Install Guide

OpenStack contributors

Oct 07, 2020

CONTENTS

CHAPTER

ONE

CONVENTIONS

The OpenStack documentation uses several typesetting conventions.

1.1 Notices

Notices take these forms:

Note: A comment with additional information that explains a part of the text.

Important: Something you must be aware of before proceeding.

Tip: An extra but helpful piece of practical advice.

Caution: Helpful information that prevents the user from making mistakes.

Warning: Critical information about the risk of data loss or security issues.

1.2 Command prompts

```
$ command
```

Any user, including the `root` user, can run commands that are prefixed with the `$` prompt.

```
# command
```

The `root` user must run commands that are prefixed with the `#` prompt. You can also prefix these commands with the **`sudo`** command, if available, to run them.

CHAPTER

TWO

PREFACE

2.1 Abstract

The OpenStack system consists of several key services that are separately installed. These services work together depending on your cloud needs and include the Compute, Identity, Networking, Image, Block Storage, Object Storage, Telemetry, Orchestration, and Database services. You can install any of these projects separately and configure them stand-alone or as connected entities.

Explanations of configuration options and sample configuration files are included.

Note: The Training Labs scripts provide an automated way of deploying the cluster described in this Installation Guide into VirtualBox or KVM VMs. You will need a desktop computer or a laptop with at least 8 GB memory and 20 GB free storage running Linux, MacOS, or Windows. Please see the OpenStack Training Labs.

This guide documents the installation of OpenStack starting with the Pike release. It covers multiple releases.

Warning: This guide is a work-in-progress and is subject to updates frequently. Pre-release packages have been used for testing, and some instructions may not work with final versions. Please help us make this guide better by reporting any errors you encounter.

2.2 Operating systems

Currently, this guide describes OpenStack installation for the following Linux distributions:

openSUSE and SUSE Linux Enterprise Server You can install OpenStack by using packages on openSUSE Leap 42.3, openSUSE Leap 15, SUSE Linux Enterprise Server 12 SP4, SUSE Linux Enterprise Server 15 through the Open Build Service Cloud repository.

Red Hat Enterprise Linux and CentOS You can install OpenStack by using packages available on both Red Hat Enterprise Linux 7 and 8 and their derivatives through the RDO repository.

..note:

```
OpenStack Ussuri is available for both CentOS 8 and RHEL 8. OpenStack
Train and earlier are available on both CentOS 7 and RHEL 7.
```

Ubuntu You can walk through an installation by using packages available through Canonicals Ubuntu Cloud archive repository for Ubuntu 16.04 (LTS).

Note: The Ubuntu Cloud Archive pockets for Pike and Queens provide OpenStack packages for Ubuntu 16.04 LTS; OpenStack Queens is installable direct using Ubuntu 18.04 LTS.

CHAPTER

THREE

GET STARTED WITH OPENSTACK

The OpenStack project is an open source cloud computing platform for all types of clouds, which aims to be simple to implement, massively scalable, and feature rich. Developers and cloud computing technologists from around the world create the OpenStack project.

OpenStack provides an *Infrastructure-as-a-Service (IaaS)* solution through a set of interrelated services. Each service offers an *Application Programming Interface (API)* that facilitates this integration. Depending on your needs, you can install some or all services.

3.1 The OpenStack services

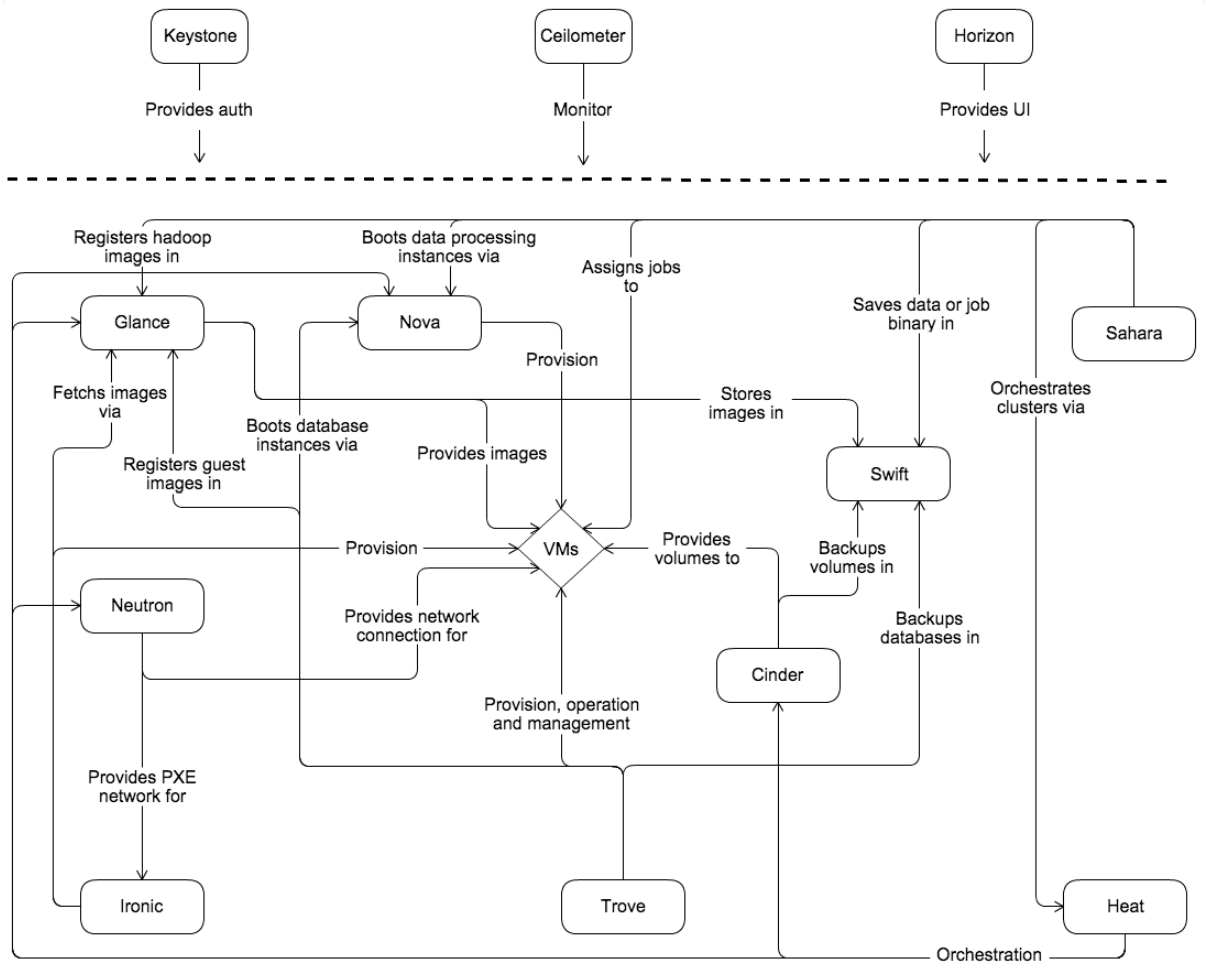
The OpenStack project navigator lets you browse the OpenStack services that make up the OpenStack architecture. The services are categorized per the service type and release series.

3.2 The OpenStack architecture

The following sections describe the OpenStack architecture in more detail:

3.2.1 Conceptual architecture

The following diagram shows the relationships among the OpenStack services:



3.2.2 Logical architecture

To design, deploy, and configure OpenStack, administrators must understand the logical architecture.

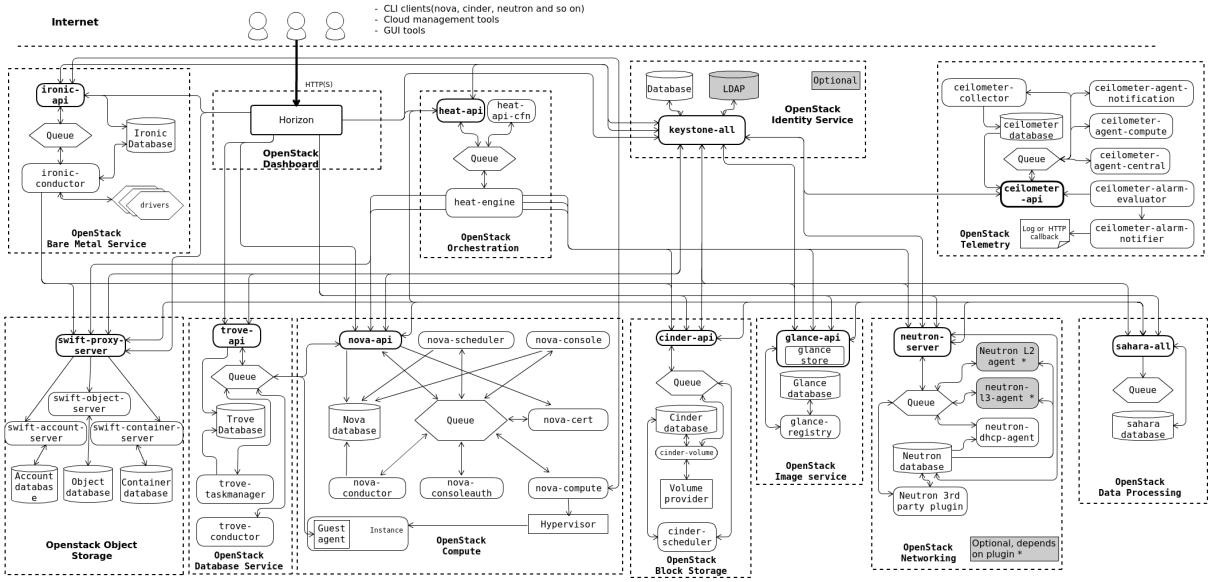
As shown in *Conceptual architecture*, OpenStack consists of several independent parts, named the OpenStack services. All services authenticate through a common Identity service. Individual services interact with each other through public APIs, except where privileged administrator commands are necessary.

Internally, OpenStack services are composed of several processes. All services have at least one API process, which listens for API requests, preprocesses them and passes them on to other parts of the service. With the exception of the Identity service, the actual work is done by distinct processes.

For communication between the processes of one service, an AMQP message broker is used. The services state is stored in a database. When deploying and configuring your OpenStack cloud, you can choose among several message broker and database solutions, such as RabbitMQ, MySQL, MariaDB, and SQLite.

Users can access OpenStack via the web-based user interface implemented by the Horizon Dashboard, via command-line clients and by issuing API requests through tools like browser plug-ins or `curl`. For applications, several SDKs are available. Ultimately, all these access methods issue REST API calls to the various OpenStack services.

The following diagram shows the most common, but not the only possible, architecture for an OpenStack cloud:



CHAPTER

FOUR

OVERVIEW

The *OpenStack* project is an open source cloud computing platform that supports all types of cloud environments. The project aims for simple implementation, massive scalability, and a rich set of features. Cloud computing experts from around the world contribute to the project.

OpenStack provides an *Infrastructure-as-a-Service (IaaS)* solution through a variety of complementary services. Each service offers an *Application Programming Interface (API)* that facilitates this integration.

This guide covers step-by-step deployment of the major OpenStack services using a functional example architecture suitable for new users of OpenStack with sufficient Linux experience. This guide is not intended to be used for production system installations, but to create a minimum proof-of-concept for the purpose of learning about OpenStack.

After becoming familiar with basic installation, configuration, operation, and troubleshooting of these OpenStack services, you should consider the following steps toward deployment using a production architecture:

- Determine and implement the necessary core and optional services to meet performance and redundancy requirements.
- Increase security using methods such as firewalls, encryption, and service policies.
- Use a deployment tool such as Ansible, Chef, Puppet, or Salt to automate deployment and management of the production environment. The OpenStack project has a couple of deployment projects with specific guides per version:
  - Ussuri release
  - Train release
  - Stein release
  - Rocky release
  - Queens release
  - Pike release

4.1 Example architecture

The example architecture requires at least two nodes (hosts) to launch a basic *virtual machine* or instance. Optional services such as Block Storage and Object Storage require additional nodes.

Important: The example architecture used in this guide is a minimum configuration, and is not intended for production system installations. It is designed to provide a minimum proof-of-concept for the purpose of learning about OpenStack. For information on creating architectures for specific use cases, or how to determine which architecture is required, see the Architecture Design Guide.

This example architecture differs from a minimal production architecture as follows:

- Networking agents reside on the controller node instead of one or more dedicated network nodes.
- Overlay (tunnel) traffic for self-service networks traverses the management network instead of a dedicated network.

For more information on production architectures for Pike, see the Architecture Design Guide, OpenStack Networking Guide for Pike, and OpenStack Administrator Guides for Pike.

For more information on production architectures for Queens, see the Architecture Design Guide, OpenStack Networking Guide for Queens, and OpenStack Administrator Guides for Queens.

For more information on production architectures for Rocky, see the Architecture Design Guide, OpenStack Networking Guide for Rocky, and OpenStack Administrator Guides for Rocky.

4.1.1 Controller

The controller node runs the Identity service, Image service, Placement service, management portions of Compute, management portion of Networking, various Networking agents, and the Dashboard. It also includes supporting services such as an SQL database, *message queue*, and *NTP*.

Optionally, the controller node runs portions of the Block Storage, Object Storage, Orchestration, and Telemetry services.

The controller node requires a minimum of two network interfaces.

4.1.2 Compute

The compute node runs the *hypervisor* portion of Compute that operates instances. By default, Compute uses the *KVM* hypervisor. The compute node also runs a Networking service agent that connects instances to virtual networks and provides firewalling services to instances via *security groups*.

You can deploy more than one compute node. Each node requires a minimum of two network interfaces.

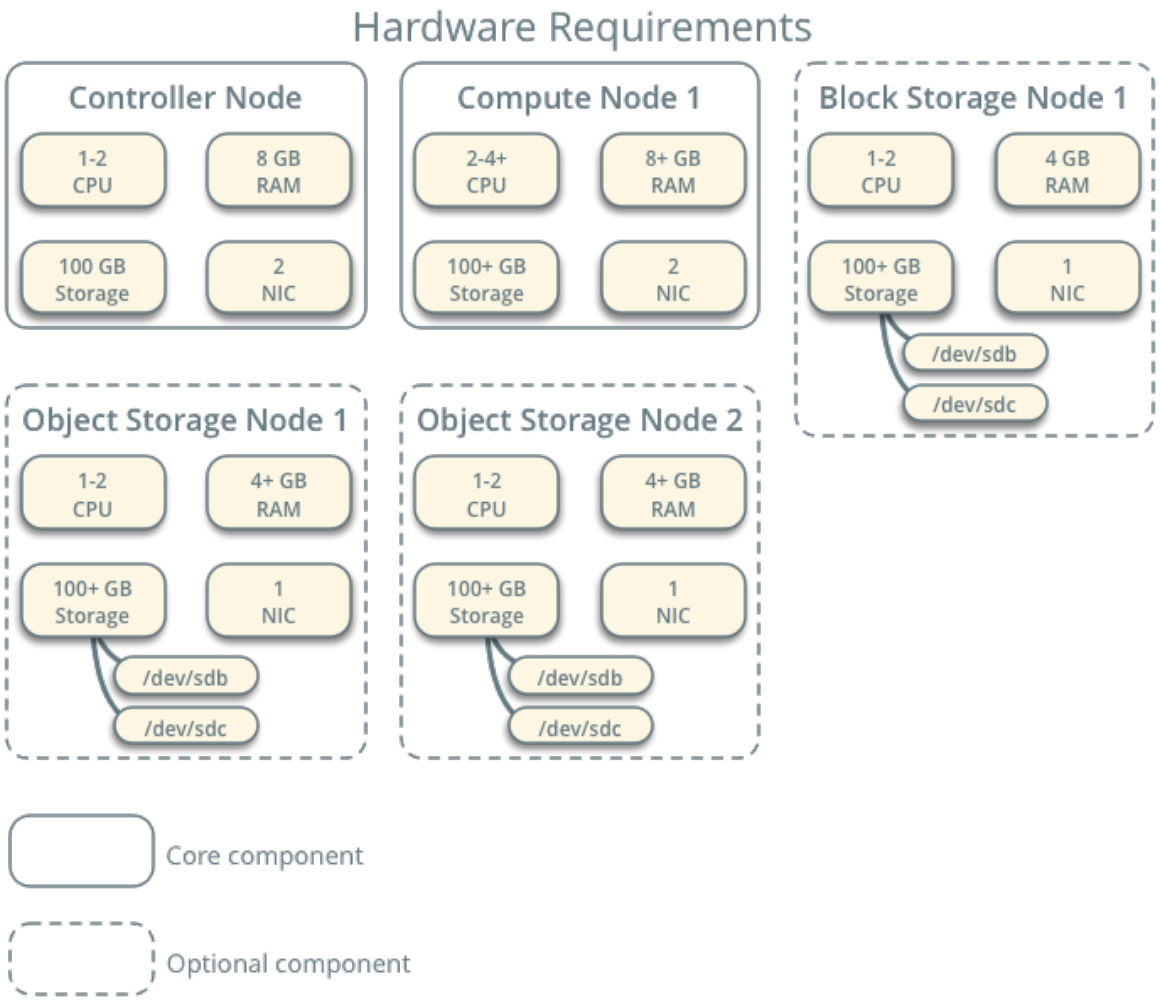


Fig. 1: **Hardware requirements**

4.1.3 Block Storage

The optional Block Storage node contains the disks that the Block Storage and Shared File System services provision for instances.

For simplicity, service traffic between compute nodes and this node uses the management network. Production environments should implement a separate storage network to increase performance and security.

You can deploy more than one block storage node. Each node requires a minimum of one network interface.

4.1.4 Object Storage

The optional Object Storage node contain the disks that the Object Storage service uses for storing accounts, containers, and objects.

For simplicity, service traffic between compute nodes and this node uses the management network. Production environments should implement a separate storage network to increase performance and security.

This service requires two nodes. Each node requires a minimum of one network interface. You can deploy more than two object storage nodes.

4.2 Networking

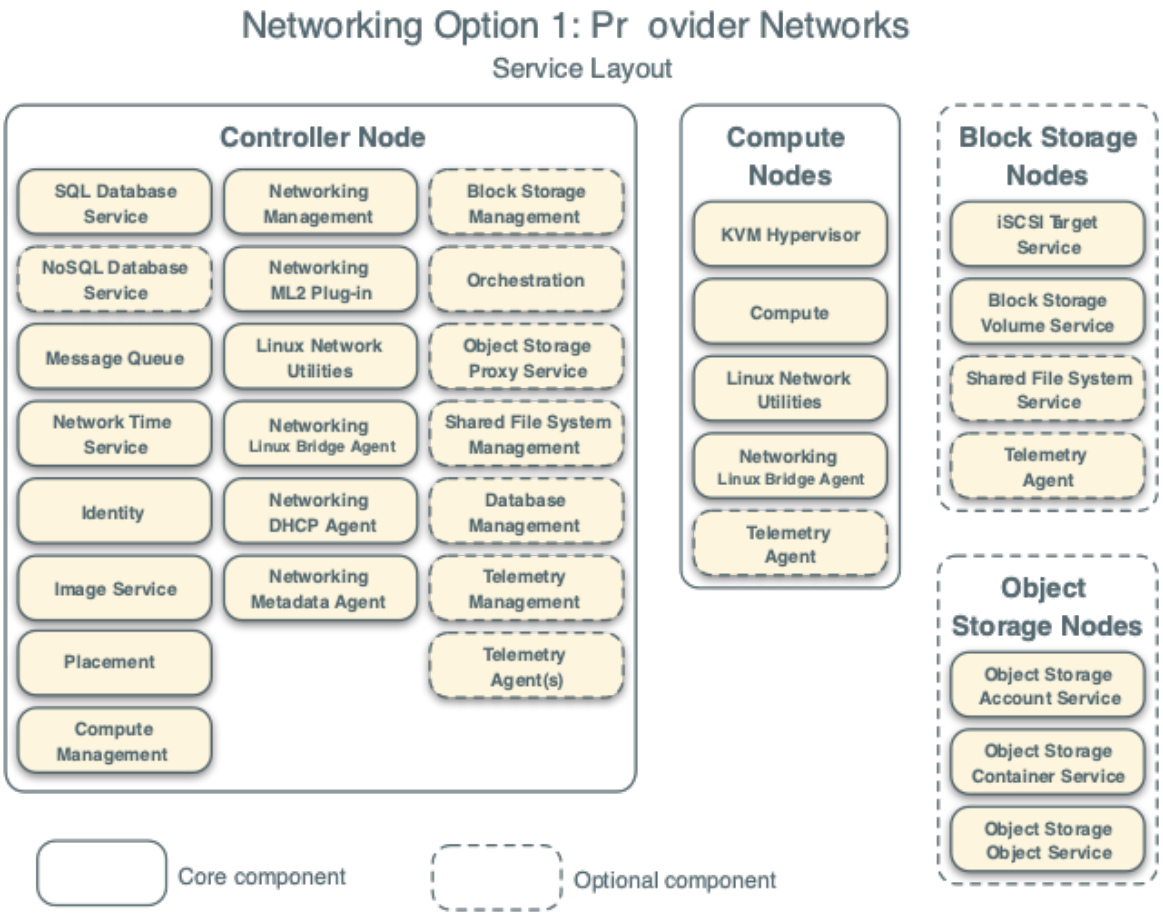
Choose one of the following virtual networking options.

4.2.1 Networking Option 1: Provider networks

The provider networks option deploys the OpenStack Networking service in the simplest way possible with primarily layer-2 (bridging/switching) services and VLAN segmentation of networks. Essentially, it bridges virtual networks to physical networks and relies on physical network infrastructure for layer-3 (routing) services. Additionally, a *DHCP* service provides IP address information to instances.

The OpenStack user requires more information about the underlying network infrastructure to create a virtual network to exactly match the infrastructure.

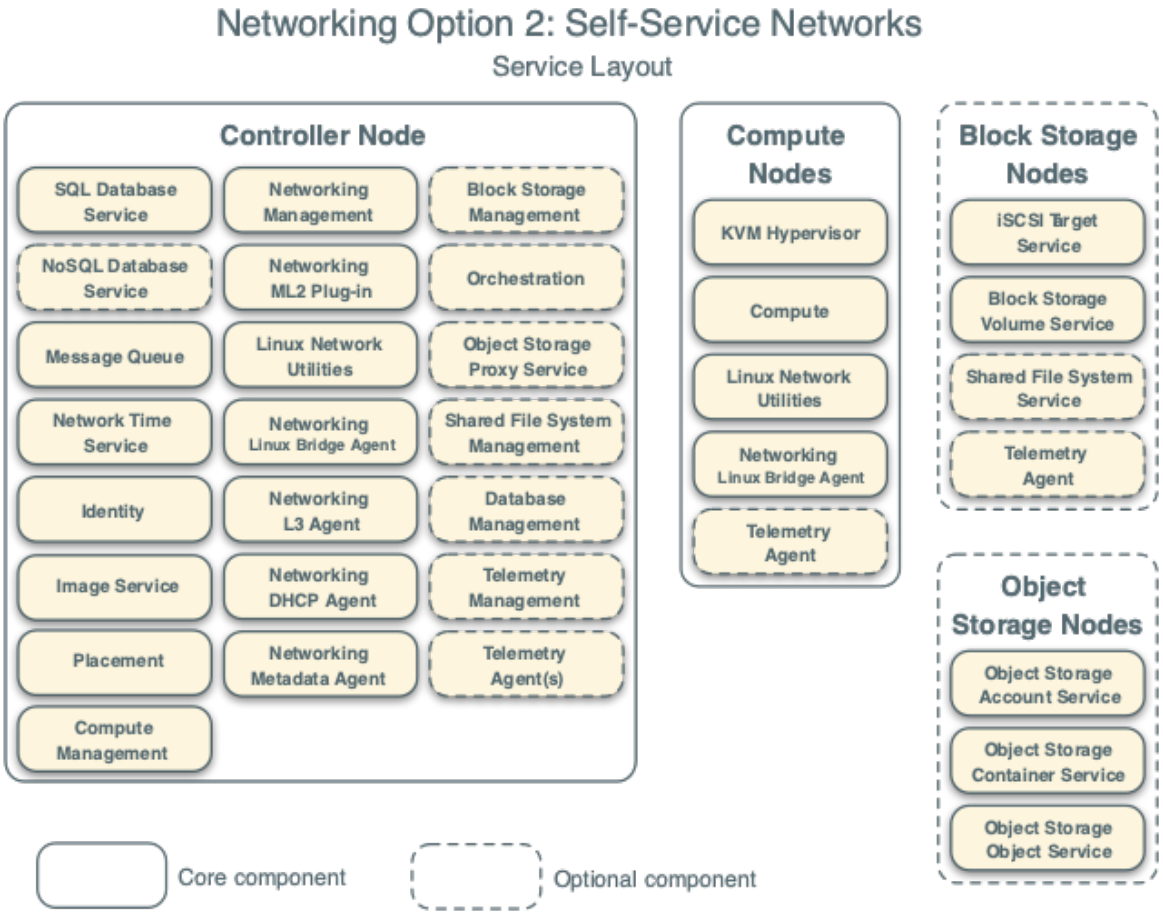
> **Warning:** This option lacks support for self-service (private) networks, layer-3 (routing) services, and advanced services such as *LBaaS* and *FWaaS*. Consider the self-service networks option below if you desire these features.


Networking Option 1: Pr ovider Networks
Service Layout
Controller Node
SQL Database Service
NoSQL Database Service
Message Queue
Network Time Service
Identity
Image Service
Placement
Compute Management
Networking Management
Networking ML2 Plug-in
Linux Network Utilities
Networking Linux Bridge Agent
Networking DHCP Agent
Networking Metadata Agent
Block Storage Management
Orchestration
Object Storage Proxy Service
Shared File System Management
Database Management
Telemetry Management
Telemetry Agent(s)
Compute Nodes
KVM Hypervisor
Compute
Linux Network Utilities
Networking Linux Bridge Agent
Telemetry Agent
Block Storage Nodes
iSCSI Target Service
Block Storage Volume Service
Shared File System Service
Telemetry Agent
Object Storage Nodes
Object Storage Account Service
Object Storage Container Service
Object Storage Object Service
Core component
Optional component

4.2.2 Networking Option 2: Self-service networks

The self-service networks option augments the provider networks option with layer-3 (routing) services that enable *self-service* networks using overlay segmentation methods such as *VXLAN*. Essentially, it routes virtual networks to physical networks using *NAT*. Additionally, this option provides the foundation for advanced services such as LBaaS and FWaaS.

The OpenStack user can create virtual networks without the knowledge of underlying infrastructure on the data network. This can also include VLAN networks if the layer-2 plug-in is configured accordingly.



CHAPTER

FIVE

ENVIRONMENT

This section explains how to configure the controller node and one compute node using the example architecture.

Although most environments include Identity, Image service, Compute, at least one networking service, and the Dashboard, the Object Storage service can operate independently. If your use case only involves Object Storage, you can skip to

- Object Storage Installation Guide for Stein
- Object Storage Installation Guide for Rocky
- Object Storage Installation Guide for Queens
- Object Storage Installation Guide for Pike

after configuring the appropriate nodes for it.

You must use an account with administrative privileges to configure each node. Either run the commands as the `root` user or configure the `sudo` utility.

Note: The **`systemctl enable`** call on openSUSE outputs a warning message when the service uses SysV Init scripts instead of native systemd files. This warning can be ignored.

For best performance, we recommend that your environment meets or exceeds the hardware requirements in *Hardware requirements*.

The following minimum requirements should support a proof-of-concept environment with core services and several *CirrOS* instances:

- Controller Node: 1 processor, 4 GB memory, and 5 GB storage
- Compute Node: 1 processor, 2 GB memory, and 10 GB storage

As the number of OpenStack services and virtual machines increase, so do the hardware requirements for the best performance. If performance degrades after enabling additional services or virtual machines, consider adding hardware resources to your environment.

To minimize clutter and provide more resources for OpenStack, we recommend a minimal installation of your Linux distribution. Also, you must install a 64-bit version of your distribution on each node.

A single disk partition on each node works for most basic installations. However, you should consider *Logical Volume Manager (LVM)* for installations with optional services such as Block Storage.

For first-time installation and testing purposes, many users select to build each host as a *virtual machine (VM)*. The primary benefits of VMs include the following:

- One physical server can support multiple nodes, each with almost any number of network interfaces.
- Ability to take periodic snap shots throughout the installation process and roll back to a working configuration in the event of a problem.

However, VMs will reduce performance of your instances, particularly if your hypervisor and/or processor lacks support for hardware acceleration of nested VMs.

Note: If you choose to install on VMs, make sure your hypervisor provides a way to disable MAC address filtering on the provider network interface.

For more information about system requirements, see the OpenStack Pike Administrator Guides, the OpenStack Queens Administrator Guides, or the OpenStack Rocky Administrator Guides.

5.1 Security

OpenStack services support various security methods including password, policy, and encryption. Additionally, supporting services including the database server and message broker support password security.

To ease the installation process, this guide only covers password security where applicable. You can create secure passwords manually, but the database connection string in services configuration file cannot accept special characters like @. We recommend you generate them using a tool such as pwgen, or by running the following command:

```
$ openssl rand -hex 10
```

For OpenStack services, this guide uses `SERVICE_PASS` to reference service account passwords and `SERVICE_DBPASS` to reference database passwords.

The following table provides a list of services that require passwords and their associated references in the guide.

Table 1: **Passwords**

Password name	Description
Database password (no variable used)	Root password for the database
ADMIN_PASS	Password of user admin
CINDER_DBPASS	Database password for the Block Storage service
CINDER_PASS	Password of Block Storage service user cinder
DASH_DBPASS	Database password for the Dashboard
DEMO_PASS	Password of user demo
GLANCE_DBPASS	Database password for Image service
GLANCE_PASS	Password of Image service user glance
KEYSTONE_DBPASS	Database password of Identity service
METADATA_SECRET	Secret for the metadata proxy
NEUTRON_DBPASS	Database password for the Networking service
NEUTRON_PASS	Password of Networking service user neutron
NOVA_DBPASS	Database password for Compute service
NOVA_PASS	Password of Compute service user nova
PLACEMENT_PASS	Password of the Placement service user placement
RABBIT_PASS	Password of RabbitMQ user openstack

OpenStack and supporting services require administrative privileges during installation and operation. In some cases, services perform modifications to the host that can interfere with deployment automation tools such as Ansible, Chef, and Puppet. For example, some OpenStack services add a root wrapper to sudo that can interfere with security policies. See the Compute service documentation for Pike, the Compute service documentation for Queens, or the Compute service documentation for Rocky for more information.

The Networking service assumes default values for kernel network parameters and modifies firewall rules. To avoid most issues during your initial installation, we recommend using a stock deployment of a supported distribution on your hosts. However, if you choose to automate deployment of your hosts, review the configuration and policies applied to them before proceeding further.

5.2 Host networking

After installing the operating system on each node for the architecture that you choose to deploy, you must configure the network interfaces. We recommend that you disable any automated network management tools and manually edit the appropriate configuration files for your distribution. For more information on how to configure networking on your distribution, see the documentation.

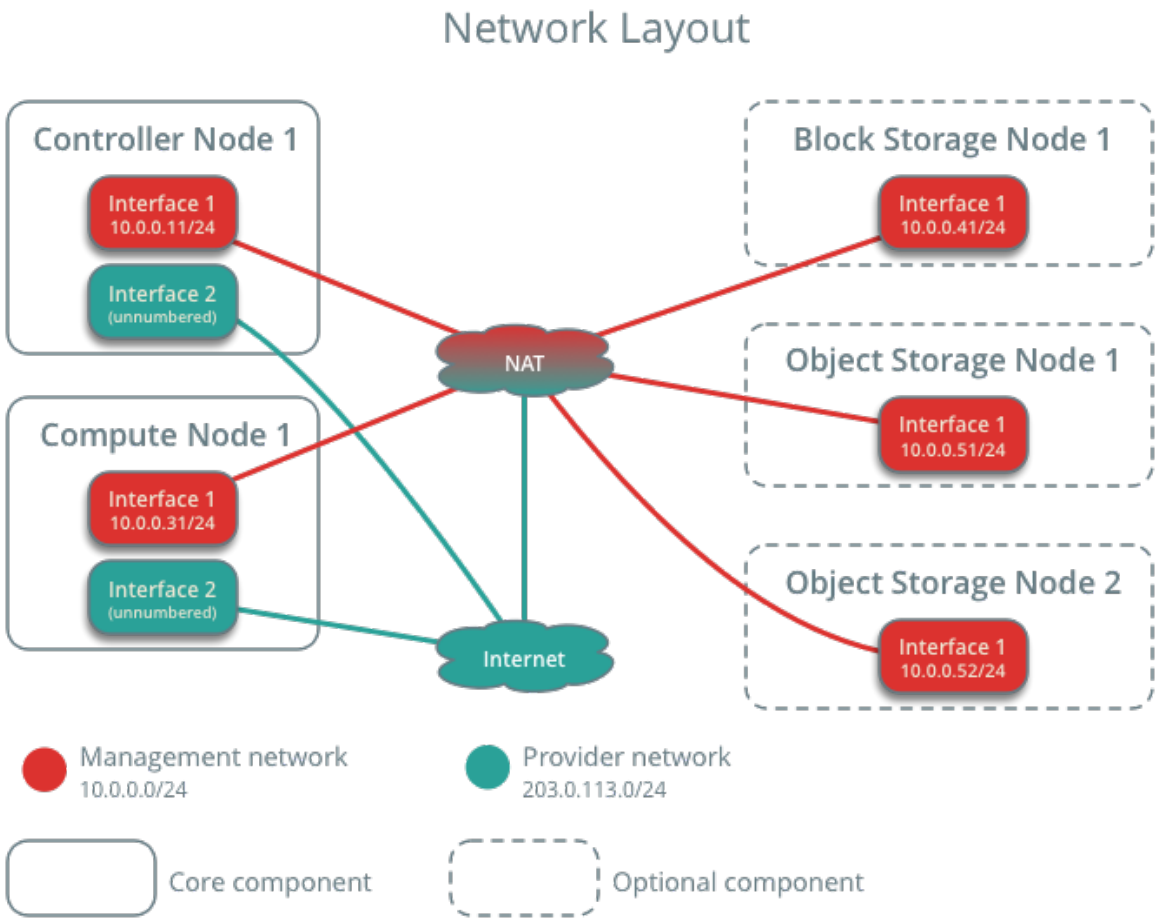
See also:

- Ubuntu Network Configuration
- RHEL 7 or RHEL 8 Network Configuration
- SLES 12 or SLES 15 or openSUSE Network Configuration

All nodes require Internet access for administrative purposes such as package installation, security updates, *DNS*, and *NTP*. In most cases, nodes should obtain Internet access through the management network interface. To highlight the importance of network separation, the example architectures use private address space for the management network and assume that the physical network infrastructure provides Internet access via *NAT* or other methods. The example architectures use routable IP address

space for the provider (external) network and assume that the physical network infrastructure provides direct Internet access.

In the provider networks architecture, all instances attach directly to the provider network. In the self-service (private) networks architecture, instances can attach to a self-service or provider network. Self-service networks can reside entirely within OpenStack or provide some level of external network access using *NAT* through the provider network.



The example architectures assume use of the following networks:

- Management on 10.0.0.0/24 with gateway 10.0.0.1

  This network requires a gateway to provide Internet access to all nodes for administrative purposes such as package installation, security updates, *DNS*, and *NTP*.

- Provider on 203.0.113.0/24 with gateway 203.0.113.1

  This network requires a gateway to provide Internet access to instances in your OpenStack environment.

You can modify these ranges and gateways to work with your particular network infrastructure.

Network interface names vary by distribution. Traditionally, interfaces use `eth` followed by a sequential number. To cover all variations, this guide refers to the first interface as the interface with the lowest number and the second interface as the interface with the highest number.

Note: Ubuntu has changed the network interface naming concept. Refer Changing Network Interfaces name Ubuntu 16.04.

Unless you intend to use the exact configuration provided in this example architecture, you must modify the networks in this procedure to match your environment. Each node must resolve the other nodes by name in addition to IP address. For example, the `controller` name must resolve to `10.0.0.11`, the IP address of the management interface on the controller node.

Warning: Reconfiguring network interfaces will interrupt network connectivity. We recommend using a local terminal session for these procedures.

Note: RHEL, CentOS and SUSE distributions enable a restrictive *firewall* by default. Ubuntu does not. For more information about securing your environment, refer to the OpenStack Security Guide.

5.2.1 Controller node

Configure network interfaces

1. Configure the first interface as the management interface:

   IP address: 10.0.0.11

   Network mask: 255.255.255.0 (or /24)

   Default gateway: 10.0.0.1

2. The provider interface uses a special configuration without an IP address assigned to it. Configure the second interface as the provider interface:

   Replace `INTERFACE_NAME` with the actual interface name. For example, *eth1* or *ens224*.

   For Ubuntu:

   - Edit the `/etc/network/interfaces` file to contain the following:

   ```
   # The provider network interface
   auto INTERFACE_NAME
   iface INTERFACE_NAME inet manual
   up ip link set dev $IFACE up
   down ip link set dev $IFACE down
   ```

   For RHEL or CentOS:

   - Edit the `/etc/sysconfig/network-scripts/ifcfg-INTERFACE_NAME` file to contain the following:

     Do not change the `HWADDR` and `UUID` keys.

   ```
   DEVICE=INTERFACE_NAME
   TYPE=Ethernet
   ```

(continues on next page)

(continued from previous page)

```
ONBOOT="yes"
BOOTPROTO="none"
```

For SUSE:

- Edit the `/etc/sysconfig/network/ifcfg-INTERFACE_NAME` file to contain the following:

```
STARTMODE='auto'
BOOTPROTO='static'
```

3. Reboot the system to activate the changes.

Configure name resolution

1. Set the hostname of the node to `controller`.
2. Edit the `/etc/hosts` file to contain the following:

```
# controller
10.0.0.11       controller

# compute1
10.0.0.31       compute1

# block1
10.0.0.41       block1

# object1
10.0.0.51       object1

# object2
10.0.0.52       object2
```

Warning: Some distributions add an extraneous entry in the `/etc/hosts` file that resolves the actual hostname to another loopback IP address such as `127.0.1.1`. You must comment out or remove this entry to prevent name resolution problems. **Do not remove the 127.0.0.1 entry.**

Note: This guide includes host entries for optional services in order to reduce complexity should you choose to deploy them.

5.2.2 Compute node

Configure network interfaces

1. Configure the first interface as the management interface:

   IP address: 10.0.0.31

   Network mask: 255.255.255.0 (or /24)

   Default gateway: 10.0.0.1

   Note: Additional compute nodes should use 10.0.0.32, 10.0.0.33, and so on.

2. The provider interface uses a special configuration without an IP address assigned to it. Configure the second interface as the provider interface:

   Replace `INTERFACE_NAME` with the actual interface name. For example, *eth1* or *ens224*.

   For Ubuntu:

   - Edit the `/etc/network/interfaces` file to contain the following:

   ```
   # The provider network interface
   auto INTERFACE_NAME
   iface  INTERFACE_NAME inet manual
   up ip link set dev $IFACE up
   down ip link set dev $IFACE down
   ```

   For RHEL or CentOS:

   - Edit the `/etc/sysconfig/network-scripts/ifcfg-INTERFACE_NAME` file to contain the following:

     Do not change the `HWADDR` and `UUID` keys.

   ```
   DEVICE=INTERFACE_NAME
   TYPE=Ethernet
   ONBOOT="yes"
   BOOTPROTO="none"
   ```

   For SUSE:

   - Edit the `/etc/sysconfig/network/ifcfg-INTERFACE_NAME` file to contain the following:

   ```
   STARTMODE='auto'
   BOOTPROTO='static'
   ```

3. Reboot the system to activate the changes.

Configure name resolution

1. Set the hostname of the node to `compute1`.
2. Edit the `/etc/hosts` file to contain the following:

```
# controller
10.0.0.11       controller

# compute1
10.0.0.31       compute1

# block1
10.0.0.41       block1

# object1
10.0.0.51       object1

# object2
10.0.0.52       object2
```

> **Warning:** Some distributions add an extraneous entry in the `/etc/hosts` file that resolves the actual hostname to another loopback IP address such as `127.0.1.1`. You must comment out or remove this entry to prevent name resolution problems. **Do not remove the 127.0.0.1 entry.**

> **Note:** This guide includes host entries for optional services in order to reduce complexity should you choose to deploy them.

5.2.3 Block storage node (Optional)

If you want to deploy the Block Storage service, configure one additional storage node.

Configure network interfaces

- Configure the management interface:
  - IP address: `10.0.0.41`
  - Network mask: `255.255.255.0` (or `/24`)
  - Default gateway: `10.0.0.1`

Configure name resolution

1. Set the hostname of the node to `block1`.
2. Edit the `/etc/hosts` file to contain the following:

```
# controller
10.0.0.11       controller

# compute1
10.0.0.31       compute1

# block1
10.0.0.41       block1

# object1
10.0.0.51       object1

# object2
10.0.0.52       object2
```

> **Warning:** Some distributions add an extraneous entry in the `/etc/hosts` file that resolves the actual hostname to another loopback IP address such as `127.0.1.1`. You must comment out or remove this entry to prevent name resolution problems. **Do not remove the 127.0.0.1 entry.**

> **Note:** This guide includes host entries for optional services in order to reduce complexity should you choose to deploy them.

3. Reboot the system to activate the changes.

5.2.4 Verify connectivity

We recommend that you verify network connectivity to the Internet and among the nodes before proceeding further.

1. From the *controller* node, test access to the Internet:

```
# ping -c 4 docs.openstack.org
PING files02.openstack.org (23.253.125.17) 56(84) bytes of data.
64 bytes from files02.openstack.org (23.253.125.17): icmp_seq=1␣
↪ttl=43 time=125 ms
64 bytes from files02.openstack.org (23.253.125.17): icmp_seq=2␣
↪ttl=43 time=125 ms
64 bytes from files02.openstack.org (23.253.125.17): icmp_seq=3␣
↪ttl=43 time=125 ms
64 bytes from files02.openstack.org (23.253.125.17): icmp_seq=4␣
↪ttl=43 time=125 ms


--- files02.openstack.org ping statistics ---
```

(continues on next page)

(continued from previous page)

```
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 125.192/125.282/125.399/0.441 ms
```

2. From the *controller* node, test access to the management interface on the *compute* node:

```
# ping -c 4 compute1

PING compute1 (10.0.0.31) 56(84) bytes of data.
64 bytes from compute1 (10.0.0.31): icmp_seq=1 ttl=64 time=0.263 ms
64 bytes from compute1 (10.0.0.31): icmp_seq=2 ttl=64 time=0.202 ms
64 bytes from compute1 (10.0.0.31): icmp_seq=3 ttl=64 time=0.203 ms
64 bytes from compute1 (10.0.0.31): icmp_seq=4 ttl=64 time=0.202 ms

--- compute1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.202/0.217/0.263/0.030 ms
```

3. From the *compute* node, test access to the Internet:

```
# ping -c 4 openstack.org

PING openstack.org (174.143.194.225) 56(84) bytes of data.
64 bytes from 174.143.194.225: icmp_seq=1 ttl=54 time=18.3 ms
64 bytes from 174.143.194.225: icmp_seq=2 ttl=54 time=17.5 ms
64 bytes from 174.143.194.225: icmp_seq=3 ttl=54 time=17.5 ms
64 bytes from 174.143.194.225: icmp_seq=4 ttl=54 time=17.4 ms

--- openstack.org ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3022ms
rtt min/avg/max/mdev = 17.489/17.715/18.346/0.364 ms
```

4. From the *compute* node, test access to the management interface on the *controller* node:

```
# ping -c 4 controller

PING controller (10.0.0.11) 56(84) bytes of data.
64 bytes from controller (10.0.0.11): icmp_seq=1 ttl=64 time=0.263 ms
64 bytes from controller (10.0.0.11): icmp_seq=2 ttl=64 time=0.202 ms
64 bytes from controller (10.0.0.11): icmp_seq=3 ttl=64 time=0.203 ms
64 bytes from controller (10.0.0.11): icmp_seq=4 ttl=64 time=0.202 ms

--- controller ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.202/0.217/0.263/0.030 ms
```

Note: RHEL, CentOS and SUSE distributions enable a restrictive *firewall* by default. During the installation process, certain steps will fail unless you alter or disable the firewall. For more information about securing your environment, refer to the OpenStack Security Guide.

Ubuntu does not enable a restrictive *firewall* by default. For more information about securing your environment, refer to the OpenStack Security Guide.

5.3 Network Time Protocol (NTP)

To properly synchronize services among nodes, you can install Chrony, an implementation of *NTP*. We recommend that you configure the controller node to reference more accurate (lower stratum) servers and other nodes to reference the controller node.

5.3.1 Controller node

Perform these steps on the controller node.

Install and configure components

1. Install the packages:

   For Ubuntu:

   ```
   # apt install chrony
   ```

   For RHEL or CentOS:

   ```
   # yum install chrony
   ```

   For SUSE:

   ```
   # zypper install chrony
   ```

2. Edit the `chrony.conf` file and add, change, or remove the following keys as necessary for your environment.

   For RHEL, CentOS, or SUSE, edit the `/etc/chrony.conf` file:

   ```
   server NTP_SERVER iburst
   ```

   For Ubuntu, edit the `/etc/chrony/chrony.conf` file:

   ```
   server NTP_SERVER iburst
   ```

   Replace `NTP_SERVER` with the hostname or IP address of a suitable more accurate (lower stratum) NTP server. The configuration supports multiple `server` keys.

   Note: By default, the controller node synchronizes the time via a pool of public servers. However, you can optionally configure alternative servers such as those provided by your organization.

3. To enable other nodes to connect to the chrony daemon on the controller node, add this key to the same `chrony.conf` file mentioned above:

   ```
   allow 10.0.0.0/24
   ```

   If necessary, replace `10.0.0.0/24` with a description of your subnet.

4. Restart the NTP service:

   For Ubuntu:

```
# service chrony restart
```

For RHEL, CentOS, or SUSE:

```
# systemctl enable chronyd.service
# systemctl start chronyd.service
```

5.3.2 Other nodes

Other nodes reference the controller node for clock synchronization. Perform these steps on all other nodes.

Install and configure components

1. Install the packages.

   For Ubuntu:

   ```
   # apt install chrony
   ```

   For RHEL or CentOS:

   ```
   # yum install chrony
   ```

   For SUSE:

   ```
   # zypper install chrony
   ```

2. Configure the `chrony.conf` file and comment out or remove all but one `server` key. Change it to reference the controller node.

   For RHEL, CentOS, or SUSE, edit the `/etc/chrony.conf` file:

   ```
   server controller iburst
   ```

   For Ubuntu, edit the `/etc/chrony/chrony.conf` file:

   ```
   server controller iburst
   ```

3. Comment out the `pool 2.debian.pool.ntp.org offline iburst` line.

4. Restart the NTP service.

   For Ubuntu:

   ```
   # service chrony restart
   ```

   For RHEL, CentOS, or SUSE:

   ```
   # systemctl enable chronyd.service
   # systemctl start chronyd.service
   ```

5.3.3 Verify operation

We recommend that you verify NTP synchronization before proceeding further. Some nodes, particularly those that reference the controller node, can take several minutes to synchronize.

1. Run this command on the *controller* node:

```
# chronyc sources

  210 Number of sources = 2
  MS Name/IP address         Stratum Poll Reach LastRx Last sample
 ␣
↪===============================================================================
  ^- 192.0.2.11                    2    7    12   137  -2814us[-
↪3000us] +/-   43ms
  ^* 192.0.2.12                    2    6   177    46    +17us[  -
↪23us] +/-   68ms
```

   Contents in the *Name/IP address* column should indicate the hostname or IP address of one or more NTP servers. Contents in the *MS* column should indicate * for the server to which the NTP service is currently synchronized.

2. Run the same command on *all other* nodes:

```
# chronyc sources

  210 Number of sources = 1
  MS Name/IP address         Stratum Poll Reach LastRx Last sample
 ␣
↪===============================================================================
  ^* controller                    3    9   377   421    +15us[  -
↪87us] +/-   15ms
```

   Contents in the *Name/IP address* column should indicate the hostname of the controller node.

5.4 OpenStack packages

Distributions release OpenStack packages as part of the distribution or using other methods because of differing release schedules. Perform these procedures on all nodes.

Note: The set up of OpenStack packages described here needs to be done on all nodes: controller, compute, and Block Storage nodes.

Warning: Your hosts must contain the latest versions of base installation packages available for your distribution before proceeding further.

Note: Disable or remove any automatic update services because they can impact your OpenStack environment.

5.4.1 OpenStack packages for SUSE

Distributions release OpenStack packages as part of the distribution or using other methods because of differing release schedules. Perform these procedures on all nodes.

Note: The set up of OpenStack packages described here needs to be done on all nodes: controller, compute, and Block Storage nodes.

Warning: Your hosts must contain the latest versions of base installation packages available for your distribution before proceeding further.

Note: Disable or remove any automatic update services because they can impact your OpenStack environment.

Enable the OpenStack repository

- Enable the Open Build Service repositories based on your openSUSE or SLES version, and on the version of OpenStack you want to install:

  On openSUSE for OpenStack Ussuri:

```
# zypper addrepo -f obs://Cloud:OpenStack:Ussuri/openSUSE_Leap_15.1␣
↪Ussuri
```

  On openSUSE for OpenStack Train:

```
# zypper addrepo -f obs://Cloud:OpenStack:Train/openSUSE_Leap_15.0␣
↪Train
```

  On openSUSE for OpenStack Stein:

```
# zypper addrepo -f obs://Cloud:OpenStack:Stein/openSUSE_Leap_15.0␣
↪Stein
```

  On openSUSE for OpenStack Rocky:

```
# zypper addrepo -f obs://Cloud:OpenStack:Rocky/openSUSE_Leap_15.0␣
↪Rocky
```

  On openSUSE for OpenStack Queens:

```
# zypper addrepo -f obs://Cloud:OpenStack:Queens/openSUSE_Leap_42.3␣
↪Queens
```

  On openSUSE for OpenStack Pike:

```
# zypper addrepo -f obs://Cloud:OpenStack:Pike/openSUSE_Leap_42.3 Pike
```

Note: The openSUSE distribution uses the concept of patterns to represent collections of packages. If you selected Minimal Server Selection (Text Mode) during the initial installation, you may be presented with a dependency conflict when you attempt to install the OpenStack packages. To avoid this, remove the minimal_base-conflicts package:

```
# zypper rm patterns-openSUSE-minimal_base-conflicts
```

On SLES for OpenStack Ussuri:

```
# zypper addrepo -f obs://Cloud:OpenStack:Ussuri/SLE_15_SP2 Ussuri
```

On SLES for OpenStack Train:

```
# zypper addrepo -f obs://Cloud:OpenStack:Train/SLE_15_SP1 Train
```

On SLES for OpenStack Stein:

```
# zypper addrepo -f obs://Cloud:OpenStack:Stein/SLE_15 Stein
```

On SLES for OpenStack Rocky:

```
# zypper addrepo -f obs://Cloud:OpenStack:Rocky/SLE_12_SP4 Rocky
```

On SLES for OpenStack Queens:

```
# zypper addrepo -f obs://Cloud:OpenStack:Queens/SLE_12_SP3 Queens
```

On SLES for OpenStack Pike:

```
# zypper addrepo -f obs://Cloud:OpenStack:Pike/SLE_12_SP3 Pike
```

Note: The packages are signed by GPG key `D85F9316`. You should verify the fingerprint of the imported GPG key before using it.

```
Key Name:         Cloud:OpenStack OBS Project <Cloud:OpenStack@build.
↪opensuse.org>
Key Fingerprint:  35B34E18 ABC1076D 66D5A86B 893A90DA D85F9316
Key Created:      2015-12-16T16:48:37 CET
Key Expires:      2018-02-23T16:48:37 CET
```

Finalize the installation

1. Upgrade the packages on all nodes:

```
# zypper refresh && zypper dist-upgrade
```

Note: If the upgrade process includes a new kernel, reboot your host to activate it.

2. Install the OpenStack client:

```
# zypper install python-openstackclient
```

5.4.2 OpenStack packages for RHEL and CentOS

Distributions release OpenStack packages as part of the distribution or using other methods because of differing release schedules. Perform these procedures on all nodes.

Warning: Starting with the Ussuri release, you will need to use either CentOS8 or RHEL 8. Previous OpenStack releases will need to use either CentOS7 or RHEL 7. Instructions are included for both distributions and versions where different.

Note: The set up of OpenStack packages described here needs to be done on all nodes: controller, compute, and Block Storage nodes.

Warning: Your hosts must contain the latest versions of base installation packages available for your distribution before proceeding further.

Note: Disable or remove any automatic update services because they can impact your OpenStack environment.

Prerequisites

Warning: We recommend disabling EPEL when using RDO packages due to updates in EPEL breaking backwards compatibility. Or, preferably pin package versions using the `yum-versionlock` plugin.

Note: The following steps apply to RHEL only. CentOS does not require these steps.

1. When using RHEL, it is assumed that you have registered your system using Red Hat Subscription Management and that you have the `rhel-7-server-rpms` or `rhel-8-for-x86_64-baseos-rpms` repository enabled by default depending on your version.

   For more information on registering a RHEL 7 system, see the Red Hat Enterprise Linux 7 System Administrators Guide.

2. In addition to `rhel-7-server-rpms` on a RHEL 7 system, you also need to have the `rhel-7-server-optional-rpms`, `rhel-7-server-extras-rpms`, and `rhel-7-server-rh-common-rpms` repositories enabled:

```
# subscription-manager repos --enable=rhel-7-server-optional-rpms \
  --enable=rhel-7-server-extras-rpms --enable=rhel-7-server-rh-common-
↪rpms
```

For more information on registering a RHEL 8 system, see the Red Hat Enterprise Linux 8 Installation Guide.

In addition to `rhel-8-for-x86_64-baseos-rpms` on a RHEL 8 system, you also need to have the `rhel-8-for-x86_64-appstream-rpms`, `rhel-8-for-x86_64-supplementary-rpms`, and `codeready-builder-for-rhel-8-x86_64-r` repositories enabled:

```
# subscription-manager repos --enable=rhel-8-for-x86_64-appstream-
↪rpms \
  --enable=rhel-8-for-x86_64-supplementary-rpms --enable=codeready-
↪builder-for-rhel-8-x86_64-rpms
```

Enable the OpenStack repository

- On CentOS, the `extras` repository provides the RPM that enables the OpenStack repository. CentOS includes the `extras` repository by default, so you can simply install the package to enable the OpenStack repository. For CentOS8, you will also need to enable the PowerTools repository.

  When installing the Ussuri release, run:

  ```
  # yum install centos-release-openstack-ussuri
  # yum config-manager --set-enabled PowerTools
  ```

  When installing the Train release, run:

  ```
  # yum install centos-release-openstack-train
  ```

  When installing the Stein release, run:

  ```
  # yum install centos-release-openstack-stein
  ```

  When installing the Rocky release, run:

  ```
  # yum install centos-release-openstack-rocky
  ```

  When installing the Queens release, run:

  ```
  # yum install centos-release-openstack-queens
  ```

  When installing the Pike release, run:

  ```
  # yum install centos-release-openstack-pike
  ```

- On RHEL, download and install the RDO repository RPM to enable the OpenStack repository.

  On RHEL 7:

```
# yum install https://rdoproject.org/repos/rdo-release.rpm
```

On RHEL 8:

```
# dnf install https://www.rdoproject.org/repos/rdo-release.el8.rpm
```

The RDO repository RPM installs the latest available OpenStack release.

Finalize the installation

1. Upgrade the packages on all nodes:

```
# yum upgrade
```

Note: If the upgrade process includes a new kernel, reboot your host to activate it.

2. Install the appropriate OpenStack client for your version.

For CentOS 7 and RHEL 7

```
# yum install python-openstackclient
```

For CentOS 8 and RHEL 8

```
# yum install python3-openstackclient
```

3. RHEL and CentOS enable *SELinux* by default. Install the `openstack-selinux` package to automatically manage security policies for OpenStack services:

```
# yum install openstack-selinux
```

5.4.3 OpenStack packages for Ubuntu

Ubuntu releases OpenStack with each Ubuntu release. Ubuntu LTS releases are provided every two years. OpenStack packages from interim releases of Ubuntu are made available to the prior Ubuntu LTS via the Ubuntu Cloud Archive.

OpenStack Queens is available directly using Ubuntu 18.04 LTS and OpenStack Mitaka is available directly using Ubuntu 16.04 LTS without having to enable an Ubuntu Cloud Archive repository.

Note: The set up of OpenStack packages described here needs to be done on all nodes: controller, compute, and Block Storage nodes.

Warning: Your hosts must contain the latest versions of base installation packages available for your distribution before proceeding further.

Note: Disable or remove any automatic update services because they can impact your OpenStack environment.

Enable the repository for Ubuntu Cloud Archive

OpenStack Stein for Ubuntu 18.04 LTS:

```
# add-apt-repository cloud-archive:stein
```

OpenStack Rocky for Ubuntu 18.04 LTS:

```
# add-apt-repository cloud-archive:rocky
```

OpenStack Queens for Ubuntu 16.04 LTS:

```
# apt install software-properties-common
# add-apt-repository cloud-archive:queens
```

Note that OpenStack Queens is included in Ubuntu 18.04 LTS without enabling the Ubuntu Cloud Archive.

OpenStack Pike for Ubuntu 16.04 LTS:

```
# apt install software-properties-common
# add-apt-repository cloud-archive:pike
```

Note: For a full list of supported Ubuntu OpenStack releases, see Ubuntu OpenStack release cycle at https://www.ubuntu.com/about/release-cycle.

Finalize the installation

1. Upgrade packages on all nodes:

   ```
   # apt update && apt dist-upgrade
   ```

   Note: If the upgrade process includes a new kernel, reboot your host to activate it.

2. Install the OpenStack client:

   OpenStack Stein for Ubuntu 18.04 LTS:

   ```
   # apt install python3-openstackclient
   ```

   Others:

   ```
   # apt install python-openstackclient
   ```

5.5 SQL database

Most OpenStack services use an SQL database to store information. The database typically runs on the controller node. The procedures in this guide use MariaDB or MySQL depending on the distribution. OpenStack services also support other SQL databases including PostgreSQL.

Note: If you see `Too many connections` or `Too many open files` error log messages on OpenStack services, verify that maximum number of connection settings are well applied to your environment. In MariaDB, you may also need to change open_files_limit configuration.

5.5.1 SQL database for SUSE

Most OpenStack services use an SQL database to store information. The database typically runs on the controller node. The procedures in this guide use MariaDB or MySQL depending on the distribution. OpenStack services also support other SQL databases including PostgreSQL.

Install and configure components

1. Install the packages:

```
# zypper install mariadb-client mariadb python-PyMySQL
```

2. Create and edit the `/etc/my.cnf.d/openstack.cnf` file and complete the following actions:

   - Create a `[mysqld]` section, and set the `bind-address` key to the management IP address of the controller node to enable access by other nodes via the management network. Set additional keys to enable useful options and the UTF-8 character set:

```
[mysqld]
bind-address = 10.0.0.11

default-storage-engine = innodb
innodb_file_per_table = on
max_connections = 4096
collation-server = utf8_general_ci
character-set-server = utf8
```

Finalize installation

1. Start the database service and configure it to start when the system boots:

```
# systemctl enable mysql.service
# systemctl start mysql.service
```

2. Secure the database service by running the `mysql_secure_installation` script. In particular, choose a suitable password for the database `root` account:

```
# mysql_secure_installation
```

5.5.2 SQL database for RHEL and CentOS

Most OpenStack services use an SQL database to store information. The database typically runs on the controller node. The procedures in this guide use MariaDB or MySQL depending on the distribution. OpenStack services also support other SQL databases including PostgreSQL.

Install and configure components

1. Install the packages:

```
# yum install mariadb mariadb-server python2-PyMySQL
```

2. Create and edit the `/etc/my.cnf.d/openstack.cnf` file (backup existing configuration files in `/etc/my.cnf.d/` if needed) and complete the following actions:

   - Create a `[mysqld]` section, and set the `bind-address` key to the management IP address of the controller node to enable access by other nodes via the management network. Set additional keys to enable useful options and the UTF-8 character set:

```
[mysqld]
bind-address = 10.0.0.11

default-storage-engine = innodb
innodb_file_per_table = on
max_connections = 4096
collation-server = utf8_general_ci
character-set-server = utf8
```

Finalize installation

1. Start the database service and configure it to start when the system boots:

```
# systemctl enable mariadb.service
# systemctl start mariadb.service
```

2. Secure the database service by running the `mysql_secure_installation` script. In particular, choose a suitable password for the database `root` account:

```
# mysql_secure_installation
```

5.5.3 SQL database for Ubuntu

Most OpenStack services use an SQL database to store information. The database typically runs on the controller node. The procedures in this guide use MariaDB or MySQL depending on the distribution. OpenStack services also support other SQL databases including PostgreSQL.

Note: As of Ubuntu 16.04, MariaDB was changed to use the unix_socket Authentication Plugin. Local authentication is now performed using the user credentials (UID), and password authentication is no longer used by default. This means that the root user no longer uses a password for local access to the server.

Note: As of Ubuntu 18.04, the `mariadb-server` package is no longer available from the default repository. To install successfully, enable the `Universe` repository on Ubuntu.

Install and configure components

1. Install the packages:

```
# apt install mariadb-server python-pymysql
```

2. Create and edit the `/etc/mysql/mariadb.conf.d/99-openstack.cnf` file and complete the following actions:

   - Create a `[mysqld]` section, and set the `bind-address` key to the management IP address of the controller node to enable access by other nodes via the management network. Set additional keys to enable useful options and the UTF-8 character set:

```
[mysqld]
bind-address = 10.0.0.11

default-storage-engine = innodb
innodb_file_per_table = on
max_connections = 4096
collation-server = utf8_general_ci
character-set-server = utf8
```

Finalize installation

1. Restart the database service:

```
# service mysql restart
```

2. Secure the database service by running the `mysql_secure_installation` script. In particular, choose a suitable password for the database `root` account:

```
# mysql_secure_installation
```

5.6 Message queue

OpenStack uses a *message queue* to coordinate operations and status information among services. The message queue service typically runs on the controller node. OpenStack supports several message queue services including RabbitMQ, Qpid, and ZeroMQ. However, most distributions that package OpenStack support a particular message queue service. This guide implements the RabbitMQ message queue service because most distributions support it. If you prefer to implement a different message queue service, consult the documentation associated with it.

The message queue runs on the controller node.

5.6.1 Message queue for SUSE

OpenStack uses a *message queue* to coordinate operations and status information among services. The message queue service typically runs on the controller node. OpenStack supports several message queue services including RabbitMQ, Qpid, and ZeroMQ. However, most distributions that package OpenStack support a particular message queue service. This guide implements the RabbitMQ message queue service because most distributions support it. If you prefer to implement a different message queue service, consult the documentation associated with it.

The message queue runs on the controller node.

Install and configure components

1. Install the package:

```
# zypper install rabbitmq-server
```

2. Start the message queue service and configure it to start when the system boots:

```
# systemctl enable rabbitmq-server.service
# systemctl start rabbitmq-server.service
```

3. Add the `openstack` user:

```
# rabbitmqctl add_user openstack RABBIT_PASS

Creating user "openstack" ...
```

   Replace `RABBIT_PASS` with a suitable password.

4. Permit configuration, write, and read access for the `openstack` user:

```
# rabbitmqctl set_permissions openstack ".*" ".*" ".*"

Setting permissions for user "openstack" in vhost "/" ...
```

5.6.2 Message queue for RHEL and CentOS

OpenStack uses a *message queue* to coordinate operations and status information among services. The message queue service typically runs on the controller node. OpenStack supports several message queue services including RabbitMQ, Qpid, and ZeroMQ. However, most distributions that package OpenStack support a particular message queue service. This guide implements the RabbitMQ message queue service because most distributions support it. If you prefer to implement a different message queue service, consult the documentation associated with it.

The message queue runs on the controller node.

Install and configure components

1. Install the package:

```
# yum install rabbitmq-server
```

2. Start the message queue service and configure it to start when the system boots:

```
# systemctl enable rabbitmq-server.service
# systemctl start rabbitmq-server.service
```

3. Add the `openstack` user:

```
# rabbitmqctl add_user openstack RABBIT_PASS

Creating user "openstack" ...
```

   Replace `RABBIT_PASS` with a suitable password.

4. Permit configuration, write, and read access for the `openstack` user:

```
# rabbitmqctl set_permissions openstack ".*" ".*" ".*"

Setting permissions for user "openstack" in vhost "/" ...
```

5.6.3 Message queue for Ubuntu

OpenStack uses a *message queue* to coordinate operations and status information among services. The message queue service typically runs on the controller node. OpenStack supports several message queue services including RabbitMQ, Qpid, and ZeroMQ. However, most distributions that package OpenStack support a particular message queue service. This guide implements the RabbitMQ message queue service because most distributions support it. If you prefer to implement a different message queue service, consult the documentation associated with it.

The message queue runs on the controller node.

Install and configure components

1. Install the package:

```
# apt install rabbitmq-server
```

2. Add the `openstack` user:

```
# rabbitmqctl add_user openstack RABBIT_PASS

Creating user "openstack" ...
```

   Replace `RABBIT_PASS` with a suitable password.

3. Permit configuration, write, and read access for the `openstack` user:

```
# rabbitmqctl set_permissions openstack ".*" ".*" ".*"

Setting permissions for user "openstack" in vhost "/" ...
```

5.7 Memcached

The Identity service authentication mechanism for services uses Memcached to cache tokens. The memcached service typically runs on the controller node. For production deployments, we recommend enabling a combination of firewalling, authentication, and encryption to secure it.

5.7.1 Memcached for SUSE

The Identity service authentication mechanism for services uses Memcached to cache tokens. The memcached service typically runs on the controller node. For production deployments, we recommend enabling a combination of firewalling, authentication, and encryption to secure it.

Install and configure components

1. Install the packages:

```
# zypper install memcached python-python-memcached
```

2. Edit the `/etc/sysconfig/memcached` file and complete the following actions:

   - Configure the service to use the management IP address of the controller node. This is to enable access by other nodes via the management network:

```
MEMCACHED_PARAMS="-l 10.0.0.11"
```

   Note: Change the existing line `MEMCACHED_PARAMS="-l 127.0.0.1"`.

Finalize installation

- Start the Memcached service and configure it to start when the system boots:

```
# systemctl enable memcached.service
# systemctl start memcached.service
```

5.7.2 Memcached for RHEL and CentOS

The Identity service authentication mechanism for services uses Memcached to cache tokens. The memcached service typically runs on the controller node. For production deployments, we recommend enabling a combination of firewalling, authentication, and encryption to secure it.

Install and configure components

1. Install the packages:

   For CentOS 7 and RHEL 7

   ```
   # yum install memcached python-memcached
   ```

   For CentOS 8 and RHEL 8

   ```
   # yum install memcached python3-memcached
   ```

2. Edit the `/etc/sysconfig/memcached` file and complete the following actions:

   - Configure the service to use the management IP address of the controller node. This is to enable access by other nodes via the management network:

     ```
     OPTIONS="-l 127.0.0.1,::1,controller"
     ```

     Note: Change the existing line `OPTIONS="-l 127.0.0.1,::1"`.

Finalize installation

- Start the Memcached service and configure it to start when the system boots:

```
# systemctl enable memcached.service
# systemctl start memcached.service
```

5.7.3 Memcached for Ubuntu

The Identity service authentication mechanism for services uses Memcached to cache tokens. The memcached service typically runs on the controller node. For production deployments, we recommend enabling a combination of firewalling, authentication, and encryption to secure it.

Install and configure components

1. Install the packages:

```
# apt install memcached python-memcache
```

2. Edit the `/etc/memcached.conf` file and configure the service to use the management IP address of the controller node. This is to enable access by other nodes via the management network:

```
-l 10.0.0.11
```

Note: Change the existing line that had `-l 127.0.0.1`.

Finalize installation

- Restart the Memcached service:

```
# service memcached restart
```

5.8 Etcd

OpenStack services may use Etcd, a distributed reliable key-value store for distributed key locking, storing configuration, keeping track of service live-ness and other scenarios.

5.8.1 Etcd for SUSE

Right now, there is no distro package available for etcd3. This guide uses the tarball installation as a workaround until proper distro packages are available.

The etcd service runs on the controller node.

Install and configure components

1. Install etcd:

   - Create etcd user:

```
# groupadd --system etcd
# useradd --home-dir "/var/lib/etcd" \
      --system \
      --shell /bin/false \
```

(continues on next page)

(continued from previous page)

```
      -g etcd \
      etcd
```

- Create the necessary directories:

```
# mkdir -p /etc/etcd
# chown etcd:etcd /etc/etcd
# mkdir -p /var/lib/etcd
# chown etcd:etcd /var/lib/etcd
```

- Determine your system architecture:

```
# uname -m
```

- Download and install the etcd tarball for x86_64/amd64:

```
# ETCD_VER=v3.2.7
# rm -rf /tmp/etcd && mkdir -p /tmp/etcd
# curl -L \
      https://github.com/coreos/etcd/releases/download/${ETCD_VER}
↪/etcd-${ETCD_VER}-linux-amd64.tar.gz \
      -o /tmp/etcd-${ETCD_VER}-linux-amd64.tar.gz
# tar xzvf /tmp/etcd-${ETCD_VER}-linux-amd64.tar.gz \
      -C /tmp/etcd --strip-components=1
# cp /tmp/etcd/etcd /usr/bin/etcd
# cp /tmp/etcd/etcdctl /usr/bin/etcdctl
```

Or download and install the etcd tarball for arm64:

```
# ETCD_VER=v3.2.7
# rm -rf /tmp/etcd && mkdir -p /tmp/etcd
# curl -L \
      https://github.com/coreos/etcd/releases/download/${ETCD_VER}
↪/etcd-${ETCD_VER}-linux-arm64.tar.gz \
      -o /tmp/etcd-${ETCD_VER}-linux-arm64.tar.gz
# tar xzvf /tmp/etcd-${ETCD_VER}-linux-arm64.tar.gz \
      -C /tmp/etcd --strip-components=1
# cp /tmp/etcd/etcd /usr/bin/etcd
# cp /tmp/etcd/etcdctl /usr/bin/etcdctl
```

2. Create and edit the `/etc/etcd/etcd.conf.yml` file and set the `initial-cluster`, `initial-advertise-peer-urls`, `advertise-client-urls`, `listen-client-urls` to the management IP address of the controller node to enable access by other nodes via the management network:

```
name: controller
data-dir: /var/lib/etcd
initial-cluster-state: 'new'
initial-cluster-token: 'etcd-cluster-01'
initial-cluster: controller=http://10.0.0.11:2380
initial-advertise-peer-urls: http://10.0.0.11:2380
advertise-client-urls: http://10.0.0.11:2379
listen-peer-urls: http://0.0.0.0:2380
listen-client-urls: http://10.0.0.11:2379
```

3. Create and edit the `/usr/lib/systemd/system/etcd.service` file:

```
[Unit]
After=network.target
Description=etcd - highly-available key value store

[Service]
# Uncomment this on ARM64.
# Environment="ETCD_UNSUPPORTED_ARCH=arm64"
LimitNOFILE=65536
Restart=on-failure
Type=notify
ExecStart=/usr/bin/etcd --config-file /etc/etcd/etcd.conf.yml
User=etcd

[Install]
WantedBy=multi-user.target
```

Reload systemd service files with:

```
# systemctl daemon-reload
```

Finalize installation

1. Enable and start the etcd service:

```
# systemctl enable etcd
# systemctl start etcd
```

5.8.2 Etcd for RHEL and CentOS

OpenStack services may use Etcd, a distributed reliable key-value store for distributed key locking, storing configuration, keeping track of service live-ness and other scenarios.

The etcd service runs on the controller node.

Install and configure components

1. Install the package:

```
# yum install etcd
```

2. Edit the `/etc/etcd/etcd.conf` file and set the `ETCD_INITIAL_CLUSTER`, `ETCD_INITIAL_ADVERTISE_PEER_URLS`, `ETCD_ADVERTISE_CLIENT_URLS`, `ETCD_LISTEN_CLIENT_URLS` to the management IP address of the controller node to enable access by other nodes via the management network:

```
#[Member]
ETCD_DATA_DIR="/var/lib/etcd/default.etcd"
ETCD_LISTEN_PEER_URLS="http://10.0.0.11:2380"
ETCD_LISTEN_CLIENT_URLS="http://10.0.0.11:2379"
ETCD_NAME="controller"
#[Clustering]
```

(continues on next page)

(continued from previous page)

```
ETCD_INITIAL_ADVERTISE_PEER_URLS="http://10.0.0.11:2380"
ETCD_ADVERTISE_CLIENT_URLS="http://10.0.0.11:2379"
ETCD_INITIAL_CLUSTER="controller=http://10.0.0.11:2380"
ETCD_INITIAL_CLUSTER_TOKEN="etcd-cluster-01"
ETCD_INITIAL_CLUSTER_STATE="new"
```

Finalize installation

1. Enable and start the etcd service:

```
# systemctl enable etcd
# systemctl start etcd
```

5.8.3 Etcd for Ubuntu

OpenStack services may use Etcd, a distributed reliable key-value store for distributed key locking, storing configuration, keeping track of service live-ness and other scenarios.

The etcd service runs on the controller node.

Install and configure components

1. Install the `etcd` package:

```
# apt install etcd
```

Note: As of Ubuntu 18.04, the `etcd` package is no longer available from the default repository. To install successfully, enable the `Universe` repository on Ubuntu.

---

2. Edit the `/etc/default/etcd` file and set the `ETCD_INITIAL_CLUSTER`, `ETCD_INITIAL_ADVERTISE_PEER_URLS`, `ETCD_ADVERTISE_CLIENT_URLS`, `ETCD_LISTEN_CLIENT_URLS` to the management IP address of the controller node to enable access by other nodes via the management network:

```
ETCD_NAME="controller"
ETCD_DATA_DIR="/var/lib/etcd"
ETCD_INITIAL_CLUSTER_STATE="new"
ETCD_INITIAL_CLUSTER_TOKEN="etcd-cluster-01"
ETCD_INITIAL_CLUSTER="controller=http://10.0.0.11:2380"
ETCD_INITIAL_ADVERTISE_PEER_URLS="http://10.0.0.11:2380"
ETCD_ADVERTISE_CLIENT_URLS="http://10.0.0.11:2379"
ETCD_LISTEN_PEER_URLS="http://0.0.0.0:2380"
ETCD_LISTEN_CLIENT_URLS="http://10.0.0.11:2379"
```

Finalize installation

1. Enable and restart the etcd service:

```
# systemctl enable etcd
# systemctl restart etcd
```

CHAPTER

SIX

INSTALL OPENSTACK SERVICES

The installation of individual OpenStack services is covered in the Project Installation Guides that are available at the following locations:

- OpenStack Installation Guides for Ussuri
- OpenStack Installation Guides for Train
- OpenStack Installation Guides for Stein
- OpenStack Installation Guides for Rocky
- OpenStack Installation Guides for Queens
- OpenStack Installation Guides for Pike

6.1 Minimal deployment for Ussuri

At a minimum, you need to install the following services. Install the services in the order specified below:

- Identity service keystone installation for Ussuri
- Image service glance installation for Ussuri
- Placement service placement installation for Ussuri
- Compute service nova installation for Ussuri
- Networking service neutron installation for Ussuri

We advise to also install the following components after you have installed the minimal deployment services:

- Dashboard horizon installation for Ussuri
- Block Storage service cinder installation for Ussuri

6.2 Minimal deployment for Train

At a minimum, you need to install the following services. Install the services in the order specified below:

- Identity service keystone installation for Train
- Image service glance installation for Train
- Placement service placement installation for Train
- Compute service nova installation for Train
- Networking service neutron installation for Train

We advise to also install the following components after you have installed the minimal deployment services:

- Dashboard horizon installation for Train
- Block Storage service cinder installation for Train

6.3 Minimal deployment for Stein

At a minimum, you need to install the following services. Install the services in the order specified below:

- Identity service keystone installation for Stein
- Image service glance installation for Stein
- Placement service placement installation for Stein
- Compute service nova installation for Stein
- Networking service neutron installation for Stein

We advise to also install the following components after you have installed the minimal deployment services:

- Dashboard horizon installation for Stein
- Block Storage service cinder installation for Stein

6.4 Minimal deployment for Rocky

At a minimum, you need to install the following services. Install the services in the order specified below:

- Identity service keystone installation for Rocky
- Image service glance installation for Rocky
- Compute service nova installation for Rocky
- Networking service neutron installation for Rocky

We advise to also install the following components after you have installed the minimal deployment services:

- Dashboard horizon installation for Rocky
- Block Storage service cinder installation for Rocky

6.5 Minimal deployment for Queens

At a minimum, you need to install the following services. Install the services in the order specified below:

- Identity service keystone installation for Queens
- Image service glance installation for Queens
- Compute service nova installation for Queens
- Networking service neutron installation for Queens

We advise to also install the following components after you have installed the minimal deployment services:

- Dashboard horizon installation for Queens
- Block Storage service cinder installation for Queens

6.6 Minimal deployment for Pike

At a minimum, you need to install the following services. Install the services in the order specified below:

- Identity service keystone installation for Pike
- Image service glance installation for Pike
- Compute service nova installation for Pike
- Networking service neutron installation for Pike

We advise to also install the following components after you have installed the minimal deployment services:

- Dashboard horizon installation for Pike
- Block Storage service cinder installation for Pike

CHAPTER

SEVEN

LAUNCH AN INSTANCE

This section creates the necessary virtual networks to support launching instances. Networking option 1 includes one provider (external) network with one instance that uses it. Networking option 2 includes one provider network with one instance that uses it and one self-service (private) network with one instance that uses it.

The instructions in this section use command-line interface (CLI) tools on the controller node. However, you can follow the instructions on any host that the tools are installed.

For more information on the CLI tools, see the OpenStackClient documentation for Pike, the OpenStackClient documentation for Queens, or the OpenStackClient documentation for Rocky.

To use the dashboard, see the Dashboard User Documentation for Pike, the Dashboard User Documentation for Queens, or the Dashboard User Documentation for Rocky.

7.1 Create virtual networks

Create virtual networks for the networking option that you chose when configuring Neutron. If you chose option 1, create only the provider network. If you chose option 2, create the provider and self-service networks.

7.1.1 Provider network

Before launching an instance, you must create the necessary virtual network infrastructure. For networking option 1, an instance uses a provider (external) network that connects to the physical network infrastructure via layer-2 (bridging/switching). This network includes a DHCP server that provides IP addresses to instances.

The `admin` or other privileged user must create this network because it connects directly to the physical network infrastructure.

Note: The following instructions and diagrams use example IP address ranges. You must adjust them for your particular environment.

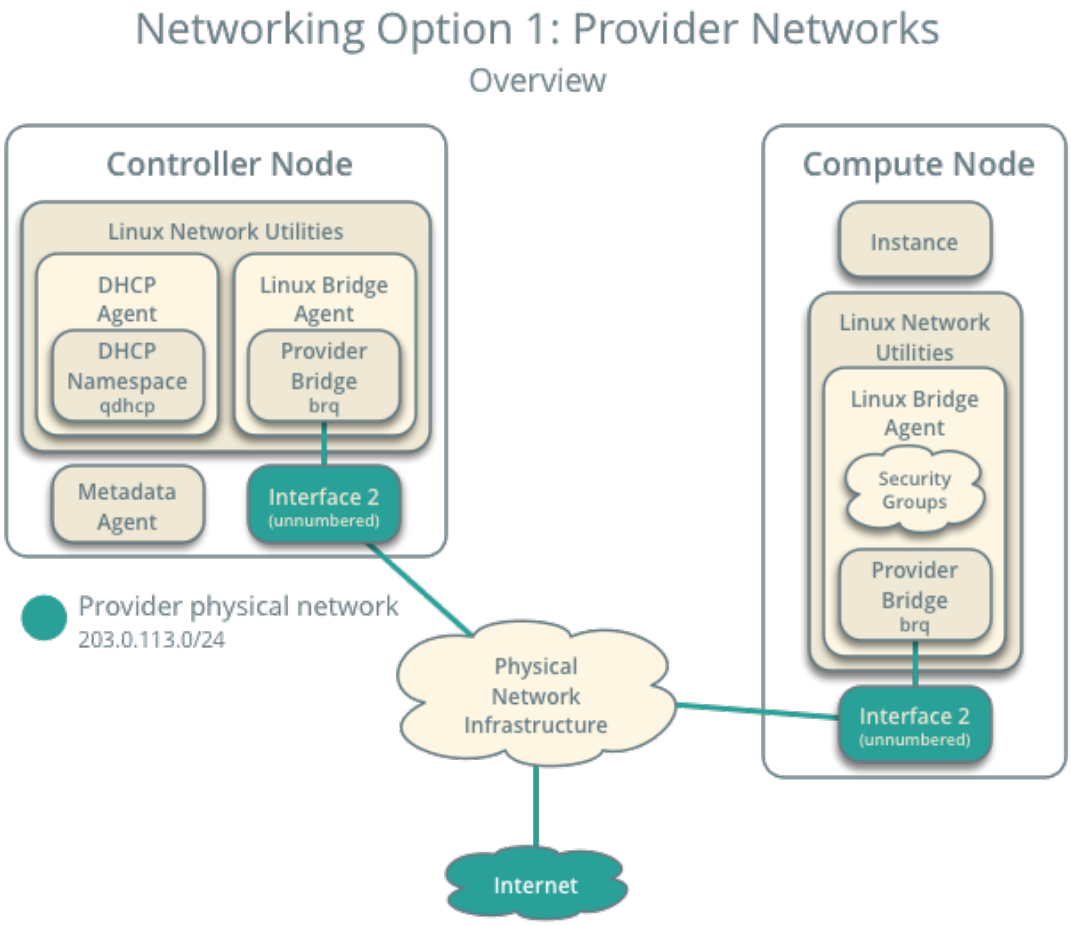


Fig. 1: **Networking Option 1: Provider networks - Overview**

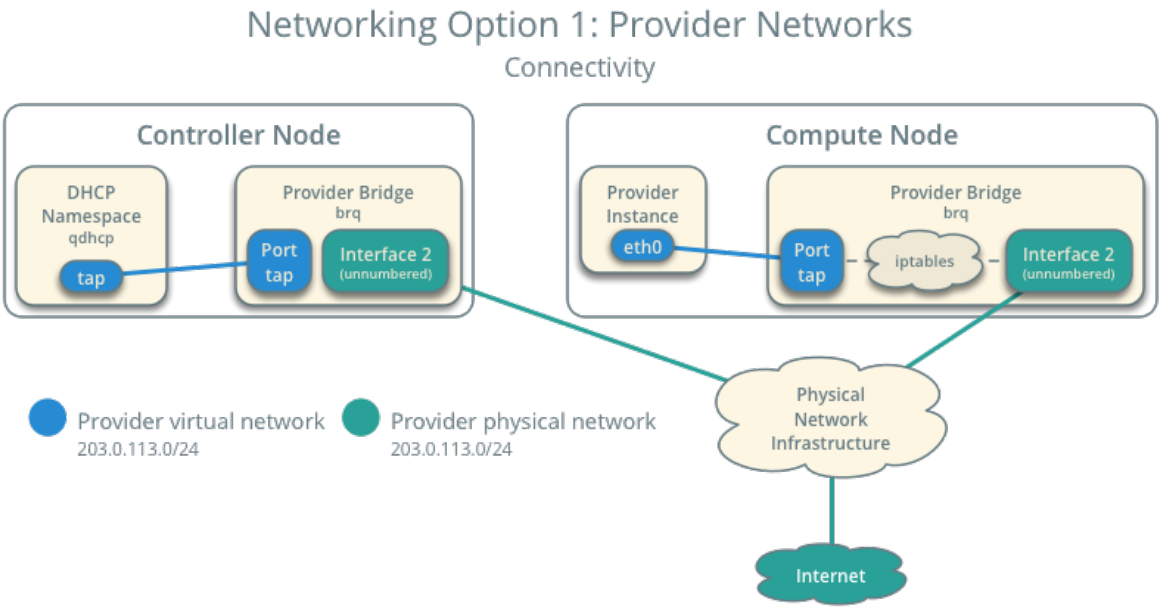


Fig. 2: **Networking Option 1: Provider networks - Connectivity**

Create the provider network

1. On the controller node, source the `admin` credentials to gain access to admin-only CLI commands:

```
$ . admin-openrc
```

2. Create the network:

```
$ openstack network create  --share --external \
  --provider-physical-network provider \
  --provider-network-type flat provider

Created a new network:

+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | UP                                   |
| availability_zone_hints   |                                      |
| availability_zones        |                                      |
| created_at                | 2017-03-14T14:37:39Z                 |
| description               |                                      |
| dns_domain                | None                                 |
| id                        | 54adb94a-4dce-437f-a33b-e7e2e7648173 |
| ipv4_address_scope        | None                                 |
| ipv6_address_scope        | None                                 |
| is_default                | None                                 |
| mtu                       | 1500                                 |
| name                      | provider                             |
| port_security_enabled     | True                                 |
| project_id                | 4c7f48f1da5b494faaa66713686a7707     |
| provider:network_type     | flat                                 |
| provider:physical_network | provider                             |
| provider:segmentation_id  | None                                 |
| qos_policy_id             | None                                 |
| revision_number           | 3                                    |
| router:external           | External                             |
| segments                  | None                                 |
| shared                    | True                                 |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| updated_at                | 2017-03-14T14:37:39Z                 |
+---------------------------+--------------------------------------+
```

The `--share` option allows all projects to use the virtual network.

The `--external` option defines the virtual network to be external. If you wish to create an internal network, you can use `--internal` instead. Default value is `internal`.

The `--provider-physical-network provider` and `--provider-network-type flat` options connect the flat virtual network to the flat (native/untagged) physical network on the `eth1` interface on the host using information from the following files:

`ml2_conf.ini`:

```
[ml2_type_flat]
flat_networks = provider
```

linuxbridge_agent.ini:

```
[linux_bridge]
physical_interface_mappings = provider:eth1
```

3. Create a subnet on the network:

```
$ openstack subnet create --network provider \
  --allocation-pool start=START_IP_ADDRESS,end=END_IP_ADDRESS \
  --dns-nameserver DNS_RESOLVER --gateway PROVIDER_NETWORK_GATEWAY \
  --subnet-range PROVIDER_NETWORK_CIDR provider
```

Replace PROVIDER_NETWORK_CIDR with the subnet on the provider physical network in CIDR notation.

Replace START_IP_ADDRESS and END_IP_ADDRESS with the first and last IP address of the range within the subnet that you want to allocate for instances. This range must not include any existing active IP addresses.

Replace DNS_RESOLVER with the IP address of a DNS resolver. In most cases, you can use one from the /etc/resolv.conf file on the host.

Replace PROVIDER_NETWORK_GATEWAY with the gateway IP address on the provider network, typically the .1 IP address.

Example

The provider network uses 203.0.113.0/24 with a gateway on 203.0.113.1. A DHCP server assigns each instance an IP address from 203.0.113.101 to 203.0.113.250. All instances use 8.8.4.4 as a DNS resolver.

```
$ openstack subnet create --network provider \
  --allocation-pool start=203.0.113.101,end=203.0.113.250 \
  --dns-nameserver 8.8.4.4 --gateway 203.0.113.1 \
  --subnet-range 203.0.113.0/24 provider

Created a new subnet:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| allocation_pools  | 203.0.113.101-203.0.113.250          |
| cidr              | 203.0.113.0/24                       |
| created_at        | 2017-03-29T05:48:29Z                 |
| description       |                                      |
| dns_nameservers   | 8.8.4.4                              |
| enable_dhcp       | True                                 |
| gateway_ip        | 203.0.113.1                          |
| host_routes       |                                      |
| id                | e84b4972-c7fc-4ce9-9742-fdc845196ac5 |
| ip_version        | 4                                    |
| ipv6_address_mode | None                                 |
| ipv6_ra_mode      | None                                 |
| name              | provider                             |
| network_id        | 1f816a46-7c3f-4ccf-8bf3-fe0807ddff8d |
```

(continues on next page)

(continued from previous page)

```
| project_id          | 496efd248b0c46d3b80de60a309177b5      |
| revision_number     | 2                                     |
| segment_id          | None                                  |
| service_types       |                                       |
| subnetpool_id       | None                                  |
| updated_at          | 2017-03-29T05:48:29Z                  |
+---------------------+---------------------------------------+
```

Return to *Launch an instance - Create virtual networks*.

7.1.2 Self-service network

If you chose networking option 2, you can also create a self-service (private) network that connects to the physical network infrastructure via NAT. This network includes a DHCP server that provides IP addresses to instances. An instance on this network can automatically access external networks such as the Internet. However, access to an instance on this network from external networks such as the Internet requires a *floating IP address*.

The `demo` or other unprivileged user can create this network because it provides connectivity to instances within the `demo` project only.

Warning: You must *create the provider network* before the self-service network.

Note: The following instructions and diagrams use example IP address ranges. You must adjust them for your particular environment.

Create the self-service network

1. On the controller node, source the `demo` credentials to gain access to user-only CLI commands:

```
$ . demo-openrc
```

2. Create the network:

```
$ openstack network create selfservice

Created a new network:
+-------------------------+--------------------------------------+
| Field                   | Value                                |
+-------------------------+--------------------------------------+
| admin_state_up          | UP                                   |
| availability_zone_hints |                                      |
| availability_zones      |                                      |
| created_at              | 2016-11-04T18:20:59Z                 |
| description             |                                      |
| headers                 |                                      |
| id                      | 7c6f9b37-76b4-463e-98d8-27e5686ed083 |
| ipv4_address_scope      | None                                 |
```

(continues on next page)

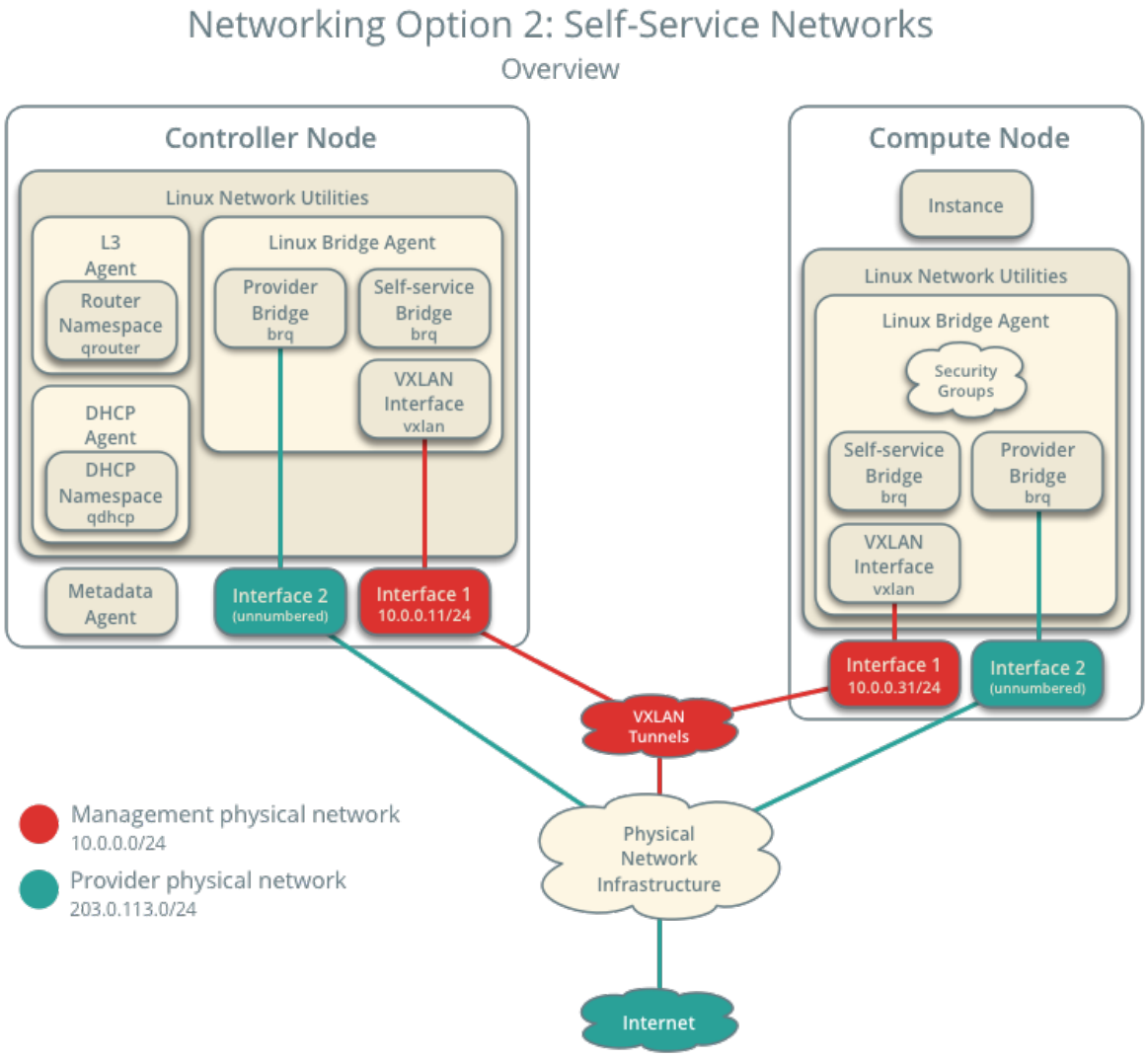


Fig. 3: **Networking Option 2: Self-service networks - Overview**

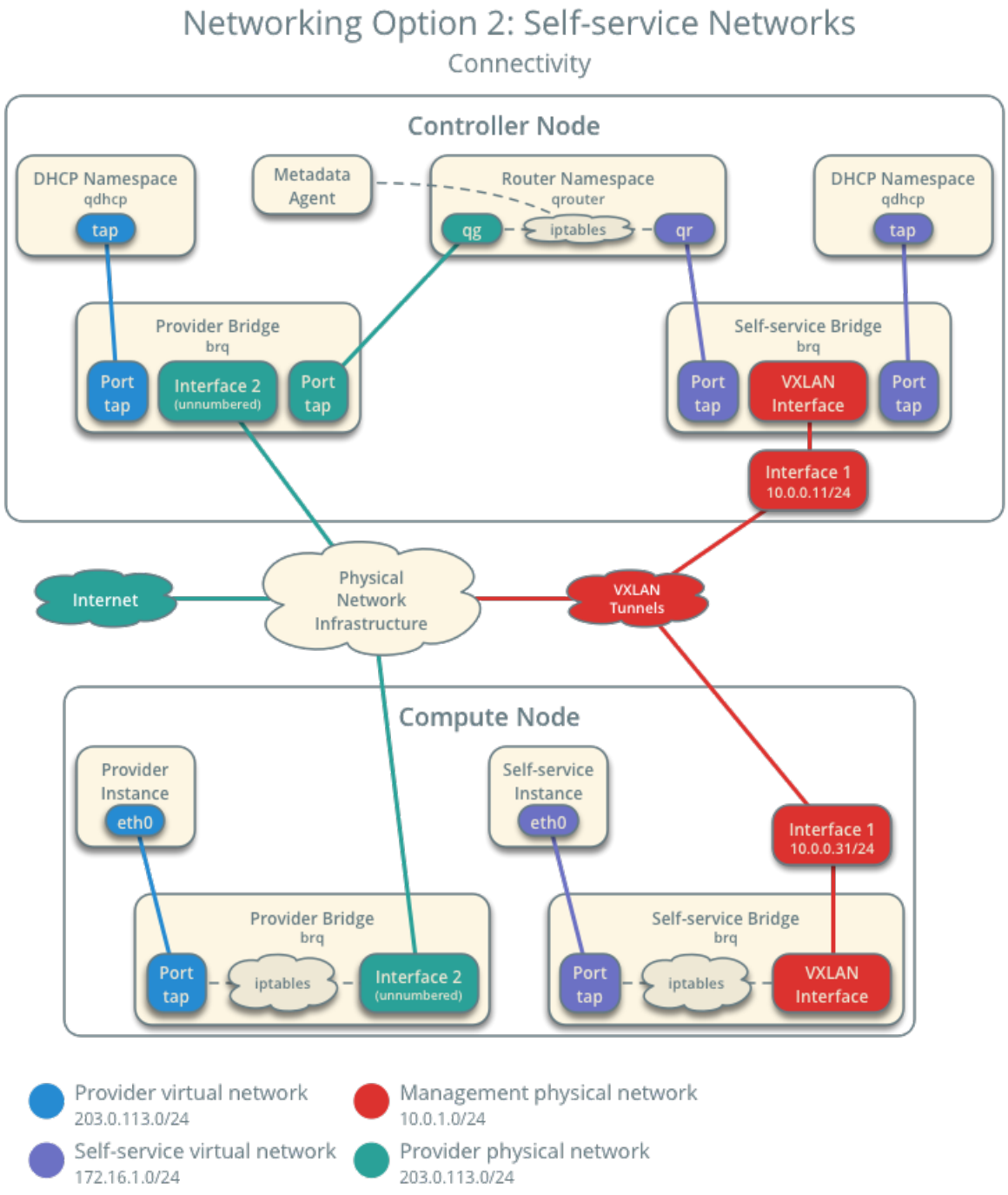


Fig. 4: **Networking Option 2: Self-service networks - Connectivity**

(continued from previous page)

```
| ipv6_address_scope        | None                                 |
| mtu                       | 1450                                 |
| name                      | selfservice                          |
| port_security_enabled     | True                                 |
| project_id                | 3828e7c22c5546e585f27b9eb5453788     |
| project_id                | 3828e7c22c5546e585f27b9eb5453788     |
| revision_number           | 3                                    |
| router:external           | Internal                             |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tags                      | []                                   |
| updated_at                | 2016-11-04T18:20:59Z                 |
+---------------------------+--------------------------------------+
```

Non-privileged users typically cannot supply additional parameters to this command. The service automatically chooses parameters using information from the following files:

`ml2_conf.ini`:

```
[ml2]
tenant_network_types = vxlan

[ml2_type_vxlan]
vni_ranges = 1:1000
```

3. Create a subnet on the network:

```
$ openstack subnet create --network selfservice \
  --dns-nameserver DNS_RESOLVER --gateway SELFSERVICE_NETWORK_GATEWAY␣
↪\
  --subnet-range SELFSERVICE_NETWORK_CIDR selfservice
```

Replace `DNS_RESOLVER` with the IP address of a DNS resolver. In most cases, you can use one from the `/etc/resolv.conf` file on the host.

Replace `SELFSERVICE_NETWORK_GATEWAY` with the gateway you want to use on the self-service network, typically the .1 IP address.

Replace `SELFSERVICE_NETWORK_CIDR` with the subnet you want to use on the self-service network. You can use any arbitrary value, although we recommend a network from RFC 1918.

Example

The self-service network uses 172.16.1.0/24 with a gateway on 172.16.1.1. A DHCP server assigns each instance an IP address from 172.16.1.2 to 172.16.1.254. All instances use 8.8.4.4 as a DNS resolver.

```
$ openstack subnet create --network selfservice \
  --dns-nameserver 8.8.4.4 --gateway 172.16.1.1 \
  --subnet-range 172.16.1.0/24 selfservice

Created a new subnet:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
```

(continues on next page)

(continued from previous page)

```
| allocation_pools  | 172.16.1.2-172.16.1.254                |
| cidr              | 172.16.1.0/24                          |
| created_at        | 2016-11-04T18:30:54Z                   |
| description       |                                        |
| dns_nameservers   | 8.8.4.4                                |
| enable_dhcp       | True                                   |
| gateway_ip        | 172.16.1.1                             |
| headers           |                                        |
| host_routes       |                                        |
| id                | 5c37348e-e7da-439b-8c23-2af47d93aee5   |
| ip_version        | 4                                      |
| ipv6_address_mode | None                                   |
| ipv6_ra_mode      | None                                   |
| name              | selfservice                            |
| network_id        | b9273876-5946-4f02-a4da-838224a144e7   |
| project_id        | 3828e7c22c5546e585f27b9eb5453788       |
| project_id        | 3828e7c22c5546e585f27b9eb5453788       |
| revision_number   | 2                                      |
| service_types     | []                                     |
| subnetpool_id     | None                                   |
| updated_at        | 2016-11-04T18:30:54Z                   |
+-------------------+----------------------------------------+
```

Create a router

Self-service networks connect to provider networks using a virtual router that typically performs bidirectional NAT. Each router contains an interface on at least one self-service network and a gateway on a provider network.

The provider network must include the `router:external` option to enable self-service routers to use it for connectivity to external networks such as the Internet. The `admin` or other privileged user must include this option during network creation or add it later. In this case, the `router:external` option was set by using the `--external` parameter when creating the `provider` network.

1. On the controller node, source the `demo` credentials to gain access to user-only CLI commands:

```
$ . demo-openrc
```

2. Create the router:

```
$ openstack router create router

Created a new router:
+-------------------------+----------------------------------------+
| Field                   | Value                                  |
+-------------------------+----------------------------------------+
| admin_state_up          | UP                                     |
| availability_zone_hints |                                        |
| availability_zones      |                                        |
| created_at              | 2016-11-04T18:32:56Z                   |
| description             |                                        |
| external_gateway_info   | null                                   |
| flavor_id               | None                                   |
| headers                 |                                        |
```

(continues on next page)

(continued from previous page)

```
| id                        | 67324374-396a-4db6-9443-c70be167a42b |
| name                      | router                               |
| project_id                | 3828e7c22c5546e585f27b9eb5453788     |
| project_id                | 3828e7c22c5546e585f27b9eb5453788     |
| revision_number           | 2                                    |
| routes                    |                                      |
| status                    | ACTIVE                               |
| updated_at                | 2016-11-04T18:32:56Z                 |
+---------------------------+--------------------------------------+
```

3. Add the self-service network subnet as an interface on the router:

```
$ openstack router add subnet router selfservice
```

4. Set a gateway on the provider network on the router:

```
$ openstack router set router --external-gateway provider
```

Verify operation

We recommend that you verify operation and fix any issues before proceeding. The following steps use the IP address ranges from the network and subnet creation examples.

1. On the controller node, source the `admin` credentials to gain access to admin-only CLI commands:

```
$ . admin-openrc
```

2. List network namespaces. You should see one `qrouter` namespace and two `qdhcp` namespaces.

```
$ ip netns

qrouter-89dd2083-a160-4d75-ab3a-14239f01ea0b
qdhcp-7c6f9b37-76b4-463e-98d8-27e5686ed083
qdhcp-0e62efcd-8cee-46c7-b163-d8df05c3c5ad
```

3. List ports on the router to determine the gateway IP address on the provider network:

```
$ openstack port list --router router

+--------------------------------------+------+-------------------+---
↪-------------------------------------------------------------------
↪--------+--------+
| ID                                   | Name | MAC Address       |␣
↪Fixed IP Addresses                                                  ␣
↪          | Status |
+--------------------------------------+------+-------------------+---
↪-------------------------------------------------------------------
↪--------+--------+
| bff6605d-824c-41f9-b744-21d128fc86e1 |      | fa:16:3e:2f:34:9b |␣
↪ip_address='172.16.1.1', subnet_id='3482f524-8bff-4871-80d4-
↪5774c2730728'      | ACTIVE |
| d6fe98db-ae01-42b0-a860-37b1661f5950 |      | fa:16:3e:e8:c1:41 |␣
↪ip_address='203.0.113.102', subnet_id='5cc70da8-4ee7-4565-be53-
↪b9c011fca011'   | ACTIVE |
```

(continues on next page)

(continued from previous page)

```
+--------------------------------------+------+------------------+---
↪-------------------------------------------------------------------
↪--------+---------+
```

4. Ping this IP address from the controller node or any host on the physical provider network:

```
$ ping -c 4 203.0.113.102

PING 203.0.113.102 (203.0.113.102) 56(84) bytes of data.
64 bytes from 203.0.113.102: icmp_req=1 ttl=64 time=0.619 ms
64 bytes from 203.0.113.102: icmp_req=2 ttl=64 time=0.189 ms
64 bytes from 203.0.113.102: icmp_req=3 ttl=64 time=0.165 ms
64 bytes from 203.0.113.102: icmp_req=4 ttl=64 time=0.216 ms

--- 203.0.113.102 ping statistics ---
rtt min/avg/max/mdev = 0.165/0.297/0.619/0.187 ms
```

Return to *Launch an instance - Create virtual networks*.

After creating the appropriate networks for your environment, you can continue preparing the environment to launch an instance.

7.2 Create m1.nano flavor

The smallest default flavor consumes 512 MB memory per instance. For environments with compute nodes containing less than 4 GB memory, we recommend creating the `m1.nano` flavor that only requires 64 MB per instance. Only use this flavor with the CirrOS image for testing purposes.

```
$ openstack flavor create --id 0 --vcpus 1 --ram 64 --disk 1 m1.nano

+----------------------------+---------+
| Field                      | Value   |
+----------------------------+---------+
| OS-FLV-DISABLED:disabled   | False   |
| OS-FLV-EXT-DATA:ephemeral  | 0       |
| disk                       | 1       |
| id                         | 0       |
| name                       | m1.nano |
| os-flavor-access:is_public | True    |
| properties                 |         |
| ram                        | 64      |
| rxtx_factor                | 1.0     |
| swap                       |         |
| vcpus                      | 1       |
+----------------------------+---------+
```

7.3 Generate a key pair

Most cloud images support *public key authentication* rather than conventional password authentication. Before launching an instance, you must add a public key to the Compute service.

1. Source the `demo` project credentials:

```
$ . demo-openrc
```

2. Generate a key pair and add a public key:

```
$ ssh-keygen -q -N ""
$ openstack keypair create --public-key ~/.ssh/id_rsa.pub mykey

+-------------+-------------------------------------------------+
| Field       | Value                                           |
+-------------+-------------------------------------------------+
| fingerprint | ee:3d:2e:97:d4:e2:6a:54:6d:0d:ce:43:39:2c:ba:4d |
| name        | mykey                                           |
| user_id     | 58126687cbcc4888bfa9ab73a2256f27                |
+-------------+-------------------------------------------------+
```

Note: Alternatively, you can skip the `ssh-keygen` command and use an existing public key.

3. Verify addition of the key pair:

```
$ openstack keypair list

+-------+-------------------------------------------------+
| Name  | Fingerprint                                     |
+-------+-------------------------------------------------+
| mykey | ee:3d:2e:97:d4:e2:6a:54:6d:0d:ce:43:39:2c:ba:4d |
+-------+-------------------------------------------------+
```

7.4 Add security group rules

By default, the `default` security group applies to all instances and includes firewall rules that deny remote access to instances. For Linux images such as CirrOS, we recommend allowing at least ICMP (ping) and secure shell (SSH).

- Add rules to the `default` security group:
  - Permit *ICMP* (ping):

```
$ openstack security group rule create --proto icmp default

+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| created_at        | 2017-03-30T00:46:43Z                 |
| description       |                                      |
| direction         | ingress                              |
```

(continues on next page)

(continued from previous page)

```
| ether_type        | IPv4                                 |
| id                | 1946be19-54ab-4056-90fb-4ba606f19e66 |
| name              | None                                 |
| port_range_max    | None                                 |
| port_range_min    | None                                 |
| project_id        | 3f714c72aed7442681cbfa895f4a68d3     |
| protocol          | icmp                                 |
| remote_group_id   | None                                 |
| remote_ip_prefix  | 0.0.0.0/0                            |
| revision_number   | 1                                    |
| security_group_id | 89ff5c84-e3d1-46bb-b149-e621689f0696 |
| updated_at        | 2017-03-30T00:46:43Z                 |
+-------------------+--------------------------------------+
```

- Permit secure shell (SSH) access:

```
$ openstack security group rule create --proto tcp --dst-port 22␣
↪default

+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| created_at        | 2017-03-30T00:43:35Z                 |
| description       |                                      |
| direction         | ingress                              |
| ether_type        | IPv4                                 |
| id                | 42bc2388-ae1a-4208-919b-10cf0f92bc1c |
| name              | None                                 |
| port_range_max    | 22                                   |
| port_range_min    | 22                                   |
| project_id        | 3f714c72aed7442681cbfa895f4a68d3     |
| protocol          | tcp                                  |
| remote_group_id   | None                                 |
| remote_ip_prefix  | 0.0.0.0/0                            |
| revision_number   | 1                                    |
| security_group_id | 89ff5c84-e3d1-46bb-b149-e621689f0696 |
| updated_at        | 2017-03-30T00:43:35Z                 |
+-------------------+--------------------------------------+
```

7.5 Launch an instance

If you chose networking option 1, you can only launch an instance on the provider network. If you chose networking option 2, you can launch an instance on the provider network and the self-service network.

7.5.1 Launch an instance on the provider network

Determine instance options

To launch an instance, you must at least specify the flavor, image name, network, security group, key, and instance name.

1. On the controller node, source the `demo` credentials to gain access to user-only CLI commands:

```
$ . demo-openrc
```

2. A flavor specifies a virtual resource allocation profile which includes processor, memory, and storage.

   List available flavors:

```
$ openstack flavor list

+----+---------+-----+------+-----------+-------+-----------+
| ID | Name    | RAM | Disk | Ephemeral | VCPUs | Is Public |
+----+---------+-----+------+-----------+-------+-----------+
| 0  | m1.nano |  64 |    1 |         0 |     1 | True      |
+----+---------+-----+------+-----------+-------+-----------+
```

   Note: You can also reference a flavor by ID.

3. List available images:

```
$ openstack image list

+--------------------------------------+--------+--------+
| ID                                   | Name   | Status |
+--------------------------------------+--------+--------+
| 390eb5f7-8d49-41ec-95b7-68c0d5d54b34 | cirros | active |
+--------------------------------------+--------+--------+
```

   This instance uses the `cirros` image.

4. List available networks:

```
$ openstack network list

+--------------------------------------+-------------+--------------
↪------------------------+
| ID                                   | Name        | Subnets      
↪                        |
+--------------------------------------+-------------+--------------
↪------------------------+
| 4716ddfe-6e60-40e7-b2a8-42e57bf3c31c | selfservice | 2112d5eb-f9d6-
↪45fd-906e-7cabd38b7c7c |
| b5b6993c-ddf9-40e7-91d0-86806a42edb8 | provider    | 310911f6-acf0-
↪4a47-824e-3032916582ff |
+--------------------------------------+-------------+--------------
↪------------------------+
```

This instance uses the `provider` provider network. However, you must reference this network using the ID instead of the name.

Note: If you chose option 2, the output should also contain the `selfservice` self-service network.

5. List available security groups:

```
$ openstack security group list

+--------------------------------------+---------+----------------------
↪----+--------------------------------------+
| ID                                   | Name    | Description
↪    | Project                              |
+--------------------------------------+---------+----------------------
↪----+--------------------------------------+
| dd2b614c-3dad-48ed-958b-b155a3b38515 | default | Default security
↪group | a516b957032844328896baa01e0f906c |
+--------------------------------------+---------+----------------------
↪----+--------------------------------------+
```

This instance uses the `default` security group.

Launch the instance

1. Launch the instance:

   Replace `PROVIDER_NET_ID` with the ID of the `provider` provider network.

Note: If you chose option 1 and your environment contains only one network, you can omit the `--nic` option because OpenStack automatically chooses the only network available.

```
$ openstack server create --flavor m1.nano --image cirros \
  --nic net-id=PROVIDER_NET_ID --security-group default \
  --key-name mykey provider-instance

+-----------------------------+------------------------------------------
↪--------+
| Field                       | Value
↪        |
+-----------------------------+------------------------------------------
↪--------+
| OS-DCF:diskConfig           | MANUAL
↪        |
| OS-EXT-AZ:availability_zone |
↪        |
| OS-EXT-STS:power_state      | NOSTATE
↪        |
| OS-EXT-STS:task_state       | scheduling
↪        |
| OS-EXT-STS:vm_state         | building
↪        |
```

(continues on next page)

(continued from previous page)

```
| OS-SRV-USG:launched_at       | None                                 ␣
↪        |
| OS-SRV-USG:terminated_at     | None                                 ␣
↪        |
| accessIPv4                   |                                      ␣
↪        |
| accessIPv6                   |                                      ␣
↪        |
| addresses                    |                                      ␣
↪        |
| adminPass                    | PwkfyQ42K72h                         ␣
↪        |
| config_drive                 |                                      ␣
↪        |
| created                      | 2017-03-30T00:59:44Z                 ␣
↪        |
| flavor                       | m1.nano (0)                          ␣
↪        |
| hostId                       |                                      ␣
↪        |
| id                           | 36f3130e-cf1b-42f8-a80b-ebd63968940e ␣
↪        |
| image                        | cirros (97e06b44-e9ed-4db4-ba67-
↪6e9fc5d0a203) |
| key_name                     | mykey                                ␣
↪        |
| name                         | provider-instance                    ␣
↪        |
| progress                     | 0                                    ␣
↪        |
| project_id                   | 3f714c72aed7442681cbfa895f4a68d3     ␣
↪        |
| properties                   |                                      ␣
↪        |
| security_groups              | name='default'                       ␣
↪        |
| status                       | BUILD                                ␣
↪        |
| updated                      | 2017-03-30T00:59:44Z                 ␣
↪        |
| user_id                      | 1a421c69342348248c7696e3fd6d4366     ␣
↪        |
| volumes_attached             |                                      ␣
↪        |
+------------------------------+--------------------------------------
↪--------+
```

2. Check the status of your instance:

```
$ openstack server list

+--------------------------------------+-------------------+--------+-
↪-----------------------+------------+
| ID                                   | Name              | Status |␣
↪Networks               | Image Name |
+--------------------------------------+-------------------+--------+-
↪-----------------------+------------+
```

(continues on next page)

(continued from previous page)

```
| 181c52ba-aebc-4c32-a97d-2e8e82e4eaaf | provider-instance | ACTIVE |␣
↪provider=203.0.113.103 | cirros      |
+--------------------------------------+-------------------+--------+-
↪----------------------+------------+
```

The status changes from `BUILD` to `ACTIVE` when the build process successfully completes.

Access the instance using the virtual console

1. Obtain a *Virtual Network Computing (VNC)* session URL for your instance and access it from a web browser:

```
$ openstack console url show provider-instance

+-------+-------------------------------------------------------------------
↪--------------------+
| Field | Value
↪                    |
+-------+-------------------------------------------------------------------
↪--------------------+
| type  | novnc
↪                    |
| url   | http://controller:6080/vnc_auto.html?token=5eeccb47-525c-
↪4918-ac2a-3ad1e9f1f493 |
+-------+-------------------------------------------------------------------
↪--------------------+
```

Note: If your web browser runs on a host that cannot resolve the `controller` host name, you can replace `controller` with the IP address of the management interface on your controller node.

The CirrOS image includes conventional user name/password authentication and provides these credentials at the login prompt. After logging into CirrOS, we recommend that you verify network connectivity using `ping`.

2. Verify access to the provider physical network gateway:

```
$ ping -c 4 203.0.113.1

PING 203.0.113.1 (203.0.113.1) 56(84) bytes of data.
64 bytes from 203.0.113.1: icmp_req=1 ttl=64 time=0.357 ms
64 bytes from 203.0.113.1: icmp_req=2 ttl=64 time=0.473 ms
64 bytes from 203.0.113.1: icmp_req=3 ttl=64 time=0.504 ms
64 bytes from 203.0.113.1: icmp_req=4 ttl=64 time=0.470 ms

--- 203.0.113.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2998ms
rtt min/avg/max/mdev = 0.357/0.451/0.504/0.055 ms
```

3. Verify access to the internet:

```
$ ping -c 4 openstack.org

PING openstack.org (174.143.194.225) 56(84) bytes of data.
64 bytes from 174.143.194.225: icmp_req=1 ttl=53 time=17.4 ms
64 bytes from 174.143.194.225: icmp_req=2 ttl=53 time=17.5 ms
64 bytes from 174.143.194.225: icmp_req=3 ttl=53 time=17.7 ms
64 bytes from 174.143.194.225: icmp_req=4 ttl=53 time=17.5 ms

--- openstack.org ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 17.431/17.575/17.734/0.143 ms
```

Access the instance remotely

1. Verify connectivity to the instance from the controller node or any host on the provider physical network:

```
$ ping -c 4 203.0.113.103

PING 203.0.113.103 (203.0.113.103) 56(84) bytes of data.
64 bytes from 203.0.113.103: icmp_req=1 ttl=63 time=3.18 ms
64 bytes from 203.0.113.103: icmp_req=2 ttl=63 time=0.981 ms
64 bytes from 203.0.113.103: icmp_req=3 ttl=63 time=1.06 ms
64 bytes from 203.0.113.103: icmp_req=4 ttl=63 time=0.929 ms

--- 203.0.113.103 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 0.929/1.539/3.183/0.951 ms
```

2. Access your instance using SSH from the controller node or any host on the provider physical network:

```
$ ssh cirros@203.0.113.103

The authenticity of host '203.0.113.102 (203.0.113.102)' can't be␣
↪established.
RSA key fingerprint is␣
↪ed:05:e9:e7:52:a0:ff:83:68:94:c7:d1:f2:f8:e2:e9.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '203.0.113.102' (RSA) to the list of known␣
↪hosts.
```

If your instance does not launch or seem to work as you expect, see the Troubleshoot Compute documentation for Pike, the Troubleshoot Compute documentation for Queens, or the Troubleshoot Compute documentation for Rocky for more information or use one of the *many other options* to seek assistance. We want your first installation to work!

Return to *Launch an instance*.

7.5.2 Launch an instance on the self-service network

Determine instance options

To launch an instance, you must at least specify the flavor, image name, network, security group, key, and instance name.

1. On the controller node, source the `demo` credentials to gain access to user-only CLI commands:

```
$ . demo-openrc
```

2. A flavor specifies a virtual resource allocation profile which includes processor, memory, and storage.

   List available flavors:

```
$ openstack flavor list

+----+---------+-----+------+-----------+-------+-----------+
| ID | Name    | RAM | Disk | Ephemeral | VCPUs | Is Public |
+----+---------+-----+------+-----------+-------+-----------+
| 0  | m1.nano |  64 |    1 |         0 |     1 | True      |
+----+---------+-----+------+-----------+-------+-----------+
```

   Note: You can also reference a flavor by ID.

3. List available images:

```
$ openstack image list

+--------------------------------------+--------+--------+
| ID                                   | Name   | Status |
+--------------------------------------+--------+--------+
| 390eb5f7-8d49-41ec-95b7-68c0d5d54b34 | cirros | active |
+--------------------------------------+--------+--------+
```

   This instance uses the `cirros` image.

4. List available networks:

```
$ openstack network list

+--------------------------------------+-------------+--------------------
↪----------------------+
| ID                                   | Name        | Subnets
↪                      |
+--------------------------------------+-------------+--------------------
↪----------------------+
| 4716ddfe-6e60-40e7-b2a8-42e57bf3c31c | selfservice | 2112d5eb-f9d6-
↪45fd-906e-7cabd38b7c7c |
| b5b6993c-ddf9-40e7-91d0-86806a42edb8 | provider    | 310911f6-acf0-
↪4a47-824e-3032916582ff |
+--------------------------------------+-------------+--------------------
↪----------------------+
```

This instance uses the `selfservice` self-service network. However, you must reference this network using the ID instead of the name.

5. List available security groups:

```
$ openstack security group list

+--------------------------------------+---------+----------------------
↪----+
| ID                                   | Name    | Description
↪    |
+--------------------------------------+---------+----------------------
↪----+
| dd2b614c-3dad-48ed-958b-b155a3b38515 | default | Default security
↪group |
+--------------------------------------+---------+----------------------
↪----+
```

This instance uses the `default` security group.

6. Launch the instance:

Replace `SELFSERVICE_NET_ID` with the ID of the `selfservice` network.

```
$ openstack server create --flavor m1.nano --image cirros \
  --nic net-id=SELFSERVICE_NET_ID --security-group default \
  --key-name mykey selfservice-instance

+-------------------------------------+--------------------------------
↪---------+
| Field                               | Value
↪         |
+-------------------------------------+--------------------------------
↪---------+
| OS-DCF:diskConfig                   | MANUAL
↪         |
| OS-EXT-AZ:availability_zone         |
↪         |
| OS-EXT-STS:power_state              | 0
↪         |
| OS-EXT-STS:task_state               | scheduling
↪         |
| OS-EXT-STS:vm_state                 | building
↪         |
| OS-SRV-USG:launched_at              | None
↪         |
| OS-SRV-USG:terminated_at            | None
↪         |
| accessIPv4                          |
↪         |
| accessIPv6                          |
↪         |
| addresses                           |
↪         |
| adminPass                           | 7KTBYHSjEz7E
↪         |
| config_drive                        |
↪         |
```

(continues on next page)

(continued from previous page)

```
| created                              | 2016-02-26T14:52:37Z
↪          |
| flavor                               | m1.nano
↪          |
| hostId                               |
↪          |
| id                                   | 113c5892-e58e-4093-88c7-
↪e33f502eaaa4  |
| image                                | cirros (390eb5f7-8d49-41ec-
↪95b7-68c0d |
|                                      | 5d54b34)
↪          |
| key_name                             | mykey
↪          |
| name                                 | selfservice-instance
↪          |
| os-extended-volumes:volumes_attached | []
↪          |
| progress                             | 0
↪          |
| project_id                           |
↪ed0b60bf607743088218b0a533d5943f      |
| properties                           |
↪          |
| security_groups                      | [{u'name': u'default'}]
↪          |
| status                               | BUILD
↪          |
| updated                              | 2016-02-26T14:52:38Z
↪          |
| user_id                              |
↪58126687cbcc4888bfa9ab73a2256f27      |
+--------------------------------------+------------------------------
↪---------+
```

7. Check the status of your instance:

```
$ openstack server list

+--------------------------------------+----------------------+-------
↪-+--------------------------+
| ID                                   | Name                 |
↪Status | Networks                 |
+--------------------------------------+----------------------+-------
↪-+--------------------------+
| 113c5892-e58e-4093-88c7-e33f502eaaa4 | selfservice-instance |
↪ACTIVE | selfservice=172.16.1.3 |
| 181c52ba-aebc-4c32-a97d-2e8e82e4eaaf | provider-instance    |
↪ACTIVE | provider=203.0.113.103 |
+--------------------------------------+----------------------+-------
↪-+--------------------------+
```

The status changes from `BUILD` to `ACTIVE` when the build process successfully completes.

Access the instance using a virtual console

1. Obtain a *Virtual Network Computing (VNC)* session URL for your instance and access it from a web browser:

```
$ openstack console url show selfservice-instance

+-------+-----------------------------------------------------------------
↪--------------------+
| Field | Value
↪                    |
+-------+-----------------------------------------------------------------
↪--------------------+
| type  | novnc
↪                    |
| url   | http://controller:6080/vnc_auto.html?token=5eeccb47-525c-
↪4918-ac2a-3ad1e9f1f493 |
+-------+-----------------------------------------------------------------
↪--------------------+
```

Note: If your web browser runs on a host that cannot resolve the `controller` host name, you can replace `controller` with the IP address of the management interface on your controller node.

The CirrOS image includes conventional user name/password authentication and provides these credentials at the login prompt. After logging into CirrOS, we recommend that you verify network connectivity using `ping`.

2. Verify access to the self-service network gateway:

```
$ ping -c 4 172.16.1.1

PING 172.16.1.1 (172.16.1.1) 56(84) bytes of data.
64 bytes from 172.16.1.1: icmp_req=1 ttl=64 time=0.357 ms
64 bytes from 172.16.1.1: icmp_req=2 ttl=64 time=0.473 ms
64 bytes from 172.16.1.1: icmp_req=3 ttl=64 time=0.504 ms
64 bytes from 172.16.1.1: icmp_req=4 ttl=64 time=0.470 ms

--- 172.16.1.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2998ms
rtt min/avg/max/mdev = 0.357/0.451/0.504/0.055 ms
```

3. Verify access to the internet:

```
$ ping -c 4 openstack.org

PING openstack.org (174.143.194.225) 56(84) bytes of data.
64 bytes from 174.143.194.225: icmp_req=1 ttl=53 time=17.4 ms
64 bytes from 174.143.194.225: icmp_req=2 ttl=53 time=17.5 ms
64 bytes from 174.143.194.225: icmp_req=3 ttl=53 time=17.7 ms
64 bytes from 174.143.194.225: icmp_req=4 ttl=53 time=17.5 ms

--- openstack.org ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 17.431/17.575/17.734/0.143 ms
```

Access the instance remotely

1. Create a *floating IP address* on the provider virtual network:

```
$ openstack floating ip create provider

+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| created_at          | 2017-01-20T17:29:16Z                 |
| description         |                                      |
| fixed_ip_address    | None                                 |
| floating_ip_address | 203.0.113.104                        |
| floating_network_id | b5b6993c-ddf9-40e7-91d0-86806a42edb8 |
| headers             |                                      |
| id                  | 88b4d06a-d794-4406-affd-6ffa2bcf1e2a |
| port_id             | None                                 |
| project_id          | ed0b60bf607743088218b0a533d5943f     |
| revision_number     | 1                                    |
| router_id           | None                                 |
| status              | DOWN                                 |
| updated_at          | 2017-01-20T17:29:16Z                 |
+---------------------+--------------------------------------+
```

2. Associate the floating IP address with the instance:

```
$ openstack server add floating ip selfservice-instance 203.0.113.104
```

Note: This command provides no output.

3. Check the status of your floating IP address:

```
$ openstack server list

+--------------------------------------+----------------------+-------
↪-+--------------------------------------+
| ID                                   | Name                 |
↪Status | Networks                           |
+--------------------------------------+----------------------+-------
↪-+--------------------------------------+
| 113c5892-e58e-4093-88c7-e33f502eaaa4 | selfservice-instance |
↪ACTIVE | selfservice=172.16.1.3, 203.0.113.104 |
| 181c52ba-aebc-4c32-a97d-2e8e82e4eaaf | provider-instance    |
↪ACTIVE | provider=203.0.113.103               |
+--------------------------------------+----------------------+-------
↪-+--------------------------------------+
```

4. Verify connectivity to the instance via floating IP address from the controller node or any host on the provider physical network:

```
$ ping -c 4 203.0.113.104

PING 203.0.113.104 (203.0.113.104) 56(84) bytes of data.
64 bytes from 203.0.113.104: icmp_req=1 ttl=63 time=3.18 ms
```

(continues on next page)

(continued from previous page)

```
64 bytes from 203.0.113.104: icmp_req=2 ttl=63 time=0.981 ms
64 bytes from 203.0.113.104: icmp_req=3 ttl=63 time=1.06 ms
64 bytes from 203.0.113.104: icmp_req=4 ttl=63 time=0.929 ms

--- 203.0.113.104 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 0.929/1.539/3.183/0.951 ms
```

5. Access your instance using SSH from the controller node or any host on the provider physical network:

```
$ ssh cirros@203.0.113.104

The authenticity of host '203.0.113.104 (203.0.113.104)' can't be␣
↪established.
RSA key fingerprint is␣
↪ed:05:e9:e7:52:a0:ff:83:68:94:c7:d1:f2:f8:e2:e9.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '203.0.113.104' (RSA) to the list of known␣
↪hosts.
```

If your instance does not launch or seem to work as you expect, see the Troubleshoot Compute documentation for Pike, the Troubleshoot Compute documentation for Queens, or the Troubleshoot Compute documentation for Rocky for more information or use one of the *many other options* to seek assistance. We want your first installation to work!

Return to *Launch an instance*.

7.6 Block Storage

If your environment includes the Block Storage service, you can create a volume and attach it to an instance.

7.6.1 Block Storage

Create a volume

1. Source the `demo` credentials to perform the following steps as a non-administrative project:

```
$ . demo-openrc
```

2. Create a 1 GB volume:

```
$ openstack volume create --size 1 volume1

+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| attachments         | []                                   |
| availability_zone   | nova                                 |
| bootable            | false                                |
```

(continues on next page)

(continued from previous page)

```
| consistencygroup_id | None                                 |
| created_at          | 2016-03-08T14:30:48.391027           |
| description         | None                                 |
| encrypted           | False                                |
| id                  | a1e8be72-a395-4a6f-8e07-856a57c39524 |
| multiattach         | False                                |
| name                | volume1                              |
| properties          |                                      |
| replication_status  | disabled                             |
| size                | 1                                    |
| snapshot_id         | None                                 |
| source_volid        | None                                 |
| status              | creating                             |
| type                | None                                 |
| updated_at          | None                                 |
| user_id             | 684286a9079845359882afc3aa5011fb     |
+---------------------+--------------------------------------+
```

3. After a short time, the volume status should change from `creating` to `available`:

```
$ openstack volume list

+--------------------------------------+--------------+-----------+---
↪---+-------------+
| ID                                   | Display Name | Status    |␣
↪Size | Attached to |
+--------------------------------------+--------------+-----------+---
↪---+-------------+
| a1e8be72-a395-4a6f-8e07-856a57c39524 | volume1      | available |  ␣
↪ 1 |             |
+--------------------------------------+--------------+-----------+---
↪---+-------------+
```

Attach the volume to an instance

1. Attach a volume to an instance:

```
$ openstack server add volume INSTANCE_NAME VOLUME_NAME
```

Replace `INSTANCE_NAME` with the name of the instance and `VOLUME_NAME` with the name of the volume you want to attach to it.

Example

Attach the `volume1` volume to the `provider-instance` instance:

```
$ openstack server add volume provider-instance volume1
```

Note: This command provides no output.

2. List volumes:

```
$ openstack volume list

+--------------------------------------+--------------+--------+------
↪+----------------------------------------------+
| ID                                   | Display Name | Status | Size␣
↪| Attached to                                  |
+--------------------------------------+--------------+--------+------
↪+----------------------------------------------+
| a1e8be72-a395-4a6f-8e07-856a57c39524 | volume1      | in-use |    1␣
↪| Attached to provider-instance on /dev/vdb    |
+--------------------------------------+--------------+--------+------
↪+----------------------------------------------+
```

3. Access your instance using SSH and use the `fdisk` command to verify presence of the volume as the `/dev/vdb` block storage device:

```
$ sudo fdisk -l

Disk /dev/vda: 1073 MB, 1073741824 bytes
255 heads, 63 sectors/track, 130 cylinders, total 2097152 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

   Device Boot      Start         End      Blocks   Id  System
/dev/vda1   *       16065     2088449     1036192+  83  Linux

Disk /dev/vdb: 1073 MB, 1073741824 bytes
16 heads, 63 sectors/track, 2080 cylinders, total 2097152 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/vdb doesn't contain a valid partition table
```

Note: You must create a file system on the device and mount it to use the volume.

For more information about how to manage volumes, see the python-openstackclient documentation for Pike, the python-openstackclient documentation for Queens, or the python-openstackclient documentation for Rocky.

Return to *Launch an instance*.

7.7 Orchestration

If your environment includes the Orchestration service, you can create a stack that launches an instance.

For more information, see the Orchestration installation guide for Pike, the Orchestration installation guide for Queens, or the Orchestration installation guide for Rocky.

7.8 Shared File Systems

If your environment includes the Shared File Systems service, you can create a share and mount it in an instance.

For more information, see the Shared File Systems installation guide for Pike, the Shared File Systems installation guide for Queens, or the Shared File Systems installation guide for Rocky.

CHAPTER

EIGHT

FIREWALLS AND DEFAULT PORTS

On some deployments, such as ones where restrictive firewalls are in place, you might need to manually configure a firewall to permit OpenStack service traffic.

To manually configure a firewall, you must permit traffic through the ports that each OpenStack service uses. This table lists the default ports that each OpenStack service uses:

Table 1: Default ports that OpenStack components use

OpenStack service	Default ports	Port type
Application Catalog (murano)	8082	
Block Storage (cinder)	8776	publicurl and adminurl
Clustering (senlin)	8778	publicurl and adminurl
Compute (nova) endpoints	8774	publicurl and adminurl
Compute API (nova-api)	8773, 8775	
Compute ports for access to virtual machine consoles	5900-5999	
Compute VNC proxy for browsers (openstack-nova-novncproxy)	6080	
Compute VNC proxy for traditional VNC clients (openstack-nova-xvpvncproxy)	6081	
Proxy port for HTML5 console used by Compute service	6082	
Data processing service (sahara) endpoint	8386	publicurl and adminurl
Identity service (keystone) administrative endpoint	5000	adminurl
Identity service public endpoint	5000	publicurl
Image service (glance) API	9292	publicurl and adminurl
Image service registry	9191	
Networking (neutron)	9696	publicurl and adminurl
Object Storage (swift)	6000, 6001, 6002	
Orchestration (heat) endpoint	8004	publicurl and adminurl
Orchestration AWS CloudFormation-compatible API (openstack-heat-api-cfn)	8000	
Orchestration AWS CloudWatch-compatible API (openstack-heat-api-cloudwatch)	8003	
Root Cause Analysis service (Vitrage)	8999	
Telemetry (ceilometer)	8777	publicurl and adminurl
Workflow service (Mistral)	8989	

To function properly, some OpenStack components depend on other, non-OpenStack services. For example, the OpenStack dashboard uses HTTP for non-secure communication. In this case, you must configure the firewall to allow traffic to and from HTTP.

This table lists the ports that other OpenStack components use:

Table 2: Default ports that secondary services related to OpenStack components use

Service	Default port	Used by
HTTP	80	OpenStack dashboard (`Horizon`) when it is not configured to use secure access.
HTTP alternate	8080	OpenStack Object Storage (`swift`) service.
HTTPS	443	Any OpenStack service that is enabled for SSL, especially secure-access dashboard.
rsync	873	OpenStack Object Storage. Required.
iSCSI target	3260	OpenStack Block Storage. Required.
MySQL database service	3306	Most OpenStack components.
Message Broker (AMQP traffic)	5672	OpenStack Block Storage, Networking, Orchestration, and Compute.

On some deployments, the default port used by a service may fall within the defined local port range of a host. To check a hosts local port range:

```
$ sysctl net.ipv4.ip_local_port_range
```

If a services default port falls within this range, run the following program to check if the port has already been assigned to another application:

```
$ lsof -i :PORT
```

Configure the service to use a different port if the default port is already being used by another application.

CHAPTER

NINE

APPENDIX

9.1 Community support

The following resources are available to help you run and use OpenStack. The OpenStack community constantly improves and adds to the main features of OpenStack, but if you have any questions, do not hesitate to ask. Use the following resources to get OpenStack support and troubleshoot your installations.

9.1.1 Documentation

For the available OpenStack documentation, see docs.openstack.org.

The following guides explain how to install a Proof-of-Concept OpenStack cloud and its associated components:

- Rocky Installation Guides

The following books explain how to configure and run an OpenStack cloud:

- Architecture Design Guide
- Rocky Administrator Guides
- Rocky Configuration Guides
- Rocky Networking Guide
- High Availability Guide
- Security Guide
- Virtual Machine Image Guide

The following book explains how to use the command-line clients:

- Rocky API Bindings

The following documentation provides reference and guidance information for the OpenStack APIs:

- API Documentation

The following guide provides information on how to contribute to OpenStack documentation:

- Documentation Contributor Guide

9.1.2 The OpenStack wiki

The OpenStack wiki contains a broad range of topics but some of the information can be difficult to find or is a few pages deep. Fortunately, the wiki search feature enables you to search by title or content. If you search for specific information, such as about networking or OpenStack Compute, you can find a large amount of relevant material. More is being added all the time, so be sure to check back often. You can find the search box in the upper-right corner of any OpenStack wiki page.

9.1.3 The Launchpad bugs area

The OpenStack community values your set up and testing efforts and wants your feedback. To log a bug, you must sign up for a Launchpad account. You can view existing bugs and report bugs in the Launchpad Bugs area. Use the search feature to determine whether the bug has already been reported or already been fixed. If it still seems like your bug is unreported, fill out a bug report.

Some tips:

- Give a clear, concise summary.
- Provide as much detail as possible in the description. Paste in your command output or stack traces, links to screen shots, and any other information which might be useful.
- Be sure to include the software and package versions that you are using, especially if you are using a development branch, such as, `"Kilo release" vs git commit bc79c3ecc55929bac585d04a03475b72e06a3208`.
- Any deployment-specific information is helpful, such as whether you are using Ubuntu 14.04 or are performing a multi-node installation.

The following Launchpad Bugs areas are available:

- Bugs: OpenStack Block Storage (cinder)
- Bugs: OpenStack Compute (nova)
- Bugs: OpenStack Dashboard (horizon)
- Bugs: OpenStack Identity (keystone)
- Bugs: OpenStack Image service (glance)
- Bugs: OpenStack Networking (neutron)
- Bugs: OpenStack Object Storage (swift)
- Bugs: Application catalog (murano)
- Bugs: Bare metal service (ironic)
- Bugs: Clustering service (senlin)
- Bugs: Container Infrastructure Management service (magnum)
- Bugs: Data processing service (sahara)
- Bugs: Database service (trove)
- Bugs: DNS service (designate)
- Bugs: Key Manager Service (barbican)
- Bugs: Monitoring (monasca)

- Bugs: Orchestration (heat)
- Bugs: Rating (cloudkitty)
- Bugs: Shared file systems (manila)
- Bugs: Telemetry (ceilometer)
- Bugs: Telemetry v3 (gnocchi)
- Bugs: Workflow service (mistral)
- Bugs: Messaging service (zaqar)
- Bugs: Container service (zun)
- Bugs: Function engine (qinling)
- Bugs: OpenStack API Documentation (developer.openstack.org)
- Bugs: OpenStack Documentation (docs.openstack.org)

9.1.4 Documentation feedback

To provide feedback on documentation, join our IRC channel `#openstack-doc` on the Freenode IRC network, or report a bug in Launchpad and choose the particular project that the documentation is a part of.

9.1.5 The OpenStack IRC channel

The OpenStack community lives in the #openstack IRC channel on the Freenode network. You can hang out, ask questions, or get immediate feedback for urgent and pressing issues. To install an IRC client or use a browser-based client, go to https://webchat.freenode.net/. You can also use Colloquy (Mac OS X), mIRC (Windows), or XChat (Linux). When you are in the IRC channel and want to share code or command output, the generally accepted method is to use a Paste Bin. The OpenStack project has one at Paste. Just paste your longer amounts of text or logs in the web form and you get a URL that you can paste into the channel. The OpenStack IRC channel is `#openstack` on `irc.freenode.net`. You can find a list of all OpenStack IRC channels on the IRC page on the wiki.

9.1.6 OpenStack mailing lists

A great way to get answers and insights is to post your question or problematic scenario to the OpenStack mailing list. You can learn from and help others who might have similar issues. To subscribe or view the archives, go to the general OpenStack mailing list. If you are interested in the other mailing lists for specific projects or development, refer to Mailing Lists.

9.1.7 OpenStack distribution packages

The following Linux distributions provide community-supported packages for OpenStack:

- **CentOS, Fedora, and Red Hat Enterprise Linux:** https://www.rdoproject.org/
- **openSUSE and SUSE Linux Enterprise Server:** https://en.opensuse.org/Portal:OpenStack
- **Ubuntu:** https://wiki.ubuntu.com/OpenStack/CloudArchive

9.2 Glossary

This glossary offers a list of terms and definitions to define a vocabulary for OpenStack-related concepts.

To add to OpenStack glossary, clone the openstack/openstack-manuals repository and update the source file `doc/common/glossary.rst` through the OpenStack contribution process.

9.2.1 0-9

6to4 A mechanism that allows IPv6 packets to be transmitted over an IPv4 network, providing a strategy for migrating to IPv6.

9.2.2 A

absolute limit Impassable limits for guest VMs. Settings include total RAM size, maximum number of vCPUs, and maximum disk size.

access control list (ACL) A list of permissions attached to an object. An ACL specifies which users or system processes have access to objects. It also defines which operations can be performed on specified objects. Each entry in a typical ACL specifies a subject and an operation. For instance, the ACL entry `(Alice, delete)` for a file gives Alice permission to delete the file.

access key Alternative term for an Amazon EC2 access key. See EC2 access key.

account The Object Storage context of an account. Do not confuse with a user account from an authentication service, such as Active Directory, /etc/passwd, OpenLDAP, OpenStack Identity, and so on.

account auditor Checks for missing replicas and incorrect or corrupted objects in a specified Object Storage account by running queries against the back-end SQLite database.

account database A SQLite database that contains Object Storage accounts and related metadata and that the accounts server accesses.

account reaper An Object Storage worker that scans for and deletes account databases and that the account server has marked for deletion.

account server Lists containers in Object Storage and stores container information in the account database.

account service An Object Storage component that provides account services such as list, create, modify, and audit. Do not confuse with OpenStack Identity service, OpenLDAP, or similar user-account services.

accounting The Compute service provides accounting information through the event notification and system usage data facilities.

Active Directory Authentication and identity service by Microsoft, based on LDAP. Supported in OpenStack.

active/active configuration In a high-availability setup with an active/active configuration, several systems share the load together and if one fails, the load is distributed to the remaining systems.

active/passive configuration In a high-availability setup with an active/passive configuration, systems are set up to bring additional resources online to replace those that have failed.

address pool A group of fixed and/or floating IP addresses that are assigned to a project and can be used by or assigned to the VM instances in a project.

Address Resolution Protocol (ARP) The protocol by which layer-3 IP addresses are resolved into layer-2 link local addresses.

admin API A subset of API calls that are accessible to authorized administrators and are generally not accessible to end users or the public Internet. They can exist as a separate service (keystone) or can be a subset of another API (nova).

admin server In the context of the Identity service, the worker process that provides access to the admin API.

administrator The person responsible for installing, configuring, and managing an OpenStack cloud.

Advanced Message Queuing Protocol (AMQP) The open standard messaging protocol used by OpenStack components for intra-service communications, provided by RabbitMQ, Qpid, or ZeroMQ.

Advanced RISC Machine (ARM) Lower power consumption CPU often found in mobile and embedded devices. Supported by OpenStack.

alert The Compute service can send alerts through its notification system, which includes a facility to create custom notification drivers. Alerts can be sent to and displayed on the dashboard.

allocate The process of taking a floating IP address from the address pool so it can be associated with a fixed IP on a guest VM instance.

Amazon Kernel Image (AKI) Both a VM container format and disk format. Supported by Image service.

Amazon Machine Image (AMI) Both a VM container format and disk format. Supported by Image service.

Amazon Ramdisk Image (ARI) Both a VM container format and disk format. Supported by Image service.

Anvil A project that ports the shell script-based project named DevStack to Python.

aodh Part of the OpenStack *Telemetry service*; provides alarming functionality.

Apache The Apache Software Foundation supports the Apache community of open-source software projects. These projects provide software products for the public good.

Apache License 2.0 All OpenStack core projects are provided under the terms of the Apache License 2.0 license.

Apache Web Server The most common web server software currently used on the Internet.

API endpoint The daemon, worker, or service that a client communicates with to access an API. API endpoints can provide any number of services, such as authentication, sales data, performance meters, Compute VM commands, census data, and so on.

API extension Custom modules that extend some OpenStack core APIs.

API extension plug-in Alternative term for a Networking plug-in or Networking API extension.

API key Alternative term for an API token.

API server Any node running a daemon or worker that provides an API endpoint.

API token Passed to API requests and used by OpenStack to verify that the client is authorized to run the requested operation.

API version In OpenStack, the API version for a project is part of the URL. For example, `example.com/nova/v1/foobar`.

applet A Java program that can be embedded into a web page.

Application Catalog service (murano) The project that provides an application catalog service so that users can compose and deploy composite environments on an application abstraction level while managing the application lifecycle.

Application Programming Interface (API) A collection of specifications used to access a service, application, or program. Includes service calls, required parameters for each call, and the expected return values.

application server A piece of software that makes available another piece of software over a network.

Application Service Provider (ASP) Companies that rent specialized applications that help businesses and organizations provide additional services with lower cost.

arptables Tool used for maintaining Address Resolution Protocol packet filter rules in the Linux kernel firewall modules. Used along with iptables, ebtables, and ip6tables in Compute to provide firewall services for VMs.

associate The process associating a Compute floating IP address with a fixed IP address.

Asynchronous JavaScript and XML (AJAX) A group of interrelated web development techniques used on the client-side to create asynchronous web applications. Used extensively in horizon.

ATA over Ethernet (AoE) A disk storage protocol tunneled within Ethernet.

attach The process of connecting a VIF or vNIC to a L2 network in Networking. In the context of Compute, this process connects a storage volume to an instance.

attachment (network) Association of an interface ID to a logical port. Plugs an interface into a port.

auditing Provided in Compute through the system usage data facility.

auditor A worker process that verifies the integrity of Object Storage objects, containers, and accounts. Auditors is the collective term for the Object Storage account auditor, container auditor, and object auditor.

Austin The code name for the initial release of OpenStack. The first design summit took place in Austin, Texas, US.

auth node Alternative term for an Object Storage authorization node.

authentication The process that confirms that the user, process, or client is really who they say they are through private key, secret token, password, fingerprint, or similar method.

authentication token A string of text provided to the client after authentication. Must be provided by the user or process in subsequent requests to the API endpoint.

AuthN The Identity service component that provides authentication services.

authorization The act of verifying that a user, process, or client is authorized to perform an action.

authorization node An Object Storage node that provides authorization services.

AuthZ The Identity component that provides high-level authorization services.

Auto ACK Configuration setting within RabbitMQ that enables or disables message acknowledgment. Enabled by default.

auto declare A Compute RabbitMQ setting that determines whether a message exchange is automatically created when the program starts.

availability zone An Amazon EC2 concept of an isolated area that is used for fault tolerance. Do not confuse with an OpenStack Compute zone or cell.

AWS CloudFormation template AWS CloudFormation allows Amazon Web Services (AWS) users to create and manage a collection of related resources. The Orchestration service supports a CloudFormation-compatible format (CFN).

9.2.3 B

back end Interactions and processes that are obfuscated from the user, such as Compute volume mount, data transmission to an iSCSI target by a daemon, or Object Storage object integrity checks.

back-end catalog The storage method used by the Identity service catalog service to store and retrieve information about API endpoints that are available to the client. Examples include an SQL database, LDAP database, or KVS back end.

back-end store The persistent data store used to save and retrieve information for a service, such as lists of Object Storage objects, current state of guest VMs, lists of user names, and so on. Also, the method that the Image service uses to get and store VM images. Options include Object Storage, locally mounted file system, RADOS block devices, VMware datastore, and HTTP.

Backup, Restore, and Disaster Recovery service (freezer) The project that provides integrated tooling for backing up, restoring, and recovering file systems, instances, or database backups.

bandwidth The amount of available data used by communication resources, such as the Internet. Represents the amount of data that is used to download things or the amount of data available to download.

barbican Code name of the *Key Manager service*.

bare An Image service container format that indicates that no container exists for the VM image.

Bare Metal service (ironic) The OpenStack service that provides a service and associated libraries capable of managing and provisioning physical machines in a security-aware and fault-tolerant manner.

base image An OpenStack-provided image.

Bell-LaPadula model A security model that focuses on data confidentiality and controlled access to classified information. This model divides the entities into subjects and objects. The clearance of a subject is compared to the classification of the object to determine if the subject is authorized for the specific access mode. The clearance or classification scheme is expressed in terms of a lattice.

Benchmark service (rally) OpenStack project that provides a framework for performance analysis and benchmarking of individual OpenStack components as well as full production OpenStack cloud deployments.

Bexar A grouped release of projects related to OpenStack that came out in February of 2011. It included only Compute (nova) and Object Storage (swift). Bexar is the code name for the second release of OpenStack. The design summit took place in San Antonio, Texas, US, which is the county seat for Bexar county.

binary Information that consists solely of ones and zeroes, which is the language of computers.

bit A bit is a single digit number that is in base of 2 (either a zero or one). Bandwidth usage is measured in bits per second.

bits per second (BPS) The universal measurement of how quickly data is transferred from place to place.

block device A device that moves data in the form of blocks. These device nodes interface the devices, such as hard disks, CD-ROM drives, flash drives, and other addressable regions of memory.

block migration A method of VM live migration used by KVM to evacuate instances from one host to another with very little downtime during a user-initiated switchover. Does not require shared storage. Supported by Compute.

Block Storage API An API on a separate endpoint for attaching, detaching, and creating block storage for compute VMs.

Block Storage service (cinder) The OpenStack service that implement services and libraries to provide on-demand, self-service access to Block Storage resources via abstraction and automation on top of other block storage devices.

BMC (Baseboard Management Controller) The intelligence in the IPMI architecture, which is a specialized micro-controller that is embedded on the motherboard of a computer and acts as a server. Manages the interface between system management software and platform hardware.

bootable disk image A type of VM image that exists as a single, bootable file.

Bootstrap Protocol (BOOTP) A network protocol used by a network client to obtain an IP address from a configuration server. Provided in Compute through the dnsmasq daemon when using either the FlatDHCP manager or VLAN manager network manager.

Border Gateway Protocol (BGP) The Border Gateway Protocol is a dynamic routing protocol that connects autonomous systems. Considered the backbone of the Internet, this protocol connects disparate networks to form a larger network.

browser Any client software that enables a computer or device to access the Internet.

builder file Contains configuration information that Object Storage uses to reconfigure a ring or to re-create it from scratch after a serious failure.

bursting The practice of utilizing a secondary environment to elastically build instances on-demand when the primary environment is resource constrained.

button class A group of related button types within horizon. Buttons to start, stop, and suspend VMs are in one class. Buttons to associate and disassociate floating IP addresses are in another class, and so on.

byte Set of bits that make up a single character; there are usually 8 bits to a byte.

9.2.4 C

cache pruner A program that keeps the Image service VM image cache at or below its configured maximum size.

Cactus An OpenStack grouped release of projects that came out in the spring of 2011. It included Compute (nova), Object Storage (swift), and the Image service (glance). Cactus is a city in Texas, US and is the code name for the third release of OpenStack. When OpenStack releases went from three to six months long, the code name of the release changed to match a geography nearest the previous summit.

CALL One of the RPC primitives used by the OpenStack message queue software. Sends a message and waits for a response.

capability Defines resources for a cell, including CPU, storage, and networking. Can apply to the specific services within a cell or a whole cell.

capacity cache A Compute back-end database table that contains the current workload, amount of free RAM, and number of VMs running on each host. Used to determine on which host a VM starts.

capacity updater A notification driver that monitors VM instances and updates the capacity cache as needed.

CAST One of the RPC primitives used by the OpenStack message queue software. Sends a message and does not wait for a response.

catalog A list of API endpoints that are available to a user after authentication with the Identity service.

catalog service An Identity service that lists API endpoints that are available to a user after authentication with the Identity service.

ceilometer Part of the OpenStack *Telemetry service*; gathers and stores metrics from other OpenStack services.

cell Provides logical partitioning of Compute resources in a child and parent relationship. Requests are passed from parent cells to child cells if the parent cannot provide the requested resource.

cell forwarding A Compute option that enables parent cells to pass resource requests to child cells if the parent cannot provide the requested resource.

cell manager The Compute component that contains a list of the current capabilities of each host within the cell and routes requests as appropriate.

CentOS A Linux distribution that is compatible with OpenStack.

Ceph Massively scalable distributed storage system that consists of an object store, block store, and POSIX-compatible distributed file system. Compatible with OpenStack.

CephFS The POSIX-compliant file system provided by Ceph.

certificate authority (CA) In cryptography, an entity that issues digital certificates. The digital certificate certifies the ownership of a public key by the named subject of the certificate. This enables others (relying parties) to rely upon signatures or assertions made by the private key that corresponds to the certified public key. In this model of trust relationships, a CA is a trusted third party for both the subject (owner) of the certificate and the party relying upon the certificate. CAs are characteristic of many public key infrastructure (PKI) schemes. In OpenStack, a simple certificate authority is provided by Compute for cloudpipe VPNs and VM image decryption.

Challenge-Handshake Authentication Protocol (CHAP) An iSCSI authentication method supported by Compute.

chance scheduler A scheduling method used by Compute that randomly chooses an available host from the pool.

changes since A Compute API parameter that downloads changes to the requested item since your last request, instead of downloading a new, fresh set of data and comparing it against the old data.

Chef An operating system configuration management tool supporting OpenStack deployments.

child cell If a requested resource such as CPU time, disk storage, or memory is not available in the parent cell, the request is forwarded to its associated child cells. If the child cell can fulfill the request, it does. Otherwise, it attempts to pass the request to any of its children.

cinder Codename for *Block Storage service*.

CirrOS A minimal Linux distribution designed for use as a test image on clouds such as OpenStack.

Cisco neutron plug-in A Networking plug-in for Cisco devices and technologies, including UCS and Nexus.

cloud architect A person who plans, designs, and oversees the creation of clouds.

Cloud Auditing Data Federation (CADF) Cloud Auditing Data Federation (CADF) is a specification for audit event data. CADF is supported by OpenStack Identity.

cloud computing A model that enables access to a shared pool of configurable computing resources, such as networks, servers, storage, applications, and services, that can be rapidly provisioned and released with minimal management effort or service provider interaction.

cloud computing infrastructure The hardware and software components such as servers, storage, and network and virtualization software that are needed to support the computing requirements of a cloud computing model.

cloud computing platform software The delivery of different services through the Internet. These resources include tools and applications like data storage, servers, databases, networking, and software. As long as an electronic device has access to the web, it has access to the data and the software programs to run it.

cloud computing service architecture Cloud service architecture defines the overall cloud computing services and solutions that are implemented in and across the boundaries of an enterprise business network. Considers the core business requirements and matches them with a possible cloud solution.

cloud controller Collection of Compute components that represent the global state of the cloud; talks to services, such as Identity authentication, Object Storage, and node/storage workers through a queue.

cloud controller node A node that runs network, volume, API, scheduler, and image services. Each service may be broken out into separate nodes for scalability or availability.

Cloud Data Management Interface (CDMI) SINA standard that defines a RESTful API for managing objects in the cloud, currently unsupported in OpenStack.

Cloud Infrastructure Management Interface (CIMI) An in-progress specification for cloud management. Currently unsupported in OpenStack.

cloud technology Clouds are tools of virtual sources orchestrated by management and automation softwares. This includes, raw processing power, memory, network, storage of cloud based applications.

cloud-init A package commonly installed in VM images that performs initialization of an instance after boot using information that it retrieves from the metadata service, such as the SSH public key and user data.

cloudadmin One of the default roles in the Compute RBAC system. Grants complete system access.

Cloudbase-Init A Windows project providing guest initialization features, similar to cloud-init.

cloudpipe A compute service that creates VPNs on a per-project basis.

cloudpipe image A pre-made VM image that serves as a cloudpipe server. Essentially, OpenVPN running on Linux.

Clustering service (senlin) The project that implements clustering services and libraries for the management of groups of homogeneous objects exposed by other OpenStack services.

command filter Lists allowed commands within the Compute rootwrap facility.

Command-Line Interface (CLI) A text-based client that helps you create scripts to interact with OpenStack clouds.

Common Internet File System (CIFS) A file sharing protocol. It is a public or open variation of the original Server Message Block (SMB) protocol developed and used by Microsoft. Like the SMB protocol, CIFS runs at a higher level and uses the TCP/IP protocol.

Common Libraries (oslo) The project that produces a set of python libraries containing code shared by OpenStack projects. The APIs provided by these libraries should be high quality, stable, consistent, documented and generally applicable.

community project A project that is not officially endorsed by the OpenStack Foundation. If the project is successful enough, it might be elevated to an incubated project and then to a core project, or it might be merged with the main code trunk.

compression Reducing the size of files by special encoding, the file can be decompressed again to its original content. OpenStack supports compression at the Linux file system level but does not support compression for things such as Object Storage objects or Image service VM images.

Compute API (nova API) The nova-api daemon provides access to nova services. Can communicate with other APIs, such as the Amazon EC2 API.

compute controller The Compute component that chooses suitable hosts on which to start VM instances.

compute host Physical host dedicated to running compute nodes.

compute node A node that runs the nova-compute daemon that manages VM instances that provide a wide range of services, such as web applications and analytics.

Compute service (nova) The OpenStack core project that implements services and associated libraries to provide massively-scalable, on-demand, self-service access to compute resources, including bare metal, virtual machines, and containers.

compute worker The Compute component that runs on each compute node and manages the VM instance lifecycle, including run, reboot, terminate, attach/detach volumes, and so on. Provided by the nova-compute daemon.

concatenated object A set of segment objects that Object Storage combines and sends to the client.

conductor In Compute, conductor is the process that proxies database requests from the compute process. Using conductor improves security because compute nodes do not need direct access to the database.

congress Code name for the *Governance service*.

consistency window The amount of time it takes for a new Object Storage object to become accessible to all clients.

console log Contains the output from a Linux VM console in Compute.

container Organizes and stores objects in Object Storage. Similar to the concept of a Linux directory but cannot be nested. Alternative term for an Image service container format.

container auditor Checks for missing replicas or incorrect objects in specified Object Storage containers through queries to the SQLite back-end database.

container database A SQLite database that stores Object Storage containers and container metadata. The container server accesses this database.

container format A wrapper used by the Image service that contains a VM image and its associated metadata, such as machine state, OS disk size, and so on.

Container Infrastructure Management service (magnum) The project which provides a set of services for provisioning, scaling, and managing container orchestration engines.

container server An Object Storage server that manages containers.

container service The Object Storage component that provides container services, such as create, delete, list, and so on.

content delivery network (CDN) A content delivery network is a specialized network that is used to distribute content to clients, typically located close to the client for increased performance.

continuous delivery A software engineering approach in which teams produce software in short cycles, ensuring that the software can be reliably released at any time and, when releasing the software, doing so manually.

continuous deployment A software release process that uses automated testing to validate if changes to a codebase are correct and stable for immediate autonomous deployment to a production environment.

continuous integration The practice of merging all developers working copies to a shared mainline several times a day.

controller node Alternative term for a cloud controller node.

core API Depending on context, the core API is either the OpenStack API or the main API of a specific core project, such as Compute, Networking, Image service, and so on.

core service An official OpenStack service defined as core by DefCore Committee. Currently, consists of Block Storage service (cinder), Compute service (nova), Identity service (keystone), Image service (glance), Networking service (neutron), and Object Storage service (swift).

cost Under the Compute distributed scheduler, this is calculated by looking at the capabilities of each host relative to the flavor of the VM instance being requested.

credentials Data that is only known to or accessible by a user and used to verify that the user is who he says he is. Credentials are presented to the server during authentication. Examples include a password, secret key, digital certificate, and fingerprint.

CRL A Certificate Revocation List (CRL) in a PKI model is a list of certificates that have been revoked. End entities presenting these certificates should not be trusted.

Cross-Origin Resource Sharing (CORS) A mechanism that allows many resources (for example, fonts, JavaScript) on a web page to be requested from another domain outside the domain from which the resource originated. In particular, JavaScripts AJAX calls can use the XMLHttpRequest mechanism.

Crowbar An open source community project by SUSE that aims to provide all necessary services to quickly deploy and manage clouds.

current workload An element of the Compute capacity cache that is calculated based on the number of build, snapshot, migrate, and resize operations currently in progress on a given host.

customer Alternative term for project.

customization module A user-created Python module that is loaded by horizon to change the look and feel of the dashboard.

9.2.5 D

daemon A process that runs in the background and waits for requests. May or may not listen on a TCP or UDP port. Do not confuse with a worker.

Dashboard (horizon) OpenStack project which provides an extensible, unified, web-based user interface for all OpenStack services.

data encryption Both Image service and Compute support encrypted virtual machine (VM) images (but not instances). In-transit data encryption is supported in OpenStack using technologies such as HTTPS, SSL, TLS, and SSH. Object Storage does not support object encryption at the application level but may support storage that uses disk encryption.

Data loss prevention (DLP) software Software programs used to protect sensitive information and prevent it from leaking outside a network boundary through the detection and denying of the data transportation.

Data Processing service (sahara) OpenStack project that provides a scalable data-processing stack and associated management interfaces.

data store A database engine supported by the Database service.

database ID A unique ID given to each replica of an Object Storage database.

database replicator An Object Storage component that copies changes in the account, container, and object databases to other nodes.

Database service (trove) An integrated project that provides scalable and reliable Cloud Database-as-a-Service functionality for both relational and non-relational database engines.

deallocate The process of removing the association between a floating IP address and a fixed IP address. Once this association is removed, the floating IP returns to the address pool.

Debian A Linux distribution that is compatible with OpenStack.

deduplication The process of finding duplicate data at the disk block, file, and/or object level to minimize storage usecurrently unsupported within OpenStack.

default panel The default panel that is displayed when a user accesses the dashboard.

default project New users are assigned to this project if no project is specified when a user is created.

default token An Identity service token that is not associated with a specific project and is exchanged for a scoped token.

delayed delete An option within Image service so that an image is deleted after a predefined number of seconds instead of immediately.

delivery mode Setting for the Compute RabbitMQ message delivery mode; can be set to either transient or persistent.

denial of service (DoS) Denial of service (DoS) is a short form for denial-of-service attack. This is a malicious attempt to prevent legitimate users from using a service.

deprecated auth An option within Compute that enables administrators to create and manage users through the `nova-manage` command as opposed to using the Identity service.

designate Code name for the *DNS service*.

Desktop-as-a-Service A platform that provides a suite of desktop environments that users access to receive a desktop experience from any location. This may provide general use, development, or even homogeneous testing environments.

developer One of the default roles in the Compute RBAC system and the default role assigned to a new user.

device ID Maps Object Storage partitions to physical storage devices.

device weight Distributes partitions proportionately across Object Storage devices based on the storage capacity of each device.

DevStack Community project that uses shell scripts to quickly build complete OpenStack development environments.

DHCP agent OpenStack Networking agent that provides DHCP services for virtual networks.

Diablo A grouped release of projects related to OpenStack that came out in the fall of 2011, the fourth release of OpenStack. It included Compute (nova 2011.3), Object Storage (swift 1.4.3), and the Image service (glance). Diablo is the code name for the fourth release of OpenStack. The design summit took place in the Bay Area near Santa Clara, California, US and Diablo is a nearby city.

direct consumer An element of the Compute RabbitMQ that comes to life when a RPC call is executed. It connects to a direct exchange through a unique exclusive queue, sends the message, and terminates.

direct exchange A routing table that is created within the Compute RabbitMQ during RPC calls; one is created for each RPC call that is invoked.

direct publisher Element of RabbitMQ that provides a response to an incoming MQ message.

disassociate The process of removing the association between a floating IP address and fixed IP and thus returning the floating IP address to the address pool.

Discretionary Access Control (DAC) Governs the ability of subjects to access objects, while enabling users to make policy decisions and assign security attributes. The traditional UNIX system of users, groups, and read-write-execute permissions is an example of DAC.

disk encryption The ability to encrypt data at the file system, disk partition, or whole-disk level. Supported within Compute VMs.

disk format The underlying format that a disk image for a VM is stored as within the Image service back-end store. For example, AMI, ISO, QCOW2, VMDK, and so on.

dispersion In Object Storage, tools to test and ensure dispersion of objects and containers to ensure fault tolerance.

distributed virtual router (DVR) Mechanism for highly available multi-host routing when using OpenStack Networking (neutron).

Django A web framework used extensively in horizon.

DNS record A record that specifies information about a particular domain and belongs to the domain.

DNS service (designate) OpenStack project that provides scalable, on demand, self service access to authoritative DNS services, in a technology-agnostic manner.

dnsmasq Daemon that provides DNS, DHCP, BOOTP, and TFTP services for virtual networks.

domain An Identity API v3 entity. Represents a collection of projects, groups and users that defines administrative boundaries for managing OpenStack Identity entities. On the Internet, separates a website from other sites. Often, the domain name has two or more parts that are separated by dots. For example, yahoo.com, usa.gov, harvard.edu, or mail.yahoo.com. Also, a domain is an entity or container of all DNS-related information containing one or more records.

Domain Name System (DNS) A system by which Internet domain name-to-address and address-to-name resolutions are determined. DNS helps navigate the Internet by translating the IP address into an address that is easier to remember. For example, translating 111.111.111.1 into www.yahoo.com. All domains and their components, such as mail servers, utilize DNS to resolve to the appropriate locations. DNS servers are usually set up in a master-slave relationship such that failure of the master invokes the slave. DNS servers might also be clustered or replicated such that changes made to one DNS server are automatically propagated to other active servers. In Compute, the support that enables associating DNS entries with floating IP addresses, nodes, or cells so that hostnames are consistent across reboots.

download The transfer of data, usually in the form of files, from one computer to another.

durable exchange The Compute RabbitMQ message exchange that remains active when the server restarts.

durable queue A Compute RabbitMQ message queue that remains active when the server restarts.

Dynamic Host Configuration Protocol (DHCP) A network protocol that configures devices that are connected to a network so that they can communicate on that network by using the Internet Protocol (IP). The protocol is implemented in a client-server model where DHCP clients request configuration data, such as an IP address, a default route, and one or more DNS server addresses from a DHCP server. A method to automatically configure networking for a host at boot time. Provided by both Networking and Compute.

Dynamic HyperText Markup Language (DHTML) Pages that use HTML, JavaScript, and Cascading Style Sheets to enable users to interact with a web page or show simple animation.

9.2.6 E

east-west traffic Network traffic between servers in the same cloud or data center. See also north-south traffic.

EBS boot volume An Amazon EBS storage volume that contains a bootable VM image, currently unsupported in OpenStack.

ebtables Filtering tool for a Linux bridging firewall, enabling filtering of network traffic passing through a Linux bridge. Used in Compute along with arptables, iptables, and ip6tables to ensure isolation of network communications.

EC2 The Amazon commercial compute product, similar to Compute.

EC2 access key Used along with an EC2 secret key to access the Compute EC2 API.

EC2 API OpenStack supports accessing the Amazon EC2 API through Compute.

EC2 Compatibility API A Compute component that enables OpenStack to communicate with Amazon EC2.

EC2 secret key Used along with an EC2 access key when communicating with the Compute EC2 API; used to digitally sign each request.

edge computing Running fewer processes in the cloud and moving those processes to local places.

Elastic Block Storage (EBS) The Amazon commercial block storage product.

encapsulation The practice of placing one packet type within another for the purposes of abstracting or securing data. Examples include GRE, MPLS, or IPsec.

encryption OpenStack supports encryption technologies such as HTTPS, SSH, SSL, TLS, digital certificates, and data encryption.

endpoint See API endpoint.

endpoint registry Alternative term for an Identity service catalog.

endpoint template A list of URL and port number endpoints that indicate where a service, such as Object Storage, Compute, Identity, and so on, can be accessed.

enterprise cloud computing A computing environment residing behind a firewall that delivers software, infrastructure and platform services to an enterprise.

entity Any piece of hardware or software that wants to connect to the network services provided by Networking, the network connectivity service. An entity can make use of Networking by implementing a VIF.

ephemeral image A VM image that does not save changes made to its volumes and reverts them to their original state after the instance is terminated.

ephemeral volume Volume that does not save the changes made to it and reverts to its original state when the current user relinquishes control.

Essex A grouped release of projects related to OpenStack that came out in April 2012, the fifth release of OpenStack. It included Compute (nova 2012.1), Object Storage (swift 1.4.8), Image (glance), Identity (keystone), and Dashboard (horizon). Essex is the code name for the fifth release of OpenStack. The design summit took place in Boston, Massachusetts, US and Essex is a nearby city.

ESXi An OpenStack-supported hypervisor.

ETag MD5 hash of an object within Object Storage, used to ensure data integrity.

euca2ools A collection of command-line tools for administering VMs; most are compatible with OpenStack.

Eucalyptus Kernel Image (EKI) Used along with an ERI to create an EMI.

Eucalyptus Machine Image (EMI) VM image container format supported by Image service.

Eucalyptus Ramdisk Image (ERI) Used along with an EKI to create an EMI.

evacuate The process of migrating one or all virtual machine (VM) instances from one host to another, compatible with both shared storage live migration and block migration.

exchange Alternative term for a RabbitMQ message exchange.

exchange type A routing algorithm in the Compute RabbitMQ.

exclusive queue Connected to by a direct consumer in RabbitMQCompute, the message can be consumed only by the current connection.

extended attributes (xattr) File system option that enables storage of additional information beyond owner, group, permissions, modification time, and so on. The underlying Object Storage file system must support extended attributes.

extension Alternative term for an API extension or plug-in. In the context of Identity service, this is a call that is specific to the implementation, such as adding support for OpenID.

external network A network segment typically used for instance Internet access.

extra specs Specifies additional requirements when Compute determines where to start a new instance. Examples include a minimum amount of network bandwidth or a GPU.

9.2.7 F

FakeLDAP An easy method to create a local LDAP directory for testing Identity and Compute. Requires Redis.

fan-out exchange Within RabbitMQ and Compute, it is the messaging interface that is used by the scheduler service to receive capability messages from the compute, volume, and network nodes.

federated identity A method to establish trusts between identity providers and the OpenStack cloud.

Fedora A Linux distribution compatible with OpenStack.

Fibre Channel Storage protocol similar in concept to TCP/IP; encapsulates SCSI commands and data.

Fibre Channel over Ethernet (FCoE) The fibre channel protocol tunneled within Ethernet.

fill-first scheduler The Compute scheduling method that attempts to fill a host with VMs rather than starting new VMs on a variety of hosts.

filter The step in the Compute scheduling process when hosts that cannot run VMs are eliminated and not chosen.

firewall Used to restrict communications between hosts and/or nodes, implemented in Compute using iptables, arptables, ip6tables, and ebtables.

FireWall-as-a-Service (FWaaS) A Networking extension that provides perimeter firewall functionality.

fixed IP address An IP address that is associated with the same instance each time that instance boots, is generally not accessible to end users or the public Internet, and is used for management of the instance.

Flat Manager The Compute component that gives IP addresses to authorized nodes and assumes DHCP, DNS, and routing configuration and services are provided by something else.

flat mode injection A Compute networking method where the OS network configuration information is injected into the VM image before the instance starts.

flat network Virtual network type that uses neither VLANs nor tunnels to segregate project traffic. Each flat network typically requires a separate underlying physical interface defined by bridge mappings. However, a flat network can contain multiple subnets.

FlatDHCP Manager The Compute component that provides dnsmasq (DHCP, DNS, BOOTP, TFTP) and radvd (routing) services.

flavor Alternative term for a VM instance type.

flavor ID UUID for each Compute or Image service VM flavor or instance type.

floating IP address An IP address that a project can associate with a VM so that the instance has the same public IP address each time that it boots. You create a pool of floating IP addresses and assign them to instances as they are launched to maintain a consistent IP address for maintaining DNS assignment.

Folsom A grouped release of projects related to OpenStack that came out in the fall of 2012, the sixth release of OpenStack. It includes Compute (nova), Object Storage (swift), Identity (keystone), Networking (neutron), Image service (glance), and Volumes or Block Storage (cinder). Folsom is the code name for the sixth release of OpenStack. The design summit took place in San Francisco, California, US and Folsom is a nearby city.

FormPost Object Storage middleware that uploads (posts) an image through a form on a web page.

freezer Code name for the *Backup, Restore, and Disaster Recovery service*.

front end The point where a user interacts with a service; can be an API endpoint, the dashboard, or a command-line tool.

9.2.8 G

gateway An IP address, typically assigned to a router, that passes network traffic between different networks.

generic receive offload (GRO) Feature of certain network interface drivers that combines many smaller received packets into a large packet before delivery to the kernel IP stack.

generic routing encapsulation (GRE) Protocol that encapsulates a wide variety of network layer protocols inside virtual point-to-point links.

glance Codename for the *Image service*.

glance API server Alternative name for the *Image API*.

glance registry Alternative term for the Image service *image registry*.

global endpoint template The Identity service endpoint template that contains services available to all projects.

GlusterFS A file system designed to aggregate NAS hosts, compatible with OpenStack.

gnocchi Part of the OpenStack *Telemetry service*; provides an indexer and time-series database.

golden image A method of operating system installation where a finalized disk image is created and then used by all nodes without modification.

Governance service (congress) The project that provides Governance-as-a-Service across any collection of cloud services in order to monitor, enforce, and audit policy over dynamic infrastructure.

Graphic Interchange Format (GIF) A type of image file that is commonly used for animated images on web pages.

Graphics Processing Unit (GPU) Choosing a host based on the existence of a GPU is currently unsupported in OpenStack.

Green Threads The cooperative threading model used by Python; reduces race conditions and only context switches when specific library calls are made. Each OpenStack service is its own thread.

Grizzly The code name for the seventh release of OpenStack. The design summit took place in San Diego, California, US and Grizzly is an element of the state flag of California.

Group An Identity v3 API entity. Represents a collection of users that is owned by a specific domain.

guest OS An operating system instance running under the control of a hypervisor.

9.2.9 H

Hadoop Apache Hadoop is an open source software framework that supports data-intensive distributed applications.

Hadoop Distributed File System (HDFS) A distributed, highly fault-tolerant file system designed to run on low-cost commodity hardware.

handover An object state in Object Storage where a new replica of the object is automatically created due to a drive failure.

HAProxy Provides a load balancer for TCP and HTTP-based applications that spreads requests across multiple servers.

hard reboot A type of reboot where a physical or virtual power button is pressed as opposed to a graceful, proper shutdown of the operating system.

Havana The code name for the eighth release of OpenStack. The design summit took place in Portland, Oregon, US and Havana is an unincorporated community in Oregon.

health monitor Determines whether back-end members of a VIP pool can process a request. A pool can have several health monitors associated with it. When a pool has several monitors associated with it, all monitors check each member of the pool. All monitors must declare a member to be healthy for it to stay active.

heat Codename for the *Orchestration service*.

Heat Orchestration Template (HOT) Heat input in the format native to OpenStack.

high availability (HA) A high availability system design approach and associated service implementation ensures that a prearranged level of operational performance will be met during a contractual measurement period. High availability systems seek to minimize system downtime and data loss.

horizon Codename for the *Dashboard*.

horizon plug-in A plug-in for the OpenStack Dashboard (horizon).

host A physical computer, not a VM instance (node).

host aggregate A method to further subdivide availability zones into hypervisor pools, a collection of common hosts.

Host Bus Adapter (HBA) Device plugged into a PCI slot, such as a fibre channel or network card.

hybrid cloud A hybrid cloud is a composition of two or more clouds (private, community or public) that remain distinct entities but are bound together, offering the benefits of multiple deployment models. Hybrid cloud can also mean the ability to connect colocation, managed and/or dedicated services with cloud resources.

hybrid cloud computing A mix of on-premises, private cloud and third-party, public cloud services with orchestration between the two platforms.

Hyper-V One of the hypervisors supported by OpenStack.

hyperlink Any kind of text that contains a link to some other site, commonly found in documents where clicking on a word or words opens up a different website.

Hypertext Transfer Protocol (HTTP) An application protocol for distributed, collaborative, hypermedia information systems. It is the foundation of data communication for the World Wide Web. Hypertext is structured text that uses logical links (hyperlinks) between nodes containing text. HTTP is the protocol to exchange or transfer hypertext.

Hypertext Transfer Protocol Secure (HTTPS) An encrypted communications protocol for secure communication over a computer network, with especially wide deployment on the Internet. Technically, it is not a protocol in and of itself; rather, it is the result of simply layering the Hypertext Transfer Protocol (HTTP) on top of the TLS or SSL protocol, thus adding the security capabilities of TLS or SSL to standard HTTP communications. Most OpenStack API endpoints and many inter-component communications support HTTPS communication.

hypervisor Software that arbitrates and controls VM access to the actual underlying hardware.

hypervisor pool A collection of hypervisors grouped together through host aggregates.

9.2.10 I

Icehouse The code name for the ninth release of OpenStack. The design summit took place in Hong Kong and Ice House is a street in that city.

ID number Unique numeric ID associated with each user in Identity, conceptually similar to a Linux or LDAP UID.

Identity API Alternative term for the Identity service API.

Identity back end The source used by Identity service to retrieve user information; an OpenLDAP server, for example.

identity provider A directory service, which allows users to login with a user name and password. It is a typical source of authentication tokens.

Identity service (keystone) The project that facilitates API client authentication, service discovery, distributed multi-project authorization, and auditing. It provides a central directory of users mapped to the OpenStack services they can access. It also registers endpoints for OpenStack services and acts as a common authentication system.

Identity service API The API used to access the OpenStack Identity service provided through keystone.

IETF Internet Engineering Task Force (IETF) is an open standards organization that develops Internet standards, particularly the standards pertaining to TCP/IP.

image A collection of files for a specific operating system (OS) that you use to create or rebuild a server. OpenStack provides pre-built images. You can also create custom images, or snapshots, from servers that you have launched. Custom images can be used for data backups or as gold images for additional servers.

Image API The Image service API endpoint for management of VM images. Processes client requests for VMs, updates Image service metadata on the registry server, and communicates with the store adapter to upload VM images from the back-end store.

image cache Used by Image service to obtain images on the local host rather than re-downloading them from the image server each time one is requested.

image ID Combination of a URI and UUID used to access Image service VM images through the image API.

image membership A list of projects that can access a given VM image within Image service.

image owner The project who owns an Image service virtual machine image.

image registry A list of VM images that are available through Image service.

Image service (glance) The OpenStack service that provide services and associated libraries to store, browse, share, distribute and manage bootable disk images, other data closely associated with initializing compute resources, and metadata definitions.

image status The current status of a VM image in Image service, not to be confused with the status of a running instance.

image store The back-end store used by Image service to store VM images, options include Object Storage, locally mounted file system, RADOS block devices, VMware datastore, or HTTP.

image UUID UUID used by Image service to uniquely identify each VM image.

incubated project A community project may be elevated to this status and is then promoted to a core project.

Infrastructure Optimization service (watcher) OpenStack project that aims to provide a flexible and scalable resource optimization service for multi-project OpenStack-based clouds.

Infrastructure-as-a-Service (IaaS) IaaS is a provisioning model in which an organization outsources physical components of a data center, such as storage, hardware, servers, and networking components. A service provider owns the equipment and is responsible for housing, operating and maintaining it. The client typically pays on a per-use basis. IaaS is a model for providing cloud services.

ingress filtering The process of filtering incoming network traffic. Supported by Compute.

INI format The OpenStack configuration files use an INI format to describe options and their values. It consists of sections and key value pairs.

injection The process of putting a file into a virtual machine image before the instance is started.

Input/Output Operations Per Second (IOPS) IOPS are a common performance measurement used to benchmark computer storage devices like hard disk drives, solid state drives, and storage area networks.

instance A running VM, or a VM in a known state such as suspended, that can be used like a hardware server.

instance ID Alternative term for instance UUID.

instance state The current state of a guest VM image.

instance tunnels network A network segment used for instance traffic tunnels between compute nodes and the network node.

instance type Describes the parameters of the various virtual machine images that are available to users; includes parameters such as CPU, storage, and memory. Alternative term for flavor.

instance type ID Alternative term for a flavor ID.

instance UUID Unique ID assigned to each guest VM instance.

Intelligent Platform Management Interface (IPMI) IPMI is a standardized computer system interface used by system administrators for out-of-band management of computer systems and monitoring of their operation. In laymans terms, it is a way to manage a computer using a direct network connection, whether it is turned on or not; connecting to the hardware rather than an operating system or login shell.

interface A physical or virtual device that provides connectivity to another device or medium.

interface ID Unique ID for a Networking VIF or vNIC in the form of a UUID.

Internet Control Message Protocol (ICMP) A network protocol used by network devices for control messages. For example, `ping` uses ICMP to test connectivity.

Internet protocol (IP) Principal communications protocol in the internet protocol suite for relaying datagrams across network boundaries.

Internet Service Provider (ISP) Any business that provides Internet access to individuals or businesses.

Internet Small Computer System Interface (iSCSI) Storage protocol that encapsulates SCSI frames for transport over IP networks. Supported by Compute, Object Storage, and Image service.

IO The abbreviation for input and output.

IP address Number that is unique to every computer system on the Internet. Two versions of the Internet Protocol (IP) are in use for addresses: IPv4 and IPv6.

IP Address Management (IPAM) The process of automating IP address allocation, deallocation, and management. Currently provided by Compute, melange, and Networking.

ip6tables Tool used to set up, maintain, and inspect the tables of IPv6 packet filter rules in the Linux kernel. In OpenStack Compute, ip6tables is used along with arptables, ebtables, and iptables to create firewalls for both nodes and VMs.

ipset Extension to iptables that allows creation of firewall rules that match entire sets of IP addresses simultaneously. These sets reside in indexed data structures to increase efficiency, particularly on systems with a large quantity of rules.

iptables Used along with arptables and ebtables, iptables create firewalls in Compute. iptables are the tables provided by the Linux kernel firewall (implemented as different Netfilter modules) and the chains and rules it stores. Different kernel modules and programs are currently used for different protocols: iptables applies to IPv4, ip6tables to IPv6, arptables to ARP, and ebtables to Ethernet frames. Requires root privilege to manipulate.

ironic Codename for the *Bare Metal service*.

iSCSI Qualified Name (IQN) IQN is the format most commonly used for iSCSI names, which uniquely identify nodes in an iSCSI network. All IQNs follow the pattern iqn.yyyy-mm.domain:identifier, where yyyy-mm is the year and month in which the domain was registered, domain is the reversed domain name of the issuing organization, and identifier is an optional string which makes each IQN under the same domain unique. For example, iqn.2015-10.org.openstack.408ae959bce1.

ISO9660 One of the VM image disk formats supported by Image service.

itsec A default role in the Compute RBAC system that can quarantine an instance in any project.

9.2.11 J

Java A programming language that is used to create systems that involve more than one computer by way of a network.

JavaScript A scripting language that is used to build web pages.

JavaScript Object Notation (JSON) One of the supported response formats in OpenStack.

jumbo frame Feature in modern Ethernet networks that supports frames up to approximately 9000 bytes.

Juno The code name for the tenth release of OpenStack. The design summit took place in Atlanta, Georgia, US and Juno is an unincorporated community in Georgia.

9.2.12 K

Kerberos A network authentication protocol which works on the basis of tickets. Kerberos allows nodes communication over a non-secure network, and allows nodes to prove their identity to one another in a secure manner.

kernel-based VM (KVM) An OpenStack-supported hypervisor. KVM is a full virtualization solution for Linux on x86 hardware containing virtualization extensions (Intel VT or AMD-V), ARM, IBM Power, and IBM zSeries. It consists of a loadable kernel module, that provides the core virtualization infrastructure and a processor specific module.

Key Manager service (barbican) The project that produces a secret storage and generation system capable of providing key management for services wishing to enable encryption features.

keystone Codename of the *Identity service*.

Kickstart A tool to automate system configuration and installation on Red Hat, Fedora, and CentOS-based Linux distributions.

Kilo The code name for the eleventh release of OpenStack. The design summit took place in Paris, France. Due to delays in the name selection, the release was known only as K. Because `k` is the unit symbol for kilo and the kilogram reference artifact is stored near Paris in the Pavillon de Breteuil in Sèvres, the community chose Kilo as the release name.

9.2.13 L

large object An object within Object Storage that is larger than 5ăGB.

Launchpad The collaboration site for OpenStack.

Layer-2 (L2) agent OpenStack Networking agent that provides layer-2 connectivity for virtual networks.

Layer-2 network Term used in the OSI network architecture for the data link layer. The data link layer is responsible for media access control, flow control and detecting and possibly correcting errors that may occur in the physical layer.

Layer-3 (L3) agent OpenStack Networking agent that provides layer-3 (routing) services for virtual networks.

Layer-3 network Term used in the OSI network architecture for the network layer. The network layer is responsible for packet forwarding including routing from one node to another.

Liberty The code name for the twelfth release of OpenStack. The design summit took place in Vancouver, Canada and Liberty is the name of a village in the Canadian province of Saskatchewan.

libvirt Virtualization API library used by OpenStack to interact with many of its supported hypervisors.

Lightweight Directory Access Protocol (LDAP) An application protocol for accessing and maintaining distributed directory information services over an IP network.

Linux Unix-like computer operating system assembled under the model of free and open-source software development and distribution.

Linux bridge Software that enables multiple VMs to share a single physical NIC within Compute.

Linux Bridge neutron plug-in Enables a Linux bridge to understand a Networking port, interface attachment, and other abstractions.

Linux containers (LXC) An OpenStack-supported hypervisor.

live migration The ability within Compute to move running virtual machine instances from one host to another with only a small service interruption during switchover.

load balancer A load balancer is a logical device that belongs to a cloud account. It is used to distribute workloads between multiple back-end systems or services, based on the criteria defined as part of its configuration.

load balancing The process of spreading client requests between two or more nodes to improve performance and availability.

Load-Balancer-as-a-Service (LBaaS) Enables Networking to distribute incoming requests evenly between designated instances.

Load-balancing service (octavia) The project that aims to provide scalable, on demand, self service access to load-balancer services, in technology-agnostic manner.

Logical Volume Manager (LVM) Provides a method of allocating space on mass-storage devices that is more flexible than conventional partitioning schemes.

9.2.14 M

magnum Code name for the *Containers Infrastructure Management service*.

management API Alternative term for an admin API.

management network A network segment used for administration, not accessible to the public Internet.

manager Logical groupings of related code, such as the Block Storage volume manager or network manager.

manifest Used to track segments of a large object within Object Storage.

manifest object A special Object Storage object that contains the manifest for a large object.

manila Codename for OpenStack *Shared File Systems service*.

manila-share Responsible for managing Shared File System Service devices, specifically the back-end devices.

maximum transmission unit (MTU) Maximum frame or packet size for a particular network medium. Typically 1500 bytes for Ethernet networks.

mechanism driver A driver for the Modular Layer 2 (ML2) neutron plug-in that provides layer-2 connectivity for virtual instances. A single OpenStack installation can use multiple mechanism drivers.

melange Project name for OpenStack Network Information Service. To be merged with Networking.

membership The association between an Image service VM image and a project. Enables images to be shared with specified projects.

membership list A list of projects that can access a given VM image within Image service.

memcached A distributed memory object caching system that is used by Object Storage for caching.

memory overcommit The ability to start new VM instances based on the actual memory usage of a host, as opposed to basing the decision on the amount of RAM each running instance thinks it has available. Also known as RAM overcommit.

message broker The software package used to provide AMQP messaging capabilities within Compute. Default package is RabbitMQ.

message bus The main virtual communication line used by all AMQP messages for inter-cloud communications within Compute.

message queue Passes requests from clients to the appropriate workers and returns the output to the client after the job completes.

Message service (zaqar) The project that provides a messaging service that affords a variety of distributed application patterns in an efficient, scalable and highly available manner, and to create and maintain associated Python libraries and documentation.

Meta-Data Server (MDS) Stores CephFS metadata.

Metadata agent OpenStack Networking agent that provides metadata services for instances.

migration The process of moving a VM instance from one host to another.

mistral Code name for *Workflow service*.

Mitaka The code name for the thirteenth release of OpenStack. The design summit took place in Tokyo, Japan. Mitaka is a city in Tokyo.

Modular Layer 2 (ML2) neutron plug-in Can concurrently use multiple layer-2 networking technologies, such as 802.1Q and VXLAN, in Networking.

monasca Codename for OpenStack *Monitoring*.

Monitor (LBaaS) LBaaS feature that provides availability monitoring using the `ping` command, TCP, and HTTP/HTTPS GET.

Monitor (Mon) A Ceph component that communicates with external clients, checks data state and consistency, and performs quorum functions.

Monitoring (monasca) The OpenStack service that provides a multi-project, highly scalable, performant, fault-tolerant monitoring-as-a-service solution for metrics, complex event processing and logging. To build an extensible platform for advanced monitoring services that can be used by

both operators and projects to gain operational insight and visibility, ensuring availability and stability.

multi-cloud computing The use of multiple cloud computing and storage services in a single network architecture.

multi-cloud SDKs SDKs that provide a multi-cloud abstraction layer and include support for OpenStack. These SDKs are excellent for writing applications that need to consume more than one type of cloud provider, but may expose a more limited set of features.

multi-factor authentication Authentication method that uses two or more credentials, such as a password and a private key. Currently not supported in Identity.

multi-host High-availability mode for legacy (nova) networking. Each compute node handles NAT and DHCP and acts as a gateway for all of the VMs on it. A networking failure on one compute node doesnt affect VMs on other compute nodes.

multinic Facility in Compute that allows each virtual machine instance to have more than one VIF connected to it.

murano Codename for the *Application Catalog service*.

9.2.15 N

Nebula Released as open source by NASA in 2010 and is the basis for Compute.

netadmin One of the default roles in the Compute RBAC system. Enables the user to allocate publicly accessible IP addresses to instances and change firewall rules.

NetApp volume driver Enables Compute to communicate with NetApp storage devices through the NetApp OnCommand Provisioning Manager.

network A virtual network that provides connectivity between entities. For example, a collection of virtual ports that share network connectivity. In Networking terminology, a network is always a layer-2 network.

Network Address Translation (NAT) Process of modifying IP address information while in transit. Supported by Compute and Networking.

network controller A Compute daemon that orchestrates the network configuration of nodes, including IP addresses, VLANs, and bridging. Also manages routing for both public and private networks.

Network File System (NFS) A method for making file systems available over the network. Supported by OpenStack.

network ID Unique ID assigned to each network segment within Networking. Same as network UUID.

network manager The Compute component that manages various network components, such as firewall rules, IP address allocation, and so on.

network namespace Linux kernel feature that provides independent virtual networking instances on a single host with separate routing tables and interfaces. Similar to virtual routing and forwarding (VRF) services on physical network equipment.

network node Any compute node that runs the network worker daemon.

network segment Represents a virtual, isolated OSI layer-2 subnet in Networking.

Network Service Header (NSH) Provides a mechanism for metadata exchange along the instantiated service path.

Network Time Protocol (NTP) Method of keeping a clock for a host or node correct via communication with a trusted, accurate time source.

network UUID Unique ID for a Networking network segment.

network worker The `nova-network` worker daemon; provides services such as giving an IP address to a booting nova instance.

Networking API (Neutron API) API used to access OpenStack Networking. Provides an extensible architecture to enable custom plug-in creation.

Networking service (neutron) The OpenStack project which implements services and associated libraries to provide on-demand, scalable, and technology-agnostic network abstraction.

neutron Codename for OpenStack *Networking service*.

neutron API An alternative name for *Networking API*.

neutron manager Enables Compute and Networking integration, which enables Networking to perform network management for guest VMs.

neutron plug-in Interface within Networking that enables organizations to create custom plug-ins for advanced features, such as QoS, ACLs, or IDS.

Newton The code name for the fourteenth release of OpenStack. The design summit took place in Austin, Texas, US. The release is named after Newton House which is located at 1013 E. Ninth St., Austin, TX. which is listed on the National Register of Historic Places.

Nexenta volume driver Provides support for NexentaStor devices in Compute.

NFV Orchestration service (tacker) OpenStack service that aims to implement Network Function Virtualization (NFV) orchestration services and libraries for end-to-end life-cycle management of network services and Virtual Network Functions (VNFs).

Nginx An HTTP and reverse proxy server, a mail proxy server, and a generic TCP/UDP proxy server.

No ACK Disables server-side message acknowledgment in the Compute RabbitMQ. Increases performance but decreases reliability.

node A VM instance that runs on a host.

non-durable exchange Message exchange that is cleared when the service restarts. Its data is not written to persistent storage.

non-durable queue Message queue that is cleared when the service restarts. Its data is not written to persistent storage.

non-persistent volume Alternative term for an ephemeral volume.

north-south traffic Network traffic between a user or client (north) and a server (south), or traffic into the cloud (south) and out of the cloud (north). See also east-west traffic.

nova Codename for OpenStack *Compute service*.

Nova API Alternative term for the *Compute API*.

nova-network A Compute component that manages IP address allocation, firewalls, and other network-related tasks. This is the legacy networking option and an alternative to Networking.

9.2.16 O

object A BLOB of data held by Object Storage; can be in any format.

object auditor Opens all objects for an object server and verifies the MD5 hash, size, and metadata for each object.

object expiration A configurable option within Object Storage to automatically delete objects after a specified amount of time has passed or a certain date is reached.

object hash Unique ID for an Object Storage object.

object path hash Used by Object Storage to determine the location of an object in the ring. Maps objects to partitions.

object replicator An Object Storage component that copies an object to remote partitions for fault tolerance.

object server An Object Storage component that is responsible for managing objects.

Object Storage API API used to access OpenStack *Object Storage*.

Object Storage Device (OSD) The Ceph storage daemon.

Object Storage service (swift) The OpenStack core project that provides eventually consistent and redundant storage and retrieval of fixed digital content.

object versioning Allows a user to set a flag on an *Object Storage* container so that all objects within the container are versioned.

Ocata The code name for the fifteenth release of OpenStack. The design summit took place in Barcelona, Spain. Ocata is a beach north of Barcelona.

Octavia Code name for the *Load-balancing service*.

Oldie Term for an *Object Storage* process that runs for a long time. Can indicate a hung process.

Open Cloud Computing Interface (OCCI) A standardized interface for managing compute, data, and network resources, currently unsupported in OpenStack.

Open Virtualization Format (OVF) Standard for packaging VM images. Supported in OpenStack.

Open vSwitch Open vSwitch is a production quality, multilayer virtual switch licensed under the open source Apache 2.0 license. It is designed to enable massive network automation through programmatic extension, while still supporting standard management interfaces and protocols (for example NetFlow, sFlow, SPAN, RSPAN, CLI, LACP, 802.1ag).

Open vSwitch (OVS) agent Provides an interface to the underlying Open vSwitch service for the Networking plug-in.

Open vSwitch neutron plug-in Provides support for Open vSwitch in Networking.

OpenDev OpenDev is a space for collaborative Open Source software development.

OpenDevs mission is to provide project hosting, continuous integration tooling, and virtual collaboration spaces for Open Source software projects. OpenDev is itself self hosted on this set of tools including code review, continuous integration, etherpad, wiki, code browsing and so on. This means that OpenDev itself is run like an open source project, you can join us and help run the system. Additionally, all of the services run are Open Source software themselves.

The OpenStack project is the largest project using OpenDev.

OpenLDAP An open source LDAP server. Supported by both Compute and Identity.

OpenStack OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a data center, all managed through a dashboard that gives administrators control while empowering their users to provision resources through a web interface. OpenStack is an open source project licensed under the Apache License 2.0.

OpenStack code name Each OpenStack release has a code name. Code names ascend in alphabetical order: Austin, Bexar, Cactus, Diablo, Essex, Folsom, Grizzly, Havana, Icehouse, Juno, Kilo, Liberty, Mitaka, Newton, Ocata, Pike, Queens, Rocky, Stein, Train, Ussuri, Victoria, and Wallaby.

Wallaby was the first code name choosen by a new policy: Code names are choosen by the community following the alphabet, for details see release name criteria.

The Victoria name was the last name where code names are cities or counties near where the corresponding OpenStack design summit took place. An exception, called the Waldon exception, was granted to elements of the state flag that sound especially cool. Code names are chosen by popular vote.

openSUSE A Linux distribution that is compatible with OpenStack.

operator The person responsible for planning and maintaining an OpenStack installation.

optional service An official OpenStack service defined as optional by DefCore Committee. Currently, consists of Dashboard (horizon), Telemetry service (Telemetry), Orchestration service (heat), Database service (trove), Bare Metal service (ironic), and so on.

Orchestration service (heat) The OpenStack service which orchestrates composite cloud applications using a declarative template format through an OpenStack-native REST API.

orphan In the context of Object Storage, this is a process that is not terminated after an upgrade, restart, or reload of the service.

Oslo Codename for the *Common Libraries project*.

9.2.17 P

panko Part of the OpenStack *Telemetry service*; provides event storage.

parent cell If a requested resource, such as CPU time, disk storage, or memory, is not available in the parent cell, the request is forwarded to associated child cells.

partition A unit of storage within Object Storage used to store objects. It exists on top of devices and is replicated for fault tolerance.

partition index Contains the locations of all Object Storage partitions within the ring.

partition shift value Used by Object Storage to determine which partition data should reside on.

path MTU discovery (PMTUD) Mechanism in IP networks to detect end-to-end MTU and adjust packet size accordingly.

pause A VM state where no changes occur (no changes in memory, network communications stop, etc); the VM is frozen but not shut down.

PCI passthrough Gives guest VMs exclusive access to a PCI device. Currently supported in OpenStack Havana and later releases.

persistent message A message that is stored both in memory and on disk. The message is not lost after a failure or restart.

persistent volume Changes to these types of disk volumes are saved.

personality file A file used to customize a Compute instance. It can be used to inject SSH keys or a specific network configuration.

Pike The code name for the sixteenth release of OpenStack. The OpenStack summit took place in Boston, Massachusetts, US. The release is named after the Massachusetts Turnpike, abbreviated commonly as the Mass Pike, which is the easternmost stretch of Interstate 90.

Platform-as-a-Service (PaaS) Provides to the consumer an operating system and, often, a language runtime and libraries (collectively, the platform) upon which they can run their own application code, without providing any control over the underlying infrastructure. Examples of Platform-as-a-Service providers include Cloud Foundry and OpenShift.

plug-in Software component providing the actual implementation for Networking APIs, or for Compute APIs, depending on the context.

policy service Component of Identity that provides a rule-management interface and a rule-based authorization engine.

policy-based routing (PBR) Provides a mechanism to implement packet forwarding and routing according to the policies defined by the network administrator.

pool A logical set of devices, such as web servers, that you group together to receive and process traffic. The load balancing function chooses which member of the pool handles the new requests or connections received on the VIP address. Each VIP has one pool.

pool member An application that runs on the back-end server in a load-balancing system.

port A virtual network port within Networking; VIFs / vNICs are connected to a port.

port UUID Unique ID for a Networking port.

preseed A tool to automate system configuration and installation on Debian-based Linux distributions.

private cloud Computing resources used exclusively by one business or organization.

private image An Image service VM image that is only available to specified projects.

private IP address An IP address used for management and administration, not available to the public Internet.

private network The Network Controller provides virtual networks to enable compute servers to interact with each other and with the public network. All machines must have a public and private network interface. A private network interface can be a flat or VLAN network interface. A flat network interface is controlled by the flat_interface with flat managers. A VLAN network interface is controlled by the `vlan_interface` option with VLAN managers.

project Projects represent the base unit of ownership in OpenStack, in that all resources in OpenStack should be owned by a specific project. In OpenStack Identity, a project must be owned by a specific domain.

project ID Unique ID assigned to each project by the Identity service.

project VPN Alternative term for a cloudpipe.

promiscuous mode Causes the network interface to pass all traffic it receives to the host rather than passing only the frames addressed to it.

protected property Generally, extra properties on an Image service image to which only cloud administrators have access. Limits which user roles can perform CRUD operations on that property. The cloud administrator can configure any image property as protected.

provider An administrator who has access to all hosts and instances.

proxy node A node that provides the Object Storage proxy service.

proxy server Users of Object Storage interact with the service through the proxy server, which in turn looks up the location of the requested data within the ring and returns the results to the user.

public API An API endpoint used for both service-to-service communication and end-user interactions.

public cloud Data centers available to many users over the Internet.

public image An Image service VM image that is available to all projects.

public IP address An IP address that is accessible to end-users.

public key authentication Authentication method that uses keys rather than passwords.

public network The Network Controller provides virtual networks to enable compute servers to interact with each other and with the public network. All machines must have a public and private network interface. The public network interface is controlled by the `public_interface` option.

Puppet An operating system configuration-management tool supported by OpenStack.

Python Programming language used extensively in OpenStack.

9.2.18 Q

QEMU Copy On Write 2 (QCOW2) One of the VM image disk formats supported by Image service.

Qpid Message queue software supported by OpenStack; an alternative to RabbitMQ.

Quality of Service (QoS) The ability to guarantee certain network or storage requirements to satisfy a Service Level Agreement (SLA) between an application provider and end users. Typically includes performance requirements like networking bandwidth, latency, jitter correction, and reliability as well as storage performance in Input/Output Operations Per Second (IOPS), throttling agreements, and performance expectations at peak load.

quarantine If Object Storage finds objects, containers, or accounts that are corrupt, they are placed in this state, are not replicated, cannot be read by clients, and a correct copy is re-replicated.

Queens The code name for the seventeenth release of OpenStack. The OpenStack summit took place in Sydney, Australia. The release is named after the Queens Pound river in the South Coast region of New South Wales.

Quick EMUlator (QEMU) QEMU is a generic and open source machine emulator and virtualizer. One of the hypervisors supported by OpenStack, generally used for development purposes.

quota In Compute and Block Storage, the ability to set resource limits on a per-project basis.

9.2.19 R

RabbitMQ The default message queue software used by OpenStack.

Rackspace Cloud Files Released as open source by Rackspace in 2010; the basis for Object Storage.

RADOS Block Device (RBD) Ceph component that enables a Linux block device to be striped over multiple distributed data stores.

radvd The router advertisement daemon, used by the Compute VLAN manager and FlatDHCP manager to provide routing services for VM instances.

rally Codename for the *Benchmark service*.

RAM filter The Compute setting that enables or disables RAM overcommitment.

RAM overcommit The ability to start new VM instances based on the actual memory usage of a host, as opposed to basing the decision on the amount of RAM each running instance thinks it has available. Also known as memory overcommit.

rate limit Configurable option within Object Storage to limit database writes on a per-account and/or per-container basis.

raw One of the VM image disk formats supported by Image service; an unstructured disk image.

rebalance The process of distributing Object Storage partitions across all drives in the ring; used during initial ring creation and after ring reconfiguration.

reboot Either a soft or hard reboot of a server. With a soft reboot, the operating system is signaled to restart, which enables a graceful shutdown of all processes. A hard reboot is the equivalent of power cycling the server. The virtualization platform should ensure that the reboot action has completed successfully, even in cases in which the underlying domain/VM is paused or halted/stopped.

rebuild Removes all data on the server and replaces it with the specified image. Server ID and IP addresses remain the same.

Recon An Object Storage component that collects meters.

record Belongs to a particular domain and is used to specify information about the domain. There are several types of DNS records. Each record type contains particular information used to describe the purpose of that record. Examples include mail exchange (MX) records, which specify the mail server for a particular domain; and name server (NS) records, which specify the authoritative name servers for a domain.

record ID A number within a database that is incremented each time a change is made. Used by Object Storage when replicating.

Red Hat Enterprise Linux (RHEL) A Linux distribution that is compatible with OpenStack.

reference architecture A recommended architecture for an OpenStack cloud.

region A discrete OpenStack environment with dedicated API endpoints that typically shares only the Identity (keystone) with other regions.

registry Alternative term for the Image service registry.

registry server An Image service that provides VM image metadata information to clients.

Reliable, Autonomic Distributed Object Store (RADOS)

A collection of components that provides object storage within Ceph. Similar to OpenStack Object Storage.

Remote Procedure Call (RPC) The method used by the Compute RabbitMQ for intra-service communications.

replica Provides data redundancy and fault tolerance by creating copies of Object Storage objects, accounts, and containers so that they are not lost when the underlying storage fails.

replica count The number of replicas of the data in an Object Storage ring.

replication The process of copying data to a separate physical device for fault tolerance and performance.

replicator The Object Storage back-end process that creates and manages object replicas.

request ID Unique ID assigned to each request sent to Compute.

rescue image A special type of VM image that is booted when an instance is placed into rescue mode. Allows an administrator to mount the file systems for an instance to correct the problem.

resize Converts an existing server to a different flavor, which scales the server up or down. The original server is saved to enable rollback if a problem occurs. All resizes must be tested and explicitly confirmed, at which time the original server is removed.

RESTful A kind of web service API that uses REST, or Representational State Transfer. REST is the style of architecture for hypermedia systems that is used for the World Wide Web.

ring An entity that maps Object Storage data to partitions. A separate ring exists for each service, such as account, object, and container.

ring builder Builds and manages rings within Object Storage, assigns partitions to devices, and pushes the configuration to other storage nodes.

Rocky The code name for the eightteenth release of OpenStack. The OpenStack summit took place in Vancouver, Canada. The release is named after the Rocky Mountains.

role A personality that a user assumes to perform a specific set of operations. A role includes a set of rights and privileges. A user assuming that role inherits those rights and privileges.

Role Based Access Control (RBAC) Provides a predefined list of actions that the user can perform, such as start or stop VMs, reset passwords, and so on. Supported in both Identity and Compute and can be configured using the dashboard.

role ID Alphanumeric ID assigned to each Identity service role.

Root Cause Analysis (RCA) service (Vitrage) OpenStack project that aims to organize, analyze and visualize OpenStack alarms and events, yield insights regarding the root cause of problems and deduce their existence before they are directly detected.

rootwrap A feature of Compute that allows the unprivileged nova user to run a specified list of commands as the Linux root user.

round-robin scheduler Type of Compute scheduler that evenly distributes instances among available hosts.

router A physical or virtual network device that passes network traffic between different networks.

routing key The Compute direct exchanges, fanout exchanges, and topic exchanges use this key to determine how to process a message; processing varies depending on exchange type.

RPC driver Modular system that allows the underlying message queue software of Compute to be changed. For example, from RabbitMQ to ZeroMQ or Qpid.

rsync Used by Object Storage to push object replicas.

RXTX cap Absolute limit on the amount of network traffic a Compute VM instance can send and receive.

RXTX quota Soft limit on the amount of network traffic a Compute VM instance can send and receive.

9.2.20 S

sahara Codename for the *Data Processing service*.

SAML assertion Contains information about a user as provided by the identity provider. It is an indication that a user has been authenticated.

Sandbox A virtual space in which new or untested software can be run securely.

scheduler manager A Compute component that determines where VM instances should start. Uses modular design to support a variety of scheduler types.

scoped token An Identity service API access token that is associated with a specific project.

scrubber Checks for and deletes unused VMs; the component of Image service that implements delayed delete.

secret key String of text known only by the user; used along with an access key to make requests to the Compute API.

secure boot Process whereby the system firmware validates the authenticity of the code involved in the boot process.

secure shell (SSH) Open source tool used to access remote hosts through an encrypted communications channel, SSH key injection is supported by Compute.

security group A set of network traffic filtering rules that are applied to a Compute instance.

segmented object An Object Storage large object that has been broken up into pieces. The reassembled object is called a concatenated object.

self-service For IaaS, ability for a regular (non-privileged) account to manage a virtual infrastructure component such as networks without involving an administrator.

SELinux Linux kernel security module that provides the mechanism for supporting access control policies.

senlin Code name for the *Clustering service*.

server Computer that provides explicit services to the client software running on that system, often managing a variety of computer operations. A server is a VM instance in the Compute system. Flavor and image are requisite elements when creating a server.

server image Alternative term for a VM image.

server UUID Unique ID assigned to each guest VM instance.

service An OpenStack service, such as Compute, Object Storage, or Image service. Provides one or more endpoints through which users can access resources and perform operations.

service catalog Alternative term for the Identity service catalog.

Service Function Chain (SFC) For a given service, SFC is the abstracted view of the required service functions and the order in which they are to be applied.

service ID Unique ID assigned to each service that is available in the Identity service catalog.

Service Level Agreement (SLA) Contractual obligations that ensure the availability of a service.

service project Special project that contains all services that are listed in the catalog.

service provider A system that provides services to other system entities. In case of federated identity, OpenStack Identity is the service provider.

service registration An Identity service feature that enables services, such as Compute, to automatically register with the catalog.

service token An administrator-defined token used by Compute to communicate securely with the Identity service.

session back end The method of storage used by horizon to track client sessions, such as local memory, cookies, a database, or memcached.

session persistence A feature of the load-balancing service. It attempts to force subsequent connections to a service to be redirected to the same node as long as it is online.

session storage A horizon component that stores and tracks client session information. Implemented through the Django sessions framework.

share A remote, mountable file system in the context of the *Shared File Systems service*. You can mount a share to, and access a share from, several hosts by several users at a time.

share network An entity in the context of the *Shared File Systems service* that encapsulates interaction with the Networking service. If the driver you selected runs in the mode requiring such kind of interaction, you need to specify the share network to create a share.

Shared File Systems API A Shared File Systems service that provides a stable RESTful API. The service authenticates and routes requests throughout the Shared File Systems service. There is python-manilaclient to interact with the API.

Shared File Systems service (manila) The service that provides a set of services for management of shared file systems in a multi-project cloud environment, similar to how OpenStack provides block-based storage management through the OpenStack *Block Storage service* project. With the Shared File Systems service, you can create a remote file system and mount the file system on your instances. You can also read and write data from your instances to and from your file system.

shared IP address An IP address that can be assigned to a VM instance within the shared IP group. Public IP addresses can be shared across multiple servers for use in various high-availability scenarios. When an IP address is shared to another server, the cloud network restrictions are modified to enable each server to listen to and respond on that IP address. You can optionally specify that the target server network configuration be modified. Shared IP addresses can be used with many standard heartbeat facilities, such as keepalive, that monitor for failure and manage IP failover.

shared IP group A collection of servers that can share IPs with other members of the group. Any server in a group can share one or more public IPs with any other server in the group. With the exception of the first server in a shared IP group, servers must be launched into shared IP groups. A server may be a member of only one shared IP group.

shared storage Block storage that is simultaneously accessible by multiple clients, for example, NFS.

Sheepdog Distributed block storage system for QEMU, supported by OpenStack.

Simple Cloud Identity Management (SCIM) Specification for managing identity in the cloud, currently unsupported by OpenStack.

Simple Protocol for Independent Computing Environments (SPICE) SPICE provides remote desktop access to guest virtual machines. It is an alternative to VNC. SPICE is supported by OpenStack.

Single-root I/O Virtualization (SR-IOV) A specification that, when implemented by a physical PCIe device, enables it to appear as multiple separate PCIe devices. This enables multiple virtualized guests to share direct access to the physical device, offering improved performance over an equivalent virtual device. Currently supported in OpenStack Havana and later releases.

SmokeStack Runs automated tests against the core OpenStack API; written in Rails.

snapshot A point-in-time copy of an OpenStack storage volume or image. Use storage volume snapshots to back up volumes. Use image snapshots to back up data, or as gold images for additional servers.

soft reboot A controlled reboot where a VM instance is properly restarted through operating system commands.

Software Development Kit (SDK) Contains code, examples, and documentation that you use to create applications in the language of your choice.

Software Development Lifecycle Automation service (solum) OpenStack project that aims to make cloud services easier to consume and integrate with application development process by automating the source-to-image process, and simplifying app-centric deployment.

Software-defined networking (SDN) Provides an approach for network administrators to manage computer network services through abstraction of lower-level functionality.

SolidFire Volume Driver The Block Storage driver for the SolidFire iSCSI storage appliance.

solum Code name for the *Software Development Lifecycle Automation service*.

spread-first scheduler The Compute VM scheduling algorithm that attempts to start a new VM on the host with the least amount of load.

SQLAlchemy An open source SQL toolkit for Python, used in OpenStack.

SQLite A lightweight SQL database, used as the default persistent storage method in many OpenStack services.

stack A set of OpenStack resources created and managed by the Orchestration service according to a given template (either an AWS CloudFormation template or a Heat Orchestration Template (HOT)).

StackTach Community project that captures Compute AMQP communications; useful for debugging.

static IP address Alternative term for a fixed IP address.

StaticWeb WSGI middleware component of Object Storage that serves container data as a static web page.

Stein The code name for the nineteenth release of OpenStack. The OpenStack Summit took place in Berlin, Germany. The release is named after the street SteinstraSSe in Berlin.

storage back end The method that a service uses for persistent storage, such as iSCSI, NFS, or local disk.

storage manager A XenAPI component that provides a pluggable interface to support a wide variety of persistent storage back ends.

storage manager back end A persistent storage method supported by XenAPI, such as iSCSI or NFS.

storage node An Object Storage node that provides container services, account services, and object services; controls the account databases, container databases, and object storage.

storage services Collective name for the Object Storage object services, container services, and account services.

strategy Specifies the authentication source used by Image service or Identity. In the Database service, it refers to the extensions implemented for a data store.

subdomain A domain within a parent domain. Subdomains cannot be registered. Subdomains enable you to delegate domains. Subdomains can themselves have subdomains, so third-level, fourth-level, fifth-level, and deeper levels of nesting are possible.

subnet Logical subdivision of an IP network.

SUSE Linux Enterprise Server (SLES) A Linux distribution that is compatible with OpenStack.

suspend The VM instance is paused and its state is saved to disk of the host.

swap Disk-based virtual memory used by operating systems to provide more memory than is actually available on the system.

swift Codename for OpenStack *Object Storage service*.

swift All in One (SAIO) Creates a full Object Storage development environment within a single VM.

swift middleware Collective term for Object Storage components that provide additional functionality.

swift proxy server Acts as the gatekeeper to Object Storage and is responsible for authenticating the user.

swift storage node A node that runs Object Storage account, container, and object services.

sync point Point in time since the last container and accounts database sync among nodes within Object Storage.

sysadmin One of the default roles in the Compute RBAC system. Enables a user to add other users to a project, interact with VM images that are associated with the project, and start and stop VM instances.

system usage A Compute component that, along with the notification system, collects meters and usage information. This information can be used for billing.

9.2.21 T

tacker Code name for the *NFV Orchestration service*

Telemetry service (telemetry) The OpenStack project which collects measurements of the utilization of the physical and virtual resources comprising deployed clouds, persists this data for subsequent retrieval and analysis, and triggers actions when defined criteria are met.

TempAuth An authentication facility within Object Storage that enables Object Storage itself to perform authentication and authorization. Frequently used in testing and development.

Tempest Automated software test suite designed to run against the trunk of the OpenStack core project.

TempURL An Object Storage middleware component that enables creation of URLs for temporary object access.

tenant A group of users; used to isolate access to Compute resources. An alternative term for a project.

Tenant API An API that is accessible to projects.

tenant endpoint An Identity service API endpoint that is associated with one or more projects.

tenant ID An alternative term for *project ID*.

token An alpha-numeric string of text used to access OpenStack APIs and resources.

token services An Identity service component that manages and validates tokens after a user or project has been authenticated.

tombstone Used to mark Object Storage objects that have been deleted; ensures that the object is not updated on another node after it has been deleted.

topic publisher A process that is created when a RPC call is executed; used to push the message to the topic exchange.

Torpedo Community project used to run automated tests against the OpenStack API.

Train The code name for the twentieth release of OpenStack. The OpenStack Infrastructure Summit took place in Denver, Colorado, US.

Two Project Team Gathering meetings in Denver were held at a hotel next to the train line from downtown to the airport. The crossing signals there had some sort of malfunction in the past causing them to not stop the cars when a train was coming properly. As a result the trains were required to blow their horns when passing through that area. Obviously staying in a hotel, by trains that are blowing their horns 24/7 was less than ideal. As a result, many jokes popped up about Denver and trains - and thus the release is called train.

transaction ID Unique ID assigned to each Object Storage request; used for debugging and tracing.

transient Alternative term for non-durable.

transient exchange Alternative term for a non-durable exchange.

transient message A message that is stored in memory and is lost after the server is restarted.

transient queue Alternative term for a non-durable queue.

TripleO OpenStack-on-OpenStack program. The code name for the OpenStack Deployment program.

trove Codename for OpenStack *Database service*.

trusted platform module (TPM) Specialized microprocessor for incorporating cryptographic keys into devices for authenticating and securing a hardware platform.

9.2.22 U

Ubuntu A Debian-based Linux distribution.

unscoped token Alternative term for an Identity service default token.

updater Collective term for a group of Object Storage components that processes queued and failed updates for containers and objects.

user In OpenStack Identity, entities represent individual API consumers and are owned by a specific domain. In OpenStack Compute, a user can be associated with roles, projects, or both.

user data A blob of data that the user can specify when they launch an instance. The instance can access this data through the metadata service or config drive. Commonly used to pass a shell script that the instance runs on boot.

User Mode Linux (UML) An OpenStack-supported hypervisor.

Ussuri The code name for the twenty first release of OpenStack. The OpenStack Infrastructure Summit took place in Shanghai, Peoples Republic of China. The release is named after the Ussuri river.

9.2.23 V

Victoria The code name for the twenty second release of OpenStack. The OpenDev + PTG was planned to take place in Vancouver, British Columbia, Canada. The release is named after Victoria, the capital city of British Columbia.

The in-person event was cancelled due to COVID-19. The event is being virtualized instead.

VIF UUID Unique ID assigned to each Networking VIF.

Virtual Central Processing Unit (vCPU) Subdivides physical CPUs. Instances can then use those divisions.

Virtual Disk Image (VDI) One of the VM image disk formats supported by Image service.

Virtual Extensible LAN (VXLAN) A network virtualization technology that attempts to reduce the scalability problems associated with large cloud computing deployments. It uses a VLAN-like encapsulation technique to encapsulate Ethernet frames within UDP packets.

Virtual Hard Disk (VHD) One of the VM image disk formats supported by Image service.

virtual IP address (VIP) An Internet Protocol (IP) address configured on the load balancer for use by clients connecting to a service that is load balanced. Incoming connections are distributed to back-end nodes based on the configuration of the load balancer.

virtual machine (VM) An operating system instance that runs on top of a hypervisor. Multiple VMs can run at the same time on the same physical host.

virtual network An L2 network segment within Networking.

Virtual Network Computing (VNC) Open source GUI and CLI tools used for remote console access to VMs. Supported by Compute.

Virtual Network InterFace (VIF) An interface that is plugged into a port in a Networking network. Typically a virtual network interface belonging to a VM.

virtual networking A generic term for virtualization of network functions such as switching, routing, load balancing, and security using a combination of VMs and overlays on physical network infrastructure.

virtual port Attachment point where a virtual interface connects to a virtual network.

virtual private network (VPN) Provided by Compute in the form of cloudpipes, specialized instances that are used to create VPNs on a per-project basis.

virtual server Alternative term for a VM or guest.

virtual switch (vSwitch) Software that runs on a host or node and provides the features and functions of a hardware-based network switch.

virtual VLAN Alternative term for a virtual network.

VirtualBox An OpenStack-supported hypervisor.

Vitrage Code name for the *Root Cause Analysis service*.

VLAN manager A Compute component that provides dnsmasq and radvd and sets up forwarding to and from cloudpipe instances.

VLAN network The Network Controller provides virtual networks to enable compute servers to interact with each other and with the public network. All machines must have a public and private network interface. A VLAN network is a private network interface, which is controlled by the `vlan_interface` option with VLAN managers.

VM disk (VMDK) One of the VM image disk formats supported by Image service.

VM image Alternative term for an image.

VM Remote Control (VMRC) Method to access VM instance consoles using a web browser. Supported by Compute.

VMware API Supports interaction with VMware products in Compute.

VMware NSX Neutron plug-in Provides support for VMware NSX in Neutron.

VNC proxy A Compute component that provides users access to the consoles of their VM instances through VNC or VMRC.

volume Disk-based data storage generally represented as an iSCSI target with a file system that supports extended attributes; can be persistent or ephemeral.

Volume API Alternative name for the Block Storage API.

volume controller A Block Storage component that oversees and coordinates storage volume actions.

volume driver Alternative term for a volume plug-in.

volume ID Unique ID applied to each storage volume under the Block Storage control.

volume manager A Block Storage component that creates, attaches, and detaches persistent storage volumes.

volume node A Block Storage node that runs the cinder-volume daemon.

volume plug-in Provides support for new and specialized types of back-end storage for the Block Storage volume manager.

volume worker A cinder component that interacts with back-end storage to manage the creation and deletion of volumes and the creation of compute volumes, provided by the cinder-volume daemon.

vSphere An OpenStack-supported hypervisor.

9.2.24 W

Wallaby The code name for the twenty third release of OpenStack. Wallabies are native to Australia, which at the start of this naming period was experiencing unprecedented wild fires.

Watcher Code name for the *Infrastructure Optimization service*.

weight Used by Object Storage devices to determine which storage devices are suitable for the job. Devices are weighted by size.

weighted cost The sum of each cost used when deciding where to start a new VM instance in Compute.

weighting A Compute process that determines the suitability of the VM instances for a job for a particular host. For example, not enough RAM on the host, too many CPUs on the host, and so on.

worker A daemon that listens to a queue and carries out tasks in response to messages. For example, the cinder-volume worker manages volume creation and deletion on storage arrays.

Workflow service (mistral) The OpenStack service that provides a simple YAML-based language to write workflows (tasks and transition rules) and a service that allows to upload them, modify, run them at scale and in a highly available manner, manage and monitor workflow execution state and state of individual tasks.

9.2.25 X

X.509 X.509 is the most widely used standard for defining digital certificates. It is a data structure that contains the subject (entity) identifiable information such as its name along with its public key. The certificate can contain a few other attributes as well depending upon the version. The most recent and standard version of X.509 is v3.

Xen Xen is a hypervisor using a microkernel design, providing services that allow multiple computer operating systems to execute on the same computer hardware concurrently.

Xen API The Xen administrative API, which is supported by Compute.

Xen Cloud Platform (XCP) An OpenStack-supported hypervisor.

Xen Storage Manager Volume Driver A Block Storage volume plug-in that enables communication with the Xen Storage Manager API.

XenServer An OpenStack-supported hypervisor.

XFS High-performance 64-bit file system created by Silicon Graphics. Excels in parallel I/O operations and data consistency.

9.2.26 Z

zaqar Codename for the *Message service*.

ZeroMQ Message queue software supported by OpenStack. An alternative to RabbitMQ. Also spelled 0MQ.

Zuul Zuul is an open source CI/CD platform specializing in gating changes across multiple systems and applications before landing a single patch.

Zuul is used for OpenStack development to ensure that only tested code gets merged.

INDEX