Red Hat OpenStack Red Hat OpenStack 3.0 (Grizzly) Deployment Guide (Foreman Technology Preview)

Deploying Red Hat Enterprise Linux OpenStack Platform using Foreman

Edition 1

Steve Gordon Steve Gordon Summer Long

Red Hat OpenStack Red Hat OpenStack 3.0 (Grizzly) Deployment Guide (Foreman Technology Preview)

Deploying Red Hat Enterprise Linux OpenStack Platform using Foreman
Edition 1

Steve Gordon
sgordon@redhat.com

Summer Long
slong@redhat.com

Legal Notice

Keywords

Abstract

Deploying Red Hat Enterprise Linux OpenStack Platform using Foreman Please note that Foreman is provided as a Technology Preview. For more information on Technology Preview status and the support scope it entails refer to https://access.redhat.com/support/offerings/techpreview/.

Table of Contents

Preface

1. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the Liberation Fonts set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later include the Liberation Fonts set by default.

1.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

`Mono-spaced Bold`

Used to highlight system input, including shell commands, file names and paths. Also used to highlight keys and key combinations. For example:

> To see the contents of the file **`my_next_bestselling_novel`** in your current working directory, enter the **`cat my_next_bestselling_novel`** command at the shell prompt and press **`Enter`** to execute the command.

The above includes a file name, a shell command and a key, all presented in mono-spaced bold and all distinguishable thanks to context.

Key combinations can be distinguished from an individual key by the plus sign that connects each part of a key combination. For example:

> Press **`Enter`** to execute the command.
>
> Press **`Ctrl`**+**`Alt`**+**`F2`** to switch to a virtual terminal.

The first example highlights a particular key to press. The second example highlights a key combination: a set of three keys pressed simultaneously.

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **`mono-spaced bold`**. For example:

> File-related classes include **`filesystem`** for file systems, **`file`** for files, and **`dir`** for directories. Each class has its own associated set of permissions.

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialog box text; labeled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

> Choose **System** → **Preferences** → **Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, select the **Left-handed mouse** check box and click **Close** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).
>
> To insert a special character into a **gedit** file, choose **Applications** → **Accessories** →

> **Character Map** from the main menu bar. Next, choose **Search** → **Find…** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit** → **Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in proportional bold and all distinguishable by context.

Mono-spaced Bold Italic or ***Proportional Bold Italic***

Whether mono-spaced bold or proportional bold, the addition of italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

> To connect to a remote machine using ssh, type **ssh *username*@*domain.name*** at a shell prompt. If the remote machine is **example.com** and your username on that machine is john, type **ssh john@example.com**.
>
> The **mount -o remount *file-system*** command remounts the named file system. For example, to remount the **/home** file system, the command is **mount -o remount /home**.
>
> To see the version of a currently installed package, use the **rpm -q *package*** command. It will return a result as follows: ***package-version-release***.

Note the words in bold italics above — username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

> Publican is a *DocBook* publishing system.

1.2. Pull-quote Conventions

Terminal output and source code listings are set off visually from the surrounding text.

Output sent to a terminal is set in **mono-spaced roman** and presented thus:

```
books        Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

Source-code listings are also set in **mono-spaced roman** but add syntax highlighting as follows:

```
static int kvm_vm_ioctl_deassign_device(struct kvm *kvm,
                 struct kvm_assigned_pci_dev *assigned_dev)
{
         int r = 0;
         struct kvm_assigned_dev_kernel *match;

         mutex_lock(&kvm->lock);

         match = kvm_find_assigned_dev(&kvm->arch.assigned_dev_head,
                                       assigned_dev->assigned_dev_id);
         if (!match) {
                 printk(KERN_INFO "%s: device hasn't been assigned before, "
                   "so cannot be deassigned\n", __func__);
                 r = -EINVAL;
                 goto out;
         }

         kvm_deassign_device(kvm, match);

         kvm_free_assigned_device(kvm, match);

out:
         mutex_unlock(&kvm->lock);
         return r;
}
```

1.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.

Note

Notes are tips, shortcuts or alternative approaches to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

Important

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring a box labeled 'Important' will not cause data loss but may cause irritation and frustration.

Warning

Warnings should not be ignored. Ignoring warnings will most likely cause data loss.

2. Getting Help and Giving Feedback

2.1. Do You Need Help?

If you experience difficulty with a procedure described in this documentation, visit the Red Hat Customer

Portal at http://access.redhat.com. Through the customer portal, you can:

- search or browse through a knowledgebase of technical support articles about Red Hat products.
- submit a support case to Red Hat Global Support Services (GSS).
- access other product documentation.

Red Hat also hosts a large number of electronic mailing lists for discussion of Red Hat software and technology. You can find a list of publicly available mailing lists at https://www.redhat.com/mailman/listinfo. Click on the name of any mailing list to subscribe to that list or to access the list archives.

2.2. We Need Feedback

If you find a typographical error in this manual, or if you have thought of a way to make this manual better, we would love to hear from you. Please submit a report in Bugzilla: http://bugzilla.redhat.com/ against the product **Red Hat OpenStack.**

When submitting a bug report, be sure to mention the manual's identifier: *Deployment_Guide_Foreman_Technical_Preview*

If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

Chapter 1. Introduction

1.1. Product Introduction

1.1.1. Overview

Red Hat Enterprise Linux OpenStack Platform provides the foundation to build a private or public Infrastructure-as-a-Service (IaaS) cloud on top of Red Hat Enterprise Linux. It offers a massively scalable, fault-tolerant platform for the development of cloud-enabled workloads.

The current Red Hat system is based on OpenStack Grizzly, and packaged so that available physical hardware can be turned into a private, public, or hybrid cloud platform including:

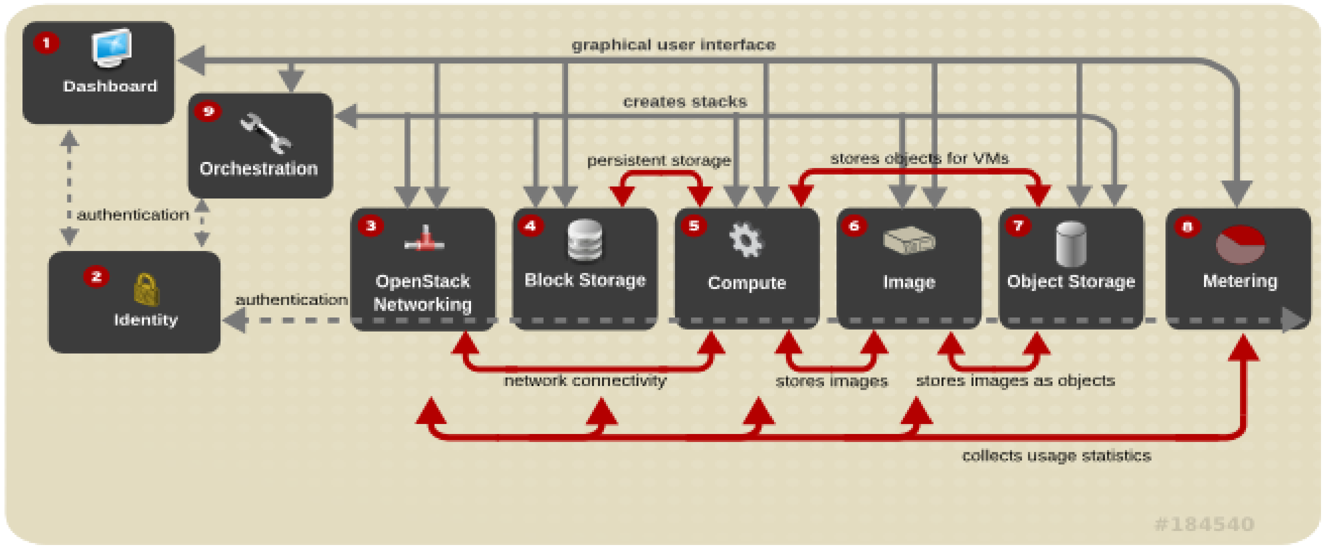
- Fully distributed object storage
- Persistent block-level storage
- Virtual-machine provisioning engine and image storage
- Authentication and authorization mechanism
- Integrated networking
- Web browser-based GUI for both users and administration.

The Red Hat Enterprise Linux OpenStack Platform IaaS cloud is implemented by a collection of interacting services that control its computing, storage, and networking resources. The cloud is managed using a web-based interface which allows administrators to control, provision, and automate OpenStack resources. Additionally, the OpenStack infrastructure is facilitated through an extensive API, which is also available to end users of the cloud.

Report a bug

1.1.2. Architecture

The following diagram provides a high-level overview of the OpenStack architecture.



Each OpenStack service has a code name, which is reflected in the names of configuration files and command-line utility programs. For example, the Identity service has a configuration file called keystone.conf.

Table 1.1. Services

	Section	Code	Description
1	Dashboard	horizon	A web-based dashboard for managing OpenStack services.
2	Identity	keystone	A centralized identity service that provides authentication and authorization for other services, and manages users, tenants, and roles.
3	OpenStack Networking	quantum	A networking service that provides connectivity between the interfaces of other OpenStack services.
4	Block Storage	cinder	A service that manages persistent block storage volumes for virtual machines.
5	Compute	nova	A service that launches and schedules networks of machines running on nodes.
6	Image	glance	A registry service for virtual machine images.
7	Object Storage	swift	A service providing object storage which allows users to store and retrieve files (arbitrary data).
8	Metering (Technical Preview)	ceilometer	A service providing measurements of cloud resources.
9	Orchestration (Technical Preview)	heat	A service providing a template-based orchestration engine, which supports the automatic creation of resource stacks.

The Service Details section provides more detailed information about the Openstack service components. Each OpenStack service is comprised of a collection of Linux services, MySQL databases, or other components, which together provide a functional group. For example, the **`glance-api`** and **`glance-registry`** Linux services, together with a MySQL database, implement the Image service.

Important

For more information on the support scope for features marked as technology previews, refer to https://access.redhat.com/support/offerings/techpreview/

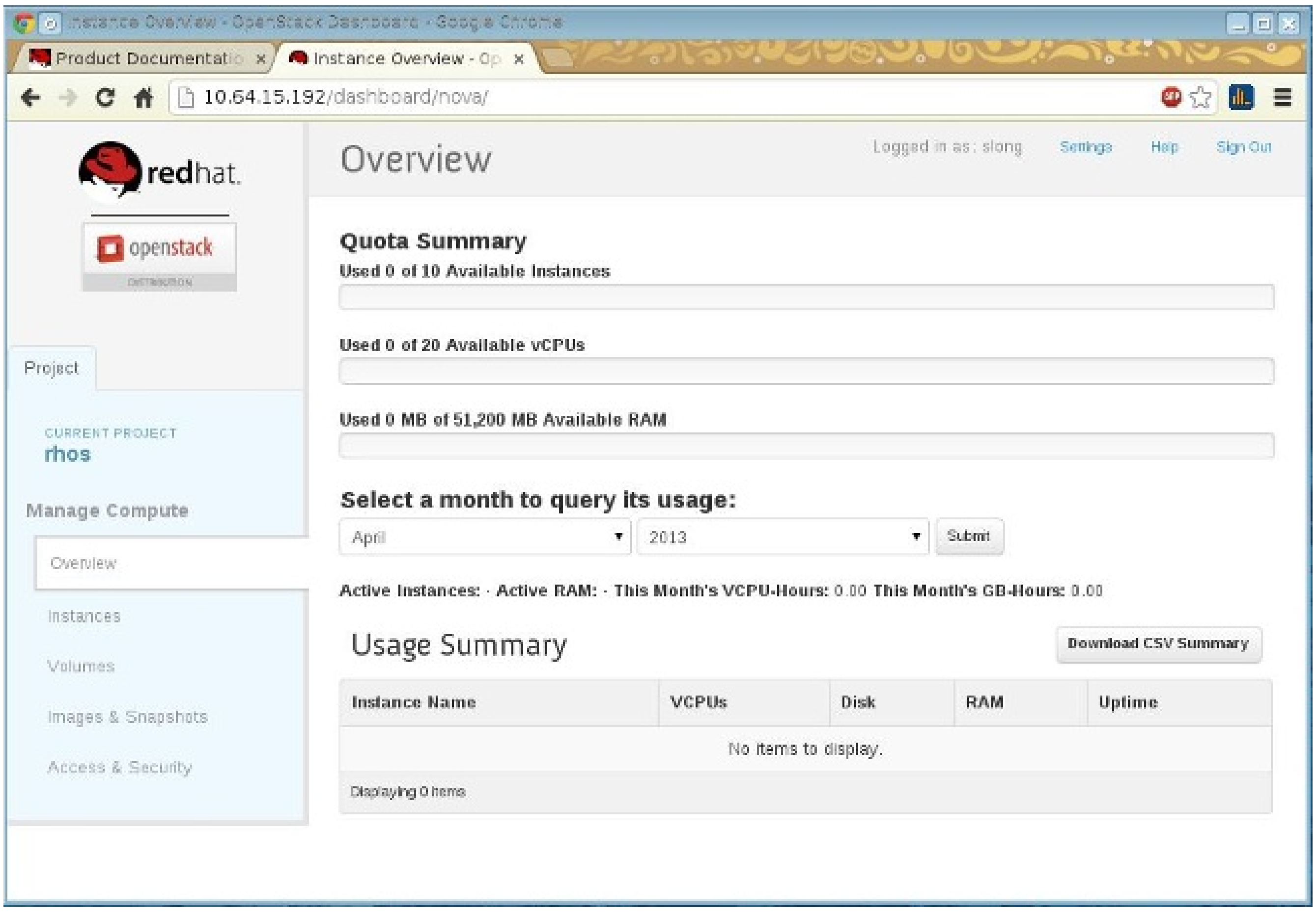
Report a bug

1.1.3. Service Details

1.1.3.1. Dashboard Service

The Dashboard service provides a graphical user interface for end users and administrators, allowing operations such as creating and launching instances, managing networking, and setting access controls. Its modular design allows interfacing with other products such as billing, monitoring, and additional management tools. The service provides three basic dashboards: user, system, and settings.

The following screenshot displays a user's dashboard after OpenStack is first installed:



Overview

Quota Summary

Used 0 of 10 Available Instances

Used 0 of 20 Available vCPUs

Used 0 MB of 51,200 MB Available RAM

Select a month to query its usage:

April 2013

Active Instances: - Active RAM: - This Month's VCPU-Hours: 0.00 This Month's GB-Hours: 0.00

Usage Summary

Instance Name	VCPUs	Disk	RAM	Uptime
No items to display.				

Displaying 0 items

The identity of the logged-in user determines the dashboards and panels that are visible in the dashboard.

The Dashboard service is composed of:

- **`openstack-dashboard`**, a Django (Python) web application, so that the dashboard can be easily accessed using any web browser.
- An Apache HTTP server (**`httpd`** service), to host the application.
- A database, for managing sessions.

Report a bug

1.1.3.2. Identity Service

The Identity service authenticates and authorizes OpenStack users (that is, keeps track of users and their permitted activities); the service is used by all OpenStack components. The service supports multiple forms of authentication including user name and password credentials, token-based systems, and AWS-style logins (Amazon Web Services).

The Identity service also provides a central catalog of services and endpoints running in a particular OpenStack cloud, which acts as a service directory for other OpenStack systems. Each endpoint is assigned:

- **`adminURL`**, the URL for the administrative endpoint for the service. Only the Identity service might use a value here that is different from publicURL; all other services will use the same value.
- **`internalURL`**, the URL of an internal-facing endpoint for the service (typically same as the publicURL).
- **`publicURL`**, the URL of the public-facing endpoint for the service.

- **`region`**, in which the service is located. By default, if a region is not specified, the 'RegionOne' location is used.

The Identity service uses the following concepts:

- Users, which have associated information (such as a name and password). In addition to custom users, a user must be defined for each cataloged service (for example, the 'glance' user for the Image service).
- Tenants, which are generally the user's group, project, or organization.
- Roles, which determine a user's permissions.

The Identity service is composed of:

- **`keystone`** service, which provides the administrative and public APIs.
- Databases for each of the internal services.

Report a bug

1.1.3.3. OpenStack Networking Service

The OpenStack Networking service provides a scalable and API-driven system for managing the network connectivity, addressing, and services within an OpenStack IaaS cloud deployment. Because the OpenStack network is software-defined, it can easily and quickly react to changing network needs (for example, creating and assigning new IP addresses).

Advantages include:

- Users can create networks, control traffic, and connect servers and devices to one or more networks.
- OpenStack offers flexible networking models, so that administrators can change the networking model to adapt to their volume and tenancy.
- IPs can be dedicated or floating; floating IPs allow dynamic traffic rerouting.

OpenStack Networking is composed of:

- **`quantum-server`** Python daemon, which manages user requests (and exposes the API).

  The **`quantum-server`** daemon is configured with a plugin that implements the OpenStack Networking API operations using a specific set of networking mechanisms. A wide choice of plugins are also available. For example, the **`openvswitch`** and **`linuxbridge`** plugins utilize native Linux networking mechanisms, while other plugins interface with external devices or SDN controllers.
- **`quantum-l3-agent`**, an agent providing L3/NAT forwarding.
- **`quantum-*-agent`**, a plug-in agent that runs on each node to perform local networking configuration for the node's VMs and networking services.
- **`quantum-dhcp-agent`**, an agent providing DHCP services to tenant networks.
- Database, for persistent storage.

Report a bug

1.1.3.4. Block Storage Service

The Block Storage (or volume) service provides persistent block storage management for virtual hard drives. The block storage system manages the creation of block devices to servers. Block storage volumes are fully integrated into both the Compute and Dashboard services, which allows cloud users to manage their own storage needs (Compute handles the attaching and detaching of devices). Both regions and zones (for details, refer to the Object Storage section) can be used to handle distributed block storage hosts.

Block storage is appropriate for performance-sensitive scenarios such as database storage, expandable file systems, or providing a server with access to raw block-level storage. Additionally, snapshots can be taken to either restore data or to create new block storage volumes (snapshots are dependent upon driver support).

Basic operations include:

- Create, list, and delete volumes.
- Create, list, and delete snapshots.
- Attach and detach volumes to running virtual machines.

The Block Storage service is composed of the following:

- **`openstack-cinder-volume`**, which carves out storage for virtual machines on demand. A number of drivers are provided for interaction with storage providers.
- **`openstack-cinder-api`**, which responds to and handles requests, and places them in the message queue.
- **`openstack-cinder-scheduler`**, which assigns tasks to the queue and determines the provisioning volume server.
- Database, for state information.

See Also:

- Section 1.1.3.5, “Compute Service”

Report a bug

1.1.3.5. Compute Service

The Compute service is the heart of the OpenStack cloud by providing virtual machines on demand. Compute schedules virtual machines to run on a set of nodes by defining drivers that interact with underlying virtualization mechanisms, and exposing the functionality to the other OpenStack components.

Compute interacts with the Identity service for authentication, Image service for images, and the Dashboard service for the user and administrative interface. Access to images is limited by project and by user; quotas are limited per project (for example, the number of instances). The Compute service is designed to scale horizontally on standard hardware, and can download images to launch instances as required.

Table 1.2. Ways to Segregate the Cloud

Concept	Description
Regions	Each service cataloged in the Identity service is identified by its region, which typically represents a geographical location, and its endpoint. In a cloud with multiple Compute deployments, regions allow for the discrete separation of services, and are a robust way to share some infrastructure between Compute installations, while allowing for a high degree of failure tolerance.
Cells (Technology Preview)	A cloud's Compute hosts can be partitioned into groups called cells (to handle large deployments or geographically separate installations). Cells are configured in a tree. The top-level cell ('API cell') runs the **`nova-api`** service, but no **`nova-compute`** services. In contrast, each child cell runs all of the other typical **`nova-*`** services found in a regular installation, except for the **`nova-api`** service. Each cell has its own message queue and database service, and also runs **`nova-cells`**, which manages the communication between the API cell and its child cells. This means that: » A single API server can be used to control access to multiple Compute installations. » A second level of scheduling at the cell level is available (versus host scheduling), which provides greater flexibility over the control of where virtual machines are run.
Host Aggregates and Availability Zones	A single Compute deployment can be partitioned into logical groups (for example, into multiple groups of hosts that share common resources like storage and network, or which have a special property such as trusted computing hardware). If the user is: » An administrator, the group is presented as a Host Aggregate, which has assigned Compute hosts and associated metadata. An aggregate's metadata is commonly used to provide information for use with **`nova-scheduler`** (for example, limiting specific flavors or images to a subset of hosts). » A user, the group is presented as an Availability Zone. The user cannot view the group's metadata, nor which hosts make up the zone. Aggregates, or zones, can be used to: » Handle load balancing and instance distribution. » Provide some form of physical isolation and redundancy from other zones (such as by using a separate power supply or network equipment). » Identify a set of servers that have some common attribute. » Separate out different classes of hardware.

Important

For more information on the support scope for features marked as technology previews, refer to https://access.redhat.com/support/offerings/techpreview/

Compute is composed of the following:

- **`openstack-nova-api`** service, which handles requests and provides access to the Compute services (such as booting an instance).
- **`openstack-nova-cert`** service, which provides the certificate manager.
- **`openstack-nova-compute`** service, which creates and terminates the virtual instances. The service interacts with Hypervisor to bring up new instances, and ensures that the state is maintained in the Compute database.
- **`openstack-nova-conductor`** service, which provides database-access support for Compute nodes (thereby reducing security risks).
- **`openstack-nova-consoleauth`** service, which handles console authorization.
- **`openstack-nova-network`** service, which handles Compute network traffic (both private and public access). This service handles such tasks as assigning an IP address to a new virtual instance, and implementing security group rules.
- **`openstack-nova-novncproxy`** service, which provides a VNC proxy for browsers (enabling VNC consoles to access virtual machines started by OpenStack).
- **`openstack-nova-scheduler`** service, which dispatches requests for new virtual machines to the correct node.
- Apache Qpid server (**`qpidd`** service), which provides the AMPQ message queue. This server (also used by Block Storage) handles the OpenStack transaction management, including queuing, distribution, security, management, clustering, and federation. Messaging becomes especially important when a OpenStack deployment is scaled and its services are running on multiple machines.
- **`libvirtd`** service, which is the driver for the hypervisor and enables the creation of virtual machines.
- KVM Linux hypervisor, which creates virtual machines and enables their live migration from node to node.
- Database, for build-time and run-time infrastructure state.

Report a bug

1.1.3.6. Image Service

The Image service acts as a registry for virtual disk images. Users can add new images or take a snapshot (copy) of an existing server for immediate storage. Snapshots can be used as back up or as templates for new servers. Registered images can be stored in the Object Storage service, as well as in other locations (for example, in simple file systems or external web servers).

The following image formats are supported:

- raw (unstructured format)
- aki/ami/ari (Amazon kernel, ramdisk, or machine image)
- iso (archive format for optical discs; for example, CDROM)
- qcow2 (Qemu/KVM, supports *Copy on Write*)
- vhd (Hyper-V, common for virtual machine monitors from VMWare, Xen, Microsoft, VirtualBox, and others)
- vdi (Qemu/VirtualBox)
- vmdk (VMWare)

Container formats can also be used by the Image service; the format determines the type of metadata stored in the image about the actual virtual machine. The following formats are supported.

- bare (no metadata is included)
- ovf (OVF format)
- aki/ami/ari (Amazon kernel, ramdisk, or machine image)

The Image service is composed of the following:

- **`openstack-glance-api`**, which handles requests, and image delivery (interacts with storage backends for retrieval and storage). This service uses the registry to retrieve image information (the registry service is never, and should never be, accessed directly).
- **`openstack-glance-registry`**, which manages all metadata associated with each image, and which requires a database.
- Database, for image metadata.

Report a bug

1.1.3.7. Object Storage Service

The Object Storage service provides object storage in virtual containers, which allows users to store and retrieve files. The service's distributed architecture supports horizontal scaling; redundancy as failure-proofing is provided through software-based data replication.

Because it supports asynchronous eventual consistency replication, it is well suited to multiple data-center deployment. Object Storage uses the concept of:

- Storage replicas, which are used to maintain the state of objects in the case of outage. A minimum of three replicas is recommended.
- Storage zones, which are used to host replicas. Zones ensure that each replica of a given object can be stored separately. A zone might represent an individual disk drive or array, a server, all the servers in a rack, or even an entire data center.
- Storage regions, which are essentially a group of zones sharing a location. Regions can be, for example, groups of servers or server farms, usually located in the same geographical area. Regions have a separate API endpoint per Object Storage service installation, which allows for a discrete separation of services.

The Object Storage service is composed of the following:

- **`openstack-swift-proxy`** service, which exposes the public API, and is responsible for handling requests and routing them accordingly. Objects are streamed through the proxy server to the user (not spooled). Objects can also be served out via HTTP.
- **`openstack-swift-object`** blob server, which stores, retrieves, and deletes objects.
- **`openstack-swift-account`** server, which is responsible for listings of containers, using the account database.
- **`openstack-swift-container`** server, which handles listings of objects (what objects are in a specific container) using the container database.
- Ring files, which contain details of all the storage devices, and which are used to deduce where a particular piece of data is stored (maps the names of stored entities to their physical location). One file is created for each object, account, and container server.
- Account database.
- Container database.
- ext4 (recommended) or XFS filesystem for object storage.
- Housekeeping processes, including replication and auditors.

Report a bug

1.1.3.8. Metering (Technical Preview)

The Metering service provides user-level usage data for OpenStack-based clouds, which can be used for customer billing, system monitoring, or alerts. Data can be collected by notifications sent by existing OpenStack components (for example, usage events emitted from Compute) or by polling the infrastructure (for example, libvirt).

Metering includes a storage daemon that communicates with authenticated agents via a trusted messaging system, to collect and aggregate data. Additionally, the service uses a plugin system, which makes it easy to add new monitors.

The Metering service is composed of the following:

- **`ceilometer-agent-compute`**, an agent that runs on each Compute node, and polls for resource utilization statistics.
- **`ceilometer-agent-central`**, an agent that runs on a central management server to poll for utilization statistics about resources not tied to instances or Compute nodes.
- **`ceilometer-collector`**, an agent that runs on one or more central management servers to monitor the message queues. Notification messages are processed and turned into metering messages, and sent back out on to the message bus using the appropriate topic. Metering messages are written to the data store without modification.
- Mongo database, for collected usage sample data.
- API Server, which runs on one or more central management servers to provide access to the data store's data. Only the Collector and the API server have access to the data store.

Report a bug

1.1.3.9. Orchestration (Technical Preview)

The Orchestration service provides a template-based orchestration engine for the OpenStack cloud, which can be used to create and manage cloud infrastructure resources such as storage, networking, instances, and applications as a repeatable running environment.

Templates are used to create stacks, which are collections of resources (for example instances, floating IPs, volumes, security groups, or users). The service offers access to all OpenStack core services via a single modular template, with additional orchestration capabilities such as auto-scaling and basic high availability.

Features include:

- A single template provides access to all underlying service APIs.
- Templates are modular (resource oriented).
- Templates can be recursively defined, and therefore reusable (nested stacks). This means that the cloud infrastructure can be defined and reused in a modular way.
- Resource implementation is pluggable, which allows for custom resources.
- Autoscaling functionality (automatically adding or removing resources depending upon usage).
- Basic high availability functionality.

The Orchestration service is composed of the following:

- **`heat`**, a CLI tool that communicates with the heat-api to execute AWS CloudFormation APIs.
- **`heat-api`**, which is an OpenStack-native REST API that processes API requests by sending them

to the heat-engine over RPC.

- **`heat-api-cfn`**, which provides an AWS-Query API that is compatible with AWS CloudFormation and processes API requests by sending them to the heat-engine over RPC.
- **`heat-engine`**, which orchestrates the launching of templates and provide events back to the API consumer.
- **`heat-api-cloudwatch`**, which provides monitoring (metrics collection) for the Orchestration service.
- **`heat-cfntools`**, which is a package of helper scripts (for example, cfn-hup, which handles updates to metadata and executes custom hooks).

> **Note**
>
> The **`heat-cfntools`** package is only installed on images that are launched by heat into Compute servers.

Report a bug

1.2. Foreman Introduction

Foreman is a deployment management tool. It provides a web user interface for managing the installation and configuration of remote systems. Deployment of changes is performed using Puppet. Additionally Foreman is able to provide *Dynamic Host Configuration Protocol* (DHCP), *Domain Name System* (DNS), *Preboot Execution Environment* (PXE), and *Trivial File Transfer Protocol* (TFTP) services. Controlling these services allows Foreman to provision even physical systems that do not yet have an operating system installed.

Foreman uses what is referred to as a *host group* definition to group hosts that share common configuration requirements together. Two host groups are provided with the version of Foreman included in Red Hat Enterprise Linux OpenStack Platform:

OpenStack Controller

This host group is intended for use on a single host that will act as a controller for the OpenStack deployment. Services that will be deployed to hosts added to this host group include:

- OpenStack Dashboard (Horizon).
- OpenStack Image Storage service (Glance).
- OpenStack Identity service (Keystone).
- MySQL database server.
- Qpid message broker.

The OpenStack API and scheduling services, including those of the Compute service (Nova), also run on the controller.

OpenStack Nova Compute

This host group is intended for use on one or more hosts that will act as compute nodes for the OpenStack deployment. These are the systems that virtual machine instances will run on, while accessing the authentication, storage, and messaging infrastructure provided by the controller node. An instance of the Compute service (Nova) runs on each compute node.

Important

At this time compute services deployed using Foreman are configured to use Nova Networking (**`openstack-nova-network`**) instead of OpenStack Networking (**`quantum-server`**) to provide network services. Additionally the OpenStack Block Storage service (Cinder) is not currently included in the provided host group definitions.

This guide documents the steps involved in:

- Installing Foreman.
- Configuring Foreman.
- Adding hosts to Foreman.

Completion of the steps detailed in this guide will result in the deployment of a Foreman server, an OpenStack controller node, and one or more OpenStack compute nodes.

Report a bug

1.3. Requirements

Deploying OpenStack using Foreman requires a minimum of three physical systems:

- A system to host Foreman itself.
- A system to act as a controller node in the OpenStack deployment.
- A system to act as a compute node in the OpenStack deployment.

Refer to the *Getting Started Guide* for more specific controller and compute node hardware requirements. Only additional requirements specific to deployment using Foreman will be covered here.

Report a bug

1.3.1. Hardware Requirements

The hardware requirements of each system involved in a Foreman deployment of OpenStack relate primarily to networking and are listed here.

Foreman Server

The system that will act as the Foreman server requires two network interfaces:

- A network interface to reach the external network.
- A network interface to own internal provisioning services including DHCP, DNS, and PXE or TFTP on the local subnet.

The Foreman server must be reachable from the controller node and the compute nodes that will act as Foreman clients. The Foreman server does not however require access to the OpenStack public and private networks.

Controller Node

The system that will act as the controller node requires three network interfaces:

- A network interface to communicate with the Foreman server and the software repositories required to retrieve OpenStack packages. This interface must be connected on the same subnet as the interface of the Foreman server that will provide host provisioning services including DHCP.

- A network interface to dedicate to traffic on the public OpenStack network.
- A network interface to dedicate to traffic on the private OpenStack network.

Compute Nodes

The systems that will act as compute nodes also require three network interfaces. These network interfaces will be used for the same purposes as those attached to the controller nodes, listed earlier in this section.

Important

All systems must have fully qualified domain names (FQDNs). Note that when provisioning systems Foreman is able to manage DNS records locally.

Report a bug

1.3.2. Software Requirements

The Foreman server, the Compute controller node, and all Compute nodes must be registered to Red Hat Network. Each system must also be subscribed to receive both Red Hat Enterprise Linux 6 Server and Red Hat Enteprise Linux OpenStack Platform packages.

Report a bug

1.3.2.1. Register to Red Hat Network

Red Hat Enterprise Linux OpenStack Platform requires that each system in the OpenStack environment be running Red Hat Enterprise Linux Server and that all systems be signed up to receive updates from Red Hat Network using Subscription Manager. For further information on managing Red Hat subscriptions refer to the Red Hat *Subscription Management Guide*.

All steps in this procedure must be executed while logged in to the account of the **`root`** user on the system being registered.

Important

RHN Classic is intended to be used with legacy systems (Red Hat Enterprise Linux 6.0 or Red Hat Enterprise Linux 5.6 and earlier releases). It is strongly recommended that Red Hat Enterprise Linux 6.1/5.7 and later systems use Customer Portal Subscription Management, Subscription Asset Manager, or similar certificate-based subscription management service. As such these instructions are **not** intended for use on systems which have been registered to Red Hat Network using RHN Classic.

1. Run the **`subscription-manager register`** command to register the system to Red Hat Network.

   ```
   # subscription-manager register
   ```

2. Enter your Red Hat Network user name when prompted.

```
Username: admin@example.com
```

Important

Your Red Hat Network account must have Red Hat Enterprise Linux OpenStack Platform entitlements. If your Red Hat Network account does not have Red Hat Enterprise Linux OpenStack entitlements then you may register for access to the evaluation program at http://www.redhat.com/openstack/.

3. Enter your Red Hat Network password when prompted.

```
Password:
```

4. When registration completes successfully system is assigned a unique identifier.

```
The system has been registered with id: IDENTIFIER
```

The system has been registered to Red Hat Network and is ready to be attached to specific software subscriptions.

Report a bug

1.3.2.2. Red Hat Enterprise Linux Repository Configuration

Follow the steps in this procedure to register a Red Hat Enterprise Linux system to receive updates from Red Hat Network. These steps must be run while logged in as the **root** user. Repeat these steps on each system in the OpenStack environment.

1. Use the **subscription-manager list** command to locate the pool identifier of the Red Hat Enterprise Linux subscription.

```
# subscription-manager list --available
+-------------------------------------------+
    Available Subscriptions
+-------------------------------------------+

Product Name:           Red Hat Enterprise Linux Server
Product Id:             69
Pool Id:                POOLID
Quantity:               1
Service Level:          None
Service Type:           None
Multi-Entitlement:      No
Expires:                01/01/2022
Machine Type:           physical
...
```

The pool identifier is indicated in the **Pool Id** field associated with the **Red Hat Enterprise Linux Server** product. The identifier will be unique to your subscription. Take note of this identifier as it will be required to perform the next step.

> **Note**
>
> The output displayed in this step has been truncated to conserve space. All other available subscriptions will also be listed in the output of the command.

2. Use the **`subscription-manager attach`** command to attach the subscription identified in the previous step.

   ```
   # subscription-manager attach --pool=POOLID
   Successfully attached a subscription for Red Hat Enterprise Linux Server.
   ```

   Replace ***`POOLID`*** with the unique identifier associated with your Red Hat Enterprise Linux Server subscription. This is the identifier that was located in the previous step.

3. Run the **`yum repolist`** command. This command ensures that the repository configuration file **`/etc/yum.repos.d/redhat.repo`** exists and is up to date.

   ```
   # yum repolist
   ```

   Once repository metadata has been downloaded and examined, the list of repositories enabled will be displayed, along with the number of available packages.

   ```
   repo id                   repo name
   status
   rhel-6-server-rpms        Red Hat Enterprise Linux 6 Server (RPMs)      8,816
   repolist: 8,816
   ```

   > **Note**
   >
   > The output displayed in this step may differ from that which appears when you run the **`yum repolist`** command on your system. In particular the number of packages listed will vary if or when additional packages are added to the **`rhel-6-server-rpms`** repository.

You have successfully configured your system to receive Red Hat Enterprise Linux updates from Red Hat Network.

Report a bug

1.3.2.3. Red Hat Enterprise Linux OpenStack Platform Repository Configuration

Follow the steps in this procedure to configure a Red Hat Enterprise Linux system to receive OpenStack packages and updates from Red Hat Network. Access to a Red Hat software entitlement that includes Red Hat Enterprise Linux OpenStack Platform is required, such entitlements include:

- Red Hat Cloud Infrastructure
- Red Hat Cloud Infrastructure (without Guest OS)
- Red Hat Enterprise Linux OpenStack Platform
- Red Hat Enterprise Linux OpenStack Platform Preview
- Red Hat Enterprise Linux OpenStack Platform (without Guest OS)

These steps must be run while logged in as the **`root`** user. Repeat these steps on each system in the environment.

1. Use the **`subscription-manager list`** command to locate the pool identifier of the relevant Red Hat Cloud Infrastructure or Red Hat Enterprise Linux OpenStack Platform entitlement.

   ```
   # subscription-manager list --available
   +-------------------------------------------+
       Available Subscriptions
   +-------------------------------------------+
   ...
   Product Name:           ENTITLEMENT
   Product Id:             ID_1
   Pool Id:                POOLID_1
   Quantity:               3
   Service Level:          None
   Service Type:           None
   Multi-Entitlement:      No
   Expires:                02/14/2013
   Machine Type:           physical

   Product Name:           ENTITLEMENT
   Product Id:             ID_2
   Pool Id:                POOLID_2
   Quantity:               unlimited
   Service Level:          None
   Service Type:           None
   Multi-Entitlement:      No
   Expires:                02/14/2013
   Machine Type:           virtual
   ...
   ```

   Locate the entry in the list where the **`Product Name`** matches the name of the entitlement that will be used to access Red Hat Enterprise Linux OpenStack Platform packages. Take note of the pool identifier associated with the entitlement, this value is indicated in the **`Pool Id`** field. The pool identifier is unique to your subscription and will be required to complete the next step.

   Note

   The output displayed in this step has been truncated to conserve space. All other available subscriptions will also be listed in the output of the command.

2. Use the **`subscription-manager attach`** command to attach the subscription identified in the previous step.

   ```
   # subscription-manager attach --pool=POOLID
   Successfully attached a subscription for ENTITLEMENT.
   ```

   Replace ***`POOLID`*** with the unique identifier associated with your Red Hat Cloud Infrastructure or Red Hat Enterprise Linux OpenStack Platform entitlement. This is the identifier that was located in the previous step.

3. Install the *yum-utils* package. The *yum-utils* package is provided by the Red Hat Enterprise Linux subscription but provides the **`yum-config-manager`** utility required to complete configuration of the Red Hat Enterprise Linux OpenStack Platform software repositories.

Note that depending on the options selected during Red Hat Enterprise Linux installation the *yum-utils* package may already be installed.

4. Use the **`yum-config-manager`** command to ensure that the correct software repositories are enabled. Each successful invocation of the command will display the updated repository configuration.

   a. Ensure that the repository for Red Hat OpenStack 1.0 (Essex) has been disabled.

```
# yum-config-manager --disable rhel-server-ost-6-preview-rpms
Loaded plugins: product-id
==== repo: rhel-server-ost-6-preview-rpms ====
[rhel-server-ost-6-preview-rpms]
bandwidth = 0
base_persistdir = /var/lib/yum/repos/x86_64/6Server
baseurl =
https://cdn.redhat.com/content/beta/rhel/server/6/6Server/x86_64/opensta
ck/essex/os
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/rhel-server-ost-6-preview-
rpms
cost = 1000
enabled = False
...
```

Note

Yum treats the values **`False`** and **`0`** as equivalent. As a result the output on your system may instead contain this string:

```
enabled = 0
```

Note

If you encounter this message in the output from **`yum-config-manager`** then the system has been registered to Red Hat Network using either RHN Classic or RHN Satellite.

```
This system is receiving updates from RHN Classic or RHN
Satellite.
```

Consult the Red Hat *Subscription Management Guide* for more information on managing subscriptions using RHN Classic or RHN Satellite.

   b. Ensure that the repository for Red Hat OpenStack 2.1 (Folsom) is disabled.

```
# yum-config-manager --disable rhel-server-ost-6-folsom-rpms
Loaded plugins: product-id
==== repo: rhel-server-ost-6-folsom-rpms ====
[rhel-server-ost-6-folsom-rpms]
bandwidth = 0
base_persistdir = /var/lib/yum/repos/x86_64/6Server
baseurl =
https://cdn.redhat.com/content/beta/rhel/server/6/6Server/x86_64/opensta
ck/folsom/os
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/rhel-server-ost-6-folsom-rpms
cost = 1000
enabled = False
...
```

c. Ensure that the repository for Red Hat Enterprise Linux OpenStack Platform 3 (Grizzly) has been enabled.

```
# yum-config-manager --enable rhel-server-ost-6-3-rpms
Loaded plugins: product-id
==== repo: rhel-server-ost-6-3-rpms ====
[rhel-server-ost-6-3-rpms]
bandwidth = 0
base_persistdir = /var/lib/yum/repos/x86_64/6Server
baseurl =
https://cdn.redhat.com/content/dist/rhel/server/6/6Server/x86_64/opensta
ck/3/os
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/rhel-server-ost-6-3-rpms
cost = 1000
enabled = True
...
```

Note

Yum treats the values **`True`** and **`1`** as equivalent. As a result the output on your system may instead contain this string:

```
enabled = 1
```

5. Run the **`yum repolist`** command. This command ensures that the repository configuration file **`/etc/yum.repos.d/redhat.repo`** exists and is up to date.

```
# yum repolist
```

Once repository metadata has been downloaded and examined, the list of repositories enabled will be displayed, along with the number of available packages.

```
repo id                    repo name                              status
rhel-6-server-rpms         Red Hat Enterprise Linux 6 Server (RPMs)    8,816
rhel-server-ost-6-3-rpms   Red Hat OpenStack 3 (RPMs)                    138
repolist: 10,058
```

Note

The output displayed in this step may differ from that which appears when you run the **`yum repolist`** command on your system. In particular the number of packages listed will vary if or when additional packages are added to the repositories.

6. Install the *yum-plugin-priorities* package. The *yum-plugin-priorities* package provides a **`yum`** plug-in allowing configuration of per-repository priorities.

```
# yum install -y yum-plugin-priorities
```

7. Use the **`yum-config-manager`** command to set the priority of the Red Hat Enterprise Linux OpenStack Platform software repository to **`1`**. This is the highest priority value supported by the *yum-plugin-priorities* plug-in.

```
# yum-config-manager --enable rhel-server-ost-6-3-rpms \
        --setopt="rhel-server-ost-6-3-rpms.priority=1"
Loaded plugins: product-id
==== repo: rhel-server-ost-6-3-rpms ====
[rhel-server-ost-6-3-rpms]
bandwidth = 0
base_persistdir = /var/lib/yum/repos/x86_64/6Server
baseurl =
https://cdn.redhat.com/content/dist/rhel/server/6/6Server/x86_64/openstack/3/
os
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/rhel-server-ost-6-3-rpms
cost = 1000
enabled = True
...
priority = 1
...
```

8. Run the **`yum`** ***`update`*** command and reboot to ensure that the most up to date packages, including the kernel, are installed and running.

```
# yum update -y
```

```
# reboot
```

You have successfully configured your system to receive Red Hat Enterprise Linux OpenStack Platform packages. You may use the **`yum repolist`** command to confirm the repository configuration again at any time.

Report a bug

Chapter 2. Installing Foreman

2.1. Prerequisites

2.1.1. Configuring the Firewall

The system that will act as the Foreman server must allow incoming network traffic on a number of ports:

TCP port 53
: The Foreman server needs to accept DNS requests on this port when provisioning systems.

UDP port 69
: The Foreman server needs to accept TFTP requests on this port when provisioning systems.

TCP ports 80 and 443
: The Foreman web user interface accepts connections on these ports.

TCP port 8140
: The Foreman server accepts connections to Puppet on this port.

Procedure 2.1. Configuring the Firewall

1. Log in to the system that will act as the Foreman server as the **`root`** user.
2. Open the **`/etc/sysconfig/iptables`** file in a text editor.
3. Add these lines to the file:

   ```
   -A INPUT -m state --state NEW -m tcp -p tcp --dport 53 -j ACCEPT
   -A INPUT -m state --state NEW -m udp -p udp --dport 69 -j ACCEPT
   -A INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT
   -A INPUT -m state --state NEW -m tcp -p tcp --dport 443 -j ACCEPT
   -A INPUT -m state --state NEW -m tcp -p tcp --dport 8140 -j ACCEPT
   ```

   They must appear *before* this line:

   ```
   -A INPUT -j REJECT --reject-with icmp-host-prohibited
   ```

Example 2.1. Foreman Server Firewall Configuration File

This is an example of a default Red Hat Enterprise Linux firewall configuration that has been updated to include rules allowing incoming network traffic on ports **80**, **443**, and **8140**:

```
# Firewall configuration written by system-config-firewall
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 53 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 69 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 443 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 8140 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

4. Restart the **iptables** service:

```
# service iptables restart
```

The firewall configuration has been updated to allow incoming network traffic on the ports required by Foreman.

Report a bug

2.2. Installing Packages

The Foreman installer is included in the *ruby193-openstack-foreman-installer* package. Additionally it is recommended that the *ruby193-foreman-selinux* package is also installed, allowing SELinux to be run in enforcing mode on the Foreman server.

Procedure 2.2. Installing Packages

1. Log in to the system that will host the Foreman installation as the **root** user.
2. Install the *ruby193-openstack-foreman-installer* and *ruby193-foreman-selinux* packages.

```
# yum install -y ruby193-openstack-foreman-installer \
                 ruby193-foreman-selinux
```

The Foreman installer is now locally installed and ready to run.

Report a bug

2.3. Configuring the Installer

The Foreman installer sets several environment variables which influence the installation process. These environment variables must be set with values specific to the target environment before running the installer.

Procedure 2.3. Configuring the Installer

1. Log in to the system that will host the Foreman installation as the **`root`** user.
2. Open the **`/usr/share/openstack-foreman-installer/bin/foreman_server.sh`** file in a text editor.

   ```
   # vi /usr/share/openstack-foreman-installer/bin/foreman_server.sh
   ```

3. Locate the OpenStack configuration keys within the file. The first configuration key listed is **`PRIVATE_CONTROLLER_IP`**.

   Example 2.2. OpenStack Configuration

   ```
   #PRIVATE_CONTROLLER_IP=10.0.0.10
   #PRIVATE_INTERFACE=eth1
   #PRIVATE_NETMASK=10.0.0.0/23
   #PUBLIC_CONTROLLER_IP=10.9.9.10
   #PUBLIC_INTERFACE=eth2
   #PUBLIC_NETMASK=10.9.9.0/24
   #FOREMAN_GATEWAY=10.0.0.1 (or false for no gateway)
   ```

4. Remove the comment character (#) from the start of each line, then edit the value of each configuration key. These configuration keys will determine the network topology of the OpenStack environment, once deployed:

   `PRIVATE_CONTROLLER_IP`

   The IP address to assign to the Compute controller node on the private OpenStack network.

   Example 2.3. Value for `PRIVATE_CONTROLLER_IP`

   `10.0.0.10`

   `PRIVATE_INTERFACE`

   The network interface on the controller node to connect to the private OpenStack network.

   Example 2.4. Values for `PRIVATE_INTERFACE`

   - ***`eth1`***
   - ***`em1`***
   - ***`p1p3`***

   `PRIVATE_NETMASK`

   The IP range to associate with the OpenStack private network is defined using Classless Inter-Domain Routing (CIDR) notation.

   Example 2.5. Value for `PRIVATE_NETMASK`

   `10.0.0.0/23`

   `PUBLIC_CONTROLLER_IP`

   The IP address to assign to the Compute controller node on the public OpenStack network.

> **Example 2.6. Value for `PUBLIC_CONTROLLER_IP`**
>
> ***`10.9.9.10`***

`PUBLIC_INTERFACE`

The network interface on the Compute controller node to connect to the public OpenStack network.

> **Example 2.7. Values for `PUBLIC_INTERFACE`**
>
> - ***`eth2`***
> - ***`em2`***
> - ***`p1p4`***

`PUBLIC_NETMASK`

The IP range to associate with the OpenStack public network is defined using Classless Inter-Domain Routing (CIDR) notation.

> **Example 2.8. Value for `PUBLIC_NETMASK`**
>
> ***`10.9.9.0/24`***

`FOREMAN_GATEWAY`

The IP address of the default gateway on the Foreman network. The nodes will use this gateway to access installation media during the provisioning process.

> **Example 2.9. Value for `FOREMAN_GATEWAY`**
>
> ***`10.0.0.1`***

Important

If bare metal provisoning is not required then set the value of the **`FOREMAN_GATEWAY`** configuration key to **`false`**. The **`FOREMAN_PROVISIONING`** configuration key found elsewhere in the file must also be set to **`false`**.

- Before:

```
if [ "x$FOREMAN_PROVISIONING" = "x" ]; then
  FOREMAN_PROVISIONING=true
fi
```

- After:

```
if [ "x$FOREMAN_PROVISIONING" = "x" ]; then
  FOREMAN_PROVISIONING=false
fi
```

Note

It is possible to use identical network definitions for the OpenStack public and private network for testing purposes. Such a configuration is not recommended for production environments.

5. Save the file and exit the text editor.

The Foreman installer is now configured and ready for use.

Report a bug

2.4. Running the Installer

The installer is a script provided by the *ruby193-openstack-foreman-installer* package for deploying Foreman. It uses Puppet manifests to deploy Foreman on the local system.

Procedure 2.4. Running the Installer

1. Log in to the system that will host the Foreman installation as the **root** user.
2. Change to the **/usr/share/openstack-foreman-installer/bin/** directory. The installer *must* be run from this directory.

```
# cd /usr/share/openstack-foreman-installer/bin/
```

3. Run the **foreman_server.sh** script.

```
# sh foreman_server.sh
```

4. A message is displayed indicating that the functionality being deployed is provided as a Technology Preview:

```
#################### RED HAT OPENSTACK #####################
Thank you for using the Red Hat OpenStack Foreman Installer!
Please note that this tool is a Technology Preview
For more information about Red hat Technology Previews, see
https://access.redhat.com/support/offerings/techpreview/
##############################################################
Press [Enter] to continue
```

Press the **Enter** key to indicate that you understand the support ramifications of the Technology Preview designation and wish to proceed with installation.

5. The installer deploys Foreman using Puppet manifests. This may take a significant amount of time.
6. A message is displayed once deployment of the Puppet manifests has been completed:

```
Foreman is installed and almost ready for setting up your OpenStack
First, you need to alter a few parameters in Foreman.
Visit:
https://FQDN/puppetclasses/quickstack::compute/edit
https://FQDN/puppetclasses/quickstack::controller/edit
Go to the Smart Class Parameters tab and work though each of the parameters
in the left-hand column

Then copy /tmp/foreman_client.sh to your openstack client nodes
Run that script and visit the HOSTS tab in foreman. Pick CONTROLLER
host group for your controller node and COMPUTE host group for the rest

Once puppet runs on the machines, OpenStack is ready!
```

In the actual output ***FQDN*** will have been replaced with the fully qualified domain name of the system to which Foreman was deployed.

Foreman has been installed successfully.

Report a bug

Chapter 3. Configuring Foreman

3.1. Changing the Password

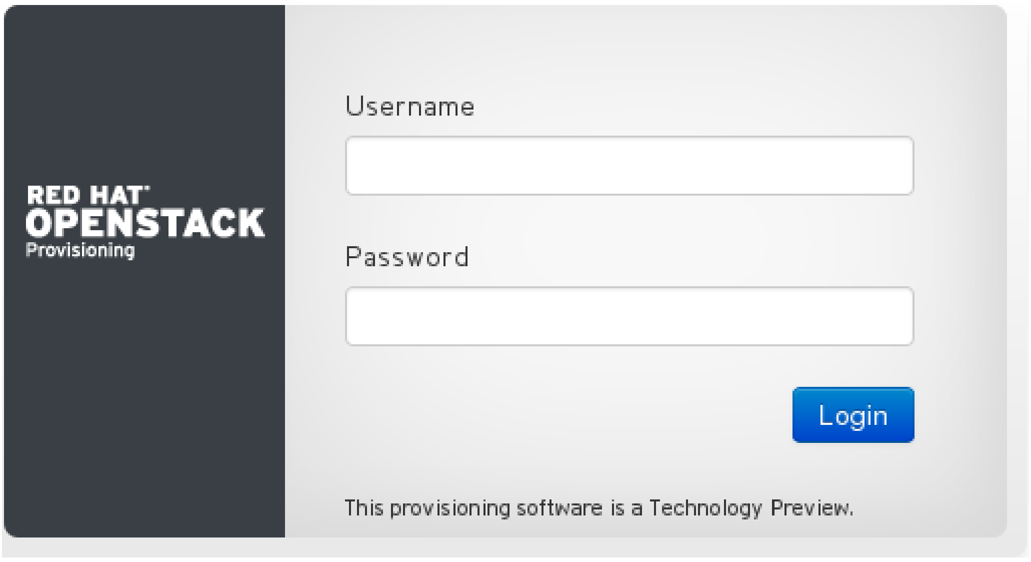
By default the Foreman installer creates an administration account with the user name **`admin`** and password **`changeme`**. It is highly recommended that users change this password immediately following installation.

Procedure 3.1. Changing the Password

1. Open a web browser either on the Foreman server itself or on a system with network access to the Foreman server.
2. Browse to **`https://`*`FQDN`*`/`**. Replace ***`FQDN`*** with the fully qualified domain name of your Foreman server.

   > **Example 3.1. Foreman URL**
   >
   > **`https://`*`foreman.example.com`*`/`**

3. The login screen is displayed. Type **`admin`** in the **`Username`** field and **`changeme`** in the **`Password`** field. Click the **`Login`** button to log in.



This provisioning software is a Technology Preview.

Figure 3.1. Foreman Login Screen

4. The **`Overview`** screen is displayed. Select the **Admin User** → **My account** option in the top right hand corner of the screen to access account settings.

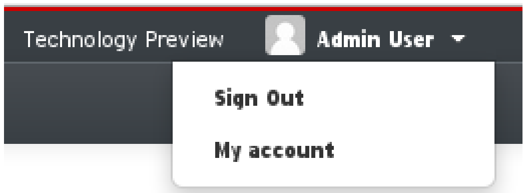


Figure 3.2. Accessing Account Settings

5. The **`Edit User`** screen is displayed. Enter a new password in the **`Password`** field.

6. Enter the new password again in the **`Verified`** field.
7. Click the **`Submit`** button to save the change.

The password of the Foreman **`admin`** user has been updated.

Report a bug

3.2. Configuring Installation Media

To support provisioning of bare-metal hosts some additional configuration is required. In particular the Foreman installation media configuration must be updated to include a path to a local copy of the Red Hat Enterprise Linux installation media. This installation media will be used when provisioning bare-metal hosts.

The installation media must already exist and be accessible using HTTP on the Foreman network. For more information on preparing installation media for network installation see the Red Hat Enterprise Linux *Installation Guide*.

Procedure 3.2. Configuring Installation Media

1. Use a web browser on a system with network access to the Foreman server to open the Foreman web user interface.
2. Log in using the **`admin`** user and the password that was set in Section 3.1, “Changing the Password”.
3. Click **More** → **Provisioning** → **Installation Media** in the top right hand corner of the page.

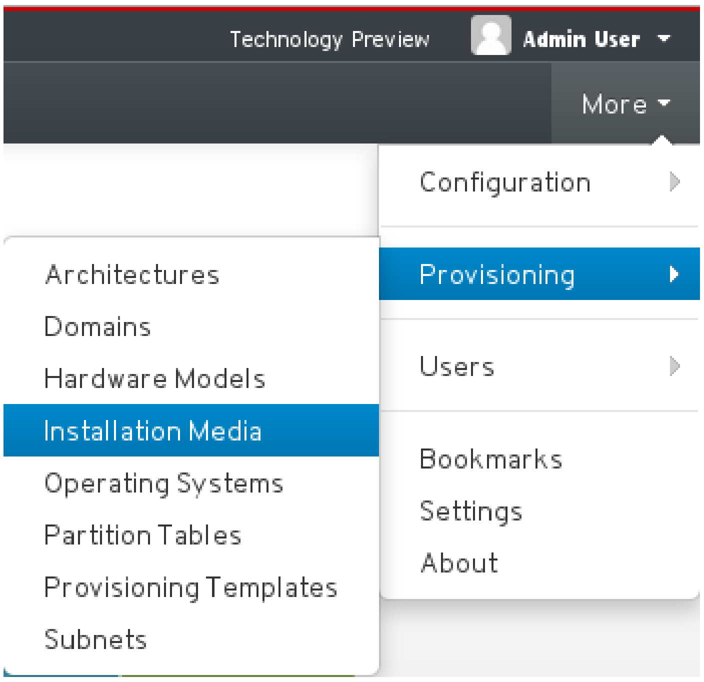
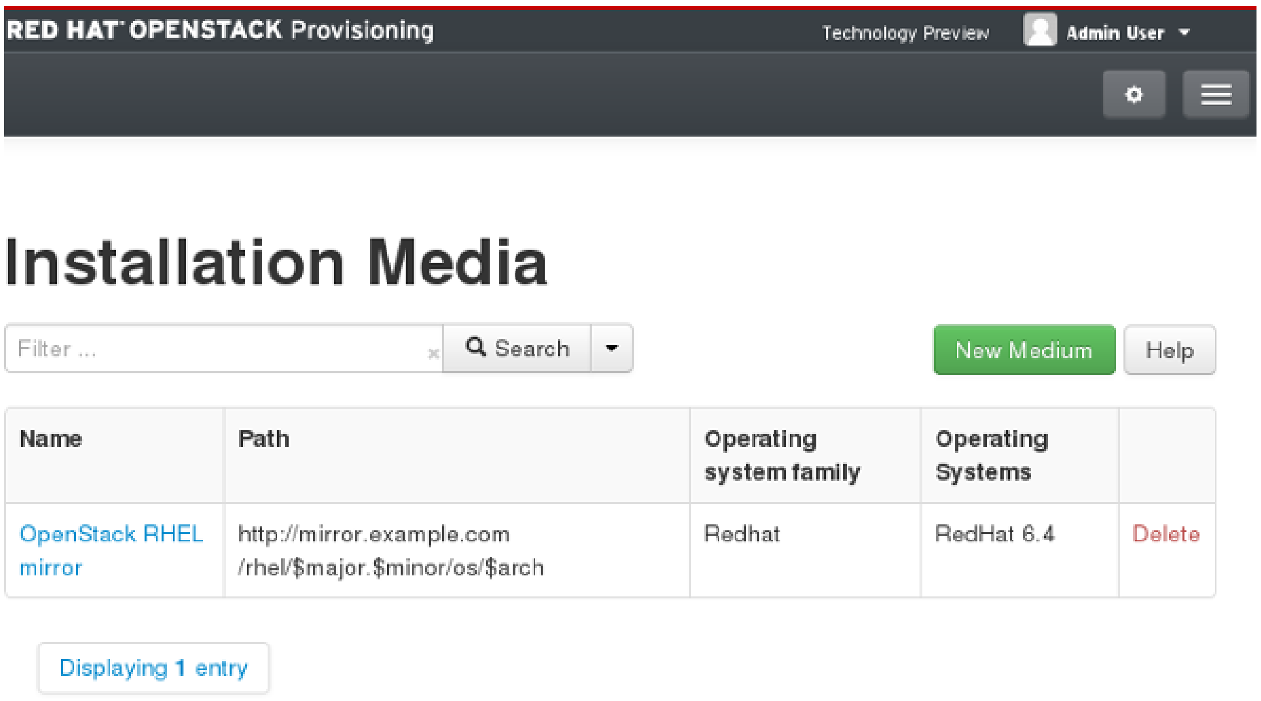


Figure 3.3. Provisioning Menu Items

4. The **`Installation Media`** page is displayed. An **`OpenStack RHEL mirror`** entry already exists by default but the associated path is only provided as an example and must be corrected.



Name	Path	Operating system family	Operating Systems	
OpenStack RHEL mirror	http://mirror.example.com /rhel/$major.$minor/os/$arch	Redhat	RedHat 6.4	Delete

Figure 3.4. Installation Media Page

5. Click the **`OpenStack RHEL mirror`** entry.
6. The **`Edit Medium`** page is displayed.
7. Update the **`Path`** field to contain the URL of a local installation mirror. These variables can be used in the URL and will be replaced automatically:

   `$arch`
   : The system architecture, for example **`x86_64`**.

   `$version`
   : The operating system version, for example **`6.4`**.

   `$major`
   : The operating system major version, for example **`6`**.

   `$minor`
   : The operating system minor version, for example **`4`**.

> **Example 3.2. Path**
>
> ***`http://download.example.com/rhel/$version/Server/$arch/os/`***

8. Click **`Submit`** to save the updated **`Path`**.

The Foreman **`OpenStack RHEL mirror`** installation media configuration has been updated.

Report a bug

3.3. Editing Host Groups

Foreman allows users to override the parameters that will be used when adding new hosts to a host group. In particular the host groups contain parameters for manipulating the passwords that will be used for a variety of user and service accounts that will be created on hosts that Foreman manages.

Important

It is highly recommended that users edit the value of the **`admin_password`** configuration key in the OpenStack Controller host group. This configuration key determines the password to be used when logging into the OpenStack dashboard as the **`admin`** user.

Procedure 3.3. Editing Host Groups

1. Use a web browser on a system with network access to the Foreman server to open the Foreman web user interface.
2. Log in using the **`admin`** user and the associated password.
3. Click **More** → **Configuration** → **Host Groups**.

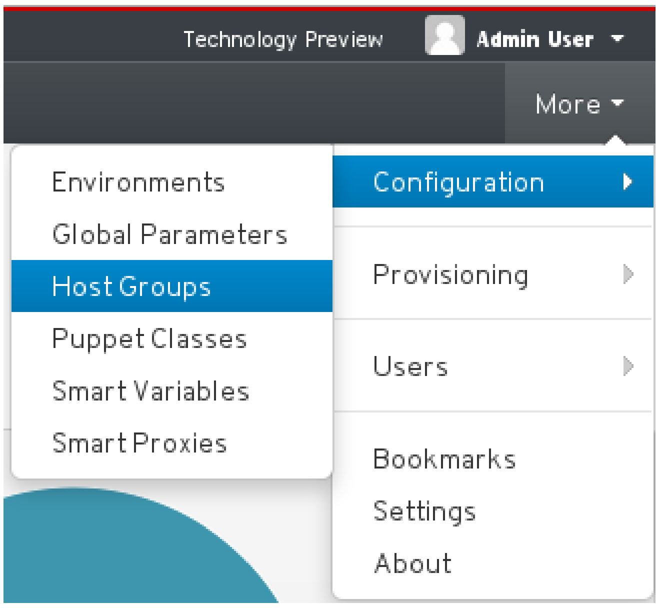
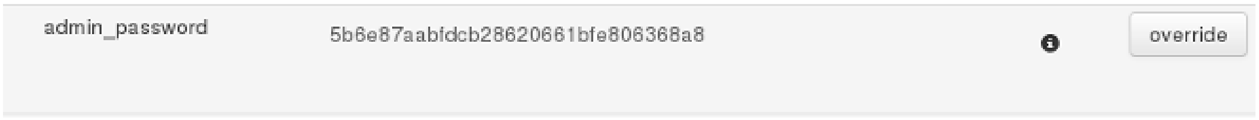


Figure 3.5. The Host Groups Menu Item

4. The **`Host Groups`** page is displayed. The available options are:
   - OpenStack Controller
   - OpenStack Nova Compute

   Click the name of the host group to edit.
5. The **`Edit`** page is displayed. Select the **`Parameters`** tab.
6. The list of parameters associated with the host group is displayed. For each parameter to be edited click the associated **`override`** button. A text field will appear at the bottom of the page. Enter the new value for the parameter in the text field.



admin_password 5b6e87aabfdcb28620661bfe806368a8 override

Figure 3.6. Override Button

For more information on the available host group parameters see:

- Section 3.3.1, “Controller Node”
- Section 3.3.2, “Compute Node”

7. Repeat the previous step for each parameter in the host group that needs to be edited.
8. Click **`Submit`** to save the updated parameter values.

The host group has been updated and the values of the selected parameters have been overridden.

Report a bug

3.3.1. Controller Node

Table 3.1. Controller Node Parameters

Parameter Name	Default Value	Description
`admin_email`	admin@***`DOMAIN`***	The email address to associate with the OpenStack **`admin`** user when it is created using the Identity service.
`admin_password`	Random	The password to associate with the **`admin`** Identity user when it is created. This is the password of the user that will be used to administer the cloud.
`cinder_db_password`	Random	The password to associate with the **`cinder`** database user, for use by the Block Storage service.
`cinder_user_password`	Random	The password to associate with the **`cinder`** Identity service user, for use by the Block Storage service.
`glance_db_password`	Random	The password to associate with the **`glance`** database user, for use by the Image Storage service.
`glance_user_password`	Random	The password to associate with the **`glance`** Identity service user, for use by the Image Storage service.
`horizon_secret_key`	Random	The unique secret key to be stored in the Dashboard configuration.
`keystone_admin_token`	Random	The unique administrator token to be used by the Identity service. This token can be used by an administrator to access the Identity service when normal user authentication is not working or not yet configured.
`keystone_db_password`	Random	The password to associate with the **`keystone`** database user, for use by the Identity service.
`mysql_root_password`	Random	The password to associate with the **`root`** database user, for use when administering the database.
`nova_db_password`	Random	The password to associate with the **`nova`** database user, for use by the Compute service.
`nova_user_password`	Random	The password to associate with the **`nova`** Identity service user,

		for use by the Compute service.
pacemaker_priv_floating_ip		
pacemaker_pub_floating_ip		
verbose	**true**	Boolean value indicating whether or not verbose logging information must be generated.

Report a bug

3.3.2. Compute Node

Table 3.2. Compute Node Parameters

Parameter Name	Default Value	Description
fixed_network_range		
floating_network_range		
nova_db_password	Random	The password associated with the **nova** database user. This password must match the value used for the same field in the controller host group.
nova_user_password	Random	The password associated with the **nova** Identity service user. This password must match the value used for the same field in the controller host group.
pacemaker_priv_floating_ip		
private_interface	**eth1**	The interface to attach to the private OpenStack network.
public_interface	**eth2**	The interface to attach to the public OpenStack network.
verbose	**true**	Boolean value indicating whether or not verbose logging information must be generated.

Report a bug

Chapter 4. Adding Hosts

To deploy OpenStack on hosts in a Foreman managed environment it is necessary to add the hosts to Foreman. Hosts are added by either adding the Puppet agent to existing servers or using the PXE boot provided by Foreman to provision new hosts.

Once the hosts have been added to Foreman they are assigned to one of the host groups defined in earlier procedures. The host group selected for each host determines which of the provided deployment templates will be applied to it. Once a host group is selected the associated templates are then applied to the host when the Puppet agent next connects to Foreman.

- To add existing Red Hat Enterprise Linux hosts to Foreman, see Section 4.1, "Registering Existing Hosts".
- To provision bare metal hosts and add them to Foreman, see Section 4.2, "Provisioning New Hosts".

Once all required hosts are visible from the Foreman web user interface follow the steps outlined in Section 4.3, "Assigning Hosts" to assign roles to the hosts and complete deployment.

Report a bug

4.1. Registering Existing Hosts

The Foreman installer generates a script for registering new hosts. By default this script is saved to the **`/tmp/foreman_client.sh`** file on the Foreman server. The script must be copied to each new host to facilitate registration with the Foreman server. When run on a new host the script performs these actions:

- Installs the *augeas* and *ruby193-puppet* packages.
- Configures the Puppet agent to access the Foreman server.
- Starts the Puppet agent.

Once started the Puppet agent registers the host to Foreman, allowing the host to be managed from the Foreman user interface.

Important

For the host to be added successfully it *must* already be registered to receive packages from the Red Hat Enterprise Linux and Red Hat OpenStack software repositories. See Section 1.3.2, "Software Requirements" for more information.

Procedure 4.1. Registering Hosts

1. Log in to the Foreman server.
2. Copy the **`/tmp/foreman_client.sh`** file from the Foreman server to the new host:

```
$ scp /tmp/foreman_client.sh USER@IP:DIR
```

Replace ***USER*** with the user to use when logging in to the new host, replace ***IP*** with the IP address or fully qualified domain name of the new host, and replace ***DIR*** with the path to the directory in which the file must be stored on the remote machine. The directory must already exist.

Example 4.1. Copying the Script

```
$ scp /tmp/foreman_client.sh root@controller.example.com:~/
```

3. Log in to the new host as the **root** user.
4. Change into the directory to which the **foreman_client.sh** script was copied:

   ```
   # cd DIR
   ```

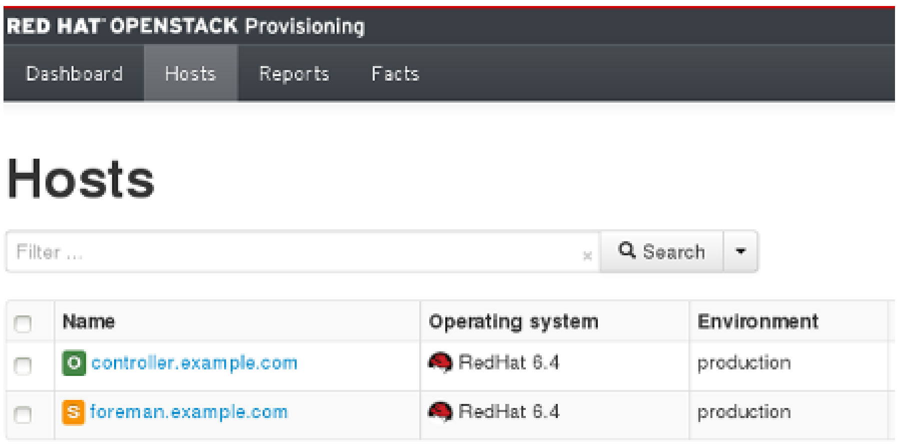
   Replace ***DIR*** with the path to the directory used when copying the file to the host.
5. Run the **foreman_client.sh** script:

   ```
   # sh foreman_client.sh
   ```

6. When the script completes successfully the Puppet agent is started and this message is displayed:

   ```
   Starting puppet agent:          [  OK  ]
   ```

7. Use a web browser on a system with network access to the Foreman server to open the Foreman web user interface.
8. Log in using the **admin** user and the password that was set in Section 3.1, “Changing the Password”.
9. Click the **Hosts** tab.
10. Verify that the newly registered host is listed:



Hosts

	Name	Operating system	Environment
	controller.example.com	RedHat 6.4	production
	foreman.example.com	RedHat 6.4	production

In this example the two hosts listed are:

- The Foreman server (**foreman.example.com**).
- The host destined to be the controller node (**controller.example.com**).

Note that although the fully qualified domain name indicates the role that the new host will take (controller node) this is not actually determined until the host is assigned to the relevant host group.

Figure 4.1. Hosts Tab

Repeat this process for all existing hosts that will be included in the OpenStack deployment. See Section 4.3, “Assigning Hosts” for information on adding the newly registered hosts to a host group and deploying OpenStack to them.

Report a bug

4.2. Provisioning New Hosts

Foreman provisions new physical hosts by using networking infrastructure it controls including DHCP, DNS, PXE, and TFTP services. New physical hosts are added to Foreman by providing the MAC address associated with the network interface of the system that is connected to the Foreman network and some other configuration details. The new host will be provisioned using this configuration when it next starts.

To provision a new host an activation key will be required. An activation key facilitates non-interactive registration of systems to Red Hat Network or an equivalent service provided by a Red Hat Network Satellite server. To learn how to create an activation key see https://access.redhat.com/site/solutions/2474. The activation key must be created with access to both the Red Hat Enterprise Linux and Red Hat OpenStack software repositories listed in Section 1.3.2, “Software Requirements”.

Procedure 4.2. Provisioning New Hosts

1. Use a web browser on a system with network access to the Foreman server to open the Foreman web user interface.
2. Log in using the **admin** user and associated password.
3. Click the **Hosts** tab.
4. Click the **New Host** button.
5. Enter the desired fully qualified domain name for the new host in the **Name** field.
6. Do not select a **Host Group** at this time.
7. Click the **Network** tab. The **Network** tab contains settings that define the networking configuration for the new host.
   a. Enter the MAC address of the network interface on the physical machine that is connected to the Foreman network in the **MAC address** field.

      Example 4.2. MAC Address

      In this example the **ip link** command is used to identify the MAC address of the **eth0** network interface.

      ```
      # ip link show eth0
      2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
      state UP qlen 1000
          link/ether 00:1a:4a:0f:18:bb brd ff:ff:ff:ff:ff:ff
      ```

      The MAC address of the **eth0** network device is ***00:1a:4a:0f:18:bb***.

   b. Ensure that the value in the **Domain** field matches the domain that Foreman is expected to manage.

      Example 4.3. Domain

      example.com

   c. Ensure that a subnet is selected in the **Subnet** field. The **IP address** field will be automatically selected based on the subnet selection.

> **Example 4.4. Subnet**
>
> ***`OpenStack (192.0.43.0/24)`***

8. Click the **`Operating System`** tab. The **`Operating System`** tab contains options for configuring the operating system installation for the host.
   a. Select **`x86_64`** in the **`Architecture`** field.
   b. Select **`RedHat 6.4`** in the **`Operating System`** field.
   c. Enter a password for the **`root`** user in the **`Root password`** field.

   > **Warning**
   >
   > It is highly recommended that a secure **`root`** password is provided. Strong passwords contain a mix of uppercase, lowercase, numeric and punctuation characters. They are six or more characters long and do not contain dictionary words. Failure to enter a secure **`root`** password in this field will result in the default **`root`** password, **`123123`**, being used.

   d. Click the **`Resolve`** button. Two templates will appear:
      - **`OpenStack PXE Template`**
      - **`OpenStack Kickstart Template`**
9. Click the **`Parameters`** tab. The **`Parameters`** tab contains settings that will be used to register the new host to Red Hat Network or a Red Hat Network Satellite server.
   A. To configure the host to register to Red Hat Network:
      a. Click the **`Override`** button next to the **`satellite_type`** configuration key. Enter **`hosted`** in the **`Value`** field.
      b. Click the **`Override`** button next to the **`satellite_host`** configuration key. Enter **`xmlrpc.rhn.redhat.com`** in the associated **`Value`** field.
      c. Click the **`Override`** button next to the **`activation_key`** configuration key. Enter the Red Hat Network activation key to be used when registering the host in the associated **`Value`** field.
   B. To configure the host to register to a Red Hat Network Satellite server:
      a. Ensure the **`satellite_type`** is set to the default value, **`site`**.
      b. Click the **`Override`** button next to the **`satellite_host`** configuration key. Enter the IP address or fully qualified domain name of the Red Hat Network Satellite server in the associated **`Value`** field.
      c. Click the **`Override`** button next to the **`activation_key`** configuration key. Enter the Red Hat Network activation key to be used when registering the host in the associated **`Value`** field.

Important

If the system is intended to be provisioned using packages from a different location instead of Red Hat Network or a Red Hat Network Satellite server then the **`OpenStack Kickstart Template`** must be edited. Access the template editor by clicking **More** → **Provisioning** → **Provisioning Templates**, selecting the **`OpenStack Kickstart Template`** entry, and removing this line from the template:

```
<%= snippets "redhat_register" %>
```

The line must be replaced with environment specific commands suitable for registering the system and adding equivalent software repositories.

10. Click the **`Submit`** button. Foreman will prepare to add the hosts the next time they boot.
11. Ensure network boot (PXE) is enabled on the new host. Instructions for confirming this will be in the manufacturer instructions for the specific system.
12. Restart the host. The host will retrieve an installation image from the Foreman PXE/TFTP server and begin the installation process.
13. The host will restart again once installation has completed. Once this occurs use the Foreman web user interface to verify that the new host appears by clicking the **`Hosts`** tab.

Repeat this process for all new hosts that will be added to the OpenStack deployment. See Section 4.3, “Assigning Hosts” for information on adding the newly added hosts to a host group and deploying OpenStack to them.

Important

Once a host has been booted successfully over the network the Foreman server updates the PXE configuration directory (**`/var/lib/tftpboot/pxelinux.cfg/`**) to ensure that future boots are performed using local boot devices. This ensures that the host does not re-provision itself when restarted. To re-provision a system that has already been registered to Foreman either:

- Use the web user interface to delete and re-add the host; or
- Use the web user interface to view the details of the host and click the **`Build`** button.

In either case use the **`Resolve Templates`** button on the resultant page to ensure the correct PXE and Kickstart templates are used. Additionally use this command on the Foreman server while logged in as the **`root`** user to allow the host to register with a new certificate:

```
# scl enable ruby193 'puppet cert clean HOST'
```

Replace ***HOST*** with the fully qualified domain name of the host.

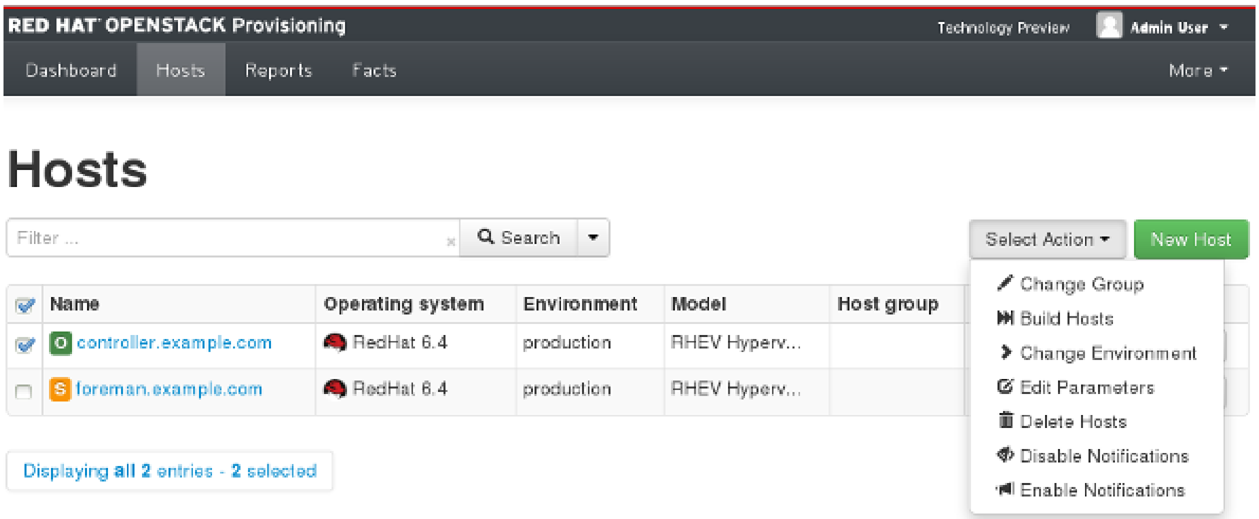
Report a bug

4.3. Assigning Hosts

Once client hosts have been registered with Foreman they must be assigned to host groups to allow installation and configuration of OpenStack. Which host group a host is assigned to defines what role it performs in the environment and which packages will be deployed.

Procedure 4.3. Assigning Hosts

1. Use a web browser on a system with network access to the Foreman server to open the Foreman web user interface.
2. Log in using the **`admin`** user and associated password.
3. Click the **`Hosts`** tab.
4. Select the host from the list displayed in the table.
5. Click the **Select Action** → **Change Group** option.



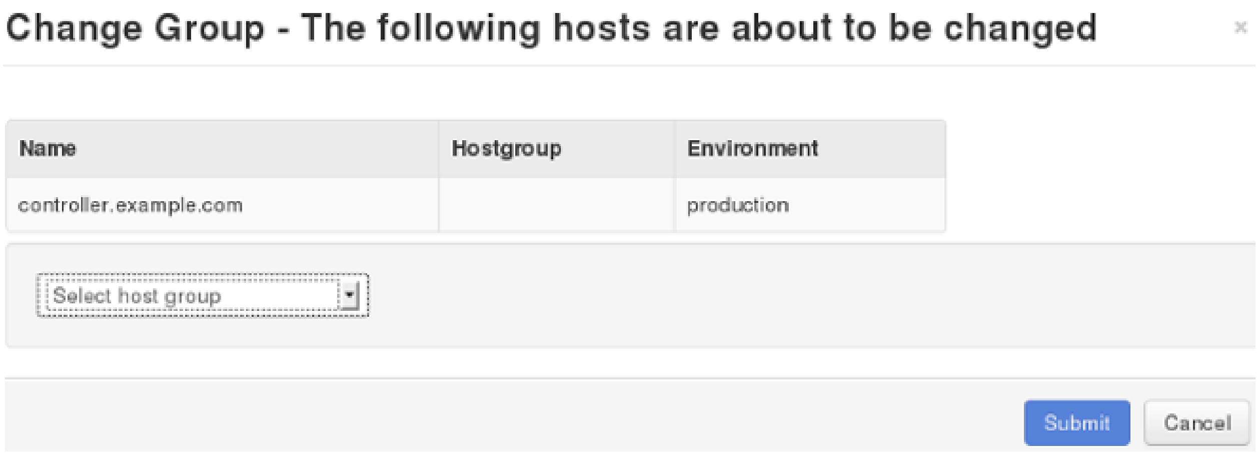
✓	Name	Operating system	Environment	Model	Host group
✓	controller.example.com	RedHat 6.4	production	RHEV Hyperv...	
☐	foreman.example.com	RedHat 6.4	production	RHEV Hyperv...	

Displaying all 2 entries - 2 selected

In this example one host (**`controller.example.com`**) is selected and the options available in the **Select Action** menu are visible.

Figure 4.2. Selecting an Action

6. The **`Change Group`** window is displayed.



Name	Hostgroup	Environment
controller.example.com		production

Figure 4.3. The `Change Group` Window

7. Use the **`Select host group`** field to select a host group. The available options are:
   - **`OpenStack Controller`**
   - **`OpenStack Nova Compute`**

If no controller node has been configured yet, select **OpenStack Controller**, otherwise select **OpenStack Nova Compute** to provision compute nodes.

Important

A controller node *must* be provisioned before attempting to provision a compute node. Puppet *must* have completed installation and configuration of the controller node before compute nodes are deployed.

8. Click the **Submit** button to save the host group selection.

The host group configuration of the host has been updated. The Puppet agent installed on the host will connect to Foreman every 30 minutes by default. When the Puppet agent next connects, the host group change will be detected and the associated configuration changes applied.

Once a controller has been provisioned using the **OpenStack Controller** host group, repeat the process to provision each compute node, selecting the **OpenStack Nova Compute** host group.

Note

To force the Puppet agent to connect to Foreman immediately, log in to the host as the **root** user and run this command:

```
# scl enable ruby193 "puppet agent --test"
```

Report a bug

Chapter 5. Logging In

Once at least one controller node and one compute node are successfully installed and configured, access the user interface with a web browser. Replace ***HOSTNAME*** with the host name or IP address of the server acting as the controller node:

- HTTPS

```
https://HOSTNAME/dashboard/
```

- HTTP

```
http://HOSTNAME/dashboard/
```

When prompted, log in using the credentials of the **`admin`** user. This is the password that was set for the **`admin_password`** configuration key of the controller node (Section 3.3.1, “Controller Node”). To begin using the OpenStack deployment refer to *Using OpenStack With the Dashboard* in the *Getting Started Guide*.

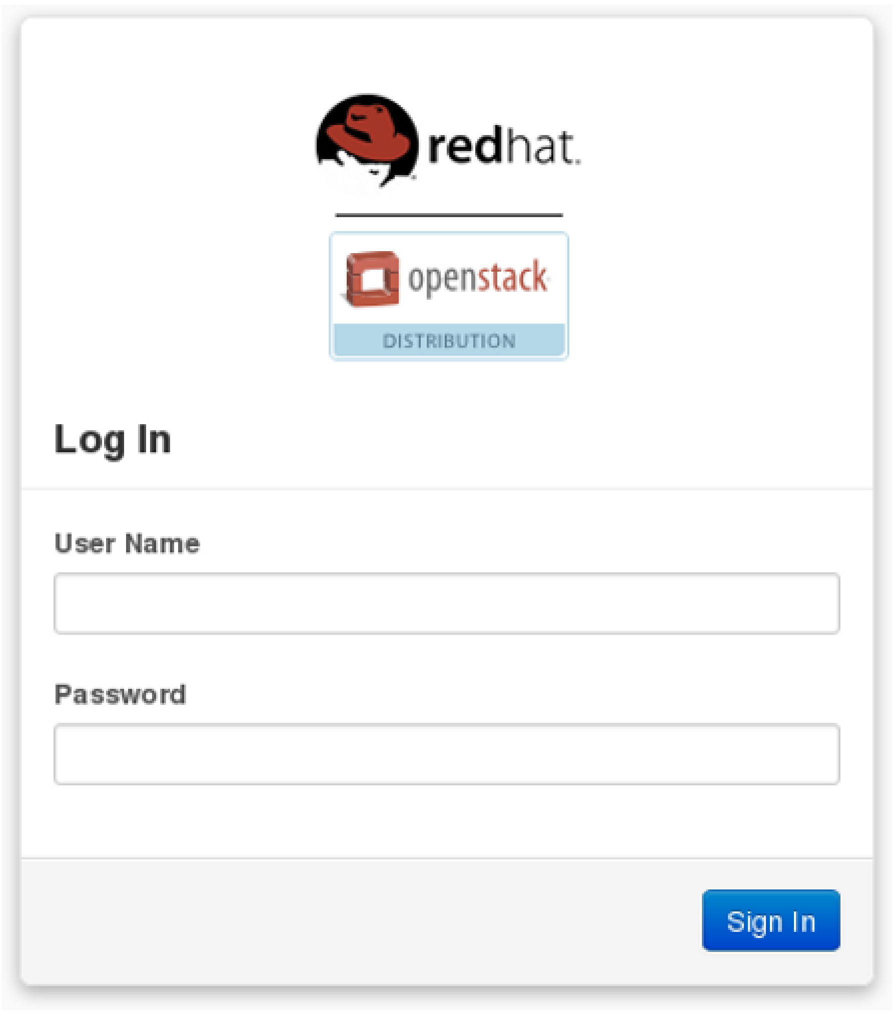


Figure 5.1. Dashboard Login Screen

Report a bug

Revision History

Revision 3.1-2	**Thu Aug 8 2013**	**Stephen Gordon**

Updated draft.

Revision 3.1-1	**Wed Jul 31 2013**	**Stephen Gordon**

BZ#986371 - Added TFTP and DNS firewall rules.

Revision 3.0-1	**Wed Jul 10 2013**	**Stephen Gordon**

Initial release.