Redbooks
ibm.com/redbooks

IBM Cloud Manager with OpenStack on z Systems V4.2

Bill White

Li Yong Li

Bo Chang Liu

Roberto Mosqueda

Livio Sousa

Qi Ye

Cloud

z Systems

IBM

Redbooks

International Technical Support Organization

IBM Cloud Manager with OpenStack on z Systems V4.2

September 2015

SG24-8295-00

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

First Edition (September 2015)

This edition applies to Version 4, Release 2, of IBM Cloud Manager with OpenStack for z Systems.

Contents

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

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

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®
Bluemix™
DB2®
developerWorks®
FICON®
Global Business Services®
IBM UrbanCode™
IBM z™
IBM z Systems™
IBM z13™
IBM®
Power Systems™
PowerVM®
POWER®
RACF®
Redbooks®
Redbooks (logo) ®
Storwize®
System z®
WebSphere®
z Systems™
z/Architecture®
z/OS®
z/VM®
z13™

The following terms are trademarks of other companies:

SoftLayer, and SoftLayer device are trademarks or registered trademarks of SoftLayer, Inc., an IBM Company.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

UNIX is a registered trademark of The Open Group in the United States and other countries.

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

THIS PAGE INTENTIONALLY LEFT BLANK

Preface

IBM® Cloud Manager with OpenStack for z Systems™, V4.2 is an easy-to-use cloud management solution that serves as a control point for cloud managed resources based on the OpenStack Juno distribution. IBM Cloud Manager with OpenStack for z Systems, V4.2 can operate as a cloud management hub that can manage IBM z Systems™, IBM Power Systems™, and x86 resources from a central point of control.

This IBM Redbooks® publication gives a broad understanding of the architecture for IBM Cloud Manager with OpenStack for z Systems, V4.2, and how it can be implemented and deployed to support cloud services on the z Systems platform.

This publication also helps you plan, install, configure, and use IBM Cloud Manager with OpenStack for z Systems, V4.2. It focuses on planning and design of your cloud environment on z Systems, as well as the installation and configuration definitions that are necessary to build and manage cloud resources under IBM z/VM®.

This information is useful to IT architects and system administrators who plan for and install IBM Cloud Manage with OpenStack for z Systems. The reader is expected to have a good understanding of IBM z Systems™ hardware, IBM z/VM, Linux on z Systems, and cloud concepts.

Authors

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

Bill White is a Project Leader and Senior z Systems Networking and Connectivity Specialist at the International Technical Support Organization, Poughkeepsie Center.

Li Yong Li is a senior IT Specialist at IBM STG Lab Services and Training in China. He has been with IBM for 10 years. His areas of expertise include virtualization on z Systems, IBM z/OS®, z/VM, and Linux on z Systems, and cloud computing solutions. Li also provides post-sales technical support and guidance to z Systems clients.

Bo Chang Liu is an advisory IT Specialist working in IBM STG Lab Services and Training in China. He has been with IBM for five years. He is currently working as a z Systems specialist providing technical support and guidance to clients, and developing and testing z Systems cloud solutions. Throughout his career, Bo Chang has worked in areas such as z/OS, z/VM, Linux on z Systems, and cloud computing solutions on z Systems. He also has working experience on other cloud solutions such as OpenStack and IBM Bluemix™. Bo Chang has co-authored IBM developerWorks® publications on subjects such as hybrid cloud solutions.

Roberto Mosqueda is an Advisory Software Engineer from IBM Mexico Software Lab. Since joining IBM in 2008, Roberto has worked as a Test Engineer and Test Leader for IBM Storage and Software solutions. Roberto holds a bachelor's degree in Computer Systems Engineering from the Instituto Tecnologico de Morelia in Michoacan, Mexico. He is a senior inventor with seven patent applications in progress. He is also an active IBM Lab Advocate for WebSphere® Application Server on z/OS solution and co-author of two IBM Storage books.

Livio Sousa is a Certified IT Specialist with over 15 years of experience with high-end platforms, which encompasses servers, storage, and networking equipment. Throughout his career, he has worked with several different software platforms, such as IBM z/OS, UNIX, Linux, and Windows, as well as different hardware processor architectures such as the IBM z/Architecture® and the IA-32 architecture and its extensions. For the past 13 years, he has worked as a z Systems Client Technical Specialist, responsible for supporting midsize to large IT/ Cloud and data center related projects and engaging with clients by sharing the company solutions, strategy and future directions. He also has developed skills for the z Systems virtualization technologies and Cloud Solutions for Latin America. He co-authored other IBM publications on subjects such as z/Architecture, z/OS UNIX System Services, z/VM performance, and Linux on z Systems. Livio holds an MBA in IT Management.

Qi Ye is responsible for IBM System z® worldwide technical enablement for cloud, virtualization, and Linux. He has 11 years of mainframe experience with IBM, including z/OS, DB2®, z/VM, and Linux. Currently Qi focuses on System z cloud solutions, Linux on z Systems, and z/VM. Before he took the role of z Systems WW Technical Enablement, he worked in the IBM China Lab, and IBM growth markets.

Thanks to the following people for their contributions to this project:

Dave Bennin
Don Brennan
Rich Conway
Bob Haimowitz
IBM Global Business Services®, Development Support Team

John Arwe
Mike Baskey
Mike Bonett
Susan Greenlee
Chen CH Ji
IBM z Systems

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

1

Understanding IBM Cloud Manager with Openstack for z Systems

This chapter introduces the concepts of cloud management, the value of z Systems in a cloud environment with z/VM (and Linux on z Systems), and an overview of IBM Cloud Manager with OpenStack on z Systems, V4.2.

This chapter includes the following sections:

- Demands on cloud management
- Why the IBM mainframe is a good fit for the cloud
- What is OpenStack
- IBM Cloud Manager with OpenStack
- IBM z/VM management components
- IBM Cloud Manager with OpenStack for z Systems
- Use cases for IBM Cloud Manager with OpenStack for z Systems
- IBM Custom Patterns for Linux on z Systems

1.1 Demands on cloud management

Cloud computing environments have been evolving quickly over the last several years. These environments are providing new services that were not offered in cloud service models before. This means new challenges in terms of service management and that more sophisticated tools are needed to manage those services.

Managing a cloud environment requires tools that can take advantage of a pool of virtualized compute, storage, and network resources, and present them to the consumer as a service in a secure way. A cloud management system should also help with these tasks:

- Offering open cloud management and application programming interfaces (APIs)
- Improving the usage of the infrastructure
- Lowering administrative overhead and improving operations productivity
- Reducing management costs and improving responsiveness to changing business needs
- Automating resource allocation
- Providing a self-service interface
- Tracking and metering resource usage

In addition, a cloud management system must allow for the management of virtualized IT resources to support different types of cloud service models and cloud deployment models.

This book focuses on IBM Cloud Manager with OpenStack for z Systems, V4.2, which satisfies a wide range of cloud management demands. It offers easy to deploy and use cloud management software based on OpenStack with IBM enhancements.

1.1.1 Cloud service models

Cloud computing is a general term for anything that involves delivering hosted services over a network or the internet. These services are broadly divided into three categories: infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS).

> **Note:** IBM Cloud Manager with OpenStack for z Systems, V4.2 helps cloud administrators manage their cloud resources for IaaS and PaaS offerings.

Infrastructure as a service (IaaS)

Virtual resources are provided in terms of underlying infrastructure (compute, storage, network) only. The consumer is responsible for installation and maintenance of operating systems, middleware, and applications, but does not manage or control the underlying infrastructure. This offering gives the consumer the most flexibility and associated responsibility.

Platform as a service (PaaS)

The underlying infrastructure, operating system, and application hosting software are all provided. The consumer does not have any control or management capabilities of the underlying OS, but can control the applications. By abstracting the hardware, OS, and application hosting software, the consumer can focus their skills and efforts strictly on their applications.

Software as a service (SaaS)

Provides an end-to-end environment that includes the underlying infrastructure, operating system, application hosting software, and applications. This service model allows applications to be made available to the consumer through various devices and API services.

1.1.2 Cloud deployment models

All cloud services require infrastructure and those services can be deployed using different models based on where they are hosted. The three deployment models are private clouds (also known as on-premises), public clouds (also known as off-premises), and hybrid clouds.

> **Note:** IBM Cloud Manager with OpenStack for z Systems, V4.2 supports all three cloud deployment models.

Private cloud

A private cloud is owned and operated by a single company that controls the way virtualized resources and automated services are customized and used by various lines of business and groups. With private cloud services, the users own the hardware, software, and all required supporting infrastructure.

Public cloud

Public clouds are owned and operated by companies that use them to offer rapid access to affordable computing resources to other organizations or individuals. With public cloud services, users do not need to purchase hardware, software, or supporting infrastructure, which is owned and managed by providers.

Hybrid cloud

A hybrid cloud uses a private cloud foundation combined with the strategic use of public cloud services. Nowadays more companies with private clouds are evolving to manage workloads across data centers, private clouds, and public clouds, creating hybrid clouds.

> **Note:** IBM cloud offerings include options for the cloud service models and cloud deployment models. For more information, see the following website:
>
> `http://www.ibm.com/cloud-computing/us/en/`

1.2 Why the IBM mainframe is a good fit for the cloud

IBM mainframes offer unique capabilities, in terms of virtualization, performance, scalability, security, reliability, and availability. These capabilities have been developed and refined over several decades, and are key to any cloud-based service.

A study[1] finds that the total cost of ownership for an IBM z13 cloud solution is lower than an x86 or public cloud solution.

[1] "TCO for an IBM z13™ Cloud Is Lower Than an x86 or Public Cloud, IBM Study Finds":
`http://www.ibmsystemsmag.com/mainframe/trends/Cloud-Computing/IBM-z13-cloud-TCO/`

1.2.1 Virtualization is key

Virtualization is the ability for a computer system to share resources so that one physical server can act as many virtual servers. Virtualization is central to building a cloud infrastructure. The fundamentals of virtualization have been part of the mainframe platform since its inception, with its *shared everything* architecture.

In addition, logical partitions (LPARs) allow the z Systems platform to run different operating systems in the same box. With LPARs, logical segmentation of the mainframe resources, such as memory, processor units, and I/O devices can be used.

The IBM z/VM hypervisor allows the sharing of the mainframe's physical resources, such as disk, memory, network adapters, and CPU. z/VM can host large numbers of VMs, making it possible to achieve massive consolidation and same system growth with limited real resources.

Hypervisor: A software component that uses a layer of code in firmware to enable dynamic resource sharing, providing flexibility in how virtual resources are managed. It is the primary technology for systems virtualization.

1.2.2 Better performance

Systems achieve the levels of efficiency demanded by modern businesses through an overall balanced design. Processors, memory, I/O, and network communications must complement each other to achieve the required levels of performance. You can have the fastest processors in the world, but if you cannot feed them, your workloads will suffer. The z Systems platform has decades of optimization built into its design.

Mainframe resources can be shared among disparate workloads, achieving the highest possible utilization of the platform with sustained throughput. The z Systems platforms offer significant capacity and performance capabilities over other platforms. Many factors contribute to this effect, including the larger number of processors, individual processor performance, memory caches, and machine instructions. Furthermore, because the z Systems platform has dedicated processors for I/O operations, workloads with high I/O demand, such as transaction processing, can run more efficiently.

1.2.3 Greater scalability

Another strength of the mainframe is its scalability, which allows you to meet changing business requirements and accommodate growth. With z Systems, you can dynamically add more capacity in seconds, increasing the elasticity of the environment without affecting ongoing operations.

Mainframes build on years of IBM leadership in virtualization and can host more virtual machines in a single footprint than any other platform. With its superior management services, a mainframe enables resources to be shared among workloads, achieving the highest possible utilization of the platform.

z Systems also allows you to run multiple copies of z/VM on a single mainframe for enhanced scalability and failover capability for the VMs.

1.2.4 Higher levels of security

Security remains a major concern for any cloud computing environment. z Systems has a long history of providing security for applications and sensitive data in virtual environments. It is the most securable platform in the industry today, beginning with the hardware and integrated throughout the stack. The mainframe has an Evaluation Assurance Level (EAL) of 5 for its LPAR technology, which remains the highest level of security certification in the industry. It also has an EAL rating of 4+ for the use of Linux in this environment.

The mainframe-integrated, hardware cryptographic coprocessors provide more levels of security. Virtual machines like Linux on z Systems can take advantage of these cryptographic coprocessors.

1.2.5 Notable reliability and availability

Reliability and availability of resources are key factors when selecting a cloud delivery option. The z Systems platform has a long history of reliability and availability that enables it to host cloud environments at unmatched levels of service. The z Systems platform has built-in hardware redundancy and fewer points of failure. It is a reliable platform that provides high availability and helps to achieve service level agreement (SLA) objectives. There are physical, redundant parts within the server that support hot failover, nondisruptive configuration changes, and dynamic replacement capabilities. Memory and processors can be added and enabled dynamically on z Systems, allowing for continuous service and availability with no interruption. The z Systems platform has five 9s of availability and the mean time to failure is measured in decades.

Nowadays, cloud is fast becoming the new infrastructure model and IBM z Systems delivers enterprise grade qualities of service for the most demanding workloads and environments in the market.

1.2.6 Standards make virtualization effective

Today's systems must be able to scale up and scale out, not just in terms of performance and size, but also in functionality. Virtualization is a core enabler of system capability, but standardization is key to making virtualization effective.

z/VM implements standard APIs, like Posix and OpenStack, and supports multiple Linux distributions as virtual machines, including Red Hat and SUSE Linux.

> **Linux on z Systems:** The operating system on the IBM mainframe is the same Linux that runs on other platforms, with the same look and feel. However, Linux on z Systems offers more benefits in terms of reliability, scalability, and security that are not available on other platforms.

1.2.7 IBM z Systems cloud solutions

IBM z Systems cloud solutions help organizations improve productivity, simplify management, and accelerate cloud adoption with virtualization.

There are several different cloud solutions offered by IBM on z Systems platforms running z/VM and Linux on z Systems, such as IBM Enterprise Cloud System, IBM Wave for z/VM, and IBM Cloud Orchestrator.

For more information about available z Systems cloud solutions, see:

http://www.ibm.com/systems/z/solutions/cloud/

1.3 What is OpenStack

The OpenStack project is a global collaboration community of developers and cloud computing technologists working to create an open source cloud computing platform for public, private, and hybrid clouds. The cloud computing platform is integrated by a list of interrelated services that provides different management features for the cloud infrastructure.

1.3.1 IBM contributes to OpenStack

IBM uses OpenStack as the foundation for its cloud solutions and contributing to the project by being a key contributor and Platinum sponsor of OpenStack.

Hundreds of IBM employees worldwide have collaborated with OpenStack, working on the development and testing of different OpenStack releases.

1.3.2 OpenStack reference architecture

OpenStack services together provide an IaaS or PaaS solution. Each OpenStack service offers an API that facilitates its integration. Based on your service needs, you can install some or all services. In Figure 1-1, you can see how the OpenStack services work together to deliver different services to the VM instance. For example, Neutron provides the network connectivity, Cinder provides the storage volumes, and so on.

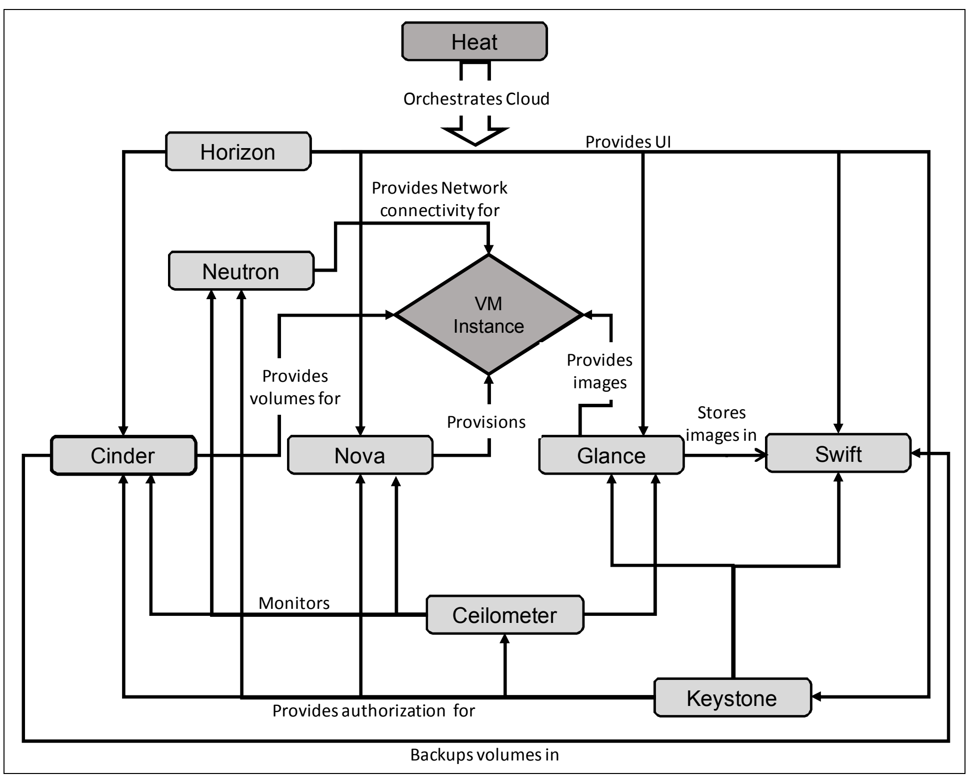


Figure 1-1 OpenStack reference architecture

1.3.3 OpenStack services overview

Table 1-1 lists the services available with OpenStack, the project name associated, and a description of the service.

Table 1-1 OpenStack services

Service	Project name	Description
Dashboard	Horizon	Provides a web-based self-service portal to interact with underlying OpenStack services, such as starting an instance, applying IP addresses, and configuring access controls.
Compute	Nova	Manages the lifecycle of VM instances in an OpenStack environment. Responsibilities include creating, scheduling, and decommissioning of virtual machines on demand.
Networking	Neutron	Provides an API for users to define networks and the attachments into them. Has a pluggable architecture that supports many popular networking vendors and technologies.
Storage Services		
Block Storage	Cinder	Provides persistent block storage to running instances. Its pluggable driver architecture facilitates the creation and management of block storage devices.
Object Storage	Swift	Stores and retrieves arbitrary unstructured data objects through a RESTful, HTTP-based API. It is highly fault tolerant with its data replication and scale-out architecture. Its implementation is not like a file server with mountable directories. In this case, it writes objects and files to multiple drives, ensuring the data is replicated across a server cluster.
Shared Services		
Identity Service	Keystone	Provides an authentication and authorization service for other OpenStack services. Provides a catalog of endpoints for all OpenStack services.
Image Service	Glance	Stores and retrieves virtual machine disk images. OpenStack Compute uses this during instance provisioning.
Telemetry	Ceilometer	Monitors and meters the OpenStack cloud for billing, benchmarking, scalability, and statistical purposes.
Higher-level Services		
Orchestration	Heat	Orchestrates multiple composite cloud applications by using either the native Heat Orchestration Template (HOT) format or the AWS CloudFormation template format. Both formats can use the OpenStack native REST API and the CloudFormation compatible Query API.
Database Service	Trove	Provides scalable and reliable Cloud database as a service functions for both relational and non-relational database engines.
Data Processing Service	Sahara	Provides capabilities to provision and scale Hadoop clusters in OpenStack by specifying parameters, such as Hadoop version, cluster topology, and nodes hardware details.

> **Note:** Swift, Trove, and Sahara are not supported in the z/VM environment.

For additional information about OpenStack services you can review the official documentation at:

http://docs.openstack.org/

OpenStack storage services options

The OpenStack stack uses two types of storage by using OpenStack storage services:

- Ephemeral storage

  Ephemeral storage means that from the user's point of view, the storage resources effectively disappear when a virtual machine is terminated.

  Ephemeral storage includes a root ephemeral volume. The root disk is associated with an instance, and exists only for the life of this instance. Generally, it is used to store an instance`s root file system. It persists across the guest operating system restarts, and is removed after the instance deletion. The size of the root ephemeral volume is defined in the flavor of the image template.

- Persistent storage

  Persistent storage means that the storage resource outlives any other resource and is always available, regardless of the state of a running instance.

OpenStack clouds explicitly support two types of persistent storage:

Block storage (cinder) Used to add more persistent storage to a virtual machine (VM) and to store user data.

Object storage (swift) Used to store virtual machine images and data.

1.4 IBM Cloud Manager with OpenStack

This section focuses on how IBM Cloud Manager with OpenStack integrates the capabilities of OpenStack and IBM cloud technologies to provide several benefits to IT infrastructure users and administrators:

- It enables rapid IT response to changing demands of business through self-service provisioning of infrastructure services because users can redeploy virtual servers with an easy to use interface.
- Yields improved virtualization operational efficiency and greater overall business effectiveness. Administrators capture and manage standard VM images with support for common business processes.
- Supports production grade cloud operations and interoperability at scale through enhanced foundation and full OpenStack API compatibility.
- Provides optimized infrastructure usage, reduced cost of cloud ownership, and higher workload quality of service.
- Open computing cloud alternative to proprietary vendors, with world-class support from IBM.
- Provides an IBM fully tested version of OpenStack that includes added value components, such as a deployment of Chef through IBM Cloud Manager with OpenStack, a Platform Resource Scheduler, and so on.

Note: IBM Cloud Manager with OpenStack is an easy to deploy and use cloud management software offering, with IBM enhancements and support.

1.4.1 IBM Cloud Manager with OpenStack components

IBM Cloud Manager with OpenStack integrates various components to automate IT infrastructure service provisioning. IBM Cloud Manager with OpenStack delivers access to OpenStack APIs and is backed with IBM support.

As you can see in Figure 1-2, IBM Cloud Manager with OpenStack provides a self-service user interface for the cloud users. This interface enables the users to manage their VMs, request new VMs, manage permissions, and so on.

Figure 1-2 also shows the z/VM driver that is included in the OpenStack platform to interact and manage z/VM. OpenStack cloud admins have two options to manage their IaaS infrastructure: Using the command-line interface (CLI), and using the Horizon User Interface.

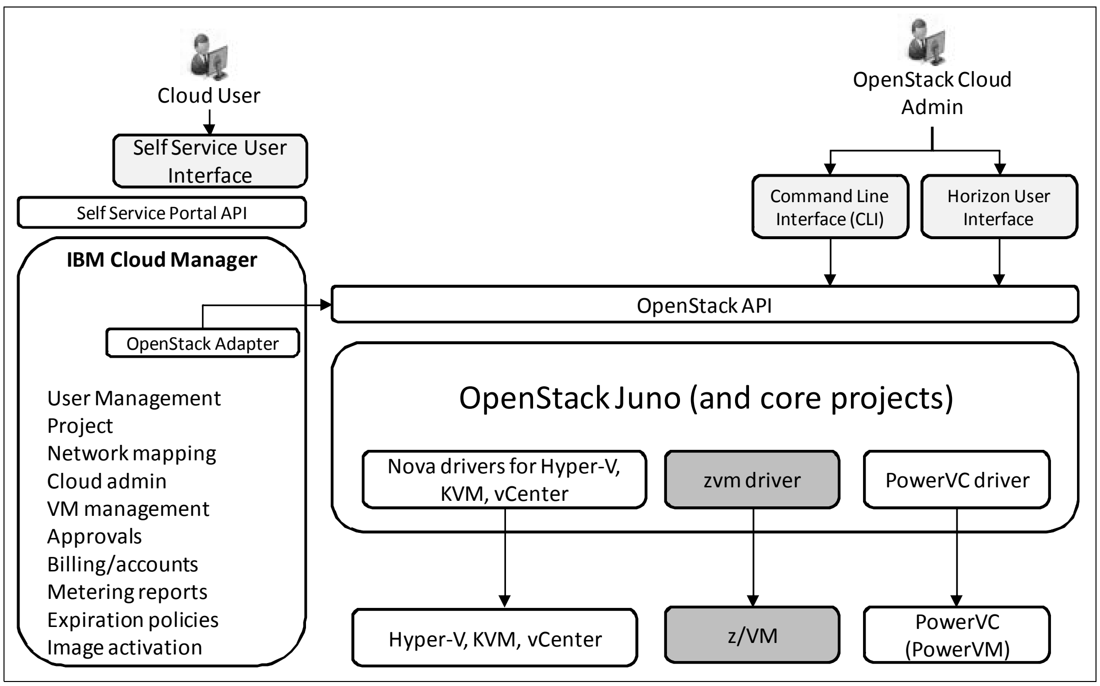


Figure 1-2 IBM Cloud Manager and OpenStack reference architecture

1.4.2 IBM Cloud Manager with OpenStack supported platforms

The IBM Cloud Manager with OpenStack environment is composed of two node classes:

- The controller node manages the virtual resources of the compute nodes in the cloud environment. Every cloud has one controller node, and each controller node can manage more than one cloud.
- The compute node represents the nodes that can be managed by the controller.

Note: Based on the cloud topology, a node can have both the controller and compute services.

Controller node

Table 1-2 lists the operating systems that are supported for the host node where the IBM Cloud Manager with OpenStack controller node can be installed.

Table 1-2 Operating Systems supported by the OpenStack controller node

OpenStack Controller Node	Host Operating System
x86	RHELs 6.5 x86_64
Power Systems	RHELs 6.5 ppc_64 BE
z Systems	z/VM 6.3

Compute node

Table 1-3 lists the operating systems that are supported for the platform where the IBM Cloud Manager with OpenStack compute node can be installed.

Table 1-3 Operating Systems supported by the IBM Cloud Manager with OpenStack compute node

	x86				Power Systems		z Systems
Hypervisor / Compute Node	VMware through SCE +VCenter	VMware through OS	Hyper-V (2012 Server) through OS	KVM (RHEL 6.5) through OS	IBM PowerVM® through PowerVC	PowerKVM through OS	z/VM through OS
Operating System	Windows SUSE Red Hat	Windows SUSE Red Hat Other Linux			IBM AIX® SUSE Red Hat	Other Linux (big endian, little endian)	SUSE Red Hat
First Supported	2Q/2013	2Q/2014	2Q/2013	4Q/2013	4Q/2013	2Q/2014	4Q/2013

1.4.3 IBM Cloud Manager with Openstack and SoftLayer

The IBM cloud offerings from SoftLayer® for IaaS provide unique visibility and control of your data and applications using managed services that can be hosted based on three different location options:

- Private (on-premises) cloud infrastructure
- Public (off-premises) cloud infrastructure
- Hybrid cloud infrastructure

> **Note:** IBM provides a complete hybrid cloud solution that offers full control, transparency, and full orchestration of business process. This solution seamlessly controls work as though it was all on-premises.

For additional information about IBM cloud offerings with SoftLayer, see:

http://www.ibm.com/cloud-computing/us/en/softlayer.html

IBM Cloud OpenStack Services are delivered through the IBM Cloud marketplace to enable interoperability between existing IT systems and off-premises cloud workloads. IBM Cloud OpenStack Services provides access to SoftLayer's infrastructure. The SoftLayer infrastructure has a range of underlying hardware configuration options based on open standards and deploy services with a managed-services approach based on the needs of the workload.

For an overview of OpenStack deployment on SoftLayer, see:

http://www.ibm.com/marketplace/cloud/Hybrid-cloud-with-open-technology/us/en-us

1.5 IBM z/VM management components

IBM z/VM enables the sharing of the mainframe physical resources, such as disk, memory, network adapters, and CPUs. These resources are managed by the z/VM hypervisor, called the control program (CP).

The latest versions of z/VM were designed to offer support for open cloud architecture based interfaces with support of OpenStack.

1.5.1 System management APIs and interfaces

IBM z/VM 6.3 ships with different components inside the z/VM LPAR as shown in Figure 1-3. The z/VM LPAR includes the Directory Maintenance Facility for z/VM (DIRMAINT), Performance Toolkit (PERFKIT), Systems Management Service (SMAPI), and the Extreme Cloud Administration Toolkit (xCAT).

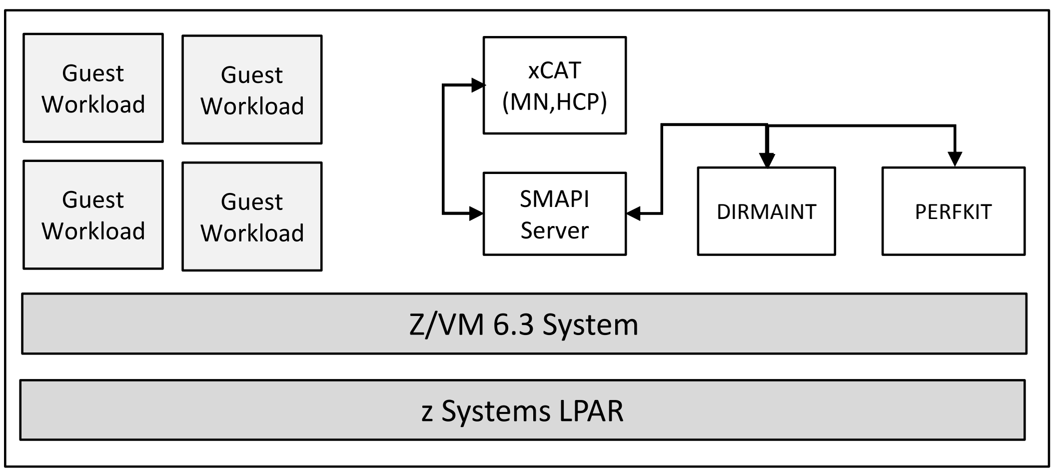


Figure 1-3 IBM z/VM 6.3 System Management APIs and Interfaces

DIRMAINT

The Directory Maintenance Facility for z/VM (DIRMAINT) provides efficient and secure interactive facilities for maintaining your z/VM system directory. Directory management is simplified by the DIRMAINT command interface and automated facilities. DIRMAINT provides a corresponding command for every z/VM directory statement, including z/VM single system image (SSI) cluster directory statements. The DIRMAINT error checking ensures that only valid changes are made to the directory and that only authorized personnel are able to make the requested changes.

PERFKIT

The Performance Toolkit (PERFKIT) assists z Systems operators and systems programmers or analysts on tasks such as processing VM history files, performance monitoring, and TCP/IP performance reporting.

SMAPI

The z/VM Systems Management Application Programming Interface (SMAPI) simplifies the management of z/VM with a standardized, platform-independent programming interface that can simplify and reduce the administration work on the environment.

xCAT

The Extreme Cloud Administration Toolkit (xCAT) is an open source tool integrated as part of the base of z/VM V6.3 to manage, provision, and monitor virtual machines in the cloud environment.

OpenStack and IBM Cloud Manager on z Systems (by using the zvm driver) uses xCAT, which in turn uses SMAPI, to provide a complete cloud management capability for the z/VM environment.

zHardware Control Program (zHCP)

zHCP manages other VMs through SMAPI and CP commands. Each z/VM system must have a zHCP.

xCAT Management Node (xCAT MN)

An OpenStack nova-compute service manages the xCAT MN. xCAT MN works as a central management server and connects to zHCP. Only one xCAT MN is needed for multiple systems.

Figure 1-4 shows how an xCAT MN node manages two different zHCP agents in two z/VM LPARs.

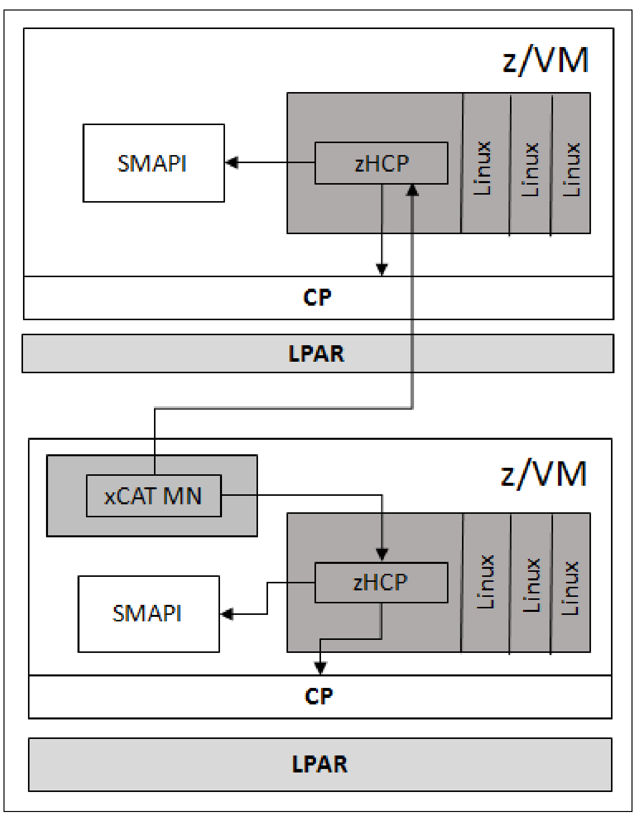


Figure 1-4 z/VM Management with xCAT

1.6 IBM Cloud Manager with OpenStack for z Systems

IBM Cloud Manager with OpenStack for z Systems, V4.2 is an easy to use cloud management solution that serves as a control point for cloud managed resources based on the OpenStack distribution.

New functions in IBM Cloud Manager with OpenStack for z Systems, V4.2 enable IBM z Systems to operate as the cloud management hub. This hub can manage IBM z Systems, IBM Power Systems, and x86 computing resources from one central point of control.

1.6.1 Benefits and capabilities

IBM Cloud Manager with OpenStack for z Systems, V4.2 is an innovative, cost effective approach for organizations and service providers that also includes security, automation, metering, and integrated platform management. It is designed to help cloud service providers achieve the following benefits:

- Lower administrative costs through simplified cloud administration with an intuitive interface.
- Improve operations productivity with an easy, intuitive user interface.
- Reduce management costs and improve responsiveness to changing business needs by standardizing virtual machines and images.
- Improve the usage of your infrastructure by consolidating and automating highly optimized IBM systems and software.

These benefits are achieved by the provision of the following key capabilities:

- An enhanced OpenStack experience, offering simplified implementation and advanced resource management that is backed by IBM support.
- Access to the latest OpenStack community releases.
- A single management interface to manage IBM z Systems, in addition to IBM Power Systems and x86 platforms.
- Support for additional IBM hardware configurations, including extra support for OpenStack drivers such as Cinder storage drivers.
- Extensibility through a REST API enables the tailoring of programs to satisfy unique business environments.
- A modular, flexible design that enables rapid innovation, interoperability, and freedom from vendor lock-in.

The announcement letter has more information about the benefits and capabilities offered by the IBM Cloud Manager with OpenStack for z Systems product. To view the announcement letter, go to the following website:

http://www.ibm.com/common/ssi/index.wss?request_locale=en

Select the Search Results tab, select **Announcement letters** for **Information type**, select **Letter Number** in the **Search in** field, and enter **215-052** in the **Search for** field. Click **Search**.

1.6.2 Environment architecture

The IBM Cloud Manager with OpenStack for z Systems is a combination of software tools that are deployed into a z/VM LPAR. When you install a cloud environment with the IBM Cloud Manager with OpenStack for z Systems, you work with the following components:

- **IBM Cloud Manager with OpenStack for z Systems** self-service portal and added value components: contains the features for VM instance management, metering, reports, and so on. It helps system users and administrators manage their cloud environments.
- OpenStack: Contains the services available for the platforms: Compute, storage, and network. The services include cinder, glance, keystone, nova, and neutron.
- IaaS gateway: A component that works as a gateway for IBM Cloud Manager with OpenStack for z Systems to manage the OpenStack cloud.
- xCAT MN (management node): Used for the deployment of the VMs.
- Chef: A cloud infrastructure automation framework that can assist with deploying instances and applications to a cloud environment, regardless of the size of the infrastructure.
- Chef cookbooks: Define sets of configurations (one or more recipes) for an instance in Chef,. Recipes are a specific configuration to be applied to an instance.

Figure 1-5 shows details about the interaction between the IBM Cloud Manager with OpenStack for z Systems components.

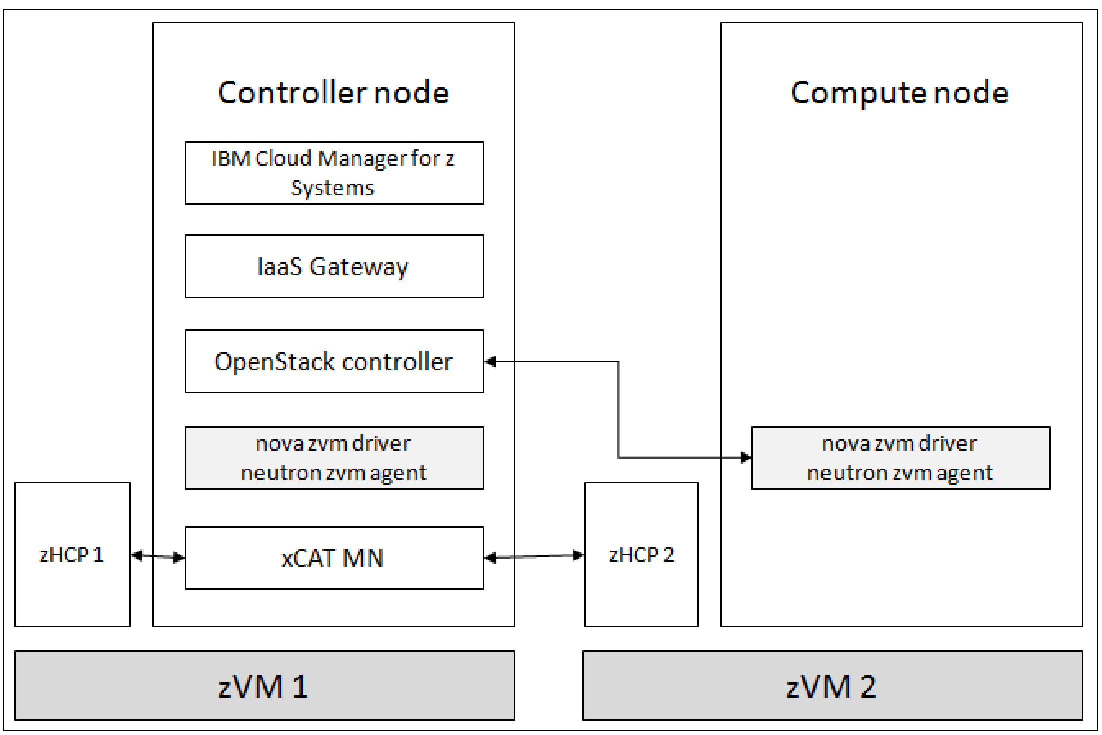


Figure 1-5 IBM Cloud Manager and OpenStack for z Systems reference architecture

For more details about the installation and configuration of these components, see Chapter 3, "Constructing the cloud foundation" on page 39.

1.6.3 IBM Cloud Manager with OpenStack for z Systems operation roles

IBM Cloud Manager with OpenStack for z Systems has two node classes. One node acts as the controller node in the cloud environment and all other nodes act as compute nodes. See “IBM Cloud Manager with OpenStack supported platforms” on page 9 for node class definitions.

Controller role

The z/VM LPAR with the controller role runs the cloud controller services, which include IBM Cloud Manager with OpenStack for z Systems, IaaS gateway, and OpenStack controller. It also operates the nova zvm driver and z/VM networking agent (neutron zVM agent).

Compute role

The z/VM LPAR with the compute role operates the compute service (nova-compute) and z/VM networking agent (neutron zvm agent) for the hypervisor.

> **Note:** If the cloud environment consists of multiple z/VM LPARs, one will run the controller role, while the others will run only the compute role (see Figure 1-5 on page 14).

1.6.4 OpenStack z/VM driver features

The OpenStack z/VM driver (also known as the nova zvm driver) uses OpenStack services to enable the following features:

- Managing virtual machines and hosts:
  - Provisioning virtual machine (Nova)
  - Resize a running virtual machine, increase memory or CPU assigned (Nova)
  - Update disk devices, add SCSI disk to virtual machine (Nova, Cinder)
  - Support for Open vSwitch (Neutron)
- Automated operations available for the user:
  - Start/Stop a virtual machine (Nova)
  - Restart Linux virtual machine (Nova)
  - Pause/Unpause virtual machine (Nova)
  - Capture/Deploy virtual machine (Nova, Glance)
  - Activate image (Nova)
- Business continuity:
  - Live guest relocation (Nova)

Figure 1-6 shows how the OpenStack services interact to provide these features to the z/VM LPAR (or instance).

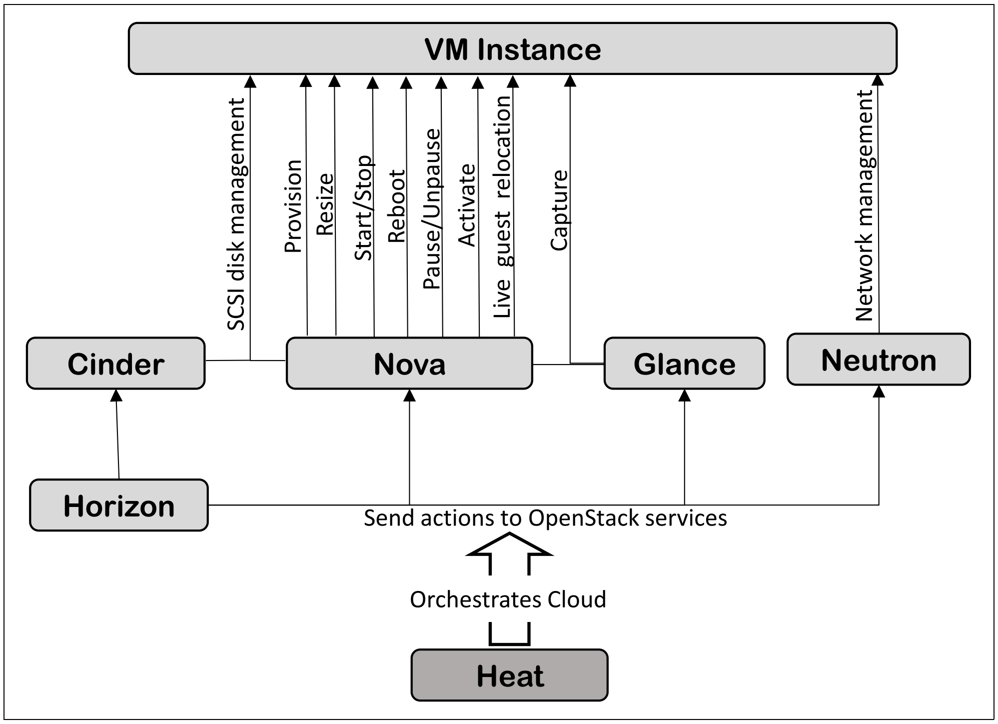


Figure 1-6 OpenStack z/VM driver features

1.6.5 Topologies for managing distributed platforms

When you manage distributed platforms (x86 or Power Systems) from z Systems, two different topologies can be applied:

- Single region topology

  You can manage x86 by using the *one controller node + n compute* nodes deployment topology. This topology means a set of OpenStack controller services (controller node) is on one z/VM hypervisor and the controller node manages all the other compute services (compute nodes), including other z/VM and x86 servers. This is also called a single region topology. The concept of OpenStack region in this book is that a cloud implementation has a full set of OpenStack services, including API endpoints, network, compute resources, and so on.

Figure 1-7 shows a single region topology.

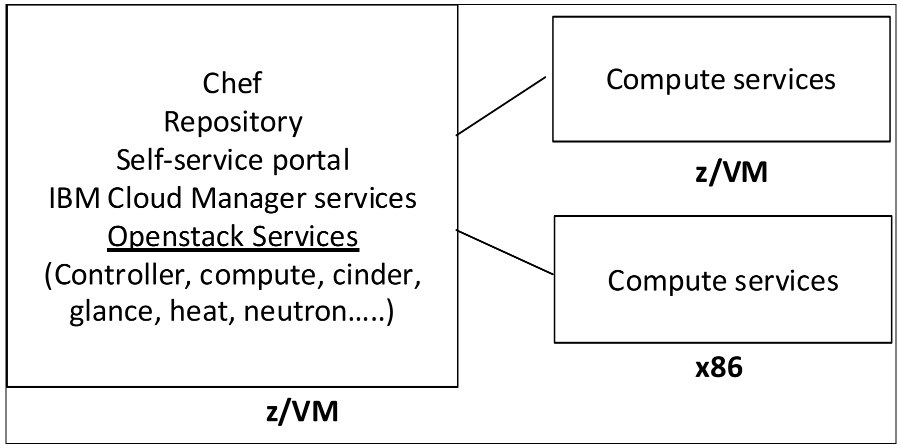


Figure 1-7 Single region topology

- Multi-region topology

  If you manage Power from z Systems, the cloud architecture should be implemented as s multiple-region topology. The multi-region stands for multiple sets of full OpenStack services as described in the previous section, but they share the OpenStack Keystone component. In the cloud on z Systems, the keystone is in the OpenStack controller node on a z/VM. See Figure 1-8 for a depiction of a multiple region architecture.

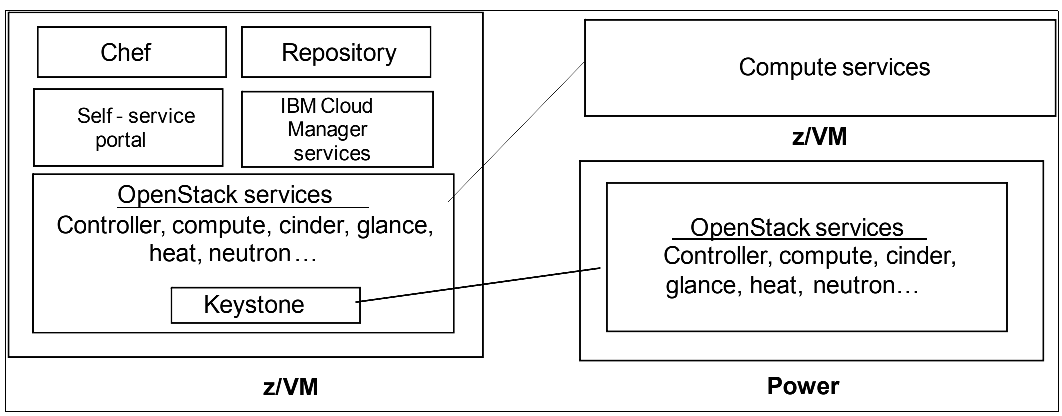


Figure 1-8 Multi-region topology

Note: Both single region and multi-region topologies use the same self-service portal to manage heterogeneous platforms.

1.6.6 Chef and HEAT environment

HEAT is the main component on the OpenStack orchestration program. It implements an orchestration engine to deploy multiple components to your cloud based on templates. The components that are deployed by the HEAT engine can be instances, network connections, IP addresses, volumes, and applications. These components define the stack to be deployed. The HEAT engine uses scripts to run actions in the instances to download, install, and configure the applications. The templates used by HEAT are created as text files that can be treated as source code. These templates are called Heat Orchestration Templates (HOTs).

You have two options when creating and deploying your HOT template:

- Use a simple text editor to create the `sample_hot_template.yaml` file, then use HEAT through the IBM Cloud Manager dashboard to deploy the template and create your stack.
- Use the IBM UrbanCode™ Deploy with Patterns product, which allows you to design and deploy full stack application environments for multiple clouds.

Chef is a leading system automation solution that turns infrastructure into code with an architecture that was built for extreme scale. Using Chef, you can automate how you build, deploy, and manage your infrastructure.

> **Note:** IBM Custom Patterns for Linux on z Systems are built on open standards based on Chef automation, making it easy to deploy and configure the middleware solutions.

Recipes

Chef relies on reusable definitions known as recipes to automate infrastructure tasks. Examples of recipes include instructions for configuring web servers, databases, and load balancers. Together, recipes describe what the infrastructure consists of and how each part of the infrastructure will be deployed, configured, and managed.

Resources

Recipes use building blocks called resources. A resource describes some piece of infrastructure, such as a file, a template, or a package to be installed.

Cookbooks

Cookbooks contain recipes, attributes, resources, libraries, files, templates (HOT), and any other artifact that might be needed by the Chef client when running installation/configuration of the nodes.

Software repo

The Software repo stores all required installation resources required when deploying the recipes.

Chef server

The Chef server stores cookbooks, the policies that are applied to nodes, and metadata that describes each registered node that is being managed by the Chef client.

Chef client

The Chef client is installed on each node in your network. Nodes use the Chef client to ask the Chef server for configuration details, such as recipes, templates, and file distributions. A node can be a physical server, a virtual server, or a container instance. The Chef client periodically communicates with the Chef server to check for the latest recipes and to verify whether the node is in compliance with the policy defined by the recipes.

> **Note:** The Chef server, Chef client, and HEAT components are included with IBM Cloud Manager with OpenStack for z Systems, V4.2.

Figure 1-9 shows the general workflow to deploy a pattern with IBM Cloud Manager with OpenStack for z Systems using Chef:

1. An administrator logs in to the IBM Cloud Manager Dashboard UI and starts the process to deploy a pattern.
2. The HEAT component receives the deployment request and, using the virtual network, it passes the automation script that describes the pattern to the Chef client in the Linux instance.
3. The Chef client on each Linux instance receives the script and requests all needed installation media and configuration resources from the Chef server/Chef repo.
4. The Chef repo provides the installation media and configuration resources to the Chef client.
5. The Chef client runs the installation/configuration on the Linux instance.

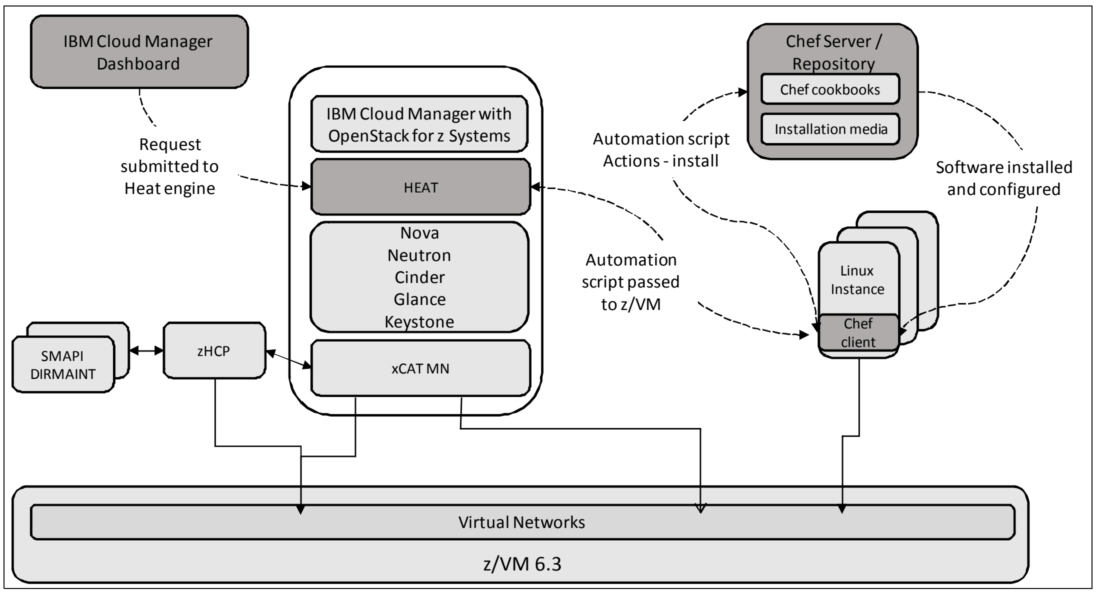


Figure 1-9 Using HEAT and Chef to deploy patterns

Note: After the pattern has been deployed, the HEAT template can be updated from HEAT or UrbanCode Deploy UI.

1.7 Use cases for IBM Cloud Manager with OpenStack for z Systems

IBM Cloud Manager with OpenStack for z Systems can be used in the IaaS and PaaS cloud service models. This section lists the use cases that are covered in subsequent chapters of this book.

Depending on the use case (IaaS or PaaS), IBM Cloud Manager with OpenStack for z Systems manages different levels of underlying infrastructure. Figure 1-10 shows the levels that are managed for each use case.

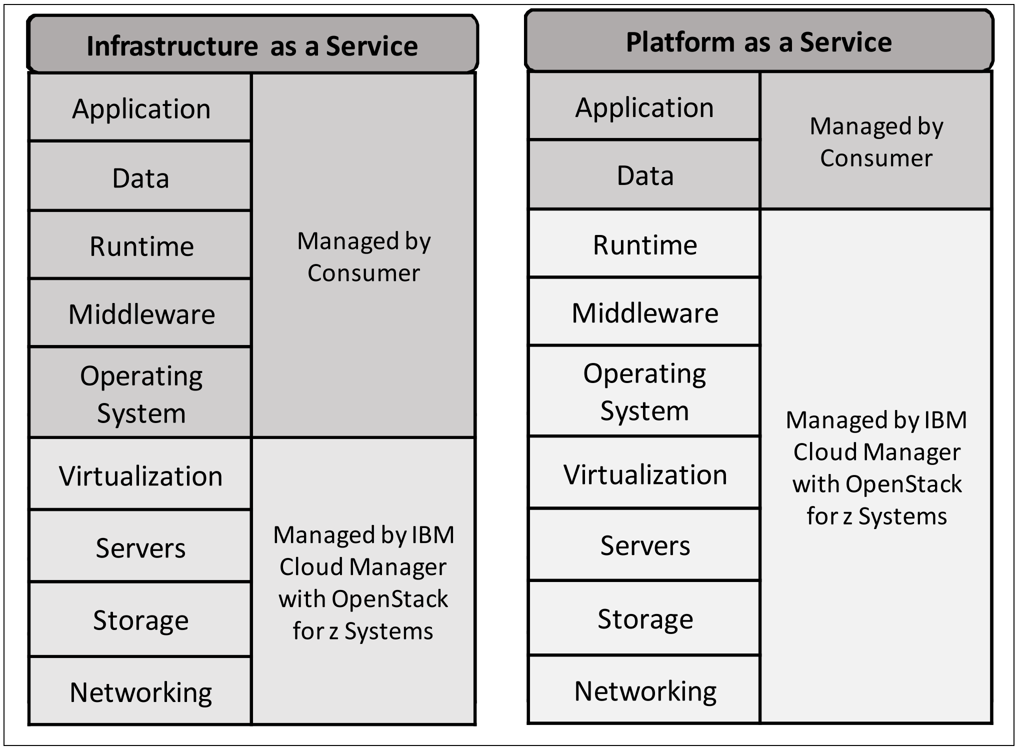


Figure 1-10 Underlying infrastructure levels of management for IaaS and PaaS

1.7.1 Use cases for IaaS

When managing your IaaS cloud infrastructure with the IBM Cloud Manager with OpenStack for z Systems, consider these options: Managing one or more z Systems, or managing x86 and Power Systems.

The subsequent chapters describe in detail how to install, configure, and manage your IaaS infrastructure based on these use cases:

- An IaaS administrator wants to manage a z Systems extended count key data (ECKD)-based environment by using IBM Cloud Manager with OpenStack for z Systems
- An IaaS administrator wants to manage a z Systems SCSI-based environment by using IBM Cloud Manager with OpenStack for z Systems
- An IaaS administrator wants to manage a x86 and Power Systems by using IBM Cloud Manager with OpenStack for z Systems

1.7.2 Use case for PaaS

You can use IBM Cloud Manager with OpenStack for z Systems to manage your PaaS infrastructure by using cloud patterns that are designed and tested to automate the deployment of your environment.

Chapter 6, “Adding patterns to the cloud infrastructure” on page 159 describes in detail how to manage and use the IBM Custom Patterns for Linux on z Systems. The chapter focuses on the use case: A PaaS administrator wants to manage IBM custom patterns by using IBM Cloud Manager with OpenStack for z Systems.

1.8 IBM Custom Patterns for Linux on z Systems

IBM Custom Patterns for Linux on z Systems uses good practices and lessons learned from several environment deployments to enable simplified and faster application deployment and management.

IBM Custom Patterns for Linux on z Systems provide the following capabilities:

- Reduce deployment errors/fixes: The patterns have already been tested and used in other environments.
- Reduce the need for deep product skills: The patterns reuse good practices from different environments, and are configured and ready to deploy.
- Improve quality of delivery: Use tested and improved predefined patterns.
- Reduces operating and capital expenses: Reduces the time to deploy your environment.
- IBM developed and tested service that provides standard deployment of applications in the organizations.

Note: IBM Custom Patterns for Linux on z Systems are built based on patterns of expertise that include proven practices and expertise learned from decades of client and partner engagements.

1.8.1 IBM Custom Patterns for Linux on z Systems offerings

IBM Custom Patterns for Linux on z Systems use HEAT and Chef. The following custom patterns are offered for Linux on z Systems:

- WebSphere Application Server Network Deployment V8.5.5 with Custom Pattern for Linux on z Systems
- WebSphere Application Server Liberty Core V8.5.5 with Custom Pattern for Linux on z Systems
- DB2 Enterprise Server Edition V10.5 with Custom Pattern for Linux on z Systems
- WebSphere MQ V8.0 with Custom Pattern for Linux on z Systems
- Integration Bus V9.0 with Custom Pattern for Linux on z Systems
- Decision Center V8.7 with Custom Pattern for Linux on z Systems
- Decision Server Advanced V8.7 with Custom Pattern for Linux on z Systems
- Process Center Advanced V8.5.5 with Custom Pattern for Linux on z Systems
- Process Server Advanced V8.5.5 with Custom Pattern for Linux on z Systems
- Business Monitor V8.5.5 with Custom Pattern for Linux on z Systems
- WebSphere Portal Server V8.5 with Custom Pattern for Linux on z Systems
- MobileFirst Platform Foundation V6.3 with Custom Pattern for Linux on z Systems

For additional information about these patterns, review the announcement letter at:

http://www.ibm.com/common/ssi/index.wss?request_locale=en

Select the Search Results tab, select **Announcement letters** for **Information type**, select **Letter Number** in the **Search in** field, and enter **215-052** in the **Search for** field. Click **Search**.

IBM continues to work on more custom patterns for other products. These new patterns might be published as part of future versions of the IBM Custom Patterns for Linux on z Systems offering.

1.8.2 IBM UrbanCode Deploy with Patterns

IBM UrbanCode Deploy with Patterns allows you to design, deploy, and update full stack environments on multiple clouds. It can be connected to IBM Cloud Manager with OpenStack for z Systems cloud and used to design your HOT templates and create stacks. It offers the following capabilities:

- Pattern designer

  The designer enables you to design open, full stack application environments in an intuitive diagram or textual editor.

- Design once, deploy anywhere

  It provides the support to deploy full stack environments to multiple clouds.

- Environment lifecycle management

  Manages infrastructure changes and makes it easy to apply changes to existing environments.

- Delivery process automation

  Provides an automated delivery process with integrated full stack environments.

For more information about IBM UrbanCode Deploy with Patterns, review the official documentation at:

http://www.ibm.com/software/products/en/ucdep-patterns

2

Planning for cloud management on z Systems

This chapter describes how to plan for the implementation of IBM Cloud Manager with OpenStack for z Systems, V4.2. It also discusses design considerations for your cloud environment when OpenStack is used for provisioning and managing the cloud resources. In addition, the chapter provides implementation advice and checklists for your reference.

This chapter includes the following sections:

- Designing a cloud infrastructure with OpenStack in mind
- Planning resources for your cloud infrastructure
- Planning the implementation of your cloud infrastructure

2.1 Designing a cloud infrastructure with OpenStack in mind

It is important to design your cloud architecture based on the types of services that you will offer to the consumer. As part of the overall design, consider how your management tools will interact with the virtualized pool of resources (compute, storage, and networking).

This section describes how to design a cloud environment from an OpenStack perspective to provide an infrastructure as a service (IaaS) and platform as a service (PaaS) environment.

There are numerous cloud infrastructure options. However, there are also some common preferred practices, such as isolating management data traffic from consumer data traffic. So, when you design your cloud infrastructure, consider using multiple data networks, each having a different function or role.

Because different storage infrastructures and protocols can be used in your environment, design your cloud architecture accordingly. This section covers cloud reference architectures for extended count key data (ECKD) and Small Computer Serial Interface (SCSI) storage infrastructures:

- “ECKD based cloud infrastructure on z Systems”
- “SCSI-based cloud infrastructure on z Systems” on page 26

> **Note:** Although not described in this book, you can design a cloud infrastructure with combined ECKD and SCSI storage devices.

In addition, if you have other platforms, such as x86 or Power Systems, you can manage them from z Systems, creating a single control point using a self-service portal. For more information about how to design a heterogeneous cloud infrastructure, see “Managing x86 and Power Systems from z Systems” on page 28.

After you have established an IaaS, you might want to expand that service offering to PaaS. For more information about designing a z Systems cloud infrastructure with PaaS capabilities, see “Expanding your cloud with software patterns” on page 28.

2.1.1 ECKD based cloud infrastructure on z Systems

In an ECKD-based environment, you can use the z/VM Single System Image and Live Guest Relocation features. This approach allows you to easily move a Linux instance from one z/VM logical partition (LPAR) to another using the cloud self-service portal.

With an ECKD based environment, define the necessary storage devices for Linux instances (root file system) in a storage group in DIRMAINT. As described in 2.2.3, “Storage planning considerations” on page 31, you can mix different storage sizes in a group. IBM Cloud Manager with OpenStack for z Systems selects the proper volume size.

IBM Cloud Manager with OpenStack for System z manages DIRMAINT storage pools for ephemeral storage, if you want to assign more disk space or persistent storage to the Linux instances, you have the following options:

- Perform Linux instance resize by using the self-service portal to increase the root file system.
- Using IBM Wave for z/VM to add ECKD disks to a Linux instance by either creating Logical Volume Managers (LVMs) or adding disks to new mount points.

- Using IBM Cloud Manager with OpenStack for z Systems to attach SCSI (persistent storage) disks to a Linux instance.
- Using scripts or recipes in ICM user data or a Heat pattern to add more storage at deployment time.

Figure 2-1 depicts a common cloud infrastructure with ECKD storage devices and multiple data networks.

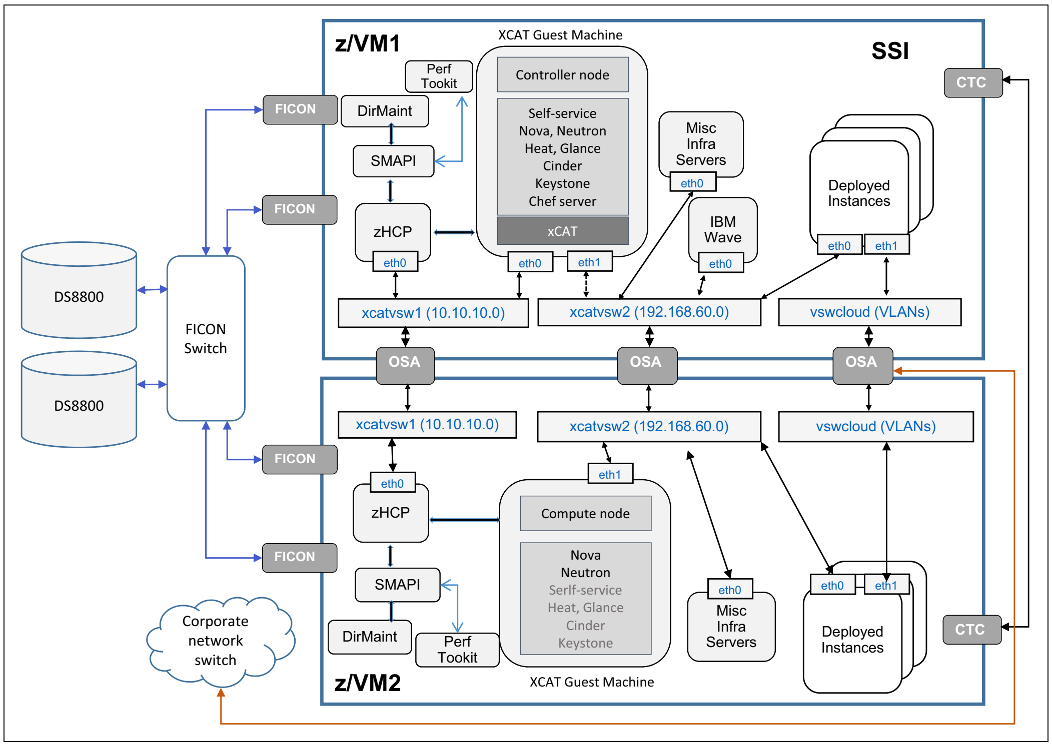


Figure 2-1 ECKD based cloud infrastructure reference architecture

This cloud infrastructure is based on a two-member single system image (SSI) environment. You can expand the infrastructure up to four z/VM members across multiple z Systems platforms.

The example separates management data traffic and consumer data traffic across three networks. The networks are built using three virtual switches:

- XCATVSW1 is used for System z Hardware Control Program (zHCP) and Extreme Cloud Administration Toolkit (xCAT) internal communication only.
- XCATVSW2 is used for management purpose. It is defined as a Layer 2 virtual switch, so you can configure the management network either as a flat network or with a virtual local area network (VLAN).
- VSWCLOUD is used for a consumer data network and it is also defined as a Layer 2 virtual switch. In the example environment, it is configured in VLAN aware mode and uses IBM Cloud Manager with OpenStack for z Systems to manage multiple VLANs. You can isolate Linux instances into different groups.

When you deploy a Linux instance, IBM Cloud Manager with OpenStack for z Systems creates two virtual network interface cards (NICs) on the Linux: One for the management network and the other for the consumer data network.

Note: You can create more than one consumer data network for your Linux instances.

The storage controllers are attached to the z/VM LPARs by using IBM Fibre Connection (FICON®) switches. Two channels are defined between each storage controller and each LPAR. You can increase the number of FICON channels according to your I/O requirements.

This scenario also uses IBM Wave for z/VM to add and manage additional disk capacity to the Linux instances. The z/VM performance toolkit is used to monitor the performance of z/VM, cloud service machines, and Linux instances.

2.1.2 SCSI-based cloud infrastructure on z Systems

In a SCSI only environment, use Fixed Block Access (FBA) for ephemeral storage and Fibre Channel Protocol (FCP) devices for persistent storage. Figure 2-2 illustrates a common cloud infrastructure with SCSI storage devices and multiple data networks.

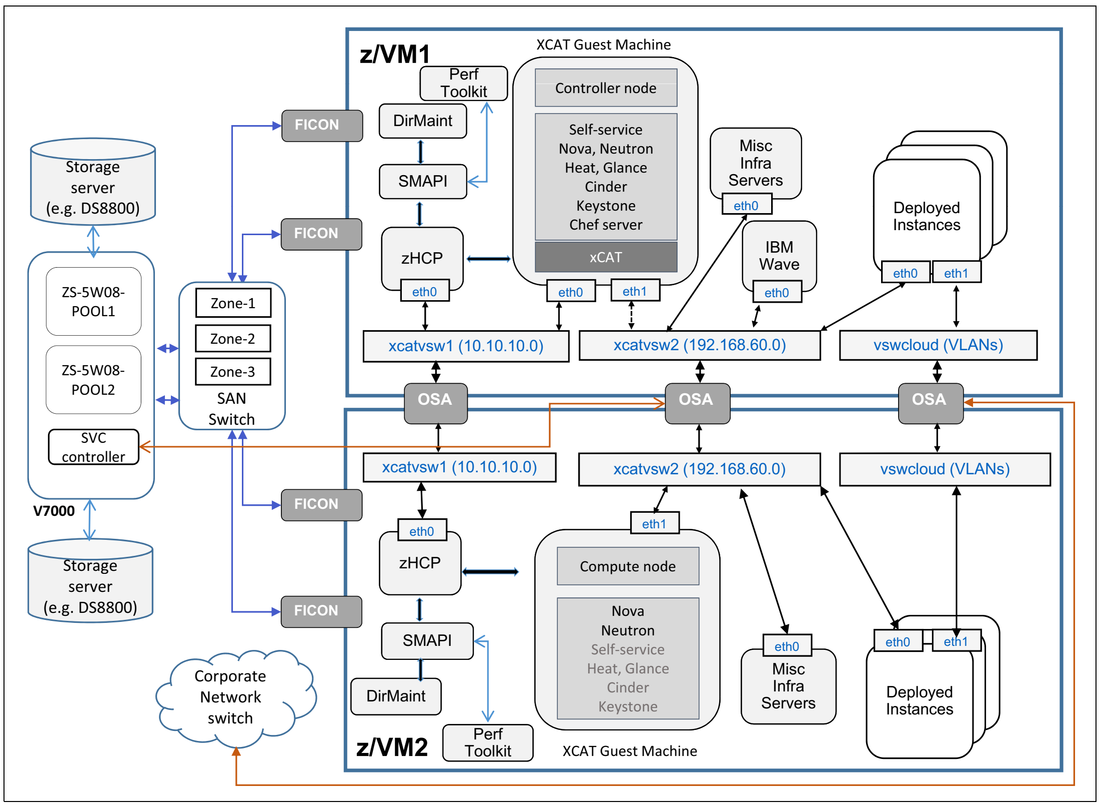


Figure 2-2 SCSI-based cloud infrastructure reference architecture

The examples use IBM Cloud Manager with OpenStack for z Systems to manage two z/VM LPARs. Both z/VM LPARs are installed on SCSI disks (FBA disks). You can manage multiple z/VM LPARs in a single cloud.

The example separates management data traffic and consumer data traffic across three networks. The networks are built using three virtual switches:

- XCATVSW1 is used for zHCP and xCAT internal communication only.
- XCATVSW2 is used for management purpose. It is defined as a Layer 2 virtual switch, so you can configure the management network either as a flat network or with a VLAN.

- VSWCLOUD is used for consumer data network and is defined as a Layer 2 virtual switch. The example environment is configured in VLAN aware mode, and uses IBM Cloud Manager with OpenStack for z Systems to manage multiple VLANs. You can use isolate Linux instances into different groups.

When you deploy a Linux instance, IBM Cloud Manager with OpenStack for z Systems creates two virtual NICs on the Linux: One for the management network and the other for the consumer data network.

Note: You can create more than one consumer data network for your Linux instances.

Two zones were defined in the storage area network (SAN) switch:

- zone-1 contains the logical unit numbers (LUNs) that are defined for FBA disks, which are for z/VM systems, services, service machines, IBM Cloud Manager with OpenStack for z Systems, and ephemeral storage for Linux instances.
- zone-2 contains the LUNs that are created and managed by OpenStack for Linux instances persistent storage.

In the SAN Volume Controller, two disk pools were defined:

- ZS-5W08-POOL1 is for FBA disks, which is used for z/VM and ephemeral storage. The disks in this pool are defined with a LUN size of 10 GB. The FBA disks for Linux ephemeral storage are defined in DIRMAINT.
- ZS-5W08-POOL2 is for FCP disks, which is used for persistent storage and managed by IBM Cloud Manager with OpenStack for z Systems. There are no disks defined in this pool. You can create any designated LUN size by using the cloud self-service portal and attach the disk to Linux instances. IBM Cloud Manager with OpenStack for z Systems creates the LUNs in that disk pool in the SAN Volume Controller and connects them to the hosts automatically.

In this scenario, IBM Wave for z/VM was used to add and manage extra disk capacity to the Linux instances. And the z/VM performance toolkit was used to monitor the performance of z/VM, cloud service machines, and Linux instances.

2.1.3 Managing x86 and Power Systems from z Systems

If you have various platforms (x86 and Power Systems) in your cloud environment and want to manage them by using IBM Cloud Manager with OpenStack for z Systems, you can do so. Figure 2-3 shows a cloud infrastructure scenario managing x86.

In this reference architecture, you can see an x86 server is connected to the ECKD-based cloud infrastructure on z Systems (SCSI-based cloud infrastructure is also supported). The OpenStack controller on the z/VM xCAT guest machine manages the x86 KVM hypervisor through the management network (XCATVSW2).

Note: A similar architecture is used to manage Power Systems.

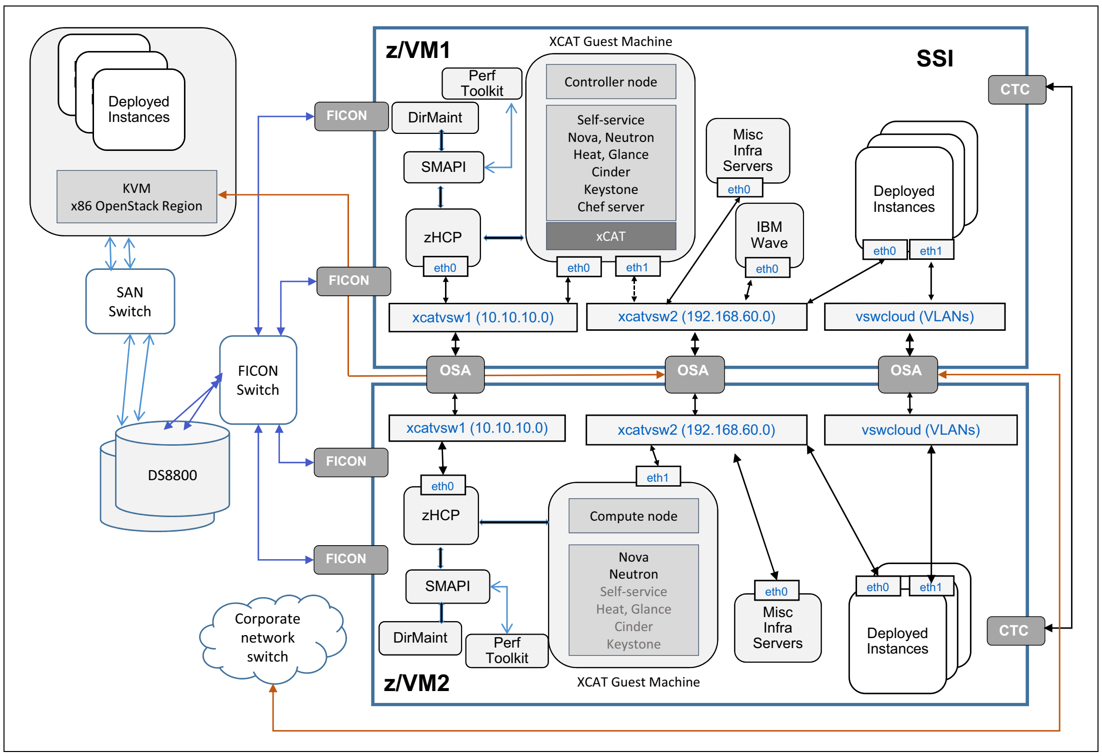


Figure 2-3 Heterogeneous cloud infrastructure reference architecture

2.1.4 Expanding your cloud with software patterns

IBM Cloud Manager with OpenStack for z Systems can provide PaaS through IBM Custom Patterns for z Systems. If you buy the patterns from IBM, you get the software licenses and the associated services.

The Custom Patterns for Linux on z Systems requires OpenStack Heat, a Chef server, and the pattern and software repository service as basic supporting elements. For more information, see “IBM Custom Patterns for Linux on z Systems” on page 21.

The software pattern deployment does not depend on disk type, so it can be used for both ECKD and SCSI environments. The reference architecture for software patterns shown in Figure 2-4 uses ECKD.

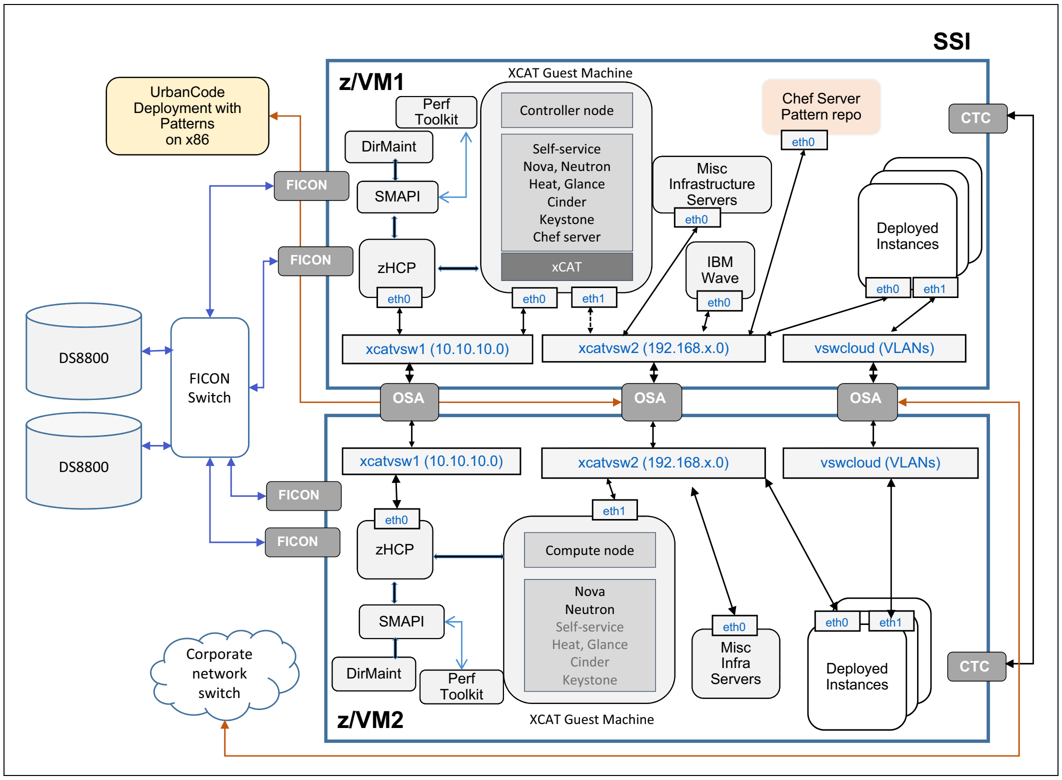


Figure 2-4 Preferred reference architecture for patterns on z Systems

The only difference in this scenario compared to the previous scenarios is that it introduces a new Custom Pattern repository service, a Chef server, and IBM UrbanCode Deploy with Patterns on two Linux systems.

Although you can repurpose the Chef server in the IBM Cloud Manager with OpenStack for z Systems, this example uses a separate Chef server for pattern deployment. One of the reasons is that the Chef server integrated in IBM Cloud Manager with OpenStack for z Systems is for IBM Cloud Manager with OpenStack deployment. It is part of the product and maintained by IBM. To avoid being affected by applying services to IBM Cloud Manager with OpenStack for z Systems or possible reinstallation, use a stand-alone Chef server instead. To install your own Chef server on a Linux on z Systems, you can get the corresponding Chef rpm package after you install the IBM Cloud Manager with OpenStack for z Systems Deployer. It is in the `/data/opt/ibm/cmwo/yum-repo/chef/s390x/` directory.

To deploy a Custom Pattern for Linux on z Systems, there are two options:

- Deploying patterns with IBM Cloud Management Dashboard
- IBM UrbanCode Deploy with Patterns

IBM Cloud Management Dashboard is part of IBM Cloud Manager with OpenStack for z Systems. You can use this dashboard to deploy patterns.

Use IBM UrbanCode Deploy with Patterns to deploy software patterns for these reasons:

- With IBM UrbanCode Deploy with Patterns, you can store your Heat Orchestration Template (HOT) in a Git repository for version control and access control purposes.
- IBM UrbanCode Deploy with Patterns provides a more efficient HOT editor.
- IBM UrbanCode Deploy with Patterns can work with IBM UrbanCode Deployment to create visualized and more easy-to-use patterns based on the existing pattern building blocks.
- IBM UrbanCode Deploy with Patterns can be a single point deployer to all private and public clouds, on-premises and off-premises. It can be used as a universal deployer.

For more information about deploying patterns using IBM UrbanCode Deploy with Patterns, see 6.5, “Using IBM UrbanCode Deploy with Patterns” on page 171.

2.2 Planning resources for your cloud infrastructure

When OpenStack is used for provisioning and managing the cloud resources, careful thought needs to go into the way you plan to use resources in your cloud infrastructure.

2.2.1 Compute planning considerations

To support your compute needs in your cloud environment, the proper CPU and memory sizing for your z Systems platforms should be based on, but not limited to, these factors:

- The number of Linux instances
- The workload types on Linux instances
- Peak time of your workloads

2.2.2 Network planning considerations

Planning of your cloud network environment should be in accordance with your corporate policies and standards. The following are some points for your reference:

- IBM Cloud Manager with OpenStack for z Systems manages the network by using virtual switches within z/VM. Proper planning of z/VM virtual switches for your cloud environment is important. You can define multiple virtual switches to separate management data traffic from consumer data traffic. Both management network and consumer data networks can be in a flat network or VLANs.
- You need to look at Open System Adapter (OSA) ports for the z/VM LPARs from high availability, bandwidth, and security standpoints. Be aware that trunk ports are required for multiple VLANs.

> **Note:** By default, IBM Cloud Manager with OpenStack for z Systems uses XCATVSW2 as the management network virtual switch. XCATVSW1 is used for xCAT and zHCP internal communication.

- IBM Cloud Manager with OpenStack for z Systems requires a network between the OpenStack controller and a deployed Linux instance as a management network. By default, the XCATVSW2 is employed for that network. The management network configuration is shown in “Configure directory manager” on page 50. And you can also add multiple virtual switches to your cloud infrastructure as consumer data networks. For information about single network and multiple network configurations, see “Configure neutron data network” on page 76.

2.2.3 Storage planning considerations

The OpenStack architecture manages two kinds of disk types. For more information, see “OpenStack storage services options” on page 8:

- Ephemeral disk: IBM Cloud Manager with OpenStack on z Systems supports ephemeral disk by using both ECKD and SCSI disks. Following are planning considerations for ephemeral storage:
  - ECKD:
    - If you use ECKD storage in your cloud environment, you can use any storage server that supports ECKD device type, for example, IBM DS8800.
    - The storage devices assigned to the Linux instances are managed by z/VM directory manager, such as z/VM Directory Maintenance Facility (DIRMAINT).
    - You can mix different size of ECKD devices in the same storage group in the EXTENTION CONTROL file of DIRMAINT (for example, 3390-09, 3390-27, and 3390-54). One cloud environment can only use one storage group. So you need to put all the direct access storage devices (DASDs) in one storage group for your Linux instances. Put larger sizes of DASDs in the storage group if possible. You might want to have your Linux instances with a bigger root file system in certain circumstances when they are deployed.
  - SCSI:
    - IBM Cloud Manager with OpenStack for z Systems manages FBA devices as ephemeral disks.
    - Enough FBA size (LUN size) should be planned to host the root file system of planned Linux instances.
    - The FBA devices for Linux instances are also put in a storage pool in DIRMAINT EXTENT CONTROL file.
    - For more flexibility, consider putting bigger sizes of FBA devices in DIRMAINT.
- Persistent disk: IBM Cloud Manager with OpenStack for z Systems is able to manage a pool of FCP devices that can be attached to Linux to access SCSI LUNs after they are deployed. These are some planning considerations for persistent storage:
  - You can attach various storage servers to your SVCs as backend disk infrastructure, for example, IBM DS8800. This approach eliminates the storage server compatibility issue.
  - IBM Cloud Manager with OpenStack for z Systems manages SAN Volume Controller devices as its disk volumes and host mapping definitions for Linux instances on z Systems. You need to set up a passwordless Secure Shell (SSH) connection between the OpenStack controller node and SAN Volume Controller. The SAN Volume Controller Internet Protocol (IP) address should be accessible by the controller node IP address of IBM Cloud Manager with OpenStack for z Systems.
  - A spare disk pool must be created on a SAN Volume Controller before you configure the OpenStack cinder and nova services to create and manage the SCSI disks automatically. The disk pool size should be planned properly to meet your business needs. If your disk infrastructure of your cloud environment is SCSI only, from preferred practice perspective, you should create two disk pools on the SAN Volume Controller. One is for FBA devices (ephemeral disks), and the other is for native SCSI disks (persistent disks) managed by the OpenStack cinder and nova services.

– IBM Cloud Manager with OpenStack for z Systems supports N Port ID Virtualization (NPIV). Consider enabling NPIV on your z Systems platform, as this will allow each virtual machine to be managed independently from zoning, LUN masking, and security perspectives.

Note: If your z/VM and storage pool in DIRMAINT for Linux instances are ECKD, you can also use SCSI disks through SAN Volume Controller and let IBM Cloud Manager with OpenStack for z Systems attach designated sizes of LUNs to the Linux instances for user data purposes (for example, database workloads).

2.2.4 SAN switch considerations

If you are using SCSI disks for both FBA and FCP devices, consider creating zones in your SAN switches. From a preferred practice perspective, create three zones in your SAN switches for isolation reasons:

- Zone one is for FBA devices, which are used to accommodate z/VM systems and other service machines (for example, IBM Wave for z/VM and backup and restore services).
- Zone two is to group ephemeral disk LUNs for root file system of Linux instances.
- The third zone is for FCP devices, which are used to contain persistent data LUNs for the Linux instances.

Note: You can have more zones for the persistent disks. However, because IBM Cloud Manager with OpenStack for z Systems creates a unique SAN Volume Controller host mapping per LUN for a Linux instance each time, ensure only that instance has access to its own LUN volumes.

2.2.5 Linux on z Systems planning considerations

IBM Cloud Manager with OpenStack for z Systems is a pre-built image provided by IBM, so you do not need to buy additional Linux service licenses. It is based on an IBM Linux distribution. IBM provides the services on Linux and IBM Cloud Manager with OpenStack for z Systems.

For the Linux instances deployed by cloud, you need to buy the service license from Red Hat or SUSE. The following Linux distributions are supported by IBM Cloud Manager with OpenStack for z Systems:

- Red Hat Enterprise Linux (RHEL) 6.6
- Red Hat Enterprise Linux (RHEL) 6.5
- Red Hat Enterprise Linux (RHEL) 6.4
- Red Hat Enterprise Linux (RHEL) 6.3
- Red Hat Enterprise Linux (RHEL) 6.2
- SUSE Linux Enterprise Server (SLES) 11.3
- SUSE Linux Enterprise Server (SLES) 11.2

Note: IBM z13 has minimum Linux kernel requirements. You need to have your Linux image and instance kernels at the proper level. For information about the supported Linux kernel levels on z13 at web address, see:

http://www.ibm.com/systems/z/os/linux/resources/testedplatforms.html

2.2.6 Considerations for deployable images

Before you deploy Linux instances using the IBM Cloud Manager with OpenStack for z Systems, you need to build your own deployable images. This requirement means you need to install Linux guest machines manually and then capture the Linux guest machines as deployable images. To make deployable images, remember the following considerations:

- Linux device type either ECKD or FBA.
- The virtual disk address of the Linux root file system in the user directory of the guest machine should be 0100.
- The Linux root file system must not be on LVM and should be in a single partition. If there are multiple partitions on the root file system or the root file system is on an LVM partition, the volume size cannot be resized and must be deployed as the same size as it was created. Otherwise, it will not boot successfully.
- You could also install the necessary software products or tools in Linux before you capture it as a deployable image. However, evaluate whether the software or tools will be affected after the IP address or host name has been changed. For those software or tools that are affected after the image is deployed, consider deploying software by using orchestration. For information about software deployment, see Chapter 6, “Adding patterns to the cloud infrastructure” on page 159.

2.2.7 Planning for managing distributed platforms

IBM Cloud Manager with OpenStack for z Systems supports managing distributed platforms (for example, x86 or Power Systems) from z Systems. As covered in “Topologies for managing distributed platforms” on page 16, you can use different kinds of topology to manage multiple platforms. You need to plan well before you implement a heterogeneous cloud infrastructure. Consider the following factors:

- IBM Cloud Manager with OpenStack for z Systems can manage KVM, HyperV, PowerVC, and PowerVM. You need to plan your x86 or Power Systems infrastructure well, including network and storage.
- To manage other platforms, you need to install IBM Cloud Manager with OpenStack Deployer on the OpenStack controller node (xCAT MN). The deployer installation file comes with IBM Cloud Manager with OpenStack for z Systems. For installation instructions, see 3.3.11, “Install IBM Cloud Manager with OpenStack Deployer” on page 64. You can also check the related information from CMA42.FILE on the MAINT 400 minidisk on your z/VM system.
- You can manage x86 from the controller node (xCAT MN) on z Systems using a single-region topology. However, you must use a multi-region environment to manage Power from z Systems. This is due to the use of different messaging queue mechanisms in the OpenStack services between z Systems and Power.
- In both single-region and multi-region topologies, you must create a Chef topology and environment files to reflect your infrastructure.
- In both single-region and multi-region topologies, you must retrieve and update Chef data bags for the compute nodes using the keystone admin passwords of the existing cloud environment on z Systems. This is required because the managed distributed platforms share the same keystone on the controller on z Systems (xCAT MN).

> **Note:** A data bag is a global variable that is stored as JSON data and is accessible from a Chef server. IBM Cloud Manager with OpenStack uses data bags to set the keystone credentials during the OpenStack services deployment using Chef.

- Both the OpenStack controller node (xCAT MN) on z Systems and the compute node on x86 or Power Systems must be accessible over the management network so that the compute nodes can communicate with the xCAT MN.

2.3 Planning the implementation of your cloud infrastructure

Before you implement any of the cloud scenarios described in 2.1, “Designing a cloud infrastructure with OpenStack in mind” on page 24, consider starting with a simple scenario for evaluation purposes, then phase your cloud environment and services into production. This section lists the tasks and phases to implement a cloud environment.

The following are the basic phases:

- Constructed the cloud foundation
- Built out the cloud infrastructure
- Configured, managed, and used the self-service portal
- Added patterns to the cloud infrastructure

2.3.1 Construct the cloud foundation

This section provides checklists for constructing a cloud foundation with IBM Cloud Manager with OpenStack for z Systems. The first checklist gives the tasks for preparing the z/VM environment, and the second checklist provides the tasks necessary for installing IBM Cloud Manager with OpenStack for z Systems. These checklists match the steps documented in Chapter 3, “Constructing the cloud foundation” on page 39.

Preparing the z/VM environment

- ☐ Preparing z/VM for IBM Cloud Manager with OpenStack for z Systems
- ☐ Preparing resources for IBM Cloud Manager with OpenStack
- ☐ Apply the latest z/VM Recommended Service Upgrade (RSU) package
- ☐ Apply the z/VM APARs required by IBM Cloud Manager with OpenStack for z Systems
- ☐ Prepare disk pools
- ☐ Verify SMAPI configuration
- ☐ Configure directory manager
- ☐ Enable External Security Manager[1]
- ☐ Configure Open Systems Adapter (OSA) equivalency identifier (EQID) for z/VM Single System Image (SSI)[2]

Installing IBM Cloud Manager with OpenStack for z Systems

- ☐ Download IBM Cloud Manager with OpenStack for z Systems image file
- ☐ Create z/VM minidisks for installation
- ☐ Upload IBM Cloud Manager with OpenStack for z Systems image file
- ☐ Unpack uploaded CMA4202 image file
- ☐ Restore unpacked CMA4202 image file
- ☐ Increase xCAT z/VM user ID memory

[1] Based on your corporate security policies
[2] This step is not needed for a SCSI only environment

☐ Customize DMSSICNF COPY configuration file

☐ Customize DMSSICMO COPY configuration file

☐ Put changes on production

☐ Bring up and validate IBM Cloud Manager with OpenStack for z Systems

☐ Install IBM Cloud Manager with OpenStack deployer

☐ Activate XCAT and ZHCP exits

Adding compute nodes to the cloud infrastructure

☐ Configure an additional z/VM compute node to the cloud

2.3.2 Build out the cloud infrastructure

After the cloud foundation is built, you can use the command-line interface (CLI) to manage the cloud environment. However, it is still a basic cloud environment. To build out a cloud infrastructure that includes multiple network and storage configurations, Linux images, and enables other services, you must complete some additional tasks. This section provides checklists to build out the cloud infrastructure and services. You can use these checklists to help track the steps documented in Chapter 4, “Building out the cloud infrastructure” on page 75.

Configure neutron data network

☐ Configure neutron Mezzanine LAN-on-Motherboard Generation 2 (ML2) plug-ins

☐ Configure neutron z/VM agent

☐ Restart the neutron server and neutron z/VM agent

Configure cinder persistent data disks

☐ Create a SAN switch zone

☐ Set up SSH connection between xCAT management node (MN) and SVC

☐ Configure nova.conf file

☐ Configure cinder.conf file

☐ Restart the nova compute and cinder services

☐ Create cinder volume type

Capture a deployable Linux image into OpenStack glance

☐ Install Linux on z Systems on a virtual machine

☐ Installation and configuration of the enablement framework

☐ Capture the node to generate the image in the xCAT MN

☐ Define the Linux image to Glance

Configure email notification

The steps for this task can be found in 4.4, “Configure email notification” on page 111.

Configure metering

The steps for this task can be found in 4.5, “Configure metering” on page 112.

Configure billing

The steps for this task can be found in 4.6, “Configure billing” on page 114.

2.3.3 Configuring, managing, and using the self-service portal

After building out the cloud infrastructure, you can configure, manage, and use the cloud services with the self-service portal. This section lists the tasks that are needed to enable the cloud services in the self-service portal. You can use these lists to help track the steps that are documented in Chapter 5, “Configuring, managing, and using the self-service portal” on page 117.

Self-service portal configurations

- ☐ Create a cloud environment in the self-service portal
- ☐ Create networks in the self-service portal
- ☐ Enable email notification in the self-service portal

Self-service portal administrator management

- ☐ Request management
- ☐ Project management
- ☐ Account management
- ☐ User management
- ☐ Image management
- ☐ Capacity management
- ☐ Instance migration
- ☐ Volume management

Self-service portal operations

- ☐ Request user account
- ☐ instance management
- ☐ Capture instance to image
- ☐ Deploy image to instance
- ☐ Withdraw or resubmit the request

2.3.4 Add patterns to the cloud infrastructure

With the completion of the previous three phases, you have an IaaS to serve your consumers. If you want to offer PaaS on z Systems, IBM provides software patterns and related services so that you do not have to create and set up pattern services yourself. Included in this section are steps for adding software patterns to your z Systems cloud infrastructure. You can use these lists to help you track the steps that are documented in Chapter 6, “Adding patterns to the cloud infrastructure” on page 159.

Prerequisites for deploying IBM Custom Patterns

- ☐ Install the Chef server and client
- ☐ Configuring the software repository for Linux instances
- ☐ Customizing the cookbook attributes

- ☐ Upload cookbooks to the Chef server
- ☐ Prepare the software repository for the patterns

Defining HOT templates

- ☐ Add parameters to the HOT template
- ☐ Add resources to the HOT template
- ☐ Execute scripts in the instance

Using IBM Cloud Manager Dashboard to deploy patterns

- ☐ Deploying patterns
- ☐ Monitor the progress of the deployment

Using IBM Urban Code Deploy with Patterns

This task is optional and is only applicable if IBM UrbanCode Deploy with Patterns will be used as a pattern management and deployment engine. The steps for this task can be found in “Leveraging IBM UrbanCode Deploy with Patterns” on page 167.

3

Constructing the cloud foundation

This chapter provides the information needed to prepare your z/VM environment for the installation of IBM Cloud Manager with OpenStack for z Systems. It also gives step-by-step instructions to help you install IBM Cloud Manager with OpenStack for z Systems, 4.2.

This chapter contains the following sections:

- Overview of scenario environment
- Preparing the z/VM environment
- Installing IBM Cloud Manager with OpenStack for z Systems
- Adding compute nodes to the cloud infrastructure

If you are not familiar with the OpenStack capabilities on z Systems, see “IBM Cloud Manager with OpenStack for z Systems” on page 13.

3.1 Overview of scenario environment

Before the details for implementing IBM Cloud Management with OpenStack for z Systems are discussed, the environment used to build the scenarios in this book is designed. The reference architectures for these scenarios can be found in "Designing a cloud infrastructure with OpenStack in mind" on page 24.

Figure 3-1 illustrates the environment that supported the example cloud infrastructure.

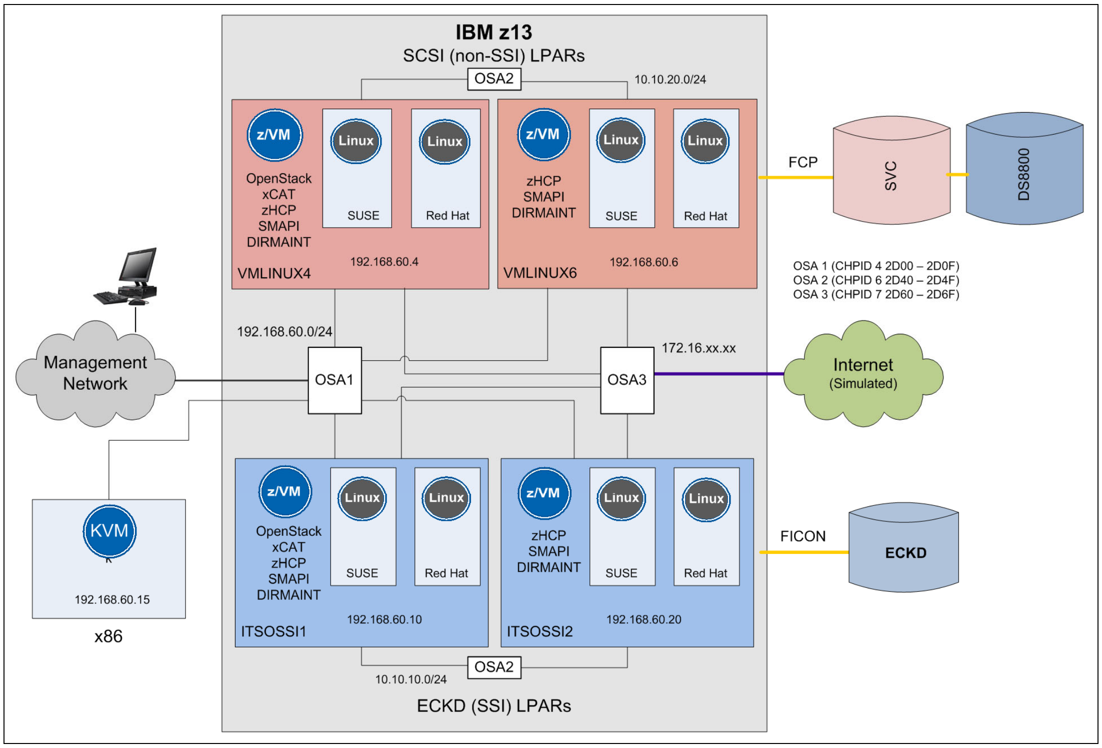


Figure 3-1 Overview of infrastructure for the environment

Figure 3-1 shows there are two LPARs for ECKD based SSI and two LPARs for SCSI. Both environments have three OSA ports (CHPIDs 04, 06, and 07) that are connected to three different networks. Then, a DS8800 is attached to the SSI environment and SAN Volume Controller devices are attached to the SCSI environment. There is an x86 server for the distributed environment scenario described in Appendix A., "Managing a distributed cloud environment from z Systems" on page 179.

Below are the worksheets used for the example cloud environment. The configuration information is used in the subsequent chapters of this book as well as you build out the environment from a basic IaaS scenario to a PaaS scenario with a self-service portal.

Table 3-1 provides network information that pertains to the z/VM SSI (ECKD) cloud environment used in all the example scenarios.

Table 3-1 z/VM SSI (ECKD) network environment worksheet

z/VM system ID	Service	OSA devices	IP address / LAN	Virtual switch
ITSOSSI1	TCPIP	2D00	192.168.60.10	vsw1
ITSOSSI1	xCAT MN (controller)	2D03	192.168.60.101	xcatvsw2

z/VM system ID	Service	OSA devices	IP address / LAN	Virtual switch
ITSOSSI1	zHCP	2D63	10.10.10.10	xcatvsw1
ITSOSSI1	xCAT to zHCP	2D63	10.10.10.20	xcatvsw1
ITSOSSI1	Data network	2D43	172.16.20.0/24 (VLAN100) 172.16.40.0/24 (VLAN200)	vswcloud
ITSOSSI2	TCPIP	2D00	192.168.60.20	vsw1
ITSOSSI2	Cloud compute	2D03	192.168.60.102	xcatvsw2
ITSOSSI2	zHCP	2D63	10.10.10.21	xcatvsw1
ITSOSSI2	Data network	2D43	172.16.20.0/24 (VLAN100) 172.16.40.0/24 (VLAN200)	vswcloud

Note: In an SSI environment, you need to define the EQID for OSA devices in the z/VM members before you implement the cloud foundation.

Table 3-2 provides network information that pertains to the z/VM SCSI cloud environment used in all of the example scenarios.

Table 3-2 z/VM SCSI network environment worksheet

z/VM system ID	Service	OSA devices	IP address / LAN	Virtual switch
VMLINUX4	TCPIP	2D00	192.168.60.4	vsw1
VMLINUX4	xCAT MN (controller)	2D03	192.168.60.201	xcatvsw2
VMLINUX4	xCAT to zHCP	2D63	10.10.20.10	xcatvsw1
VMLINUX4	zHCP	2D63	10.10.20.20	xcatvsw1
VMLINUX4	Data network	2D43	172.16.20.0/24 (VLAN100) 172.16.40.0/24 (VLAN200)	vswcloud
VMLINUX6	TCPIP	2D00	192.168.60.6	vsw1
VMLINUX6	Cloud compute	2D03	192.168.60.202	xcatvsw2
VMLINUX6	zHCP	2D63	10.10.20.30	xcatvsw1
VMLINUX6	Data network	2D43	172.16.20.0/24 (VLAN100) 172.16.40.0/24 (VLAN200)	vswcloud

For the scenarios in this book, Table 3-3 shows the ECKD devices for the IBM Cloud Manager with OpenStack for z Systems and new Linux instances.

Table 3-3 ECKD volumes worksheet for ITSOSSI1 and ITSOSSI2

Disk use	Device	VOLSER	MOD
IBM Cloud Manager with OpenStack for z Systems installation images	6003	CM6003	3390-27
xCAT LVM (controller data)	9810-9815	CM9810-CM9815	3390-09
Disk pool for ephemeral disks of Linux instances	6006-600F	CM6006-CM600F	3390-27

For the scenarios in this book, Table 3-4 shows the SCSI devices used for IBM Cloud Manager with OpenStack for z Systems and new Linux instances.

Table 3-4 SCSI disk worksheet for VMLINUX4 and VMLINUX6

Disk use	Device Type	Device ID	VOLSER
IBM Cloud Manager with OpenStack for z Systems installation images	FBA (FB-512)	3000, 3009	CM3000, CM3009
xCAT LVM (controller data)	FBA (FB-512)	3001-3003	CM3001-CM3003
Disk pool for ephemeral disks of Linux instances	FBA (FB-512)	3004-3008	CM3004-CM3008

Note: The LUN size for each FBA device is 10 GB in this scenario's cloud environment.

In the example environment, two FCP channels are defined (each channel has 32 FCP devices) with two channels for the Linux instances on z Systems (for a multipath configuration). The following list notes the FCP devices that are defined to VMLINUX4 and VMLINUX6:

- B600-B61F on CHPID 76
- B700-B71F on CHPID 77

To demonstrate the implementation of an x86 environment, a low-end desktop is used that has these characteristics:

- Host IP: `192.168.60.15`
- Network adapter: `eth0`
- Disk: `internal HDD`

The checklists for the preparation and installation tasks found in "Construct the cloud foundation" on page 34 are applied to all the subsequent sections in this chapter.

3.2 Preparing the z/VM environment

There are z/VM prerequisites for the installation of IBM Cloud Manager with OpenStack for z Systems. For z/VM Single System Image members, where some disks are shared, you must restore one new XCAT 101 minidisk on each z/VM node. Different configuration files are updated depending on whether your z/VM LPAR is an OpenStack controller node or a compute node (see 3.3, "Installing IBM Cloud Manager with OpenStack for z Systems" on page 52).

An OpenStack controller node loads all IBM Cloud Manager with OpenStack for z Systems, which are implemented components that are required to manage the cloud. An OpenStack compute node loads only the components that are required to be managed from the controller node.

It is important to note that compute nodes can only be managed from a controller node. Figure 1-5 on page 14 shows the controller node and compute node roles.

IBM Cloud Manager, with OpenStack for z Systems, can be installed on ECKD or SCSI disks. To maximize the OpenStack persistent disks feature implemented by the OpenStack component (cinder), a SCSI infrastructure is required.

3.2.1 Preparing z/VM for IBM Cloud Manager with OpenStack for z Systems

The first step in preparation is to ensure z/VM 6.3 meets the software requirements for the implementation of IBM Cloud Manager with OpenStack for z Systems, 4.2.

Applying prerequisite APARs to z/VM

IBM Cloud Manager with OpenStack for z Systems requires some mandatory APARs to be installed on z/VM. Before you apply the APARs, the latest z/VM RSU package should be applied.

Note: RSU1501 is the latest RSU level at the time of writing this publication.

To acquire the prerequisite APARs for IBM Cloud Manager with OpenStack for z Systems, V4.2, see this website:

http://www.vm.ibm.com/sysman/osmntlvl.html

At the time of writing this publication, the APARs in Table 3-5 must be applied.

Table 3-5 APARs for z/VM

APAR	PTF	Abstract
VM65721	UM34637	FIXES FOR ICM 4.2.0.2; xCAT, zhcp PTF date: 2015-06-25
VM65706	UM34590	SECURITY APAR PTF date: 2015-04-17
VM65676	UM34565	Download and install instructions for appliance at ICM 4.2, FP2 level - support for Juno NOTE: You might need to increase the size of the MAINT630 3D2 by 250 cylinders and MAINT630 400 by 150 cylinders of 3390 space. PTF date: 2015-03-13
VM65721 VM65669	UM34637 UM34564	Fixes for xCAT and zHCP PTF date: 2015-03-25 Provides user exits for xCAT and zHCP initialization PTF date: 2015-03-13

Important: Make sure that you apply APARs for xCAT, zHCP, and SMAPI before you apply the APARs in Table 3-5. For more information, see this website:

http://www.vm.ibm.com/sysman/xcmntlvl.html

If you want to manage multiple z/VM LPARs by using IBM Cloud Manager with Openstack for z Systems, all the z/VM LPARs must have the required APARs applied.

Enabling your z/VM services for cloud

To implement a cloud environment on z Systems platforms, you need to configure and enable these mandatory services on z/VM:

- z/VM System Management API (SMAPI): z/VM SMAPI is part of z/VM and provides API services to applications.
- z/VM directory manager: z/VM has a pre-built licensed directory manager feature called z/VM Directory Maintenance Facility (DIRMAINT).

> **Note:** You can also choose other directory managers from other vendors. The examples in this book use z/VM Directory Maintenance Facility as the default directory manager.

For more information about enabling and configuring SMAPI and DIRMAINT in z/VM, see the following publications:

- For z/VM SMAPI, see *z/VM: Systems Management Application Programming*, SC24-6234
- For z/VM Directory Maintenance Facility, see the z*/VM: Directory Maintenance Facility Tailoring and Administration Guide*, SC24-6190

3.2.2 Preparing resources for IBM Cloud Manager with OpenStack

The IBM Cloud Manager with OpenStack for z Systems is installed and running in the xCAT guest machine on z/VM. The xCAT guest machine is pre-built in z/VM and the default definition is 1 GB memory with two virtual CPs. To support IBM Cloud Manager with OpenStack for z Systems, 4.2, increase the memory of the xCAT guest machine to 8 GB.

For a large installation, you might also increase the number of virtual CPs and the memory size of the zHCP guest machine to increase provisioning performance.

The disk space that is required by IBM Cloud Manager with OpenStack for z Systems is divided into these two parts:

- Product base code: The base code of IBM Cloud Manager with OpenStack for z Systems requires a single 3338-cylinder minidisk in the xCAT guest machine.
- System and user data: Disk space for the cloud system and user data is determined mainly by the size and number of your Linux images. To accommodate the fix pack, cloud deployment server, Linux images, and user data, more minidisks are required.IBM Cloud Manager with OpenStack for z Systems attaches those minidisks to the xCAT guest machine and creates an Logical Volume Manager (LVM) for you automatically. You can add extra disk space to the LVM later. Consider using 50 GB for the LVM as a starting point.

3.2.3 Apply the latest z/VM RSU package

Verify that z/VM 6.3 service level 1501 (provided by 6304RSU) is installed using the QUERY CPLEVEL command as shown in Figure 3-2.

```
QUERY CPLEVEL
z/VM Version 6 Release 3.0, service level 1501 (64-bit)
```

Figure 3-2 Minimum z/VM RSU verification

If your environment is not at this required level, go to the IBM Shopz website to order the latest available z/VM 6.3 RSU, using the RSU PTF Number UM97630.

For more information about recommended service upgrades (RSUs), see the following websites:

- IBM Shopz website:

  http://www.ibm.com/software/ShopzSeries

- IBM z/VM 6.3 RSU service:

  http://www.vm.ibm.com/service/rsu/stk630.html

3.2.4 Apply the z/VM APARs required by IBM Cloud with OpenStack for z Systems

Before starting the IBM Cloud Manager with OpenStack for z Systems installation process, ensure that all required z/VM services are applied.

Log on as `MAINT630` and verify if the required serviceis applied in your environment using the APAR list in Table 3-5 on page 43.

Figure 3-3 shows how to use the **SERVICE** procedure for this task. Each PTF must be in production (PUT2PROD) status. If you have a z/VM Single System Image environment as in this example, verify the status of each z/VM SSI member, which in this case are `ITSOSSI1` and `ITSOSSI2`. Also notice that the required APARs are the same for ECKD and SCSI installations,

```
SERVICE ALL STATUS UM34564
VMFSRV1226I    PUT2PROD  05/28/15 11:52:38 ITSOSSI2
VMFSRV1226I    PUT2PROD  05/28/15 11:49:32 ITSOSSI1
VMFSRV2760I SERVICE processing completed successfully

SERVICE ALL STATUS UM34565
VMFSRV1226I    PUT2PROD  05/28/15 11:52:38 ITSOSSI2
VMFSRV1226I    PUT2PROD  05/28/15 11:49:32 ITSOSSI1
VMFSRV2760I SERVICE processing completed successfully

SERVICE ALL STATUS UM34590
VMFSRV1226I    PUT2PROD  06/08/15 09:56:24 ITSOSSI2
VMFSRV1226I    PUT2PROD  06/08/15 09:46:24 ITSOSSI1
VMFSRV2760I SERVICE processing completed successfully
```

Figure 3-3 Verifying z/VM required service

3.2.5 Prepare disk pools

IBM Cloud Manager with OpenStack for z Systems can be installed and run by using either ECKD or SCSI storage devices. Three disk pools are needed as noted in the following list:

- z/VM work disk pool: Needed disks to implement IBM Cloud Manager with OpenStack for z Systems embedded system. This area holds the product binary code and is required for controller and compute roles.
- OpenStack controller data disk: Needed disk pool to implement IBM Cloud Manager with OpenStack for System z Logical Volume Manager (xCAT LVM) for controller database. This area holds the OpenStack controller database, configuration files, captured images,

and all needed information for all OpenStack services that the controller provide. This pool is only required for OpenStack controller role, and is not needed for the OpenStack compute role.

- OpenStack Ephemeral disk pool: Needed disk pool to be used to deploy new virtual instances (called Ephemeral disk by OpenStack terminology).

z/VM work disk pool

Three ECKD (3338 cylinders each) or SCSI (4,806,720 blocks each) minidisks are necessary to install IBM Cloud Manager with OpenStack for z Systems as noted in the following list:

- One minidisk (per z/VM controller or compute node) to restore the IBM Cloud Manager with OpenStack for z Systems solution. This minidisk is defined as the `XCAT 101` minidisk, which becomes the `CMOAPP` volume.
- One minidisk to hold the packed copy of the `CMA4202.ECKD` or `CMA4202.FBA` file. This minidisk is defined as `MAINT630 102` minidisk.

> **Tip:** Checksum information for CMA4202.ECKD and CMA4202.FBA can be found at:
>
> `https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/W21ed5ba0f4a9_46f4_9626_24cbbb86fbb9/page/zVM%20Cloud%20Manager%20Appliance%20%28CMA%29%20deployment%20information?section=Checksums`

- One minidisk to hold the unpacked copy of the `CMA4202.ECKD` or `CMA4202.FBA` file. This minidisk is defined as the `MAINT630 103` minidisk.

To define those minidisks, be sure that there is enough available area defined at DIRMAINT EXTENT CONTROL.

ECKD z/VM work disk pool

In the ECKD environment, the scenario defined a group named `GRPSYS that` holds two 3390-27 DASDs (CM6002 and CM6003). Use the **`DIRM DASD QUERY`** command to verify your configuration as shown in Figure 3-4.

```
DIRM DASD QUERY GROUP GRPSYS
 DVHDSD3561I GROUP=GRPSYS ALLOCATE=(LINEAR) REGIONS=2
 DVHDSD3562I GROUP=GRPSYS REGIONS=CM6002   CM6003

DIRM DASD QUERY VOLUME CM6002
 DVHDSD3565I VOLUME DEV-TYPE    SIZE        END
 DVHDSD3566I CM6002 3390-27       30051       30050

DIRM DASD QUERY VOLUME CM6003
 DVHDSD3565I VOLUME DEV-TYPE    SIZE        END
 DVHDSD3566I CM6003 3390-27       30051       30050
```

Figure 3-4 GRPSYS EXTENT CONTROL disk pool for ECKD z/VM work disk

SCSI z/VM work disk pool

To install IBM Cloud Manager with OpenStack for z Systems in SCSI LUNs, all LUNs reserved for the three disk pools mentioned previously are defined as `FBA DASD` (EDEVICE). They must be online in your z/VM environment and attached to the system.

Figure 3-5 shows details about EDEVICE 3000 as an example of the 10 GB LUNs (20971520 blocks) that are used in this scenario.

```
QUERY EDEVICE 3000 DETAILS
EDEV 3000 TYPE FBA ATTRIBUTES 2145
  VENDOR: IBM PRODUCT: 2145 REVISION: 0000
  BLOCKSIZE: 512 NUMBER OF BLOCKS: 20971520
  PATHS:
    FCP_DEV: B600 WWPN: 500507680130BB91 LUN: 0000000000000000 PREF
      CONNECTION TYPE: SWITCHED STATUS: ONLINE
    FCP_DEV: B700 WWPN: 500507680120BB91 LUN: 0000000000000000 PREF
      CONNECTION TYPE: SWITCHED STATUS: ONLINE

QUERY 3000
DASD 3000 CP SYSTEM CM3000
```

Figure 3-5 3000 EDEVICE 10 GB LUN

For more information about how to define and work with EDEVICES, see *Fibre Channel Protocol for Linux and z/VM on IBM System z,* SG-247266.

In this scenario, for the SCSI environment, a group named `GRPSYS` is defined and holds one 9336-10G FBA DASD (CM3000). Use the **`DIRM DASD QUERY`** command to verify your configuration as shown in Figure 3-6.

```
DIRM DASD QUERY GROUP GRPSYS
DVHDSD3561I GROUP=GRPSYS ALLOCATE=(LINEAR) REGIONS=1
 DVHDSD3562I GROUP=GRPSYS REGIONS=CM3000
 DVHREQ2289I Your DASD request for MAINT at * has completed; with RC = 0.

DIRM DASD QUERY VOLUME CM3000
DVHDSD3565I VOLUME DEV-TYPE    SIZE        END
 DVHDSD3566I CM3000 9336-10G   20971520   20971519
 DVHREQ2289I Your DASD request for MAINT at * has completed; with RC = 0.
```

Figure 3-6 GRPSYS EXTENT CONTROL disk pool for z/VM work disk

Note: `DEV-TYPE 9336-10G` (20971520 blocks) is a non-default configuration and was manually added at `DASDType` inside the `DIRMAINT EXTENT CONTROL` configuration file.

OpenStack controller data disk

Consider using at least 50 GB for this disk pool for the solution installation and setup. IBM Cloud Manager with OpenStack for z Systems creates an LVM in this area for OpenStack controller database use. This logical volume holds product database, customization, captured images, and instances snapshots. You can increase this area after installation just by adding new volids in the DNSSICMO (`cmo_data_disk = new_volid`) and restarting IBM Cloud Manager with OpenStack for z Systems.

Important: Do not manually include volume IDs in the DIRMAINT EXTENT CONTROL for the OpenStack controller disk pool. This action will cause the LVM setup to fail during the OpenStack controller initialization process. The volume IDs must be listed with IBM Cloud Manager with OpenStack for z Systems in the DMSSICMO copy file. From that point, the volume IDs are automatically added to DIRMAINT EXTENT CONTROL and configured for OpenStack controller use when IBM Cloud Manager with OpenStack for z Systems is started.

ECKD controller data disk

For ECKD controller data disks, CM9180, CM9181, CM9182, CM9183, CM9184, and CM9185 were reserved as 3390-9 DASD volumes (see Table 3-3 on page 41).

SCSI controller data disk

For SCSI controller data disks, CM3001, CM3002, and CM3003, which are 10 GB SCSI LUNs, were reserved as (EDEVICE) volumes (see Table 3-4 on page 42).

Ephemeral disk pool

IBM Cloud Manager with OpenStack for z Systems uses the ephemeral disk pool to deploy new instances. The required size of the ephemeral disk pool depends on the number of instances you plan to deploy in your environment. The volumes belonging to this pool must be manually added to DIRMAINT EXTENT CONTROL to be available to IBM Cloud Manager with OpenStack for z Systems. The ephemeral disk pool can be increased dynamically as your environment grows by manually adding more volumes to DIRMAINT EXTENT CONTROL.

ECKD Ephemeral disk pool

Figure 3-7 shows that in the example environment, a DIRMAINT group named GRPICM was created that contains ten 3390-27 volumes.

```
DIRM DASD QUERY GROUP GRPICM
 DVHDSD3561I GROUP=GRPICM ALLOCATE=(LINEAR) REGIONS=10
 DVHDSD3562I GROUP=GRPICM REGIONS=CM6006   CM6007   CM6008   CM6009
 DVHDSD3562I CM600A   CM600B   CM600C   CM600D   CM600E   CM600F

DIRM DASD QUERY VOLUME CM6006
 DVHDSD3565I VOLUME DEV-TYPE    SIZE        END
 DVHDSD3566I CM6006 3390-27       30051      30050
 DVHREQ2289I Your DASD request for MAINT at * has completed; with RC = 0
```

Figure 3-7 IBM Cloud Manager with OpenStack for z Systems ephemeral disk pool

SCSI Ephemeral disk pool

Figure 3-8 shows that in the example SCSI environment, a DIRMAINT group was created named GRPICM that contains eleven 9336-10G volumes.

```
DIRM DASD QUERY GROUP GRPICM
DVHDSD3561I GROUP=GRPICM ALLOCATE=(LINEAR) REGIONS=11
 DVHDSD3562I GROUP=GRPICM REGIONS=CM3004   CM3005   CM3006   CM3007
 DVHDSD3562I CM3008   CM3009   CM300A   CM300B   CM300C   CM300D   CM300E

DIRM DASD QUERY VOLUME CM3004
 DVHDSD3565I VOLUME DEV-TYPE    SIZE        END
 DVHDSD3566I CM3004 9336-10G   20971520   20971519
```

Figure 3-8 IBM Cloud Manager with OpenStack for z Systems ephemeral disk pool

3.2.6 Verify SMAPI configuration

Managing many deployed instances requires a thorough understanding of z/VM concepts and the knowledge and skills to run a complex set of commands. The SMAPIs simplify managing many virtual images running under a z/VM environment. A standard, platform-independent client interface reduces the number of z/VM-specific programming skills that are required and provides a basic set of interfaces to allow the products to interact remotely with z/VM.

SMAPI must be running and able to receive remote calls. At a minimum, the z/VM standard SMAPI customization is needed for IBM Cloud Manager with OpenStack for z Systems. This implementation assumes that SMAPI is enabled in your z/VM environment. For more information about SMAPI, see *Systems Management Application Programming,* SG24-6234.

You should include XAUTOLOG VSMGUARD in your z/VM start procedure. VSMGUARD is a worker sever that is responsible for starting other worker servers. A worker server processes API function requests. Validate that the worker servers are running as highlighted in Figure 3-9.

```
QUERY NAMES
SLES11B  - DSC , MAINT630 - DSC , ZHCP     - DSC , XCAT     - DSC
VSMEVSRV - DSC , VSMREQIU - DSC , VSMREQI6 - DSC , VSMREQIN - DSC
DTCSMAPI - DSC , PERSMAPI - DSC , VSMWORK3 - DSC , VSMWORK2 - DSC
VSMWORK1 - DSC , VSMGUARD - DSC , ITSO0002 - DSC , LOHCOST  - DSC
FTPSERVE - DSC , TCPIP    - DSC , DIRMAINT - DSC , DTCVSW2  - DSC
DTCVSW1  - DSC , VMSERVP  - DSC , VMSERVR  - DSC , VMSERVU  - DSC
VMSERVS  - DSC , OPERSYMP - DSC , DISKACNT - DSC , EREP     - DSC
OPERATOR - DSC , ITSO0004 - DSC , MAINT    -L0004
VSM     - TCPIP
```

Figure 3-9 Worker servers - z/VM virtual machines

The web material associated with *The Virtualization Cookbook for IBM z/VM 6.3, RHEL 6.4, and SLES 11 SP3,* SG24-8147 has a procedure to test SMAPI using a CMS REXX EXEC called CALLSM1. It is included in the SG248147.tgz file that can be download from:

http://www.vm.ibm.com/devpages/mikemac/SG248147.tgz

3.2.7 Configure directory manager

The use of a Directory Manager product is required to remotely provision virtual images using IBM Cloud Manager with OpenStack for z Systems. The example environment uses IBM Directory Maintenance for z/VM (DIRMAINT).

This implementation also assumes that DIRMAINT is enabled in your z/VM environment. If you need more information about implementation, see *zVM V6.3 Getting Started with Linux on System z*, SC24-6194, for step-by-step examples.

For IBM Cloud Manager with OpenStack for z Systems, DIRMAINT must be enabled to receive TCP and UDP asynchronous notifications for directory updates enabled by SMAPI. The DATAMOVE server is also required. The DATAMOVE server is responsible for manipulating minidisks (format, copying, and cleaning) on behalf of the DIRMAINT server.

Example 3-1 shows a DIRMAINT configuration file with these updates.

Example 3-1 DIRMAINT sample configuration

```
 ALLOW_ASUSER_NOPASS_FROM= VSMGUARD *
 ALLOW_ASUSER_NOPASS_FROM= VSMWORK1 *
...
 ASYNCHRONOUS_UPDATE_NOTIFICATION_EXIT.TCP= DVHXNE EXEC
 ASYNCHRONOUS_UPDATE_NOTIFICATION_EXIT.UDP= DVHXNE EXEC
 DISK_CLEANUP= YES
 ONLINE= IMMED
 RUNMODE= OPERATIONAL
 DATAMOVE_MACHINE= DATAMOVE * *
```

A z/VM SSI Cluster requires one DATAMOVE virtual server per z/VM node. The z/VM SSI installation creates the CONFIGSS DATADVH DIRMAINT configuration file by default with these definitions:

```
DATAMOVE_MACHINE= DATAMOVE ITSOSSI1 *
DATAMOVE_MACHINE= DATAMOV2 ITSOSSI2 *
```

For more information about the definitions, see *z/VM 6.3 Directory Maintenance Facility Tailoring and Administration Guide*, SC24-6190.

External Security Manager

A z/VM external security manager, such as IBM RACF® for z/VM, can be used to enhance the security and integrity of your system in the following ways:

- Helping your z/VM installation implement its security policy
- Identifying and authenticating each z/VM user
- Controlling each user's access to sensitive data
- Logging and reporting events that are relevant to the system's security

> **Note:** IBM Cloud Manager with OpenStack with z Systems can reside together with RACF for z/VM. However, this configuration requires additional tasks that are not covered in this book. For more information about RACF for z/VM, see *The Virtualization Cookbook for IBM z/VM 6.3, RHEL 6.4, and SLES 11 SP3,* SG24-8147.

3.2.8 Configure OSA EQID for z/VM SSI

> **Note:** Skip this section if you do not have z/VM Single System Image cluster.

If you are planning to implement IBM Cloud Manager with OpenStack for z Systems in a z/VM Single System Image environment, you can use the Live Guest Relocation feature from the cloud portal. For Live Guest Relocation to work through the cloud portal, add RDEV statements with the EQID option for the OSA devices in the SYSTEM CONFIG file.

Real device mapping provides a means of identifying a device by an equivalency ID (EQID). This mapping is used to ensure that the virtual machines relocated by Live Guest Relocation continue to use the same or equivalent devices after a relocation. VMLAN MACPREFIX and USERPREFIX also must be set up according to your infrastructure. Example 3-2 shows the example VMLAN definitions.

Example 3-2 EQID and VMLAN SYSTEM CONFIG definition for z/VM SSI implementation

```
ITSOSSI1: BEGIN
RDEV 2D00-2D0F EQID OSASET1 TYPE OSA
RDEV 2D20-2D2F EQID OSASET2 TYPE OSA
RDEV 2D40-2D4F EQID OSASET3 TYPE OSA
RDEV 2D60-2D6F EQID OSASET4 TYPE OSA
...
VMLAN MACPREFIX 02001A USERPREFIX 020000
ITSOSSI1: END

ITSOSSI2: BEGIN
RDEV 2D00-2D0F EQID OSASET1 TYPE OSA
RDEV 2D20-2D2F EQID OSASET2 TYPE OSA
RDEV 2D40-2D4F EQID OSASET3 TYPE OSA
RDEV 2D60-2D6F EQID OSASET4 TYPE OSA
...
VMLAN MACPREFIX 02001B USERPREFIX 020000
ITSOSSI2: END
```

Figure 3-10 shows how to verify your configuration by using the QUERY command.

```
(at ITSOSSI1)
QUERY EQID FOR 2D00
2D00: EQID = OSASET1

QUERY EQID FOR 2D20
2D20: EQID = OSASET2

QUERY EQID FOR 2D40
2D40: EQID = OSASET3

QUERY EQID FOR 2D60
2D60: EQID = OSASET4

QUERY VMLAN
VMLAN MAC address assignment:
  MACADDR Prefix: 02001A USER Prefix: 020000

(at ITSOSSI2)
QUERY EQID FOR 2D00
2D00: EQID = OSASET1

QUERY EQID FOR 2D20
2D20: EQID = OSASET2

QUERY EQID FOR 2D40
2D40: EQID = OSASET3

QUERY EQID FOR 2D60
2D60: EQID = OSASET4

QUERY VMLAN
VMLAN maintenance level:
  MACADDR Prefix: 02001B USER Prefix: 020000
```

Figure 3-10 EQID and VMLAN environment verification for z/VM SSI implementation

3.3 Installing IBM Cloud Manager with OpenStack for z Systems

The following sections provide instructions about product installation and initial setup for ECKD or SCSI devices.

3.3.1 Download IBM Cloud Manager with OpenStack for z Systems image file

IBM Cloud Manager with OpenStack for z Systems must be downloaded from IBM Fix Central to a workstation that will be restored on an xCAT 0101 minidisk.

You must be a registered user of IBM Fix Central to download the product. If you do not have an ID, click **Register** in the upper right corner of the following web page and follow the registration instructions:

`http://www.ibm.com/support/fixcentral`

As a registered user at IBM Fix Central, complete these steps to download the product:

1. In the IBM Fix Central page, click the Select Product tab
2. Select **Other Software** from the **Product Group** menu.
3. Enter `IBM Cloud Manager with OpenStack` in the **Product Selector** menu.
4. Select **4.2.0.2** in the **Installed Version** menu.
5. Select **z/VM** in the **Platform** menu.
6. Click **Continue**, and again on the next page. Depending on the type of disk used, select the IMAGE file for your installation using one of the following options:
   - 4.2.0.2-IBM-CMWO-zVM_ECKD_DASD to install the product on ECKD
   - 4.2.0.2-IBM-CMWO-zVM_FBA_DASD to install the product on SCSI

   Then select:
   - 4.2.0.2-IBM-CMWO-zVM_CMA_Installer
7. Click **Continue**.
8. Select a download method and download the files.

The following files will be downloaded:

- CMA4202.ECKD (2 GB): Contains IBM Cloud Manager with OpenStack for z Systems image for ECKD.
- CMA4202.FBA (2 GB): Contains IBM Cloud Manager with OpenStack for z Systems image for FBA.
- cmwo420_cma_install.tar (5 GB): Contains IBM Cloud Manager with OpenStack for z Systems Deployer installer files.

Instructions for how to use those files are covered in the following sections.

3.3.2 Create z/VM minidisks for installation

As described in “z/VM work disk pool” on page 46, three ECKD (3338 cylinders each) or SCSI (4,806,720 blocks each) are necessary to install the IBM Cloud Manager with OpenStack for z Systems solution.

ECKD minidisks

Use the DIRMAINT AMD command to add an XCAT-1 (a 3338 cylinders minidisk) and use `101` as the virtual address as shown in Figure 3-11.

```
DIRM FOR XCAT-1 AMD 101 X AUTOG 3338 GRPSYS MR PW ALL WRITE MULTIPLE
 DVHREQ2289I Your AMDISK request for XCAT-1 at * has completed; with RC
 DVHREQ2289I = 0.
```

Figure 3-11 ECKD - XCAT-1 101 minidisk definition

Add two 3338 cylinders minidisks to MAINT630 using `102` and `103` as the virtual addresses. See Figure 3-12 for the commands used.

```
DIRM FOR MAINT630 AMD 102 X AUTOG 3338 GRPSYS MR PW ALL WRITE MULTIPLE
 DVHREQ2289I Your AMDISK request for MAINT630 at * has completed; with RC
 DVHREQ2289I = 0.

DIRM FOR MAINT630 AMD 103 X AUTOG 3338 GRPSYS MR PW ALL WRITE MULTIPLE
 DVHREQ2289I Your AMDISK request for MAINT630 at * has completed; with RC
 DVHREQ2289I = 0.
```

Figure 3-12 ECKD - MAINT630 102 and 103 minidisks definitions

Note: The DIRMAINT requests must complete with a reason code (RC) of 0, as in Figure 3-12.

SCSI minidisks

Use the DIRMAINT AMD command to add to the XCAT a 4806720 blocks minidisk using `101` as the virtual address as shown in Figure 3-13.

```
DIRM FOR XCAT AMD 101 X AUTOG 4806720 GRPSYS MW PW ALL WRITE MULTIPLE
 DVHREQ2289I Your AMDISK request for XCAT at * has completed; with RC =
 DVHREQ2289I 0.
```

Figure 3-13 SCSI - XCAT 101 minidisk definition

Add two 4806720 block minidisks to MAINT630 `102` and *103* virtual addresses as shown in Figure 3-14.

```
DIRM FOR MAINT630 AMD 102 X AUTOG 4806720 GRPSYS MW PW ALL WRITE MULTIPLE
 DVHREQ2289I Your AMDISK request for MAINT630 at * has completed; with RC
 DVHREQ2289I = 0.

DIRM FOR MAINT630 AMD 103 X AUTOG 4806720 GRPSYS MW PW ALL WRITE MULTIPLE
 DVHREQ2289I Your AMDISK request for MAINT630 at * has completed; with RC
 DVHREQ2289I = 0.
```

Figure 3-14 SCSI - MAINT630 102 and 103 minidisks definitions

Note: The DIRMAINT requests must complete with a reason code (RC) of 0, as in Figure 3-14.

3.3.3 Upload IBM Cloud Manager with OpenStack for z Systems image file

Log on to MAINT630 and format the `102` and `103` minidisks that will upload and unpack the ECKD file (`CMA4202.ECKD`) or SCSI installation, use the `CMA4202.FBA` file instead. See Figure 3-15 for the commands used.

```
QUERY V 102-103
 DASD 0102 3390 CM6002 R/W       3338 CYL ON DASD  6002 SUBCHANNEL = 0044
 DASD 0103 3390 CM6002 R/W       3338 CYL ON DASD  6002 SUBCHANNEL = 0045

FORMAT 102 T
DMSFOR733I Formatting disk T
DMSFOR732I 3338 cylinders formatted on T(102)

FORMAT 103 U
DMSFOR733I Formatting disk U
DMSFOR732I 3338 cylinders formatted on U(103)
```

Figure 3-15 Preparing MAINT630 102 and 103 minidisks

LINK TCPMAINT the 592 minidisk (to be able to use z/VM FTP client) and upload the file to the MAINT630 102 minidisk. Be sure to use binary transfer and fixed 1024 block. Otherwise the upload will fail.

Tip: Checksum information for CMA4202.ECKD and CMA4202.FBA can be found at:

https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/W21ed5ba0f4a9_46f4_9626_24cbbb86fbb9/page/zVM%20Cloud%20Manager%20Appliance%20%28CMA%29%20deployment%20information?section=Checksums

Figure 3-16 shows an FTP example for the ECKD file.

```
VMLINK TCPMAINT 592
DMSVML2060I TCPMAINT 592 linked as 0120 file mode Z
FTP 192.168.60.63
VM TCP/IP FTP Level 630
Connecting to 192.168.60.63, port 21
USER (identify yourself to the host):
sousa
>>>USER sousa
331 Password required for sousa
Password:
>>>PASS ********
230 Logged on
Command:
BINARY
>>>TYPE i
200 Type set to I
Command:
LOCSITE FIX 1024
Command:
GET CMA4202.ECKD CMA4202.ECKD.T
...
1975762944 bytes transferred.
226 Transfer OK
```

Figure 3-16 CMA4202 upload

3.3.4 Unpack uploaded CMA4202 image file

The z/VM virtual machine that runs the IBM Cloud Manager with OpenStack for z Systems solution is xCAT, and it uses the XCAT 101 minidisk that was defined previously. Use MAINT630 to restore the IBM Cloud Manager with OpenStack for z Systems from the image fileto the XCAT 101 minidisk.

Use the COPYFILE command with the UNPACK function to restore the file to its original unpacked form. Figure 3-17 shows an example for an ECKD file. Note that the LRECL has been changed from `1024` to `49284` after the copy. For the SCSI installation, this scenario used the `CMA4202.FBA` file, with an LRECL of `65535`.

```
LISTFILE * * T (FORMAT
FILENAME FILETYPE FM FORMAT LRECL
CMA4202  ECKD     T1 F       1024

COPYFILE CMA4202 ECKD T CMA4202 ECKD U (UNPACK OLDDATE

LISTFILE * * U (FORMAT
FILENAME FILETYPE FM FORMAT LRECL
CMA4202  ECKD     U1 V      49284
```

Figure 3-17 Unpacking the CMA4202.ECKD file

3.3.5 Restore unpacked CMA4202 image file

Use the DDRREST utility to restore the unpacked CMA4202 image file to the XCAT 101 minidisk. Access the MAINT630 193 minidisk and restore the unpacked CMA4202 image file to the XCAT 101 minidisk.

Important: You need to detach the XCAT 101 minidisk when the restore ends.

Figure 3-18 shows a restore example for the ECKD file (use the `CMA4202.FBA` file for the SCSI environment).

```
LINK XCAT 101 101 MR
ACCESS 193 T
ACCESS 103 U
DDRREST 101 CMA4202 ECKD U
z/VM DASD DUMP/RESTORE PROGRAM
HCPDDR698I DATA DUMPED FROM CMOAPP TO BE RESTORED
HCPDDR697I NO VOL1 LABEL FOUND
RESTORING CMOAPP
DATA DUMPED   03/05/15 AT 08.16.27  GMT FROM CMOAPP RESTORED
INPUT CYLINDER EXTENTS      OUTPUT CYLINDER EXTENTS
      START       STOP         START        STOP
          0       3337             0        3337
END OF RESTORE
 BYTES RESTORED  2467590380

END OF JOB

DET 101
DASD 0101 DETACHED
```

Figure 3-18 Restoring IBM Cloud Manager with OpenStack for z Systems to z/VM XCAT 101 minidisk

3.3.6 Increase xCAT z/VM user ID memory

The original z/VM xCAT implementation loads xCAT from the 100 minidisk. IBM Cloud Manager with OpenStack for z Systems will not use this original implementation. After being configured, xCAT will automatically IPL from the new 101 minidisk that was restored from the CMA4202 file. This file contains the IBM Cloud Manager with OpenStack for z Systems solution. This scenario ran xCAT with an initial storage value of 8 GB of virtual storage. Use the MAINT z/VM user ID to update the z/VM XCAT directory definition from 2 GB to 8 GB as shown in Figure 3-19.

```
DIRM FOR XCAT MAXSTOR 8G
 DVHREQ2289I Your MAXSTORE request for XCAT at * has completed; with RC =
 DVHREQ2289I 0.

DIRM FOR XCAT STORAGE 8G
 DVHREQ2289I Your STORAGE request for XCAT at * has completed; with RC
 DVHREQ2289I = 0.
```

Figure 3-19 Increasing z/VM XCAT user ID to 8 GB

3.3.7 Customize the DMSSICNF configuration file

Log on to MAINT630 to have access to the DMSSICNF configuration file. This configuration file is one of the two files that are used by IBM Cloud Manager with OpenStack for z Systems to build its own configuration files during its start procedure. The other configuration file is DMSSICNF.

For more information about DMOSSICNF configuration options, see *Enabling z/VM for OpenStack (Support for OpenStack Juno Release),* SC24-6249.

As shown in Figure 3-20, this scenario used the LOCALMOD command to update the DMSSICNF configuration file. Enter `1` when prompted. After the updates, enter FILE to save the changes and finish the XEDIT utility. The LOCALMOD command shows that it completed successfully.

```
LOCALMOD CMS DMSSICNF $COPY
RDR FILE 0030 SENT FROM MAINT630 CON WAS 0030 RECS 0005 CPY  001 T NOHOLD
NOKEEP
VMFUTL2767I Reading VMFINS DEFAULTS B for additional options
VMFLMD2760I LOCALMOD processing started
VMFSET2760I VMFSETUP processing started for SERVP2P CMS
VMFSET2204I Linking 6VMLEN20 49E as 49E with link mode MR
VMFSET2204I Linking PMAINT 550 as 550 with link mode MR
VMFUTL2205I Minidisk|Directory Assignments:
            String    Mode  Stat  Vdev  Label/Directory
VMFUTL2205I LOCALMOD  E      R/W  3C4   MNT3C4
...
VMFLMD1301R Local modification L0002 is being created for update part DMSSICNF
            $COPY.
            Enter (0) to quit; (1) to continue.
            If you choose to continue, you will be put into an XEDIT session.
            Make your changes and then enter FILE.
1
DMSXUP178I Applying DMSSICNF 065463DS J1
DMSXUP178I Applying DMSSICNF 065667DS J1
DMSXUP180W Missing PTF file DMSSICNF UPL0002 A
VMFEXU2760I VMFEXUPD processing started
DMSUPD178I Updating DMSSICNF $COPY A1
...
VMFEXU2760I VMFEXUPD processing completed successfully
VMFLMD2760I LOCALMOD processing completed successfully
```

Figure 3-20 Updating DMSSICNF configuration file using LOCALMOD

The DMSSICNF config file used in this scenario's configuration is provided in Example 3-3.

Example 3-3 DMSSICNF config file

```
/* XCAT server defaults                                             */
XCAT_User     = "XCAT"                /* xCAT z/VM user ID         */
XCAT_Addr     = "10.10.10.10"         /* XCAT IP Address           */
XCAT_Host     = "xcat"                /* xCAT hostname             */
XCAT_Domain   = ".itso.ihost.com"     /* xCAT domain name          */
XCAT_vswitch  = "XCATVSW1"            /* xCAT Vswitch name         */
XCAT_OSAdev   = "2D63"                /* OSA address for xCAT      */
XCAT_zvmsysid = "itsossi1"            /* xCAT z/VM system id       */
```

```
XCAT_notify   = "OPERATOR"             /* Notify when xCAT started  */
XCAT_gateway  = "10.10.10.1"           /* Network gateway IP addr.  */
XCAT_netmask  = "255.255.255.0"        /* Default network mask      */
XCAT_vlan     = "NONE"
XCAT_iso = ""

XCAT_MN_Addr    = "192.168.60.101"     /* xCAT mgmt node IP address */
XCAT_MN_vswitch = "XCATVSW2"           /* xCAT MN Vswitch name      */
XCAT_MN_OSAdev  = "2D03"               /* OSA address for xCAT MN   */
XCAT_MN_gateway = "192.168.60.1"       /* Network gateway IP addr.  */
XCAT_MN_Mask    = "255.255.255.0"      /* Netmask for xCAT MN       */
XCAT_MN_vlan    = "NONE"

XCAT_MN_admin = "mnadmin"              /* MN administrator userid   */
XCAT_MN_pw    = "passwd"               /* MN admin password         */
                                       /* (if NOLOG, userid cannot  */
                                       /*  ssh into XCAT MN)        */

/* ZHCP server defaults                                             */
ZHCP_User    = "ZHCP"                  /* zhcp z/VM user ID         */
ZHCP_Addr    = "10.10.10.20"           /* zhcp IP ADDRESS           */
ZHCP_Host    = "zhcp1"                 /* zhcp hostname             */
ZHCP_Domain  = ".itso.ihost.com"       /* zhcp domain name          */
ZHCP_gateway  = "10.10.10.1"           /* Network gateway IP addr.  */
ZHCP_netmask  = "255.255.255.0"        /* Default network mask      */
ZHCP_vswitch = "XCATVSW1"              /* zhcp Vswitch name         */
ZHCP_OSAdev  = "2D63"                  /* OSA address for zhcp      */
ZHCP_vlan    = "NONE"
```

The meaning for the configuration options are as follows:

XCAT_Domain OpenStack controller domain name

XCAT_OSAdev XCATVSW1 (for xCAT internal network) real OSA device address

XCAT_zvmsysid z/VM SYSID where OpenStack controller will run (lowercase)

XCAT_MN_Addr OpenStack controller (xCAT Management Node) IP address

XCAT_MN_OSAdev XCATVSW2 (for management network) real OSA device address

XCAT_MN_gateway OpenStack controller default gateway

XCAT_MN_netmask OpenStack controller default netmask

XCAT_MN_pw xCAT Management Node (`mnadmin`) default password

ZHCP_Host ZHCP host name

ZHCP_Domain ZHCP domain name

ZHCP_OSAdev ZHCP (internal network) real OSA device address

> **Note:** XCAT_MN_vswitch is the z/VM virtual switch for the management network. The default value is `XCATVSW2`.

3.3.8 Customize the DMSSICMO COPY configuration file

IBM Cloud Manager with OpenStack for z Systems configures its services based on the properties in the DMSSICMO configuration file. A script reads the data in this file and updates various OpenStack configuration files when the server starts.

For more information about DMSSICMO configuration options, see *Enabling z/VM for OpenStack (Support for OpenStack Juno Release),* SC24-6249.

The process used for the DMSSICMO configuration file is the same as for the previous DMSSICNF configuration file (see Figure 3-20 on page 58). The LOCALMOD command updates the DMSSICMO configuration file, and replies `1` when prompted. After the updates, FILE is entered to save the changes and finish the `XEDIT` utility. The LOCALMOD command should complete successfully.

The DMSSICMO configuration file, used in this scenario's ECKD configuration is provided in Example 3-4.

Example 3-4 DMSSICMO config file for ECKD OpenStack controller role

```
/* CMO User Configurable Settings                                    */
cmo_admin_password                  = "passwd"
cmo_data_disk         = "CM9180 CM9181 CM9182 CM9183 CM9184 CM9185"
openstack_system_role               = "controller"
openstack_controller_address        = ""
openstack_zvm_diskpool              = "ECKD:GRPICM"
openstack_instance_name_template    = "itso%04x"
openstack_zvm_fcp_list              = "NONE"
openstack_zvm_timeout               = "300"
openstack_zvm_scsi_pool             = "NONE"
openstack_zvm_zhcp_fcp_list         = "NONE"
openstack_san_ip                    = "NONE"
openstack_san_private_key           = "NONE"
openstack_storwize_svc_volpool_name  = "NONE"
openstack_storwize_svc_vol_iogrp    = "NONE"
openstack_zvm_image_default_password = "password"
openstack_xcat_mgt_ip               = ""
openstack_xcat_mgt_mask             = ""
openstack_zvm_xcat_master           = "xcat"
openstack_zvm_vmrelocate_force      = "NONE"
```

The meaning for the configuration options are as follows:

cmo_admin_password	OpenStack controller default password
cmo_data_disk	volids for OpenStack controller data disk
openstack_system_role	OpenStack controller role
openstack_zvm_diskpool	ECKD or FBA:DIRMAINT region for OpenStack Ephemeral disk pool
openstack_instance_name_template	z/VM user ID instance name template
openstack_zvm_image_defaults_password	Default root password for deployed instances
openstack_zvm_xcat_master	xCAT Management Node name

Note: For the SCSI environment, the value used for configuration option was `openstack_zvm_diskpool = “FBA:GRPICM”`.

3.3.9 Put changes on production

After updating the DMSSICNF and DMSSICMO configuration files using the LOCALMOD command, apply the changes to the CMS by using the SERVICE procedure (Figure 3-21).

```
SERVICE CMS BUILD
VMFSET2760I VMFSETUP processing completed successfully
VMFSRV1233I The following products have been serviced.
VMFSRV1233I CMS
VMFSRV2760I SERVICE processing completed successfully
```

Figure 3-21 Applying DMSSICNF and DMSSICMO file changes to CMS

Then, copy the changes to the production disk with the **PUT2PROD** command as shown in Figure 3-22.

```
PUT2PROD
VMFP2P1217I There are no products to put into production.
VMFP2P2760I PUT2PROD processing completed successfully
```

Figure 3-22 Copying DMSSICNF and DMSSICMO files to z/VM production disks

Log off MAINT630 and follow the next sections to start IBM Cloud Manager with OpenStack for z Systems.

3.3.10 Start and validate IBM Cloud Manager with OpenStack for z Systems

As a part of the start procedure, the IBM Cloud Manager with OpenStack for z Systems creates XCATVSW1 and XCATVSW2 z/VM virtual switches. This is done automatically based on the information included in the configuration files in the previous section.

Before starting xCAT for the first time with the new configuration, the z/VM virtual switches (XCATVS1 and XCATVSW2) will have to be detached using the DET VSWITCH command. The suggested method to start IBM Cloud Manager with OpenStack for z Systems for the first time after configuration is shown in Figure 3-23.

```
DET VSWITCH XCATVSW1
VSWITCH SYSTEM XCATVSW1 is destroyed

DET VSWITCH XCATVSW2
VSWITCH SYSTEM XCATVSW2 is destroyed

FORCE VSMGUARD
USER DSC   LOGOFF AS  VSMGUARD USERS = 27    FORCED BY MAINT

XAUTOLOG VSMGUARD

SET OBSERVER XCAT *
```

Figure 3-23 Starting IBM Cloud Manager with OpenStack for z Systems for the first time

The OBSERVER utility permits MAINT to see all xCAT console messages. As xCAT loads its 101 minidisk containing the IBM Cloud Manager with OpenStack solution, you can see all of the controller node console messages.

One of the tasks that the controller node performs as part of its initialization process is the setup of the controller database. The controller node also takes the following actions:

- Read volids listed in DMSSICMO configuration file (cmo_data_disk) and adds these volids to the DIRMAINT EXTENT CONTROL file (by ZHCP)
- Create an XCAT1 region in the DIRMAINT EXTENT CONTROL file with these volids (by ZHCP)
- LINK the new minidisk, vary on, format, and create XCAT LVM at `/dev/mapper/cmo-data`
- Mount LVM at `/data` and create the controller database

> **Note:** The `mnadmin` home directory is also mounted into `/data`.

The IBM Cloud Manager with OpenStack for z Systems initialization process is finished when the 'CMO APPLIANCE RUNNING' message is displayed. After this message appears, the observer XCAT can be disabled as shown in Figure 3-24.

```
'CMO APPLIANCE RUNNING'

SET OBSERVER XCAT OFF
```

Figure 3-24 IBM Cloud with OpenStack for z Systems initialization complete message

After initialization, open an SSH connection to your OpenStack controller (xCAT Management Node IP address) using the `mnadmin` user and password in the configuration files.

Verify the controller network connection and XCAT LVM mount point (/data) (Figure 3-25).

```
[mnadmin@xcat ~] $ ifconfig
eth0      Link encap:Ethernet  HWaddr 02:00:1A:00:00:FC
          inet addr:10.10.10.10  Bcast:10.10.10.255  Mask:255.255.255.0
          inet6 addr: fe80::200:1a00:200:fc/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1492  Metric:1

eth1      Link encap:Ethernet  HWaddr 02:00:1A:00:00:FD
          inet addr:192.168.60.101  Bcast:192.168.60.255  Mask:255.255.255.0
          inet6 addr: fe80::1aff:fe00:fd/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1492  Metric:1

[mnadmin@xcat ~] $ df -h /dev/mapper/cmo-data
Filesystem           Size  Used Avail Use% Mounted on
/dev/mapper/cmo-data   30G  1.7G   27G   6% /data
```

Figure 3-25 OpenStack controller network and XCAT LVM

As the mnadmin user, you can issue OpenStack service commands. For example, you can check the status of the main OpenStack services as shown in Figure 3-26.

```
[mnadmin@xcat ~] $ sudo service openstack-nova-compute status
openstack-nova-compute (pid  4809) is running...
[mnadmin@xcat ~] $ sudo service openstack-cinder-volume status
openstack-cinder-volume (pid  4588) is running...
[mnadmin@xcat ~] $ sudo service neutron-server status
neutron (pid  4463) is running...
[mnadmin@xcat ~] $ sudo service openstack-keystone status
openstack-keystone (pid  2659) is running...
[mnadmin@xcat ~] $ sudo service qpidd status
qpidd (pid 3353) is running...
[mnadmin@xcat ~] $ sudo service openstack-glance-registry status
openstack-glance-registry (pid  4647) is running...
[mnadmin@xcat ~] $ sudo service openstack-iaasgateway status
openstack-iaasgateway (pid  4666) is running...
[mnadmin@xcat ~] $ sudo service openstack-heat-engine status
openstack-heat-engine (pid  4273) is running...

[mnadmin@xcat ~] $ source openrc
[mnadmin@xcat ~] $ nova service-list
+----+------------------+----------+----------+---------+-------+
| Id | Binary           | Host     | Zone     | Status  | State |
+----+------------------+----------+----------+---------+-------+
| 1  | nova-cert        | itsossi1 | internal | enabled | up    |
| 2  | nova-console     | itsossi1 | internal | enabled | up    |
| 3  | nova-conductor   | itsossi1 | internal | enabled | up    |
| 5  | nova-consoleauth | itsossi1 | internal | enabled | up    |
| 6  | nova-scheduler   | itsossi1 | internal | enabled | up    |
| 7  | nova-compute     | itsossi1 | nova     | enabled | up    |
+----+------------------+----------+----------+---------+-------+
```

Figure 3-26 OpenStack controller services

You can also validate IBM Cloud Manager with OpenStack for z Systems is running by connecting to the self-service portal as shown in Figure 3-27. The web interface for self-service portal is available at `https://IP:18443/cloud/web/login.html`, where IP is the XCAT_MN_Addr in the DMSSICNF file. You will login to the self-service portal as administrator. The User ID is admin and the password is the value of cmo_admin_password that you defined in the DMSSICMO file.

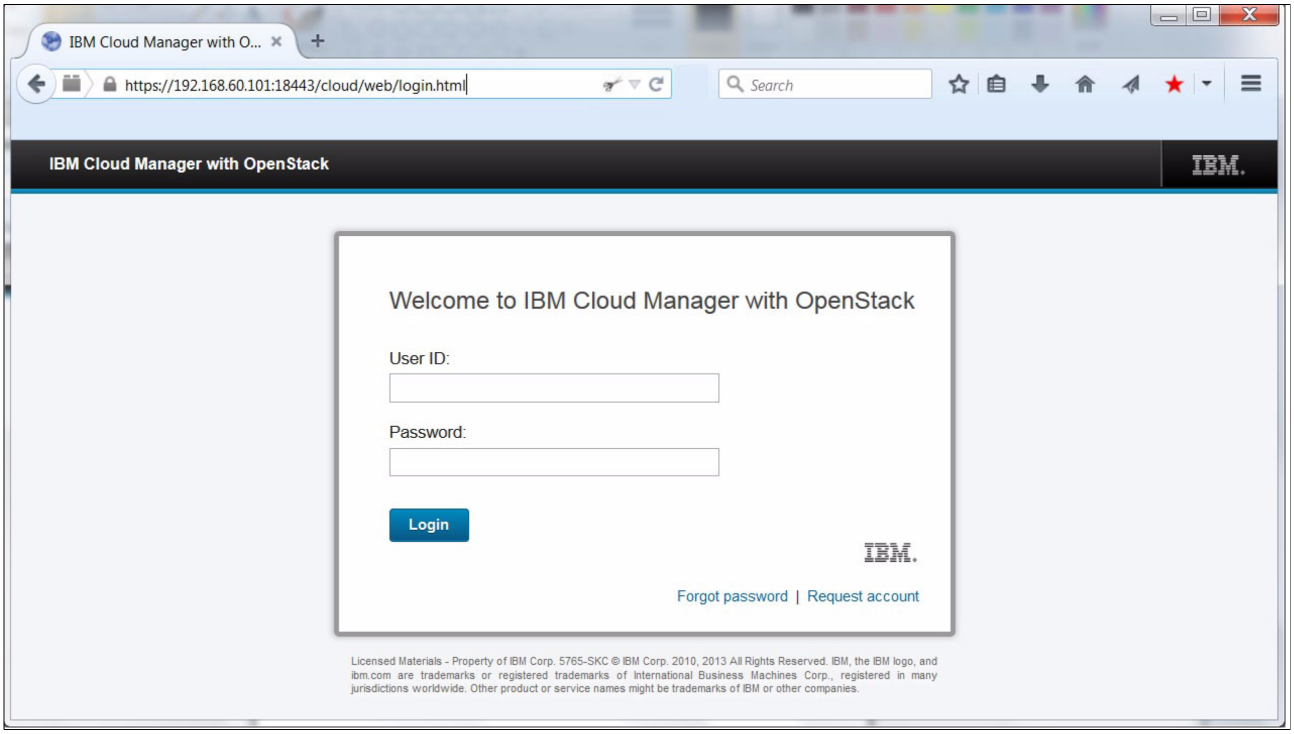


Figure 3-27 IBM Cloud Manager with OpenStack for z Systems self-service portal

3.3.11 Install IBM Cloud Manager with OpenStack Deployer

To complete your installation, you will need to install IBM Cloud Manager with OpenStack Deployer. Transfer the downloaded `cmwo420_cma_install.tar` file to the xCAT Management Node.

> **Tip:** Checksum information for cmwo420_cma_install.tar can be found at:
>
> `https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/W21ed5ba0f4a9_46f4_9626_24cbbb86fbb9/page/zVM%20Cloud%20Manager%20Appliance%20%28CMA%29%20deployment%20information?section=Checksums`

First of all, verify that you have 50 GB of available area derived from the disk pool at `/data` (mount point). You can use an sftp client tool, such as scp, to copy the IBM Cloud Manager with OpenStack Deployer file to the controller machine. In this scenario, the user name is `mnadmin` and the password is `passwd`. The installer file should be stored in the `mnadmin` home directory.

After the transferof the `cmwo420_cma_install.tar` file, use an ssh client to connect to the xCAT Manager Node as `mnadmin` user. Then the commands shown in Figure 3-28 are entered to complete the installation.

> **Note:** This procedure is the same for both ECKD and SCSI environments.

```
[mnadmin@xcat ~] $ pwd
/home/mnadmin
[mnadmin@xcat ~] $ ls -lah cmwo420_cma_install.tar
-rw-r--r-- 1 mnadmin root 4.9G May 05 17:38 cmwo420_cma_install.tar
[mnadmin@xcat ~] $ tar xvf cmwo420_cma_install.tar
cmwo420_zlinux_install.bin
cmwo420_zlinux_install_pkg_01.tar.gz
cmwo420_zlinux_install_pkg_02.tar.gz
cmwo420_zlinux_install_pkg_03.tar.gz
cmwo_4.2.lic
cmwo-install-sample.rsp
[mnadmin@xcat ~] $ sudo /usr/local/bin/cmoinstaller.sh -r
cmwo420_zlinux_install.bin -directory /home/mnadmin/
Temporary stop Heat services before install ICM
Stopping openstack-heat-api: [  OK  ]
Stopping openstack-heat-api-cfn: [  OK  ]
Stopping openstack-heat-api-cloudwatch: [  OK  ]
Stopping openstack-heat-engine: [  OK  ]
Warning or non fatal error found during installation, continue. Please check
IBM_Cloud_Manager_with_OpenStack_Install_06_02_2015_19_59_34.log for sure.
Enable CMA PRS installer cookbook
Uploading ibm-openstack-zvm-appliance-prs [0.1.0]
Uploaded 1 cookbook.
Start Heat services
Starting openstack-heat-api: [  OK  ]
Starting openstack-heat-api-cfn: [  OK  ]
Starting openstack-heat-api-cloudwatch: [  OK  ]
Starting openstack-heat-engine: [  OK  ]
```

Figure 3-28 IBM Cloud Manager with OpenStack Deployer installation

A warning or non fatal error message appears after the installation. Figure 3-29 shows the information found in the `IBM_Cloud_Manager_with_OpenStack_Install_(`*`your_date`*`).log` file that caused this warning message on the installation process. It points to another log file named `cmwo-installer`.

```
CheckShellExitCode
                          Status: ERROR
                          Additional Notes: ERROR - Script INSTALL-OS-UTILS
failed with a return code of '1'.  See log /tmp/cmwo-installer.log for more
information.
```

Figure 3-29 Error message

Figure 3-30 shows the warning message found in the `cmwo-installer.log`. This message indicates that the package named (`openstack-utils-2014.2.2-201502130059`) was already installed. In this scenario, the message can be ignored.

```
Preparing packages for installation...

CheckShellExitCode: INSTALL-OS-UTILS: (1)       package
openstack-utils-2014.2.2-201502130059.ibm.el6.31.noarch is already installed
```

Figure 3-30 Warning message

3.3.12 Activate XCAT and ZHCP exits

Two sample exit scripts are provided on the z/VM MAINT 400 disk to allow the xCAT or ZHCP guest machine to erase various log files within the virtual server when it is started. The sample scripts are named XCATEXIT SAMPLE and ZHCPEXIT SAMPLE.

To activate an exit, complete the following steps:

1. Make a copy of the sample file on the MAINT 400 disk with the file type of PL instead of SAMPLE. You should make a copy rather than rename the file so that the original sample remains available for use later.
2. Restart the XCAT and ZHCP guest machines by restarting SMAPI. This procedure restarts IBM Cloud Manager with OpenStack for z Systems. Issue the following commands:
   - FORCE VSMGUARD
   - XAUTOLOG VSMGUARD
3. Verify that the log files have been removed. The z/VM console of the XCAT and ZHCP guest machines will contain messages that indicate the success of the log file removal. The Run Script window in the xCAT GUI allows you to verify that the files have been removed and new log files started.
4. Erase the active exit scripts (file type PL) from the MAINT 400 disk so that future starts of the virtual servers will not continue to remove the log files at start. Only remove the log files when necessary, and allow automated pruning to handle log file rotation and pruning.

To verify that your installation was successful, log in to IBM Cloud Manager with OpenStack for z Systems as administrator using a browser. The user ID is `admin` and the password is the value of `cmo_admin_password` as you defined in the `DMSSICMO` file (see Example 3-4 on page 60).

After you log in, the welcome page is displayed as shown in Figure 3-31.

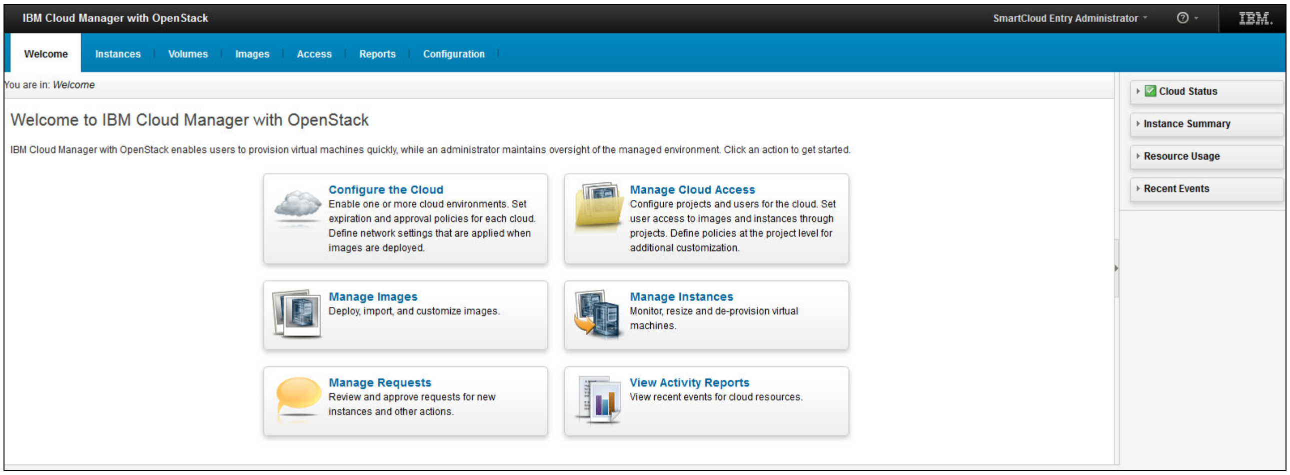


Figure 3-31 Administrator welcome page

If you have additional z/VM LPARs that will be part of your cloud infrastructure, follow the next section's instructions to include those LPARs as OpenStack compute nodes. Otherwise, skip to Chapter 4, "Building out the cloud infrastructure" on page 75.

3.4 Adding compute nodes to the cloud infrastructure

This is an optional step to include compute nodes in your cloud infrastructure to be managed by a single IBM Cloud Manager with OpenStack controller node. This step can be skipped if you plan to manage only a single z/VM LPAR.

This procedure is required for the following situations:

- Two or more z/VM SSI members
- Additional z/VM LPARs (ECKD or SCSI) in the same cloud environment

After one z/VM LPAR has IBM Cloud Manager with OpenStack for z Systems configured as a *controller* role, other z/VM LPARs should be configured as an OpenStack *compute* role to be managed from the OpenStack controller. This process allows you to manage multiple z/VM LPARs as a single cloud from a single point of management using the IBM Cloud Manager with OpenStack portal that runs in the OpenStack controller. All z/VM LPARs (OpenStack compute nodes) are managed from a single z/VM LPAR (OpenStack controller node).

3.4.1 How to configure an extra z/VM compute node to the cloud

The process to install an IBM Cloud Manager with OpenStack for z Systems compute node is similar to the installation process of the controller. The main differences are noted in the following list:

- OpenStack controller data disk pool is not required for a compute node
- For z/VM SSI members, MAINT630 102 and 103 minidisks are shared
- `DMSSICNF` configuration file points additional ZHCP to the controller xCAT
- `DMSSICMO` configuration files set up more z/VM LPARs as compute nodes

Use the following sections with companion steps to add more z/VM compute nodes to the cloud:

- z/VM environment preparation
- Installing the IBM Cloud Manager with OpenStack for z Systems

z/VM environment preparation

To prepare the z/VM environment, use the following steps:

1. Apply the latest z/VM RSU package to your z/VM environment as listed in 3.2.3, “Apply the latest z/VM RSU package” on page 44.
2. Apply the z/VM APARs required by IBM Cloud Manager with OpenStack for z Systems as listed in 3.2.4, “Apply the z/VM APARs required by IBM Cloud with OpenStack for z Systems” on page 45.
3. Prepare the following disk pools, if needed:
   - z/VM work disk pool: Skip preparation if your additional z/VM compute node is a z/VM member of the same SSI cluster. In this scenario, the z/VM member shares the same MAINT630 102 and 103 minidisk.
   - OpenStack controller data disk: Skip this step. The controller data disk pool is not required for a compute node.
   - OpenStack ephemeral disk pool: Skip this step if your additional z/VM compute node is a z/VM member of the same SSI cluster. In this scenario, the z/VM member shares the ephemeral disk pool previously defined in the DIRMAINT EXTENT CONTROL file.

   Otherwise, define the required disk pool as described in 3.2.5, “Prepare disk pools” on page 45.
4. Verify SMAPI configuration as listed in 3.2.6, “Verify SMAPI configuration” on page 49.
5. Configure the directory manager. Skip this step if your additional z/VM compute node is a z/VM member of the same SSI cluster. In this scenario, the z/VM member shares the DIRMAINT configuration. Otherwise, configure your directory manager as listed in 3.2.7, “Configure directory manager” on page 50.
6. Configure OSA EQID for z/VM SSI as listed in 3.2.8, “Configure OSA EQID for z/VM SSI” on page 51. This step is required only for additional z/VM compute nodes that are z/VM members of the same SSI Cluster.

Installing the IBM Cloud Manager with OpenStack for z Systems

Use the following steps to install the IBM Cloud Manager with OpenStack for z Systems:

1. Download the IBM Cloud Manager with OpenStack for z Systems image file. Follow instructions as listed in 3.3.1, “Download IBM Cloud Manager with OpenStack for z Systems image file” on page 52. Skip this step if you already have access to the installation files.
2. Create the z/VM minidisks for installation as listed in 3.3.2, “Create z/VM minidisks for installation” on page 53. Each z/VM compute node has its own XCAT 101 minidisk. Add MAINT630 102 and 103 minidisks. Do not perform MAINT630 of 102 and 103 minidisks definitions if your additional z/VM LPAR is a member of the same SSI cluster. In this scenario, the z/VM member has the same MAINT630 102 and 103 minidisk definitions.
3. Upload IBM Cloud Manager with OpenStack for z Systems image file as listed in 3.3.3, “Upload IBM Cloud Manager with OpenStack for z Systems image file” on page 55. Skip this step if your additional z/VM node is a z/VM member of the same SSI cluster. In this scenario, the z/VM member has the same MAINT630 102 that already has the packed CMA4202 file. Do not run the FORMAT commands in this case.

4. Unpack the uploaded CMA4202 image file as listed in 3.3.4, “Unpack uploaded CMA4202 image file” on page 56. Skip this step if your additional z/VM node is a z/VM member of the same SSI cluster. In this scenario, the z/VM member has the same MAINT630 103, which already has the unpacked CMA4202 file.
5. Restore the unpacked CMA4202 image file. To restore the XCAT 101 minidisk, do so as listed in 3.3.5, “Restore unpacked CMA4202 image file” on page 57 For the other z/VM SSI members, be sure that you log on as MAINT630 in the additional z/VM node to run the XCAT 101 restoration using the DDRREST command. In this scenario, the XCAT-2 101 minidisk is restored.
6. Increase xCAT z/VM user ID memory. Skip this step for a z/VM compute node (xCAT does not require 8 GB).
7. Customize the DMSSICNF COPY configuration file. Log on as MAINT and access the 193 minidisk to update the `DMSSICNF` and config file for this member, as shown in Figure 3-32. The following are considerations for z/VM SSI additional nodes:
   – Because z/VM SSI shares service DASDs between z/VM members, you can substitute the configuration files from one to another z/VM member. To do so, use the LOCALMOD, SERVICE, and PUT2PROD procedures to update the DMSSICNF configuration file in the additional z/VM compute node. Use the LOCALMOD, SERVICE, and PUT2PROD procedures only at the z/VM controller. For z/VM compute nodes, update the DMSSICNF COPY configuration file directly from the MAINT 193 minidisk. Consider keeping a backup copy of your configuration file.
   – Be sure to log on as MAINT at the z/VM compute node to update the DMSSICNF configuration file.

```
ACC 193 G
LISTFILE DMSSICNF COPY *
DMSSICNF COPY     G2
```

Figure 3-32 Accessing DMSSICNF config files of z/VM compute node

8. Use the **XEDIT** utility to update additional z/VM compute nodes in the DMSSICNF config file as shown in Example 3-5. For more information about DMSSICNF configuration options, see *Enabling z/VM for OpenStack (Support for OpenStack Juno Release),* SC24-6249.

Example 3-5 DMSSICNF config file for z/VM compute node

```
/* XCAT server defaults                                          */
XCAT_User     = "XCAT"              /* xCAT z/VM user ID         */
XCAT_Addr     = "10.10.10.10"       /* XCAT IP Address           */
XCAT_Host     = "xcat"              /* xCAT hostname             */
XCAT_Domain   = ".itso.ihost.com"   /* xCAT domain name          */
XCAT_vswitch  = "XCATVSW1"          /* xCAT Vswitch name         */
XCAT_OSAdev   = "2D63"              /* OSA address for xCAT      */
XCAT_zvmsysid = "itsossi2"          /* xCAT z/VM system id       */
XCAT_notify   = "OPERATOR"          /* Notify when xCAT started  */
XCAT_gateway  = "10.10.10.1"        /* Network gateway IP addr.  */
XCAT_netmask  = "255.255.255.0"     /* Default network mask      */
XCAT_vlan     = "NONE"
XCAT_iso = ""
XCAT_MN_Addr    = "192.168.60.102"  /* xCAT mgmt node IP address */
XCAT_MN_vswitch = "XCATVSW2"        /* xCAT MN Vswitch name      */
XCAT_MN_OSAdev  = "2D03"            /* OSA address for xCAT MN   */
```

```
XCAT_MN_gateway = "192.168.60.1"        /* Network gateway IP addr.  */
XCAT_MN_Mask    = "255.255.255.0"       /* Netmask for xCAT MN       */
XCAT_MN_vlan    = "NONE"
XCAT_MN_admin = "mnadmin"               /* MN administrator userid   */
XCAT_MN_pw    = "passwd"                /* MN admin password         */
                                        /* (if NOLOG, userid cannot  */
                                        /*  ssh into XCAT MN)        */
/* ZHCP server defaults                                              */
ZHCP_User    = "ZHCP"                   /* zhcp z/VM user ID         */
ZHCP_Addr    = "10.10.10.21"            /* zhcp IP ADDRESS           */
ZHCP_Host    = "zhcp2"                  /* zhcp hostname             */
ZHCP_Domain  = ".itso.ihost.com"        /* zhcp domain name          */
ZHCP_gateway  = "10.10.10.1"            /* Network gateway IP addr.  */
ZHCP_netmask  = "255.255.255.0"         /* Default network mask      */
ZHCP_vswitch = "XCATVSW1"               /* zhcp Vswitch name         */
ZHCP_OSAdev  = "2D63"                   /* OSA address for zhcp      */
ZHCP_vlan    = "NONE"
```

The meanings for the configuration options are as follows:

XCAT_Domain OpenStack compute node domain name

XCAT_OSAdev XCATVSW1 (for xCAT internal network) real OSA device address

XCAT_zvmsysid z/VM SYSID where OpenStack compute node will run (lowercase)

XCAT_MN_Addr OpenStack compute node (xCAT Management Node) IP address

XCAT_MN_OSAdev XCATVSW2 (for xCAT external network) real OSA device address

XCAT_MN_gateway OpenStack compute node default gateway

XCAT_MN_netmask OpenStack compute node default netmask

XCAT_MN_pw xCAT Management Node (mnadmin) default password

ZHCP_Addr ZHCP IP address, which must be different from the controller ZHCP IP address

ZHCP_Host ZHCP host name

ZHCP_Domain ZHCP domain name

ZHCP_OSAdev ZHCP (internal network) real OSA device address

Important: The integrated xCAT that is running with the OpenStack controller must reach all ZHCP services virtual machines (one per z/VM Operating System). In this scenario, it does so through the 10.10.10.x network. Be sure that you have the appropriate network definitions.

9. Customize the DMSSICMO COPY configuration file. Log on as MAINT and access the 193 minidisk to update the DMSSICMO and config file for this member as shown in Figure 3-33 on page 71. The following are considerations for z/VM SSI additional nodes:
   - Because z/VM SSI shares service DASDs between z/VM members, you can substitute the configuration files from one to another z/VM member. To do this, use the LOCALMOD, SERVICE, and PUT2PROD procedures to update the DMSSICMO configuration file in the additional z/VM compute node. Use the LOCALMOD, SERVICE, and PUT2PROD procedures only at the z/VM controller level. For z/VM compute node, update the DMSSICNF COPY configuration file directly from the MAINT 193 minidisk. Consider keeping a backup copy of your configuration file.

– Be sure to log on as MAINT at the additional z/VM compute node to update the DMSSICMO configuration file.

```
ACC 193 G

LISTFILE DMSSICMO COPY *
DMSSICMO COPY     G2
```

Figure 3-33 Accessing DMSSICMO config files of z/VM compute node

10. Use the XEDIT utility to update extra z/VM compute nodes in the `DMSSICMO` config file as shown in Example 3-6. For more information about DMSSICMO configuration options, see *Enabling z/VM for OpenStack (Support for OpenStack Juno Release),* SC24-6249.

Example 3-6 DMSSICMO config file for additional z/VM compute role

```
/* CMO User Configurable Settings                                   */
cmo_admin_password                   = "passwd"
cmo_data_disk                        = ""
openstack_system_role                = "compute"
openstack_controller_address         = "192.168.60.101"
openstack_zvm_diskpool               = "ECKD:GRPICM"
openstack_instance_name_template     = "itso%04x"
openstack_zvm_fcp_list               = "NONE"
openstack_zvm_timeout                = "300"
openstack_zvm_scsi_pool              = "NONE"
openstack_zvm_zhcp_fcp_list          = "NONE"
openstack_san_ip                     = "NONE"
openstack_san_private_key            = "NONE"
openstack_storwize_svc_volpool_name  = "NONE"
openstack_storwize_svc_vol_iogrp     = "NONE"
openstack_zvm_image_default_password = "password"
openstack_xcat_mgt_ip                = ""
openstack_xcat_mgt_mask              = ""
openstack_zvm_xcat_master            = "xcat"
```

The meanings for the configuration options are as follows:

cmo_admin_password	OpenStack compute node default password
openstack_system_role	OpenStack compute node role
openstack_controller_address	OpenStack controller IP address
openstack_zvm_diskpool	DIRMAINT region for Ephemeral (disk pool)
openstack_instance_name_template	z/VM user ID instance name template
openstack_zvm_image_defaults_password	Default root password for deployed instances

11. Put changes into production. Skip this step if your additional z/VM node is a z/VM member of the same SSI cluster.

    This scenario does not use the LOCALMOD command for the other z/VM SSI members to update the DMSSICNF and DMSSICMO configuration files. It is not necessary run the SERVICE and PUT2PROD procedures.

12. Copy the ID_RSA.PUB key file from the controller node to the compute node. The ZHCP z/VM service that runs in the z/VM compute nodes must communicate with the OpenStack controller node. Copy the controller node's public key to the z/VM compute node. Log on as MAINT in the z/VM compute node and follow the steps to run this copy using z/VM FTP Server as shown in Figure 3-34.

```
ACC 193 G
VMLINK TCPMAINT 592
DMSVML2060I TCPMAINT 592 linked as 0120 file mode Z
FTP 192.168.60.10
VM TCP/IP FTP Level 630
Connecting to 192.168.60.10, port 21
220-FTPSERVE IBM VM Level 630 at ITSOSSI1.ITSO.IBM.COM, 17:09:27 EDT MONDAY
2015-06-15
220 Connection will close if idle for more than 5 minutes.
USER (identify yourself to the host):
MAINT
>>>USER MAINT
331 Send password please.
Password:
>>>PASS ********
230 MAINT logged in; working directory = MAINT 191
Command:
GET ID_RSA.PUB ID_RSA.PUB.G
>>>EPRT |1|192.168.60.20|1029|
200 Port request OK.
>>>RETR ID_RSA.PUB
150 Sending file 'ID_RSA.PUB'
250 Transfer completed successfully.
```

Figure 3-34 Copying ID_RSA from the OpenStack controller to the compute node

13. Start and validate the IBM Cloud Manager with OpenStack for z Systems. To do so, start the z/VM compute note as listed in 3.3.10, "Start and validate IBM Cloud Manager with OpenStack for z Systems" on page 61.

    As shown in Figure 3-35 on page 73, that there are significant differences between the OpenStack controller and the OpenStack compute node:

    – Network: The z/VM compute node does participate in the internal 10.10.10.x xCAT network.

    – Services: Several OpenStack services that are running in the controller node are stopped in the compute node.

    – Nova-compute service: The nova-compute service-list shows one service for each z/VM compute node

```
[mnadmin@xcat ~] $ ifconfig
eth1      Link encap:Ethernet  HWaddr 02:00:1B:00:00:19
          inet addr:192.168.60.102  Bcast:192.168.60.255  Mask:255.255.255.0
          inet6 addr: fe80::1bff:fe00:19/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1492  Metric:1

[mnadmin@xcat ~] $ sudo service openstack-nova-compute status
openstack-nova-compute (pid  1980) is running...
[mnadmin@xcat ~] $ sudo service openstack-cinder-volume status
openstack-cinder-volume is stopped
[mnadmin@xcat ~] $ sudo service neutron-server status
neutron is stopped
[mnadmin@xcat ~] $ sudo service openstack-keystone status
openstack-keystone is stopped
[mnadmin@xcat ~] $ sudo service qpidd status
qpidd is stopped
[mnadmin@xcat ~] $ sudo service openstack-glance-registry status
openstack-glance-registry is stopped
[mnadmin@xcat ~] $ sudo service openstack-iaasgateway status
openstack-iaasgateway is stopped
[mnadmin@xcat ~] $ sudo service openstack-heat-engine status
openstack-heat-engine is stopped

[mnadmin@xcat ~] $ source openrc
[mnadmin@xcat ~] $ nova service-list
+----+------------------+----------+----------+---------+-------+
| Id | Binary           | Host     | Zone     | Status  | State |
+----+------------------+----------+----------+---------+-------+
| 1  | nova-cert        | itsossi1 | internal | enabled | up    |
| 2  | nova-console     | itsossi1 | internal | enabled | up    |
| 3  | nova-conductor   | itsossi1 | internal | enabled | up    |
| 5  | nova-consoleauth | itsossi1 | internal | enabled | up    |
| 6  | nova-scheduler   | itsossi1 | internal | enabled | up    |
| 7  | nova-compute     | itsossi1 | nova     | enabled | up    |
| 8  | nova-compute     | itsossi2 | nova     | enabled | up    |
+---+------------------+----------+----------+---------+-------+
```

Figure 3-35 OpenStack controller services verification

14. Install IBM Cloud Manager with OpenStack deployer as listed in 3.3.11, "Install IBM Cloud Manager with OpenStack Deployer" on page 64.

4

Building out the cloud infrastructure

This chapter provides instructions for configuring IBM Cloud Manager with OpenStack for z Systems and building out the cloud infrastructure.

This chapter contains these sections:

- Configure neutron data network
- Configure cinder persistent data disks
- Capture a deployable Linux image into OpenStack glance
- Configure email notification
- Configure metering
- Configure billing

If you are not familiar with the OpenStack capabilities on z Systems, see 1.6, “IBM Cloud Manager with OpenStack for z Systems” on page 13.

The scenarios created in this chapter are built on the reference architecture described in “Designing a cloud infrastructure with OpenStack in mind” on page 24 and the environment depicted in “Overview of scenario environment” on page 40.

4.1 Configure neutron data network

The OpenStack component that handles network communication is called *neutron*. Most of the neutron configuration was already done during installation (see “Customize the DMSSICNF configuration file” on page 58 and “Customize the DMSSICMO COPY configuration file” on page 60).

If you are using a single flat network (by default, the virtual switch is `xcatvsw2`) for both management data traffic and consumer data traffic, then no additional configuration steps are required. You can create the flat network and the associated subnet by using the self-service portal described in “Create networks in the self-service portal” on page 124.

The example scenario describes a preferred neutron data network configuration (flat and VLAN Mixed Network), as shown in Figure 2-1 on page 25. This configuration separates the management and data traffic into the following virtual switches:

- `xcatvsw2`**:** Used for management functions. In the example, it is configured as a flatnetwork.
- `data_vsw`: Used for the consumer data network. In the example, it is configured as a VLAN-aware network and name it `vswcloud`.

> **Note:** If you are using a different network scenario, you need to change your network configuration accordingly. For more information about network configuration, see *Enabling z/VM for OpenStack (Support for OpenStack Juno Release),* SC24-6249.

4.1.1 Configure neutron ML2 plug-ins

The Modular Layer 2 (ML2) plug-ins include drivers for the Flat and VLAN network types. To configure Flat and VLAN networks according to your network plan, you need to update the ML2 plug-ins configuration file (its default path is `/etc/neutron/plugins/ml2/ml2_conf.ini`), and change the `flat_networks` and `network_vlan_ranges` settings as shown in Example 4-1.

Before doing what is shown in Example 4-1, use an SSH logon to xCAT with the `mnadmin` user ID and issue the following command to make the changes:

```
[mnadmin@xcat ~] $ sudo vi /etc/neutron/plugins/ml2/ml2_conf.ini
```

Example 4-1 Sample changes in ml2_conf.ini configuration file

```
[ml2_type_flat]
# (ListOpt) List of physical_network names with which flat networks
# can be created. Use * to allow flat networks with arbitrary
# physical_network names.
#
# flat_networks =
# Example:flat_networks = physnet1,physnet2
# Example:flat_networks = *
flat_networks = xcatvsw2

[ml2_type_vlan]
# (ListOpt) List of <physical_network>[:<vlan_min>:<vlan_max>] tuples
# specifying physical_network names usable for VLAN provider and
# tenant networks, as well as ranges of VLAN tags on each
# physical_network available for allocation as tenant networks.
#
```

```
# network_vlan_ranges =
# Example: network_vlan_ranges = physnet1:1000:2999,physnet2
network_vlan_ranges = vswcloud:2:2999
```

Where:

- `xcatvsw2` is the virtual switch network name, used for management purposes, and does not support VLAN tagging (it is VLAN unaware).
- `vswcloud` is a virtual switch for one of the consumer data networks. It expects VLAN tagging (VLAN aware).

> **Important:** If you have compute nodes in your environment, you must update this configuration file accordingly. In the example case, the neutron ML2 configuration shown in Example 4-1 on page 76 is for a controller node z/VM `itsossi1`, so you need to update the corresponding options to configure for compute node z/VM `itsossi2`.

4.1.2 Configure neutron z/VM agent

The neutron z/VM driver is designed to enable OpenStack to use z/VM virtual network facilities. The main component of the neutron z/VM driver is `neutron-zvm-agent`, which is designed to work with a neutron server running with the ML2 plug-ins. The neutron ML2 plug-ins performs the database-related work, whereas `neutron-zvm-agent` uses the xCAT REST API to do configuration work on z/VM.

To configure the neutron z/VM agent, you must update the neutron z/VM plug-ins configuration file (its default path is `/etc/neutron/plugins/zvm/neutron_zvm_plugin.ini`), and change several options as shown in Example 4-2.

Before doing what is shown in Example 4-2, use an SSH logon to xCAT with the `mnadmin` user ID and then issue the following command to make the changes:

```
[mnadmin@xcat ~] $ sudo vi /etc/neutron/plugins/zvm/neutron_zvm_plugin.ini
```

Example 4-2 Sample changes in neutron_zvm_plugin.ini configuration file

```
[AGENT]
zvm_xcat_server = 192.168.60.101
zvm_xcat_username = admin
zvm_xcat_password = admin
zvm_host = itsossi1
xcat_zhcp_nodename = zhcp1
...
[vswcloud]
rdev_list=2D43
```

Where:

- `vswcloud` is the data virtual switch name.
- `2D43` is the real device address for the data virtual switch `vswcloud`.

In this case, the neutron z/VM agent automatically creates a virtual switch named `vswcloud`using rdev `2D43`, and the virtual switch will be VLAN aware.

Note: Each virtual switch needs at least one OSA portto be defined. The related `rdev_list` must be updated to list one of the OSA ports.

The OSA port needs to be connected to the trunk port in an Ethernet switch if the VLAN support is enabled. Also, make sure that your Ethernet switch knows the VLAN IDs that will be used in IBM Cloud Manager with OpenStack for z Systems.

If you have compute nodes in your environment, you must update this configuration file accordingly. In the example, the neutron z/VM agent configuration options shown in Example 4-2 on page 77 are for controller node z/VM `itsossi1`. Here are the corresponding options that were updated for z/VM itsossi2 (compute node):

- `zvm_host = itsossi2`
- `xcat_zhcp_nodename = zhcp2`
- `rdev_list = 2D43`

4.1.3 Restart the neutron server and neutron z/VM agent

To restart the neutron server and neutron z/VM agent on the controller node, issue the commands shown in Figure 4-1.

```
[mnadmin@xcat ~] $ sudo service openstack-nova-compute restart
Stopping openstack-nova-compute: [  OK  ]
Starting openstack-nova-compute: [  OK  ]
[mnadmin@xcat ~] $ sudo service neutron-server restart
Stopping neutron: [  OK  ]
Starting neutron: [  OK  ]
[mnadmin@xcat ~] $ sudo service neutron-zvm-agent restart
Stopping neutron-zvm-agent: [  OK  ]
Starting neutron-zvm-agent: [  OK  ]
```

Figure 4-1 Restarting neutron services on the controller node

On the compute node, issue the commands shown in Figure 4-2.

```
[mnadmin@xcat ~] $ sudo service openstack-nova-compute restart
Stopping openstack-nova-compute: [  OK  ]
Starting openstack-nova-compute: [  OK  ]
[mnadmin@xcat ~] $ sudo service neutron-zvm-agent restart
Stopping neutron-zvm-agent: [  OK  ]
Starting neutron-zvm-agent: [  OK  ]
```

Figure 4-2 Restarting neutron services on the compute node

Note: The neutron service does not run on the compute node, so you do not need to restart it on the compute nodes.

After restarting the neutron service and neutron z/VM agent, you can create the networks (and the subnet for each physical network) using the self-service portal as described in “Create networks in the self-service portal” on page 124.

4.2 Configure cinder persistent data disks

The OpenStack component that handles block storage connections is called *cinder*. Cinder integrates with several block storage components and provides users with an interface to request and consume block storage areas transparently. Consumers do not require knowledge of where the storage is deployed, what type of device is being used, or how to create the storage volume.

The storage volumes created through cinder are presented to the Linux instancesas extra persistent data disks. For more about persistent data disks, see “OpenStack storage services options” on page 8 and the content about persistent disks in “Storage planning considerations” on page 31.

> **Note:** At the time of writing, a SAN Volume Controller or IBM Storwize® V7000 storage system is required to use the cinder component in IBM Cloud Manager with OpenStack for z Systems.

This scenario describes the configuration of cinder persistent data disks based on the SCSI-oriented cloud infrastructure reference architecture shown in Figure 2-2 on page 26. Connect IBM z13 to the SAN Volume Controller through SAN switches and create the following disk pools:

- ZS-5W08-POOL1: Used for EDEVdisks, which are used for z/VM and ephemeral disks.
- ZS-5W08-POOL2: Used for FCP disks, which are used for persistent data disks.

> **Note:** Cinder services do not run on compute nodes.

To configure the cinder persistent disks, several configuration steps must be run on the controller node.

4.2.1 Create a SAN switch zone

Each Linux instanceneeds to have an FCP device attached to allow it to connect to the SAN and the respective LUN volume as defined in the SAN Volume Controller by cinder and managed by IBM Cloud Manager with OpenStack for z Systems.

IBM Cloud Manager with OpenStack for z Systems handles these tasks:

- Hold a pool of FCP devices
- Attach an available FCP device to the instance when requested
- Create a host mapping for the instance
- Configure and make the volume available to the instance to use

As there is no pre-definition of which FCP device will be attached to a specific cloud instance, all FCP devices managed by IBM Cloud Manager with OpenStack for z Systems must be able to connect to the SAN Volume Controller. So you need a SAN switch zone with all of the FCP devices and SAN Volume Controller ports.

IBM Cloud Manager with OpenStack for z Systems creates a unique SAN Volume Controller host mapping for each instance to ensure that each instance only has access to its own LUN volumes. The product also deletes this SAN Volume Controller host mapping when the volume is detached from the Linux instance.

4.2.2 Set up SSH connection between xCAT MN and SAN Volume Controller

To handle storage connections, cinder requires a *passwordless* SSH connection to the SAN Volume Controller. To enable this, you need to copy an SSH public key from `mnadmin` on the controller node to the SAN Volume Controller.

Important: Make sure that cinder can reach SAN Volume Controller through TCP/IP.

To set up the SSH connection, complete these steps:

1. Log on to xCAT on the controller node with the `mnadmin` user ID to generate an RSA key.

   Figure 4-3 shows how to generate an RSA key for `mnadmin` on the controller node.

```
[mnadmin@xcat ~] $ pwd
/home/mnadmin
[mnadmin@xcat ~] $ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/mnadmin/.ssh/id_rsa):
Created directory '/home/mnadmin/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/mnadmin/.ssh/id_rsa.
Your public key has been saved in /home/mnadmin/.ssh/id_rsa.pub.
The key fingerprint is:
d1:a5:6f:31:a7:68:2e:4a:5c:55:5b:53:c2:5e:16:ca mnadmin@xcat.itso.ihost.com
The key's randomart image is:
+--[ RSA 2048]----+
|            o.+oo|
|         . +.oooo|
|        . + +E.o |
|         o o =.  |
|        S o +    |
|     . . o .     |
|      o . .      |
|     . . .       |
|      .          |
+-----------------+
[mnadmin@xcat ~] $ chmod go+rx /home/mnadmin/.ssh
[mnadmin@xcat ~] $ chmod go+r /home/mnadmin/.ssh/id_rsa
```

Figure 4-3 Generating an RSA key for mnadmin on the controller node

Note: For cinder to connect to the SAN Volume Controller, you must change the permission of the private key file and .ssh directory as shown in Figure 4-3.

2. Download the public key (`/home/mnadmin/.ssh/id_rsa.pub`) to your local system.

3. Upload the public key to the SAN Volume Controller:

   a. Log on to the SAN Volume Controller GUI, open the **Access** pane, and then click **Users** as shown in Figure 4-4.

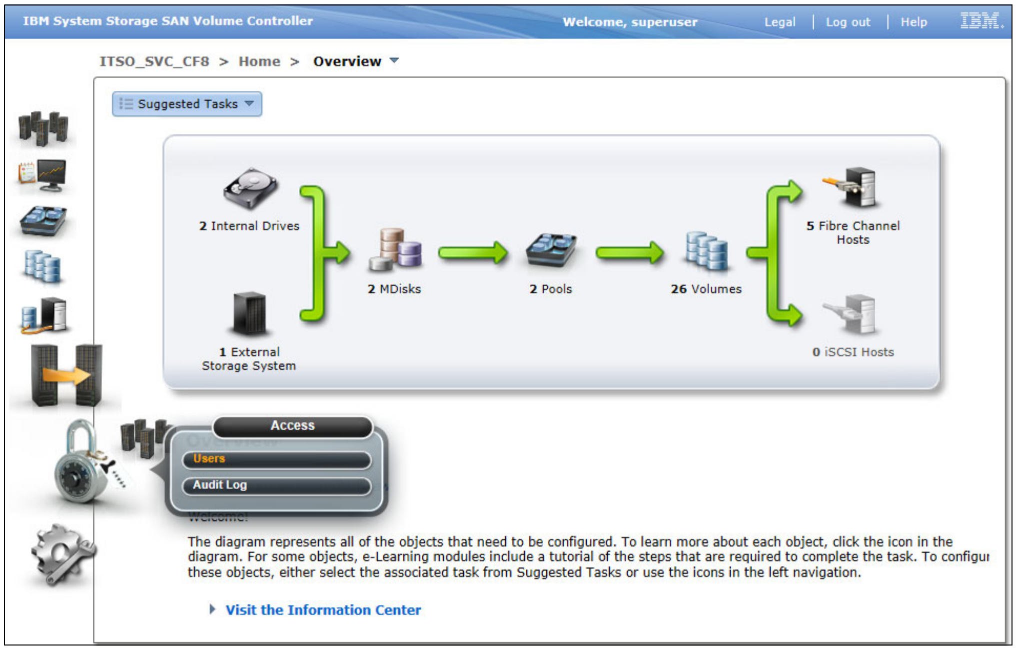


Figure 4-4 SAN Volume Controller Access Management panel

   b. On the users list (Figure 4-5), double-click **superuser** (the user ID).

> **Note:** You can use a different user ID if the ID that you select has sufficient authorities in SAN Volume Controller. In this example, **superuser** is in the **SecurityAdmin** group.

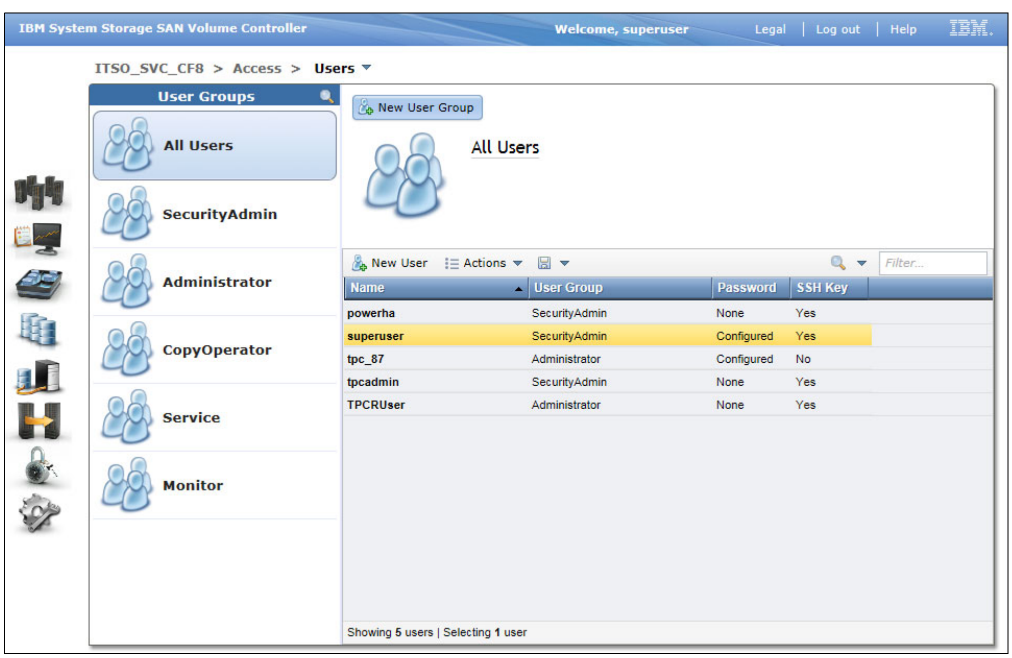


Figure 4-5 SAN Volume Controller users list

c. On the **User Properties** window, click **Change** in the SSH Public Key section as shown in Figure 4-6.

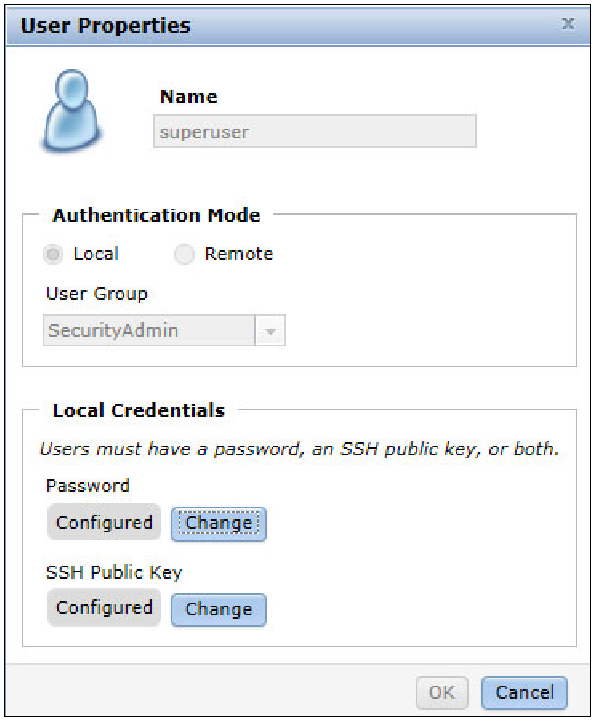


Figure 4-6 SAN Volume Controller User Properties panel

d. Specify the location of your public key file as shown in Figure 4-7.

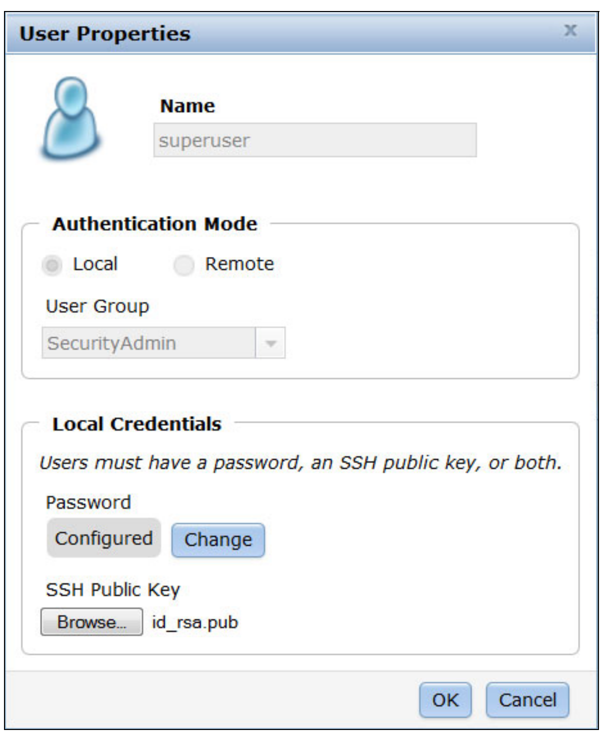


Figure 4-7 Specifying the location of the Public Key

e. Click **OK** to upload the SSH public key. Make sure the task completes without any errors, as shown in Figure 4-8.

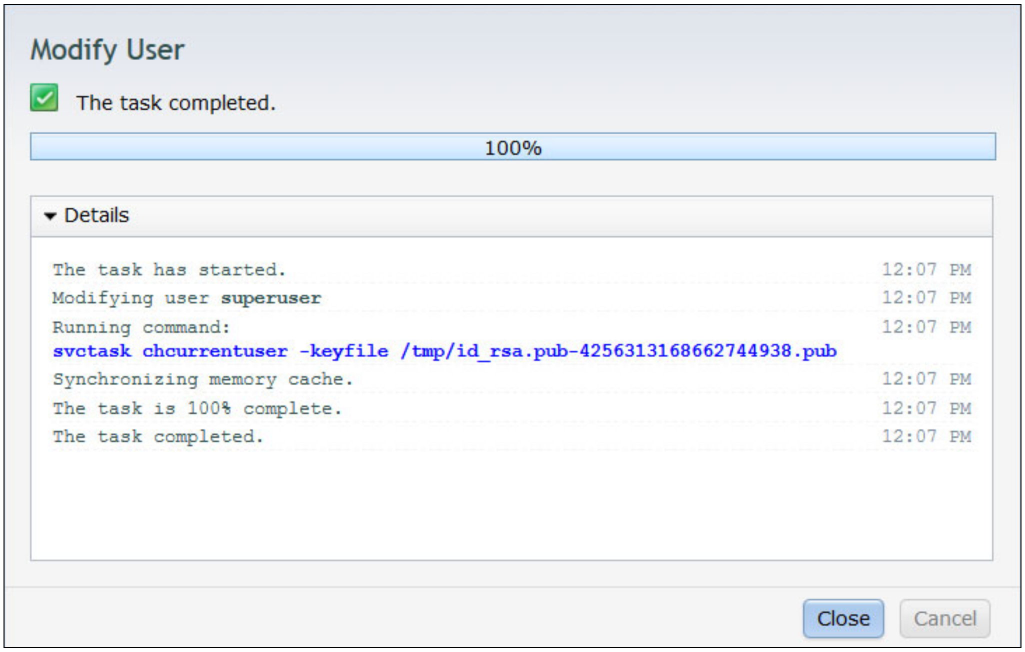


Figure 4-8 Result of modifying user in SAN Volume Controller

4.2.3 Configure nova.conf file

This section describes the configuration settings related to the nova z/VM driver. In the nova configuration file (`/etc/nova/nova.conf`), you can change the parameters as shown in Example 4-3.

Before undertaking what is shown in Example 4-3, use an SSH logon to xCAT with the `mnadmin` user ID and then issue the following command to make the changes:

```
[mnadmin@xcat ~] $ sudo vi /etc/nova/nova.conf
```

Example 4-3 Sample changes in nova.conf configuration file

```
zvm_fcp_list = b601-b61e;b701-b71e
zvm_scsi_pool = scsipool
zvm_zhcp_fcp_list = b61f;b71f
```

Where:

- `zvm_fcp_list` is the list of FCPs used by virtual machine instances.
- `zvm_scsi_pool` is the name of the xCAT SCSI pool.
- `zvm_zhcp_fcp_list` is the list of FCPs used only by the ZHCP node.

4.2.4 Configure cinder.conf file

This section describes the configuration settings related to the cinder z/VM driver. In the cinder configuration file (`/etc/cinder/cinder.conf`), you can change the parameters as shown in Example 4-4.

Before starting what is shown in Example 4-3, use an SSH logon to xCAT with the `mnadmin` user ID and then issue the following command to make the changes:

```
[mnadmin@xcat ~] $ sudo vi /etc/cinder/cinder.conf
```

Example 4-4 Sample changes in cinder.conf configuration file

```
san_ip = 192.168.60.5
san_private_key = /home/mnadmin/.ssh/id_rsa
storwize_svc_volpool_name = ZS-5W08-POOL2
storwize_svc_vol_iogrp = 0
storwize_svc_npiv_compatibility_mode=true
```

Where:

- `san_ip` is the IP address of your SAN Volume Controller.
- `san_private_key` is the fully qualified specification of the private key file to use for SSH authentication to your SAN Volume Controller storage.
- `storwize_svc_volpool_name` is the name of the volume pool within the SAN Volume Controller storage from which cinder allocates disks.
- `storwize_svc_vol_iogrp` is the `io_group_id` within the SAN Volume Controller storage to which the virtual disk is associated.
- `storwize_svc_npiv_compatibility_mode` specifies whether N_Port ID Virtualization (NPIV) mode has been enabled.

If you plan to have multiple backend storage systems, change the cinder configuration parameters as shown in Example 4-5. Use an SSH logon to xCAT with the `mnadmin` user ID and then issue the following command to make the changes:

```
[mnadmin@xcat ~] $ sudo vi /etc/cinder/cinder.conf
```

Example 4-5 Changes to the cinder configuration parameters for multiple backend storage systems

```
# A list of backend names to use. These backend names should
# be backed by a unique [CONFIG] group with its options (list
# value)
enabled_backends=itso_svc_pool2,itso_svc_pool3
...
[itso_svc_pool2]
volume_backend_name=IBM_ITSO_SVC_1
san_ip = 192.168.60.5
san_private_key = /home/mnadmin/.ssh/id_rsa
storwize_svc_volpool_name = ZS-5W08-POOL2
storwize_svc_vol_iogrp = 0
storwize_svc_npiv_compatibility_mode=true
volume_driver = cinder.volume.drivers.ibm.storwize_svc.StorwizeSVCDriver
storwize_svc_connection_protocol = FC
[itso_svc_pool3]
volume_backend_name=IBM_ITSO_SVC_2
san_ip = 192.168.60.5
san_private_key = /home/mnadmin/.ssh/id_rsa
```

```
storwize_svc_volpool_name = ZS-5W08-POOL3
storwize_svc_vol_iogrp = 0
storwize_svc_npiv_compatibility_mode=true
volume_driver = cinder.volume.drivers.ibm.storwize_svc.StorwizeSVCDriver
storwize_svc_connection_protocol = FC
```

Where:

- `enable_backends` is the list of back-end names to use. These back-end names should be backed by a unique [CONFIG] group with its options.
- `[itso_svc_storage 2/3]` is a unique [CONFIG] group with specific back-end options.
- `volume_backend_name` is the back-end name that you want to use.

4.2.5 Restart the nova compute node and cinder services

Issue the following commands to restart the nova compute node and cinder services (see Figure 4-9).

```
[mnadmin@xcat ~] $ sudo service openstack-nova-compute restart
Stopping openstack-nova-compute: [  OK  ]
Starting openstack-nova-compute: [  OK  ]
[mnadmin@xcat ~] $ sudo service openstack-cinder-api restart
Stopping openstack-cinder-api: [  OK  ]
Starting openstack-cinder-api: [  OK  ]
[mnadmin@xcat ~] $ sudo service openstack-cinder-scheduler restart
Stopping openstack-cinder-scheduler: [  OK  ]
Starting openstack-cinder-scheduler: [  OK  ]
[mnadmin@xcat tmp] $ sudo service openstack-cinder-volume restart
Stopping openstack-cinder-volume: [  OK  ]
Starting openstack-cinder-volume: [  OK  ]
```

Figure 4-9 Restarting the nova and cinder services

4.2.6 Create cinder volume type

To allow users to specify the back-end on which each volume is created, a cinder volume type must first be defined in the OpenStack database. Issue the commands shown in Figure 4-10 to create a cinder volume type (`svc-volume`) for the default storage back-end.

```
[mnadmin@xcat ~] $ source openrc
[mnadmin@xcat ~] $ cinder type-create svc-volume
+--------------------------------------+------------+
|                  ID                  |    Name    |
+--------------------------------------+------------+
| 78f72279-23af-4761-8930-e7ab3d3c9c4c | svc-volume |
+--------------------------------------+------------+
[mnadmin@xcat ~] $ cinder type-list
+--------------------------------------+------------+
|                  ID                  |    Name    |
+--------------------------------------+------------+
| 78f72279-23af-4761-8930-e7ab3d3c9c4c | svc-volume |
+--------------------------------------+------------+
```

Figure 4-10 Creating volume type for cinder configuration

If you have multiple back-end storage areas defined, issue the commands shown in Figure 4-11.

```
[mnadmin@xcat ~] $ cinder type-create gold
+--------------------------------------+------+
|                  ID                  | Name |
+--------------------------------------+------+
| c0b803a7-f351-4cf7-a480-66bc2186c325 | gold |
+--------------------------------------+------+
[mnadmin@xcat ~] $ cinder type-key gold set volume_backend_name=IBM_ITSO_SVC_1
[mnadmin@xcat ~] $ cinder type-create silver
+--------------------------------------+--------+
|                  ID                  |  Name  |
+--------------------------------------+--------+
| 35864e58-852b-453d-ad1e-29731c245615 | silver |
+--------------------------------------+--------+

[mnadmin@xcat ~]$ cinder type-key silver set volume_backend_name=IBM_ITSO_SVC_2
```

Figure 4-11 Creating volume type for cinder configuration for multiple back-end storage areas

After successfully restarting the nova and cinder services,and creating a volume type, you can manage the volumesby using the self-service portal described in “Volume management” on page 145.

4.3 Capture a deployable Linux image into OpenStack glance

The xCAT management node (MN) can capture an image from a running Linux node and create an image for later use. This section discusses how to set up the Linux on z Systems environment that is the target of the initial image capture, along with the process to define the environment to xCAT. In addition, the section describes capturing the system in xCAT and then defining the image into OpenStack glance.

If you already have an image file created by xCAT (for example, `/root/0100.img`), see “Image management” on page 140 to learn how to upload it to glance. Otherwise, create the image file by using the following steps.

4.3.1 Install Linux on z Systems on a virtual machine

The first step is installing Linux on z Systems on a virtual machine. The following are the prerequisites for capturing and deploying an image in IBM Cloud Manager with OpenStack for z Systems:

- Supported Linux distributions:
  - RHEL6.5
  - RHEL6.4
  - RHEL6.3
  - RHEL6.2
  - SLES11.3
  - SLES11.2
- The supported device types are ECKD and FBA. Based on the device type of your environment, you need to install the corresponding Linux guest machine.
- You must capture an instance with a root disk size that is no larger than 5 GB. If you deploy a larger root device, it is likely to time out while capturing the Linux system as an image. Otherwise, you can log on to xCAT MN and modify the timeout value in the `httpd` service to make image-create work successfully. See *Enabling z/VM for OpenStack (Support for OpenStack Juno Release)*, SC24-6249 for details.
- As a preferred practice, the root disk size should be defined in full gigabytes (for example, 4 GB or 5 GB). OpenStack specifies disk sizes in full gigabyte values.
- The Linux root file system must not be on logical volume and shouldbe in a single partition.
- The virtual address of the Linux root file system in the user directory of the guest machine should be `0100`.
- The image being captured should not have any network interface cards (NICs) defined below virtual address `1000`.

To set up a target Linux virtual machine that will be captured as an image, complete these steps:

1. Prepare a Linux on z Systems virtual machine on the z/VM system.

   In this example, prepare `rhel65a` for the Red Hat virtual machine and `sles11a` for the SUSE virtual machine.

   For more information about installing and configuring Linux on z Systems, see *The Virtualization Cookbook for IBM z/VM 6.3, RHEL 6.4, and SLES 11 SP3,* SG24-8147. Adjust the procedures that are described in the book to keep the resulting virtual machine within the bounds of the image requirements described earlier.

2. Install the `genisoimage` and `openssl` modules on the virtual machine and verify the installations as shown in Figure 4-12.

```
[root@rhel65a ~]# rpm -qa | grep genisoimage
genisoimage-1.1.9-12.el6.s390x
[root@rhel65a ~]# rpm -qa | grep openssl
openssl-1.0.1e-15.el6.s390x
```

Figure 4-12 Verifying the required modules have been installed

3. Confirm that SELinux is disabled as shown in Figure 4-13, and that the *SSH* connection (default port number is 22) can pass through the firewall.

```
[root@rhel65a ~]# getenforce
Disabled
```

Figure 4-13 Verifying the SELinux is disabled

4. Set the `/etc/ssh/sshd_config` file so that domain name services (DNS) are not used (UseDNS no) to improve the inventory collection efficiency, and then verify it as shown in Figure 4-14.

```
[root@rhel65a ~]# cat /etc/ssh/sshd_config | grep UseDNS
UseDNS no

If the option is set to yes, update the file and restart the sshd service:
[root@rhel65a ~]# service sshd restart
Stopping sshd: [  OK  ]
Starting sshd: [  OK  ]
```

Figure 4-14 Verifying UseDNS option is set to no

Define the source system as an xCAT node

At the time of writing, the xCAT GUI is required to capture a Linux system as an OpenStack glance image.

Perform the following steps to define the source Linux system as an xCAT node:

1. Open the xCAT GUI, log in to xCAT, and then go to the Script window for the xCAT MN node.

   a. In a browser, open a connection to the IP address of the xCAT Management Node (for example, `https://192.168.60.101/xcat`).

      The logon window is displayed as shown in Figure 4-15. Use `admin` and `password` as the user name and password.

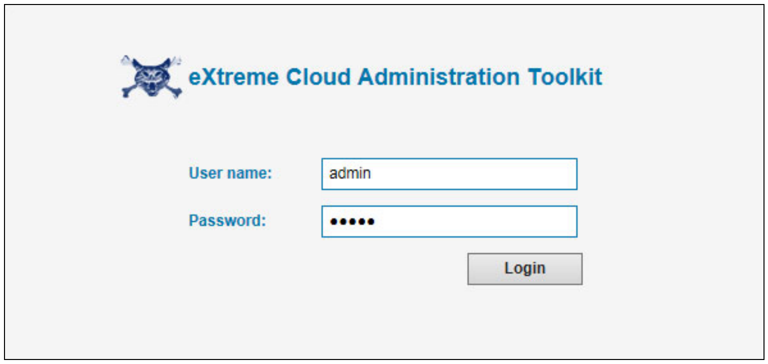


Figure 4-15 xCAT logon window

   b. Go to **Nodes** → **groups** → **all** → **Nodes**. This window is shown in Figure 4-16.

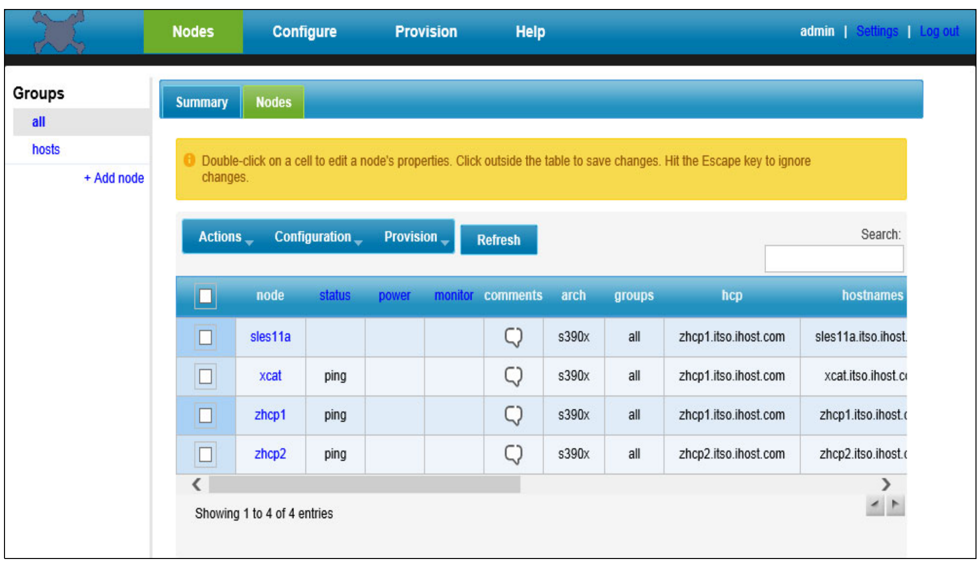


Figure 4-16 xCAT node list

   c. Select the xcat nodeby clicking the check box beside the node name.

d. Click **Actions** → **Run Script** as shown in Figure 4-17.

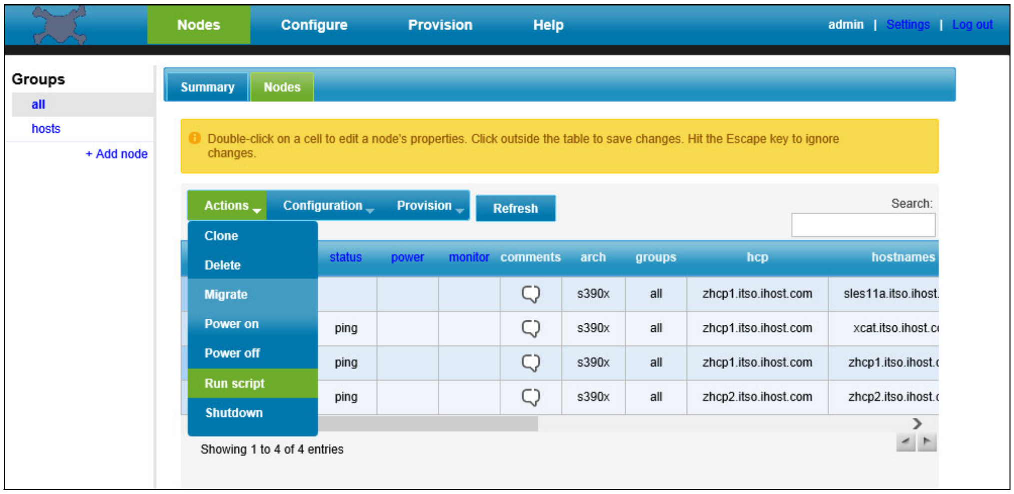


Figure 4-17 Run script action for xCAT node

e. In the script box, enter this statement:

/opt/xcat/bin/mkdef -t node -o **rhel65a** userid=**rhel65a** hcp=**zhcp1.itso.ihost.com** mgt=zvm groups=all

Where:

- `-o rhel65a` is the node name of your target system that you prepared in "Prepare a Linux on z Systems virtual machine on the z/VM system." on page 87. The node name is the short DNS host name for the system.
- `userid=rhel65a` is the user ID name of your target system in z/VM CP user directory.
- `zhcp1.itso.ihost.com` is the ZHCP server's host name.

Figure 4-18 shows the script box content.

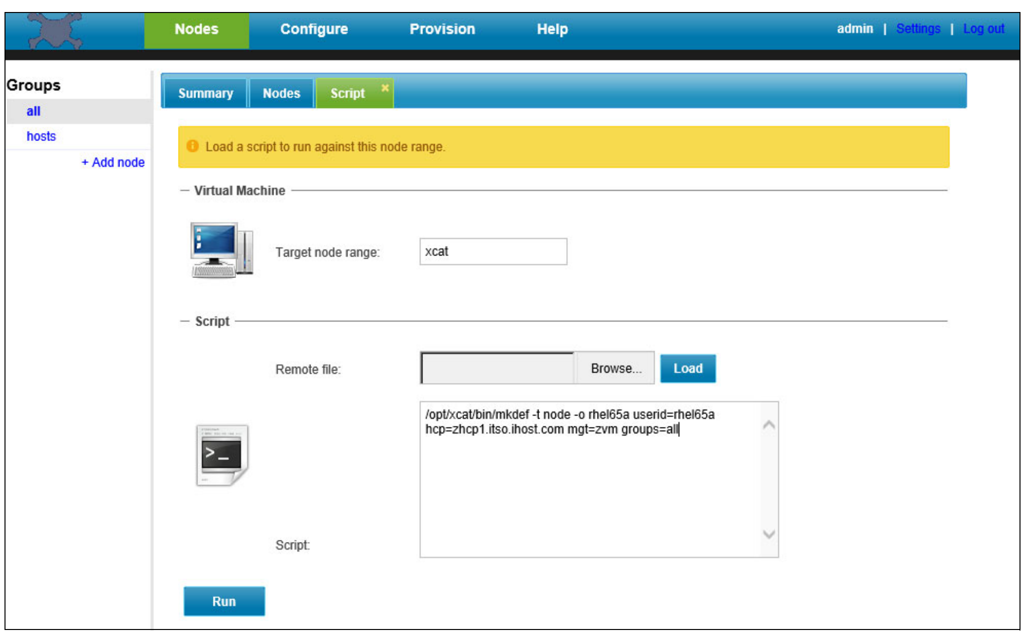


Figure 4-18 xCATnode definition sample

f. Click **Run**. Make sure that the return code is `0` as shown in Figure 4-19.

> **Note:** It is important that every Run Script request completes with reason code `(RC) 0`.

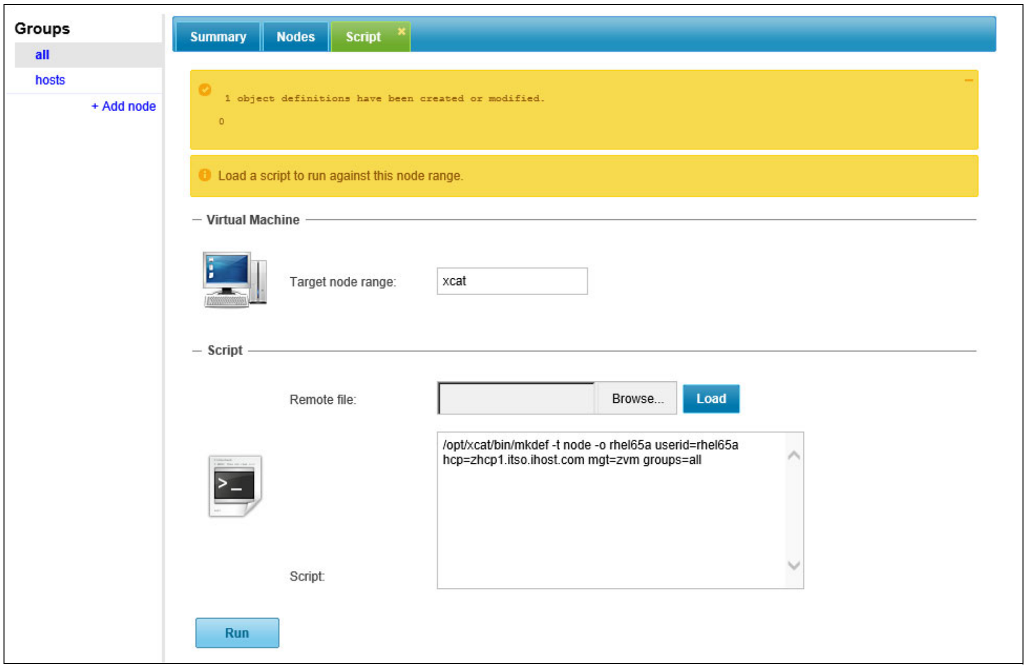


Figure 4-19 xCAT node definition result

2. Update the node's properties by issuing the **chtab** command using the XCAT GUI Script window. Enter the following command and then click **Run**. Make sure that the return code is `0`.

```
/opt/xcat/sbin/chtab node=rhel65a hosts.ip="192.168.60.99"
hosts.hostnames="rhel65a.itso.ihost.com" noderes.netboot=zvm
nodetype.os=rhel6.5 nodetype.arch=s390x nodetype.profile=rhel65a
nodetype.provmethod=netboot
```

> **Note:** This command is a single-line command that should be issued without inserting carriage returns. If you see `Command not found` errors, there is likely a carriage return in your command. Paste it into a separate text editor to locate the problem.

Where:

– `node=rhel65a` identifies the node name of the target system that you prepared.

– `192.168.60.99` is the IP address of your target system.

– `rhel65a.itso.ihost.com` is the host name of your target system.

– `rhel6.5` is the Linux on z Systems distribution version. The value should show the distribution name in lowercase (in other words, `rhel` or `sles`), followed by the version. The value should not contain blanks. For example, `rhel6, rhel6.4, sles11, sles11.2, sles11sp2`, and so on.

– `nodetype.profile=rhel65a` identifies the profile name for the node.

Figure 4-20 shows the script box content.

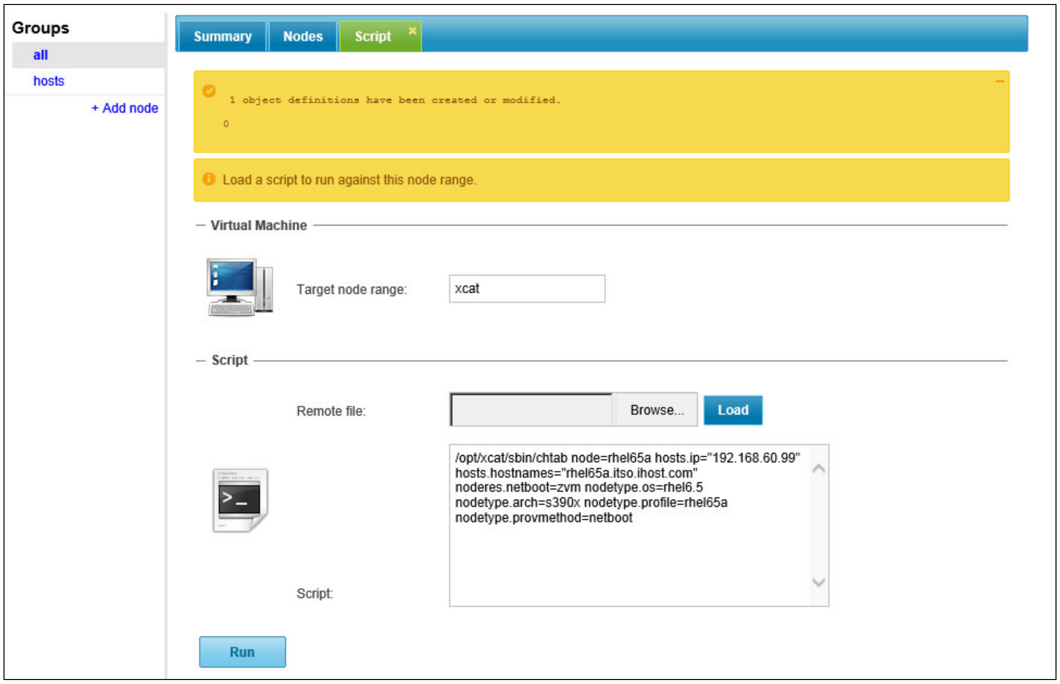


Figure 4-20 xCAT node definition update example

3. Create a host by issuing the **`makehosts`** command using the XCAT GUI Script window. Enter the following command and then click **Run**. Make sure that the return code is `0`.

`/opt/xcat/sbin/makehosts`

Figure 4-21 shows the script box content.

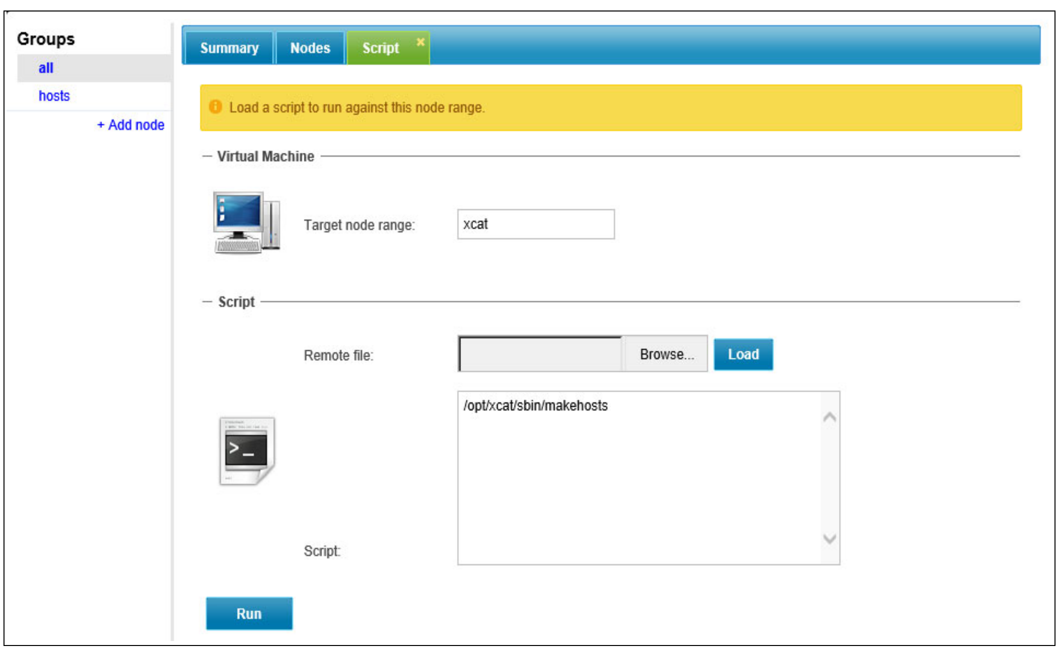


Figure 4-21 xCAT host definition example

4. Start the virtual machine of the Linux system that you want to capture, if it is not already started. You can start the virtual server using the xCAT GUI by navigating to the **Nodes** → **groups** → **all** → **Nodes** window and then selecting the virtual machine from the list as shown in Figure 4-22.

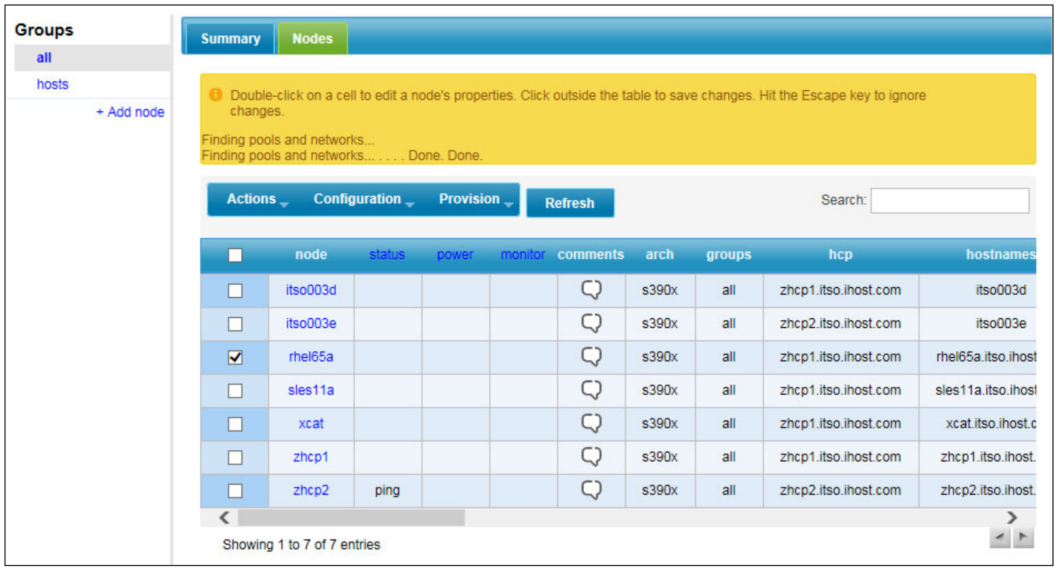


Figure 4-22 xCAT - Selecting target system

5. Click **Actions** → **Power On**, as shown in Figure 4-23.

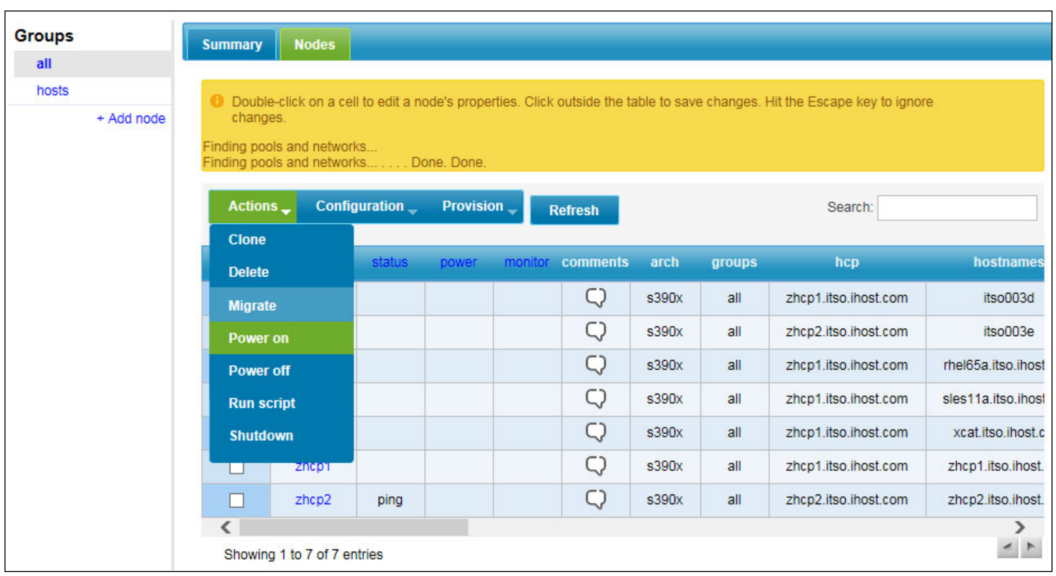


Figure 4-23 xCAT Power on Node - Action

6. Unlock the node to allow xCAT MN to communicate with it by completing these steps:

   a. Go to the **Nodes** → **groups** → **all** → **Nodes** window. The Groups selection (in the Groups frame on the left side of the window) should be **all**.

   b. Designate the node to be unlocked by selecting the check box beside its name.

c. Click **Configuration** → **Unlock**, as shown in Figure 4-24.

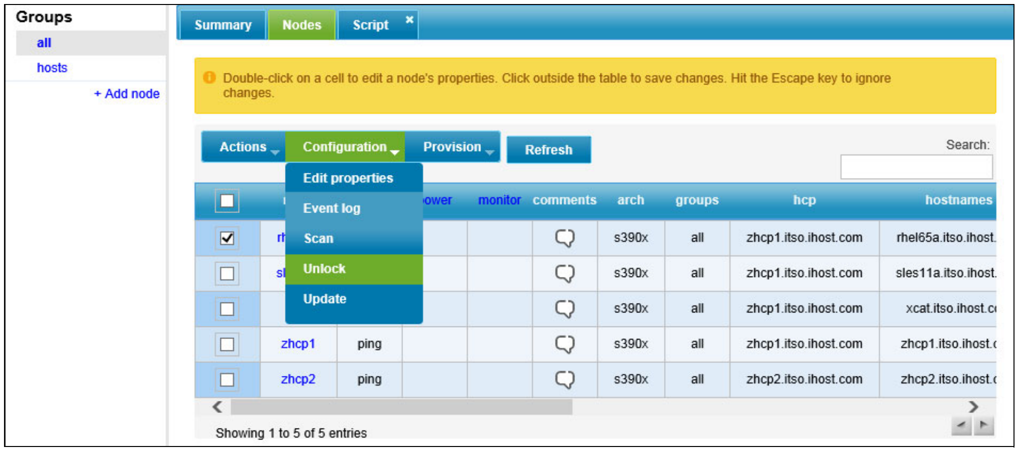


Figure 4-24 xCAT node unlocking example

d. In the password input field, specify the root password for the system that is being unlocked and then click **Unlock**, as shown in Figure 4-25.

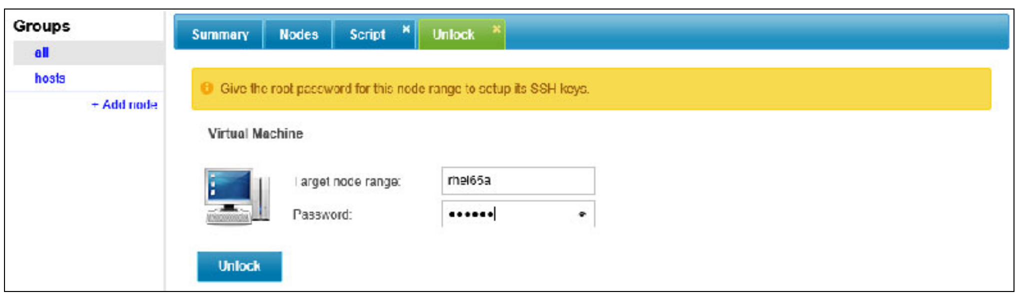


Figure 4-25 Providing a password to unlock the node

Configure xcatconf4z

xCAT supports making changes to a Linux on z Systems virtual machine when Linux is shut down or the virtual machine is logged off. The changes to Linux are implemented when Linux is next started. To do this, a script must be installed in the Linux on z Systems virtual machine so it can process change request files transmitted by the xCAT MN through ZHCP to the reader of the virtual machine as a class X file.

The script is called `xcatconf4z`, and is at `/opt/xcat/share/xcat/scripts` in the xCAT MN. The script should be installed in a machine that can be managed when it is logged off. This includes a Linux on z Systems image that is captured for netboot or sysclone deploys.

The Juno version of the OpenStack requires that the `xcatconf4z` be at version 2.0 or later. You should always use the latest version of the script in your images.

Complete the following steps to configure `xcatconf4z` for the target Linux system:

1. For images that are being used with the OpenStack Juno, and later releases, ensure that the `xcatconf4z` script you are using is version 2.0 or later. To verify this, complete these steps:

   a. Open the xCAT GUI and authenticate into xCAT.

   b. Go to the Script window for the xCAT MN node (`xcat`) and issue the **`xcatconf4z`**command to show the version of the file.

In the Script box, enter the following command and then click **Run**:

`/opt/xcat/share/xcat/scripts/xcatconf4z version`

See Figure 4-26.

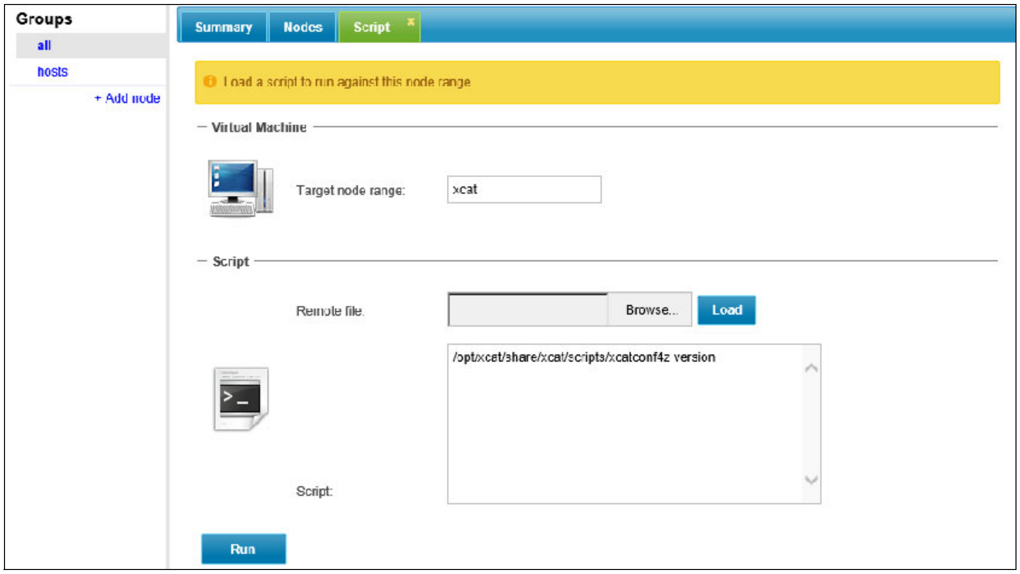


Figure 4-26 xCAT xcatconf4 version request

If the command is successful, you see the version information as shown in Figure 4-27.

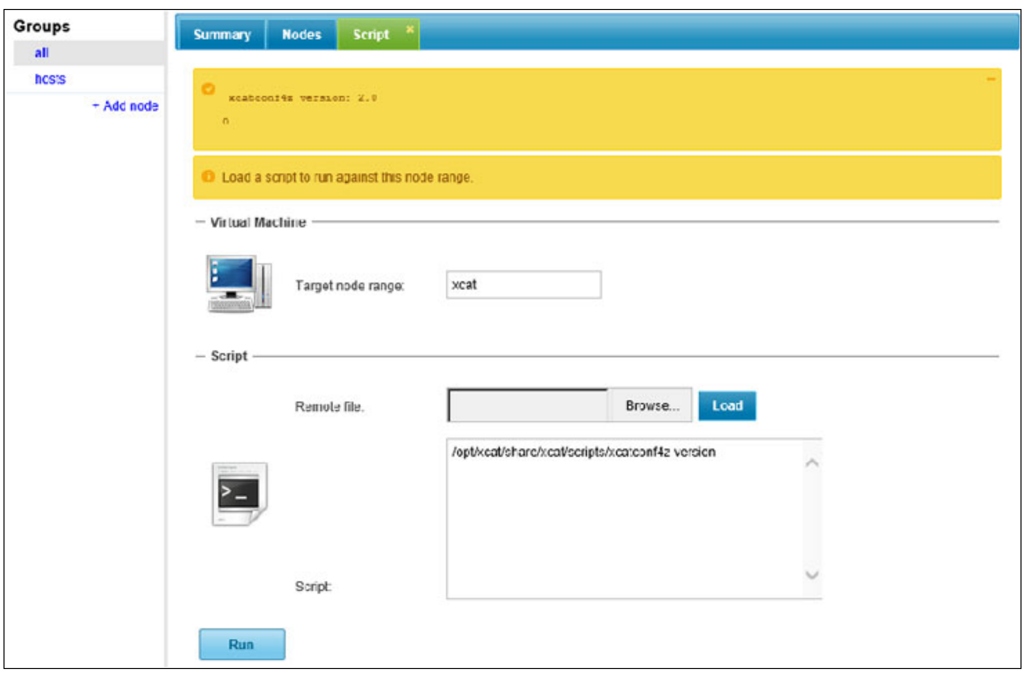


Figure 4-27 xCAT xcatconf4 version display

2. Copy the xcatconf4z script from the xCAT MNto the target system.

   To accomplish this, open the xCAT GUI and authenticate into xCAT. Go to the Script window for the xCAT MN node (`xcat`) issue the **`scp`** command to copy the `xcatconf4z` file to the target system, and then click **Run**. Make sure that the return code is `0`.

   `/usr/bin/scp /opt/xcat/share/xcat/scripts/xcatconf4z` **`rhel65a:/opt`**`/xcatconf4z`

   Where:

   – `rhel65a` is the node name of the system that is being set up for image capture.

   – `/opt` is the target location to receive the `xcatconf4z` file.

   See Figure 4-28.

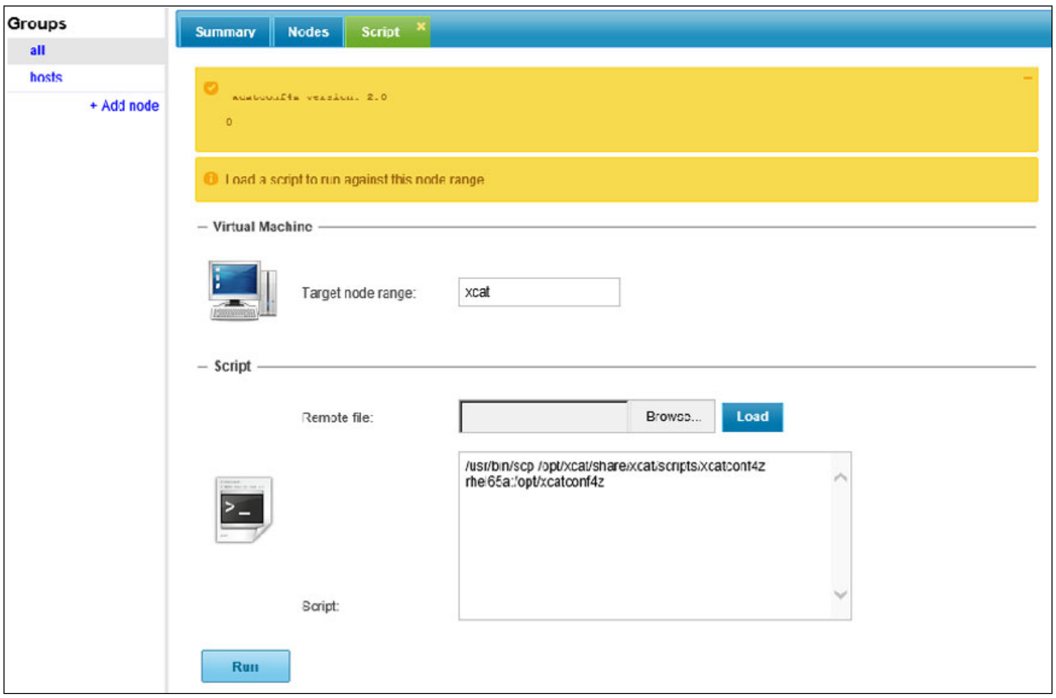


Figure 4-28 xCAT xcatconf4 copied to target Linux system

3. In the target Linux machine, log in with root user ID and change the script to specify the authorized senders.It can be set to *, which indicates that any virtual machine can send configuration requests to it (not a preferred practice). Or it can be used to indicate a list of user IDs of xCAT ZHCP machines that are allowed to transmit changes to the machine.

   Issue the following command to make the change:

   `[root@rhel65a ~]#` **`vi /opt/xcatconf4z`**

   Change it from:

   `authorizedSenders=''`

   to:

   `authorizedSenders='ZHCP'`

4. Install the `xcatconf4z` file in the target Linux machine:

   a. Copy the `xcatconf4z` file to `/etc/init.d` and make it executable:

      `cp /opt/xcatconf4z /etc/init.d/`

      `chmod +x /etc/init.d/xcatconf4z`

b. Add the xcatconf4z as a service by issuing this command:

`chkconfig --add xcatconf4z`

c. Activate the script:

`chkconfig xcatconf4z on`

5. Verify that you installed the correct version of `xcatconf4z` on the target machine. Do this by issuing the command that is shown in Figure 4-29.

```
[root@rhel65a ~]# service xcatconf4z version
xcatconf4z version: 2.0
```

Figure 4-29 Verifying the version of xcatconf4z

6. Verify that `xcatconf4z` on the target machine is configured to handle configuration requests from ZHCP servers. Also, verify that the user ID of the machine that is running ZHCP is correctly specified. Do this by issuing the command shown in Figure 4-30.

```
[root@rhel65a ~]# service xcatconf4z status
xcatconf4z is enabled to accept configuration reader files from: ZHCP
```

Figure 4-30 Verifying that xcatconf4z can accept reader files from ZHCP

4.3.2 Installation and configuration of the enablement framework

An activation engine is an enablement framework that is used for boot-time customization of virtual images. You can choose your own underlying activation engine, such as `cloud-init` or `scp-cloud-init`, according to your requirements. This example uses `cloud-init` when showing how to configure an image.

OpenStack uses `cloud-init` as its activation engine. z/VM OpenStack has been tested with `cloud-init` versions 0.7.4 and 0.7.5. The examples presented here use version 0.7.4. If you are using a different version, change your commands accordingly.

Perform these steps to install and configure cloud-init:

- Install and configure cloud-init on RHEL
- Install and configure cloud-init on SLES

Install and configure cloud-init on RHEL

To install and configure cloud-init on RHEL, complete the following steps:

1. Verify that the `setuptools` package has been installed in your target system by using the commands shown in Figure 4-31.

   For more information about `setuptools`, see the following website:

   https://pypi.python.org/pypi/setuptools

```
[root@rhel65a ~]# rpm -qa | grep setuptools
python-setuptools-0.6.10-3.el6.noarch
```

Figure 4-31 Verifying the setuptools is installed on the target system

2. Download the `cloud-init` tar file from this address:

   https://launchpad.net/cloud-init/+download

3. Decompress the tar file using the commands shown in Figure 4-32.

```
[root@rhel65a tmp]# pwd
/tmp
[root@rhel65a tmp]# ls cloud-init-0.7.4.tar.gz
cloud-init-0.7.4.tar.gz
[root@rhel65a tmp]# tar -zxvf cloud-init-0.7.4.tar.gz -C /opt
```

Figure 4-32 Decompressing the cloud-init tar file

> **Note:** In Figure 4-32, `cloud-init-0.7.4` is placed in the `/opt` directory. You can use a different directory if needed.

4. Build `cloud-init` using the commands in Figure 4-33.

```
[root@rhel65a tmp]# cd /opt/cloud-init-0.7.4
[root@rhel65a cloud-init-0.7.4]# python setup.py build
```

Figure 4-33 Building cloud-init

5. Establish a network connection to the following address (required for `cloud-init` installation):

   `https://pypi.python.org/simple/`

6. Install `cloud-init`. If your target system has successfully connected to the address provided, issue the command shown in Figure 4-34.

```
[root@rhel65a tmp]# cd /opt/cloud-init-0.7.4
[root@rhel65a cloud-init-0.7.4]# python setup.py install
```

Figure 4-34 Installing cloud-init

If the connection is not valid, the command will fail and display the error message shown in Figure 4-35.

```
Installed
/usr/local/lib64/python2.6/site-packages/cloud_init-0.7.4-py2.6.egg
Processing dependencies for cloud-init==0.7.4
Searching for requests
Reading https://pypi.python.org/simple/requests/
Download error on https://pypi.python.org/simple/requests/: [Errno -2] Name
or service not known -- Some packages may not be found!
Couldn't find index page for 'requests' (maybe misspelled?)
Scanning index of all packages (this may take a while)
Reading https://pypi.python.org/simple/
Download error on https://pypi.python.org/simple/: [Errno -2] Name or
service not known -- Some packages may not be found!
No local packages or download links found for requests
error: Could not find suitable distribution for
Requirement.parse('requests')
```

Figure 4-35 Sample error messages when installing cloud-init

To circumvent the connection problem, set up a local repository of the website. The example uses `/tmp/simple` as the local repository. If you choose another location, change the commands accordingly.

To set up the local repository, complete these steps:

a. Set up the directory `/tmp/simple` as the local repository:

```
[root@rhel65a tmp]# mkdir /tmp/simple
```

b. Go to the following website:

`https://pypi.python.org/simple/`

c. Download the latest packages for the folders shown in Figure 4-36, which shows the sample folder structure after downloading.

```
[root@rhel65a simple]# pwd
/tmp/simple
[root@rhel65a simple]# ls
argparse  cheetah    jsonpatch    Markdown  PrettyTable  pyyaml    six
boto      configobj  jsonpointer  oauth     pyserial     requests
[root@rhel65a simple]# cd requests/
[root@rhel65a requests]# ls
requests-2.7.0.tar.gz
```

Figure 4-36 Sample folder structure of local repository

d. Make the following changes to the `easy_install.py` script. This script can be found in `/usr/lib/python2.6/site-packages/setuptools/command/easy_install.py`.

From

```
self.index_url = None
```

To:

```
self.index_url = '/tmp/simple'
```

e. Install cloud-init using the commands shown in Figure 4-37.

```
[root@rhel65a tmp]# cd /opt/cloud-init-0.7.4
[root@rhel65a cloud-init-0.7.4]# python setup.py install
```

Figure 4-37 Installing cloud-init

Note: You might also encounter the following error messages:

```
Installed /usr/local/lib64/python2.6/site-packages/jsonpatch-1.11-py2.6.egg
Searching for configobj
Couldn't find index page for 'configobj' (maybe misspelled?)
Scanning index of all packages (this may take a while)
No local packages or download links found for configobj
error: Could not find suitable distribution for
Requirement.parse('configobj')
```

If these messages are displayed, issue the following command to circumvent the problem:

```
easy_install /tmp/simple/configobj/configobj-5.0.6.tar_.gz
```

Then issue the following commands again:

```
[root@rhel65a tmp]# cd /opt/cloud-init-0.7.4
[root@rhel65a cloud-init-0.7.4]# python setup.py install
```

7. Copy `cloud-init-related` services to the `/etc/init.d` directory, as shown in Figure 4-38.

```
[root@rhel65a tmp]# cp /opt/cloud-init-0.7.4/sysvinit/redhat/* /etc/init.d
```

Figure 4-38 Copying cloud-init related services to /etc/init.d

8. Update **/etc/init.d/cloud-init-local** to ensure that it starts after the `xcatconf4z` and `sshd` services:

   `vi /etc/init.d/cloud-init-local`

   On RHEL 6, change the # Required-Start line in the ### BEGIN INIT INFO section from:

```
### BEGIN INIT INFO
# Provides: cloud-init-local
# Required-Start: $local_fs $remote_fs
# Should-Start: $time
# Required-Stop:
```

   to:

```
### BEGIN INIT INFO
# Provides: cloud-init-local
# Required-Start: $local_fs $remote_fs xcatconf4z sshd
# Should-Start: $time
# Required-Stop:
```

9. Tailor the default configuration file `/etc/cloud/cloud.cfg` for RHEL (the default is for ubuntu) by completing these steps:

   a. Replace `distro:ubuntu` with `distro:rhel` (at approximately line 81).

   b. Change the default user name, password, and gecos as needed (at approximately lines 84 to 86).

   c. Change the `groups` tag to remove user groups that are not available for this distribution. After the change, the `groups` tag (at approximately line 87) should appear similar to the following:

```
groups: [adm, audio, cdrom, dialout, dip, floppy, video]
```

See Example 4-6.

Example 4-6 Sample changes in cloud.cfg configuration file

```
system_info:
   # This will affect which distro class gets used
   distro: rhel
   # Default user name + that default users groups (if added/used)
   default_user:
     name: rhel
     lock_passwd: True
     gecos: rhel
     groups: [adm, audio, cdrom, dialout, floppy, video,dip]
     sudo: ["ALL=(ALL) NOPASSWD:ALL"]
     shell: /bin/bash
```

10. The `cloud-init` process tries to add the user named `syslog` to the `group` adm. RHEL does not have a syslog user by default, so issue this command:

```
useradd syslog
```

11. Add the `cloud-init-related` service with the following commands:

```
chkconfig --add cloud-init-local
chkconfig --add cloud-init
chkconfig --add cloud-config
chkconfig --add cloud-final
```

Then, start the services with this sequence:

```
chkconfig cloud-init-local on
chkconfig cloud-init on
chkconfig cloud-config on
chkconfig cloud-final on
```

You can issue **`ls -l /etc/rc3.d | grep -e xcat -e cloud`** to find the services. Make sure that **xcatconf4z** starts before any `cloud-init` service.

See Figure 4-39.

```
[root@rhel65a ~]# ls -l /etc/rc3.d/ | grep -e xcat -e cloud
lrwxrwxrwx  1 root root 20 Jun  4 15:48 S50xcatconf4z -> ../init.d/xcatconf4z
lrwxrwxrwx  1 root root 26 Jun  4 15:48 S56cloud-init-local ->
../init.d/cloud-init-local
lrwxrwxrwx  1 root root 20 Jun  4 15:48 S57cloud-init -> ../init.d/cloud-init
lrwxrwxrwx  1 root root 22 Jun  4 15:48 S58cloud-config -> ../init.d/cloud-config
lrwxrwxrwx  1 root root 21 Jun  4 15:48 S59cloud-final -> ../init.d/cloud-final
```

Figure 4-39 Verifying xcatconf4z starts before any cloud-init services

12. To verify `cloud-init` configuration, issue this command:

```
cloud-init init --local
```

Make sure that no errors occur. These warning messages can be ignored:

```
/usr/lib/python2.6/site-packages/Cheetah-2.4.4-py2.6.egg/Cheetah/Compiler.py:15
09: UserWarning:
You don't have the C version of NameMapper installed! I'm disabling Cheetah's
useStackFrames option as it is painfully slow with the Python version of
NameMapper. You should get a copy of Cheetah with the compiled C version of
NameMapper.
```

```
You don't have the C version of NameMapper installed!
```

13. Finally, issue the command **rm -rf /var/lib/cloud** so that cloud-init will work after a restart.

Install and configure cloud-init on SLES

To install and configure `cloud-init` on SLES, complete these steps:

1. Verify that the `setuptools` package has been installed in your target system by issuing the command shown in Figure 4-40. For more information about `setuptools`, see the following website:

   https://pypi.python.org/pypi/setuptools

   In the example, `setuptools-17.0` was installed in the `/opt` directory. If you are using a different version and location, change the commands accordingly.

```
sles11a:/tmp # ls setuptools-17.0.zip
setuptools-17.0.zip
sles11a:/tmp # unzip setuptools-17.0.zip -d /opt/
sles11a:/tmp # cd /opt/setuptools-17.0/
sles11a:/opt/setuptools-17.0 # python setup.py install
```

Figure 4-40   Installing setuptools on SLES Linux system

2. Download the **cloud-init** tar file from the following address:

   https://launchpad.net/cloud-init/+download

3. Decompress the tar file using the commands shown in see Figure 4-41.

```
sles11a:/tmp # pwd
/tmp
sles11a:/tmp # ls cloud-init-0.7.4.tar.gz
cloud-init-0.7.4.tar.gz
sles11a:/tmp # tar -zxvf cloud-init-0.7.4.tar.gz -C /opt
```

Figure 4-41   Extract cloud-init tar file

> **Note:** In the example, `cloud-init-0.7.4` was placed in the `/opt` directory. You can change the directory if needed.

4. Build `cloud-init` using the commands shown in Figure 4-42.

```
sles11a:/tmp # cd /opt/cloud-init-0.7.4
sles11a:/opt/cloud-init-0.7.4 # python setup.py build
```

Figure 4-42   Building cloud-init

5. Establish a network connection to the following address (required for cloud-init installation):

   `https://pypi.python.org/simple/`

6. Install cloud-init. If your target system has successfully connected to the website, issue the commands shown in Figure 4-43.

```
sles11a:/tmp # cd /opt/cloud-init-0.7.4
sles11a:/opt/cloud-init-0.7.4 # python setup.py install
```

Figure 4-43 Installing cloud-init

If the connection is not valid, the command fails and displays the error message shown in Figure 4-44.

```
Installed /usr/local/lib64/python2.6/site-packages/cloud_init-0.7.4-py2.6.egg
Processing dependencies for cloud-init==0.7.4
Searching for requests
Reading https://pypi.python.org/simple/requests/
Download error on https://pypi.python.org/simple/requests/: [Errno -2] Name or
service not known -- Some packages may not be found!
Couldn't find index page for 'requests' (maybe misspelled?)
Scanning index of all packages (this may take a while)
Reading https://pypi.python.org/simple/
Download error on https://pypi.python.org/simple/: [Errno -2] Name or service not
known -- Some packages may not be found!
No local packages or download links found for requests
error: Could not find suitable distribution for Requirement.parse('requests')
```

Figure 4-44 Sample error messages for installing cloud-init

To circumvent the problem, set up a local repository of the website. In the example, a directory called `/tmp/simple` was set as the local repository. If you choose another directory to do it, change the commands accordingly.

Complete these steps:

a. Set up the directory `/tmp/simple` as the local repository by issuing these commands.

```
sles11a:/tmp # mkdir /tmp/simple
```

b. Go to this website:

```
https://pypi.python.org/simple/
```

c. Download the latest packages for the folders shown in Figure 4-45, which shows the sample folder structure after downloading:

```
sles11a:/tmp/simple # pwd
/tmp/simple
sles11a:/tmp/simple # ls
argparse  cheetah    jsonpatch    Markdown  PrettyTable  pyyaml    six
boto      configobj  jsonpointer  oauth     pyserial     requests
sles11a:/tmp/simple # cd requests/
sles11a:/tmp/simple/requests # ls
requests-2.7.0.tar.gz
```

Figure 4-45 Sample folder structure of local repository

d. Update the script **easy_install.py** with the following changes. The script can be found in `/opt/setuptools-17.0/setuptools/command/easy_install.py`).

From:

```
self.index_url = None
```

To:

```
self.index_url = '/tmp/simple'
```

e. Reinstall the `setuptools` package as shown in Figure 4-46.

```
sles11a:/tmp # cd /opt/setuptools-17.0
sles11a:/opt/setuptools-17.0 # python setup.py install
```

Figure 4-46 Installing setuptools

f. Install cloud-init by issuing the commands shown in Figure 4-47.

```
sles11a:/tmp # cd /opt/cloud-init-0.7.4
sles11a:/opt/cloud-init-0.7.4 # python setup.py install
```

Figure 4-47 Installing cloud-init

Note: You might also encounter the following error messages:

```
Installed /usr/local/lib64/python2.6/site-packages/jsonpatch-1.11-py2.6.egg
Searching for configobj
Couldn't find index page for 'configobj' (maybe misspelled?)
Scanning index of all packages (this may take a while)
No local packages or download links found for configobj
error: Could not find suitable distribution for
Requirement.parse('configobj')
```

If this occurs, issue the following command to resolve the issue:

```
easy_install /tmp/simple/configobj/configobj-5.0.6.tar_.gz
```

Next, issue the following commands again:

```
sles11a:/tmp # cd /opt/cloud-init-0.7.4
sles11a:/opt/cloud-init-0.7.4 # python setup.py install
```

7. Copy and update the `cloud-init-related` services to the `/etc/init.d` directory (see Figure 4-48).

```
sles11a:/tmp # cp /opt/cloud-init-0.7.4/sysvinit/redhat/* /etc/init.d
sles11a:/tmp # cd /etc/init.d
```

Figure 4-48 Copying cloud-init related services to /etc/init.d

Four scripts, `cloud-init-local`, `cloud-init`, `cloud-config`, and `cloud-final` are added to `/etc/init.d`. Modify each of them by replacing the variable as follows:

Relace this:

```
cloud_init="/usr/bin/cloud-init"
```

With this:

```
cloud_init="/usr/local/bin/cloud-init"
```

8. Update the cloud-init script `DataSourceConfigDrive.py` for the SLES image.

   At the time of writing, for some levels of SLES, cloud-init does not perform the customization indicated by the `user_data` input.To circumvent this problem, edit the file `/usr/local/lib64/python2.6/site-packages/cloud_init-0.7.4-py2.6.egg/cloudinit/sources/DataSourceConfigDrive.py` using the following command. Comment out the lines that are indicated in bold in Example 4-7.

   `vi /usr/local/lib64/python2.6/site-packages/cloud_init-0.7.4-py2.6.egg/cloudinit/sources/DataSourceConfigDrive.py`

   Example 4-7 Sample changes in DataSourceConfigDrive.py file

```
# we want to do some things (writing files and network config)
        # only on first boot, and even then, we want to do so in the
        # local datasource (so they happen earlier) even if the configured
        # dsmode is 'net' or 'pass'. To do this, we check the previous
        # instance-id
        prev_iid = get_previous_iid(self.paths)
        cur_iid = md['instance-id']
        #if prev_iid != cur_iid and self.dsmode == "local":
        # self.helper.on_first_boot(results)
```

9. Update the file `/etc/init.d/cloud-init-local` to ensure that it starts after the `xcatconf4z` service.

   Change from this:

```
sles11a:/etc/init.d # vi /etc/init.d/cloud-init-local
change the # Required-Start line in the ### BEGIN INIT INFO section from:
### BEGIN INIT INFO
# Provides: cloud-init-local
# Required-Start: $local_fs $remote_fs
# Should-Start: $time
# Required-Stop:
```

   To this:

```
### BEGIN INIT INFO
# Provides: cloud-init-local
# Required-Start: $local_fs $remote_fs xcatconf4z
# Should-Start: $time
# Required-Stop:
```

10. Tailor the default configuration file (`/etc/cloud/cloud.cfg`) for SLES (the default is `ubuntu`) by completing these steps:

    a. Replace `distro:ubuntu` with `distro:sles` (at approximately line 81).

    b. Change the default user name, password, and gecos as needed (at approximately lines 84 - 86).

c. Change the groups tag (at approximately line 87) so it appears similar to these groups:

```
[adm, audio, cdrom, dialout, dip, floppy, video]
```

See Example 4-8.

Example 4-8 Sample changes in cloud.cfg configuration file

```
79 system_info:
 80    # This will affect which distro class gets used
 81    distro: sles
 82    # Default user name + that default users groups (if added/used)
 83    default_user:
 84      name: sles
 85      lock_passwd: True
 86      gecos: sles
 87      groups: [adm, audio, cdrom, dialout, dip, floppy, video]
 88      sudo: ["ALL=(ALL) NOPASSWD:ALL"]
 89      shell: /bin/bash
```

11.The `cloud-init` process tries to add the user `syslog` to the `group` adm, but SLES does not have this user and group by default. Therefore, you must issue the following commands:

```
useradd syslog
groupadd adm
```

12.Start the `cloud-init-related` services with the following commands. You can ignore the error `insserv: Service network is missed in the runlevels 4 to use service cloud-init` if it occurs.

```
insserv cloud-init-local
insserv cloud-init
insserv cloud-config
insserv cloud-final
```

The services in `/etc/init.d/rcX.d` are now as you expect them to be. Make sure the `xcatconf4z` script starts before any other cloud-init service.

```
sles11a:/etc/init.d # ls -l /etc/rc.d/rc3.d | grep -e xcat -e cloud
lrwxrwxrwx 1 root root 13 Jun  5 15:03 S04xcatconf4z -> ../xcatconf4z
lrwxrwxrwx 1 root root 19 Jun  5 16:01 S09cloud-init-local ->
../cloud-init-local
lrwxrwxrwx 1 root root 13 Jun  5 16:01 S10cloud-init -> ../cloud-init
lrwxrwxrwx 1 root root 15 Jun  5 16:01 S11cloud-config -> ../cloud-config
lrwxrwxrwx 1 root root 14 Jun  5 16:01 S12cloud-final -> ../cloud-final
```

13.Verify `cloud-init` configuration by issuing this command:

```
cloud-init init --local
```

Make sure that no errors occur. Several warning messages can be ignored:

```
/usr/local/lib64/python2.6/site-packages/Cheetah-2.4.4-py2.6.egg/Cheetah/Compil
er.py:1509: UserWarning:
You don't have the C version of NameMapper installed! I'm disabling Cheetah's
useStackFrames option as it is painfully slow with the Python version of
NameMapper. You should get a copy of Cheetah with the compiled C version of
NameMapper.
  "\nYou don't have the C version of NameMapper installed! "
```

14.Finally, issue the command **rm -rf /var/lib/cloud** so that cloud-init works after restart.

4.3.3 Capture the node to generate the image in the xCAT MN

This section discusses capturing a Linux node to generate an image in xCAT MN.

Start the xCAT GUI and authenticate into xCAT. Then, go to the Script window for the xCAT MN node (`xcat`) and issue the **`imgcapture`** command to capture the node's root disk, and then click **Run**. The command is as follows:

```
/opt/xcat/bin/imgcapture rhel65a --profile rhel65a
```

Where:

- `rhel65a` is the node name. `rhel65a` is the profile name that you want to store the captured image in xCAT.

See Figure 4-49.

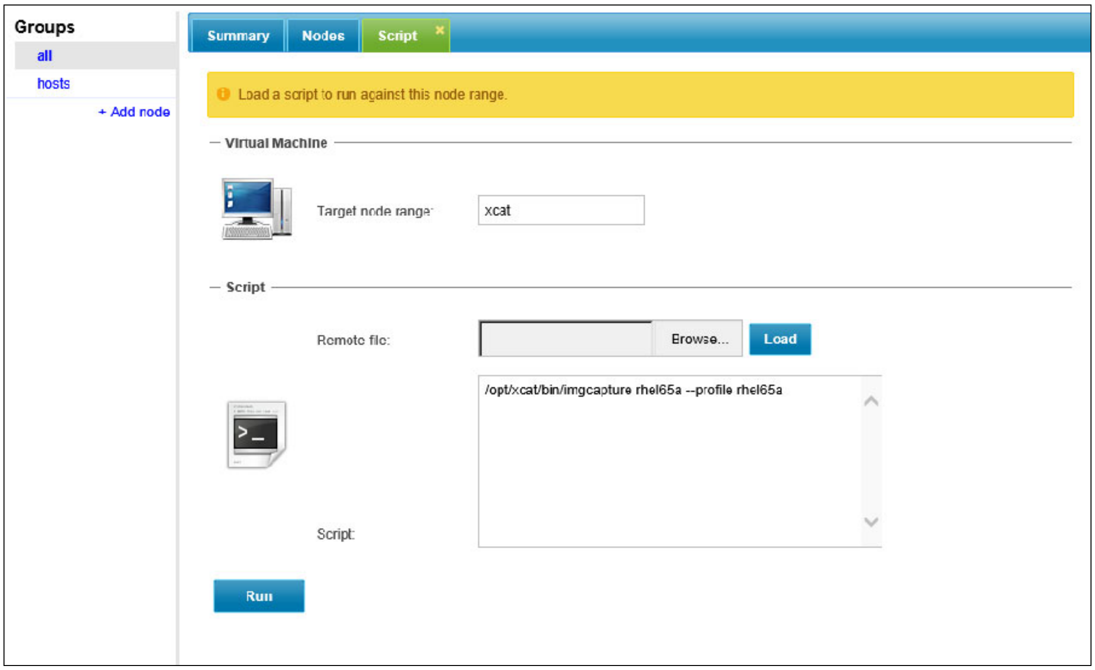


Figure 4-49 xCAT image capture example

The final result is shown in Figure 4-50.

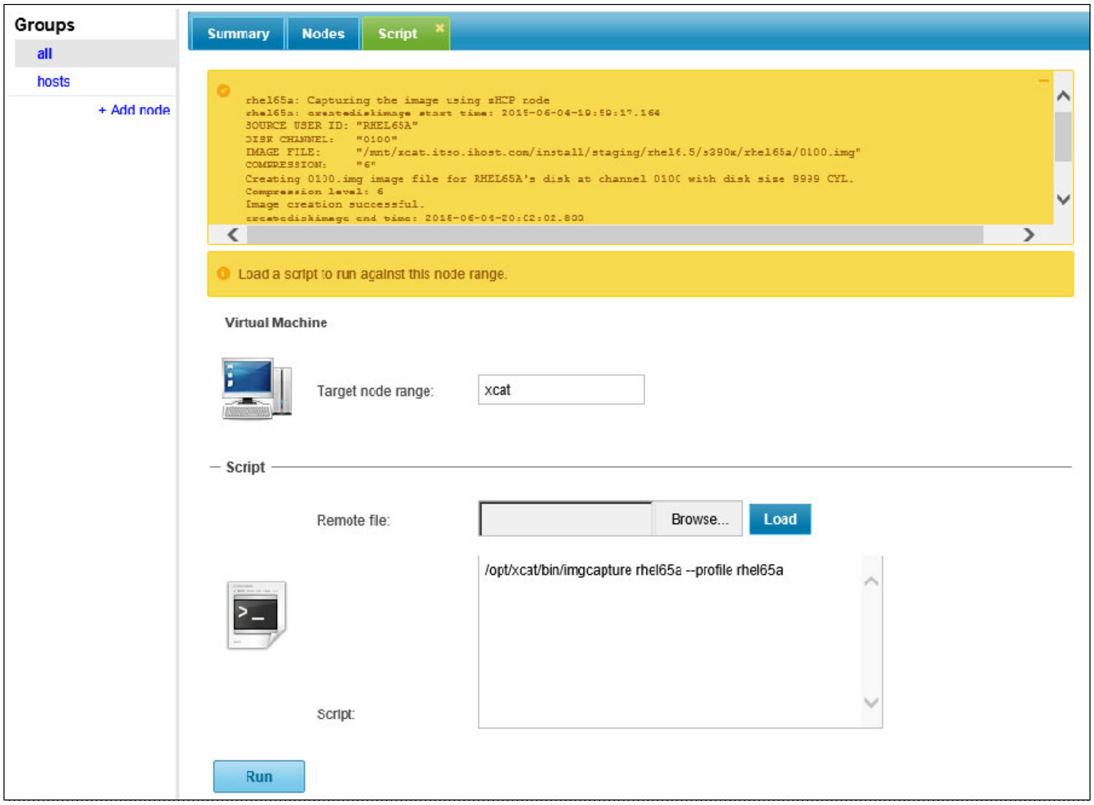


Figure 4-50   xCAT image capture sample output

If the image was successfully created, by default the image file is stored at `/install/netboot/rhel6.5/s390x/rhel65a/0100.img`.

> **Note:** As part of the capture process, the virtual server must be shut down before capturing the image.

4.3.4 Define the image to Glance

Before you can deploy a Linux virtual machine from the image that was generated in 4.3.3, “Capture the node to generate the image in the xCAT MN” on page 107, the image must be defined in the Glance image repository.

By default, the captured image is stored in the `/install` directory. In the example case, it is in `/install/netboot/rhel6.5/s390x/rhel65a/0100.img`.

Perform the following steps to define the captured image into OpenStack Glance:

1. Locate the image file.
2. Log on to xCAT and the controller node with the `mnadmin` user ID and define the image to Glance with this command:

```
glance image-create --name rhel65a --disk-format=raw --container-format=bare
--is-public=True < /install/netboot/rhel6.5/s390x/rhel65a/0100.img
```

Where:

– `rhel65a` is the name of the image as known to Glance.
– `/install/netboot/rhel6.5/s390x/rhel65a/0100.img` is the file specification of the image file.

> **Note:** Image names should be restricted to the UTF-8 subset, which corresponds to the ASCII character set. In addition, special characters such as /, \, $, %, @ should not be used.

Figure 4-51 is an example of the captured image file for RHEL 6.5.

```
[mnadmin@xcat ~] $ source openrc
[mnadmin@xcat ~] $ glance image-create --name rhel65a --disk-format=raw
--container-format=bare --is-public=True <
/install/netboot/rhel6.5/s390x/rhel65a/0100.img
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| checksum         | 97f18fb8552b8b51adcf920f71aa733b     |
| container_format | bare                                 |
| created_at       | 2015-06-05T20:50:35                  |
| deleted          | False                                |
| deleted_at       | None                                 |
| disk_format      | raw                                  |
| id               | 2765fd5a-1934-40ba-be42-2fb75915d9ea |
| is_public        | True                                 |
| min_disk         | 0                                    |
| min_ram          | 0                                    |
| name             | rhel65a                              |
| owner            | 7ccb50cd927e4d87bf1f56d033a777a6     |
| protected        | False                                |
| size             | 646184823                            |
| status           | active                               |
| updated_at       | 2015-06-05T20:50:41                  |
| virtual_size     | None                                 |
+------------------+--------------------------------------+
```

Figure 4-51 Defining captured image to Glance

Take a note of the ID, shown in Figure 4-51 in bold, that will be used in next step.

3. Update the image properties for the image generated in Glance in the previous step. Use the following command:

```
glance image-update --property image_type_xcat=linux --property
architecture=s390x --property os_name=Linux --property os_version=rhel6.5
--property provisioning_method=netboot --property image_file_name=0100.img uuid
```

Where:

– `rhel6.5` is the operating system version of your capture node.
– `uuid` is the ID value that wass generated in step 2.

Figure 4-52 shows the updated image properties for Glance.

> **Note:** At the time of writing, only Red Hat- and SUSE-type images are supported. For a Red Hat image, you can specify the OS version as `rhelx.y`, `redhatx.y`, or `red hatx.y`, where `x.y` is the release number. For a SUSE type image, you can specify the OS version as `slesx.y` or `susex.y`, where `x.y` is the release number.

```
[mnadmin@xcat ~] $ glance image-update --property image_type_xcat=linux
--property architecture=s390x --property os_name=Linux --property
os_version=os_version --property provisioning_method=netboot --property
image_file_name=0100.img 2765fd5a-1934-40ba-be42-2fb75915d9ea
+--------------------------------+--------------------------------------+
| Property                       | Value                                |
+--------------------------------+--------------------------------------+
| Property 'architecture'        | s390x                                |
| Property 'image_file_name'     | 0100.img                             |
| Property 'image_type_xcat'     | linux                                |
| Property 'os_name'             | Linux                                |
| Property 'os_version'          | os_version                           |
| Property 'provisioning_method' | netboot                              |
| checksum                       | 97f18fb8552b8b51adcf920f71aa733b     |
| container_format               | bare                                 |
| created_at                     | 2015-06-05T20:50:35                  |
| deleted                        | False                                |
| deleted_at                     | None                                 |
| disk_format                    | raw                                  |
| id                             | 2765fd5a-1934-40ba-be42-2fb75915d9ea |
| is_public                      | True                                 |
| min_disk                       | 0                                    |
| min_ram                        | 0                                    |
| name                           | rhel65a                              |
| owner                          | 7ccb50cd927e4d87bf1f56d033a777a6     |
| protected                      | False                                |
| size                           | 646184823                            |
| status                         | active                               |
| updated_at                     | 2015-06-05T20:54:40                  |
| virtual_size                   | None                                 |
+--------------------------------+--------------------------------------+
```

Figure 4-52 Updating image properties in Glance

If the command was run successfully, you see the image that you created as shown in Figure 4-53.

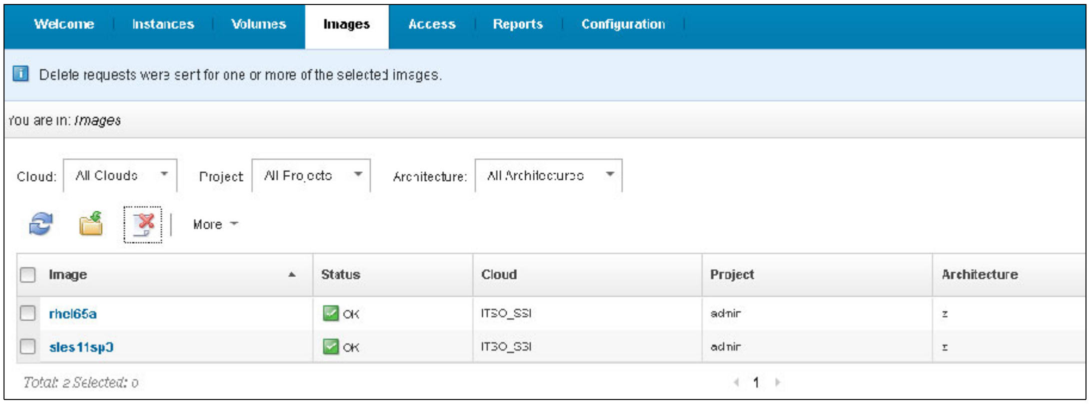


Figure 4-53 Image list in Glance

4.4 Configure email notification

To receive email notifications when various actions are taken, you must configure the notification properties in the IBM Cloud Manager with OpenStack for z Systems home directory.

The following user and administrator events can be configured to trigger an email notification:

- When an account is created or deleted, reaches its balance threshold, is delinquent, receives a payment, or has a bill created
- When instances are deployed, fail, expire, or have snapshots completed
- Project access is requested or granted
- A user is created, requests access, or has a password reset
- A virtual machine is backed up, restored, has a snapshot created, or reverts to a snapshot
- Volumes are created, deleted, detached, or resized

To set up email notifications using the IBM Cloud Manager with OpenStack self-service portal, complete these configuration steps:

1. Log on to xCAT and the controller node with the `mnadmin` user ID. Open the email configuration file (the default file name is `/var/opt/ibm/.SCE42/email.properties`) and issue the following command to change the parameters as shown in Example 4-9:

   [mnadmin@xcat ~] $ **sudo vi /var/opt/ibm/.SCE42/email.properties**

   Example 4-9 Sample email configuration file

```
# Licensed Materials - Property of IBM
#
# (C) Copyright IBM Corp. 2011, 2013 All Rights Reserved
#
# US Government Users Restricted Rights - Use, duplicate or
# disclosure restricted by GSA ADP Schedule Contract with
# IBM Corp.
# email.properties used to configure email capabilities.
```

```
# The relay host to use for outgoing SMTP emails. This
# must be specified in order for emailing to work as there
# is no sensible default.
com.ibm.cfs.email.relay.host=192.168.60.7

# The prefix to use by default for emails. If this
# is not set then email subjects will not have a default
# prefix in their subject.
com.ibm.cfs.email.subject.prefix=icm

# The default 'from address' for outgoing emails.
# If this is not set the system will use a sensible default
# such as the CFS admin username @ fully qualified hostname.
com.ibm.cfs.email.from.address=admin@itso.ibm.com

# The default sender for outgoing emails.
# The default value is "Cloud Foundation Admin"
com.ibm.cfs.email.from.name=ICM Admin  Mail

# If email notifications are enabled by default for users.
# If set to true then by default users will be setup
# to receive email notifications by default, if false they will not.
# The default value is true.
com.ibm.cfs.email.default.notifications=true
```

Where:

- `com.ibm.cfs.email.relay.host` is the host name of the relay host that the self-service portal uses for outgoing SMTP emails.
- `com.ibm.cfs.email.subject.prefix` is the subject prefix for all self-service portal emails.
- `com.ibm.cfs.email.from.name` is the email `From` name you can see when you receive the email.
- `com.ibm.cfs.email.default.notifications` is set to `true` if you want to enable email notifications.

> **Note:** The administrator can disable individual users email notifications through the self-service portal (see “User management” on page 140).

2. Restart the portal service to enable email notifications by issuing this command:

```
[mnadmin@xcat ~] $ sudo service sce restart
```

4.5 Configure metering

The IBM Cloud Manager with OpenStack self-service portal has a configurable metering framework that enables the portal to record and present metering data. The metering function is disabled by default, so you must enable it to use it.

To enable metering with the self-service portal, complete the following configuration steps:

1. Log on to xCAT and the controller node using the `mnadmin` user ID. In the metering configuration file (the default file name is `/var/opt/ibm/.SCE42/metering.properties`), issue the following command to change the parameters as shown in Example 4-10:

   `[mnadmin@xcat ~] $` **`sudo vi /var/opt/ibm/.SCE42/metering.properties`**

   Example 4-10 Sample metering configuration file

```
# Licensed Materials - Property of IBM
#
# (C) Copyright IBM Corp. 2011, 2013 All Rights Reserved
#
# US Government Users Restricted Rights - Use, duplicate or
# disclosure restricted by GSA ADP Schedule Contract with
# IBM Corp.
# Enable metering for Cloud resources upon deployment
com.ibm.cfs.metering.enabled=true

# The integer number of minutes between each metering time.
# If not specified or value is invalid, the default value will be 1440(1 day).
com.ibm.cfs.metering.interval=10

# The path of the folder where the metrics data file is saved, for the relative
path, the root path will be the installation path
# If not specified or value is invalid, the data file will be saved into
{skc_home}/metricsdata/ .
#com.ibm.cfs.metering.data.path=C:/metricsdata/
# The number of hours between each metering data export. The first exporting
will happen at integral point.
# If not specified or value is invalid, the default value will be 1.
#com.ibm.cfs.metering.data.interval=1
# The number of days the metering data will be expired in the database.
# In other words, the metering data will not be save more than this number of
days.
# If not specified or value is invalid, this number will be 370.
#com.ibm.cfs.metering.data.expired.days=370
```

   Where:

   - `com.ibm.cfs.metering.enabled` is set to `true` to enable metering.
   - `com.ibm.cfs.metering.interval` is the time in minutes between each metering record synchronization.

2. Restart the portal service to enable the metering function by issuing this command:

   `[mnadmin@xcat ~] $` **`sudo service sce restart`**

For more information about using metering on the self-service portal, see “Account management” on page 134.

4.6 Configure billing

The IBM Cloud Manager with OpenStack self-service portal has a configurable billing and accounting interface. The interface allows the portal to monitor resource use and create the bills needed to charge for it.

Note: Before enabling the billing function, you must enable metering. See “Configure metering” on page 112.

To enable billing functions, complete these configuration steps:

1. Log on to xCat and the controller node using the `mnadmin` user ID. In the billing configuration file (the default file name is `/var/opt/ibm/.SCE42/billing.properties`), use the following command to change the parameters as shown in Example 4-11:

   [mnadmin@xcat ~] $ **sudo vi /var/opt/ibm/.SCE42/billing.properties**

   Example 4-11 Sample billing configuration file

```
# Licensed Materials - Property of IBM
#
# (C) Copyright IBM Corp. 2011, 2013 All Rights Reserved
#
# US Government Users Restricted Rights - Use, duplicate or
# disclosure restricted by GSA ADP Schedule Contract with
# IBM Corp.
# Enable charging for Cloud resources upon deployment
# If not set the default is true
com.ibm.cfs.billing.enabled=true

# Delinquency policy. Default polices provided by the framework, with their
corresponding IDs:
# Ignore delinquent accounts ->
com.ibm.cfs.services.billing.policies.do.nothing
# Destroy all deployments and detach all volumes upon delinquent account ->
com.ibm.cfs.services.billing.policies.destroy
# Destroy all deployments and delete all volumes upon delinquent account ->
com.ibm.cfs.services.billing.policies.destroydelete
# Suspend all deployments upon delinquent account ->
com.ibm.cfs.services.billing.policies.shutdown
#Suspend all deployments and detach all volumes upon delinquent account ->
com.ibm.cfs.services.billing.policies.shutdowndetach
#Suspend all deployments and delete all volumes upon delinquent account ->
com.ibm.cfs.services.billing.policies.shutdowndelete
#Detach all volumes upon delinquent account ->
com.ibm.cfs.services.billing.policies.detach
#Delete all volumes upon delinquent account ->
com.ibm.cfs.services.billing.policies.delete

# If not set the default is com.ibm.cfs.services.billing.policies.shutdown
com.ibm.cfs.billing.delinquency.policy=com.ibm.cfs.services.billing.policies.sh
utdown

# The number of seconds between delinquent account finder job executions.
# If not set or is negative the default is 60 seconds.
com.ibm.cfs.billing.delinquency.finder.interval=60
```

```
# The number of hours between account balance threshold finder job executions.
# If not set or is negative the default is 24 hours.
com.ibm.cfs.billing.account.balance.threshold.interval=24
```

Where:

- `com.ibm.cfs.billing.enabled` is defined as `true` to enable the billing and accounting functions in IBM Cloud Manager with OpenStack.
- `com.ibm.cfs.billing.delinquency.policy` determines the action that IBM Cloud Manager with OpenStack takes against existing instances and volumes when an account becomes delinquent.
- `com.ibm.cfs.billing.delinquency.finder.interval` is the property that represents the number of seconds to wait before running a job that examines each account to determine whether the account is delinquent.
- `com.ibm.cfs.billing.account.balance.threshold.interval` is the property that represents the number of hours to wait before running a job to find accounts that are at their account balance threshold.

2. Set the price for CPU, Memory, Disk and Volumes, and Snapshots.

   IBM Cloud Manager with OpenStack can produce charges that are billed back to users of a specific cloud resource, such as an instance. But first you need to set the price for CPU, Memory, Disk, and Volumes.

   a. To configure processor price, issue this command:

   `[mnadmin@xcat ~] $ sudo vi /var/opt/ibm/.SCE42/products/Openstack_CPU.xml`

   As shown in Example 4-12, these propertiesspecify that the default collector collects charges on virtual machines by using the number of processors that are assigned to each machine. The example rate is $0.0167 per minute, which is about $1.00 (US) per hour.

   Example 4-12 Sample changes in Openstack_CPU.xml

   ```
   <pricing currency="USD" interval="60" price="0.0167"/>
   ```

   b. To configure memory price, issue this command:

   `[mnadmin@xcat ~] $ sudo vi /var/opt/ibm/.SCE42/products/Openstack_RAM.xml`

   As shown in Example 4-13, these propertiesspecify that the default collector collects charges on virtual machines based on the number of bytes of memory that are assigned to each machine. The example rate is $0.00001627604167 per MB per minute, which is about $1.00 (US) per hour per GB.

   Example 4-13 Sample changes in Openstack_RAM.xml

   ```
   <pricing currency="USD" interval="60" price="1.627604167E-5"/>
   ```

   c. To configure disk price, issue this command:

   `[mnadmin@xcat ~] $ sudo vi /var/opt/ibm/.SCE42/products/Openstack_Disk.xml`

   As shown in Example 4-14, these properties specify that the default collector collects charges on virtual machines based on the number of disks that are assigned to them.

   Example 4-14 Sample changes in Openstack_Disk.xml

   ```
   <pricing currency="USD" interval="60" price="1.627604167E-5"/>
   ```

d. To configure volume price, issue this command:

```
[mnadmin@xcat ~] $ sudo vi /var/opt/ibm/.SCE42/products/Openstack_Volume.xml
```

As shown in Example 4-15, these properties specify that the default collector collects charges on virtual machines based on the volumes that are assigned to them.

Example 4-15 Sample changes in Openstack_Volume.xml

```
<pricing currency="USD" interval="60" price="1.627604167E-5"/>
```

e. To configure Snapshot price, issue this command:

```
[mnadmin@xcat ~] $ sudo vi
/var/opt/ibm/.SCE42/products/Openstack_Snapshot.xml
```

As shown in Example 4-16, these properties specify that the default collector collects charges on virtual machines based on the snapshots that are assigned to them.

Example 4-16 Sample changes in Openstack_Snapshot.xml

```
<pricing currency="USD" interval="60" price="1.627604167E-5"/>
```

Note: The currency for all configurable products must be consistent. For example, set US dollar (USD) for both or Chinese Yuan (CNY) for both. Using inconsistent currencies causes incorrect product charges.

3. Restart the portal service to enable account billing by issuing the following command:

```
[mnadmin@xcat ~] $ sudo service sce restart
```

After you configure account billing, you can manage account billing using the information provided in “Account management” on page 134.

5

Configuring, managing, and using the self-service portal

This chapter describes how to use the self-service portal of IBM Cloud Manager with OpenStack for z Systems to manage and operate within a z/VM environment.

IBM Cloud Manager with OpenStack for z Systems features a self-service portal for performing cloud operations. Using this self-service portal, you can perform the following public and private cloud operations:

- Providing access to multiple clouds from a single portal
- Starting and stopping instances
- Capturing instances
- Resizing existing instances
- Creating projects to give team-specific access to instances
- Providing network configurations, which set unique network properties to different instances
- Creating expiration policies to reduce abandoned instances
- Providing request and approval workflow support
- Monitoring of resource allocations and billing for services

In the previous chapters, you learned about constructing the cloud foundation and building out the cloud infrastructures. After your cloud environment is complete, you can use the self-service portal to manage it.

The self-service portal provides two predefined roles, which are administrator and user. The administrator is responsible for administering all resources in the cloud including configuring the cloud. The user is responsible for deploying images and managing instances.

This chapter includes the following sections:

- Self-service portal configurations
- Self-service portal administrator management
- Self-service portal user operations

5.1 Self-service portal configurations

After the installation of IBM Cloud Manager with OpenStack for z Systems, the self-service portal is ready to use. The web interface for self-service portal is available at the following address:

`https://IP:Port/cloud/web/login.html`

IP Is the `XCAT_MN_Addr` in the `DMSSICNF` file, as shown in Example 3-3 on page 58.

Port The default port is `18443`.

This section shows the following configuration steps for self-service portal administrator:

- Create a cloud environment in the self-service portal
- Create networks in the self-service portal
- Enable email notification in the self-service portal

5.1.1 Create a cloud environment in the self-service portal

Before you use the self-service portal of IBM Cloud Manager with OpenStack for z Systems, you must complete the cloud environment configurations. For each controller node, you need to define its cloud configuration. This section shows how to define the cloud configuration for a z/VM controller node.

A few common icons in the self-service portal are shown in Figure 5-1.

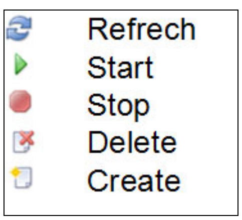


Figure 5-1 Common icons

Complete these steps to create the cloud environment:

1. Point a browser to your self-service portal, for example:

   `https://192.168.60.101:18443/cloud/web/login.html`

   The login page is displayed as shown in Figure 5-2.

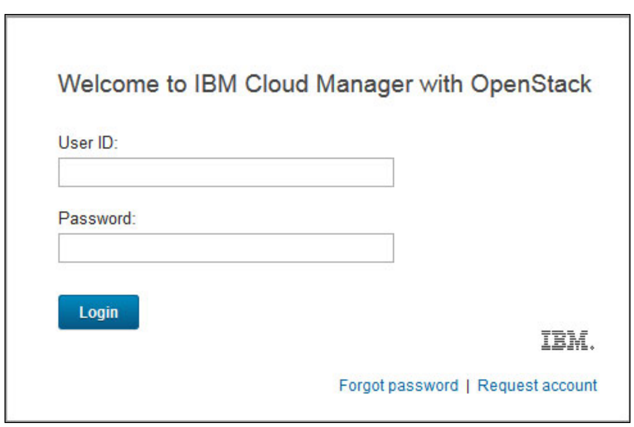


Figure 5-2 Login page

2. You will log in to the self-service portal as administrator. The User ID is `admin` and the password is the value of `cmo_admin_password` that you defined in the `DMSSICMO` file. See Example 3-4 on page 60.

   After login, you can see the Welcome tab as shown in Figure 5-3. Click **Configure the Cloud**.

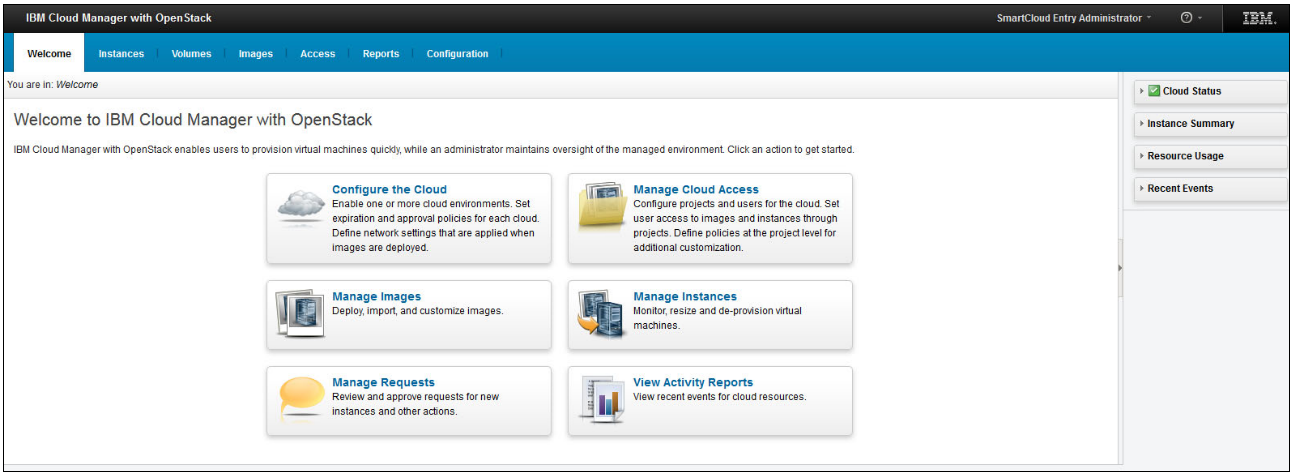


Figure 5-3 Administrator welcome page

3. Click the **Create** icon (see Figure 5-4).

Figure 5-4 Cloud status list

4. Define the cloud parameters as shown in Figure 5-5. After you enter all of the parameters, click **Test connection.**

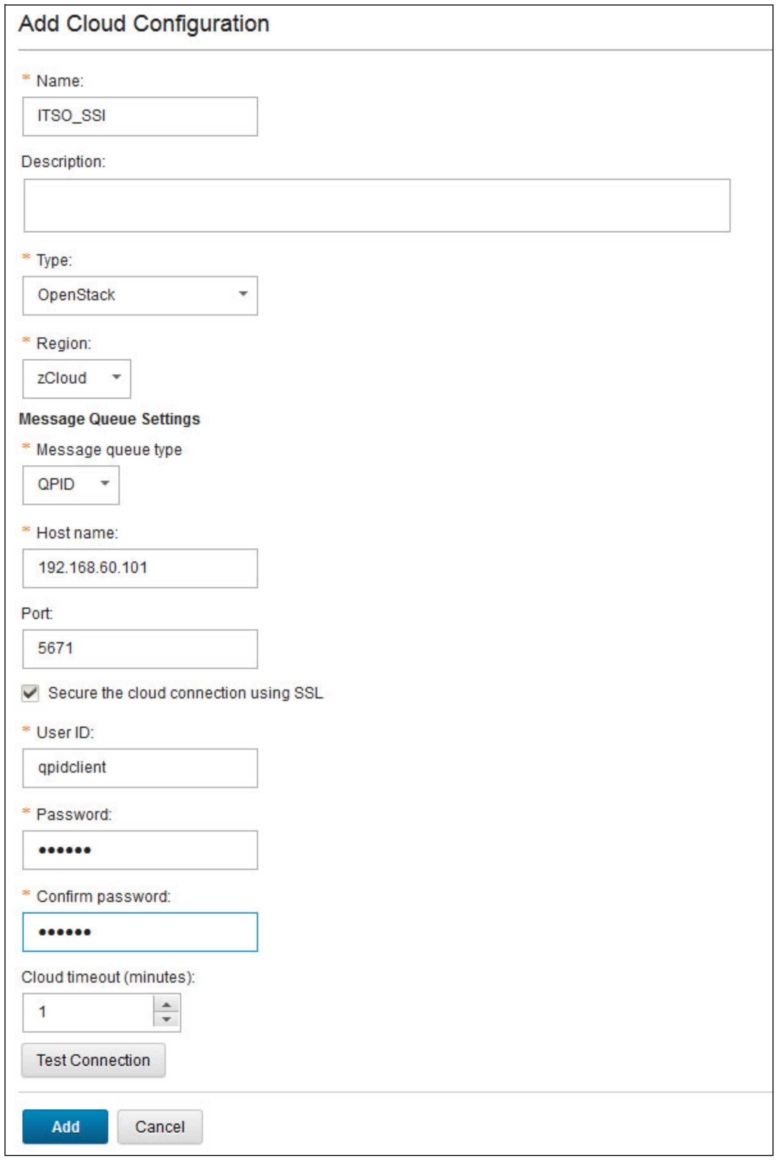


Figure 5-5 Cloud configurations

The meaning for the configuration options are as follows:

Name The name that you choose for this cloud environment

Description Add your own description in this field

Type Select **OpenStack**

Region Select **zCloud**

Message queue type Select **QPID**

Host name The value of `XCAT_MN_Addr` in the `DMSSICNF` file, as shown in Example 3-3 on page 58.

Secure the cloud connection using SSL Select this check box

Port Enter `5671`

User ID Enter `qpidclient`

Password The value of `cmo_admin_password` in the `DMSSICMO` file, as shown in Example 3-4 on page 60

Cloud timeout Increase if you have a slower network, otherwise use the default value

5. If the message shown in Figure 5-6 is displayed, the connection is successful and you can continue. Otherwise, check your configurations.

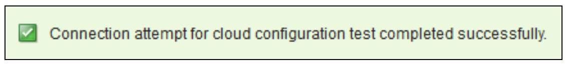


Figure 5-6 Successful message

6. Click **Add**.
7. The certificate information is displayed as shown in Figure 5-7. Click **Accept Certificate**.

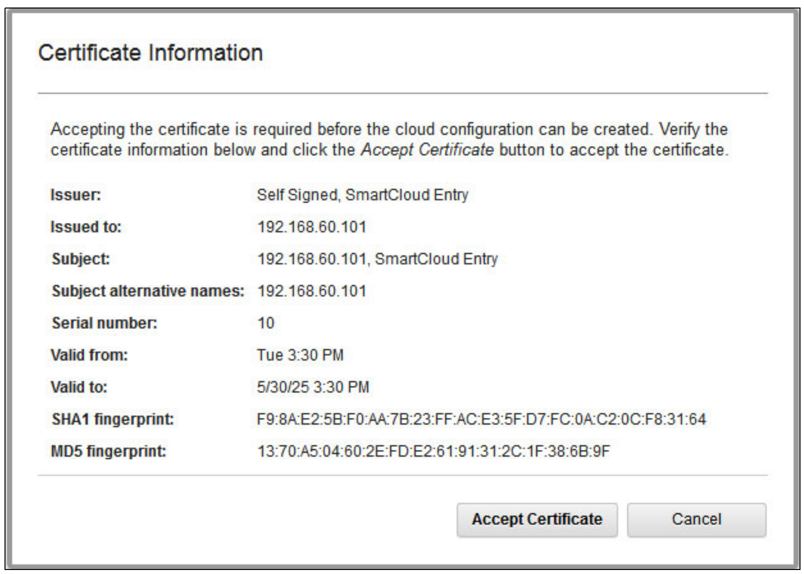


Figure 5-7 Certificate information

8. Add the cloud configuration as shown in Figure 5-8. Click the cloud name, which in this scenario is **ITSO_SSI**.

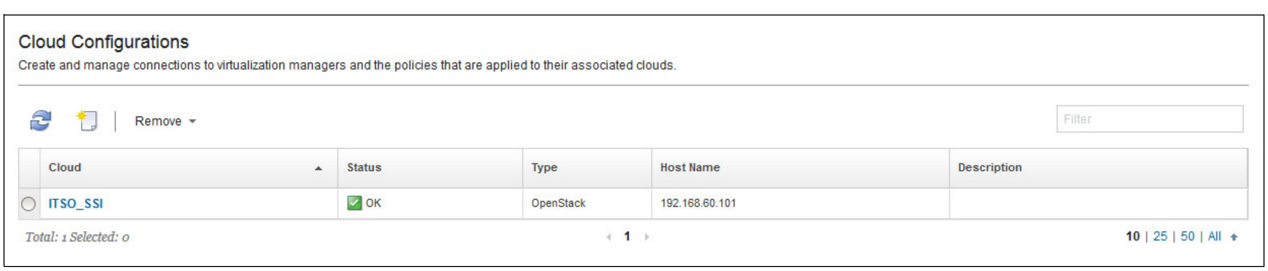


Figure 5-8 Cloud list

9. The properties of this cloud environment are displayed as shown in Figure 5-9. Click **Edit**.

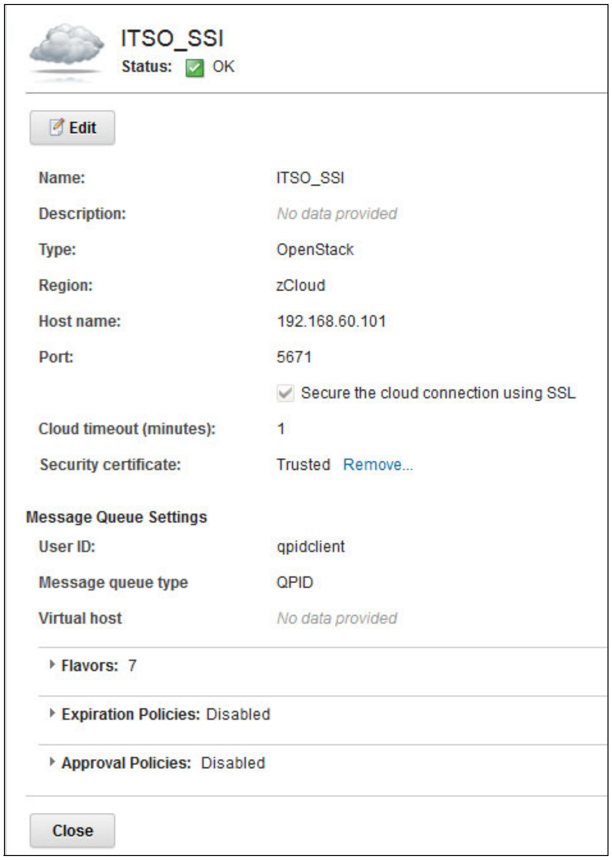


Figure 5-9 Cloud properties

10. Scroll down and expand the **Flavors** section as shown in Figure 5-10. You see the flavors that you have currently. To add your own flavor, click **Create**.

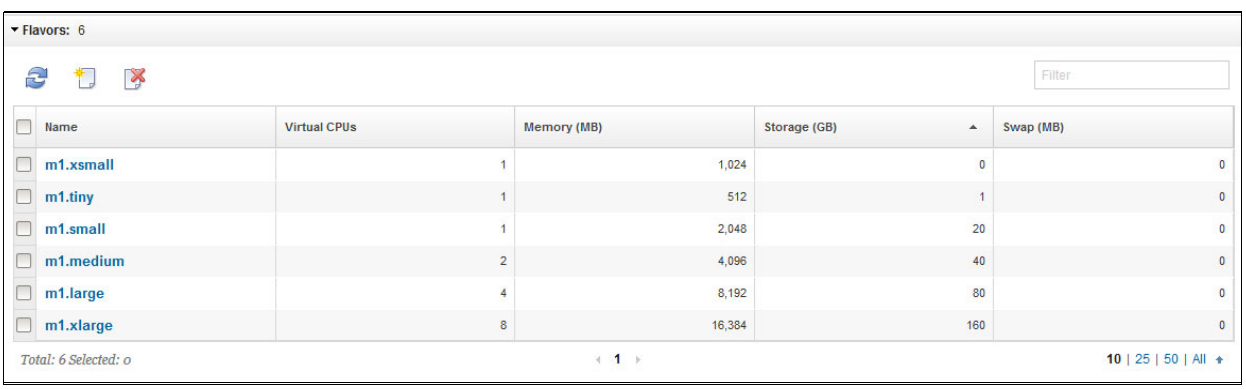


Figure 5-10 Flavors list

11. Input the parameters as shown in Figure 5-11. Click **Save**.

> **Note:** If you set the storage value to 0, it uses the image default storage size when deployed.

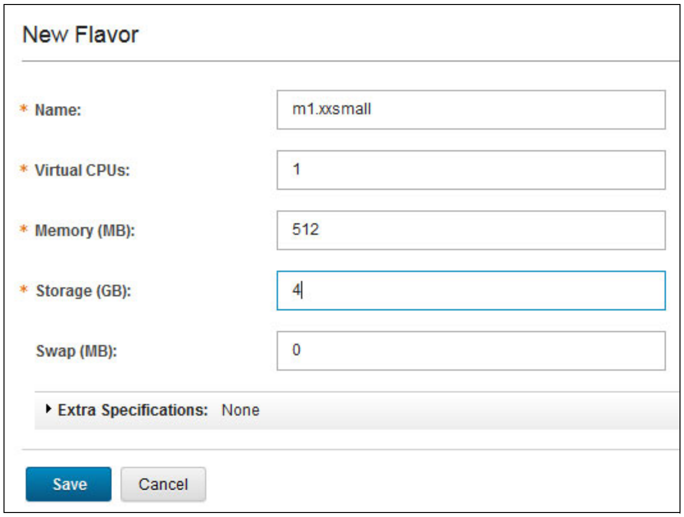


Figure 5-11 New flavor

12. Scroll down and expand the Expiration Policies section as shown in Figure 5-12. You can set your expiration policies in this window.

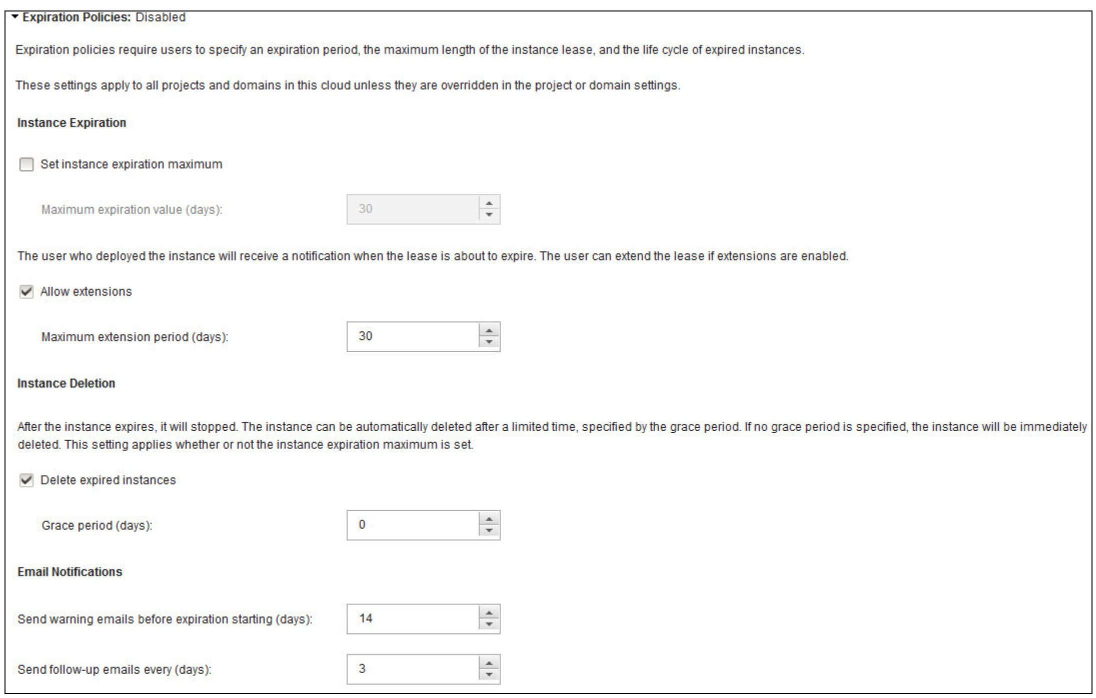


Figure 5-12 Expiration policies

13.Scroll down and expand the Approval Policies section as shown in Figure 5-13. You can set your approval policies in this window.

When you finish all configurations, click **Save**.

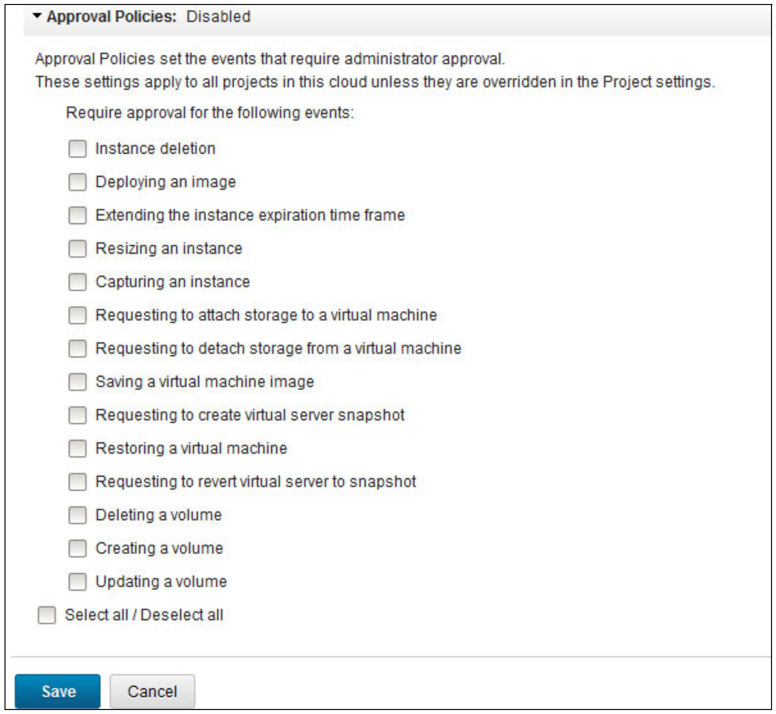


Figure 5-13 Approval policies

Now you have successfully added the z/VM controller node into the self-service portal.

5.1.2 Create networks in the self-service portal

Before you can deploy an instance, you need to finish the network configurations. This section shows how to configure a network for self-service portal.

You can create a *Flat* network as a management network, which allows xCAT to communicate with the instances. If you only want to use a single network, you can skip step 3 on page 126.

You can also create a *VLAN* network as a data network. The instance runs on both the management network and the data network as shown in Figure 2-1 on page 25. In this scenario, you need to follow all the steps in this section.

For more scenarios about VLAN and Flat networks, see “Configure neutron data network” on page 76.

Complete these steps to create both types of networks:

1. Click the Configuration tab and then click **Network** on the left navigation pane, as shown in Figure 5-14. Click **Create**.

Figure 5-14 Network status list

2. To create a Flat network as the management network, define the network configuration as shown in Figure 5-15, and click **Save**.

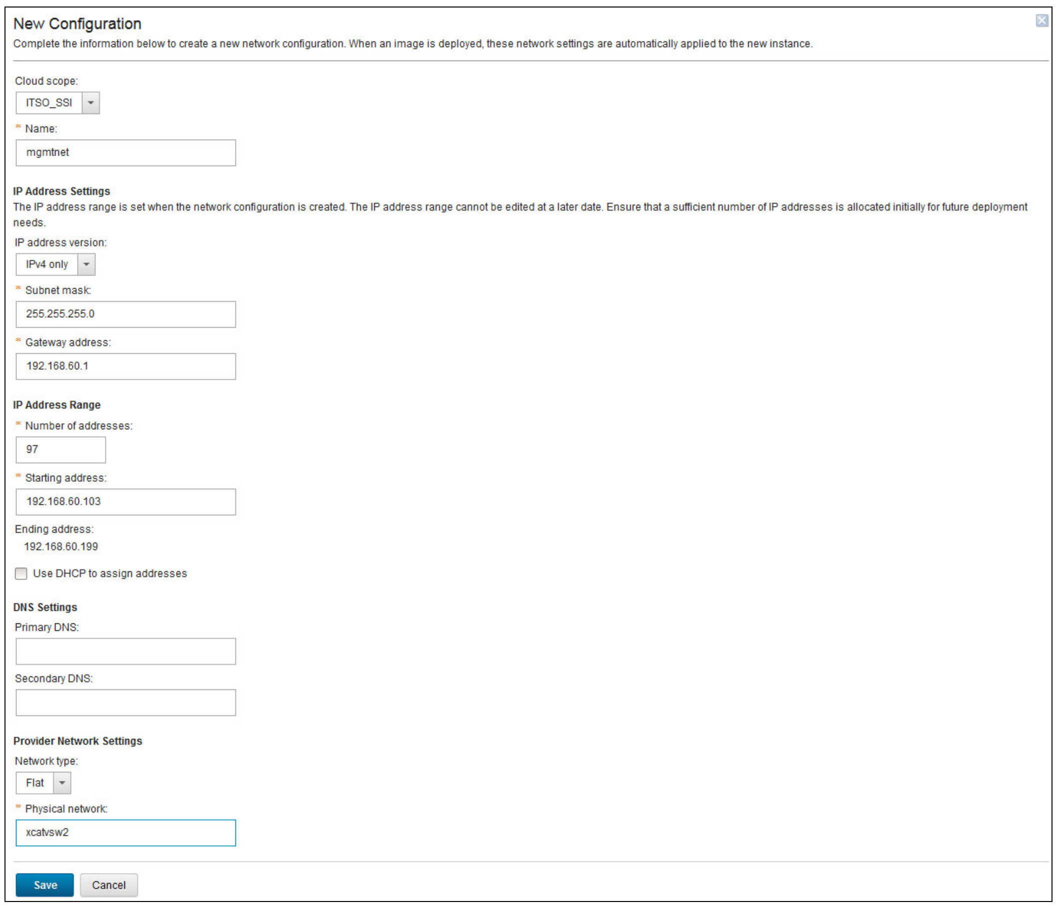


Figure 5-15 Management network

The following are the fields and definitions defined in Figure 5-15:

Cloud scope	Choose the z/VM Cloud environment
Name	Name your management network
IP address version	Select **IPV4 only**
Subnet mask	Use your subnet mask setting

Gateway address Use your gateway setting

Number of addresses The number of addresses in the IP pools

Starting address Set your first IP address in the IP pools

Use DHCP to assign address Do not select this check box

Primary DNS Use your primary DNS or leave it blank

Secondary DNS Use your secondary DNS or leave it blank

Network Type Select **Flat**

Physical network Use the value of `XCAT_MN_vswitch` in the `DMSSICNF` file

3. Repeat step 1 on page 125 to create a VLAN network as a data network. The parameters for a VLAN network are different. Repeat step 2 on page 125, but modify the following parameters:

   Network type Select **VLAN**

   Physical network Use the vswitch name that you set in the `ml2_conf` file (as shown in Example 4-1 on page 76)

   VLAN ID Use the VLAN you set in your physical switch

   Define the network configuration (see Figure 5-16), and click **Save**.

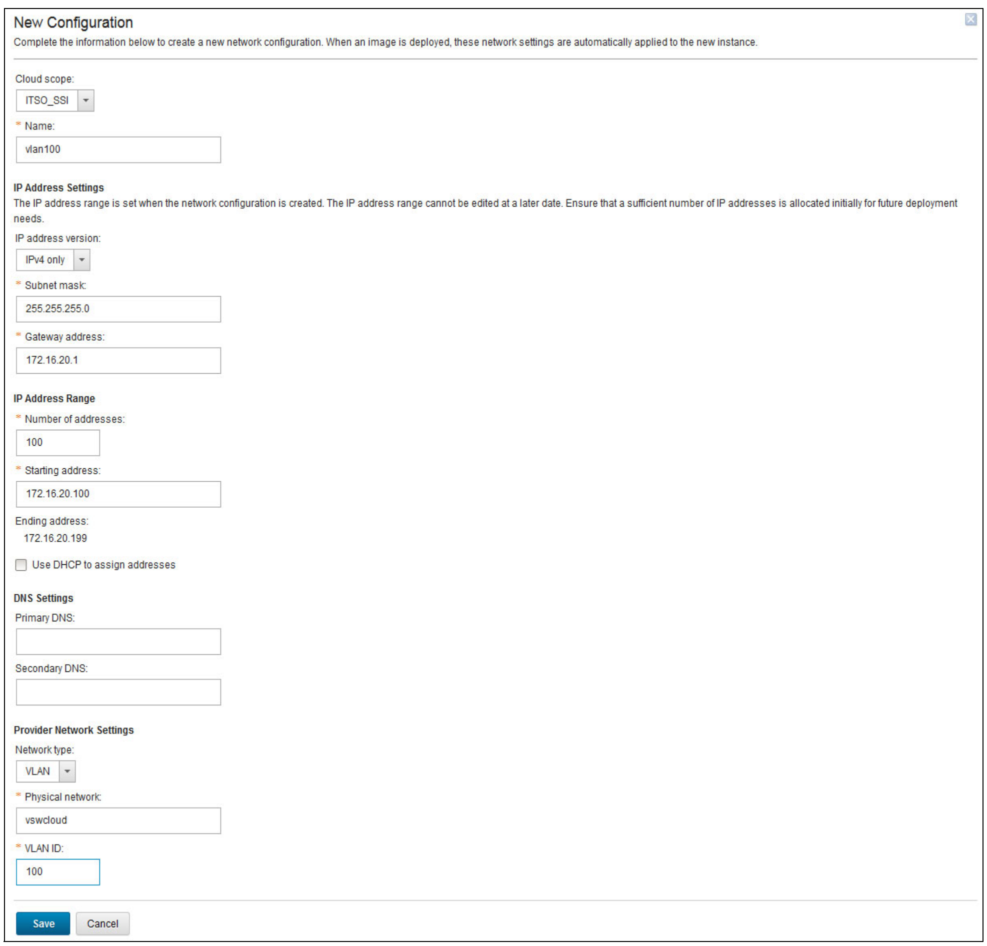


Figure 5-16 Data network

4. Now you have finished the network configuration, you can see the network list as shown in Figure 5-17. You can also see the status of IP pools by clicking **Manage IP Addresses**.

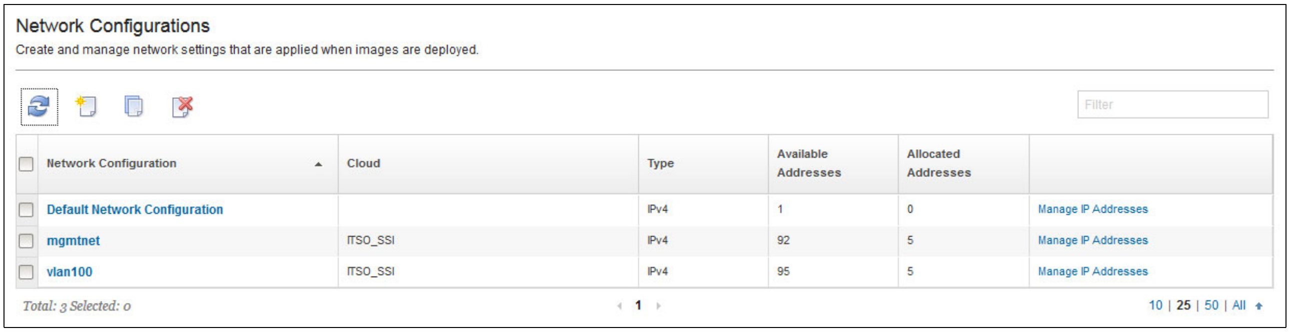


Figure 5-17 Network new status list

5.1.3 Enable email notification in the self-service portal

You configured the email notification in “Configure email notification” on page 111. This section describes how to set the email address of the administrator on the self-service portal.

To receive email notifications when various actions are taken, you also need to configure the notification properties in user preferences.

Follow these steps to enable email notification:

1. Log in as administrator, and click **Show user preferences** in the upper right banner of the web page as shown in Figure 5-18.

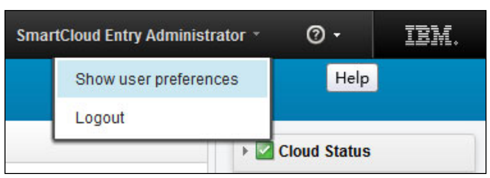


Figure 5-18 Show user preferences

2. In the User Profile (Figure 5-19), enter the email address of the administrator and select **Send notifications about instances and other events**. Click **Update**.

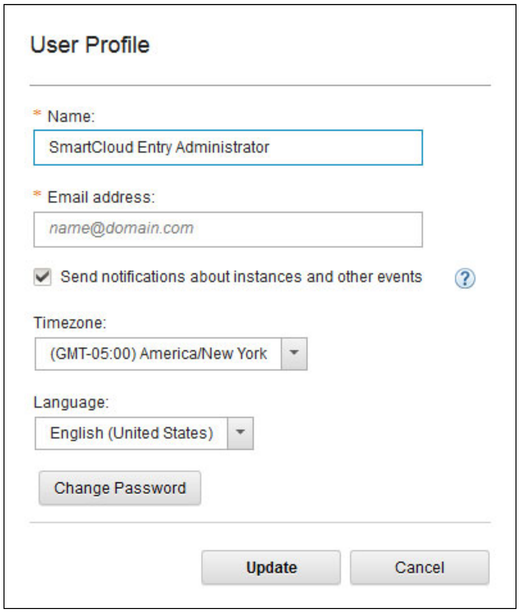


Figure 5-19 User profile

3. When user or administrator events trigger an email notification, the administrator receives an email like the one shown in Figure 5-20.

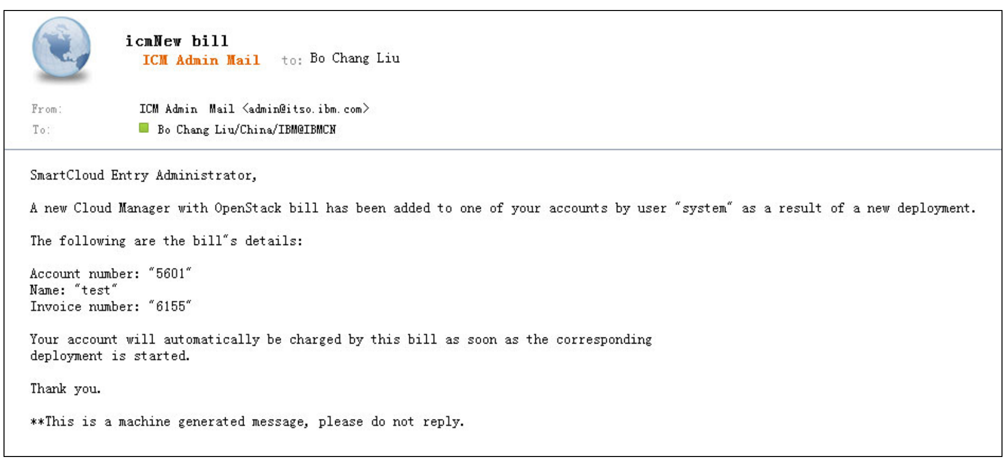


Figure 5-20 Email notifications example

5.2 Self-service portal administrator management

A user ID in IBM Cloud Manager with OpenStack for z Systems self-service portal must have the administrator role configured before it can perform tasks such as managing projects, images, instances, and requests.

This section shows the following management operations that are available to the administrator:

- Request management
- Project management
- Account management
- User management
- Image management
- Capacity management
- Instance migration
- Volume management

5.2.1 Request management

In “Create a cloud environment in the self-service portal” on page 118, you set the approval policies. When a user initiates an action that requires approval from an administrator, a request is created and submitted to the administrator for approval. The status of the request is set to Pending until the administrator handles the approval request.

If you have set the email notifications, you will receive an email (as shown previously in Figure 5-20 on page 128).

To approve, reject, or withdraw a request, complete these steps:

1. Log in as administrator, and click the Access tab. Click **Requests** on the left navigation pane. All the pending requests are listed here as shown in Figure 5-21. Click the request number to see the details.

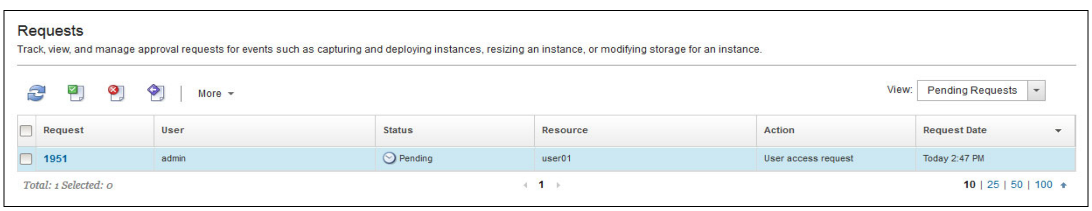


Figure 5-21 Request list

2. You can process the request by clicking **Approve**, **Reject**, or **Withdraw** as shown in Figure 5-22.

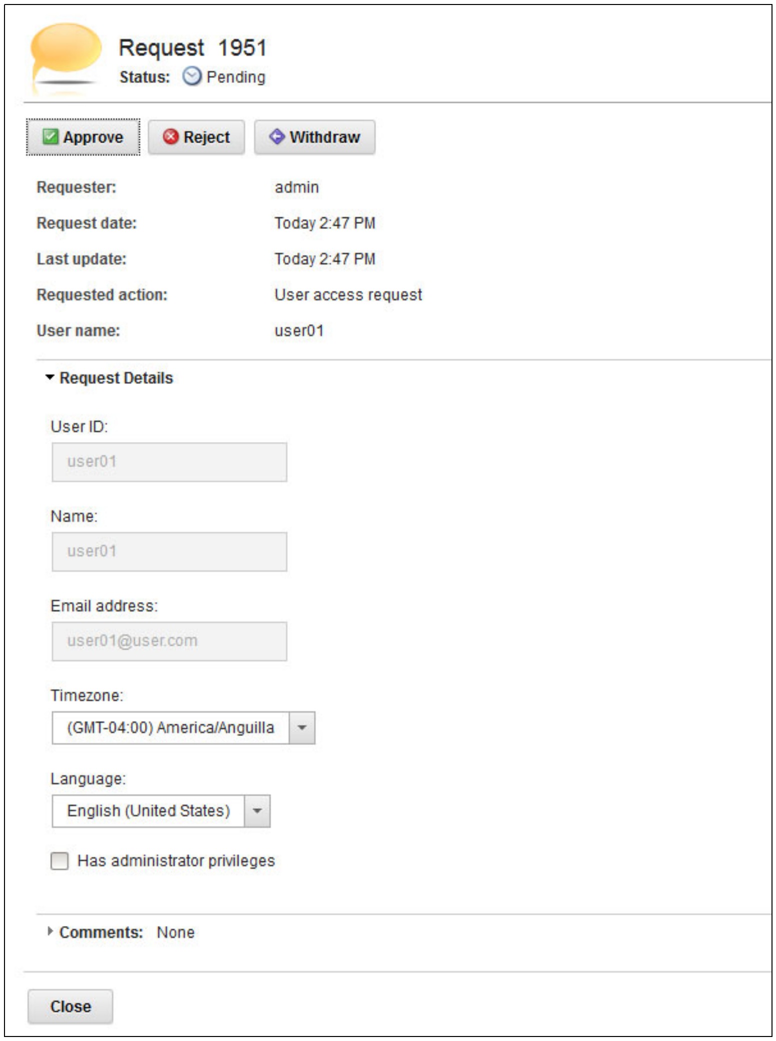


Figure 5-22 Request processing actions

5.2.2 Project management

From the IBM Cloud Manager with OpenStack for z Systems self-service portal, you can use projects, which are groups of virtual images and instances that are visible only to the members of that project.

IBM Cloud Manager with OpenStack for z Systems comes with a default project called the *Public* project, to which all users belong. All virtual images and instances that are created outside of the IBM Cloud Manager with OpenStack for z Systems are assigned to the Public project by default.

When you are added as a member of a project, one of these membership roles are assigned to you:

- Owner: A project owner has administrator authority over the project and its contents. The project owner primarily manages the contents of the project and who has access to the project and its contents.

- User: A project user has the authority to use the project and the objects within the project. For example, a project user can deploy a virtual image to the project. A user can also view the instances created by other users, depending on how the project and roles were initially set up. The project user primarily handles their own deployments.
- Viewer: A project viewer only has authority to view the project and the virtual images and instances that are contained in the project.

To create a project and add members to the project, complete these steps:

1. Log in as administrator, click the Access tab and then click **Projects** on the left navigation pane. The projects list is displayed as shown in Figure 5-23. Click **Create**.

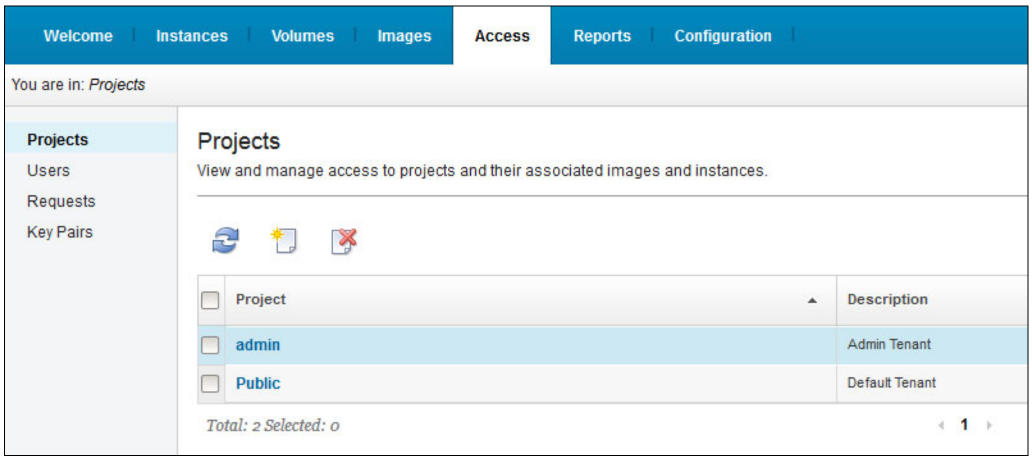


Figure 5-23 Projects list

2. Input the project name as shown in Figure 5-24, and click **Save**.

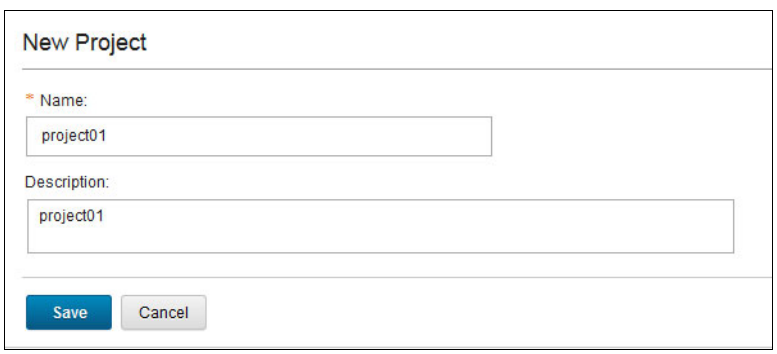


Figure 5-24 New Project window

3. The new project is added to the projects list as shown in Figure 5-25. Click the project name to view the project properties.

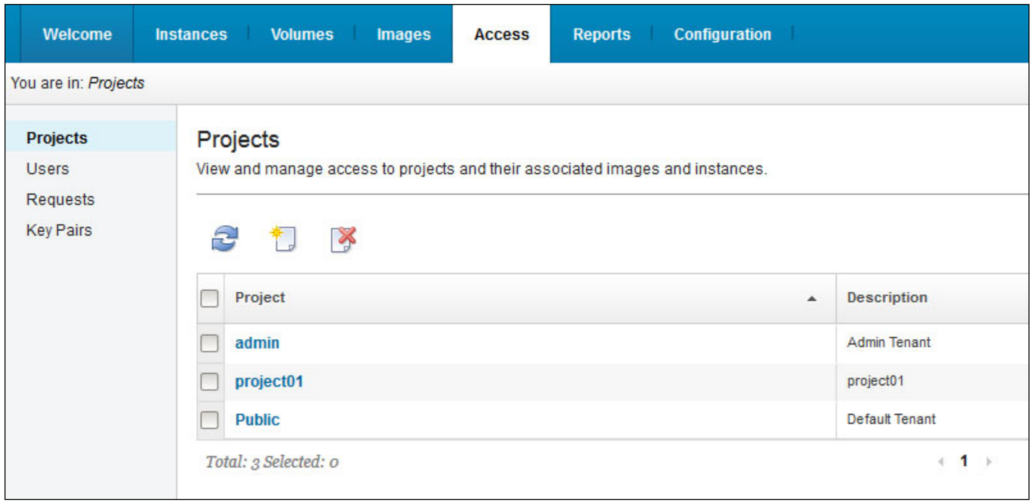


Figure 5-25 New projects list

4. Click **Edit** to edit the project properties as shown in Figure 5-26.

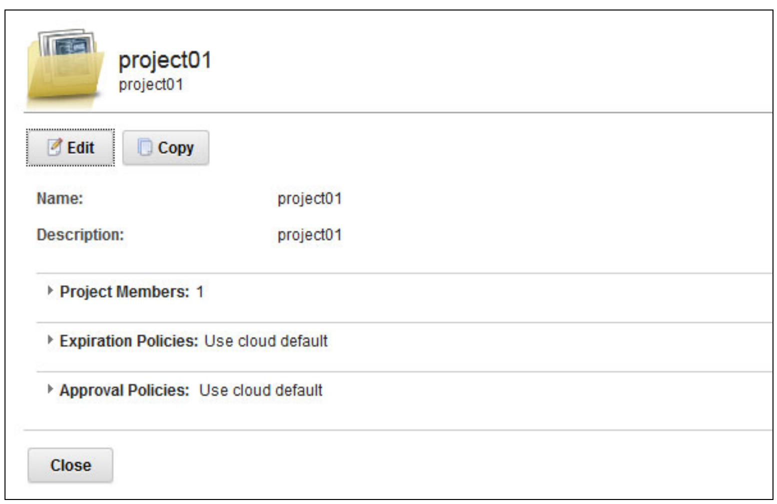


Figure 5-26 Project properties

5. In the Project Members section (Figure 5-27), click the **Add** icon to add members.

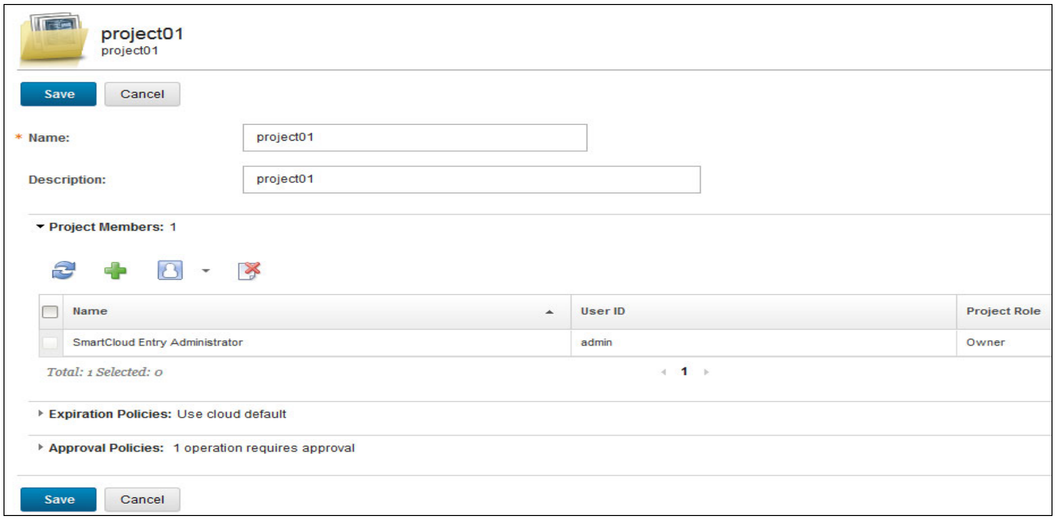


Figure 5-27 Add members to the project

6. As shown in Figure 5-28, select the user whom you want to add and click **OK**.

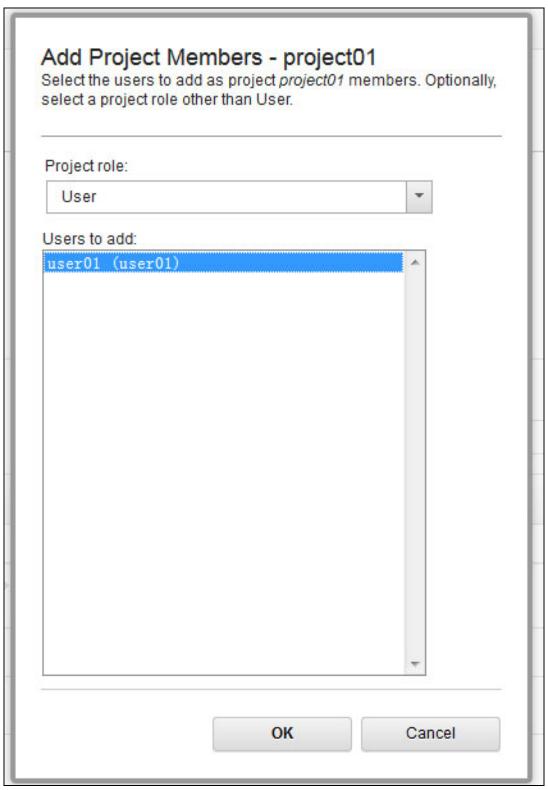


Figure 5-28 Add project member

7. As shown in Figure 5-29, select the check box of the user name, click the **Set role to** icon, and then select the role for this user.

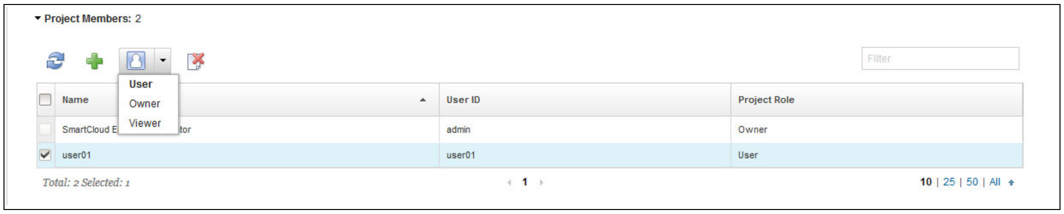


Figure 5-29 Edit member role

Now you have created a project and added a member to it.

5.2.3 Account management

The IBM Cloud Manager with OpenStack for z Systems self-service portal has configurable metering and a metering interface. The metering interface allows the self-service portal to monitor resource usage, and create subsequent billing to the self-service portal user accounts for the usage.

This section shows how to manage metering information, and how to create and manage accounts.

To enable metering and billing, see “Configure metering” on page 112 and “Configure billing” on page 114.

Metering management

After you enable metering, you can monitor the cloud resource usage from the *Usage Metering* tab. You can view details about a specific instance by selecting the instance from that tab.

Complete these steps to configure resource usage metering:

1. On the administrator welcome page (see Figure 5-3 on page 119), click the Report tab.
2. Click **Usage Metering** on the left navigation pane.

3. In the Resource Filter section, shown in Figure 5-30, select the search conditions. For example, you can enter `user01` in **User name** to see only the instances for that owner.

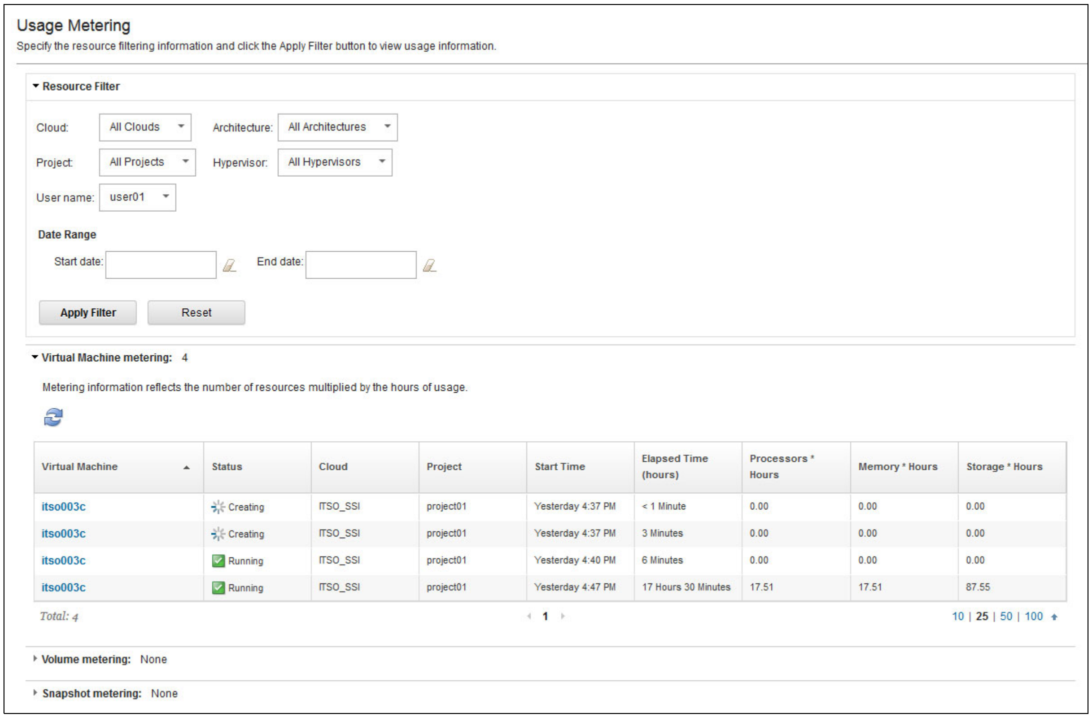


Figure 5-30 Usage metering

As shown previously in Figure 5-30, you can see the metering information of instances, volumes, and snapshots for `user01`.

> **Note:** A user can also view the Usage Metering tab and see the instance usage, but only if the user is the owner of the instance.

Account creation

You must create accounts when billing is enabled.

To create an account, follow these rules:

- Only self-service portal administrators can create accounts, but users can be made an account owner.
- You can deploy images to instances only if you are an account member and the account has a positive balance with which to pay for server use.
- Accounts have a balance, an owner, an account balance threshold, and account members.

To create accounts, complete these steps:

1. Click the Access tab, and then click **Accounts** on the left navigation pane as shown in Figure 5-31.

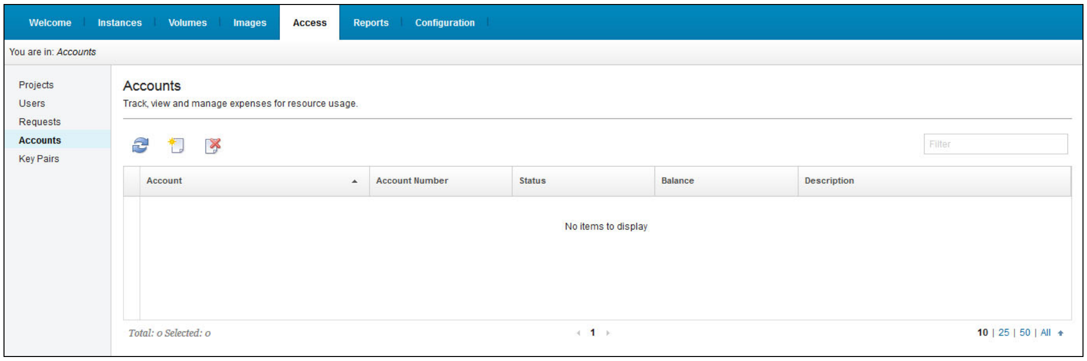


Figure 5-31 Accounts list

2. Click the **Create** icon.
3. You can name the account and set the account owner, balance, and members as shown in Figure 5-32. After you set the parameters, click **Save**.

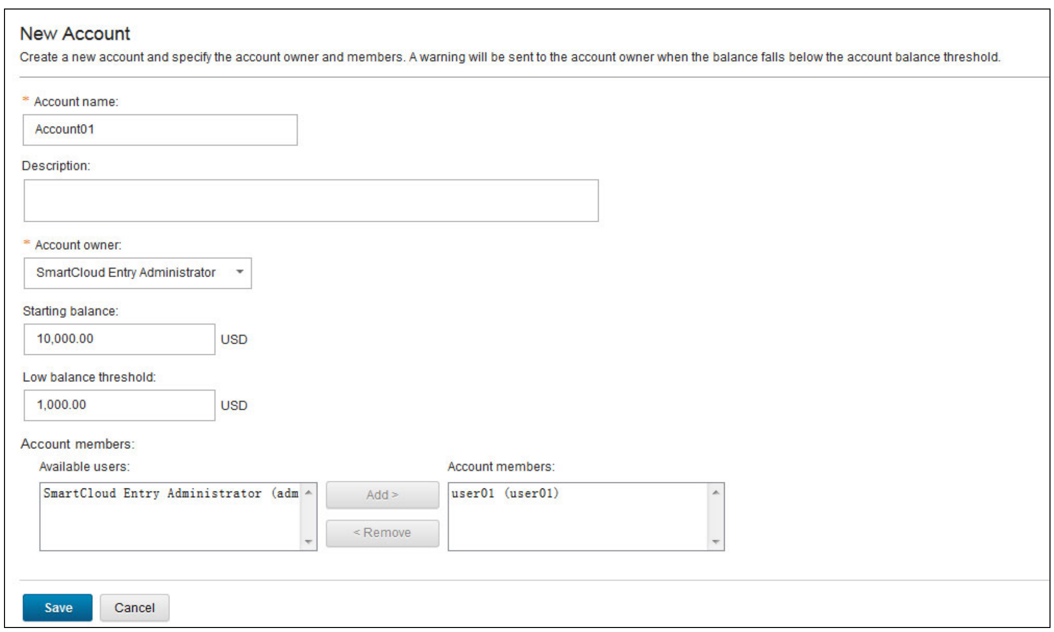


Figure 5-32 Create an account

The following are the configuration options:

Account owner The self-service portal user who is accountable for crediting and paying the account.

Starting balance the monetary balance of the account. The cost of each request and running deployment is subtracted from the balance over time.

Low balance threshold A value that represents the amount at which the account balance becomes a low balance.

Account members The self-service portal users who belong to the account. When account members deploy images to instances in the self-service portal, the instances are billed to their account.

Now you have successfully created an account.

Account management

You can manage accounts after they are created.

To manage accounts, follow these rules:

- Only account owners and self-service portal administrators can manage accounts.
- Each instance has an invoice in the account that it belongs to.

To manage accounts, complete the following steps:

1. On the administrator welcome window (Figure 5-3 on page 119), click the Access tab and then click **Accounts** on the left navigation pane. You can see the account that you created is listed (Figure 5-33).

Figure 5-33 New account list

2. Click the account name that you want to manage. The account properties window for that account is displayed as shown in Figure 5-34.

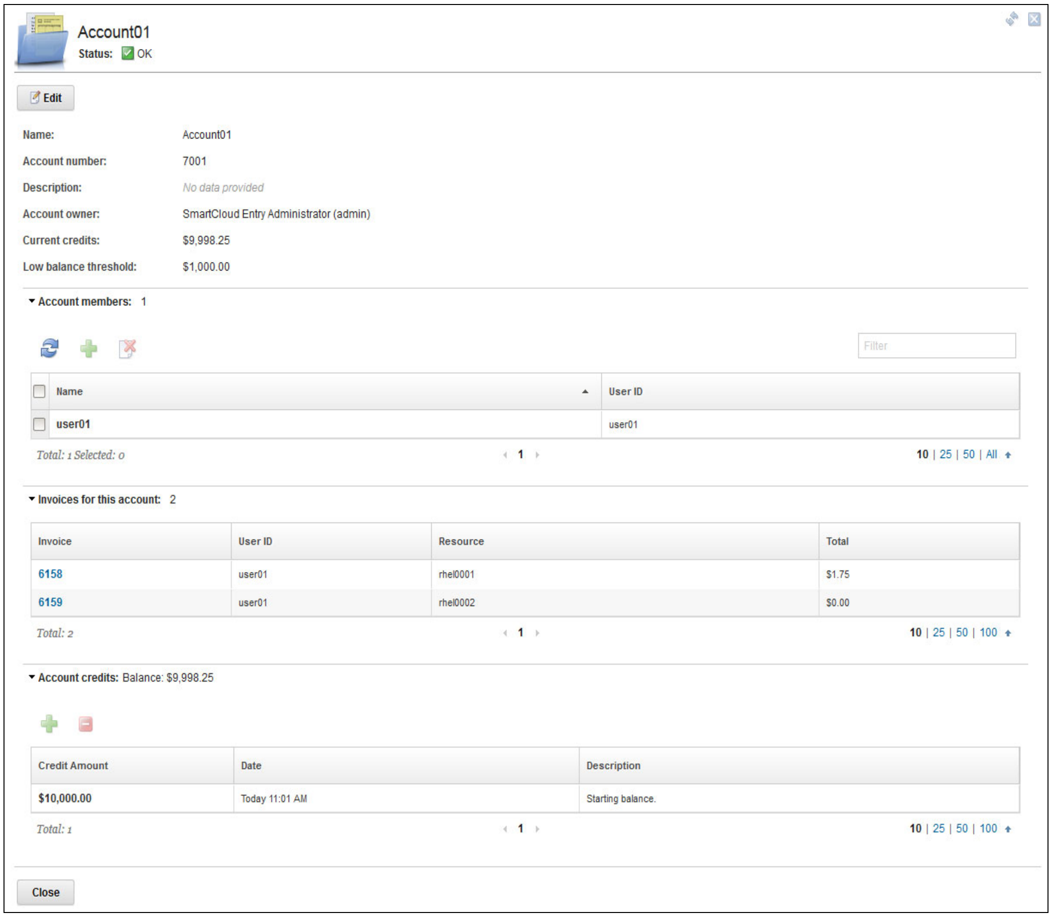


Figure 5-34 Account properties

3. Expand the Invoices for this account section. You can then see one invoice for each instance that has been created by the member of this account. Click the invoice name to see the details (see Figure 5-35).

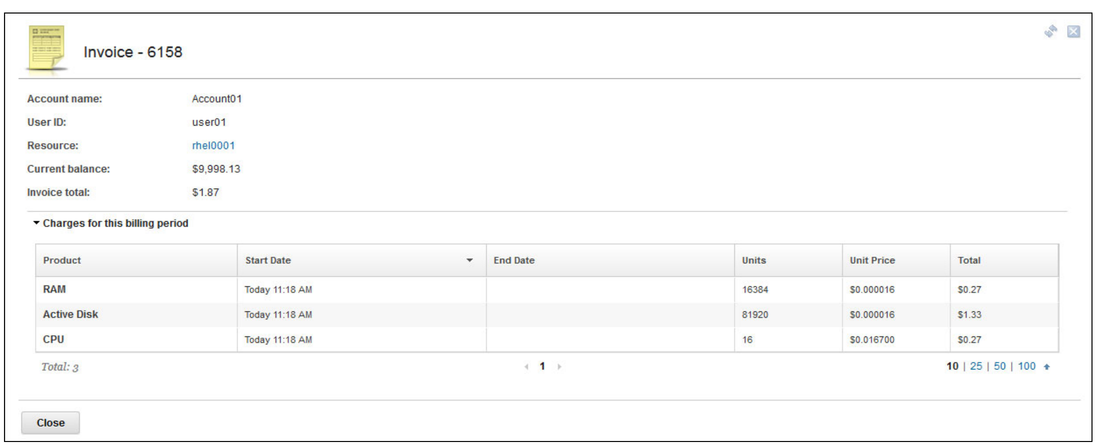


Figure 5-35 Account invoice

4. Go back to the Account properties page and click **Edit**. You can edit the account properties, as shown in Figure 5-36, and add credits for this account.

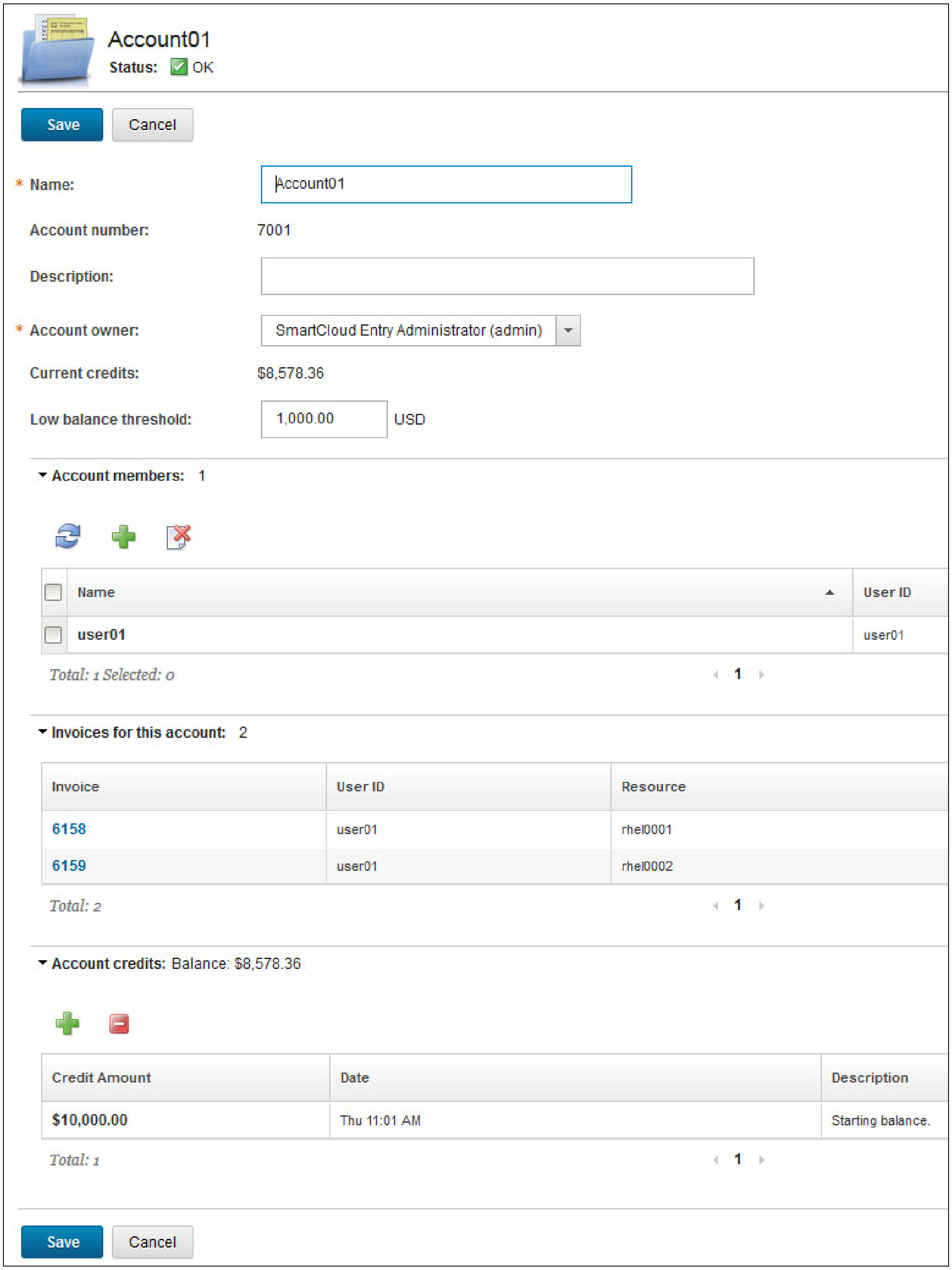


Figure 5-36 Edit account properties

Note: You can delete an account only if you are the owner of the account, and only when the account is not associated with any active instances.

5.2.4 User management

The Users tab in the self-service portal is enabled for administrative users and is used for creating, viewing, and managing users.

On the administrator welcome page (see Figure 5-3 on page 119), click the Access tab and then click **Users** on the left navigation pane. The User profiles are displayed (see Figure 5-37).

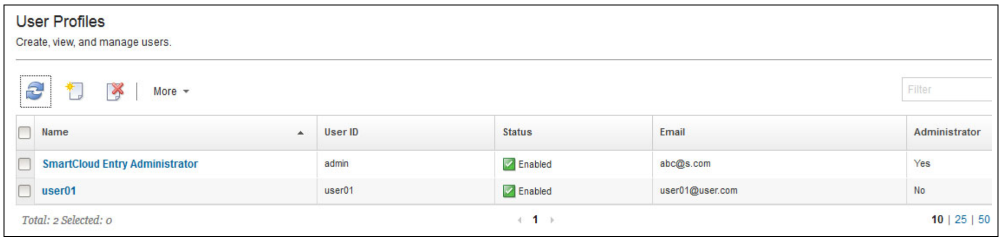


Figure 5-37 User profiles

You can create or delete a user by clicking the **Create** icon or **Delete** icon.

> **Note:** You can only create valid user accounts when using local authentication. When using LDAP authentication, user accounts are created and managed directly through the LDAP server.

You can also lock or unlock a user by selecting **More** and then clicking **Lock** or **Unlock**.

> **Note:** If a user has three invalid login attempts in a 24-hour period, the user account becomes locked and requires an administrator to unlock it.

A user can also request to create a user account. To request a user account, see "Request user account" on page 148. To approve or reject these requests, see "Request management" on page 129.

5.2.5 Image management

This section shows how to import an existing image and how to configure the images so they are available for deployment.

To manually create and upload an image, see "Capture a deployable Linux image into OpenStack glance" on page 87.

Import image

If you already have an OpenStack compatible image, you can import the image by completing these steps:

1. On the administrator welcome page (see Figure 5-3 on page 119), click the Images tab, click **More** and select **Import Image**, as shown in Figure 5-38.

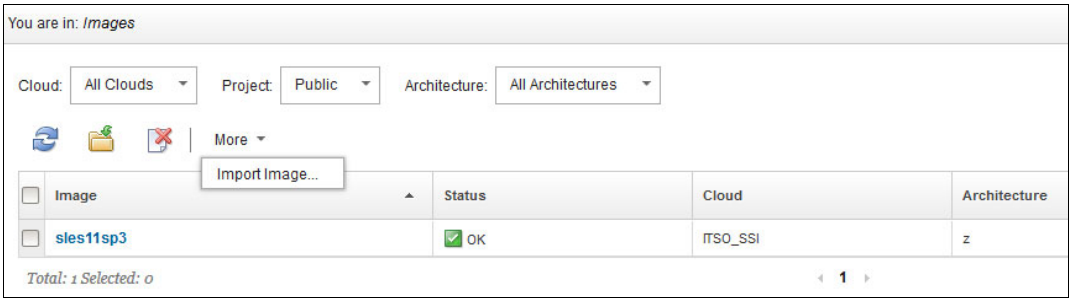


Figure 5-38 Image list

2. Input the image parameters, as shown in Figure 5-39, then click **Import**.

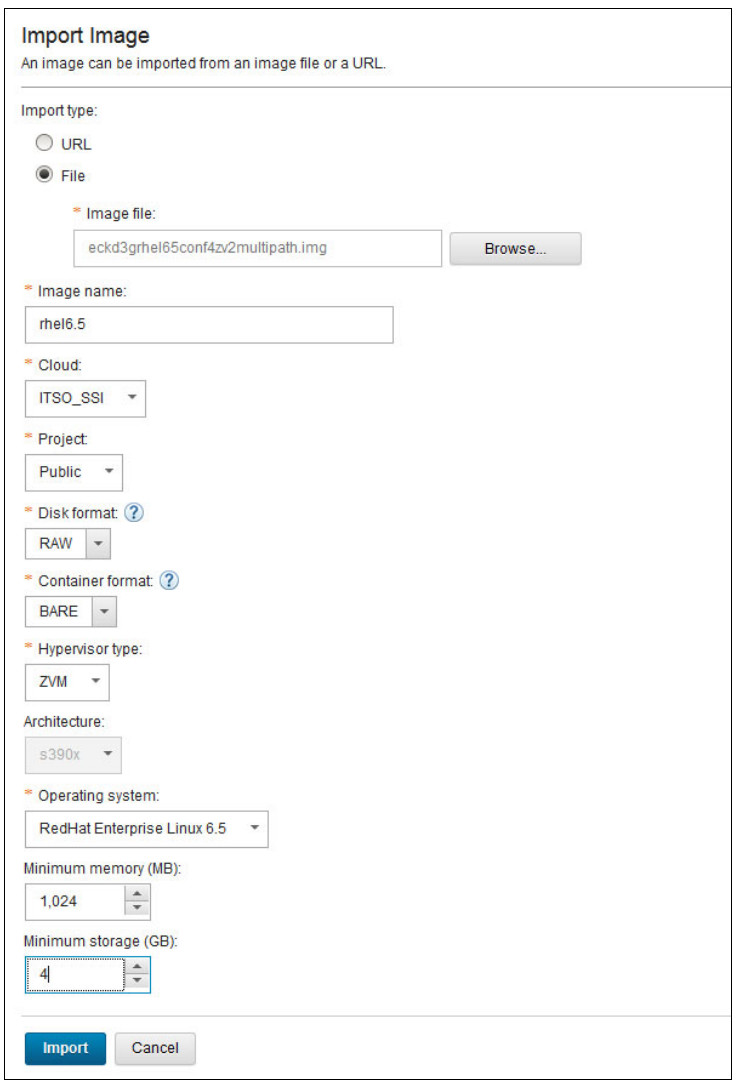


Figure 5-39 Import image

The following are the configuration options:

Import type You can choose to use a URL or a local file location.

Image name You can name your image.

Cloud Choose the z/VM cloud that you created during cloud environment configurations.

Project Choose which project can use this image.

Disk format Select **RAW**.

Container format Select **Bare**.

Hypervisor type Select **z/VM**, and the architecture automatically changes to s390x.

Operating system type Select the OS version for your image. ICM only supports RHEL 6.2, RHEL6.3, RHEL6.4, RHEL6.5, SLES11.2, and SLES11.3 on z.

Minimum memory This value affects which flavor you can use on deployment.

Minimum storage This value affects which flavor you can use on deployment.

3. The image shows up in the image list when the import is finished as shown in Figure 5-40. You can see a new image is generated.

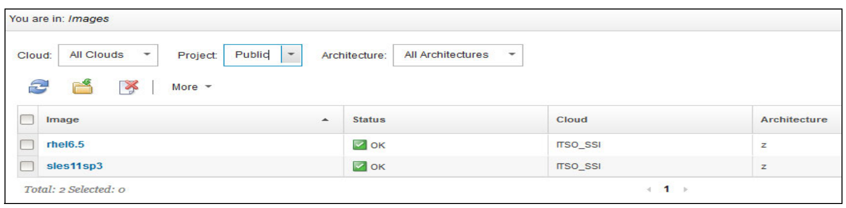


Figure 5-40 New image list

Image configuration

You can deploy an image with either basic or advanced configuration options.

Users can only deploy an image by using the basic deployment form. Project owners or administrators can use the basic or advanced deployment forms.

In this section, you will configure the default parameters for the image deployment and configure settings that are shown on the basic deployment form.

As shown in Figure 5-41, set the default parameters on deployment target, flavors, software, and network. You also need to verify which part will show in the basic deployment settings.

Note: `Enable config drive` is not supported on z Systems. You must select management network as your first network adapter from the menu as shown in Figure 5-41.

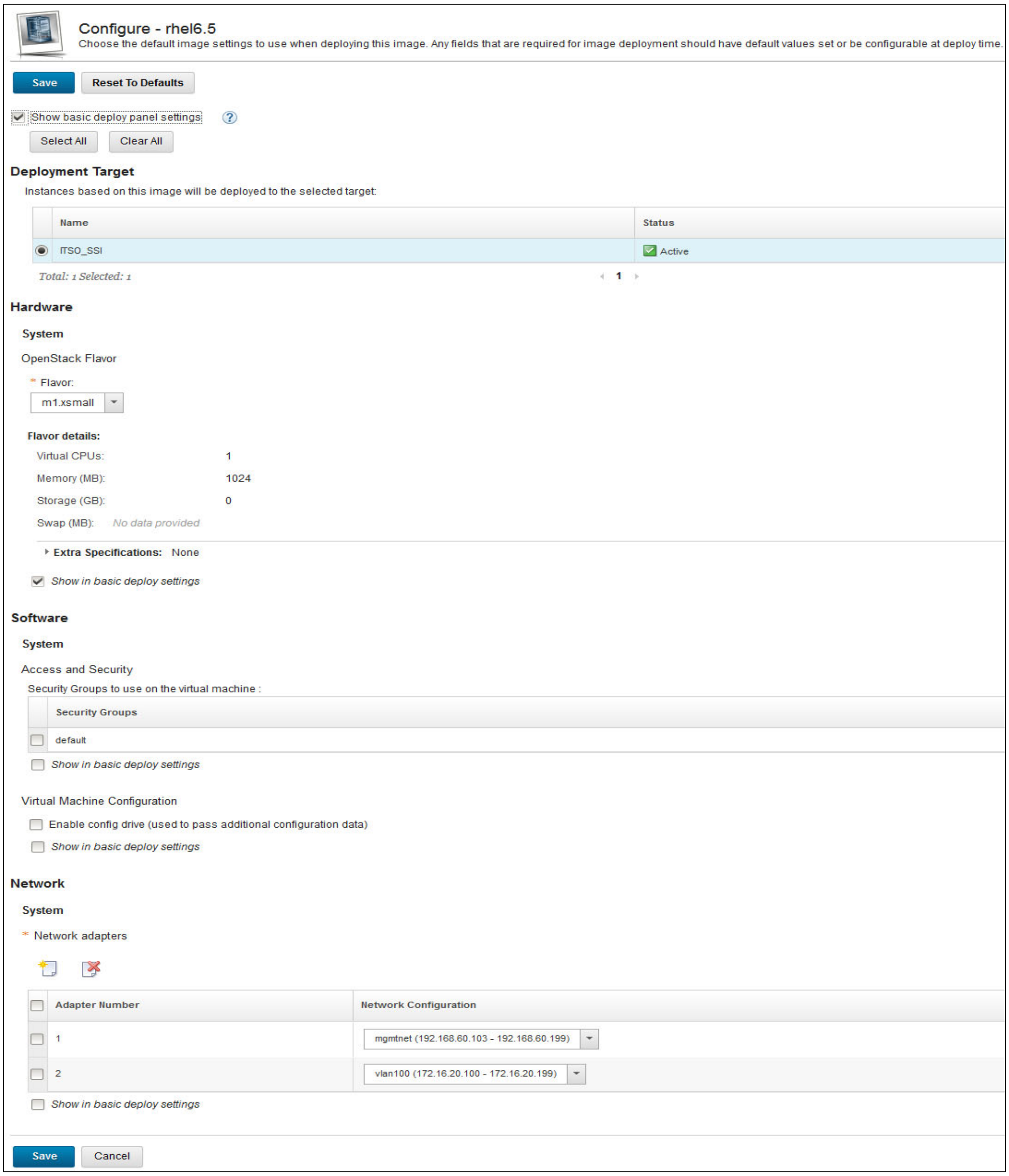


Figure 5-41 Image configurations

5.2.6 Capacity management

Using the Capacity view, you can identify the current capacity of the resources in your virtualization environment.

Understanding the capacity of resources within the cloud helps you gauge the health of the entire cloud. It also helps you determine suitable targets for instance migration.

To access the Capacity view, on the administrator welcome page (see Figure 5-3 on page 119), click the Reports tab, then click **Capacity** from the left navigation pane as shown in Figure 5-42.

> **Note:** The color of the capacity indicator can be green or yellow. Green indicates that the used resources are less than the physical resources. Yellow indicates that the used resource is overcommitted on the available physical resources, but you can still deploy.

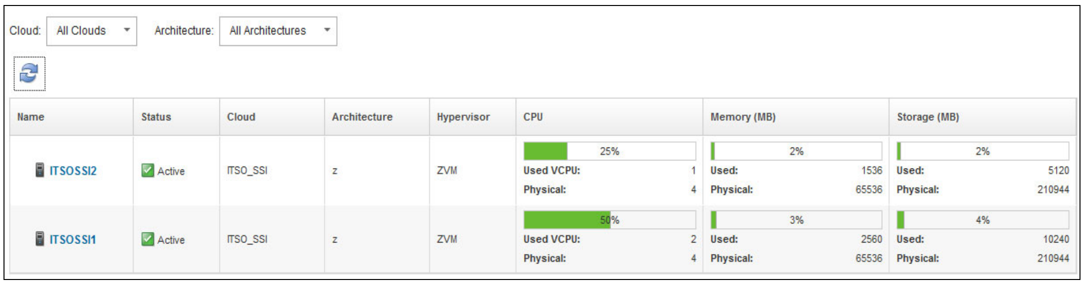


Figure 5-42 Capacity management

5.2.7 Instance migration

To balance the resources between physical hosts, You can migrate an instance to a specific physical host. You can only do instance migration in the ECKD environment.

> **Note:** Make sure that the instances that you want to migrate have a status of `OK`.

Complete these steps to migrate an instance:

1. On the administrator welcome page (see Figure 5-3 on page 119), select the Instances tab, click **More**, and then select **Migrate to new host** as shown in Figure 5-43.

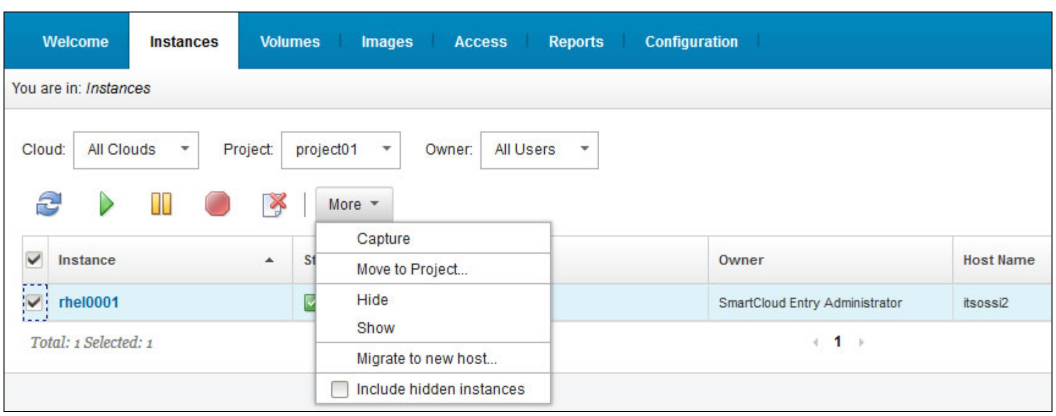


Figure 5-43 Instance migration

2. Select **Manually select** and choose the physical host, then click **Migrate** (Figure 5-44).

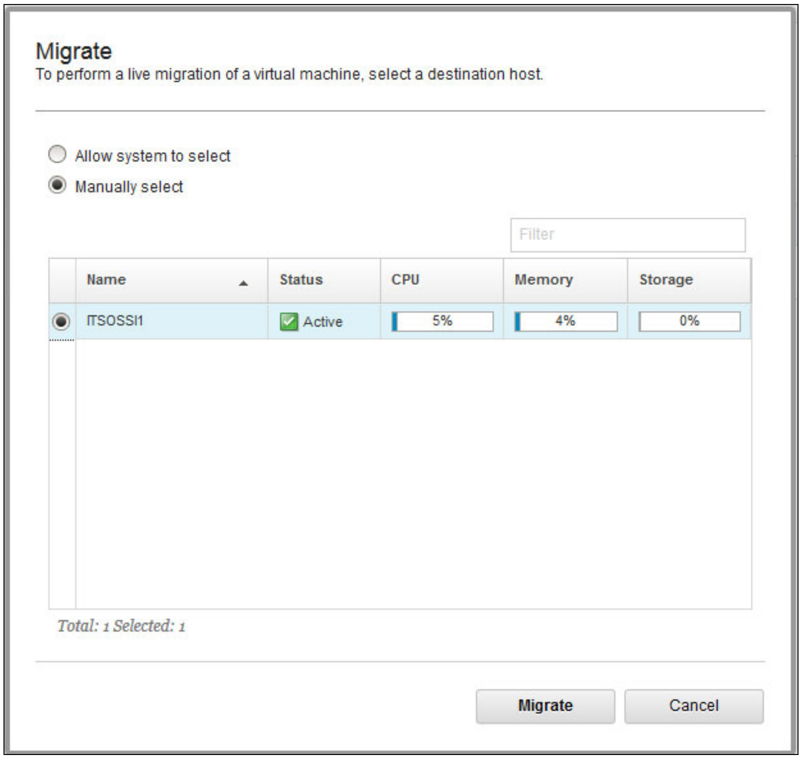


Figure 5-44 Select migrate target

After the migration completes, you can check the new host for this instance in the *Instances* tab.

5.2.8 Volume management

After "Configure cinder persistent data disks" on page 79 is completed, you can use the *Volumes* tab on self-service portal to create and manage volumes for your instance.

When you add a volume, you need to select the empty volume as the source for the new volume. When you delete a volume, all data is lost from that volume. Alternatively, you can detach a volume from a virtual machine to retain the content of the volume and use it in the future by attaching it to a different instance.

Complete these steps to manage your volumes:

1. Click the Volumes tab as shown in Figure 5-45, then click **Create**.

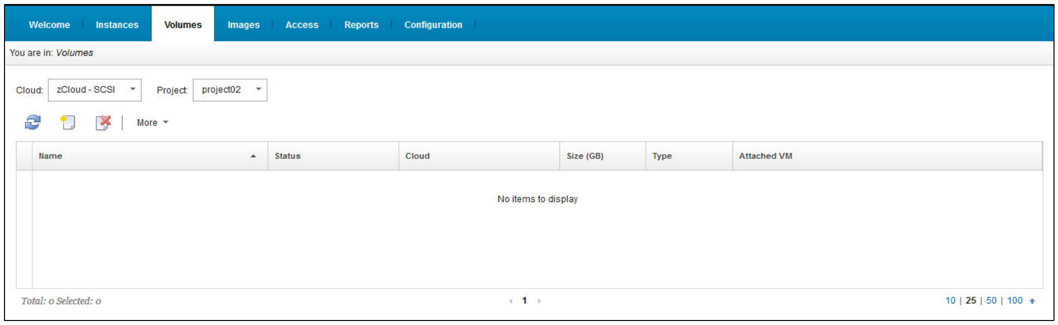


Figure 5-45 Volumes list

2. Define the volume name, project, and size (Figure 5-46). You must select the type that you defined during “Create cinder volume type” on page 86, and select the source as **No source, empty volume**. Then click **Save**.

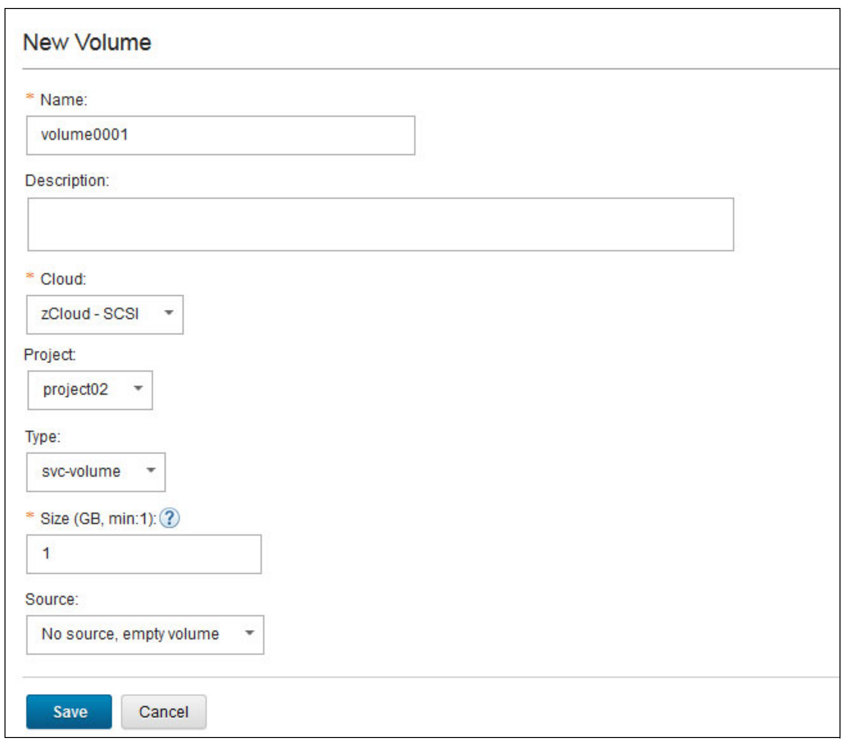


Figure 5-46 Create new volume

3. Refresh the Volume tab, and you see the new volume in the volumes list (Figure 5-47). To add a volume to an instance, select the volume and click **More** → **Attach**.

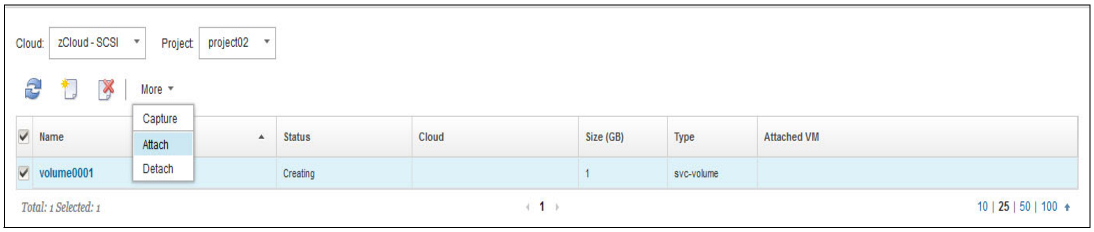


Figure 5-47 Volume operations

4. Define the device address and select the target instance as shown in Figure 5-48. You can set the block device to be a second disk device (for example: `/dev/sdb`).

Figure 5-48 Attach volume

You can detach the volume from the instance by clicking **More** → **Detach** as shown in Figure 5-49. You can also delete the volume by clicking the **Delete** icon when it is detached.

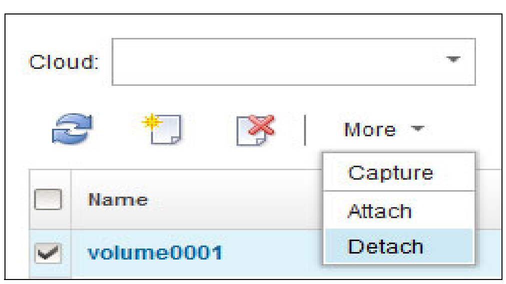


Figure 5-49 Detach volume

5.3 Self-service portal user operations

With the user role in the IBM Cloud Manager with OpenStack on z Systems self-service portal, you can perform tasks such as deploy images, capture instances, and instance lifecycle management.

This section shows how to use the self-service portal from a user perspective.

For daily operations, the user typically focuses on creating and managing the instances related to them.

5.3.1 Request user account

Before the user logs in to the IBM Cloud Manager with OpenStack for z Systems for the first time, the user must request a user account through the system administrator. The user also must know the URL of your IBM Cloud Manager with OpenStack server.

Complete these steps to request a user account:

1. In a browser, navigate to your self-service portal. In this scenario's environment, the following URL was used:

   `https://192.168.60.101:18443/cloud/web/login.html`

2. Login in to the self-service portal as shown in Figure 5-50.

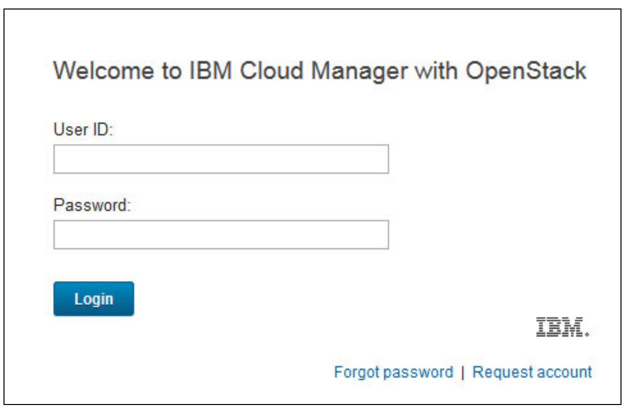


Figure 5-50 User login window

3. Click **Request account**.

4. Input data into the following parameters to request an account, and click **Send Request** as shown in Figure 5-51. After the administrator approves the request, an account is created.

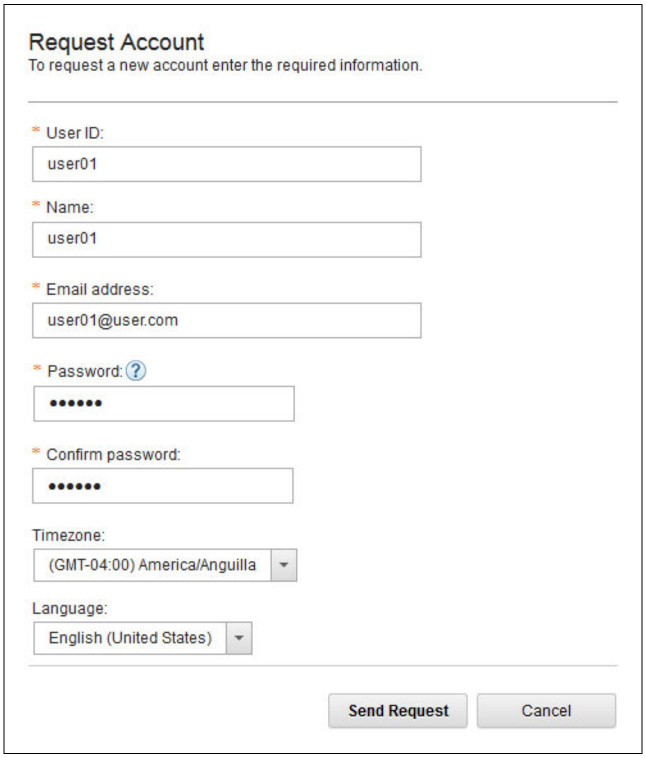


Figure 5-51 Request account

5. After your account is created, you can log in with your user ID and password as shown in Figure 5-52.

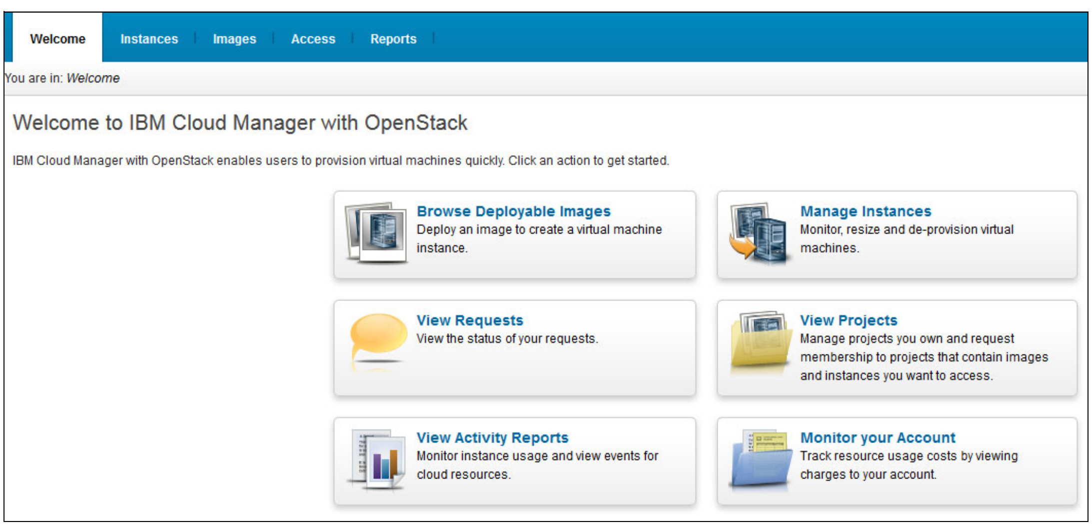


Figure 5-52 User welcome page

5.3.2 Instance management

You can use the *Instances* tab in the self-service portal to manage instances after they are created. You can view the instances by specified cloud, project, or owner.

View instance properties

Click the instance name on the Instance tab to see the instance properties as shown in Figure 5-53.

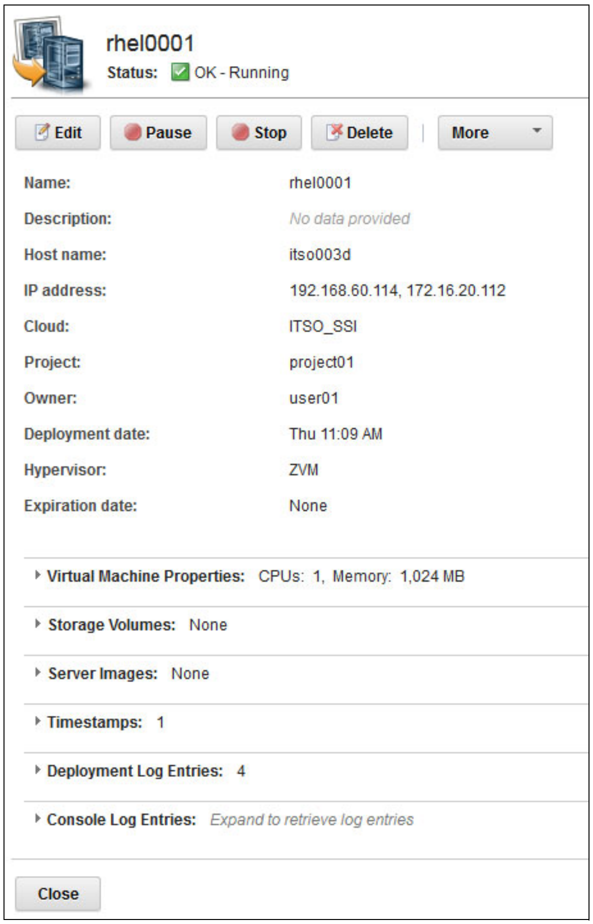


Figure 5-53 Instance properties

On the Instance properties window, you can view the instance status and perform actions on the instance. You can also see the basic properties for the instance, including the logs if they are available.

Expand the Virtual Machine Properties section to see more properties of the instance.

Stop and start the instance

To stop an instance, select the target instance on the Instance tab, and click **Stop**, as shown in Figure 5-54.

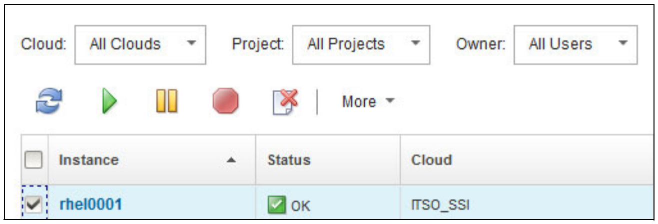


Figure 5-54 Stop instance

Verify the stop operation by clicking **Continue**, as shown in Figure 5-55.

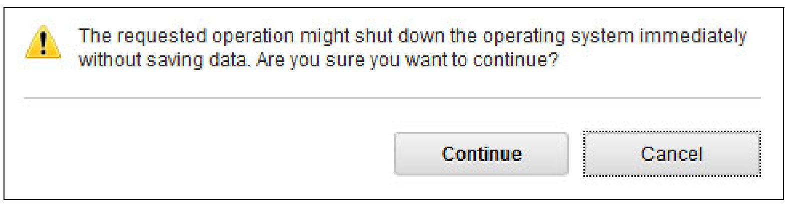


Figure 5-55 Instance stop verification

To start an instance, select the target instance on the Instance tab, and click **Start,** as shown in Figure 5-56.

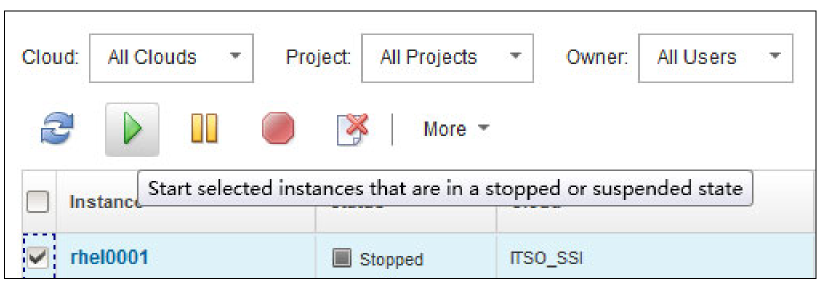


Figure 5-56 Start instance

You can also stop or start the instance from the instance properties page if your user ID has an authorized role.

> **Note:** Only the resource creator and users with the following roles are authorized: [Admin, Owner].

Resize instance

You can change the amount of resources that an instance uses by modifying the flavor of the instance.

On the User Properties window, click **More** → **Resize** as shown in Figure 5-57.

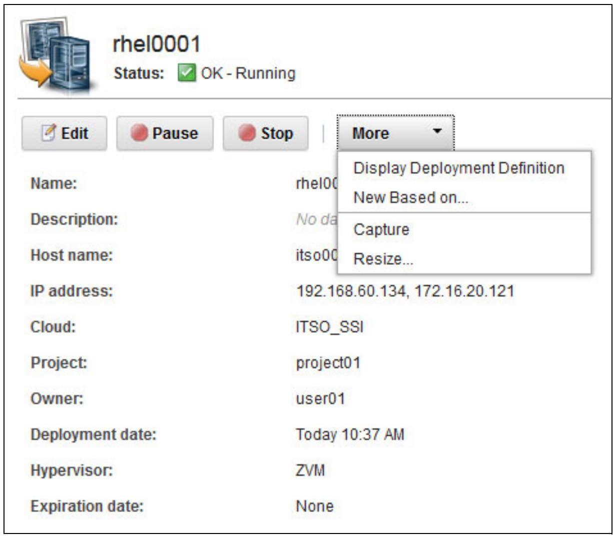


Figure 5-57 Resize instance

Select the flavor that you want to use for the instance and click **Resize** as shown in Figure 5-58.

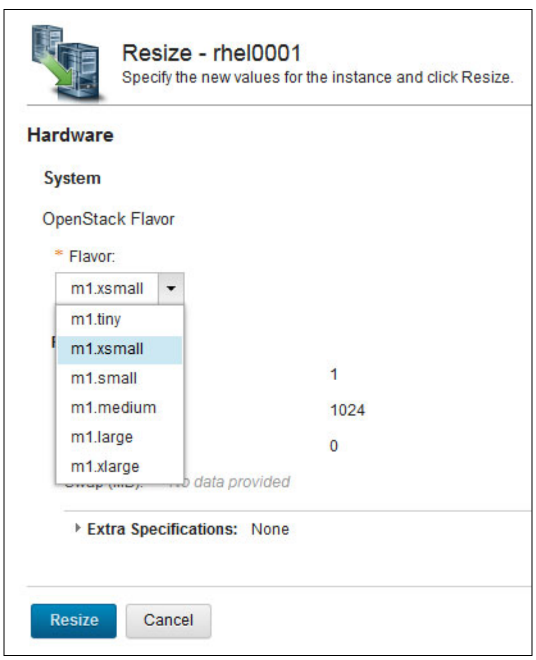


Figure 5-58 Choose flavor on resizing

Delete instance

To delete an instance, select the target instance on the Instances tab, and click **Delete**, as shown in Figure 5-59.

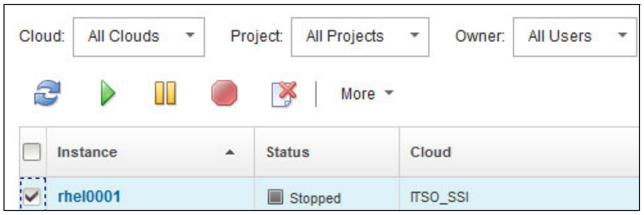


Figure 5-59 Delete instance

You need to verify the delete operation by clicking **Delete** as shown in Figure 5-60.

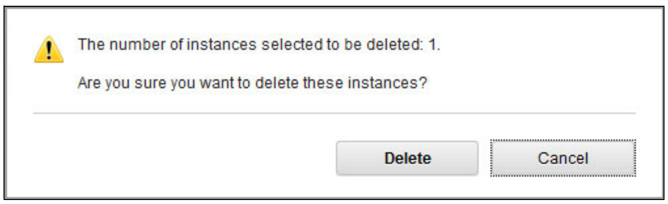


Figure 5-60 Instance deletion verification

5.3.3 Capture instance to image

You can capture a snapshot of the instance that can be used to create a new virtual image based on that instance. The virtual image can be deployed later.

To capture an instance, you can select the target instance using the Instances tab. Click **More** and select **Capture** as shown in Figure 5-61.

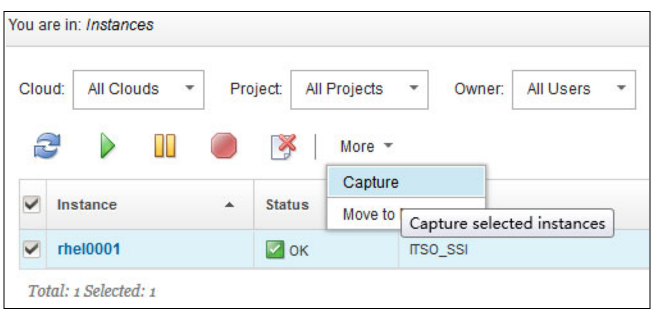


Figure 5-61 Capture an instance

Note: The status of the instance you want to capture must be `OK`.

When completed, you see the snapshot on the Images tab as shown in Figure 5-62.

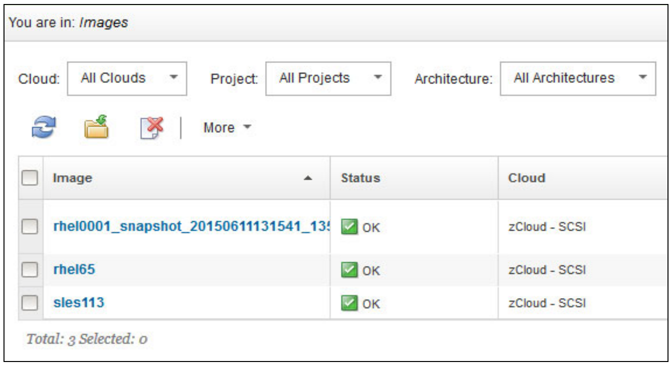


Figure 5-62 Capture result

5.3.4 Deploy image to instance

This section describes how a user can deploy an image to an instance by using the basic deployment functions. For more information about basic and advanced deployment, see "Image management" on page 140.

Complete these steps, starting from the User Welcome window (see Figure 5-52 on page 149):

1. On the Images tab, click the image name that you want to deploy, as shown in Figure 5-63.

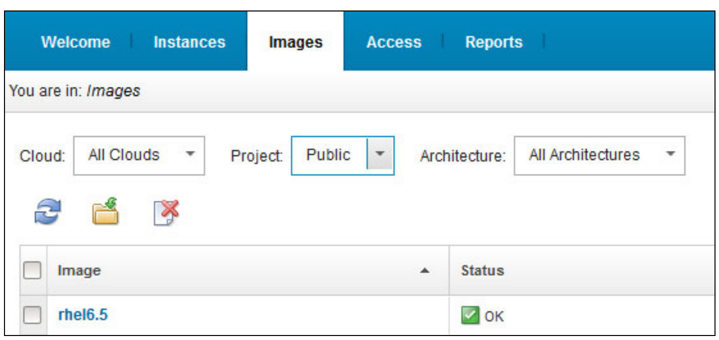


Figure 5-63 Images list

2. Click **Deploy**, as shown in Figure 5-64.

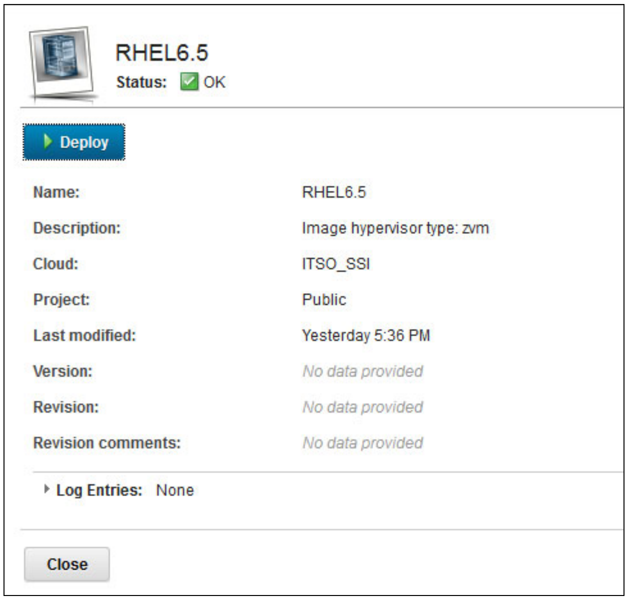


Figure 5-64 Image properties

3. Input the parameters as shown in Figure 5-65 and click **Deploy**. The parameters are configured by the administrator in "Image management" on page 140.

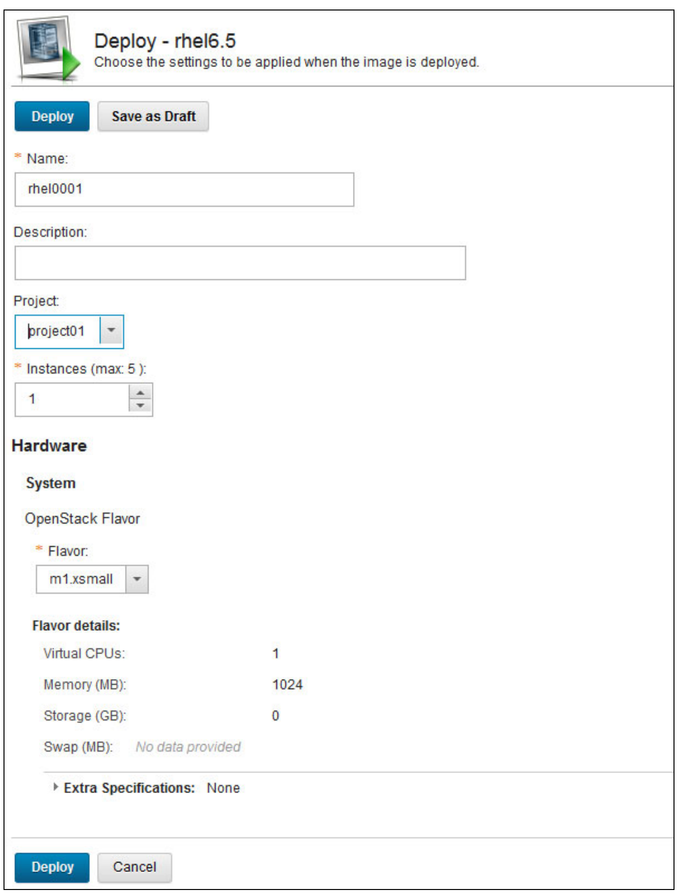


Figure 5-65 Deploy image

4. On the Instances page, you see the new instance status is deploying, as shown in Figure 5-66.

   The status will not automatically refresh. Click **Refresh** to see the latest status.

   Note: If an event requires approval, the deployment process begins when the administrator approves it.

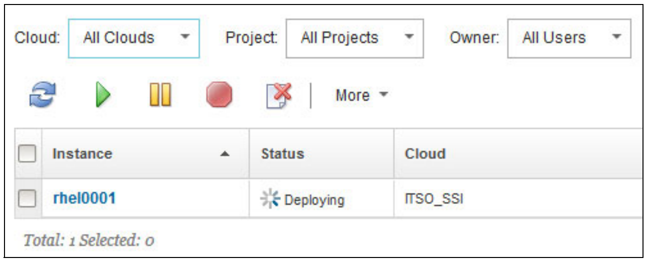


Figure 5-66 Deploying status

5. The status changes to `OK` as shown in Figure 5-67. This status change means that you have deployed your images to an instance. You can then see the details of the image, such as an IP address, in the Instance Properties window.

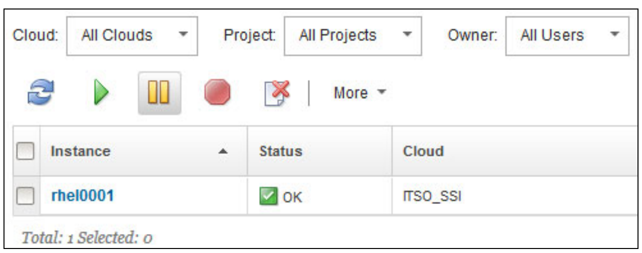


Figure 5-67 A finished deployment

5.3.5 Withdraw or resubmit the request

You can use the Requests tab to view, withdraw, and resubmit the requests.

For example, when you deploy an image to an instance, an instance request is created and submitted to the administrator for approval. The instance status is set to `Pending` until the administrator process the approval request.

On the User Welcome window (see Figure 5-52 on page 149), select the Access tab, and click **Requests** on the left navigation pane. You then see all of your requests and their status.

To withdraw a request, select the request and click **Withdraw,** as shown in Figure 5-68.

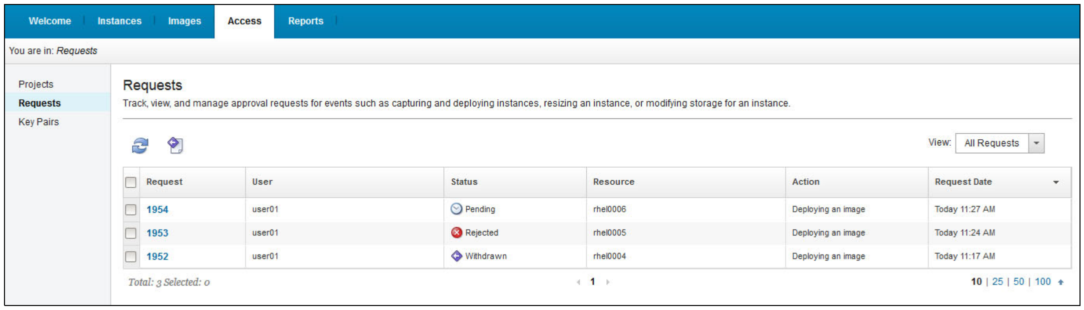


Figure 5-68 Request list

If an administrator rejects your request, you can modify your request or provide additional comments to resolve any issues and resubmit your request.

Click the request name to see the Properties window as shown in Figure 5-69 on page 158. From there, you can modify your request or provide additional comments, and then click **Resubmit**.

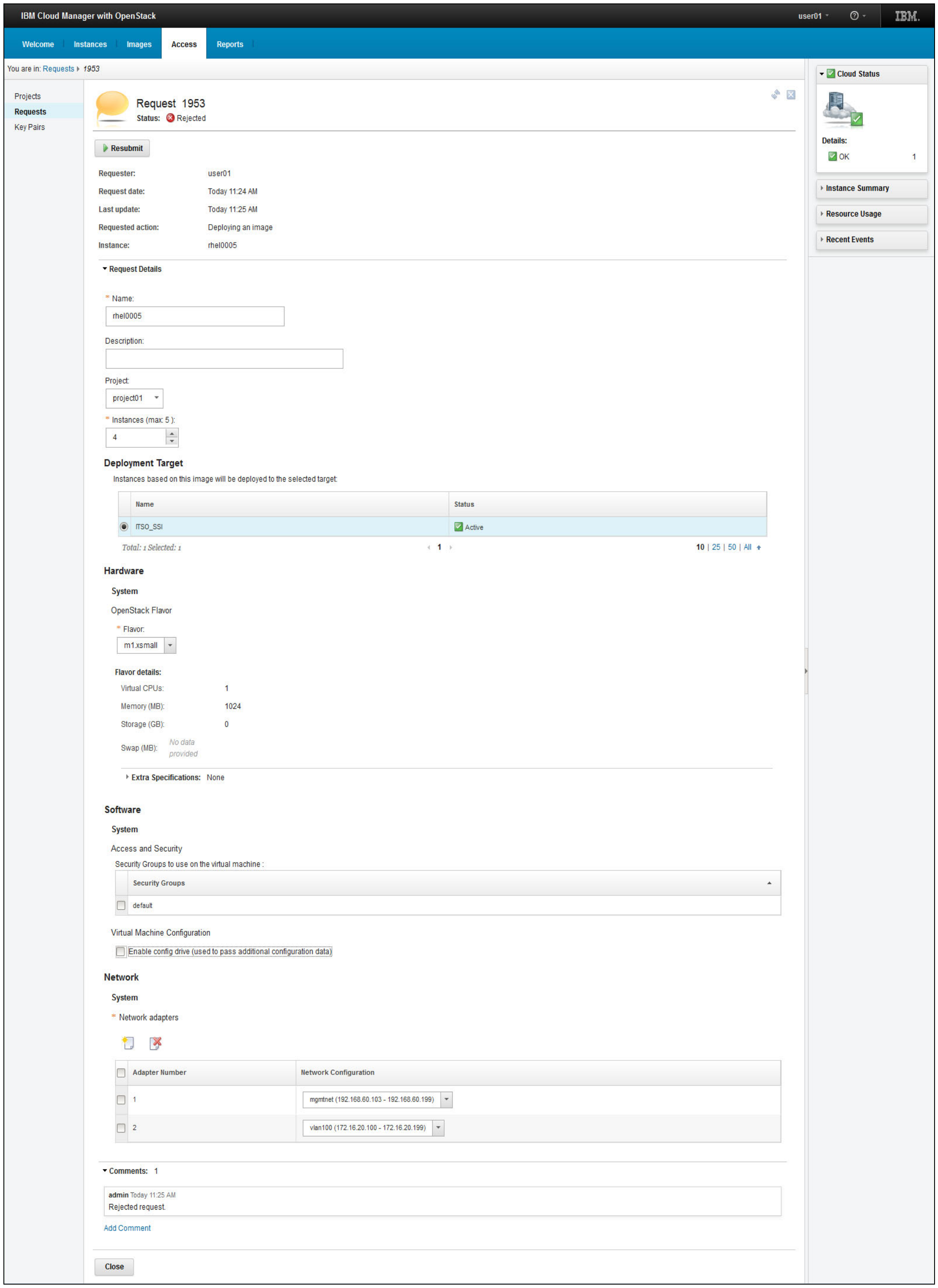


Figure 5-69 Modify and resubmit the request

6

Adding patterns to the cloud infrastructure

This chapter describes how to deploy a pattern definition in a Heat Orchestration Template (HOT) by using the IBM Cloud Management Dashboard and the IBM UrbanCode Deploy with Patterns.

If you are not familiar with the available patterns and how they can be deployed, see "IBM Custom Patterns for Linux on z Systems" on page 21.

This chapter includes the following sections:

6.1 Deployment environment description

IBM Custom Patterns for Linux on z Systems is a set of IBM software, Chef cookbooks and recipes, and associated documentation. You can deploy those patterns by composing OpenStack HOT templates and run these templates using the Heat engine in IBM Cloud Manager with OpenStack for z Systems. The HOT template contains all the related Chef cookbook and recipes. The architecture of the pattern deployment is shown in Figure 6-1. You can either write the HOT template by using the dashboard portal or import the HOT template file into the dashboard portal.

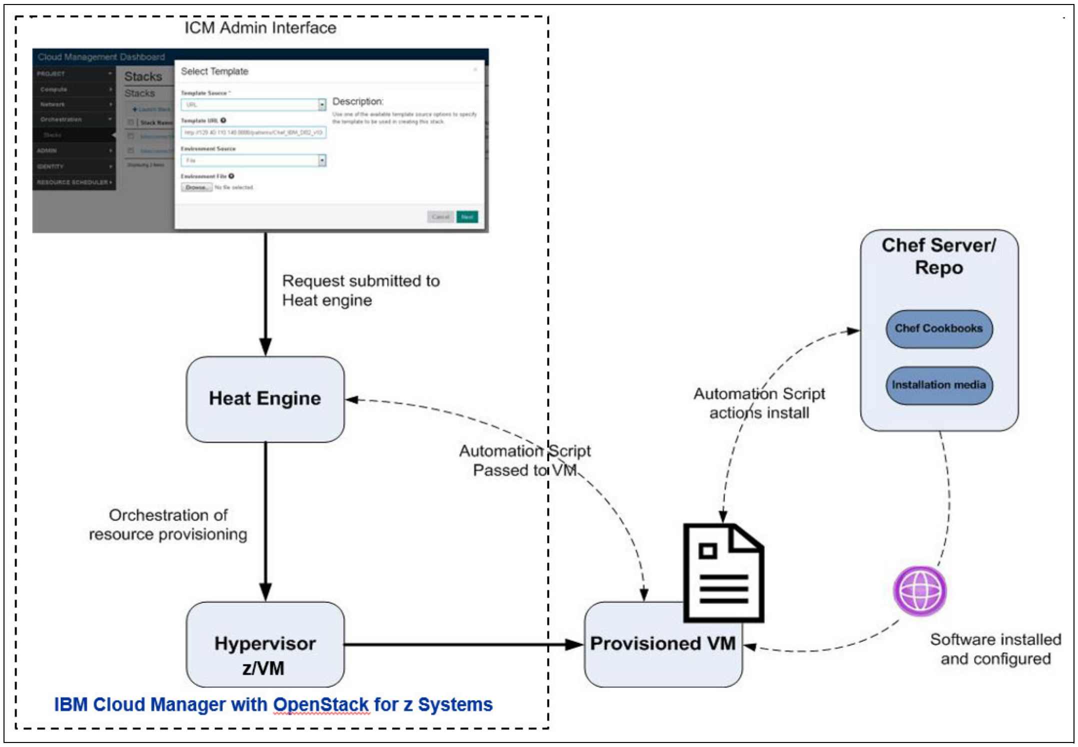


Figure 6-1 IBM Custom Patterns for Linux on z Systems architecture

In the example environment, the Heat engine is part of the IBM Cloud Manager with OpenStack for z Systems. The Chef server is on the same x86 server as the software repository of the IBM Custom Patterns.

6.2 Prerequisites for deploying IBM Custom Patterns

Several prerequisites are needed before deploying the patterns, some of which depend on the patterns being deployed.

When you get an IBM Custom Patterns for Linux on z Systems, it includes documentation about the prerequisites that need to be met before deploying the IBM Custom Pattern, some of which are:

- Install instructions for the Chef server and client
- How to configure Linux repository connection (for example, yum repository for Red Hat)
- Software installation images and other configuration needed

For more information about the different IBM Custom Patterns for Linux on z Systems, you can review the documentation at:

http://www.ibm.com/common/ssi/index.wss?request_locale=en

Go to Search Results tab, select **Announcement letters** for the **Information Type**, in **Search for** select **Letter Number**, and search for **215-052**.

> **Note:** When you order an IBM Custom Patterns for Linux on z Systems, it includes software product license (for example, DB2, WebSphere Application Server) and patterns (Chef cookbooks and recipes, documentation, and services). Some of these steps are described in this section for your reference.

6.2.1 Installing the Chef server and client

To implement the patterns, a Chef server is required to host the cookbooks of the patterns. The Chef server in the controller node of IBM Cloud Manager with OpenStack for z Systems can be used. You can also install a separate Chef server either in a Linux on z Systems or a Linux on distributed platform.

The example environment uses a Chef server on an x86 Linux server. You also need to install a Chef client to a deployable Linux image or during the Linux instance deployment process automatically by using a HOT template.

6.2.2 Configuring the software repository for Linux instances

IBM Custom Patterns for Linux on z Systems requires prerequisite libraries for a pattern to function. Therefore, configure the Linux instance to connect to the corresponding software repository, such as the yum repository for Red Hat.

Consider running a full system update before deploying the pattern.

6.2.3 Customizing the cookbook attributes

Every IBM Custom Pattern has a list of customizable cookbook attributes that can be updated. These attributes are the default parameter values for software deployment. Some examples of the values in the case of the WebSphere Application Server are: WAS admin user ID, installation directory, server name, and HTTP listen port. You can expose or hide the attributes to the deployment UI, or customize the default values as you want in the HOT templates.

You need to review each attribute and update it according to your environment.

6.2.4 Uploading cookbooks to the Chef server

Upload each of the cookbooks for your patterns to the Chef server.

Refer to the Chef documentation on how to upload a cookbook:

https://docs.chef.io/knife_upload.html

Here is an example for uploading a cookbook (my_cookbook), using the knife command:

```
knife cookbook upload -o ${unpack_location}/cookbooks my_cookbook
```

6.2.5 Preparing the software repository for the patterns

When deploying an IBM Custom Pattern, review the pattern documentation to determine the electronic images that are required, then make those images available to the deployment hosts.

The software repository can be in the same server as the Chef server or a separate server. IBM Custom Patterns for Linux on z Systems support the following two access methods to retrieve the software installation images on the deployment host:

Local The software installation packages required by the pattern file can be copied to a local directory (or made available through NFS or SMB) on the deployment host.

HTTP The software installation media can be hosted on an HTTP server in your environment.

6.3 Defining HOT templates

HOT templates are defined as YAML files. YAML Ain't Markup Language (YAML) is a human readable data serialization format inspired by the XML, Python, C, and Perl programming languages.

Example 6-1 shows the elements and structure that a HOT template includes.

Example 6-1 HOT template structure

```
heat_template_version: 2014-10-16

description:
  # a description of the template

parameter_groups:
  # a declaration of input parameter groups and order

parameters:
  # declaration of input parameters

resources:
  # declaration of template resources

outputs:
  # declaration of output parameters
```

HEAT template version

The key with value `2014-10-16` indicates that the YAML document is a HOT template and it can contain features added or removed up until the Juno release.

Description

This optional key allows you to provide a description of the template, or the workload that can be deployed using the template.

Parameter groups

This section allow you to specify how the input parameters must be grouped and the order to provide the parameters. This section is optional and can be omitted if wanted.

Parameters

This section allows for specifying input parameters that must be provided when instantiating the template. The section is optional and can be omitted when no input is required.

Resources

This section contains the declaration of the single resources of the template. This section with at least one resource must be defined in a HOT template, or the template will not really do anything when being instantiated.

Outputs

This section allows for specifying output parameters available to users after the template has been instantiated. This section is optional and can be omitted when no output values are required.

6.3.1 Adding parameters to the HOT template

In the HOT template, you can define parameters that are used by the applications being deployed. Example 6-2 shows the parameter defined for the example scenario. The parameter name is `WAS_Server_Name` and the default value is `was-server01`. You can also see in Figure 6-4 on page 166 and Figure 6-5 on page 167 how these parameters are presented to the user using the IBM Cloud Manager Dashboard.

Example 6-2 Adding parameters to the HOT template

```
parameters:
  WAS_Server_Name:
    type: string
    description: WAS server name
    default: was-server01
```

You can add parameters to the HOT template based on the cookbooks you have. Using parameters can be a significant benefit when the deployment requires several input parameters. Parameters make it easier for a user to deploy the HOT template, and most of the fields will already be filled with default values.

6.3.2 Adding resources to the HOT template

Resources are also a part of the HOT template. Resources include the instances that will be deployed and information such as the network where resources will be connected to, name, image that will be installed, flavor, and so on. Example 6-3 shows how resources can be defined for the WebSphere Application Server. This is part of the example template deployed in the following sections.

Example 6-3 Adding resource to the HOT template

```
WAS_server:
    type: OS::Nova::Server
    depends_on: DB2_wait_condition
    properties:
      networks:
```

```
        - port: { get_resource: WAS_server_net_port  }
      name: { get_param: WAS_Server_Name }
      image: 189e8b79-6749-4dba-a4c4-29aef3632ded  #rhel6.4 image
      flavor: { get_param: flavor }  #Size of the server to deploy
      config_drive: "True"
      user_data_format: RAW
      user_data: { get_resource: WAS_server_mime }
```

Note: Example 6-3 on page 163 shows how dependencies can be added to the resources. This example shows how the resource `WAS_server` depends on a `DB2_wait_condition`. This means that the `WAS_server` resource will not be created or configured if the DB2 configuration is not complete.

6.3.3 Running scripts in the instance

Another useful capability in the example HOT template is the option to run scripts as part of the deployment. The scripts can automate any tasks that you need to complete, including installation of applications.

Example 6-4 shows a script used in the example HOT template. The execution steps include the setup of hosts file entries, retrieval and installation of the Chef client, and some other sample commands used to complete the configuration of the DB2 server.

The scripts are passed to the instances by using the `cloud-init` software, so the metadata service or a `config_drive` must be used to run the scripts in the instance after its deployment. The HOT template must specify the `user_data` property in the instance resource section. The user_data must use the **`str_replace`** command to look for the template section of the `cloud-init` and pass the script.

Note: The `cloud-init` software package is a framework to handle early initialization of a cloud instance. It is supported by IBM Cloud Manager with OpenStack and can be used to run tasks such as setting a default locale and host name, and generating SSH private host keys. The Linux on z Systems images require the `cloud-init` software to be installed and configured.

Example 6-4 Adding scripts to HOT templates for software configurations

```
DB2_install:
    type: OS::Heat::SoftwareConfig
    properties:
      config:
        str_replace:
          params:
            ${WAIT-URL}: { get_resource: DB2_wait_handle }
            ${HOSTNAME}: { get_param: DB2_Server_Name }
          template: |
            #!/bin/bash
            # Setup hosts entries as DNS not implemnted.
            echo repoIP repoURL Host_name" >> /etc/hosts
            echo "127.0.0.1 $(hostname) $(hostname -f)" >> /etc/hosts
            # Retrieve and install the chef client
            curl http://repoURL:PORT/chef/chef-11.x.rpm -o /tmp/chef-11.x.rpm
            rpm -Uvh /tmp/chef-11.x.rpm
            curl http://repoURL:PORT/../chef-validator.pem -o /../validation.pem
```

```
# Mount install repo
mkdir -p /mnt/rhel64
mount NFS_server_IPAddress:/mnt/rhel64dvd /mnt/rhel64
# Start SSH
service sshd start
# Run the chef-client
chef-client
# Clean up environment after install
umount /mnt/rhel64
```

6.4 Using IBM Cloud Manager Dashboard to deploy patterns

After meeting all the prerequisites and creating your own HOT templates, you can deploy IBM software patterns. One of the approaches you can use to deploy Linux instances and software products is the IBM Cloud Manager Dashboard in IBM Cloud Manager with OpenStack for z Systems.

6.4.1 Deploying patterns

This section shows an example of deployment of a typical three-tier (DB2, WebSphere Application Server, and HTTP Server) pattern using the IBM Cloud Manager Dashboard. The three tiers are deployed to three Linux instances in a predefined sequence.

1. Log in to the IBM Cloud Manager Dashboard:

   `https://`*`your_cloud_controller_ip`*`/dashboard`

2. Click **Project** → **Orchestration** → **Stacks**.
3. Click **Launch Stack (**see Figure 6-2).

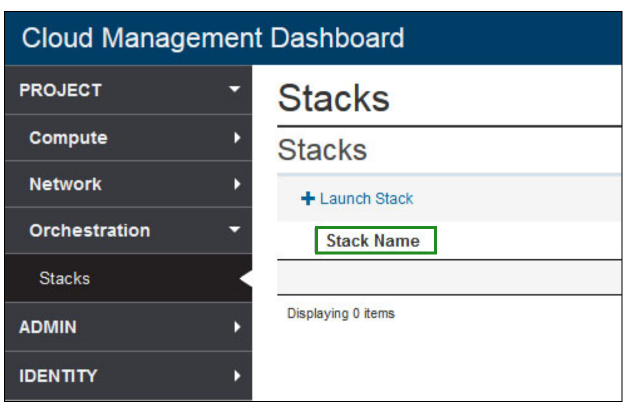


Figure 6-2 IBM Cloud Manager Dashboard - Launching a stack

The selected template window is displayed (see Figure 6-3).

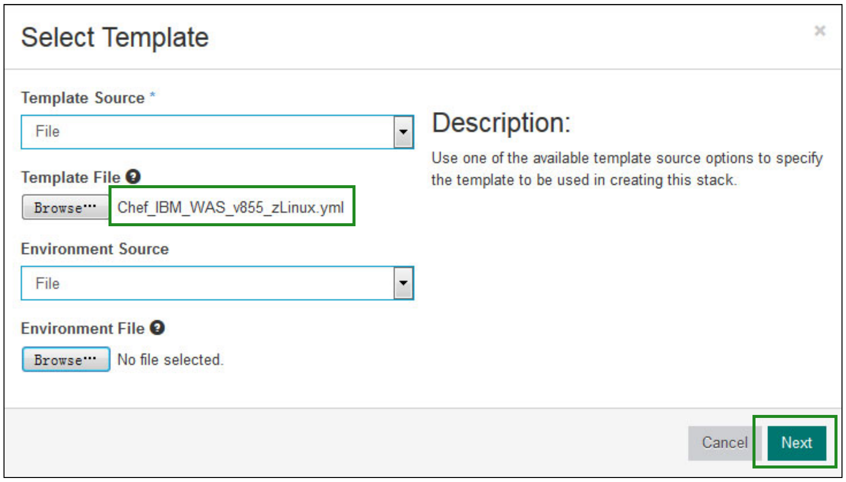


Figure 6-3 IBM Cloud Manager Dashboard - Selecting a template

4. Select **File** as the **Template Source**, and click **Browse** to select the HOT template that you want to deploy, which in this example is `Multi_Tier_z13_RHEL6.5.yml`.
5. Click **Next**.
6. In the Launch Stack window, provide values for all missing parameters or update the listed ones as shown in Figure 6-4.

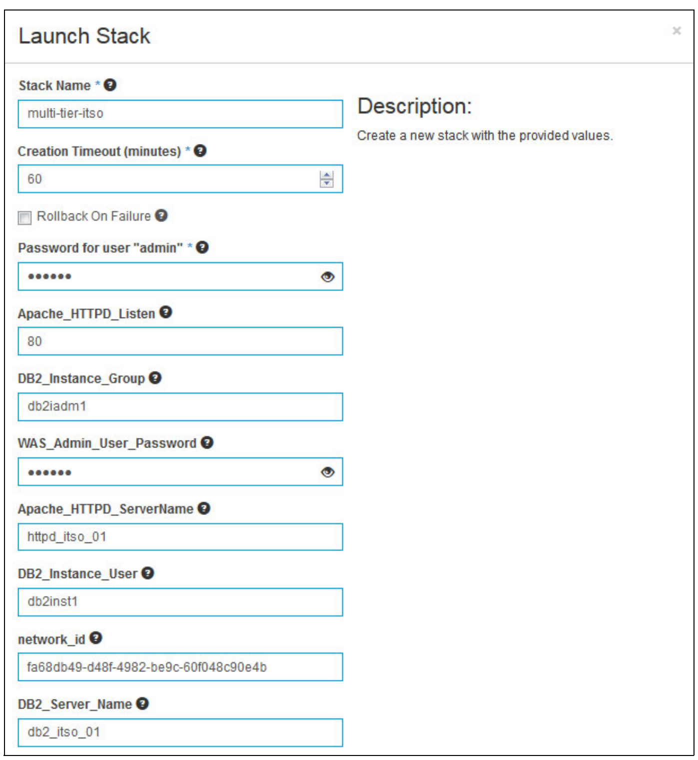


Figure 6-4 IBM Cloud Manager Dashboard - Launch stack part 1

7. Click **Launch (**see Figure 6-5).

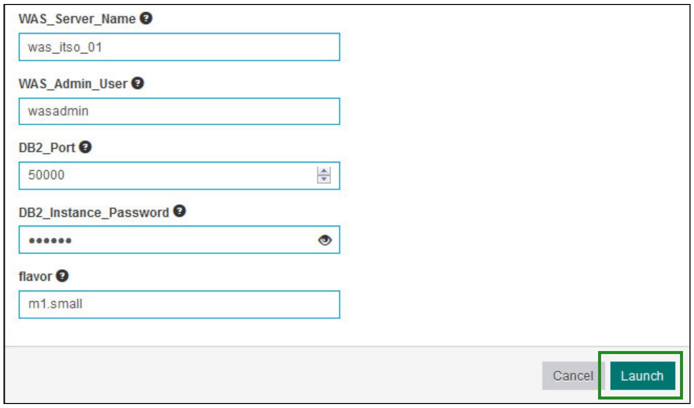


Figure 6-5 IBM Cloud Manager Dashboard - Launch stack part 2

8. After the process is started, you will see the status of the launched operation as shown in Figure 6-6.

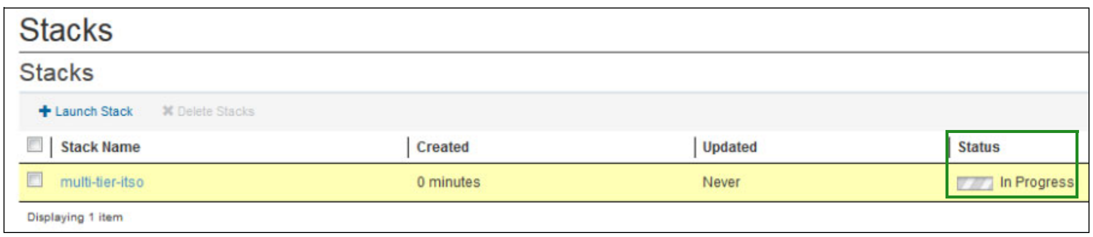


Figure 6-6 IBM Cloud Manager Dashboard - Stack creation status

6.4.2 Monitoring the progress of the deployment

This section describes two different views where you can verify the status of the components being created for your stack: The Stack Detail and Instance Details sections.

Using the Stack Detail view

After the process to create the stack is launched, you can monitor the status of the stack creation using these steps:

1. Click **Project** → **Orchestration** → **Stacks**.
2. Click stack name link, in this example `multi-tier-itso` (see Figure 6-7).

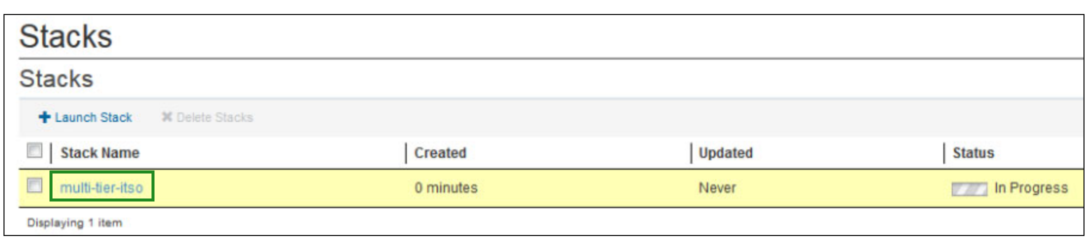


Figure 6-7 IBM Cloud Manager Dashboard - Stack creation status

The stack details are shown. In the first tab (Topology), an overview diagram showing the different components that will be created is displayed. You can click the different elements in the diagram to see the status for each one.

For example, Figure 6-8 shows selecting the `db2_server` node. The information for the node is shown in the left side of the image.

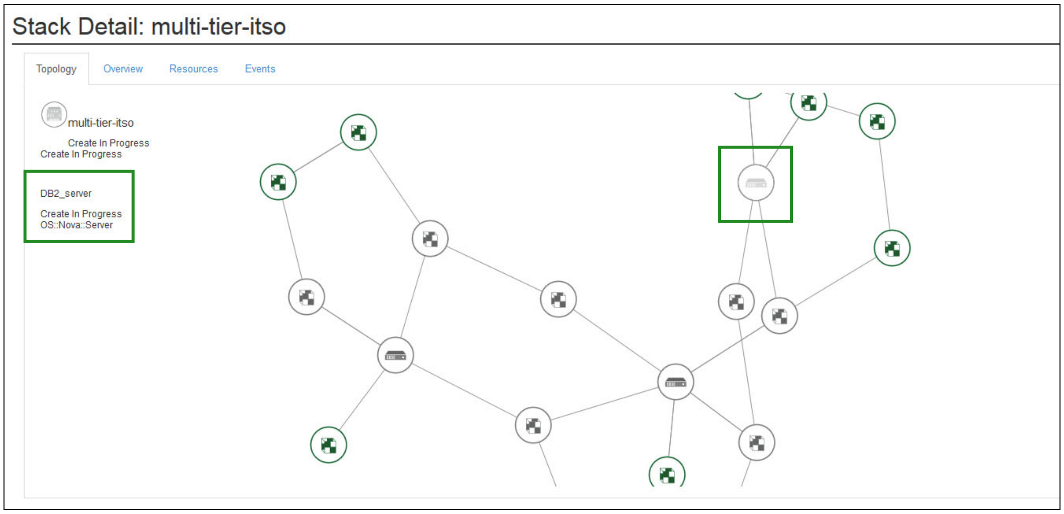


Figure 6-8 IBM Cloud Management Dashboard - Stack detail - Topology section

Note: In the Stack Detail - Topology window (shown on Figure 6-8), you might find component that are flashing. This means that the component is being created.

3. Click the Overview tab to review the details for the stack being created such as the status and the parameters used to create the stack (see Figure 6-9 on page 169).

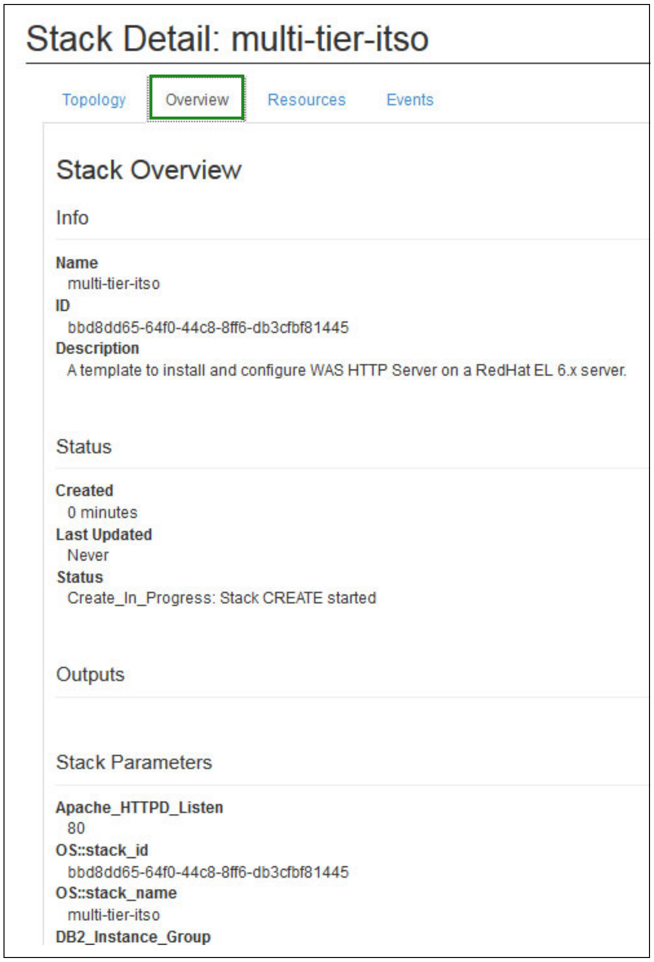


Figure 6-9 IBM Cloud Management Dashboard - Stack details - Overview section

4. Click the Resources tab for a list of all resources being created. This list includes information such as resource type, date updated, and status. Figure 6-10 shows how the resource information is displayed.

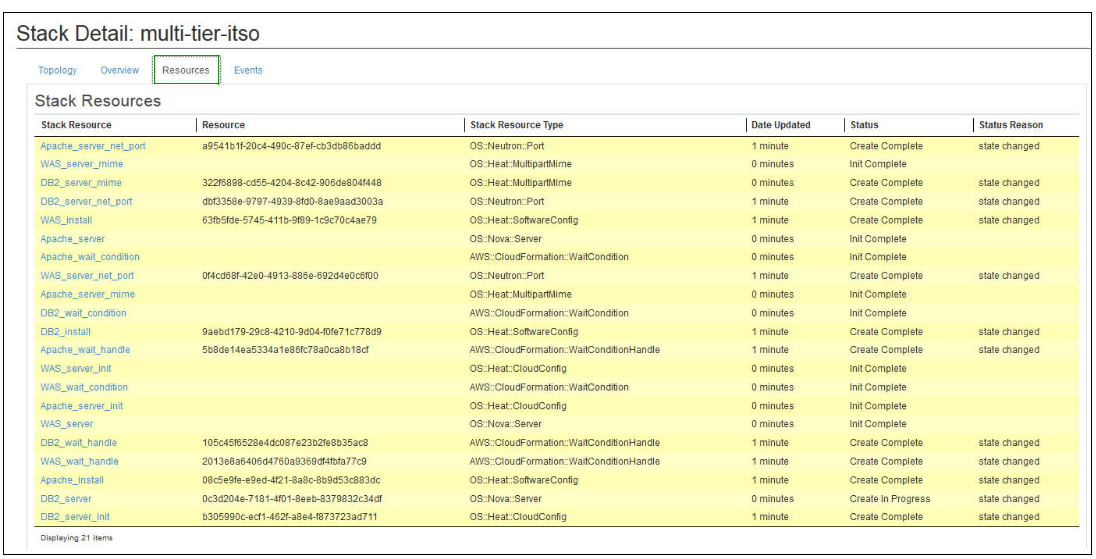


Figure 6-10 IBM Cloud Management Dashboard - Stack details resources section

5. Click the **Events** tab. You can review the details for each of the events being launched in the environment. These details include stack resource name, time since the event, status, and status reason (see Figure 6-11).

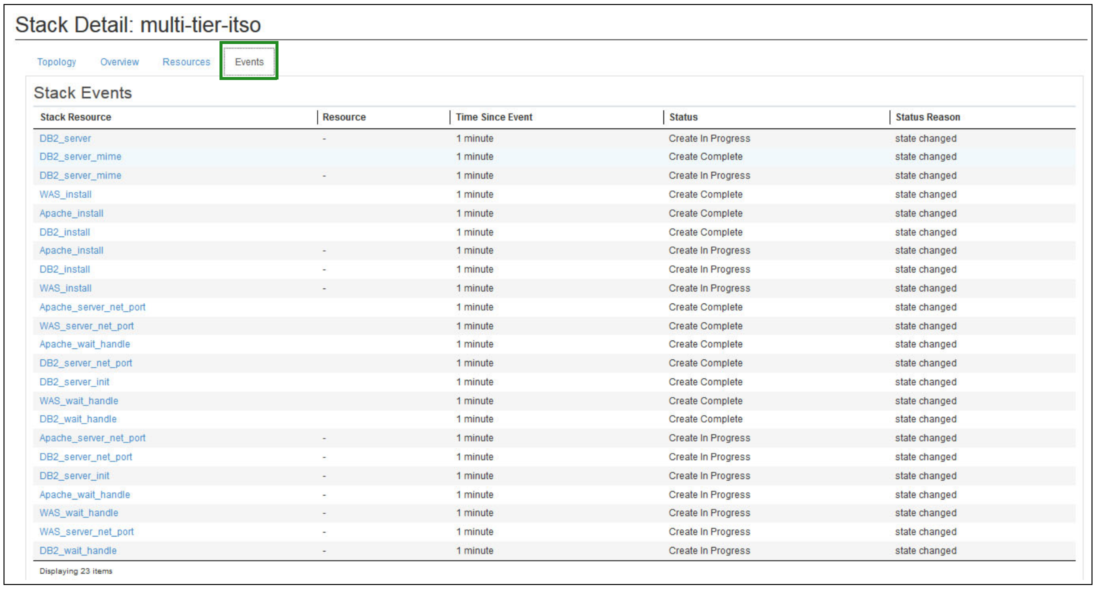


Figure 6-11 IBM Cloud Management Dashboard - Stack detail events section

Using the Instance Details view

When the deployment of the stack is in progress, you can review the details of the instances being created from the instances details section.

1. Click **Project** → **Compute** → **Instances**. In Figure 6-12 you can see how the details of the `db2` instance in the example environment are shown for `db2_itso_01`.

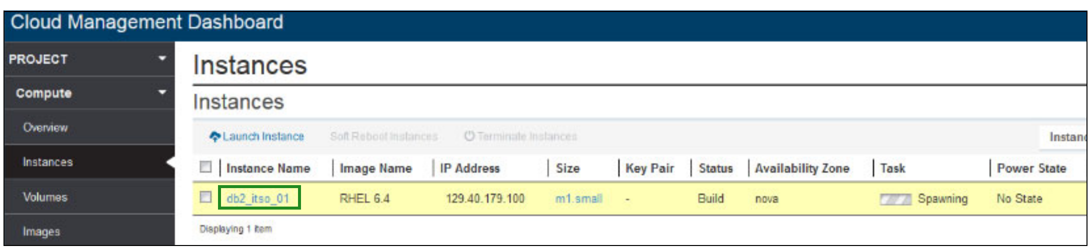


Figure 6-12 IBM Cloud Management Dashboard - Instances details

2. Click the instance name link as shown in Figure 6-12 to open the instance details section.

   The instance details section includes information that is distributed into four different tabs: Overview, Log, Console, and Action Log. On these tabs, you can review general details about the instance and the logs collected during creation and deployment.

Figure 6-13 shows the Overview tab of the Instance Details section where you can find different instance information such as name, status, uptime, flavor, and RAM.

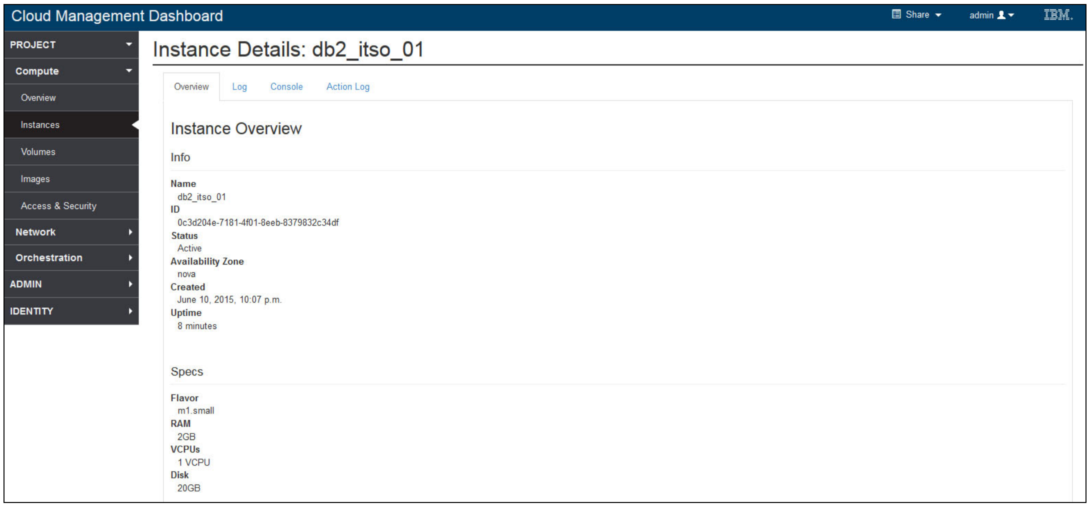


Figure 6-13 IBM Cloud Management Dashboard - Instance details overview section

After the creation of the instance is complete, you can see that the status is now *Active*.

3. Click **Project** → **Compute** → **Instances** as shown in Figure 6-14.

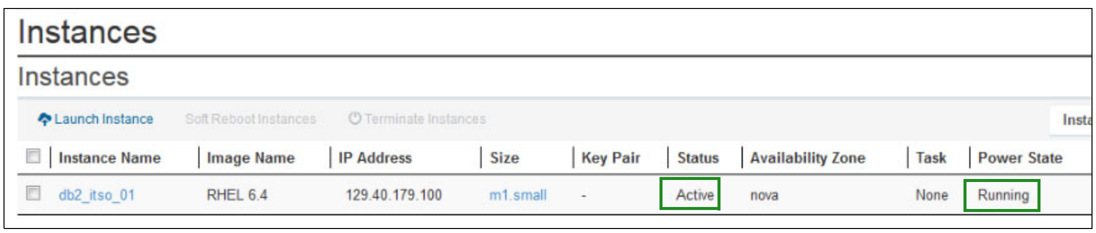


Figure 6-14 IBM Cloud Management Dashboard - Instance details creation completed

6.5 Using IBM UrbanCode Deploy with Patterns

In addition to the deployment of a pattern using HOT templates in IBM Cloud Management Dashboard, the patterns can be deployed using IBM UrbanCode Deploy with Patterns. IBM UrbanCode Deploy is an IBM tool for automating application deployments through your environments. It is designed to facilitate rapid feedback and continuous delivery in agile development while providing the audit trails, versioning, and approvals needed in production. IBM UrbanCode Deploy with Patterns uses the Heat engine to deploy patterns.This section provides the steps needed to deploy a pattern using IBM UrbanCode Deploy with Patterns.

Note: The following steps assume that you have already installed IBM UrbanCode Deploy with Patterns and that the system settings are updated to point to the Heat engine in the IBM Cloud Manager with OpenStack for z Systems controller node.

To use IBM UrbanCode Deploy with Patterns, complete these steps:

1. Open the IBM UrbanCode Deploy with Patterns Login window, provide your *user name* and *password* and click **Log In** to access the IBM UrbanCode Deploy with Patterns application (see Figure 6-15).

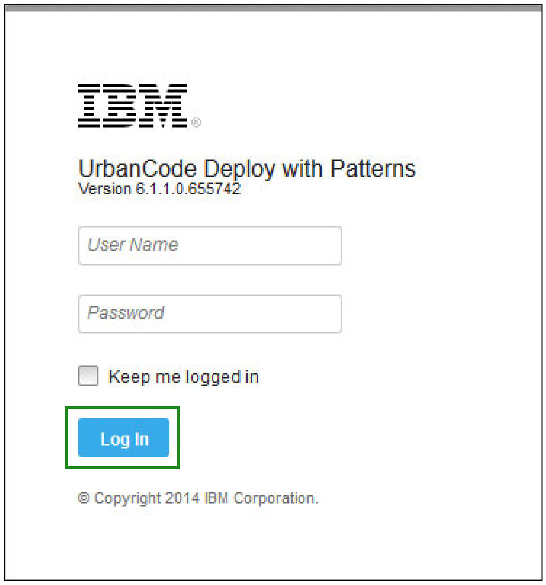


Figure 6-15 IBM UrbanCode Deploy with Patterns - Login window

The IBM UrbanCode Deploy with Patterns home page is displayed.

2. From the Blueprints list shown in the home page, click the link name for the one that you want to provision in our example we use *multitier* (see Figure 6-16 on page 173).

Note: This example uses the previously saved multitier blueprint in IBM UrbanCode Deploy with Patterns user interfac. You have two options to upload your blueprints:

- Use the *blueprints editor*. This use case creates the content of the HOT template that you want to use with the editor and saves it as a blueprint named *multitier*. The included editor offers useful tools such as the syntax reviewer for the HOT template.
- Use a *Git repository*. After creating your HOT templates you can put them into a Git repository. IBM UrbanCode Deploy with Patterns supports Git repositories to control different versions of HOT.

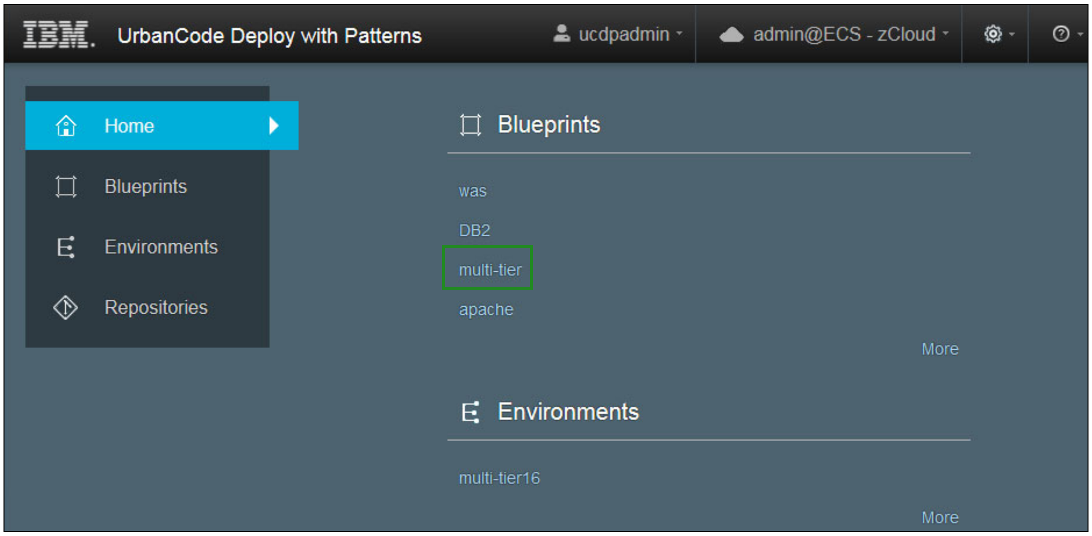


Figure 6-16 IBM UrbanCode Deploy with Patterns - Home page

In this window, you can review how the three tier application that you plan to deploy is represented in the IBM UrbanCode Deploy with Patterns diagram. Each node will be an instance running the software that you require in your deployment environment: The HTTP server, the WebSphere Application Server, and the DB2 server (see Figure 6-17).

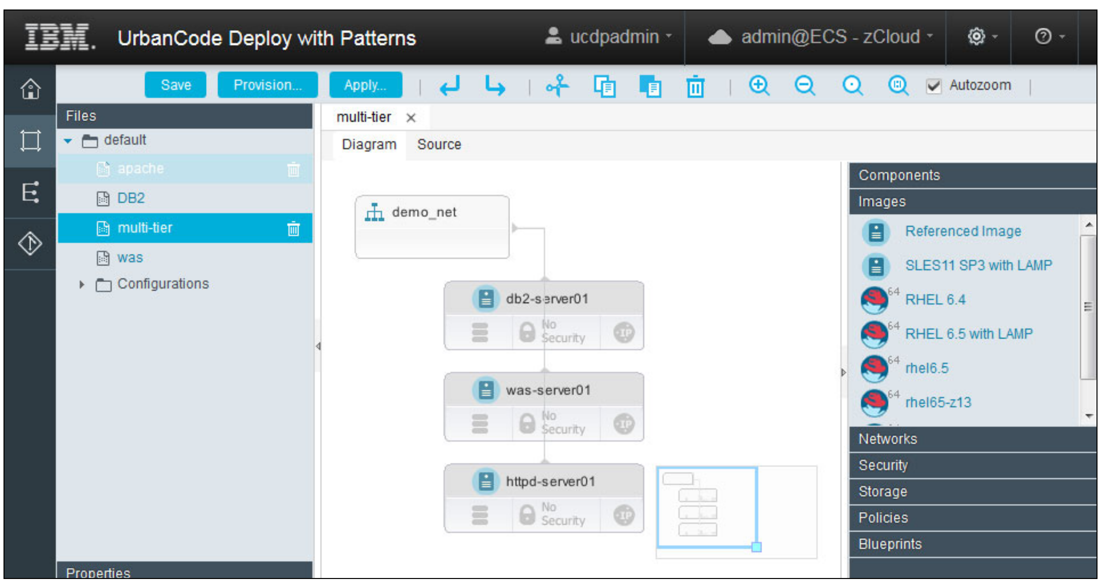


Figure 6-17 IBM UrbanCode Deploy with Patterns - Blueprints diagram

3. Click the Source tab to see the content of the HOT template, then click **Provision** to start the deployment (see Figure 6-18).

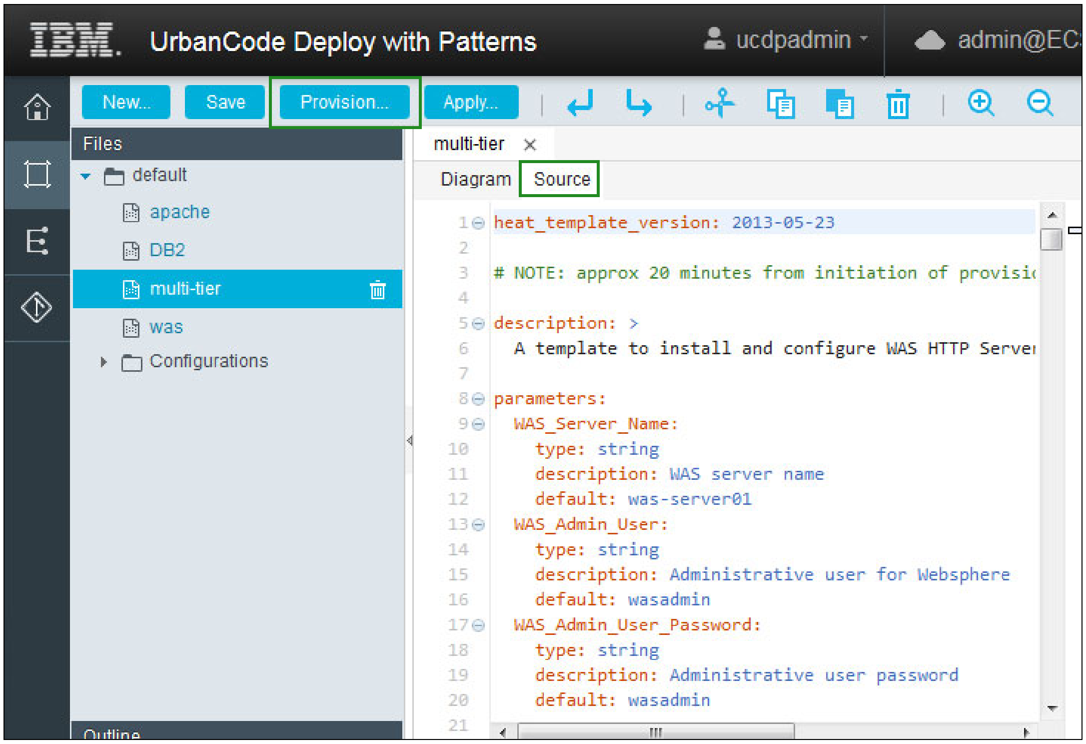


Figure 6-18 IBM UrbanCode Deploy with Patterns - Blueprints HOT source view

4. On the provision window, verify and complete all required parameters for the deployment, then click **Provision** (see Figure 6-19).

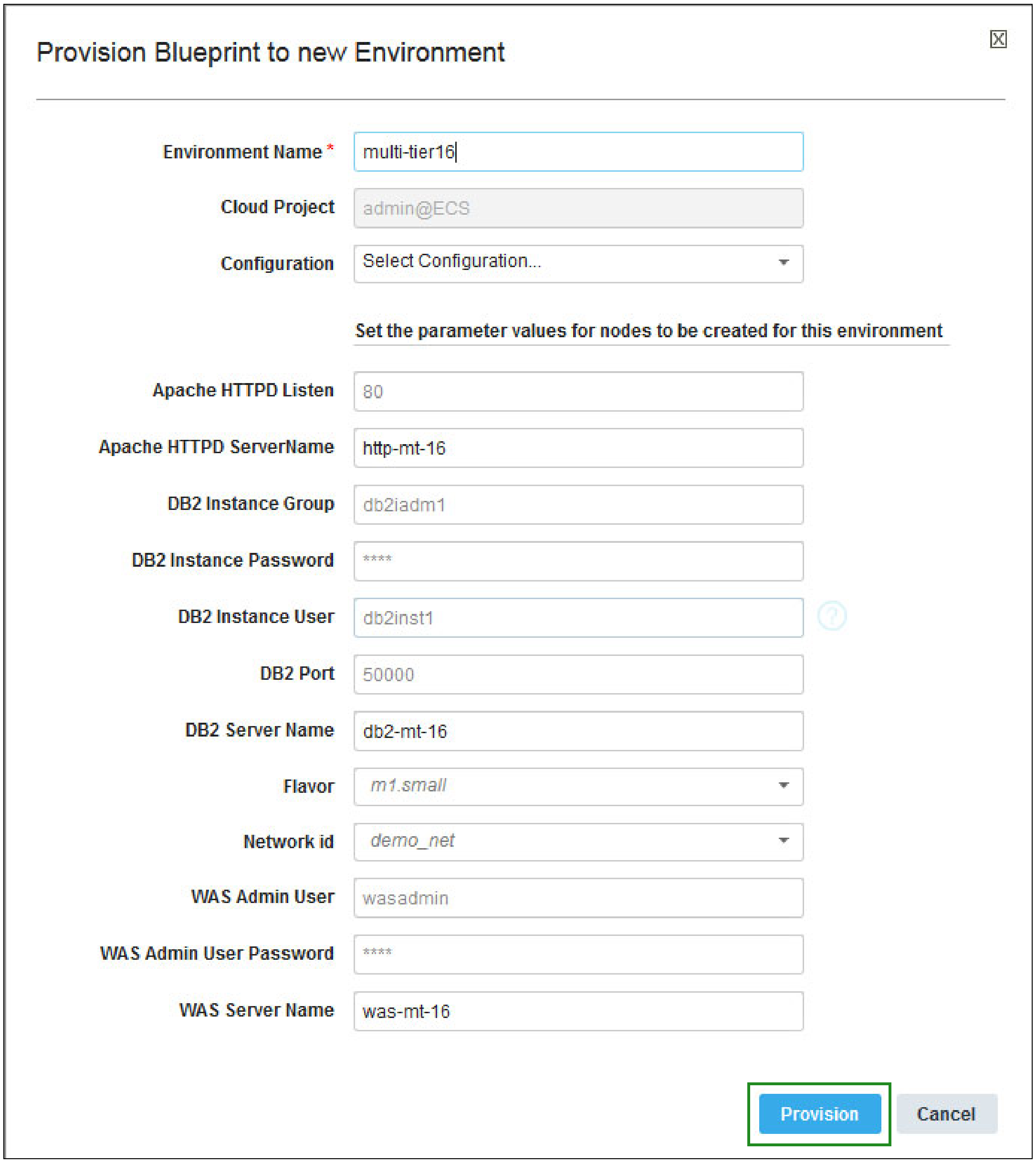


Figure 6-19 IBM UrbanCode Deploy with Patterns - Provisioning window

5. After the deployment starts, you can click **Environments** as shown in Figure 6-20 to review the status of the provisioning process.

Figure 6-20 IBM UrbanCode Deploy with Patterns - Environments menu

In the Environments view, you can see that the stack creation has started (see Figure 6-21).

Figure 6-21 IBM UrbanCode Deploy with Patterns - Provisioning started

6. Click the environment link name, in this case `multi-tier16` (see Figure 6-21). The resources and events list being created for the deployment of this environment are shown in Figure 6-22.

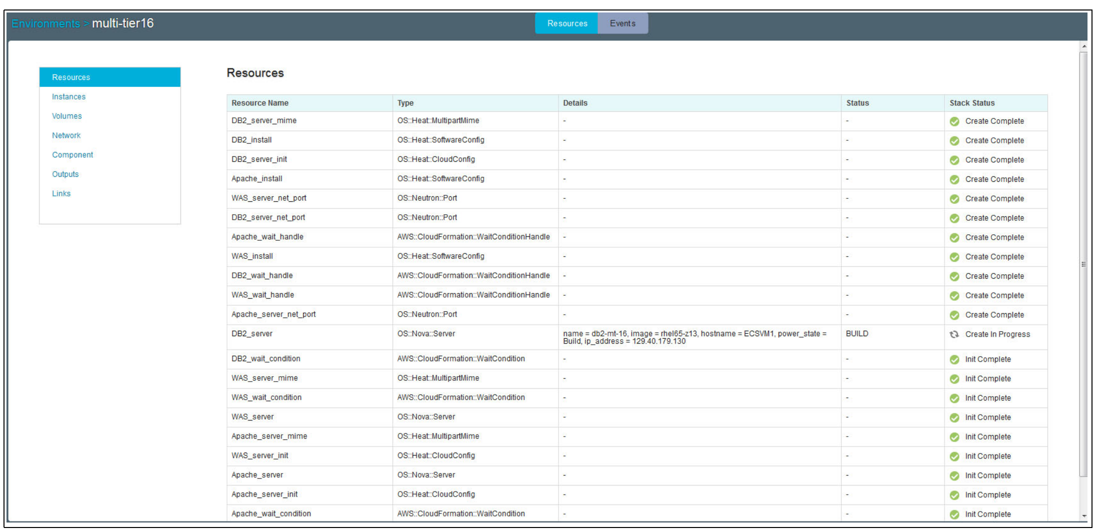
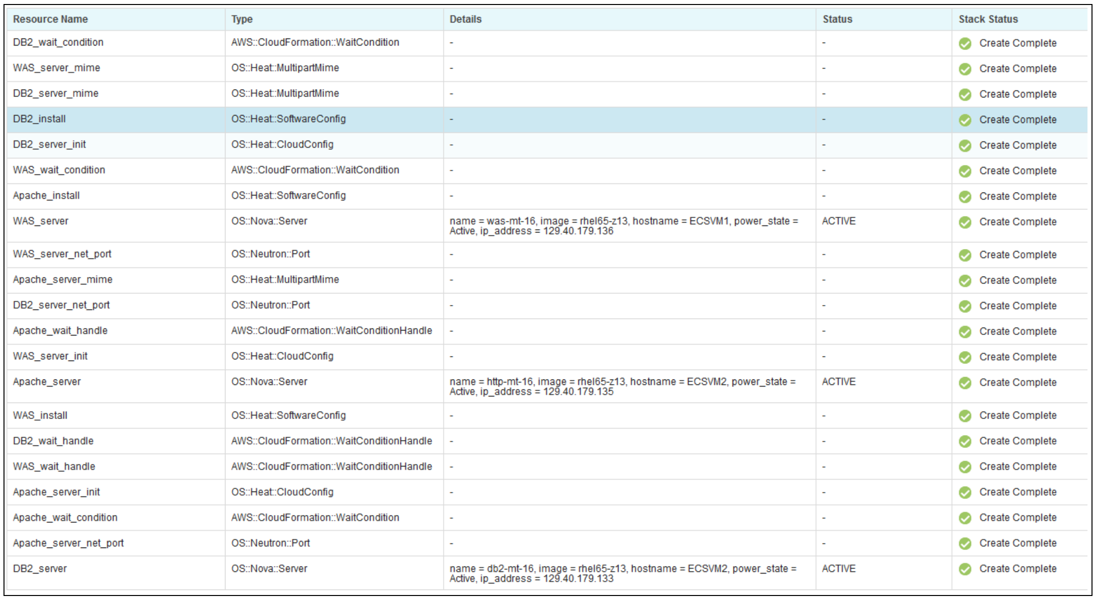


Figure 6-22 IBM UrbanCode Deploy with Patterns - Environment creation in progress

In Figure 6-22 on page 176 you can see that the creation for some of the resources are still in progress. After all resources are created, you can see that the status is marked as completed as shown on Figure 6-23.

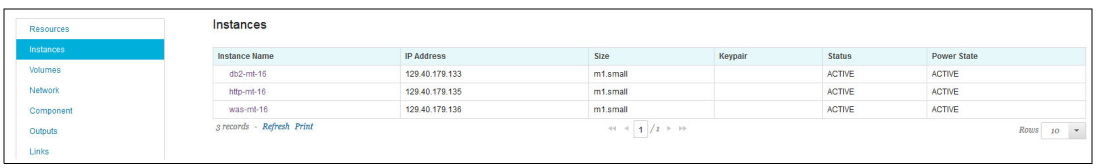


Resource Name	Type	Details	Status	Stack Status
DB2_wait_condition	AWS::CloudFormation::WaitCondition	-	-	Create Complete
WAS_server_mime	OS::Heat::MultipartMime	-	-	Create Complete
DB2_server_mime	OS::Heat::MultipartMime	-	-	Create Complete
DB2_install	OS::Heat::SoftwareConfig	-	-	Create Complete
DB2_server_init	OS::Heat::CloudConfig	-	-	Create Complete
WAS_wait_condition	AWS::CloudFormation::WaitCondition	-	-	Create Complete
Apache_install	OS::Heat::SoftwareConfig	-	-	Create Complete
WAS_server	OS::Nova::Server	name = was-mt-16, image = rhel65-z13, hostname = ECSVM1, power_state = Active, ip_address = 129.40.179.136	ACTIVE	Create Complete
WAS_server_net_port	OS::Neutron::Port	-	-	Create Complete
Apache_server_mime	OS::Heat::MultipartMime	-	-	Create Complete
DB2_server_net_port	OS::Neutron::Port	-	-	Create Complete
Apache_wait_handle	AWS::CloudFormation::WaitConditionHandle	-	-	Create Complete
WAS_server_init	OS::Heat::CloudConfig	-	-	Create Complete
Apache_server	OS::Nova::Server	name = http-mt-16, image = rhel65-z13, hostname = ECSVM2, power_state = Active, ip_address = 129.40.179.135	ACTIVE	Create Complete
WAS_install	OS::Heat::SoftwareConfig	-	-	Create Complete
DB2_wait_handle	AWS::CloudFormation::WaitConditionHandle	-	-	Create Complete
WAS_wait_handle	AWS::CloudFormation::WaitConditionHandle	-	-	Create Complete
Apache_server_init	OS::Heat::CloudConfig	-	-	Create Complete
Apache_wait_condition	AWS::CloudFormation::WaitCondition	-	-	Create Complete
Apache_server_net_port	OS::Neutron::Port	-	-	Create Complete
DB2_server	OS::Nova::Server	name = db2-mt-16, image = rhel65-z13, hostname = ECSVM2, power_state = Active, ip_address = 129.40.179.133	ACTIVE	Create Complete

Figure 6-23 IBM UrbanCode Deploy with Patterns - Environment creation completed

7. Click **Instance** from the left menu.

   The details for the three instances are shown, including instance name, IP address, size, status, and power state (see Figure 6-24).



Instance Name	IP Address	Size	Keypair	Status	Power State
db2-mt-16	129.40.179.133	m1.small		ACTIVE	ACTIVE
http-mt-16	129.40.179.135	m1.small		ACTIVE	ACTIVE
was-mt-16	129.40.179.136	m1.small		ACTIVE	ACTIVE

Figure 6-24 IBM UrbanCode Deploy with Patterns - Instance status completed

8. Click **Outputs** from the left menu.

   The details about the deployment outputs are shown, including the HTTP server URL and the WAS server URL (see Figure 6-25).

Name	Description	Value
Blueprint_url	Blueprint Origin URL	https://129.40.179.249:443/landscaper/view/projects?open=ucdpadmin_00000000_0000_0000_0000_000000000002-OrionContent/Demo_Assets/Multi_Tier/Multi_Tier.yml
DB2_Details	IBM DB2 Enterprise Server Edition v10.5	Server db2-mt-16 (129.40.179.133) is configured with instance db2inst1 which is running on port 50000.
Apache_url	Apache HTTP Server URL	http://129.40.179.135:80
WAS_url	WAS Console URL	https://129.40.179.136:9043/ibm/console

Figure 6-25 IBM UrbanCode Deploy with Patterns - Outputs view

9. Go back to the Environments view as shown in Figure 6-20 on page 176.

   Now, you can see that the creation has completed successfully and all instances statuses are *ACTIVE* (see Figure 6-26).

Figure 6-26 IBM UrbanCode Deploy with Patterns - Stack overall status view

Finally, you can also check the status of the instances from the IBM Cloud Manager with the OpenStack self-service portal. The information is synchronized and you can see the status updates (see Figure 6-27).

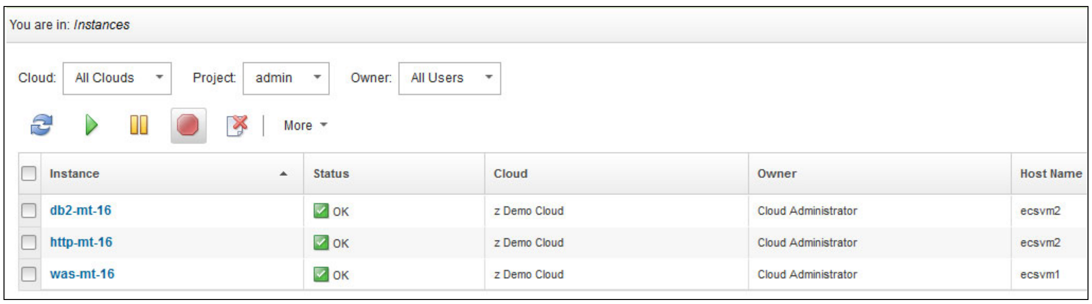


Figure 6-27 Instance status view from selef service portal

A

Managing a distributed cloud environment from z Systems

You can use IBM Cloud Manager with OpenStack for z Systems to manage distributed platforms from z Systems, including x86 and IBM POWER® platforms. This appendix describes an example of how to manage KVM on x86 platform with a single OpenStack controller from z Systems. Managing the Power platform with multi-region has a similar approach.

This appendix includes the following sections:

- x86 environment preparation
- Configure Chef environment and topology files
- Verify the heterogeneous cloud infrastructure

A.1 x86 environment preparation

IBM Cloud Manager with OpenStack for z Systems V4.2 uses a single controller node and multiple compute nodes topology. This means the OpenStack controller on z Systems manages the compute services on x86. The example in this appendix manages one KVM hypervisor on an x86 server and only shows the process of managing x86 platform from z Systems. And for simplicity, the example uses a GRE network instead of Open vSwitch on x86 server.

To manage x86 servers, the following minimum requirements should be met:

- Red Hat Enterprise Linux 6.5
  - Also, configure the yum repository to connect to the RHEL 6.5 repository
- A libvirt version 0.9.11 or later
- Python version 2.6.x.
- If you want to use Open vSwitch in your environment, Open vSwitch 2.0 should be installed and configured.
  - Update the host device drivers to the latest level to work with Open vSwitch

> **Note:** For more information, see the "KVM prerequisites" section in IBM Cloud Manager with OpenStack library:
>
> http://www.ibm.com/support/knowledgecenter/SST55W_4.2.0/liaca/liacasysxprereqs.html?lang=en

A.2 Configure Chef environment and topology files

As described in 3.3.11, "Install IBM Cloud Manager with OpenStack Deployer" on page 64, you can deploy compute services to new platforms after the deployer is installed. This section describes the steps to configure the Chef environment and topology files to deploy new compute services to an x86 hypervisor.

> **Note:** The configuration is performed by using the controller node (xCAT MN) SSH session. So you need to log in to the xCAT MN with `mnadmin`.

A.2.1 Prepare Chef environment and topology files

- Create a directory for related files. The example environment uses a directory named *kvm_env*.
- Create a x86 compute node topology *topology.json* file in the directory as shown in Example A-1.

Example A-1 Sample topology.json file

```
{

 "name": "kvm-topology",

  "environment": "kvmenv",
```

```
  "secret_file": "/opt/ibm/cmwo/chef-repo/data_bags/example_data_bag_secret",

  "nodes": [

{
      "fqdn": "192.168.60.15",
      "password": "passwd",
      "run_order_number": 0,
      "quit_on_error": true,
      "runlist": [
        "role[ibm-os-compute-node-kvm]"
      ]

    }

  ]

}
```

- Get a sample environment file called `kvm.json` as shown in Example A-2.

Example A-2 Get a sample environment file kvm.json

```
[mnadmin@xcat kvm_en] $ sudo knife environment show
example-ibm-os-single-controller-n-compute -d -Fjson > kvm.json
```

A.2.2 Configure the Chef environment and topology files

Because the x86 compute services share the same controller on z Systems, the password in the keystone should be kept unchanged. In the Chef environment, use data bag to make sure that the passwords are configured properly. To configure the environment and files, complete these steps:

1. Get a data bag copy from the Chef server by using the command shown in Example A-3.

Example A-3 Get a data bag copy

```
[mnadmin@xcat kvm_env] $ sudo knife os manage get passwords --topology-file
topology.json data_bags
Passwords and secrets written to the data_bags directory.
[mnadmin@xcat kvm_env] $ ls data_bags
db_passwords  kvmenv_passwords_file.json  secrets  service_passwords
user_passwords
```

2. After you get the data bag, you can see a `data_bags` directory has been created on your disk. Because the directory owner is *root*, you need to **su** to the root user to change the data bag passwords.
3. Within `data_bag/secrets`, update `openstack_identity_bootstrap_token.json`. The `openstack_identity_bootstrap_token` attribute must match the `admin_token` attribute that is defined in the `/etc/keystone/keystone.conf` file.

For the example, change the attribute in the `openstack_identity_bootstrap_token.json` file as shown in Example A-4.

Example A-4 Change openstack_identity_bootstrap_token.json file

```
{
  "id": "openstack_identity_bootstrap_token",
  "openstack_identity_bootstrap_token": "882e4ab9f8a0aa128d0c"
}
```

4. Within `data_bags/service_passwords`, update attributes in the following files:
   - `openstack-block-storage.json`
   - `openstack-ceilometer.json`
   - `openstack-compute.json`
   - `openstack-image.json`
   - `openstack-network.json`
   - `openstack-iaas-gateway.json`

   For example, change the openstack-compute.json file as shown in Example A-5.

Example A-5 openstack-compute.json

```
{
  "id": "openstack-compute",
  "openstack-compute": "passwd"
}
```

Note: In the IBM Cloud Manager with OpenStack for z Systems, the passwords for all the data bag files are the same. The password value is the `cmo_admin_password` in the DMSSICMO file.

- Within `data_bags/user_passwords`, the `qpidclient` attribute within the `qpidclient.json` file must match what is defined in the controller. The password is the same value that is defined in `cmo_admin_password` in the DMSSICMO file. For example, the qpidclient.json file contains the value shown in Example A-6.

Example A-6 qpidclient.json

```
{
  "id": "qpidclient",
  "qpidclient": "passwd"
}
```

- After you change the data bag files, issue the command that is shown in Example A-7 to update the data bags to the Chef server.

Example A-7 Update data bags to Chef server

```
[mnadmin@xcat kvm_env] $ sudo knife os manage update passwords --topology-file
topology.json data_bags
Passwords updated and encrypted in data bag 'service_passwords'.
Passwords updated and encrypted in data bag 'db_passwords'.
Passwords updated and encrypted in data bag 'user_passwords'.
Passwords updated and encrypted in data bag 'secrets'.
Passwords and secrets have been updated based on the data_bags directory. The
deployment needs to be updated to apply the changes.
```

A.2.3 Customize and update the Chef environment

This section details how to customize and update your Chef environment.

A.2.3.1 Customize the Chef environment file

Customize the Chef environment file according to your controller configurations and the x86 managed environment that you want. An example `kvm.json` environment is shown in Example A-8.

Example A-8 kvm.json environment

```
{
  "name": "kvmenv",
  "description": "Example environment for the IBM OpenStack single controller + n
compute topology",
  "cookbook_versions": {
    "apache2": "~> 1.9.6",
    "apt": "~> 2.3.8",
    "aws": "~> 1.0.0",
    "build-essential": "~> 1.4.2",
    "chef_handler": "~> 1.1.5",
    "database": "~> 2.3.0",
    "erlang": "~> 1.4.2",
    "homebrew": "~> 1.5.4",
    "iptables": "~> 0.13.2",
    "logrotate": "~> 1.5.0",
    "memcached": "~> 1.7.2",
    "mysql": "~> 5.4.4",
    "mysql-chef_gem": "~> 0.0.2",
    "ntp": "~> 1.5.4",
    "openssl": "~> 1.1.0",
    "pacman": "~> 1.0.4",
    "postgresql": "~> 3.3.4",
    "python": "~> 1.4.6",
    "rabbitmq": "~> 3.3.0",
    "runit": "~> 1.5.10",
    "selinux": "~> 0.8.0",
    "windows": "~> 1.30.0",
    "xfs": "~> 1.1.0",
    "yum": "~> 3.1.4",
    "yum-epel": "~> 0.3.4",
    "yum-erlang_solutions": "~> 0.1.4",
    "yum-mysql-community": "~> 0.1.10",
    "openstack-block-storage": "~> 10.0.1",
    "openstack-common": "~> 10.2.0",
    "openstack-compute": "~> 10.0.0",
    "openstack-dashboard": "~> 10.0.0",
    "openstack-identity": "~> 10.0.1",
    "openstack-image": "~> 10.0.0",
    "openstack-network": "~> 10.1.1",
    "openstack-ops-database": "~> 10.0.0",
    "openstack-ops-messaging": "~> 10.0.1",
    "openstack-orchestration": "~> 10.0.0",
    "openstack-telemetry": "~> 10.0.1",
    "db2": "~> 1.0.6",
```

```
    "ibm-openstack-apache-proxy": "~> 10.0.1",
    "ibm-openstack-common": "~> 10.1.11",
    "ibm-openstack-iaas-gateway": "~> 10.0.1",
    "ibm-openstack-iptables": "~> 10.0.0",
    "ibm-openstack-network": "~> 10.2.4",
    "ibm-openstack-powervc-driver": "~> 10.1.3",
    "ibm-openstack-prs": "~> 10.1.0",
    "ibm-openstack-roles": "~> 10.0.2",
    "ibm-openstack-simple-token": "~> 10.0.0",
    "ibm-openstack-yum-server": "~> 10.0.1",
    "ibm-openstack-zvm-appliance": "~> 10.0.0",
    "ibm-openstack-zvm-driver": "~> 10.2.2",
    "ibm-sce": "~> 10.1.5",
    "qpid": "~> 1.0.2"
  },
  "json_class": "Chef::Environment",
  "chef_type": "environment",
  "default_attributes": {
    "selinux": {
      "state": "nothing"
    },
    "ibm-openstack": {
      "iptables": {
        "status": "enabled",
        "use_default_rules": true,
        "include_ssh_default_rule": true,
        "custom_rules": null
      },
      "zvm-driver": {
        "hosts": [
          "host1"
        ],
        "host1": {
          "xcat": {
            "server": "0.0.0.0",
            "username": "",
            "zhcp_nodename": "",
            "master": "",
            "mgt_ip": "0.0.0.0",
            "mgt_mask": "255.255.255.0",
            "connection_timeout": "3600",
            "image_clean_period": "30",
            "free_space_threshold": "50",
            "timeout": "300"
          },
          "config": {
            "ram_allocation_ratio": "3"
          },
          "image": {
            "tmp_path": "/var/lib/nova/images",
            "cache_manager_interval": "86400"
          },
          "rpc_response_timeout": "180",
          "reachable_timeout": "600",
          "polling_interval": "5",
```

```
          "config_drive": {
            "format": "tgz",
            "inject_password": "False"
          },
          "ml2": {
            "type_drivers": "local,flat,vlan,gre,vxlan",
            "tenant_network_types": "vlan",
            "mechanism_drivers": "zvm",
            "flat_networks": "default",
            "network_vlan_ranges": "default:1:4094"
          },
          "diskpool": "",
          "diskpool_type": "",
          "zvm_host": "",
          "host": "",
          "user_profile": "",
          "scsi_pool": "",
          "fcp_list": "",
          "zhcp_fcp_list": "",
          "external_vswitch_mappings": ""
        }
      },
      "network": {
        "ip_movement": {
          "enable": true,
          "wait_before_ping": 5,
          "timeout": 120
        }
      }
    },
    "memcached": {
      "maxconn": 1024,
      "memory": 64,
      "max_object_size": "1m"
    },
    "openstack": {
      "endpoints": {
        "network-openvswitch": {
          "bind_interface": "eth0"
        },
        "compute-vnc-bind": {
          "bind_interface": "eth0"
        }
      },
      "compute": {
        "debug": "False",
        "verbose": "True",
        "state_path": "/var/lib/nova",
        "driver": "libvirt.LibvirtDriver",
        "instance_name_template": "itso-%04x",
        "libvirt": {
          "virt_type": "kvm",
          "live_migration_flag":
"VIR_MIGRATE_UNDEFINE_SOURCE,VIR_MIGRATE_PEER2PEER,VIR_MIGRATE_LIVE"
        }
```

```
      },
      "network": {
        "verbose": "False",
        "debug": "False",
        "openvswitch": {
          "bridge_mappings": "",
          "bridge_mapping_interface": ""
        }
      },
      "block-storage": {
        "debug": "False",
        "verbose": "True",
        "volume": {
          "create_volume_group": false,
          "volume_group_size": 40,
          "iscsi_ip_address": "127.0.0.1"
        }
      },
      "telemetry": {
        "debug": "False",
        "verbose": "True"
      }
    }
  },
  "override_attributes": {
    "ntp": {
      "servers": [
        "0.pool.ntp.org",
        "1.pool.ntp.org",
        "2.pool.ntp.org",
        "3.pool.ntp.org"
      ]
    },
    "ibm-openstack-yum-server": {
      "yumpriorities_enabled": false
    },
    "ibm-openstack": {
      "remote_files": {
        "mysql_gem_download_enabled": true,
        "mysql_gem_filename": "mysql-2.9.1.gem"
      },
      "password-obfuscate": true,
      "prs": {
        "ego": {
          "master_list": [

          ]
        }
      },
      "powervc-driver": {
        "powervc": {
          "admin_user": "root",
          "host": "powervc_host",
          "rabbitmq": {
            "username": "powervcdriver_mq",
```

```
        "ssl_keyfile_source": "/root/powervcmq_ssl/key.pem",
        "ssl_certfile_source": "/root/powervcmq_ssl/cert.pem",
        "ssl_ca_certs_source": "/root/powervcmq_ssl/cacert.pem"
      },
      "mq": {
        "service_type": "rabbitmq"
      },
      "ssh": {
        "access": true
      },
      "scg": [
        "Any host, all VIOS"
      ]
    },
    "db": {
      "max_pool_size": 10,
      "max_overflow": 20
    },
    "db_create": false
  },
  "iaas-gateway": {
    "listen_port": 9973,
    "keystone_ca_certs": null,
    "keystone_insecure": true,
    "logging": {
      "enabled": false,
      "debug": false
    }
  },
  "upgrade": {
    "inplace": false
  },
  "apache-proxy": {
    "enabled": false,
    "certfile": "/etc/certs/cert.pem",
    "certkeyfile": "/etc/certs/certkey.pem"
  },
  "network": {
    "l3": {
      "enable": true
    }
  }
},
"openstack": {
  "release": "juno",
  "region": "zCloud",
  "auth": {
    "strategy": "uuid"
  },
  "yum": {
    "rdo_enabled": false
  },
  "developer_mode": false,
  "use_databags": true,
  "databag_type": "encrypted",
```

```
"secret": {
  "key_path": "/etc/chef/encrypted_data_bag_secret",
  "service_passwords_data_bag": "service_passwords",
  "db_passwords_data_bag": "db_passwords",
  "user_passwords_data_bag": "user_passwords",
  "secrets_data_bag": "secrets"
},
"db": {
  "service_type": "mysql",
  "server_role": "ibm-os-database-server-node",
  "telemetry": {
    "db_name": "ceilodb2",
    "username": "ceilodb2",
    "port": "50000",
    "nosql": {
      "used": true
    },
    "options": {
      "nosql": "?socketTimeoutMS=20000"
    }
  },
  "root_user_use_databag": true
},
"memcached_servers": [
  "localhost:11211"
],
"mq": {
  "service_type": "qpid",
  "server_role": "ibm-os-messaging-server-node",
  "durable_queues": true,
  "user": "qpidclient",
  "image": {
    "notification_topic": "notifications"
  },
  "qpid": {
    "protocol": "ssl"
  },
  "rabbitmq": {
    "use_ssl": true
  }
},
"endpoints": {
  "host": "192.168.60.101",
  "identity-admin": {
    "port": "35357"
  },
  "identity-internal": {
    "port": "5000"
  },
  "identity-api": {
    "port": "5000"
  },
  "network-api": {
    "port": "9696"
  },
```

```
    "image-api": {
      "port": "9292"
    },
    "orchestration-api": {
      "port": "8004"
    },
    "orchestration-api-cfn": {
      "port": "8000"
    },
    "orchestration-api-cloudwatch": {
      "port": "8003"
    },
    "telemetry-api": {
      "port": "8777"
    },
    "compute-api": {
      "port": "8774"
    },
    "block-storage-api": {
      "port": "8776"
    },
    "compute-novnc": {
      "scheme": "http"
    },
    "bind-host": "192.168.60.101",
    "mq": {
      "host": "192.168.60.101",
      "port": "5671"
    },
    "db": {
      "host": "192.168.60.101",
      "port": "50001"
    }
  },
  "identity": {
    "verbose": "False",
    "debug": "False",
    "public_workers": null,
    "admin_workers": null,
    "token": {
      "hash_algorithm": "md5"
    }
  },
  "image": {
    "control_exchange": "glance",
    "verbose": "False",
    "debug": "False",
    "notification_driver": "messaging",
    "filesystem_store_datadir": "/var/lib/glance/images",
    "api": {
      "auth": {
        "memcached_servers": null,
        "memcache_security_strategy": null,
        "memcache_secret_key": null,
        "hash_algorithms": "md5",
```

```
            "cafile": null,
            "insecure": false
          },
          "block-storage": {
            "cinder_catalog_info": "volume:cinder:publicURL",
            "cinder_api_insecure": false,
            "cinder_ca_certificates_file": null
          }
        },
        "registry": {
          "auth": {
            "memcached_servers": null,
            "memcache_security_strategy": null,
            "memcache_secret_key": null,
            "hash_algorithms": "md5",
            "cafile": null,
            "insecure": false
          }
        },
        "upload_image": {
        },
        "upload_images": [

        ]
      },
      "network": {
        "core_plugin": "neutron.plugins.ml2.plugin.Ml2Plugin",
        "use_namespaces": "True",
        "allow_overlapping_ips": "True",
        "enable_vpn": false,
        "rpc_thread_pool_size": 128,
        "rpc_conn_pool_size": 60,
        "rpc_response_timeout": 300,
        "service_plugins": [
          "neutron.services.l3_router.l3_router_plugin.L3RouterPlugin",
          "neutron.services.loadbalancer.plugin.LoadBalancerPlugin",
          "neutron.services.vpn.plugin.VPNDriverPlugin"
        ],
        "service_provider": [

"LOADBALANCER:Haproxy:neutron.services.loadbalancer.drivers.haproxy.plugin_driver.
HaproxyOnHostPluginDriver:default",

"VPN:openswan:neutron.services.vpn.service_drivers.ipsec.IPsecVPNDriver:default"
        ],
        "lbaas": {
          "device_driver":
"neutron.services.loadbalancer.drivers.haproxy.namespace_driver.HaproxyNSDriver"
        },
        "vpn": {
          "vpn_device_driver": [
            "neutron.services.vpn.device_drivers.ipsec.OpenSwanDriver"
          ]
        },
        "dhcp": {
```

```
      "ovs_use_veth": "False"
    },
    "l3": {
      "external_network_bridge": "br-ex",
      "external_network_bridge_interface": "eth0"
    },
    "quota": {
      "driver": "neutron.db.quota_db.DbQuotaDriver",
      "items": "network,subnet,port",
      "default": -1,
      "network": 10,
      "subnet": 10,
      "port": 50,
      "security_group": 10,
      "security_group_rule": 100,
      "router": 10,
      "floatingip": 50
    },
    "ml2": {
      "type_drivers": "local,flat,vlan,gre,vxlan",
      "tenant_network_types": "gre",
      "mechanism_drivers": "openvswitch",
      "flat_networks": "",
      "network_vlan_ranges": "",
      "tunnel_id_ranges": "1:1000",
      "vni_ranges": "1001:2000"
    },
    "openvswitch": {
      "tenant_network_type": "gre",
      "network_vlan_ranges": "",
      "enable_tunneling": "True",
      "tunnel_type": "gre",
      "tunnel_id_ranges": "1:1000",
      "veth_mtu": 1500,
      "tunnel_types": "gre,vxlan"
    },
    "api": {
      "auth": {
        "memcached_servers": null,
        "memcache_security_strategy": null,
        "memcache_secret_key": null,
        "hash_algorithms": "md5",
        "cafile": null,
        "insecure": false
      }
    },
    "nova": {
      "nova_ca_certificates_file": null,
      "nova_api_insecure": false
    }
  },
  "compute": {
    "enabled_apis": "osapi_compute",
    "rpc_backend": "nova.openstack.common.rpc.impl_qpid",
    "rpc_thread_pool_size": 2048,
```

```
        "rpc_conn_pool_size": 60,
        "rpc_response_timeout": 60,
        "vif_plugging_is_fatal": true,
        "vif_plugging_timeout": 300,
        "ssl_only": false,
        "cert": "",
        "key": "",
        "network": {
          "use_ipv6": true,
          "service_type": "neutron",
          "plugins": [
            "openvswitch"
          ],
          "neutron": {
            "ca_certificates_file": null,
            "service_neutron_metadata_proxy": false,
            "libvirt_vif_driver": "nova.virt.libvirt.vif.LibvirtGenericVIFDriver",
            "linuxnet_interface_driver":
"nova.network.linux_net.LinuxOVSInterfaceDriver",
            "dns_server": [

            ],
            "api_insecure": false
          }
        },
        "scheduler": {
          "default_filters": [
            "RetryFilter",
            "AvailabilityZoneFilter",
            "RamFilter",
            "ComputeFilter",
            "ComputeCapabilitiesFilter",
            "ImagePropertiesFilter"
          ]
        },
        "config": {
          "notification_drivers": [
            "nova.openstack.common.notifier.rpc_notifier"
          ],
          "notification_topics": [
            "notifications"
          ],
          "instance_usage_audit": "True",
          "instance_usage_audit_period": "hour",
          "notify_on_state_change": "vm_and_task_state",
          "allow_resize_to_same_host": true,
          "force_config_drive": null,
          "flat_injected": true,
          "quota_driver": "nova.quota.DbQuotaDriver",
          "quota_cores": 20,
          "quota_instances": 10,
          "quota_ram": 51200,
          "quota_floating_ips": 10,
          "quota_fixed_ips": -1,
          "quota_security_groups": 10,
```

```
      "quota_security_group_rules": 20,
      "quota_metadata_items": 128,
      "quota_injected_files": 5,
      "quota_injected_file_path_length": 255,
      "quota_injected_file_content_bytes": 10240,
      "quota_key_pairs": 100,
      "osapi_max_limit": 3000
    },
    "api": {
      "auth": {
        "memcached_servers": null,
        "memcache_security_strategy": null,
        "memcache_secret_key": null,
        "hash_algorithms": "md5",
        "cafile": null,
        "insecure": false
      }
    },
    "image": {
      "glance_api_insecure": false,
      "ssl": {
        "ca_file": null,
        "cert_file": null,
        "key_file": null
      }
    },
    "block-storage": {
      "cinder_ca_certificates_file": null,
      "cinder_api_insecure": false
    }
  },
  "dashboard": {
    "secret_key_path": "/var/lib/openstack-dashboard/.secret_key_store",
    "wsgi_socket_prefix": "/var/run/wsgi",
    "ssl_no_verify": "True",
    "ssl_cacert": null,
    "hash_algorithm": "md5",
    "traceenable": "Off"
  },
  "block-storage": {
    "quota_driver": "cinder.quota.DbQuotaDriver",
    "quota_volumes": 10,
    "quota_gigabytes": 1000,
    "quota_snapshots": 10,
    "no_snapshot_gb_quota": false,
    "use_default_quota_class": true,
    "api": {
      "auth": {
        "memcached_servers": null,
        "memcache_security_strategy": null,
        "memcache_secret_key": null,
        "hash_algorithms": "md5",
        "cafile": null,
        "insecure": false
      }
```

```
      },
      "image": {
        "glance_api_insecure": false,
        "glance_ca_certificates_file": null
      }
    },
    "orchestration": {
      "debug": "False",
      "verbose": "True",
      "api": {
        "auth": {
          "memcached_servers": null,
          "memcache_security_strategy": null,
          "memcache_secret_key": null,
          "hash_algorithms": "md5",
          "cafile": null,
          "insecure": false
        }
      },
      "clients": {
        "ca_file": null,
        "cert_file": null,
        "key_file": null,
        "insecure": false
      },
      "heat_stack_user_role": "heat_stack_user",
      "stack_user_domain_name": "heat",
      "stack_domain_admin": "heat_stack_admin"
    },
    "telemetry": {
      "api": {
        "auth": {
          "memcached_servers": null,
          "memcache_security_strategy": null,
          "memcache_secret_key": null,
          "hash_algorithms": "md5",
          "cafile": null,
          "insecure": false
        }
      },
      "service-credentials": {
        "cafile": null,
        "insecure": false
      }
    }
  },
  "ibm-sce": {
    "os": {
      "user": "sce",
      "group": "sce"
    },
    "config": {
      "prop": {
        "silent": "1"
      }
```

```
  },
  "user": {
    "input": {
      "authentication": {
        "username": "admin",
        "name": "Administrator"
      }
    },
    "install": {
      "folder": "/opt/ibm"
    },
    "prop": {
      "folder": "/var/opt/ibm"
    },
    "shortcuts": "/root"
  },
  "choose": {
    "license": {
      "type": {
        "boolean": "0"
      }
    }
  },
  "license": {
    "path": ""
  },
  "service": {
    "enabled": true
  },
  "package": {
    "fixpack": {
      "file": null
    },
    "update": {
      "jre": {
        "file": null
      }
    }
  },
  "in-place-upgrade": {
    "db2": {
      "target_db_name": "cfs42",
      "target_user_name": "cfs42"
    },
    "secret": {
      "inplace_upgrade_passwords_data_bag": "inplace_upgrade_passwords"
    }
  }
},
"mysql": {
  "allow_remote_root": true,
  "root_network_acl": [
    "127.0.0.1"
  ]
},
```

```
    "apache": {
      "proxy": {
        "deny_from": "none",
        "allow_from": "all"
      }
    },
    "db2": {
      "ssl": {
        "enable": true
      }
    }
  }
}
```

In the environment file, you at least need to customize these attributes:

- Update the environment *name* and make sure that the environment has the same name coded in the topology file. For example, `kvmenv`.
- Update the `openstack.compute.instance_name_template` to the same name as that of in the z/VM environment. For example, the example environment uses `itso-%04x` for the Linux instances name in z/VM.
- Set `openstack.network.openvswitch.bridge_mappings` to `null`.
- Set `openstack.network.openvswitch.bridge_mapping_interface` to `null`.
- Update the `openstack.region` name to `zCloud`.
- Change the `openstack.db.service_type` to `mysql`.
- Change the `openstack.mq.service_type` to `qpid`.
- Change the `openstack.mq.user` to `qpidclient`.
- Change the IP address for these endpoints to one of OpenStack controller nodes, for example `192.168.60.101` in the example environment.
  - `openstack.endpoints.host`
  - `openstack.endpoints.bind-host`
  - `openstack.endpoints.mq.hosts`
  - `openstack.endpoints.db.hosts`
- To use GRE network, update `openstack.network.ml2.tenant_network_types` to `gre`.
- Set `openstack.network.ml2.flat_networks` to `null`.
- Set `openstack.network.ml2.network_vlan_ranges` to `null`.
- Update `network.openvswitch.tenant_network_types` to `gre`.
- Set `network.openvswitch.network_vlan_ranges` to `null`.
- Set `compute.rpc_backend` to `nova.openstack.common.rpc.impl_pqid`.

A.2.3.2 Update the Chef environment

Having customized the environment file, update the changes to the Chef environment by using the command shown in Example A-9.

Example A-9 Update Chef environment

```
[mnadmin@xcat kvm_env] $ sudo knife environment from file kvm.json
Updated Environment kvmenv
```

A.2.4 Deploy compute service on a x86 node

After all these steps, you can deploy a new compute service to the x86 node. Issue the command shown in Example A-10 to deploy the OpenStack service.

Example A-10 Deploy compute service

```
[mnadmin@xcat kvm_env] $ sudo knife os manage deploy topology topology.json
Deploying topology 'kvm-topology' ...
Deploying to nodes with run_order_number '0'
Bootstraping nodes with environment kvmenv...
Bootstrapping node ...
Connecting to 192.168.60.15
192.168.60.15 Starting Chef Client on Node
192.168.60.15 Bootstrapping Node
192.168.60.15 Synchronizing Cookbooks
192.168.60.15 Compiling Cookbooks
All nodes for environment kvmenv bootstrapped.
Deploying bootstrapped nodes with run_order_number '0'...
Setting run list for node 192.168.60.15...
192.168.60.15:
  run_list: role[ibm-os-compute-node-kvm]
192.168.60.15 Converging Node
192.168.60.15 Synchronizing Cookbooks
192.168.60.15 Compiling Cookbooks
192.168.60.15 Running Recipe chef_handler::default
192.168.60.15 Running Recipe ibm-openstack-common::cmwo-version
192.168.60.15 Running Recipe yum::default
192.168.60.15 Running Recipe ibm-openstack-yum-server::yumrepo
192.168.60.15 Running Recipe ibm-openstack-yum-server::yumpriorities
192.168.60.15 Running Recipe ntp::default
192.168.60.15 Running Recipe selinux::_common
192.168.60.15 Running Recipe selinux::default
192.168.60.15 Running Recipe ibm-openstack-common::openstack-selinux
192.168.60.15 Running Recipe iptables::default
192.168.60.15 Running Recipe ibm-openstack-iptables::default
192.168.60.15 Running Recipe chef_handler::default
192.168.60.15 Running Recipe ibm-openstack-common::cmwo-version
192.168.60.15 Running Recipe openstack-common::default
192.168.60.15 Running Recipe openstack-common::logging
192.168.60.15 Running Recipe openstack-common::sysctl
192.168.60.15 Running Recipe openstack-compute::nova-common
192.168.60.15 Running Recipe openstack-identity::client
192.168.60.15 Running Recipe openstack-network::default
192.168.60.15 Running Recipe openstack-network::openvswitch
192.168.60.15 Running Recipe openstack-compute::compute
```

```
192.168.60.15 Running Recipe openstack-compute::libvirt
192.168.60.15 Running Recipe openstack-telemetry::identity_registration
192.168.60.15 Running Recipe openstack-telemetry::common
192.168.60.15 Running Recipe openstack-telemetry::agent-compute
192.168.60.15 Running Recipe ntp::default
192.168.60.15 Running Recipe iptables::default
192.168.60.15 Running Recipe openstack-compute::compute
192.168.60.15 Running Recipe openstack-network::default
192.168.60.15 Running Recipe openstack-network::openvswitch
192.168.60.15 Running Recipe openstack-telemetry::agent-compute
192.168.60.15 Completed
All bootstrapped nodes with run_order_number '0' deployed.

Results for deploy of topology 'kvm-topology'
Results for nodes with run_order_number '0'
Deploy of node at 192.168.60.15 was successful.
Deploy of topology 'topology.json' completed in 417 seconds.
```

A.3 Verify the heterogeneous cloud infrastructure

After you deploy the compute services on the x86 node, you can perform the following verification steps to make sure all the components are working properly.

A.3.1 Verify the infrastructure using commands

In the OpenStack controller node (xCAT MN), you can issue the following command to verify whether all the related nova services are up and running as shown in Example A-11.

Example A-11 Check the nova services

```
[mnadmin@xcat kvm_env] $ nova service-list
+----+------------------+--------------------------+----------+---------+-------+-
---------------------------+-----------------+
| Id | Binary           | Host                     | Zone     | Status  | State |
Updated_at                 | Disabled Reason |
+----+------------------+--------------------------+----------+---------+-------+-
---------------------------+-----------------+
| 1  | nova-cert        | itsossi1                 | internal | enabled | up    |
2015-08-11T14:55:08.000000 | -               |
| 2  | nova-console     | itsossi1                 | internal | enabled | up    |
2015-08-11T14:55:10.000000 | -               |
| 3  | nova-conductor   | itsossi1                 | internal | enabled | up    |
2015-08-11T14:55:08.000000 | -               |
| 5  | nova-consoleauth | itsossi1                 | internal | enabled | up    |
2015-08-11T14:55:10.000000 | -               |
| 6  | nova-scheduler   | itsossi1                 | internal | enabled | up    |
2015-08-11T14:55:11.000000 | -               |
| 7  | nova-compute     | itsossi1                 | nova     | enabled | up    |
2015-08-11T14:55:05.000000 | -               |
| 8  | nova-compute     | itsossi2                 | nova     | enabled | up    |
2015-08-11T14:55:09.000000 | -               |
| 9  | nova-compute     | testx8601.itso.ihost.com | nova     | enabled | up    |
2015-08-11T14:55:10.000000 | -               |
```

```
+----+------------------+--------------------------+----------+---------+-------+-
--------------------------+-----------------+
```

If the x86 nova-compute service is not up, you need to check whether the nova-compute service is running in the x86 node. You might also need to check the nova `compute.log` in the x86 node for problem determination.

A.3.2 Verify the infrastructure from self-service portal

If the cloud infrastructure works correctly, you can see a report from the self-service portal as shown in Figure A-1.

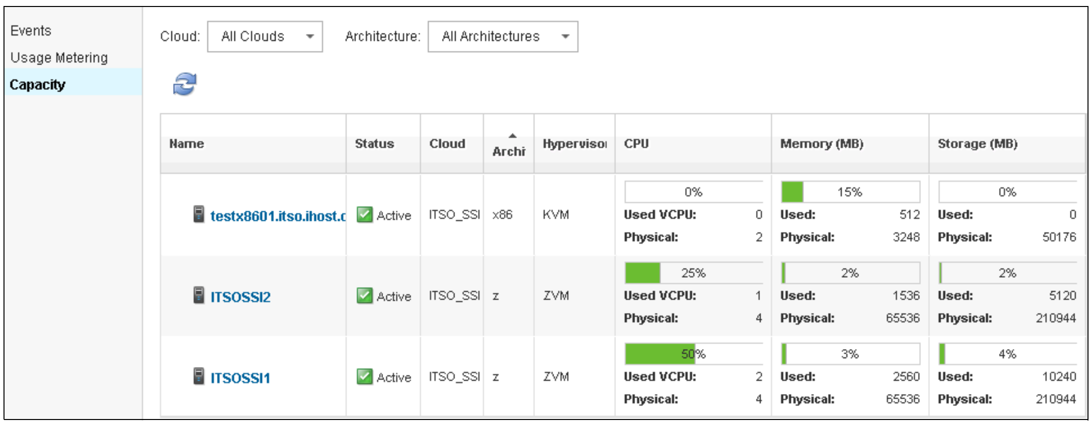


Figure A-1 Verifying the cloud infrastructure

After that, you can import x86 Linux images and deploy Linux instances to the x86 server.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- *IBM Software Defined Environment*, SG24-8238
- *The Virtualization Cookbook for IBM z Systems Volume 1: IBM z/VM 6.3,* SG24-8147
- *z/OS Infoprint Server Implementation*, SG24-6234

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Other publications

These publications are also relevant as further information sources:

- *Enabling z/VM for OpenStack (Support for OpenStack Juno Release)*, SC24-6249
- *z/VM V6.3 CMS Commands and Utilities Reference*, SC24-6166
- *z/VM V6.3 CP Commands and Utilities Reference*, SC24-6175
- *z/VM V6.3 CP Planning and Administration*, SC24-6178
- *z/VM V6.3 Directory Maintenance Facility Commands Reference*, SC24-6188
- *z/VM V6.3 Directory Maintenance Facility Tailoring and Administration Guide*, SC24-6190
- *z/VM V6.3 Getting Started with Linux on System z*, SC24-6194

Online resources

These websites are also relevant as further information sources:

- IBM Shopz

  http://www.ibm.com/software/ShopzSeries
- IBM z/VM 6.3 RSU service

  http://www.vm.ibm.com/service/rsu/stk630.html
- APAR VM65676 - Support for z/VM Cloud Manager Appliance 4.2.0.2

  http://www.ibm.com/support/docview.wss?uid=isg1VM65676

- IBM Cloud Manager 4.2 (Juno) with OpenStack for z Systems

  http://www.vm.ibm.com/sysman/osmntlvl.html

- z/VM xCAT maintenance

  http://www.vm.ibm.com/sysman/xcmntlvl.html

- IBM Fix Central

  http://www.ibm.com/support/fixcentral

- IBM developerWorks

  https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/W21ed5ba0f4a9_46f4_9626_24cbbb86fbb9/page/Documentation

- IBM Knowledge Center

  https://www.ibm.com/support/knowledgecenter/SST55W_4.2.0/liaca/liaca_kc_welcome.html

- Setuptools

  https://pypi.python.org/pypi/setuptools

- Cloud-init tar file

  https://launchpad.net/cloud-init/+download

- OpenStack documentation:

  http://docs.openstack.org/

- IBM Cloud Manager with OpenStack for z Systems, V4.2 announcement letter:

  http://www.ibm.com/common/ssi/ShowDoc.wss?docURL=/common/ssi/rep_ca/6/897/ENUS215-106/index.html&lang=en&request_locale=en

- IBM Custom Patterns for Linux on z Systems announcement letter.

  http://www.ibm.com/common/ssi/cgi-bin/ssialias?infotype=AN&subtype=CA&htmlfid=897/ENUS215-052&appname=USN

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Redbooks

IBM Cloud Manager with OpenStack on z Systems V4.2

(0.2”spine)
0.17”<->0.473”
90<->249 pages

SG24-8295-00

ISBN 0738440973

Printed in U.S.A.