Red Hat OpenStack Red Hat OpenStack 3.0 (Grizzly) Release Notes

Release Notes for Red Hat Enterprise Linux OpenStack Platform 3 (Grizzly)

Edition 2

Red Hat Engineering Content Services
Red Hat Engineering Content Services

Red Hat OpenStack Red Hat OpenStack 3.0 (Grizzly) Release Notes

Release Notes for Red Hat Enterprise Linux OpenStack Platform 3 (Grizzly)
Edition 2

Red Hat Engineering Content Services

Legal Notice

Keywords

Abstract

The Release Notes document the major features, enhancements, and known issues of the Red Hat Enterprise Linux OpenStack Platform 3 (Grizzly) release.

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: *doc-Release_Notes*

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. Product Introduction

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.

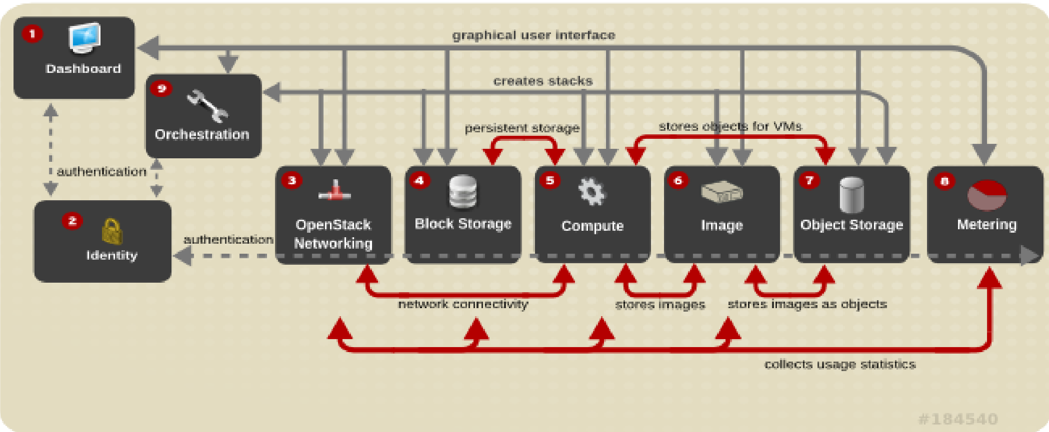
The current Red Hat system is based on OpenStack Grizzly, and packaged so that available physical hardware can be turned into a private, public, or hybrid cloud platform including:

- 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.

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 name	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

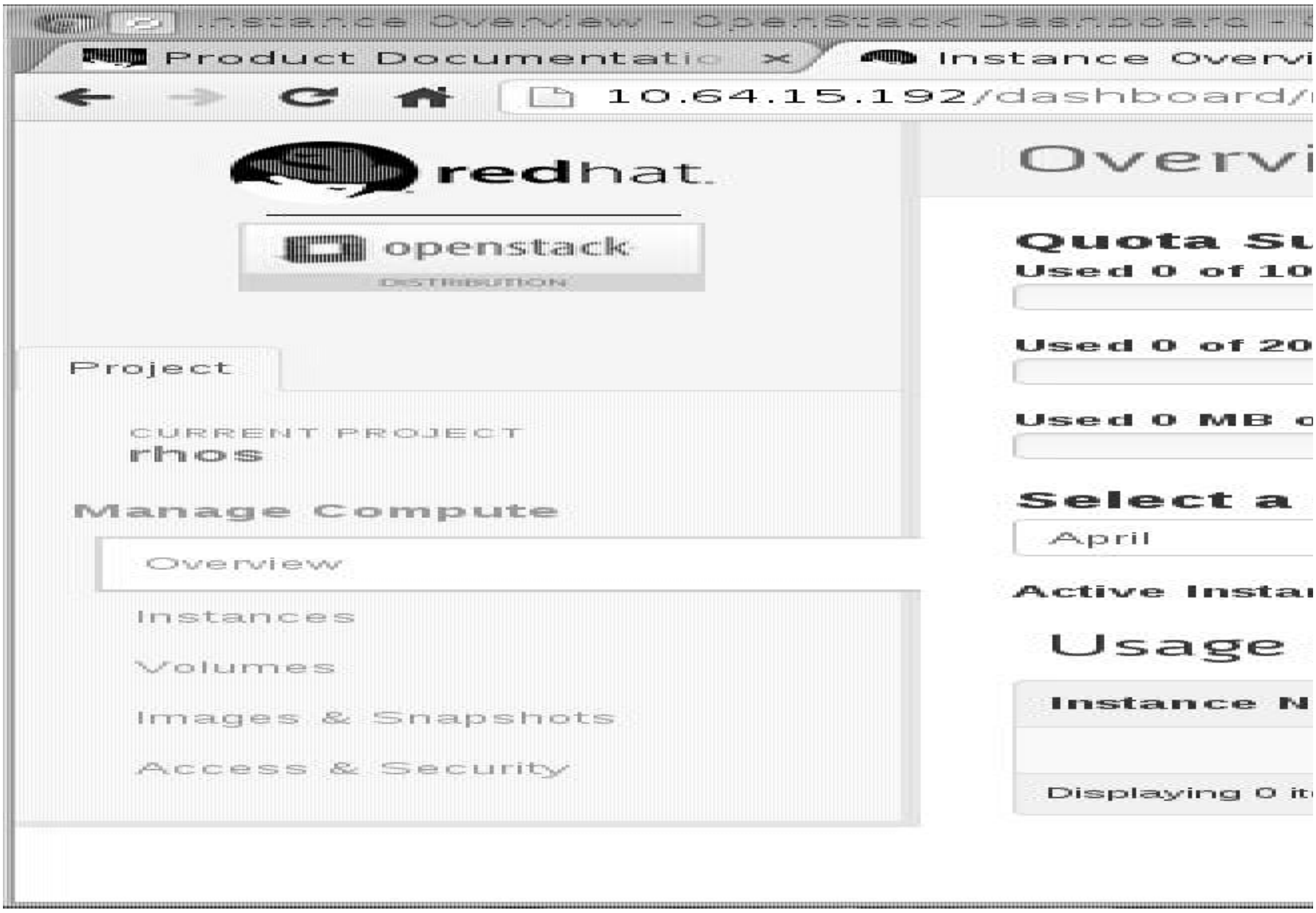
For more information on the support scope for features marked as technical previews, refer to https://access.redhat.com/support/offerings/techpreview/

1.3. Service Details

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:



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.

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`**, which is 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`**, which is the URL of an internal-facing endpoint for the service (typically same as the publicURL).
- **`publicURL`**, which is 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 is automatically defined for each cataloged service (for example, the 'glance' user for the Image service), who belongs to the special tenant '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.

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.

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). For more information, see Section 1.3.5, “Compute Service”. Both regions and zones (for details, refer to Section 1.3.7, “Object Storage Service”) 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.

1.3.5. Compute Service

The Compute service is the heart of the OpenStack cloud by providing virtual machines on demand. That is, Compute schedules virtual machines to run on a set of nodes. It does this 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 (Technical 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 meta-data. An aggregate's meta-data 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 technical 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-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 AMQP 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 enables the creation of virtual machines (that is, it is the driver for the hypervisor).
- 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.

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.

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.

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 data and aggregate it. 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.

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 orientated)
- 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.

Chapter 2. Release Introduction

2.1. About this Release

This release of Red Hat Enterprise Linux OpenStack Platform is based on the OpenStack "Grizzly" release. It includes updates made both in the initial "Grizzly" release and subsequent updates to fix various security issues and bugs. It also includes additional features, known issues, and resolved issues specific to Red Hat Enterprise Linux OpenStack Platform.

Only changes specific to Red Hat Enterprise Linux OpenStack Platform are included in this release notes document. The release notes for the OpenStack "Grizzly" release itself, and subsequent updates, are available at these locations:

OpenStack "Grizzly" Release Notes
: https://wiki.openstack.org/wiki/ReleaseNotes/Grizzly

OpenStack "Grizzly" 2013.1.1 Release Notes
: https://wiki.openstack.org/wiki/ReleaseNotes/2013.1.1

OpenStack "Grizzly" 2013.1.2 Release Notes
: https://wiki.openstack.org/wiki/ReleaseNotes/2013.1.2

> **Note**
>
> To evaluate Red Hat Enterprise Linux OpenStack Platform sign up at http://www.redhat.com/openstack/.

2.2. Product Support

Available resources include:

Customer Portal

The Red Hat Customer Portal offers a wide range of resources to help guide you through planning, deploying, and maintaining your OpenStack deployment. Facilities available via the Customer Portal include:

- Knowledge base articles and solutions.
- Reference architectures.
- Technical briefs.
- Product documentation.
- Support case management.

Access the Customer Portal at https://access.redhat.com/.

Mailing Lists

Red Hat provides these public mailing lists that are relevant to OpenStack users:

- The **rhsa-announce** mailing list provides notification of the release of security fixes for all Red Hat products, including Red Hat Enterprise Linux OpenStack Platform.

  Subscribe at https://www.redhat.com/mailman/listinfo/rhsa-announce.
- The **rhos-list** mailing list provides a forum for discussions about installing, running, and using OpenStack on Red Hat based distributions.

  Subscribe at https://www.redhat.com/mailman/listinfo/rhos-list.

> **Note**
>
> The full list of updates released for Red Hat OpenStack is maintained at https://rhn.redhat.com/errata/rhel6-rhos-folsom-errata.html.

Community Documentation

Additional documentation provided by the wider OpenStack community is available at http://docs.openstack.org.

Chapter 3. Release Notes

These release notes highlight technology preview items, recommended practices, known issues, and deprecated functionality to be taken into consideration when deploying this release of Red Hat Enterprise Linux OpenStack Platform.

Notes for updates released during the support lifecycle of this Red Hat Enterprise Linux OpenStack Platform release will appear in the advisory text associated with each update or the *Technical Notes* book.

3.1. Technology Previews

The items listed in this section are provided as Technology Previews. For further information on the scope of Technology Preview status, and the associated support implications, refer to https://access.redhat.com/support/offerings/techpreview/.

BZ#956409

```
Red Hat OpenStack includes the OpenStack Metering service (Ceilometer) as a
Technology Preview. Ceilometer provides infrastructure for collecting
metering and monitoring data from other OpenStack services.

For information on installing and configuring Ceilometer refer to
https://access.redhat.com/site/articles/406923.
```

BZ#956410

```
Red Hat OpenStack includes the OpenStack Orchestration service (Heat) as a
Technology Preview. Heat provides the ability to manage most OpenStack
resources in response to application demands based on template files
configured by the cloud administrator.

For information on installing and configuring Heat refer to
https://access.redhat.com/site/articles/406933.
```

BZ#971535

```
Foreman is being released as part of Red Hat OpenStack as a Technology
Preview feature. Rather than committing to providing full support for
Foreman in a future release, Red Hat is investigating other possible
management technologies. Over time these may replace Foreman as the
deployment mechanism for Red Hat OpenStack.

Foreman will support provisioning, via host groups. Deployment of the
following types of Red Hat OpenStack hosts is supported without further
configuration:

* OpenStack Controller: This will deploy MySQL, Qpid, Keystone, Glance,
all the Nova components and Horizon on a single machine. This configuration
will use Nova Networking (as opposed to OpenStack Networking) to provide
networking services.

* OpenStack Nova Compute: This will deploy the necessary Nova components to
run a single compute node

Other combinations of supported OpenStack services can be provisioned by
customers by creating a new host group containing the relevant Puppet
modules. Note however that  these custom host groups will not verified to
work with Red Hat OpenStack.

Foreman requires that both the Foreman server and the client systems acting
as deployment targets have Fully Qualified Domain Names (FQDN).
Additionally, the Foreman host and the OpenStack hosts must be able to reach
each other on the ports specified below.  If using Foreman for bare metal
provisioning, the Foreman server must have two (2) active Network Interface
Cards (NICs), one for outgoing traffic and one to manage the client systems.

* The Foreman server should not have SELinux set to enforcing mode, running
SELinux in permissive mode is recommended.

* The Foreman server should have its firewall configured to allow inbound
network traffic on TCP ports 80, 443 and 8140 for Foreman and Puppet to
function correctly

* If using Foreman for bare metal provisioning, the client systems should
also have firewall rules allowing inbound network traffic on TCP port 8139
to support manual Puppet runs.

For detailed installation instructions, please refer to this Knowledge Base
article:

https://access.redhat.com/site/articles/404923
```

3.2. Recommended Practices

You must take these recommended practices into account to ensure the best possible outcomes for your Red Hat Enterprise Linux OpenStack Platform deployment:

BZ#842116

```
Avoid using the nova-rootwrap command, because Nova attempts a use the sudo
chown command even if the instances directory is located on the NFS share.
In order to use nova-rootwrap, you must be aware of the issues with using
NFS and root owned files. The NFS share must be configured with the
no_root_squash option enabled.
```

BZ#843300

```
Red Hat OpenStack is fully supported for use with the AMQP messaging
provided by Apache Qpid. The RabbitMQ messaging service driver is
included, but Red Hat is unable to provide direct technical support for
this driver.
```

BZ#843302

```
Red Hat OpenStack is only supported for use with the libvirt driver, using
KVM as the hypervisor on Nova compute nodes. Red Hat is unable to provide
support for other Nova virtualization drivers, or non-KVM libvirt
hypervisors.
```

BZ#877297

```
Some packages in the Red Hat OpenStack software repositories conflict with
packages provided by the Extra Packages for Enterprise Linux (EPEL)
software repositories.

The use of Red Hat OpenStack on systems with the EPEL software repositories
enabled is unsupported.
```

BZ#892040

```
Red Hat Support maintains a list of Network Interface Cards (NICs) supported
for use with Open vSwitch in combination with VLAN tagging:

    https://access.redhat.com/knowledge/articles/289823

If you wish to use Open vSwitch in combination with VLAN tagging and your
NIC is not listed as supported then please contact Red Hat Support for more
information.
```

BZ#894440

```
Red Hat OpenStack does not yet fully support being used with ipv6
networking technologies.  Only ipv4 is supported at this time.
```

BZ#894888

```
Support for SPICE remote console access was recently added to the Compute
(Nova) and Dashboard (Horizon) services. The spice-htm5 package require to
support SPICE access is however not included in this release. As such SPICE
remote console access remains unsupported at this time.
```

BZ#912284

```
Setting the configuration option resume_guests_state_on_host_boot to True
(it is False by default) is not recommended. Setting it to True causes
problems with re-spawning instances when many services are being restarted
simultaneously. This usually occurs when the services are running on the
same host that gets restarted.
```

BZ#912744

```
If the vncserver_listen parameter in /etc/nova/nova.conf is set to the IP
address of host, then it will no longer be possible to migrate virtual
machine instances to another host. Attempts at migration will result in an
error when starting the guest on the target host.

To workaround this flaw, the listen address must be set to the wildcard
address (e.g. 0.0.0.0). Firewall rules on the host must then be used to lock
down access to the VNC console port number range (as configured in
/etc/libvirt/qemu.conf, to only those networks which should have access. In
particular care must be taken to prevent virtual machines instances from
accessing the VNC ports on the host, if the host and guest share any
networks.
```

BZ#915929

```
When using the NFS driver for the Cinder volume storage service all
compute nodes that will access volumes stored on NFS to host virtual
machines must be configured to allow this access. To ensure this is the case
log in to each node as the root user and set the virt_use_nfs SELinux
boolean to true:

# setsebool -P virt_use_nfs true
```

BZ#916649

```
The nova-manage service list only lists services listening on the message
bus, and hence should not be used to determine which OpenStack daemons are
running on a host.

Use the standard service interface to check the status instead.
```

BZ#918552

```
When using the Horizon dashboard images with these underlying disk formats
are hidden from view:

* Amazon kernel image (AKI).
* Amazon ramdisk image (ARI).

It is not possible to launch instances from disk images that use these
formats and they are not intended to be editable by regular users. To
interact with ARI and AKI formatted disk images access the dashboard as an
administrative user or use the glance command line client.
```

BZ#919046

```
Nova's cloudpipe VPN functionality is not supported. Red Hat is investing
heavily in the development of OpenStack Networking Services to provide
OpenStack networking.  Support for equivalent functionality in Nova has thus
been deferred.
```

BZ#919122

```
OpenStack Networking has several requirements to its initial setup and
configuration. A knowledge base of relevant configuration and networking
issues can be found at https://access.redhat.com/knowledge/articles/339573.
```

BZ#920282

```
Changing the network address/range for a static network using FlatDHCP
networking is an operation that requires Virtual Machine networking
downtime.

The dnsmasq process doing the DHCP is controlled by nova-networking, and it
is done in such a way that it is related to the presence of a configured
bridge that has been used at least once. The bridge is updated or launched
when nova-network is started, or when a virtual machine instance is started.

This means that Nova will not attempt to reconfigure the bridge if there is
one, so changing the range would require manually killing the running
dnsmasq process. Then restarting nova networking and all virtual machine
instances that were previously running.

Further, FlatDHCP networking will not work with multiple private networks
if you are attempting to use a single bridge interface (e.g. br100). While
a dnsmasq instance can support multiple simultaneous DHCP ranges, this
scenario is not supported by OpenStack at this time. In order to get this to
work, each distinct private network needs to have its own bridge interface.
They can all connect through the same physical NIC, but they must be
distinct bridge instances.
```

BZ#922143

```
Red Hat OpenStack includes Puppet (http://www.puppetlabs.com/). Puppet is
provided to support the rapid deployment of Red Hat OpenStack on existing
servers using the packstack utility. Use of Puppet and the provided Puppet
manifests outside of this context is not currently supported by Red Hat.
```

BZ#923621

```
In order to enable periodic cleaning of unused base images on hypervisor
nodes, both "image_cache_manager_interval" config option must be set to a
non-zero value and "remove_unused_base_images" must be set to True.

If either of these is not properly configured, Nova will not remove unused
base images, which might result in high disk utilization on hypervisor
nodes.
```

BZ#924022

```
Due to dependencies between OpenStack services, a package update will
trigger a service restart if that service is running at the time of the
update. However, to have tight control over how and when restarts are done,
you should follow a process which implements a preparation specific to your
site, then a yum update, and then a restart of services that were stopped
during the preparation.
```

BZ#924328

```
When using the following nova-manage commands you must restart all the
networking services, including all nova-network and the dnsmasq processes,
in order for the changes to be picked up:

nova-manage network create
nova-manage network delete

These commands cannot be performed via the API and need to be done through
nova-manage.
```

BZ#948249

```
Red Hat OpenStack is only supported for use with the MySQL database driver.
The PostgreSQL database driver is also included and although not yet
supported, Red Hat would like feedback on any deployment success or
problems, with a view to providing full support in a future release.

For more information regarding database support refer to
https://access.redhat.com/knowledge/articles/340383.
```

BZ#948357

```
Note that the version of Horizon dashboard shipped with the Red Hat
OpenStack 3.0 (Grizzly) release is not officially supported for use with
components of the 2.1 (Folsom) release.
```

BZ#950133

```
A rebase of openstack-nova set the default fixed-IP quota to unlimited (-1)
instead of the previous 10. This was done to avoid problems when upgrading
due to tighter constraints.

Users upgrading from Red Hat OpenStack 2.1 who have not yet changed the
value of quota_fixed_ips should be aware that they will be vulnerable to
(CVE-2013-1838) unless they change the default values. CVE-2013-1838
documents a vulnerability where the quota for fixed IPs is not properly
implemented, which could allow a remote authenticated user to cause a
denial of service by resource exhaustion and failure to spawn new instances.
```

BZ#967884

```
The recommended database backend for Ceilometer is the MongoDB database
service, provided by the mongodb package. This package is now included in
Red Hat OpenStack. The use of MongoDB is only supported as a backend for
Ceilometer. Using MongoDB for other workloads is not supported by Red Hat
at this time.
```

BZ#968213

```
OpenStack Networking provided by the component formerly known as Quantum is
only supported when network namespaces are enabled. Many common
configurations rely on support for network namespaces and will not function
correctly without them. As such the use of OpenStack Networking without a
network namespaces enabled kernel, such as the one provided in this
release, is not supported.
```

BZ#975050

```
The default PackStack configuration of Keystone generates UUID tokens.
Administrators wishing to generate and use PKI tokens must:

1) Generate the PKI files using the keystone-manage command:

# keystone-manage pki_setup \
   --keystone-user keystone \
   --keystone-group keystone

2) Ensure that Keystone has ownership of the files in the /etc/keystone/ssl/
and /var/log/keystone/ directories:

# chown -R keystone:keystone /etc/keystone/ssl/ /var/log/keystone/

3) Update the value of the token_format configuration key in
/etc/keystone/keystone.conf to PKI:

# openstack-config --set /etc/keystone/keystone.conf \
  token_format PKI

4) Restart the openstack-keystone service:

# service openstack-keystone restart
```

BZ#976116

```
It is recommended that systems used to host OpenStack API endpoints are
assigned fixed IP addresses or fully qualified domain names.

Hosting of OpenStack API endpoints on systems that have IP addresses
dynamically assigned by a DHCP server results in a loss of service if or
when the assigned address changes. When this occurs the endpoint definitions
stored in the database of the Identity service must be updated manually.
```

3.3. Known Issues

These known issues exist in Red Hat Enterprise Linux OpenStack Platform at this time:

BZ#873449

```
It is recommended that you do not run sample scripts when installing
production systems. Sample scripts are for demonstration and testing only.
Specifically, the openstack-demo-install script, will create OpenStack
Keystone accounts with default credentials.
```

BZ#889370

```
The identity server's token database table grows unconditionally over time
as new tokens are generated. Expired tokens are never automatically
deleted.

To clear the token table an administrator must backup all expired tokens for
audit purposes using an SQL select statement. Once the tokens have been
backed up the administrator must manually delete the records from the
database table.
```

BZ#912384

```
Attaching a volume stored on GlusterFS to a Nova compute instance is known
to fail when the version of the selinux-policy package installed is less
than 3.7.19-195.el6_4.2. Attaching such a volume with SELinux in Enforcing
mode will result in AVC messages being generated.

To work around this issue update to selinux-policy-3.7.19-195.el6_4.2 or
later. This package is available in the Red Hat Enterprise Linux 6.4.z
channel. Users who have updated to the latest version of selinux-policy
package are able to run in Enforcing mode.
```

BZ#919181

```
The quantum-l3-agent does not create firewall rules to allow traffic to be
forwarded between its interface and gateway ports, and the Red Hat
Enterprise Linux firewall's FORWARD chain rejects traffic by default. As a
consequence, traffic between virtual machines and the external network is
not forwarded.

A workaround is to add the following iptable rules on each note hosting the
quantum-l3-agent service:

# iptables -t filter -I FORWARD -i qr-+ -o qg-+ -j ACCEPT
# iptables -t filter -I FORWARD -i qg-+ -o qr-+ -j ACCEPT
# iptables -t filter -I FORWARD -i qr-+ -o qr-+ -j ACCEPT
# service iptables save

This allows traffic to be forwarded as expected.
```

BZ#919497

```
Glance does not fully support a graceful restart yet. Hence, image
transfers that are still in progress will be lost when Glance services are
restarted.  This will occur when updating the openstack-glance package.

The workaround to avoid losing images is to wait for image transfers that
are in progress to complete, before updating the openstack-glance package
or restarting Glance services.

If there are no image transfers in progress during installation of a new
version of the openstack-glance package or during a restart of the Glance
services, then this problem will not occur.
```

BZ#920638

```
When the openvswitch quantum plugin is used and Nova is configured with
"libvirt_vif_driver = nova.virt.libvirt.vif.LibvirtHybridOVSBridgeDriver",
the necessary forwarding rules are not created automatically. As a result
the Red Hat Enterprise Linux firewall blocks forwarding of network traffic
between virtual machine instances located on different compute nodes.

Workarounds to avoid blocking traffic between VMs located on different
compute nodes:

* If using nova security groups, add the following iptables rule on each
compute node:

iptables -t filter -I FORWARD -i qbr+ -o qbr+ -j ACCEPT
service iptables save

Either reboot or restart nova-compute after adding this rule, since the
rules nova-compute adds at startup must precede this rule.

* If not using nova security groups, an alternative solution is to set
"libvirt_vif_driver =
nova.virt.libvirt.vif.LibvirtOpenVswitchVirtualPortDriver" in the
/etc/nova/nova.conf configuration file.
```

BZ#920662

```
The limit set for Nova processes may be exceeded in very large
deployments. Then a problem may occur where you get AVC denials for
sys_resource and sys_admin while running Nova. For example:

avc:  denied  { sys_admin } for  pid=16497 comm="iptables-save"
capability=21  scontext=unconfined_u:system_r:iptables_t:s0
tcontext=unconfined_u:system_r:iptables_t:s0 tclass=capability

Due to the way process inheritance is set up on Linux, calling sudo inherits
the caller's ulimit. Processes owned by the new UID are counted against the
inherited ulimit. Transitioning to the iptables domain drops the ability to
break the soft ulimit for number of processes, which causes iptables
commands to fail in certain cases. Currently the limit to the number of
processes is set to 2048 for the Nova user.

While this limit should work for most installations, very large deployments
may need a workaround.  The workaround is to increase the limit by editing
the /etc/security/limits.d/91-nova.conf file. For example, change:

nova     soft    nproc     2048

to:

nova     soft    nproc     4096
```

BZ#953637

```
The start and reboot actions of the Compute command line interface, nova,
do not restore block device connections. As a result these actions will not
start virtual machines with attached volumes following a reboot of the
compute node on which they are hosted.

To work around this issue provide the --hard argument to the reboot action,
forcing all block device and network connections associated with the
virtual machine to be re-established. For example:

nova reboot --hard 52a15fce-748b-463a-b1f0-57a44b09b06e

Once an instance has been started with "nova reboot --hard", the standard
start and reboot commands will function as expected until such time that the
host compute node is rebooted again.
```

BZ#970424

```
When using the Gluster driver for the Cinder volume storage service all
compute nodes that will access volumes stored on Gluster to host virtual
machines must be configured to allow this access. To ensure this is the case
log in to each node as the root user and set the virt_use_fusefs SELinux
boolean to true:

# setsebool -P virt_use_fusefs true
```

BZ#973333

```
If a configuration file contains invalid or otherwise unparseable data
OpenStack services that attempt to read values from it will fail to start.
While this is expected behaviour in this situation the service
initialization scripts do not currently display any output indicating that
there was a failure.

To work around this issue when changing configuration files restart the
service as normal and then manually confirm that the service is running. For
example:

# service openstack-keystone restart
Stopping keystone:           [  OK  ]
Starting keystone:           [  OK  ]
# service openstack-keystone status
keystone (pid  12632) is running...
```

BZ#975916

```
A bug in the ThinLVM driver for the Block Storage service (Cinder) is known
to cause problems when creating volumes from snapshots. Volume creation may
fail or the volume may be created with an invalid configuration. To avoid
these issues it is recommended that the default LVM driver is used with
Cinder rather than the ThinLVM driver.
```

Chapter 4. Upgrading

Users who installed Red Hat OpenStack 2.1 (Folsom) on systems running Red Hat Enterprise Linux 6.4 Server may follow this procedure to upgrade their systems to Red Hat Enterprise Linux OpenStack Platform 3 (Grizzly).

This procedure must be followed on each system in the OpenStack environment. Additional notes on upgrading to the OpenStack "Grizzly" release are available at https://wiki.openstack.org/wiki/ReleaseNotes/Grizzly.

1. Stop all OpenStack services that are currently active on the system.
   a. Use the **openstack-status** command to identify active OpenStack services.

```
$ sudo openstack-status
== Nova services ==
openstack-nova-api            active
openstack-nova-cert           active
openstack-nova-compute        inactive
openstack-nova-network        active
openstack-nova-scheduler      active
openstack-nova-volume         inactive (disabled on boot)
== Glance services ==
openstack-glance-api          active
openstack-glance-registry     active
== Keystone service ==
openstack-keystone            active
== Horizon service ==
openstack-dashboard           active
== Cinder services ==
openstack-cinder-api          active
openstack-cinder-scheduler    active
openstack-cinder-volume       inactive
== Support services ==
httpd:                         active
libvirtd:                      active
tgtd:                          active
qpidd:                         active
memcached:                     active
== Keystone users ==
Warning keystonerc not sourced
```

> **Note**
>
> When the **openstack-status** command is run while Keystone environment variables are set additional information will be displayed. This information is not required to complete this procedure.

   b. Use the **service** command to stop each active service that has a name that starts with **openstack**.

```
$ sudo service openstack-COMPONENT stop
```

2. Use the **subscription-manager** command to verify that the system has subscriptions that provide both **Red Hat Enterprise Linux Server** and an entitlement that provides Red Hat Enterprise Linux OpenStack Platform. 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)

```
$ sudo subscription-manager list --consumed
+-------------------------------------------+
   Consumed Subscriptions
+-------------------------------------------+

Subscription Name:      ENTITLEMENT
Provides:               Red Hat OpenStack
                        Red Hat Enterprise Linux Server
SKU:                    SER0406
Contract:               3169240
Account:                901578
Serial Number:          1667264867340998574
Active:                 True
Quantity Used:          1
Service Level:          None
Service Type:           None
Starts:                 08/12/2012
Ends:                   08/12/2013
```

If the correct entitlements are not shown then review the steps for configuring software repositories as provided in the Red Hat Enterprise Linux OpenStack Platform *Getting Started Guide*.

3. Use the **yum-config-manager** command to disable Red Hat Enterprise Linux 6 beta repositories.

```
$ sudo yum-config-manager --disable rhel-6-server-beta-rpms
Loaded plugins: product-id
========== repo: rhel-6-server-beta-rpms ==========
[rhel-6-server-beta-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/os
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/rhel-6-server-beta-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
```

4. Use the **yum-config-manager** command to enable Red Hat Enterprise Linux 6 repositories.

```
$ sudo yum-config-manager --enable rhel-6-server-rpms
Loaded plugins: product-id
========== repo: rhel-6-server-rpms ==========
[rhel-6-server-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/os
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/rhel-6-server-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. Use the **yum-config-manager** command to disable Red Hat OpenStack 2.1 (Folsom) repositories.

```
$ sudo 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/dist/rhel/server/6/6Server/x86_64/openstack/fo
lsom/os
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/rhel-server-ost-6-folsom-rpms
cost = 1000
enabled = False
...
```

6. Use the **yum-config-manager** command to enable Red Hat Enterprise Linux OpenStack Platform 3 (Grizzly) repositories.

```
$ sudo 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/openstack/3/
os
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/rhel-server-ost-6-3-rpms
cost = 1000
enabled = True
...
```

7. Use the **yum update** command to ensure that your system has the most up-to-date versions of all Red Hat Enterprise Linux and Red Hat Enterprise Linux OpenStack Platform packages.

```
$ sudo yum update -y
```

Important

Red Hat Enterprise Linux OpenStack Platform includes a custom Red Hat Enterprise Linux kernel. This kernel provides support for advanced networking configurations. It is highly recommended that each system is rebooted once Yum has been used to update the installed packages, loading the new kernel.

8. When updating packages Yum attempts to also update their configuration files. In some cases when performing this task Yum finds a conflict which it is unable to resolve. In these cases Yum chooses one of the following actions depending on the options specified by the package for deploying its configuration files:

   - Creates a copy of the original configuration file from before the update commenced in a file with the suffix **.rpmsave**.

     Any user defined configuration values found in the **.rpmsave** file must be manually merged into the base configuration file.

   - Creates a copy of the new configuration file from after the update in a file with the suffix **.rpmnew**.

     Any package defined configuration values found in the **.rpmnew** file must be manually merged into the base configuration file.

   Use the **find** command to identify any instances where the update identified conflicting changes to configuration files.

```
$ sudo find /etc/ -name '*.rpm?*'
```

   Use the **diff** command to compare each file to the base configuration file. Then manually merge the conflicting changes from the **.rpmsave** and **.rpmnew** files into the base configuration file.

9. For each OpenStack component that uses a database run the appropriate database synchronization script from a host that:

   - Has the relevant service installed.
   - Is configured with the correct database connection string for the relevant service.

A. **Identity Service**

```
# keystone-manage db_sync
```

B. **Image Service**

```
# glance-manage db_sync
```

C. **Block Storage**

```
# cinder-manage db sync
```

D. **Networking**

```
# quantum-db-manage upgrade
```

E. **Compute**

```
# nova-manage db sync
```

10. In environments where Nova Networking is in use it is necessary to manually restore the **novanetwork** network following database synchronization. Log in to the system hosting the Compute database as the **root** user and run this command to restore the network:

```
# mysql -u root -p<<EOT
use nova;
update networks set deleted=0 where label='novanetwork';
EOT
```

Enter the password of the MySQL **root** user when prompted.

11. The Identity service now generates PKI tokens instead of UUID tokens by default. To ensure that token generation functions without error though it is necessary to either configure the PKI infrastructure or configure the service to continue generating UUID tokens.

A. **PKI Configuration**

To configure the PKI infrastructure files log in to the server hosting the Identity service as the **root** user. The use thus command to setup the files:

```
# keystone-manage pki_setup \
   --keystone-user keystone \
   --keystone-group keystone
```

```
# chown -R keystone:keystone /var/log/keystone \ /etc/keystone/ssl/
```

B. **UUID Configuration**

To switch back to using tokens in UUID format log in to the server hosting the Identity service as the **root** user. Then set the value of the **token_format** configuration key in the **/etc/keystone/keystone.conf** file to **UUID**.

```
# openstack-config --set /etc/keystone/keystone.conf \
   signing token_format UUID
```

12. This release of OpenStack Nova introduces the conductor. The conductor is intended to orchestrate communication between compute nodes and the database server. In previous

releases all compute nodes communicated with the database server directly.

To complete the upgrade it is necessary to install and start the conductor on at least one node. Log in to the chosen node as the **root** user and follow these steps to install the conductor:

a. Install the *openstack-nova-conductor*:

```
# yum install -y openstack-nova-conductor
```

b. Open the **/etc/nova/nova.conf** file in a text editor and set the values of these configuration keys to match those used on existing compute nodes in your environment:

   - Set **sql_connection** to a value of the form **mysql://*USER*:*PASS*@*HOST*/*DATABASE***. Replace:
     - ***USER*** with the name of the database user with access to the compute database.
     - ***PASS*** with the password of the database user with access to the compute database.
     - ***HOST*** with the hostname of the database server hosting the compute database.
     - ***DATABASE*** with the name of the compute database.

     Example 4.1. sql_connection Value

     ```
     sql_connection = mysql://nova:nova@example.com/nova
     ```

     Important

     To get the full benefit of the conductor it is also recommended that the **sql_connection** configuration key is removed from the configuration of each compute node that does not run the conductor service and does not act as a compute controller.

   - Set **rpc_backend** to **nova.openstack.common.rpc.impl_qpid**
   - Set **qpid_hostname** to the IP address or host name of the server hosting the Qpid message broker.

     Note

     If the Qpid message broker is configured to authenticate connections you must also set the values of the **qpid_username** and **qpid_password** configuration keys.
     If the Qpid message broker is configured to use SSL you must also set the values of the **qpid_protocol** and **qpid_port** configuration keys.

c. Start the **openstack-nova-conductor** service:

```
# service openstack-nova-conductor start
```

d. Enable the **openstack-nova-conductor** service permanently:

```
# chkconfig openstack-nova-conductor on
```

13. Use the **service** command to start all OpenStack services that were stopped in preparation for

the upgrade.

```
$ sudo service openstack-COMPONENT start
```

The OpenStack environment has been updated to the latest release.

Revision History

Revision 2.0-7	Tue Jun 25 2013	Steve Gordon

Red Hat OpenStack 3.0 General Availability.

Revision 2.0-2	Tue May 21 2013	Steve Gordon

Red Hat OpenStack 3.0 Preview.