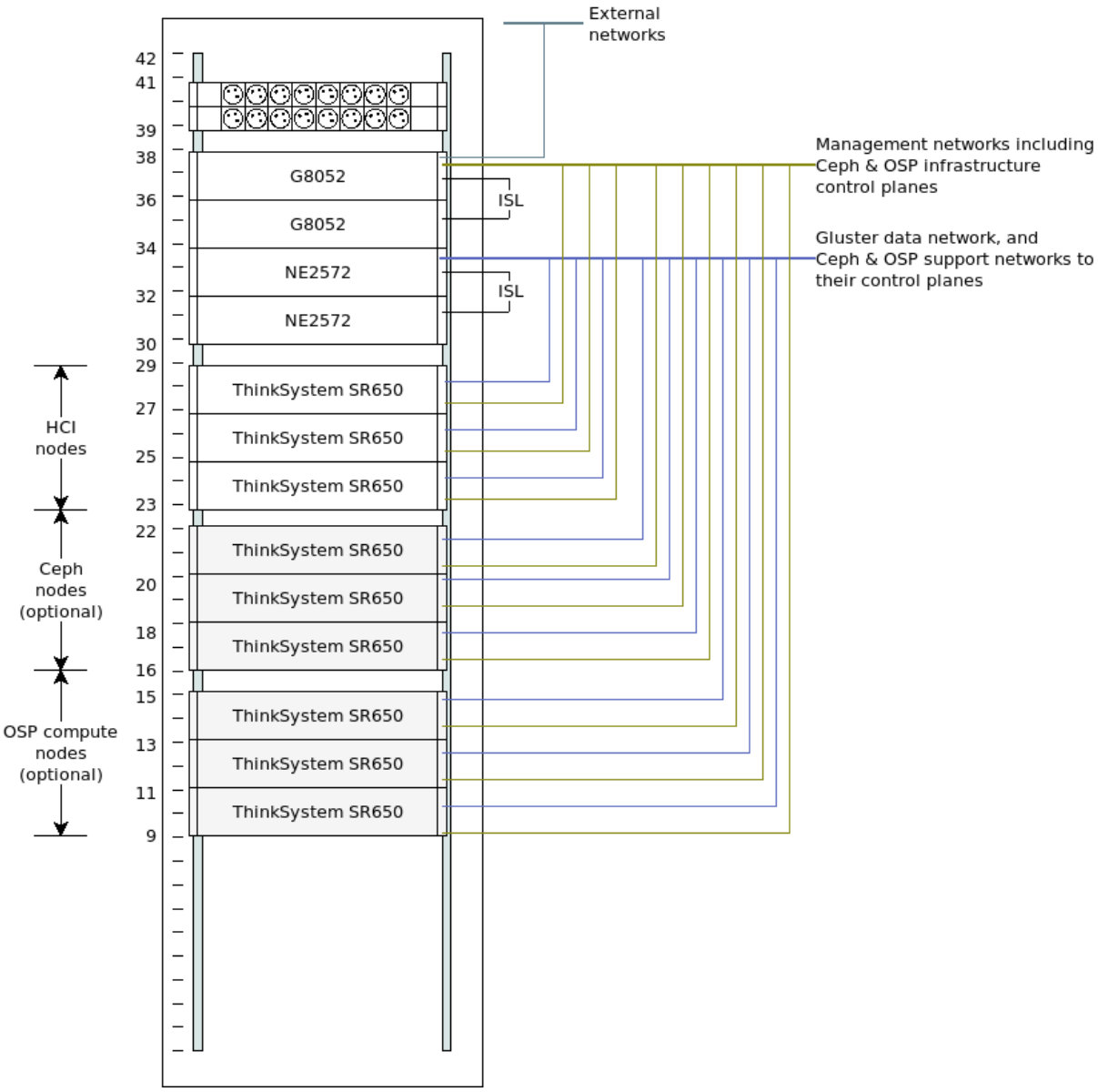

Figure 6.1: An example deployment of LOC on a 42U rack

6.1 Virtual control plane

All LOC services are deployed as one or more virtual machines. Default runtime for the virtual machines is a LOC RHHI[1]. This is the key feature of using LOC to build other infrastructure — infrastructure control planes are now virtual.

6.1.1 VM sizing

Table 6.1: LOC management services VM sizing

Index	LOC Management Service	CPU Cores	Mem (GB)	Storage (GB)
1	Software repo(upstream)	4	20	1000
	Software repo(downstream)	4	20	500
2	Automation	2	4	100
3	Server config & OS deployment	4	8	200
4	Inventory planning	2	4	50
5	Discovery	2	4	50
6	Config repo	2	4	100
7	Single pane of glass	4	8	100
8	Validation	4	8	300
9	OS Image	2	4	500
10	Launchpad	2	4	50

Also, refer to Ceph control plane and OpenStack control plane for details of their virtual machine sizing and specific requirements.

6.1.2 License

By default all platform services are enabled and covered by proper licenses. Ceph management services and OSP management services are installed but are disabled, meaning that they can not yet build Ceph and OpenStack infrastructure without first installing licenses.

1. To enable Ceph only storage infrastructure, see Ceph only software BOM.
2. To enable OpenStack only infrastructure, see OpenStack software BOM.
3. To enable HCI Ceph and OpenStack infrastructure, see HCI Ceph and OpenStack BOM.

Once they have been enabled, virtual control plane can build new infrastructure from scratch, or

adding or removing hardware from an existing cluster.

6.1.3 LOC RHHI as the Runtime service

Lenovo Open Cloud RHHI[1] is the default Runtime service for LOC. VM's primary storage is the gluster. There is additional NFS storage on each host, and VM has the option to be affinitized to a host:

Table 6.2: LOC management services VM on LOC RHHI

Index	LOC Management Service	NFS (GB)	Boot Options	Affinity (Y/N)	Enable Auto Migration
1	Software repo	n/a	HD	No	Yes
2	Automation	n/a	HD	No	Yes
3	Server config & OS deployment	n/a	HD	No	Yes
4	Inventory planning	n/a	HD	No	Yes
5	Discovery	n/a	HD	No	Yes
6	Config repo	n/a	HD	No	Yes
7	Single pane of glass	n/a	HD	No	Yes
8	Validation	n/a	HD	No	Yes
9	OS Image	n/a	HD	No	Yes
10	Launchpad	n/a	HD	No	Yes

Also, refer to Ceph control plane and OpenStack control plane for details of their virtual machine requirements.

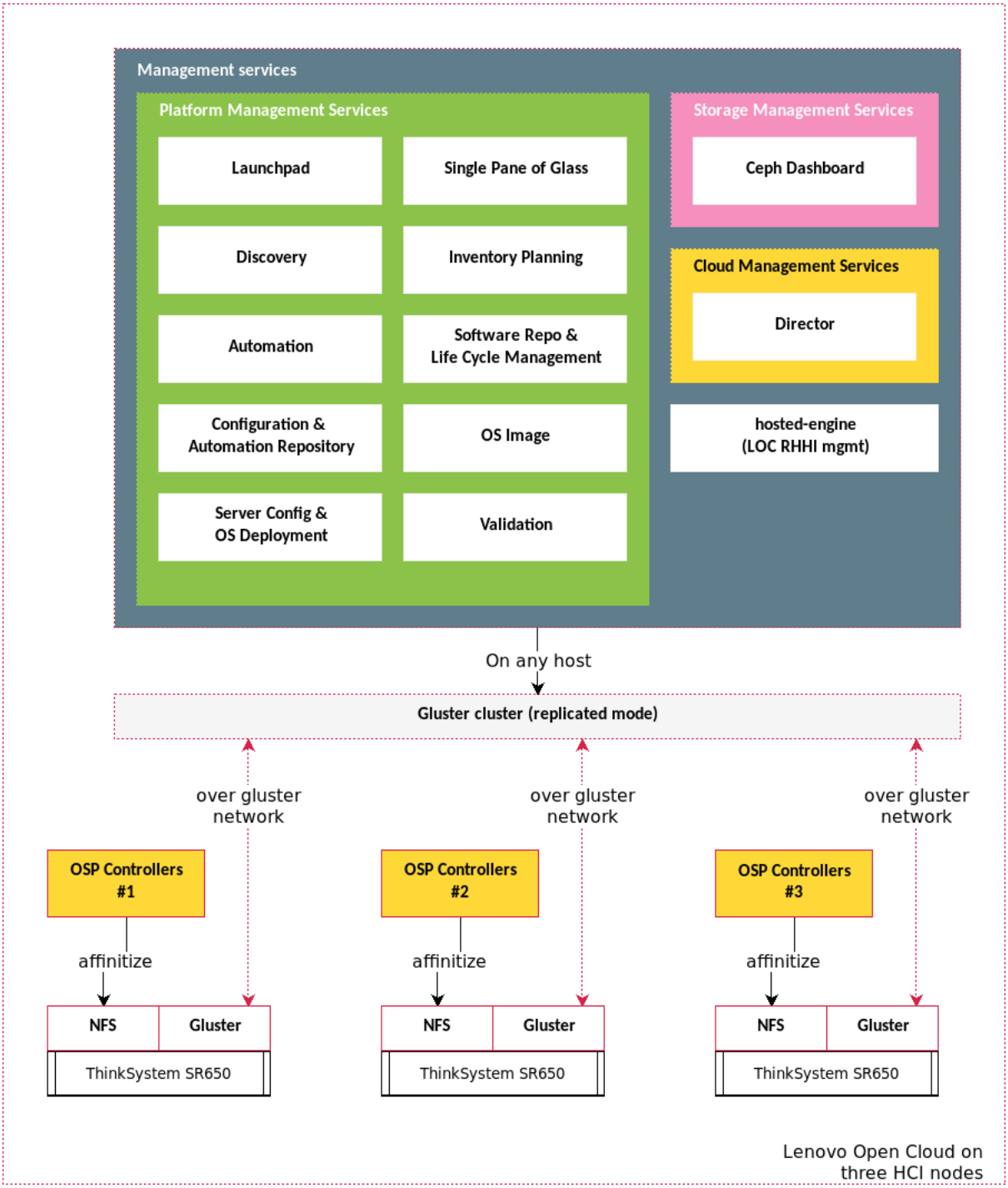

Figure 6.2: LOC service deployment on LOC RHHI as the Runtime service

`Hosted-engine` has the capability to cut over to a different server if its current host is ex-

periencing failure. Other LOC services, however, **do not have this feature in this version**. Therefore, it is critical to select qualified hardware components (see the Three-node LOC RHHI hardware BOM section for details) to compensate for this limitation.

6.2 Hardware

6.2.1 Lenovo SR650 Servers

Figure 6.3: Lenovo ThinkSystem SR650

The Lenovo ThinkSystem SR650 server is an enterprise class 2U two-socket versatile server that incorporates outstanding reliability, availability, and serviceability (RAS), security, and high efficiency for business-critical applications and cloud deployments. Unique Lenovo AnyBay technology provides the flexibility to mix-and-match SAS/SATA HDDs/SSDs and NVMe SSDs in the same drive bays. Four direct-connect NVMe ports on the motherboard provide ultra-fast read-/writes with NVMe drives and reduce costs by eliminating PCIe switch adapters. Plus, storage can be tiered for greater application performance, to provide the most cost-effective solution.

Combined with the Intel® Xeon® Scalable processors product family, the Lenovo ThinkSystem

SR650 server offers a high density combination of workloads and performance. Its flexible, pay-as-you-grow design and great expansion capabilities solidify dependability for any kind of virtualized workload, with minimal downtime. Additionally, it supports two 300W high-performance GPUs and ML2 NIC adapters with shared management.

The Lenovo ThinkSystem SR650 server provides internal storage density of up to 100 TB (with up to 26 x 2.5-inch drives) in a 2U form factor with its impressive array of workload-optimized storage configurations. The ThinkSystem SR650 offers easy management and saves floor space and power consumption for the most demanding storage virtualization use cases by consolidating the storage and server into one system. The Lenovo ThinkSystem SR650 server supports up to twenty-four 2.5-inch or fourteen 3.5-inch hot-swappable SAS/SATA HDDs or SSDs together with up to eight on-board NVMe PCIe ports that allow direct connections to the U.2 NVMe PCIe SSDs. The ThinkSystem SR650 server also supports up to two NVIDIA GRID cards for AI or media processing acceleration.

The SR650 server supports up to two processors, each with up to 28-core or 56 threads with hyper-threading enabled, up to 38.5 MB of last level cache (LLC), up to 2666 MHz memory speeds and up to 3 TB of memory capacity. The SR650 also support up to 6 x PCIe slots. Its on-board Ethernet solution provides 2/4 standard embedded Gigabit Ethernet ports and 2/4 optional embedded 10 Gigabit Ethernet ports without occupying PCIe slots. All these advanced features make the server ideal to run data and bandwidth intensive VNF workload and storage functions of NFVI platform.

For more information, see product guide[10].

6.2.2 Lenovo RackSwitch G8052 (1Gb)

Figure 6.4: Lenovo RackSwitch G8052

The Lenovo RackSwitch™ G8052 (as shown in fig. 6.4) is a top-of-rack data center switch that delivers unmatched line-rate Layer 2/3 performance at an attractive price. It has 48x 10/100/1000BASE-T RJ-45 ports and four 10 Gigabit Ethernet SFP+ ports (it also supports 1

GbE SFP transceivers), and includes hot-swap redundant power supplies and fans as standard, which minimizes your configuration requirements. Unlike most rack equipment that cools from side-to-side, the G8052 has rear-to-front or front-to-rear airflow that matches server airflow.

For more information, see product guide[11].

6.2.3 Lenovo ThinkSystem NE2572 (25Gb)

Figure 6.5: Lenovo RackSwitch NE2572

The NE2572 is optimized for enterprise data centers with features, such as:

- ☞ Software Defined Data Center interfaces ensure smooth interoperability with popular enterprise management applications.
- ☞ Open Architecture: NE2572 supports automation and control while

- ☞ Lenovo Network Orchestrator automates vlan changes to improve. integrating with industry-standard application and APIs, such as REST, OpenStack, OpenContrail, and Ansible.
- ☞ Resiliency: NE2572 offers outstanding fault tolerance with one of the quickest high-availability failovers in the industry so running applications are less likely to require a restart.

For more information, see product guide[12].

6.3 How LOC builds an infrastructure

In LOC's view, building an infrastructure follows a common pattern:

1. Resource planning — servers, networks, roles, and tenants
2. Configure server — firmware, RAID, boot order
3. Provision OS
4. OS customization — `cloud-init`, package/subscription manager, network (remote administration)
5. Infrastructure installer
6. Infrastructure post-install configuration
7. Infrastructure testing
    1. function/validation
    2. health monitoring
    3. specific interest — performance, stress

LOC's management services are designed to fit this pattern following these steps:

1. Save hardware info in the Inventory service, such as manufacturer, model, serial number.
2. User uses Inventory service to plan for hardware's usage by assigning rack position, tenant it belongs to, and its network assignment — IP addresses, which interface on which subnet, and vlans.
3. User assigns a `tenant`, `role` and `status` to a server in the Inventory service:
    1. `tenant`: eg. `cloud xyz`, meaning that this server will be a resource for this tenant. The type of resource is defined in `role` (see next).

   2. `role`: eg. `OpenStack compute`, that determines which configs and which automation procedure will be applied to the server.
   3. `status == available`: marks the server as new resource.

4. Automation service calls the Discovery service to scan the BMC network, and uses detected server UUID to query the Inventory service for its `role` and `status`.

5. If found matching servers that are `available`, Automation service calls the Inventory service for server's BMC network assignments, and apply them to the Discovery service to set up OOB management of the server.

6. Automation service retrieves config patterns by the `role` from the Config repo service, and applies it to the OS Deployment service to configure server — firmware upgrade/downgrade, RAID, boot order.

7. Inventory service retrieves server component information such as disks, RAID controller, network controller, firmware, as part of server's meta data.

8. Automation service retrieves OS config by the `role` from the Config repo service, and applies to the OS Image to setup OS image to use.

9. Automation service retrieves network assignment the server, such as IP addresses, root user name, password, from the Inventory service, and calls the OS Deployment service to provision the server, including `cloud-init` to set up user accounts.

   By this point, server is provisioned and is SSH ready.

10. (optional) Automation service registers the new server to the Software repo & life cycle management service if the server is to use this service as software package upstream.

11. Automation service changes server's status to `provisioned` in the Inventory service.

12. Automation retrieve infrastructure config files by the `tenant` from the Config repo.

13. Automation service starts a pre-defined workflow to install infrastructure or to extend existing infrastructure with these servers.

14. Automation service sets servers' status to `in-use`.

15. Automation service updates tenant's infrastructure config files in the Config repo.

16. Automation service updates the Inventory service w/ `tenant` config ID, type, and default test cases.

    User can activate additional test cases for the infrastructure in the Inventory service.

    Test can set to `run once` or `CRON`.

17. Automation service retrieves infrastructure configs from the Config service, and passes it to the Validation service to run validation tests as sanity check.

18. Automation service retrieves test cases by `tenant` from the Inventory service, and calls Validation service to run tests.

19. For new infrastructure, the Automation service registers the infrastructure in the Single pane of glass.

20. Validation service runs CRON tests to continuously monitor the health of the infrastructure.

6.4 Build a storage infrastructure

Storage infrastructure in the cloud can vary widely depending on the application. Lenovo Open Cloud's storage options are designed to give the end user flexible options to easily support a variety of needs. Given that LOC is built upon Red Hat and OpenStack technologies many different storage backends are available to the end user. For this release of LOC we focus on Gluster and Ceph storage to support LOC's workloads.

In this section we will introduce Ceph storage and how the integration of storage is simplified when deploying an LOC environment.

6.4.1 Storage virtual control plane

Ceph control plane includes a Ceph dashboard that monitors the status of Ceph cluster. It is a VM deployed on the LOC Platform:

Table 6.3: LOC Platform VM requirement for Ceph deployment

VM	CPU Cores	Mem (GB)	Gluster (GB)	NFS (GB)	Boot Options	Affinity (Y/N)	Enable Auto Migration
Ceph dashboard	2	8	60	n/a	HD	No	Yes

6.4.2 Deploy a new Ceph cluster

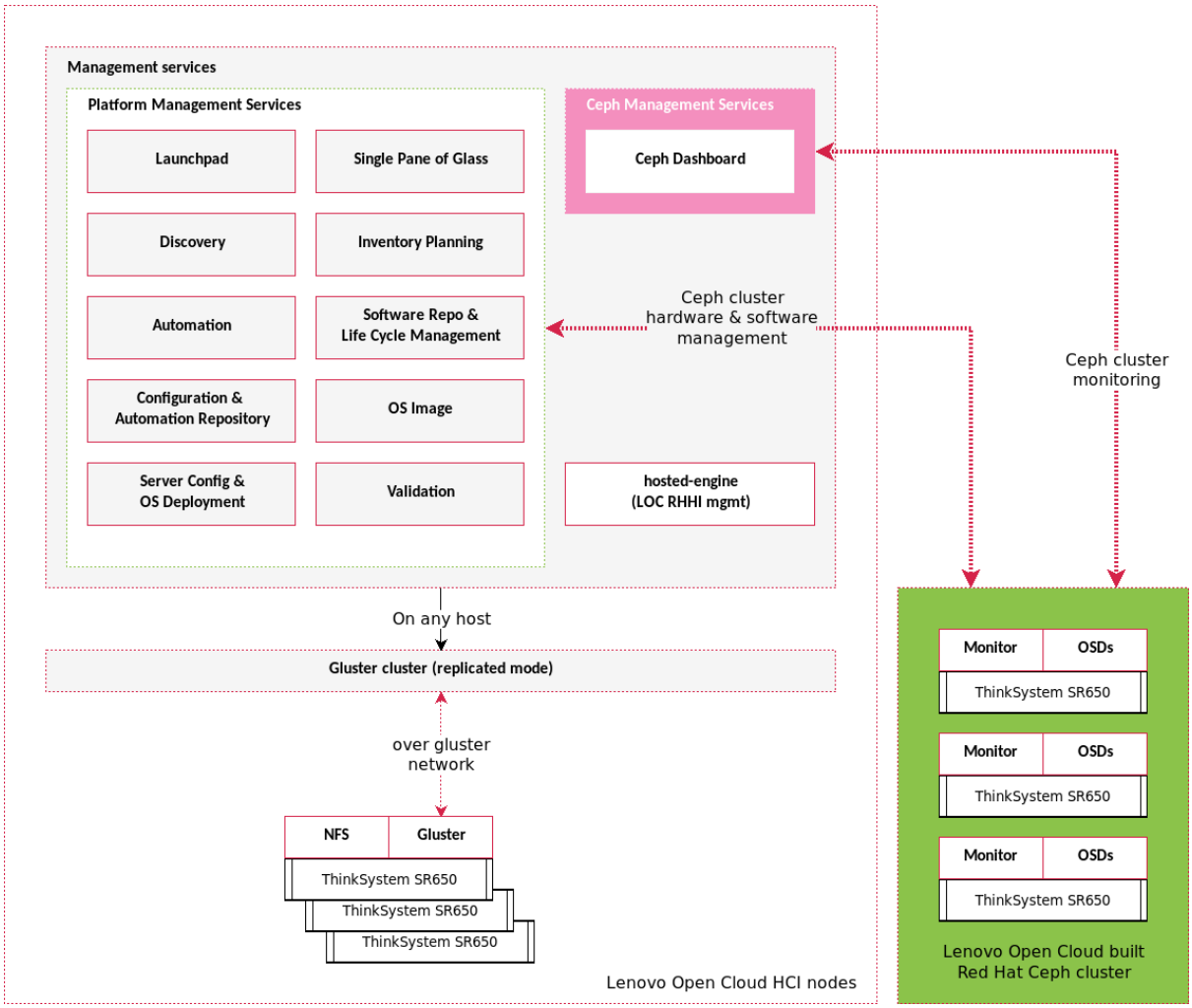

Figure 6.6: Deploy a new Ceph cluster by LOC

Lenovo Open Cloud uses Red Hat Ceph Storage to provide scalable, open, software-defined storage to Red Hat OpenStack Platform. Like Gluster storage, Red Hat Ceph Storage is built from a number of Ceph storage nodes to provide fault-tolerant storage.

Additional details on Lenovo Open Cloud's RedHat Ceph Storage support may be seen in the Lenovo's Reference Architecture for Ceph Storage[2]. For more details on Red Hat Ceph Storage integration into RHOSP, see the Red Hat's OpenStack Platform documentation[13].

The automation service can trigger an automated workflow on demand to create a Ceph Cluster for Lenovo Open Cloud platform. The automation workflow pulls in the necessary configuration files from the config repo to determine the size, disk configuration, monitor placement and target OpenStack cloud for the requested cluster. The automation workflow then requests the Runtime service to create a VM in the LOC platform to support the Ceph Dashboard. The Runtime service also requests baremetal resources from the Inventory service to serve as the Ceph Storage Nodes.

Once the VM/BM resources are ready, each node gets an operating system deployed to it and the node's network interfaces and VLANs are configured.

After configuring each node as a Storage Node the automation then enables the appropriate software repos to support installing Red Hat Ceph Storage 3.0. The binaries and libraries for Ceph support are installed and Ceph is deployed using the disks requested via the config repo in each node, creating a cluster across the BM resources made available by the Inventory service.

Once Lenovo's storage automation has verified that the Ceph cluster has been properly deployed it continues on to create storage pools for each of the OpenStack components which require Ceph storage. The following pools are created by default:

1. cloudx_nova
2. cloudx_volumes
3. cloudx_images
4. cloudx_backups
5. cloudx_metrics

Note:

☞ x: cloud index, eg. 1-9. An LOC Platform can support multiple cloud instances. Ceph storage pools follow this naming convention so that it is possible to easily associate a Ceph pool with a cloud.

The names for each of the pools are descriptive of their purpose:

1. nova - backs ephemeral storage for Nova instances to enable VM migration
2. volumes - storage for the Cinder volume service
3. images - storage for the Glance image service
4. backups - storage to be utilized for backing up volumes
5. metrics - storage utilized by OpenStack's Telemetry Service

After the pools have been created the storage automation creates a client with a name that is indexed on the cloud number like the pools. So, cloud 1, for instance will have the cloud1.client user created. A key is created for accessing the cluster and the appropriate authorities are configured for the cloud to access each pool.

All of the information necessary for accessing the newly created cluster is then added to the config repo and may be accessed, based on the cloud index, to subsequently deploy a cloud.

The final step in deploying a Ceph cluster is to add the cluster to the Ceph Dashboard. So, the last step the automation runs is the necessary scripting on the dashboard VM to discover the Ceph cluster and start monitoring it.

6.4.3 Extending an existing Ceph cluster

To extend an existing cluster much of the same process is followed as when creating a new Ceph cluster. The Runtime service will pull the updated configuration files from the config repo. Once the files are processed a barmetal resource will then be requested from the Inventory service. The new baremetal resource will be prepared to be a Ceph node and the node is then added into the Ceph cluster. Once the automation has determined that the node has been successfully integrated, storage will be rebalanced to make use of the new node and the Ceph Dashboard will be updated. Note, when extending the cluster that the pool and client creation steps do not need to be repeated.

6.5 Build a Red Hat OpenStack Platform 13 infrastructure

Red Hat OpenStack Platform 13[13] provides an integrated solution to create, deploy, and scale a secure and reliable private OpenStack cloud as one of the supported types of workload cluster in Lenovo Open Cloud platform. Red Hat OpenStack Platform 13 is based on the OpenStack Queens release. It is packaged so that available physical hardware can be turned into a private, public, or hybrid cloud platform that includes:

- ☞ Fully distributed object storage
- ☞ Persistent block-level storage
- ☞ Virtual machine provisioning engine and image storage
- ☞ Authentication and authorization mechanisms
- ☞ Integrated networking
- ☞ Web browser-based interface accessible to users and administrators

Lenovo's solution supports deploying OpenStack cloud with various storage configurations. Furthermore, with Red Hat OpenStack 13, the controller cluster of the OpenStack cloud supports two approaches of deployment. The controller nodes can be installed to baremetal nodes, as well as virtual machines spread across multiple physical instances.

In this reference architecture, we can fully benefit from this feature by deploying the OpenStack cloud controller nodes as VMs on the LOC Platform Runtime Service. This is extremely useful to decrease TCO for the workload cluster, and provides users with a scalable cloud-ready infrastructure with much easier operational management. The virtualized control plane of Open Stack cloud is easy to rebuild, maintain and replace. Furthermore the 3-node configuration Platform Runtime Service of LOC guarantees the HA capability for cloud controllers. Virtualized controller deployment is most suitable for medium scale clouds of less than 40 nodes.

Further, multiple cloud clusters are supported to locate on one LOC platform. This enables customer to easily deploy isolated clouds with different configurations for various business purposes.

6.5.1 OpenStack deployment configurations

LOC support three OpenStack deployment configurations.

Deployment without a Ceph Cluster(Swift collocated):

Table 6.4: LOC OSP 13 deployment with Swift collocated

Description	VM/BM	Qty
Director	VM	1
Cloud Controller with Swift service colocated(HA)	VM	3
Cloud compute	BM	1

Deployment with External Ceph Cluster:

Table 6.5: LOC OSP 13 deployment with external Ceph

Description	VM/BM	Qty
Director	VM	1
Cloud Controller(HA)	VM	3
Cloud compute	BM	1
Ceph Storage Node	BM	3

Deployment with Hyperconverged Ceph Cluster:

Table 6.6: LOC OSP 13 deployment with HCI Ceph

Description	VM/BM	Qty
Director	VM	1
Cloud Controller(HA)	VM	3
HCI Ceph node	BM	3

6.5.2 OpenStack virtual control plane

In order to deploy OpenStack director and controller nodes as VM instances on LOC platform. The VM instances should be properly configured. Below is the configuration Lenovo recommends for each of the VM roles.

Table 6.7: LOC Platform VM requirement for OSP 13 deployment

VM	CPU Cores	Mem (GB)	Gluster (GB)	NFS (GB)	Boot Options	Affinity (Y/N)	Enable Auto Migration
Director	4	16	80	n/a	HD	No	Yes
Cloud controllers (HA)	8	16	100	n/a	PXE as primary, and HD as secondary	Yes	No
Cloud controllers w/ Swift service colocated (HA)	8	16	100	50	PXE as primary, and HD as secondary	Yes	No

Considerations to take:

1. Storage: The LOC Platform Runtime Service built on top of RHHI exposes two types of storage that can be shared by upper VM instances.

    1. **Red Hat Gluster storage**: The RHHI platform will deploy a Gluster cluster by default. Several storage domains are created on Gluster storage for different purposes. The “vmstore” is the default storage domain for VM workloads.

    2. **NFS storage**: The NFS storage spaces are created from the peripheral disks on each RHHI node as the optional storage for special LOC workloads. As OpenStack Swift has built-in features for 3 member replicas, and Gluster also applies replica mechanisms for data protection, we will specify NFS storage on each RHHI node for OpenStack Swift data to avoid double replication penalty.

    You can refer to the Lenovo Open Cloud RHHI Reference Architecture[1] for detailed information about RHHI storage.

2. HA: In order for the OpenStack controller HA feature to function properly on LOC platform, we will also disable the VM auto migration on RHHI for controller nodes, and enable host affinity to make sure all three controller nodes run on different RHHI physical nodes. This guarantees the OpenStack cloud to sustain two RHHI physical nodes in case of an outage.

6.5.3 Deploy a Red Hat OpenStack

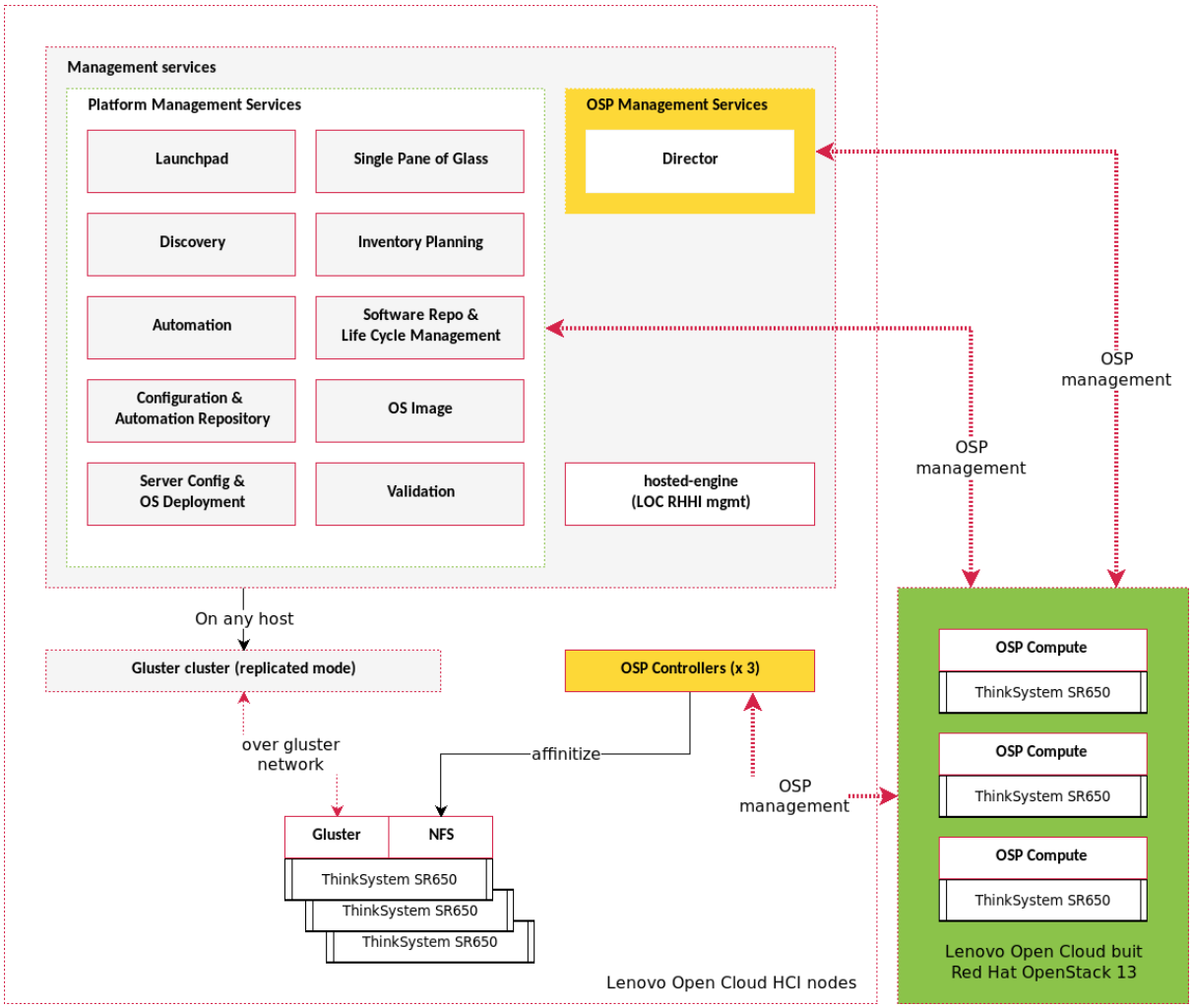

Figure 6.7: Deploy a new Red Hat OpenStack 13 by LOC

The automation service can trigger a cloud automation workflow on demand to deploy a cloud. It will load the scripts and heat-templates required by Director from Config repo service. Config repo

service will also provide platform service config that contains information on the endpoint info of platform services, and input from the system adminstrator for the expected cloud configuration, and VM and BM requirements used for cloud nodes.

The cloud deployment workflow will then request the Runtime service to create and configure the necessary virtual machines for its controller and Director. It will in parallel work with the Inventory service to allocate and provision the baremetal compute nodes for the cloud.

After all VM/BM resources are ready, the cloud deployment will be executed via Director with rendered heat-templates to reflect the expected cloud configuration.

The whole automation process can be orchestrated in a simple and flexible way that can efficiently improve the IT work for cloud deployment, scaling, operational management, and maintenance.

7

Deployment considerations

7.1 Lenovo Open Cloud logical networks

LOC's network is based on the same set of logical networks defined in the Lenovo Open Cloud RHHI Reference Architecture[1].

Table 7.1: LOC logical network and VLANs

Logical Network	VLAN	Description
Campus internal	1	For public access of admin portal and API
BMC	2	OOB management
Physical server management	3	SSH admin access
RHHI provisioning	10	New baremetal server provisioning network
OVIRT management	100	LOC RHHI management portal and API
GlusterFS	400	LOC RHHI hyperconverged storage

7.2 VLANs of the LOC management services

For the LOC management services, it is important that they have access to the correct network in order to fulfill its function:

Table 7.2: Platform management services to VLAN Mapping

Index	Service	1	2	3	10	100
1	Runtime	x				x
2	Software repo	x		x		x
3	Automation	x		x		
4	Discovery	x	x			
5	Inventory	x	x	x		
6	OS deployment	x	x			
7	Configuration repo	x				
8	Validation	x	x	x		x
9	OS Image	x				
10	Single pane of glass	x				x
11	Launchpad	x	x	x	x	x

Considerations for such arragnements are:

1. Runtime service:
   - ☞ `ovirt management` (V100): managing LOC RHHI servers, and serving the admin portal and its API endpoint.
2. Single pane of glass:
   - ☞ `Campus internal` (V1): integration point to other services except the Runtime service.
   - ☞ `ovirt management` (V100): integration point w/ the Runtime service.
3. Software repo:
   - ☞ `Campus internal` (V1): serving admin portal and its API endpoint.
   - ☞ `Physical server management` (V3): remote administration of LOC physical servers after OS provisioning.
   - ☞ `ovirt management` (V100): integration point w/ the Runtime service.
4. Automation:
   - ☞ `Campus internal` (V1): serving admin portal and its API endpoint.

- ☞ `Physical server management` (V3): remote administration of LOC physical servers using Ansible.

5. Server config:
   - ☞ `Campus internal` (V1): serving admin portal and its API endpoint.
   - ☞ `BMC` (V2): server OOB management, and OS provisioning.
6. Inventory planning:
   - ☞ `Campus internal` (V1): serving admin portal and its API endpoint, and accessing LOC switches for port configuration and cable schema mapping.
   - ☞ `BMC` (V2): server OOB query for building inventory attributes and cable schema.
   - ☞ `Physical server management` (V3): query server networking interfaces for cable schema mapping.
7. Discovery:
   - ☞ `Campus internal` (V1): integration point w/ the Inventory planning service.
   - ☞ `BMC` (V2): monitoring new baremetal server.
8. Config repo:
   - ☞ `Campus internal` (V1): serving admin portal and API endpoint.
9. OS image:
   - ☞ `Campus internal` (V1): serving admin portal and API endpoint.
10. Validation:
    - ☞ `Campus internal` (V1): serving admin portal and API endpoints.
    - ☞ `BMC` (V2): run tests via OOB module.
    - ☞ `Physical server management` (V3): run tests via remote access such as SSH.
    - ☞ `ovirt management` (V100): run tests via the Runtime service API such as creating new virtual machine or workload.
    - ☞ `RHHI provisioning` (V10): run tests such as adding a new LOC hardware resource.

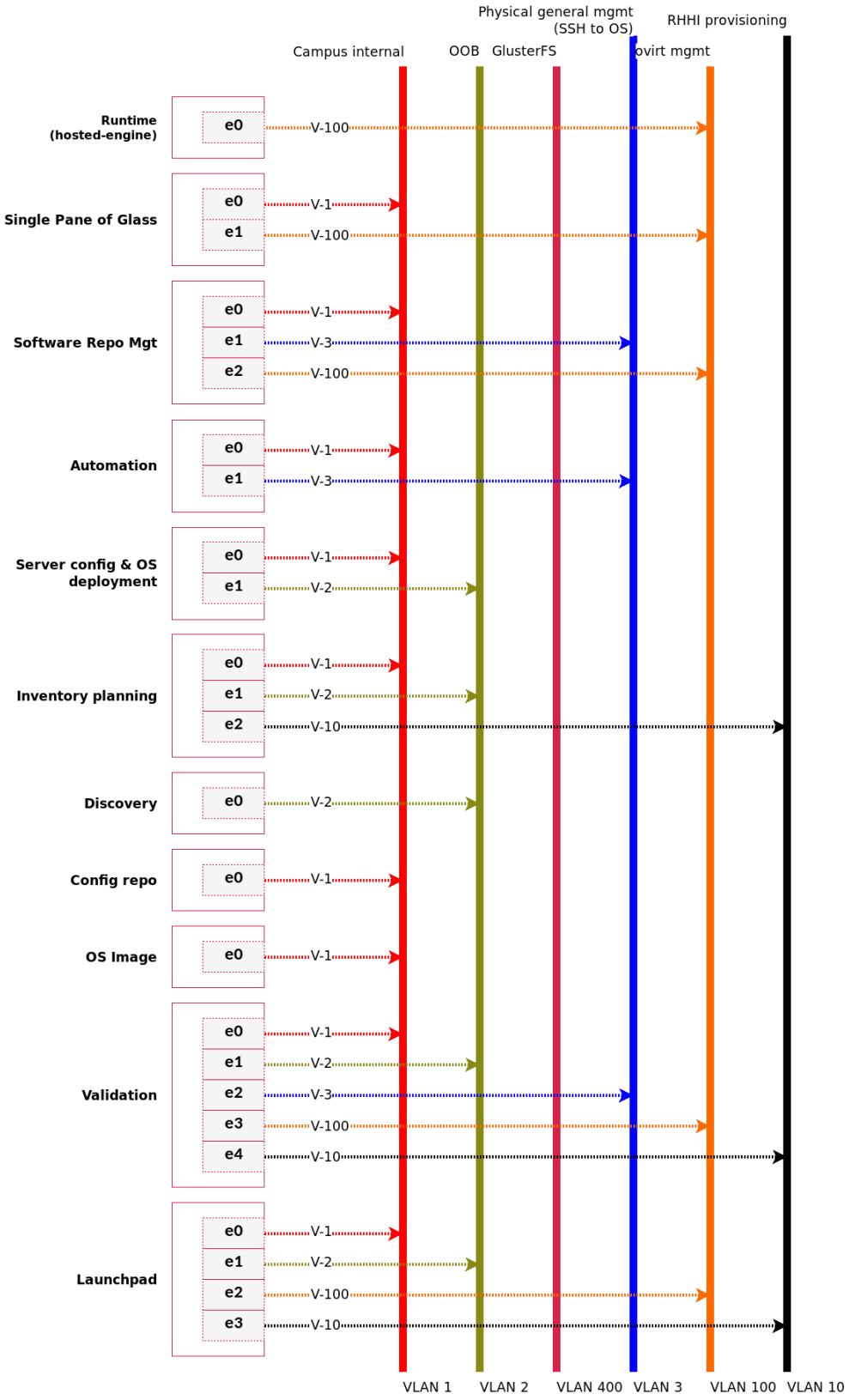

Figure 7.1: LOC networks of the Platform management services

7.3 Storage networks

Lenovo Open Cloud supports Ceph storage backends. A storage backend can be shared among multiple workloads and platforms, such as an OpenStack on-premise cloud. By deploying Ceph on top of Open Cloud Platform, we can also share common services and networks.

Storage networks can be viewed in two groups:

1. Storage specific networks: these are networks only applicable for the storage instance.
2. Shared Platform networks: networks extended from the LOC Platform networks to support storage infrastructure components that are deployed on the Platform and/or using the Platform management services.

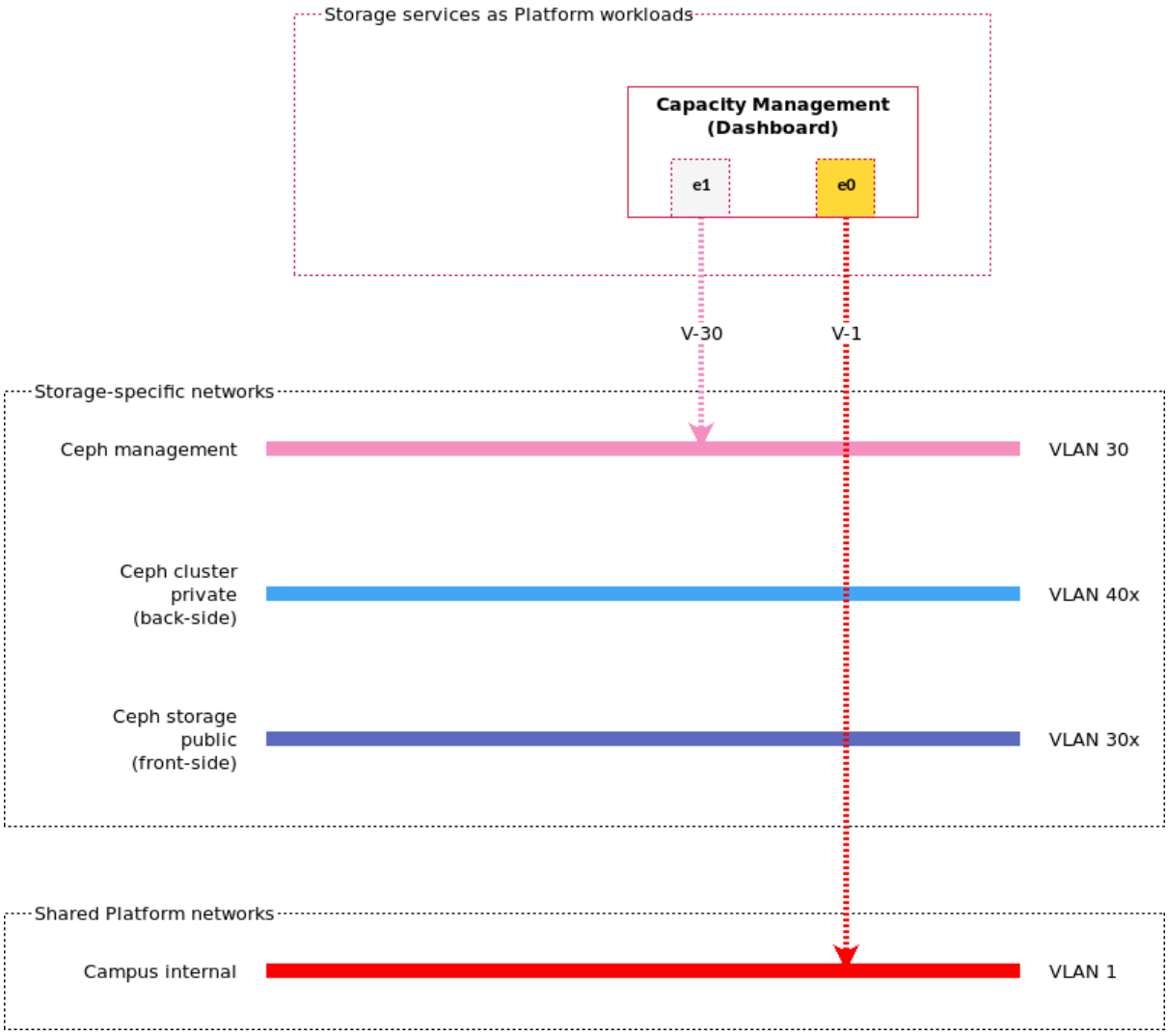

Figure 7.2: LOC Storage networks for storage services

7.3.1 Ceph specific logical networks

Ceph itself adds three new networks — Ceph management, Ceph storage public, and Ceph cluster private.

Ceph management communication between the Ceph dashboard and Ceph nodes, e.g. RPC and transferring zabbix monitoring data.

Ceph storage public data traffic by workloads who will use Ceph as their storage backend, also known as `front-side`.

Ceph cluster private Ceph private data traffic such as used in data rebalancing, also known as `back-side`.

7.3.2 Ceph specific VLANs

Table 7.3: LOC Ceph as storage infrastructure logical Network Definition

Logical Network	VLAN
Ceph management	30
Ceph storage public	30x
Ceph cluster private	40x

Note:

☞ `x`: cloud index, eg. 1-9. A LOC Platform can support multiple cloud instances. VLAN schema follows the pattern by using a cloud index to maintain network isolation.

7.3.3 Shared Platform networks

To manage Storage physical servers, we will connect them to three existing Platform networks — `BMC` (VLAN 2) for OOB management and node discovery, `RHHI Provisioning` (VLAN 10) for server OS provisioning, and `Physical server management` (VLAN 3) for general admin tasks such as SSH access.

In addition, to supporting the `Storage capacity service`, which is the main management tool of Storage deployed on Open Cloud Platform, we also need `Campus` (VLAN 1) for accessing its UI, and `VM general management` (VLAN 600).

Table 7.4: Ceph storage logical networks shared with Platform

Platform Logical Networks	VLAN	Ceph Dashboard (VM)	Ceph nodes (BM)
BMC	2	No	Yes

Platform Logical Networks	VLAN	Ceph Dashboard (VM)	Ceph nodes (BM)
Campus internal	1	Yes	No
Physical server management	10	No	Yes

7.3.4 Added Platform RHV networks for storage infrastructure

Two new Platform RHV networks are created to support Ceph specific functions including compute node provisioning, and cloud management.

Ceph management As defined in storage logical networks, this network is mapped onto the LOC Platform so that the Ceph Dashboard can monitor the Ceph cluster.

Cloud all A general bucket to support all cloud management. This includes making the `Ceph storage` **`public`** network available for these management applications.

7.4 Cloud networks

By building cloud infrastructure on LOC Platform and potentially a storage backend that is also supported by LOC Platform, cloud network can be viewed in three groups:

1. Cloud specific networks: these are networks only applicable for the cloud instance.
2. Shared storage networks: network to utilize a storage backend such as Ceph.
3. Shared Platform networks: networks extended from the LOC Platform networks to support cloud infrastructure components that are deployed on the Platform and/or using the Platform management services.

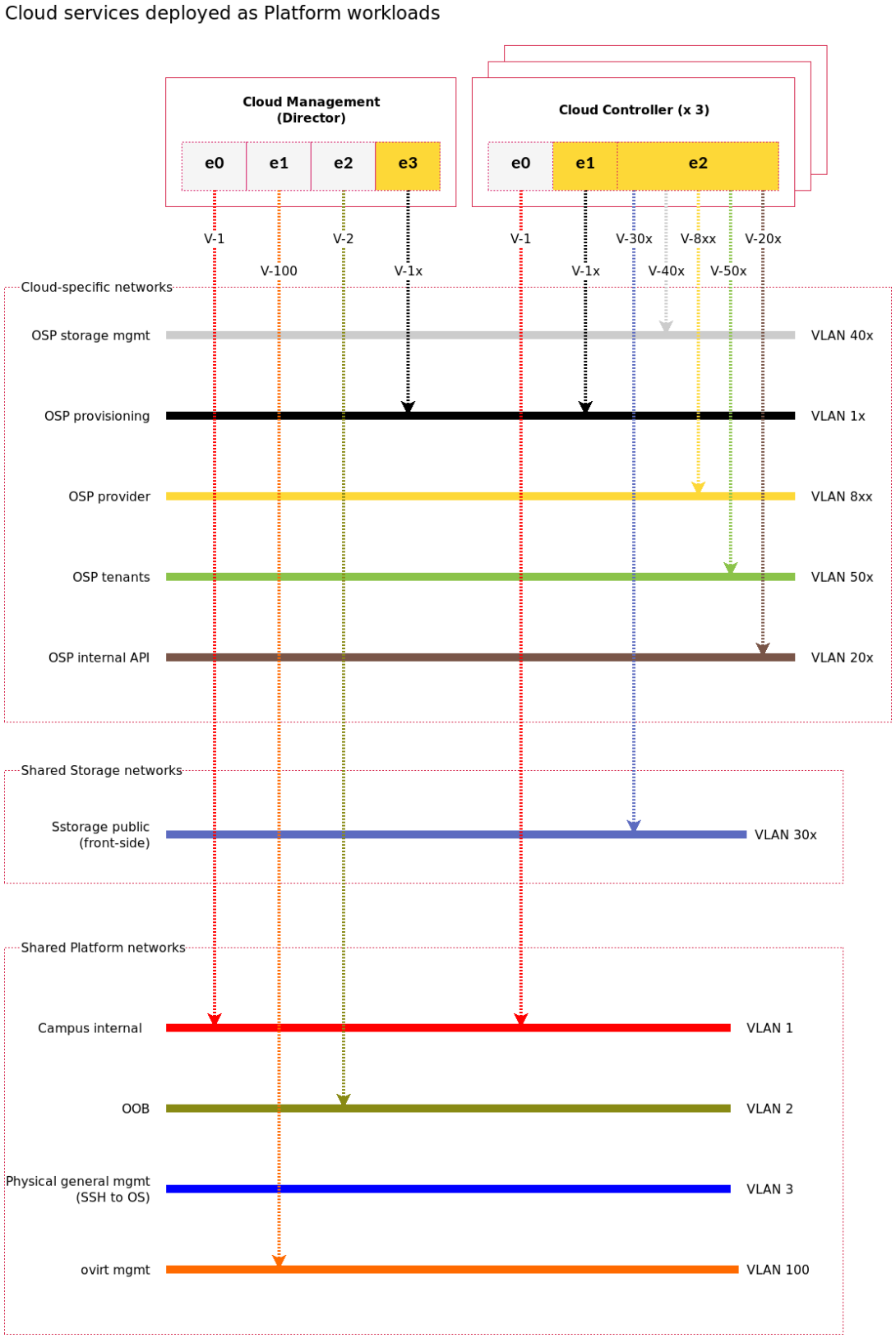

Figure 7.3: LOC Cloud networks for cloud services

7.4.1 Cloud specific logical networks

Multiple Red Hat OpenStack clouds are supported to locate on one LOC platform. As the cloud controller nodes and OpenStack director are deployed as VM instances on the LOC Platform, and there can be an external storage solution as the storage backend, cloud network is designed to be consistent with LOC Platform, choice of storage backend, and the requirement of OpenStack cloud.

These networks are to support OSP deployment and its workload such as NFV.

OSP provisioning Provides DHCP and PXE boot functions to help discover bare metal systems for use in the OpenStack installer to provision the system. This network is used by all cloud nodes.

OSP tenants This is the subnet for allocating the VM private IP addresses. Through this network, the VM instances can talk to each other. This network is used by cloud controller and compute nodes.

OSP storage mgmt OpenStack Object Storage (Swift) uses this network to synchronize data objects between participating replica nodes. Used by Swift, cloud controller with Swift service collocated.

OSP internal API The Internal API network is used for communication between the OpenStack services using API communication, RPC messages, and database communication. Used by cloud controller and compute nodes.

OSP DPDK The NFV Data(OSP DPDK) Network is used for NFV data plane traffic for VM applications that require high performance. Used by compute nodes with NFV workload.

OSP provider This is the provider network used for VM traffic on dedicated network infrastructure that do not require high performance. Used by controller and compute nodes.

Ceph storage The front-side storage network where Ceph clients (through Glance API, Cinder API, or Ceph CLI) access the Ceph cluster. Ceph Monitors operate on this network. Used by cloud controller and compute nodes.

7.4.2 Cloud specific VLANs

VLAN technology provides a simple way for logical grouping and isolation of the various networks. In order to achieve multiple OpenStack clouds, we will also apply rules on VLAN ID selections for different clouds.

Table 7.5: LOC cloud infrastructure cloud-specific logical networks

Logical Network	VLAN
OSP provisioning	1x
OSP internal API	20x
Ceph storage	30x
OSP storage mgt	40x
OSP tenants	50x
OSP DPDK	7xb
OSP provider	8xb

Note:

- ☞ `x`: cloud index, eg. 1-9. A LOC Platform can support multiple cloud instances. VLAN schema follows the pattern by using a cloud index to maintain network isolation.
- ☞ `xb`: VLAN index assigned for DPDK and provider inside a cloud instance. For example, VLANs for DPDK workloads on cloud `1` can be `711-719`. This, in turn, determines switch port VLAN configurations.

7.4.3 Shared storage networks

cloud infrastructure can take advantage of an external storage solutions such as Ceph as its storage backend. This way, storage can be centralized and different solutions can be used.

Storage public Interface to the storage solution. In Ceph, this is the `front-side` network serving storage data traffic.

Storage Logical Networks	VLAN	Director (VM)	Controller (VM)	Compute (BM)
Storage public	30x[1]	No	Yes	Yes

[1] `x` is the cloud index

Note:

By convention, the Storage public is connected to 10G ports (or faster) on cloud compute nodes.

7.4.4 Shared Platform networks

All LOC based cloud deployment builds Director and cloud controllers as virtual machines. This, therefore, requires them to be on Platform networks in order to use Platform service during initial setup and possibly their life cycle management such as software upgrade and fixes. Similarly, cloud servers also need to be on a set of Platform networks to be managed by the Platform service, and by the cloud services.

Table 7.7: LOC cloud infrastructure cloud-specific logical networks

Platform Networks	VLAN	Director (VM)	Controller (VM)	Compute (BM)
BMC	2	Yes	No	Yes
Campus internal	1	Yes	Yes	No
ovirt management	100	Yes	No	No

7.4.5 Added Platform RHV networks for cloud infrastructure

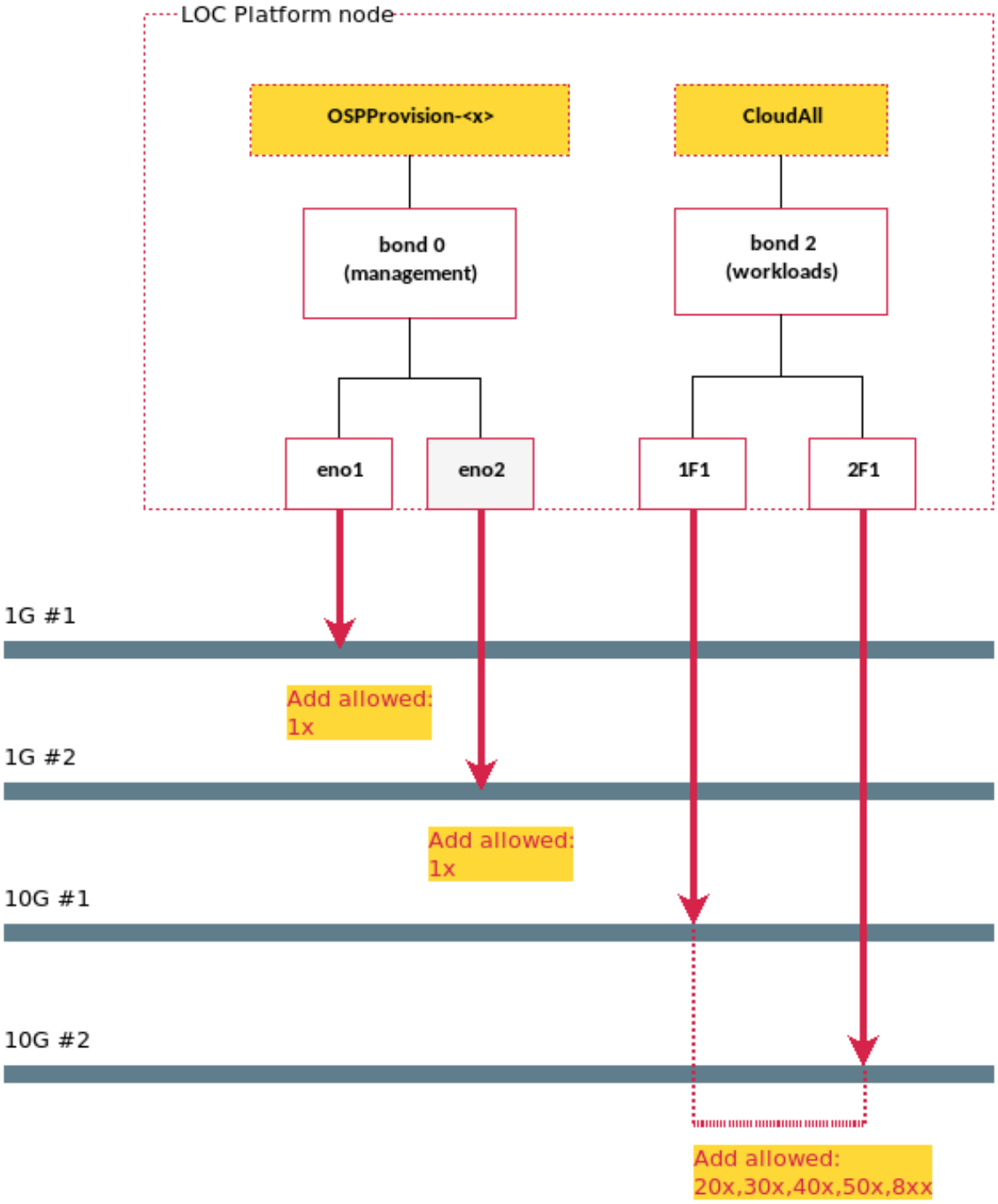

Figure 7.4: LOC cloud infrastructures added Platform RHV networks

Two new Platform RHV networks are created to support cloud specific functions including compute node provisioning, and cloud management.

OSP Provisioning As defined in cloud logical networks, this network is mapped onto the LOC Platform so that the Director and cloud controllers can initiate and control compute node provisioning.

Cloud all A general bucket to support all cloud management and some service data traffic that need to be through the Platform. This includes:

- ☞ OSP internal API
- ☞ OSP storage management
- ☞ OSP tenants
- ☞ OSP providers
- ☞ Ceph storage

Such design simplifies integration between the LOC Platform and its cloud infrastructures by keeping these cloud-driven decisions internal to workload designers.

8

Appendix: Bill of Material

Minimal hardware requirement is to have a 3-node Lenovo Open Cloud RHHI-v cluster (see the Lenovo Open Cloud RHHI Reference Architecture[1]).

Control planes for OpenStack, Ceph and Kubernetes are software only, which will be addressed here. Their hardware bill of materials and configuration options can be viewed in detail in respective documents:

1. For OpenStack, see the Red Hat OpenStack Platform with ThinkSystem Servers Reference Architecture[3].
2. For Ceph, see the Red Hat Ceph Reference Architecture[2].
3. For OpenShift, see the Red Hat OpenShift Container Platform on Lenovo ThinkSystem Servers[14]

8.1 Three-node LOC RHHI hardware BOM

8.1.1 1Gb switch

Table 8.1: LOC RHHI-v RackSwitch G8052 1Gb bill of materials, rear to front

Part Number	Product Description	Qty
7159HC1	Switch-G8052 : Lenovo RackSwitch G8052 (Rear to Front)	1
ASY2	Lenovo RackSwitch G8052 (Rear to Front)	1
3793	3m Yellow Cat5e Cable	up to 37
A1PJ	3m Passive DAC SFP+ Cable	1

Part Number	Product Description	Qty
6311	2.8m, 10A/100-250V, C13 to C14 Jumper Cord	2
00WW816	Essential Service - 3Yr 24x7 4Hr Response	1

For front-to-rear configuration, replace `ASY2` with `ASY1`.

8.1.2 25Gb switch

Table 8.2: LOC RHHI-v RackSwitch NE2572 25Gb Bill of Material

Part Number	Product Description	Qty
7159HE3	Switch-25G : Lenovo ThinkSystem NE2572 RackSwitch (Rear to Front)	1
AV19	Lenovo ThinkSystem NE2572 RackSwitch (Rear to Front)	1
AV1F	Lenovo 3m 25G SFP28 Active Optical Cable	up to 24
AV1L	Lenovo 3m 100G QSFP28 Active Optical Cable	1
3793	3m Yellow Cat5e Cable	1
6204	2.8m, 10A/100-250V, C13 to IEC 320-C20 Rack Power Cable	2
5WS7A06889	Premier with Essential - 3Yr 24x7 4Hr Response	1

For front-to-rear configuration, replace `AV19` with `AV1A`.

8.1.3 Servers

Table 8.3: LOC Lenovo SR650 Server Bill of Materials

Part Number	Product Description	Qty
7X06CTO1W	LOC RHHI: ThinkSystem SR650 - 3yr Warranty	1
AUVV	ThinkSystem SR650 2.5” Chassis with 8, 16 or 24 bays	1
AWEC	Intel Xeon Silver 4114 10C 85W 2.2GHz Processor	2
AUND	ThinkSystem 32GB TruDDR4 2666 MHz (2Rx4 1.2V) RDIMM	12
AURA	ThinkSystem 2U/Twr 2.5” SATA/SAS 8-Bay Backplane	2
AUNK	ThinkSystem RAID 930-16i 4GB Flash PCIe 12Gb Adapter	1
B49M	ThinkSystem 2.5” Intel S4610 480GB Mainstream SATA 6Gb Hot Swap SSD	2
B49N	ThinkSystem 2.5” Intel S4610 960GB Mainstream SATA 6Gb Hot Swap SSD	2
B0YS	ThinkSystem 2.5” 2.4TB 10K SAS 12Gb Hot Swap 512e HDD	12
AUR7	ThinkSystem 2U x8/X8/x8ML2 PCIE FH Riser 1	1
AURC	ThinkSystem SR550/SR590/SR650 (x16/x8)/(x16/x16) PCIe FH Riser 2 Kit	1
AUKK	ThinkSystem 1Gb 4-port RJ45 LOM	1
B0WY	Intel XXV710-DA2 10/25GbE SFP28 2-Port PCIe Ethernet Adapter	2
AVWF	ThinkSystem 1100W (-48V DC) Platinum Hot-Swap Power Supply	2
6400	2.8m, 13A/100-250V, C13 to C14 Jumper Cord	2
AUPW	ThinkSystem XClarity Controller Standard to Enterprise Upgrade	1
AXCH	ThinkSystem Toolless Slide Rail Kit with 2U CMA	1
A51P	2m Passive DAC SFP+ Cable	4

Feature code enables function of a Lenovo server and its component. The following feature code are enabled for RHHI server:

Table 8.4: LOC Lenovo SR650 Server Feature Code

Feature code	Feature type	Description
0034	52	Lenovo XClarity Controller Advanced Upgrade
0035	53	Lenovo XClarity Controller Enterprise Upgrade
801C	32796	IBM Security Key Lifecycle Manager for Lenovo SEDs

Note:

1. Listed feature code is included in server. No additional purchase is necessary.

8.2 Three-node LOC RHHI software BOM

Table 8.5: LOC Software BOM, 3 year premium service support

SKU	Product Description	Qty
RS00139F3	Red Hat Hyperconverged Infrastructure for Virtualization (RHHI-V)	1
RH00031F3	Smart Management	3
MCT3475F3	Red Hat Insights	1
00MT202	Lenovo xClarity Enterprise	3

8.3 Three-node HCI Ceph and Openstack software BOM

Table 8.6: LOC 3-node HCI Ceph and Openstack Software BOM, 3 year premium service support

SKU	Product Description	Qty
RS00032F3	Red Hat Cep Storage	1
MCT2982F3	Red Hat Openstack Platform (w/o Guest) with Smart Management	4
MCT2980F3	Red Hat OpenStack Platform with Smart Management & Guests	3
00MT202	Lenovo xClarity Enterprise	3

8.4 Three-node Ceph ONLY software BOM

Up to 256TB on a maximum of 12 physical nodes.

Table 8.7: LOC 3-node Ceph ONLY Software BOM, 3 year premium service support

SKU	Product Description	Qty
RS00032F3	Red Hat Cep Storage	1
00MT202	Lenovo xClarity Enterprise	3

8.5 Three-node Openstack software BOM

Table 8.8: LOC HCI 3-node Openstack Software BOM, 3 year premium service support

SKU	Product Description	Qty
MCT2982F3	Red Hat Openstack Platform (w/o Guest) with Smart Management	4
MCT2980F3	Red Hat OpenStack Platform with Smart Management & Guests	3
00MT202	Lenovo xClarity Enterprise	3

9

Appendix: Unified Hardware Management

Unified Hardware Management refers to the common steps that a hardware, in particular a baremetal server, will go through in order to be included into an infrastructure:

```
autonumber "<font color=red><b>Message 00:"

actor Admin

box "Platform Services"
  participant "Inventory Service" as Netbox
  participant "Automation Service" as Tower
  participant "OS Deployment Service" as LXCA
  participant "Software Repository Service" as Satellite
  participant "Config Repo Service" as Config
end box

collections Servers #ff6a00

== PHASE 1: Register resource ==

Admin -> Tower: apply_workflow(inventory_server_id,
   ↪ workflow_id)
Tower -> Netbox: get_server_bmc_info(inventory_server_id)
Netbox -> Tower: (bmc_ip, bmc_credentials)
Tower -> LXCA: manage(bmc_ip, bmc_credentials)
LXCA -> Tower: `hw_mgt_server_id` if successful; NULL if
```

```
    ↪ failed.
22
23 == PHASE 2: Apply HW configs ==
24
25 alt role based
26   Tower -> Netbox: get_server_role(inventory_server_id)
27   Netbox -> Tower: server_role
28   Tower -> Config: get_server_configs(server_role)
29   Config -> Tower: role based `hw_configs`
30   note left: configs is determined by `device role`
31
32   Tower -> LXCA: config_server(hw_mgt_server_id, hw_configs
       ↪ ).
33
34 else UUID based
35   Tower -> Config: get_server_configs(uuid)
36   Config -> Tower: UUID based `hw_configs`
37   note left: configs is determined by device `UUID`
38
39   Tower -> LXCA: config_server(uuid, hw_configs).
40 end
41
42 LXCA -> Servers: apply_configs(hw_configs)
43 LXCA -> Tower: config result `TRUE|FALSE`
44
45 == PHASE 3: PROVISION OS ==
46
47 alt role based
48   Tower -> Netbox: get_server_role(inventory_server_id)
49   Netbox -> Tower: server_role
50   Tower -> Config: get_server_os(server_role)
51   Config -> Tower: role based `os_configs`
52   Tower -> LXCA: provision_os(hw_mgt_server_id, os_configs)
53   LXCA -> Servers: provision_os(hw_mgt_server_id,
       ↪ os_configs)
54   LXCA -> Tower: provision result `TRUE|FALSE`
55   Tower -> Netbox: update_server_ssh_info(ip, username, pwd
```

```
      ↪ )
56
57 else UUID based
58   Tower -> Config: get_server_os(uuid)
59   Config -> Tower: UUID based `os_configs`
60   Tower -> LXCA: provision_os(uuid, os_configs)
61 end
62
63 LXCA -> Servers: provision_os(hw_mgt_server_id, os_configs)
64 LXCA -> Tower: provision result `TRUE|FALSE`
65 Tower -> Netbox: update_server_ssh_info(ip, username, pwd)
66
67 == PHASE 4: REGISTER TO SOFTWARE REPO ==
68
69 alt role based
70   Tower -> Netbox: get_server_role(inventory_server_id)
71   Netbox -> Tower: server_role
72   Tower -> Config: get_server_repo_key(server_role)
73   Config -> Tower: role based `repo_service_key_name`
74
75 else UUID based
76   Tower -> Config: get_server_repo_key(uuid)
77   Config -> Tower: UUID based `repo_service_key_name`
78 end
79
80 Tower -> Satellite: register_server_to_repo(
     ↪ repo_service_key_name)
```

10

Contributors

A special thank you to the following Lenovo colleagues for their contributions to this document:

- Weiwei Chen, Engineer — Data Center Group
- Chengzhen Chu, Engineer — Data Center Group
- Leilei Lei, Engineer — Data Center Group
- Guannan Sun, Engineer — Data Center Group
- Ricky Stambach, Engineer — Data Center Group
- Zelin Wang, Engineer — Data Center Group

11

References

[1] F. Xia and M. Halas, “Lenovo open cloud rhhi reference architecture,” May 2019 [Online]. Available: https://lenovopress.com/lp1148

[2] F. Lei, M. Halas, J. Bryant, L. Liu, D. Ding, and M. Perks, “Reference architecture: Red hat ceph storage,” May 2019 [Online]. Available: https://lenovopress.com/lp1147

[3] J. Xiaotong, X. Lin, M. Perks, Y. Huang, and S. Angalur, “Reference architecture: Red hat openstack platform with thinksystem servers,” Sep. 2017 [Online]. Available: https://lenovopress.com/lp0762

[4] Red Hat, “Red hat satellite.” [Online]. Available: https://access.redhat.com/products/red-hat-satellite

[5] Red Hat, “RED hat satellite 6.2 server administration guide: 3.3. LIFE cycle environments.” [Online]. Available: https://access.redhat.com/documentation/en-us/red_hat_satellite/6.2/html/server_administration_guide/sect-red_hat_satellite-server_administration_guide-configuring_organizations_locations_and_life_cycle_environments-life_cycle_environments

[6] Red Hat, “RED hat satellite 6.1 user guide: CHAPTER 6. USING content views.” [Online]. Available: https://access.redhat.com/documentation/en-us/red_hat_satellite/6.1/html/user_guide/chap-red_hat_satellite-user_guide-using_content_views

[7] Red Hat, “RED hat satellite 6.1 user guide: CHAPTER 11. CONFIGURING activation keys.” [Online]. Available: https://access.redhat.com/documentation/en-us/red_hat_satellite/6.1/html/user_guide/chap-red_hat_satellite-user_guide-configuring_activation_keys

[8] Red Hat, “Red hat ansible tower.” [Online]. Available: https://access.redhat.com/products/ansible-tower-red-hat

[9] Red Hat OpenStack Documentation Team, “RED hat openstack platform 13 director instal-

lation and usage: An end-to-end scenario on using red hat openstack platform director to create an openstack cloud." [Online]. Available: https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/13/html/director_installation_and_usage

[10] Lenovo, "Lenovo thinksystem sr650 server (xeon sp gen 1): Product guide." [Online]. Available: https://lenovopress.com/lp0644-thinksystem-sr650-server-xeon-sp-gen1

[11] Lenovo, "Lenovo rackswitch g8052: Product guide." [Online]. Available: https://lenovopress.com/tips1270-lenovo-rackswitch-g8052

[12] Lenovo, "Lenovo thinksystem ne2572 rackswitch: Product guide." [Online]. Available: https://lenovopress.com/lp0608-lenovo-thinksystem-ne2572-rackswitch

[13] Red Hat, "Product documentation for red hat openstack platform 13." [Online]. Available: https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/13/

[14] X. Jiang, M. Perks, B. Sun, and S. Angaluri, "Reference architecture: Red hat openshift container platform on lenovo thinksystem servers," 2019 [Online]. Available: https://lenovopress.com/lp0968