Guia de Instalação do OpenStack

para Ubuntu 14.04

juno (December 31, 2014)

DRAFT

Kilo

docs.openstack.org

Guia de Instalação do OpenStack para Ubuntu 14.04

juno (2014-12-31)

Resumo

The OpenStack® system consists of several key projects that you install separately but that work together depending on your cloud needs. These projects include Compute, Identity Service, Networking, Image Service, Block Storage, Object Storage, Telemetry, Orchestration, and Database. You can install any of these projects separately and configure them stand-alone or as connected entities. This guide walks through an installation by using packages available through Ubuntu 14.04. Explanations of configuration options and sample configuration files are included.

Índice

Lista de Figuras

Lista de Tabelas

Preface

Conventions

The OpenStack documentation uses several typesetting conventions.

Notices

Notices take these forms:

Nota

A handy tip or reminder.

Importante

Something you must be aware of before proceeding.

Atenção

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

Command prompts

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

 # prompt 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.

Document change history

This version of the guide replaces and obsoletes all earlier versions.

The following table describes the most recent changes:

Revision Date	Summary of Changes
October 15, 2014	• For the Juno release, this guide contains these updates: Replace openstack-config commands with general configuration file editing. Standardize on a single message queue system (RabbitMQ). Reference generic SQL database, enabling MySQL or MariaDB where appropriate. Replace auth_port and auth_protocol with identity_uri, and auth_host with auth_uri. Multiple edits for consistency. It has been updated for Juno and new distribution versions.
June 3, 2014	• Iniciar documentação para Juno.
April 16, 2014	• Update for Icehouse, rework Networking setup to use ML2 as plugin, add new chapter for Database Service setup, improved basic configuration.
October 25, 2013	• Adicionado suporte inicial ao Debian.
October 17, 2013	• Versão Havana.
October 16, 2013	• Adição do suporte para SUSE Linux Enterprise.

Revision Date	Summary of Changes
October 8, 2013	• Reorganização completa para Havana.
September 9, 2013	• Build para openSUSE.
August 1, 2013	• Fixes to Object Storage verification steps. Fix bug 1207347.
July 25, 2013	• Adds creation of cinder user and addition to the service tenant. Fix bug 1205057.
May 8, 2013	• Atualizado o título do livro para consistência.
May 2, 2013	• Atualizada a capa e corrigidos pequenos erros no apêndice.

Capítulo 1. Arquitetura

Índice

Visão Geral

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 complemental services. Each service offers an application programming interface (*API*) that facilitates this integration. The following table provides a list of OpenStack services:

Tabela 1.1. OpenStack services

Service	Project name	Description
Dashboard	*Horizon*	Provides a web-based self-service portal to interact with underlying OpenStack services, such as launching an instance, assigning IP addresses and configuring access controls.
Compute	*Nova*	Manages the lifecycle of compute instances in an OpenStack environment. Responsibilities include spawning, scheduling and decommissioning of virtual machines on demand.
Networking	*Neutron*	Enables Network-Connectivity-as-a-Service for other OpenStack services, such as OpenStack Compute. Provides an API for users to define networks and the attachments into them. Has a pluggable architecture that supports many popular networking vendors and technologies.
Storage		
Object Storage	*Swift*	Stores and retrieves arbitrary unstructured data objects via a *RESTful*, HTTP based API. It is highly fault tolerant with its data replication and scale out architecture. Its implementation is not like a file server with mountable directories.
Block Storage	*Cinder*	Provides persistent block storage to running instances. Its pluggable driver architecture facilitates the creation and management of block storage devices.
Shared services		
Identity service	*Keystone*	Provides an authentication and authorization service for other OpenStack services. Provides a catalog of endpoints for all OpenStack services.
Image Service	*Glance*	Stores and retrieves virtual machine disk images. OpenStack Compute makes use of this during instance provisioning.
Telemetry	*Ceilometer*	Monitors and meters the OpenStack cloud for billing, benchmarking, scalability, and statistical purposes.
Higher-level services		

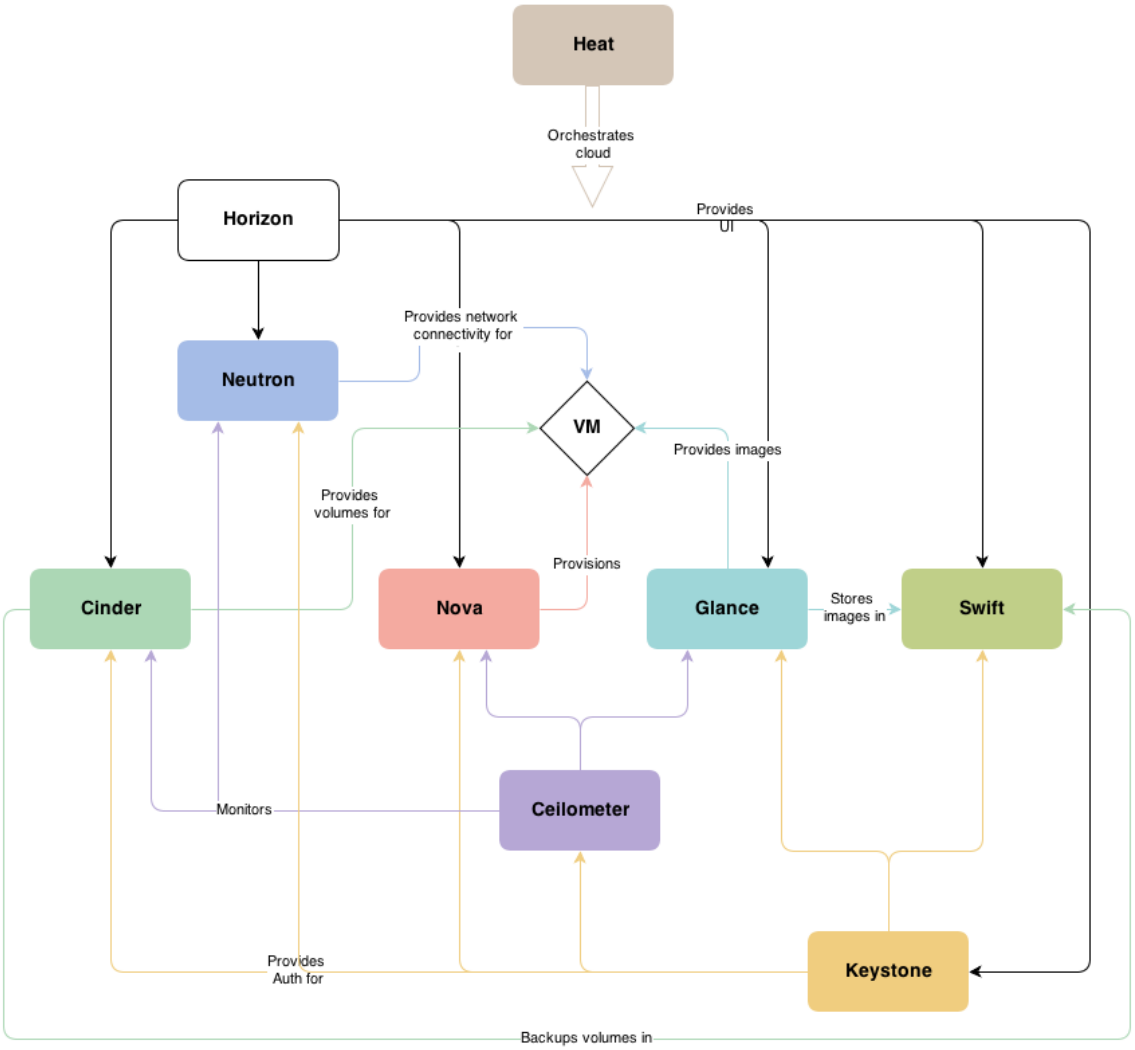
Service	Project name	Description
Orchestration	*Heat*	Orchestrates multiple composite cloud applications by using either the native *HOT* template format or the AWS CloudFormation template format, through both an OpenStack-native REST API and a CloudFormation-compatible Query API.
Database Service	*Trove*	Provides scalable and reliable Cloud Database-as-a-Service functionality for both relational and non-relational database engines.

This guide describes how to deploy these services in a functional test environment and, by example, teaches you how to build a production environment. Realistically, you would use automation tools such as Ansible, Chef, and Puppet to deploy and manage a production environment.

Arquitetura conceitual

Launching a virtual machine or instance involves many interactions among several services. The following diagram provides the conceptual architecture of a typical OpenStack environment.

Figura 1.1. Arquitetura conceitual



Arquiteturas de exemplo

OpenStack is highly configurable to meet different needs with various compute, networking, and storage options. This guide enables you to choose your own OpenStack adventure using a combination of core and optional services. This guide uses the following example architectures:

- Arquitetura de três nodos com nodo OpenStack Networking (neutron) e nodos opcionais para serviços Block Storage e Object Storage.

  - The *controller node* runs the Identity service, Image Service, management portions of Compute and Networking, Networking plug-in, and the dashboard. It also includes supporting services such as a SQL database, *message queue*, and *Network Time Protocol (NTP)*.

    Optionally, the controller node runs portions of Block Storage, Object Storage, Orchestration, Telemetry, Database, and Data Processing services. These components provide additional features for your environment.

  - The network node runs the Networking plug-in and several agents that provision tenant networks and provide switching, routing, *NAT*, and *DHCP* services. This node also handles external (Internet) connectivity for tenant virtual machine instances.

  - The *compute node* runs the *hypervisor* portion of Compute that operates *tenant virtual machines* or instances. By default, Compute uses *KVM* as the *hypervisor*. The compute node also runs the Networking plug-in and an agent that connect tenant networks to instances and provide firewalling (*security groups*) services. You can run more than one compute node.

    Optionally, the compute node runs a Telemetry agent to collect metrics. Also, it can contain a third network interface on a separate storage network to improve performance of storage services.

  - The optional Block Storage node contains the disks that the Block Storage service provisions for tenant virtual machine instances. You can run more than one of these nodes.

    Optionally, the Block Storage node runs a Telemetry agent to collect metrics. Also, it can contain a second network interface on a separate storage network to improve performance of storage services.

  - The optional Object Storage nodes contain the disks that the Object Storage service uses for storing accounts, containers, and objects. You can run more than two of these nodes. However, the minimal architecture example requires two nodes.

    Optionally, these nodes can contain a second network interface on a separate storage network to improve performance of storage services.

Nota

When you implement this architecture, skip "Rede legada (nova-network)" [84] in Capítulo 6, Adicione o componente de rede [60]. Opti-

onal services might require additional nodes or additional resources on existing nodes.

Figura 1.2. Requisitos de Hardware para arquitetura mínima de exemplo com OpenStack Networking (neutron)

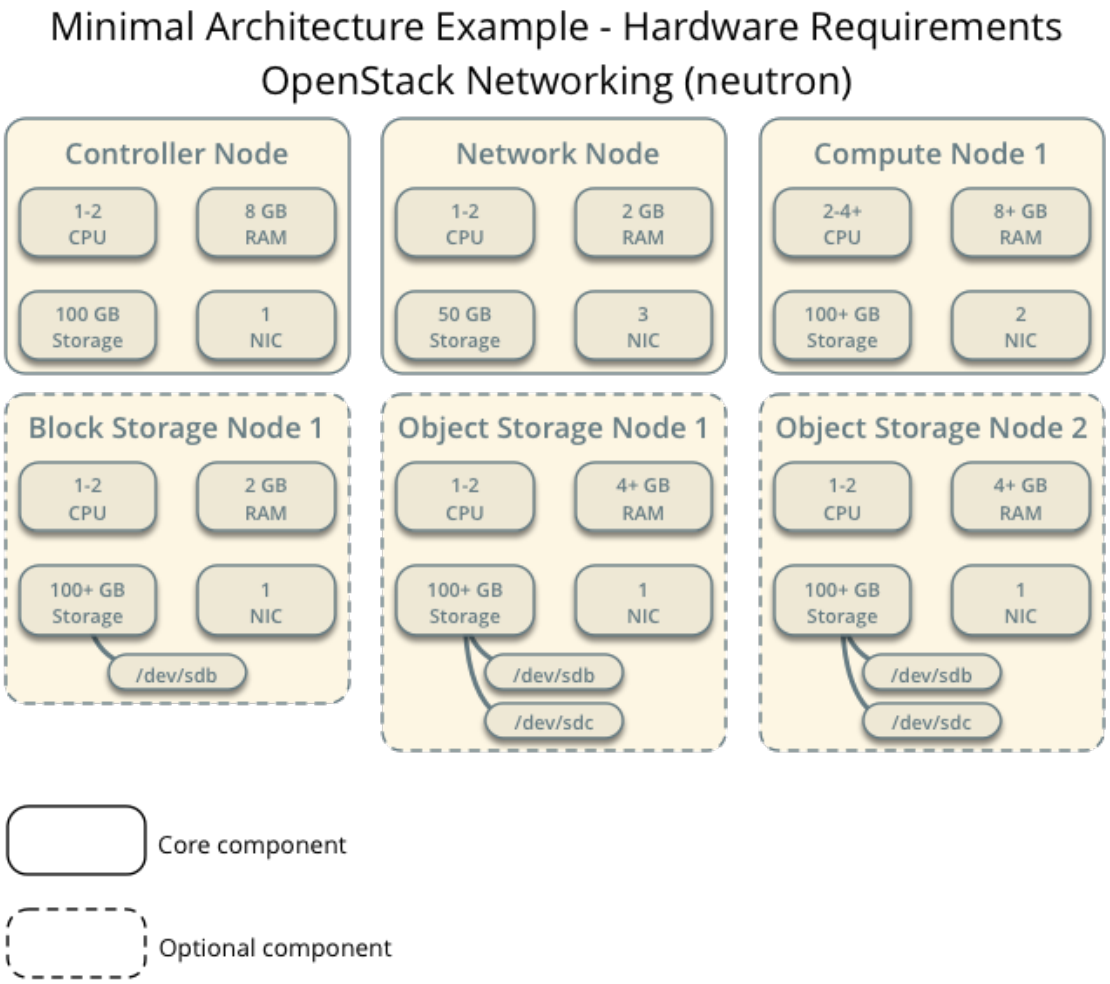


Figura 1.3. Exemplo de arquitetura mínima com o layout de Rede do OpenStack (neutron)

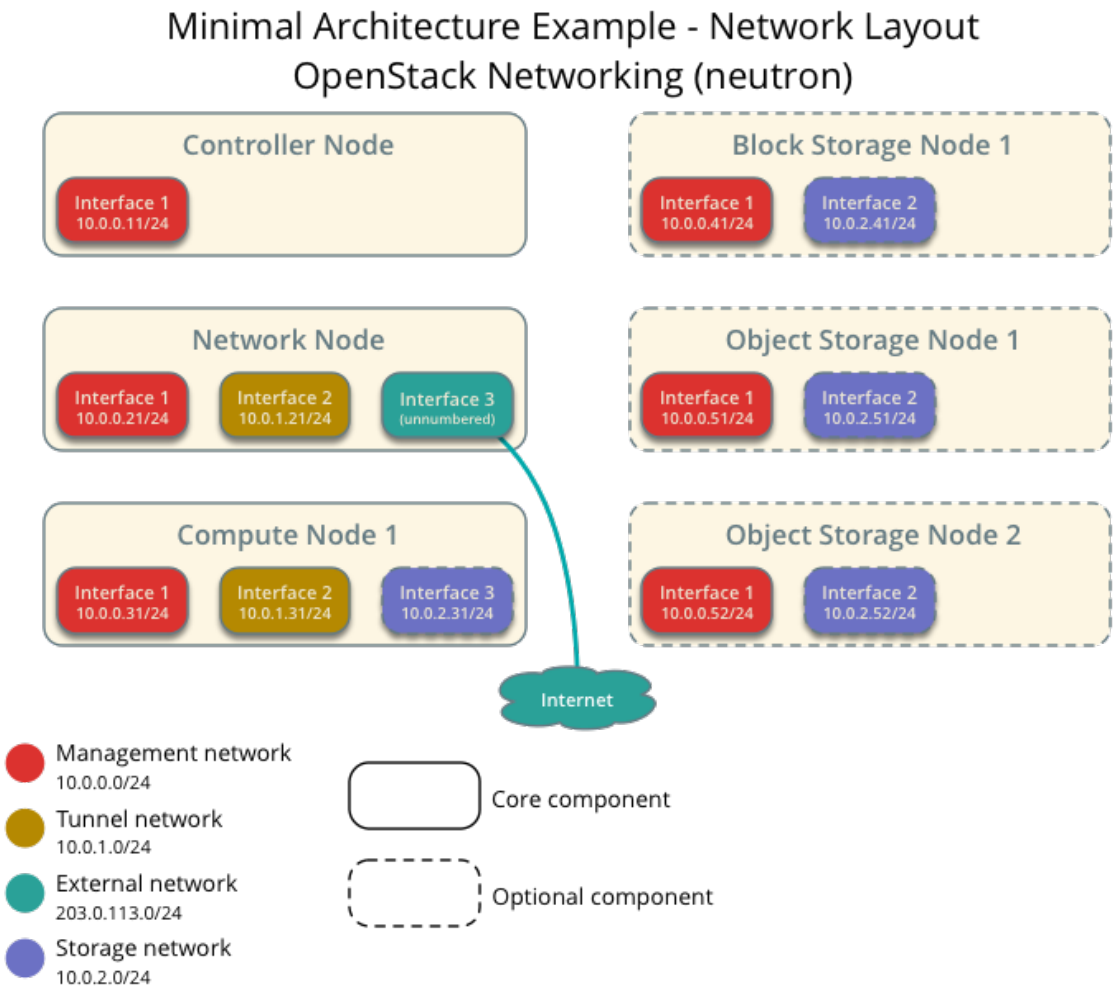
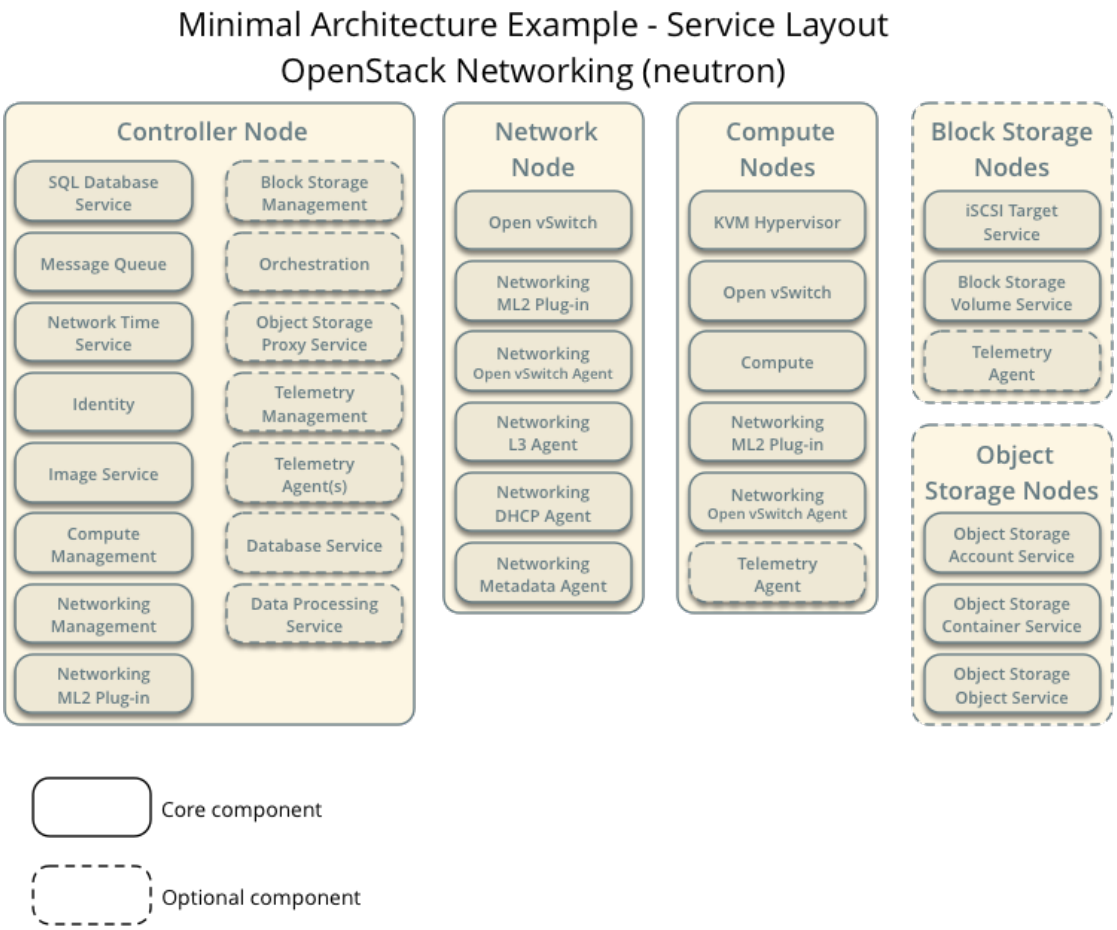


Figura 1.4. Layout de serviço para arquitetura mínima de exemplo com OpenStack Networking (neutron)



- Two-node architecture with legacy networking (nova-network) and optional nodes for Block Storage and Object Storage services.

  - The *controller node* runs the Identity service, Image Service, management portion of Compute, and the dashboard. It also includes supporting services such as a SQL database, *message queue*, and *Network Time Protocol (NTP)*.

    Optionally, the controller node runs portions of Block Storage, Object Storage, Orchestration, Telemetry, Database, and Data Processing services. These components provide additional features for your environment.

  - The *compute node* runs the *hypervisor* portion of Compute that operates *tenant virtual machines* or instances. By default, Compute uses *KVM* as the *hypervisor*. Compute also provisions tenant networks and provides firewalling (*security groups*) services. You can run more than one compute node.

Optionally, the compute node runs a Telemetry agent to collect metrics. Also, it can contain a third network interface on a separate storage network to improve performance of storage services.

- The optional Block Storage node contains the disks that the Block Storage service provisions for tenant virtual machine instances. You can run more than one of these nodes.

  Optionally, the Block Storage node runs a Telemetry agent to collect metrics. Also, it can contain a second network interface on a separate storage network to improve performance of storage services.

- The optional Object Storage nodes contain the disks that the Object Storage service uses for storing accounts, containers, and objects. You can run more than two of these nodes. However, the minimal architecture example requires two nodes.

  Optionally, these nodes can contain a second network interface on a separate storage network to improve performance of storage services.

Nota

When you implement this architecture, skip "Rede OpenStack (neutron)" [60] in Capítulo 6, Adicione o componente de rede [60]. To use optional services, you might need to build additional nodes, as described in subsequent chapters.

Figura 1.5. Minimal architecture example with legacy networking (nova-network)Hardware requirements

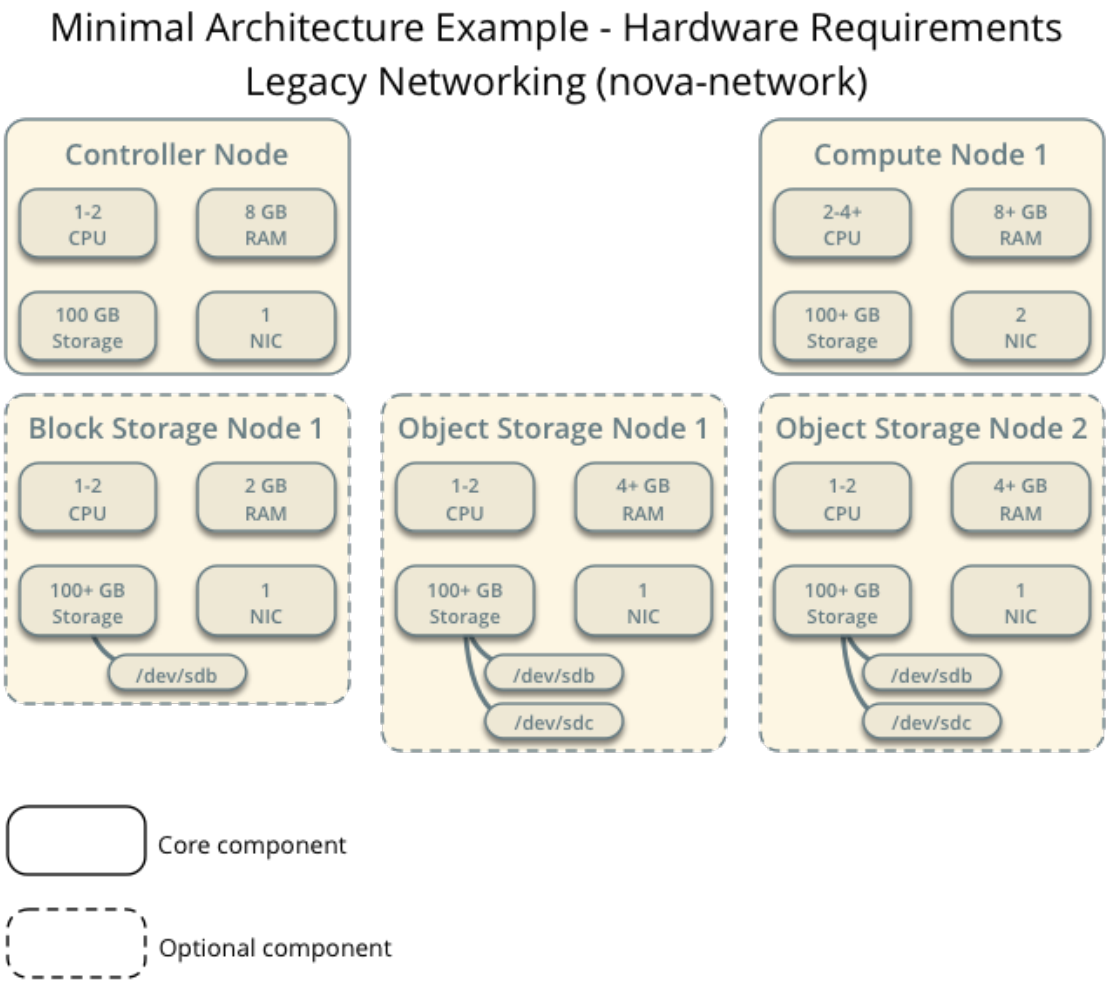


Figura 1.6. Layout de Rede de exemplo de arquitetura mínima com rede legada (nova-network)

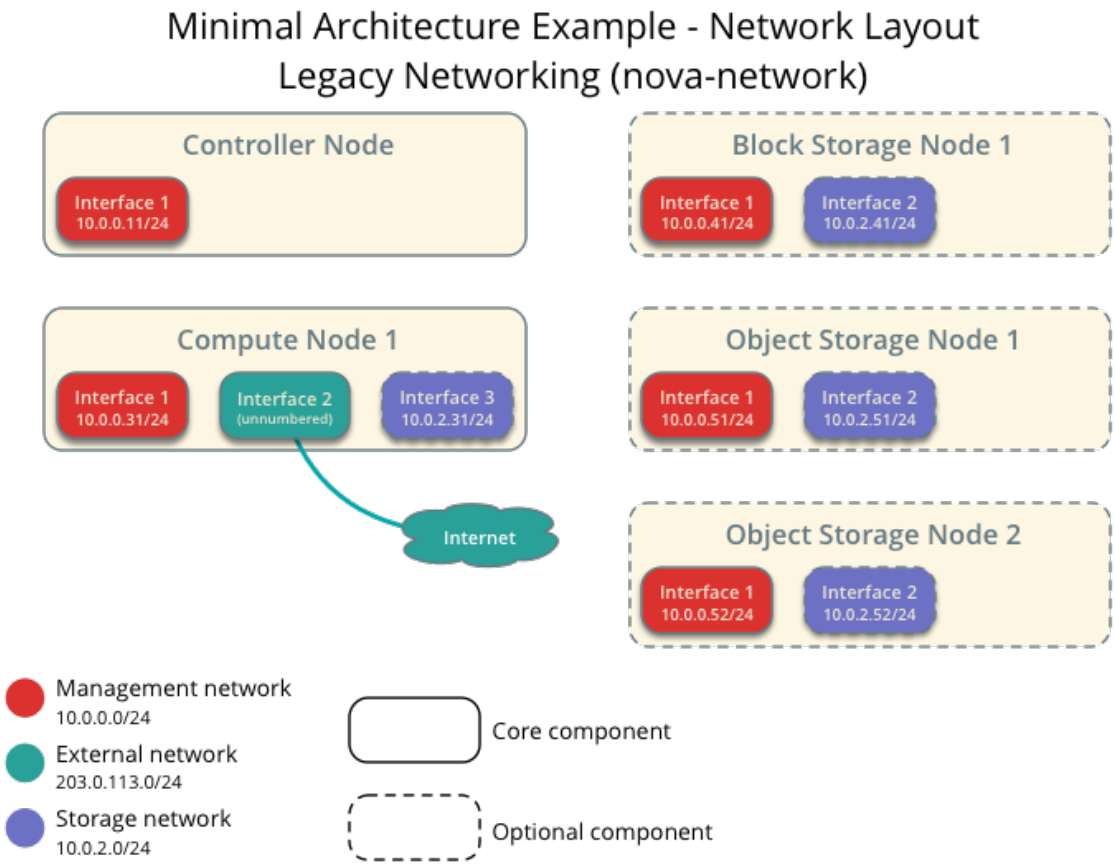
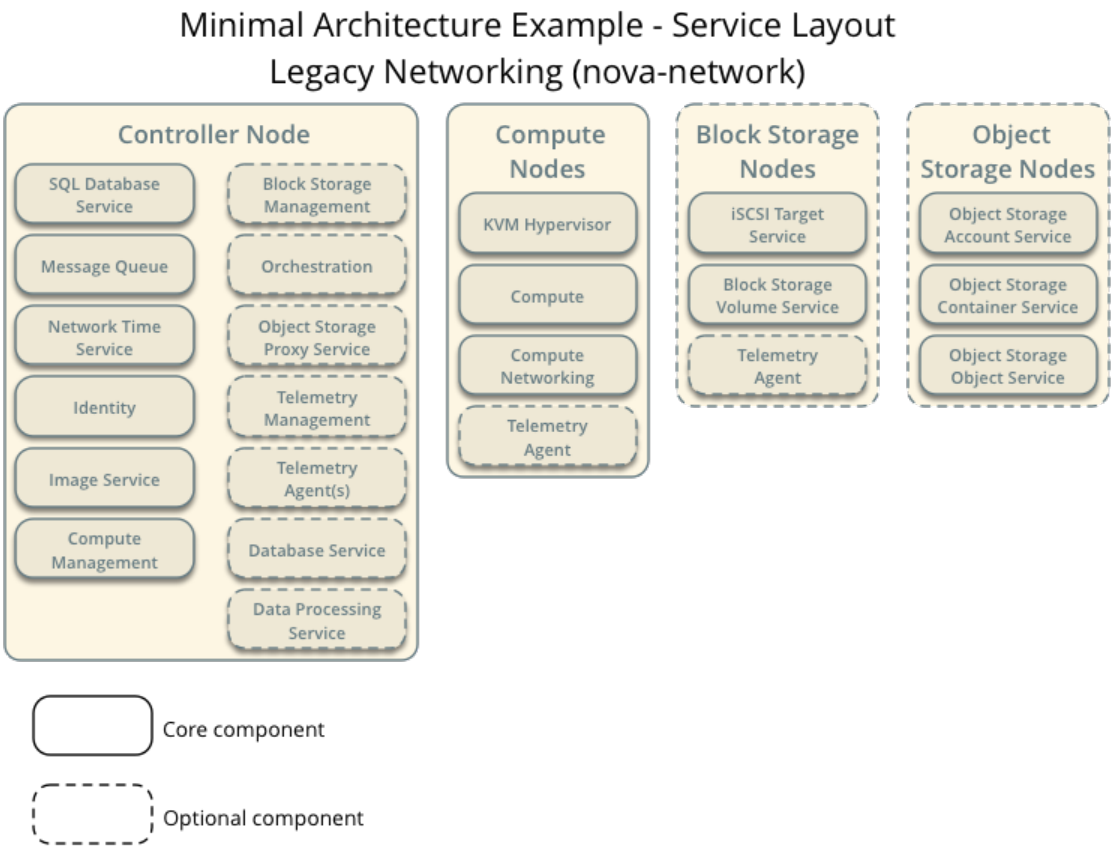


Figura 1.7. Minimal architecture example with legacy networking (nova-network)Service layout



Capítulo 2. Ambiente básico

Índice

Nota

The trunk version of this guide focuses on the future Kilo release and will not work for the current Juno release. If you want to install Juno, you must use the Juno version of this guide instead.

This chapter explains how to configure each node in the example architectures including the two-node architecture with legacy networking and three-node architecture with OpenStack Networking (neutron).

Nota

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 Capítulo 9, Adicionar Object Storage [100] after configuring the appropriate nodes for it. However, the dashboard requires at least the Image Service and Compute.

Nota

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.

Nota

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.

Antes de você começar

For best performance, we recommend that your environment meets or exceeds the hardware requirements in Figura 1.2, “Requisitos de Hardware para arquitetura mínima de exemplo com OpenStack Networking (neutron)” [4] or Figura 1.5, “Minimal architecture example with legacy networking (nova-network)Hardware requirements” [8]. However,

OpenStack does not require a significant amount of resources and the following minimum requirements should support a proof-of-concept environment with core services and several *CirrOS* instances:

- Nodo Controlador: 1 processador, 2 GB de memória, e 5GB de armazenamento
- Nodo de Rede: 1 processador, 512 MB de memória, e 5GB de armazenamento
- Nodo de Computação: 1 processador, 2 GB de memória, e 10GB de armazenamento

Para minimizar a desordem e fornecer mais recursos para o OpenStack, recomendamos uma instalação mínima de sua distribuição Linux. Também, recomendamos fortemente que você instale uma versão de 64-bits da sua distribuição pelo menos no nodo de Computação. se você instalar a versão de 32 bits da sua distribuição no nodo de computação, a tentativa de iniciar uma instância usando uma imagem de 64-bits irá falhar.

Nota

Uma partição simples de disco em cada nodo funciona para as instalações mais básicas. Contudo, você deve considerar o *Gerenciador de volumes lógicos (LVM)* para instalações com serviços opcionais como o Block Storage.

Muitos usuários constroem seus ambientes de testes em *máquinas virtuais (VMs)*. O principal benefício das VMs incluem os seguintes:

- Um servidor físico pode suportar múltiplos nodes, cada um com quase qualquer número de interfaces de rede.
- Habilidade de pegar "snap shots" através do processo de instalação e "retornar" para uma configuração funcional em caso de problemas.

Contudo, as VMs irão reduzir o desempenho de suas instâncias, particularmente se seu hypervisor e/ou seu processador não possui suporte para aceleração por hardware de VMs aninhadas.

Nota

If you choose to install on VMs, make sure your hypervisor permits *promiscuous mode* and disables MAC address filtering on the *external network*.

For more information about system requirements, see the OpenStack Operations Guide.

Segurança

Os serviços OpenStack suportam vários métodos de segurança, incluindo senha, políticas e criptografia. Adicionalmente, os serviços de suporte, incluindo o servidor de banco de dados e o intermediador de mensagens suportam pelo menos segurança de senha.

To ease the installation process, this guide only covers password security where applicable. You can create secure passwords manually, generate them using a tool such as pwgen, or by running the following command:

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

Para os serviços OpenStack, este guia utiliza *SERVICE_PASS* para referenciar volume de serviços, senhas e replaceable>SERVICE_DBPASS

A seguinte tabela fornece uma lista de serviços que requerem senhas e suas referências associadas no guia:

Tabela 2.1. Senhas

nome da Senha	Descrição
Senha de Banco de Dados (não usa variável)	Senha de root para o banco de dados
RABBIT_PASS	Senha de visitante para o RabbitMQ
KEYSTONE_DBPASS	Senha de banco de dados para o Serviço de Identificação
DEMO_PASS	Senha do usuário `demo`
ADMIN_PASS	Senha do usuário `admin`
GLANCE_DBPASS	Senha de Banco de Dados para o serviço de Imagem
GLANCE_PASS	Senha para o usuário do serviço de Imagem `glance`
NOVA_DBPASS	Senha de Banco de Dados para o serviço e Computação
NOVA_PASS	Senha para o serviço de Computação `nova`
DASH_DBPASS	Senha de Banco de Dados para o Dashboard
CINDER_DBPASS	Senha de Banco de Dados para o serviço de Block Storage
CINDER_PASS	Senha para o usuário de Block Storage `cinder`
NEUTRON_DBPASS	Senha de Banco de Dados para o serviço de Rede
NEUTRON_PASS	Senha para o usuário do serviço Neutron `neutron`
HEAT_DBPASS	Senha de Banco de Dados para o serviço de Orquestração
HEAT_PASS	Senha para o usuário do serviço de Orquestraçãoheat
CEILOMETER_DBPASS	Senha de Banco de Dados para o serviço de Telemetria
CEILOMETER_PASS	Senha para o usuário do serviço de Telemetria`ceilometer`
TROVE_DBPASS	Senha de Banco de Dados para o serviço de Banco de Dados
TROVE_PASS	Senha para o usuário do serviço de Banco de Dados`trove`

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 Cloud Administrator Guide for more information. Also, 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.

Rede

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.

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 network infrastructure provides Internet access via *NAT*. To illustrate the flexibility of *IaaS*, the example architectures use public IP address space for the external network and assume that network infrastructure provides direct Internet access to instances in your OpenStack environment. In environments with only one block of public IP address space, both the management and external networks must ultimately obtain Internet access using it. For simplicity, the diagrams in this guide only show Internet access for OpenStack services.

Nota

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

Prossiga com a configuração de rede para a arquitetura de exemplo OpenStack Networking (neutron) ou legacy networking (nova-network).

Rede OpenStack (neutron)

A arquitetura de exemplo com Rede OpenStack (neutron) requer um nodo controlador, um nodo de rede, e pelo menos um nodo de computação. O nodo controlador contém uma interface de rede na *rede de gerenciamento*. O nodo de rede contém uma interface de rede na rede de gerenciamento, uma na *rede de túneis de instância*, e uma na *rede externa*. o nodo de computação contém uma interface de rede na rede de gerenciamento e uma na rede de túneis de instância.

The example architecture assumes use of the following networks:

- Management on 10.0.0.0/24 with gateway 10.0.0.1

Nota

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

- Instance tunnels on 10.0.1.0/24 without a gateway

Nota

This network does not require a gateway because communication only occurs among network and compute nodes in your OpenStack environment.

- External on 203.0.113.0/24 with gateway 203.0.113.1

Nota

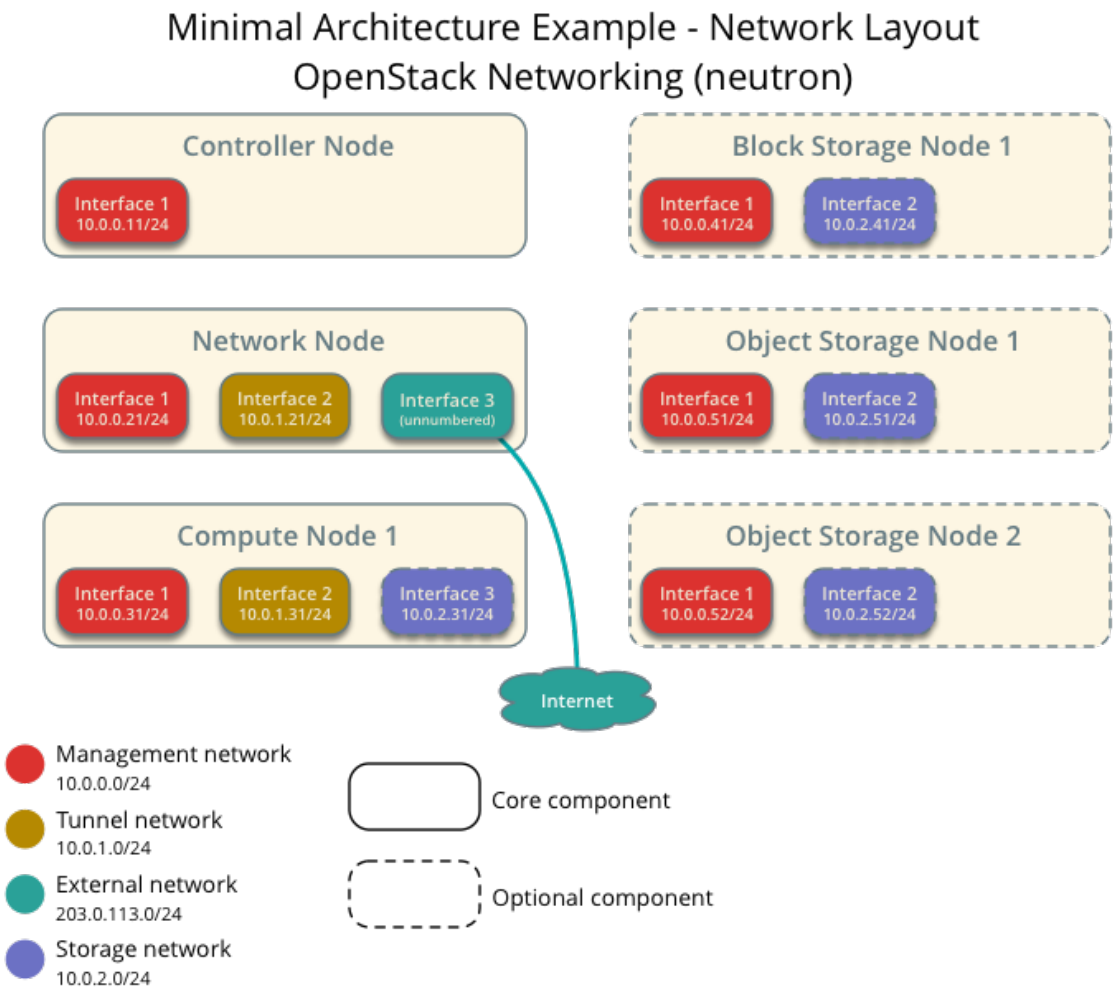
This network requires a gateway to provide Internet access to instances in your OpenStack environment.

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

Nota

Os nomes das interfaces de rede variam por distribuição. Tradicionalmente, a interfaces utilizam "eth" seguido de um número sequencial. Para cobrir todas as variações, este guia simplesmente refere-se à primeira interface como a interface com o menor número, à segunda interface como a interface de número médio, e à terceira interface como a interface de número maior.

Figura 2.1. Exemplo de arquitetura mínima com o layout de Rede do OpenStack (neutron)



A menos que você pretenda utilizar a configuração exata fornecida nesta arquitetura de exemplo, você deve modificar as redes neste procedimento para corresponder ao seu ambiente. Também, cada nodo deve resolver os outros nodos pelo nome adicionalmente ao endereço IP. Por exemplo, o nome de *`controlador`* deve resolver para `10.0.0.11`, o endereço IP da interface de gerenciamento no nodo controlador.

Atenção

A reconfiguração das intefaces de rede irá interromper a conectividade de rede. Recomendamos utilizar uma sessão de terminal local para estes procedimentos.

Nodo controlador

Para configurar a rede:

1. Configure a primeira interface como interface de gerenciamento:

   Endereço IP: 10.0.0.11

   Máscara de rede: 255.255.255.0 (ou /24)

   Gateway padrão: 10.0.0.1

2. Reinicialize o sistema para ativar as alterações.

Para configurar a resolução de nomes:

1. defina o hostname do nodo para *`controller`*.

2. Edite o arquivo `/etc/hosts` para conter o seguinte:

```
# controller
10.0.0.11       controller

# network
10.0.0.21       network

# compute1
10.0.0.31       compute1
```

Atenção

Você deve remover ou comentar a linha começando com `127.0.1.1`.

Nodo de rede

Para configurar a rede:

1. Configure a primeira interface como interface de gerenciamento:

   Endereço IP: 10.0.0.21

   Máscara de rede: 255.255.255.0 (ou /24)

   Gateway padrão: 10.0.0.1

2. Configure a segunda interface com interface de instância de túneis:

   Endereço IP: 10.0.1..21

Máscara de rede: 255.255.255.0 (ou /24)

3. A interface externa utiliza uma configuração especial sem um endereço IP atribuído a ela. Configure a terceira interface como interface externa.

   Substitua *INTERFACE_NAME* com o nome atual da interface. Por exempo, *eth2* ou *ens256*.

   - Edite o arquivo `/etc/network/interfaces` para conter o seguinte:

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

4. Reinicialize o sistema para ativar as alterações.

Para configurar a resolução de nomes:

1. defina o hostname do nodo para `network`.

2. Edite o arquivo `/etc/hosts` para conter o seguinte:

   ```
   # network
   10.0.0.21       network

   # controller
   10.0.0.11       controller

   # compute1
   10.0.0.31       compute1
   ```

Atenção

Você deve remover ou comentar a linha começando com `127.0.1.1`.

Nodo de computação

Para configurar a rede:

1. Configure a primeira interface como interface de gerenciamento:

   Endereço IP: 10.0.0.31

   Máscara de rede: 255.255.255.0 (ou /24)

   Gateway padrão: 10.0.0.1

Nota

Nodos de Computação adicionais devem utilizar 10.0.0.32, 10.0.0.33, e assim por diante.

2. Configure a segunda interface com interface de instância de túneis:

Endereço IP: 10.0.1.31

Máscara de rede: 255.255.255.0 (ou /24)

Nota

Nodos de computação adicionais devem utilizar 10.0.1.32, 10.0.1.33, e assim por diante.

3. Reinicialize o sistema para ativar as alterações.

Para configurar a resolução de nomes:

1. Defina o hostname do nodo para `compute1`.

2. Edite o arquivo `/etc/hosts` para conter o seguinte:

```
# compute1
10.0.0.31       compute1

# controller
10.0.0.11       controller

# network
10.0.0.21       network
```

Atenção

Você deve remover ou comentar a linha começando com `127.0.1.1`.

Verifique a conectividade

Recomendamos que você verifique a conectividade de rede à Internet e entre os nodos antes de prosseguir adiante.

1. Do nodo *controlador*, **ping** um site na 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
```

2. Do nodo *controlador*, **ping**a interface de gerenciamento no nodo de *rede*:

```
# ping -c 4 rede
PING network (10.0.0.21) 56(84) bytes of data.
64 bytes from network (10.0.0.21): icmp_seq=1 ttl=64 time=0.263 ms
64 bytes from network (10.0.0.21): icmp_seq=2 ttl=64 time=0.202 ms
64 bytes from network (10.0.0.21): icmp_seq=3 ttl=64 time=0.203 ms
64 bytes from network (10.0.0.21): icmp_seq=4 ttl=64 time=0.202 ms
```

```
--- network 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. A partir do nodo *controlador*, **ping** a interface de gerenciamento no nodo de *computação*:

```
# 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

--- network 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
```

4. A partir do nodo de *rede*, **ping** um site na 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
```

5. A partir do nodo de *rede*, **ping** a interface de gerenciamento no nodo *controlador*:

```
# ping -c 4 controlador
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
```

6. A partir do nodo de *rede*, **ping** a interface de instâncias de túneis no nodo de *computação*:

```
# ping -c 4 10.0.1.31
PING 10.0.1.31 (10.0.1.31) 56(84) bytes of data.
64 bytes from 10.0.1.31 (10.0.1.31): icmp_seq=1 ttl=64 time=0.263 ms
64 bytes from 10.0.1.31 (10.0.1.31): icmp_seq=2 ttl=64 time=0.202 ms
64 bytes from 10.0.1.31 (10.0.1.31): icmp_seq=3 ttl=64 time=0.203 ms
64 bytes from 10.0.1.31 (10.0.1.31): icmp_seq=4 ttl=64 time=0.202 ms

--- 10.0.1.31 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
```

7. A partir do nodo de *computação*, **ping** um site na 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
```

8. A partir do nodo de *computação*, **ping** a interface de gerenciamento no nodo *controlador*:

```
# ping -c 4 controlador
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
```

9. A partir do nodo de *computação*, **ping** a interface de instâncias de túneis no nodo de *rede*:

```
# ping -c 4 10.0.1.21
PING 10.0.1.21 (10.0.1.21) 56(84) bytes of data.
64 bytes from 10.0.1.21 (10.0.1.21): icmp_seq=1 ttl=64 time=0.263 ms
64 bytes from 10.0.1.21 (10.0.1.21): icmp_seq=2 ttl=64 time=0.202 ms
64 bytes from 10.0.1.21 (10.0.1.21): icmp_seq=3 ttl=64 time=0.203 ms
64 bytes from 10.0.1.21 (10.0.1.21): icmp_seq=4 ttl=64 time=0.202 ms

--- 10.0.1.21 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
```

Rede legada (nova-network)

A arquitetura de exemplo com rede legada (nova-network) requer um nodo controlador e pelo menos um nodo de Computação. O nodo controlador contém uma interface de rede na *rede de gerenciamento*. O nodo de Computação contém uma interface de rede na rede de gerenciamento e uma na *rede externa*.

The example architecture assumes use of the following networks:

- Management on 10.0.0.0/24 with gateway 10.0.0.1

Nota

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

- External on 203.0.113.0/24 with gateway 203.0.113.1

Nota

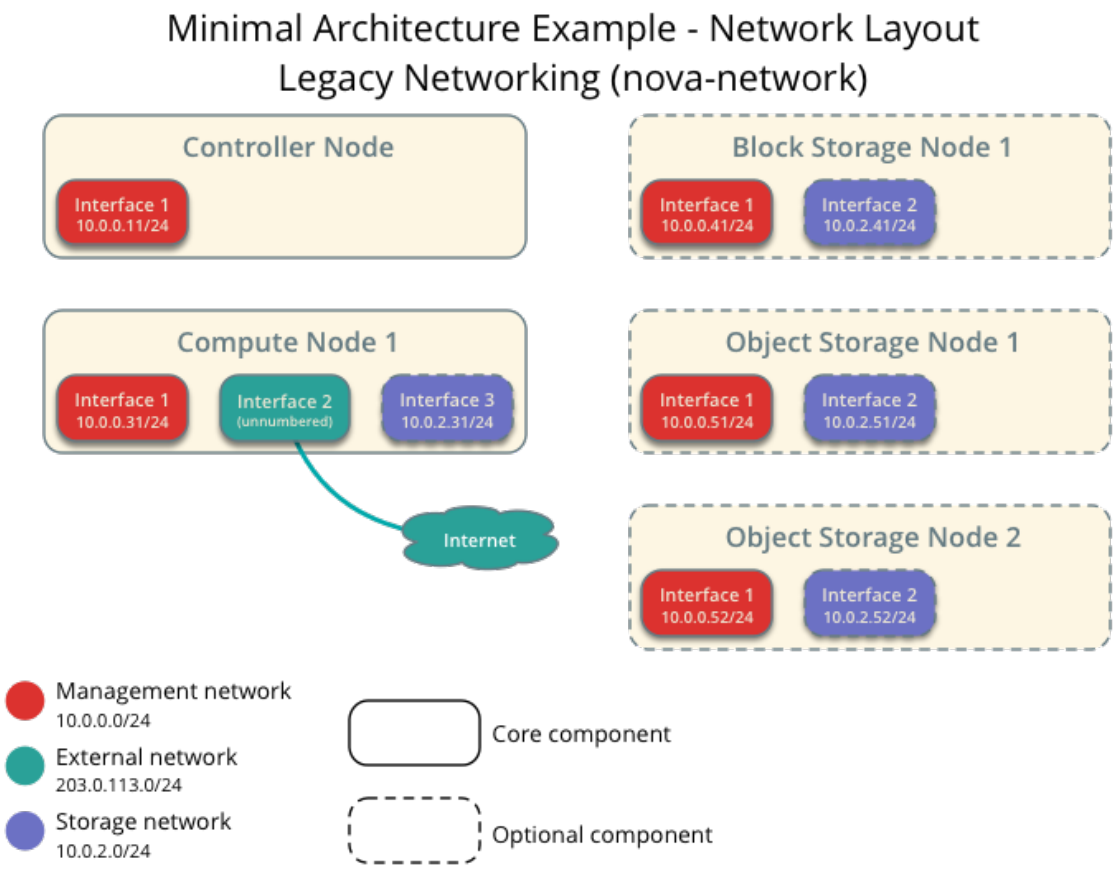
This network requires a gateway to provide Internet access to instances in your OpenStack environment.

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

Nota

Os nomes das interfaces de rede variam com a distribuição. Tradicionalmente, as interfaces utilizam "eth" seguido por um número sequencial. Para cobrir todas as variações, este guia simplesmente refere-se à primeira interface como a interface com o menor número e à segunda interface como a interface com o maior número.

Figura 2.2. Layout de Rede de exemplo de arquitetura mínima com rede legada (nova-network)



A menos que você pretenda utilizar a configuração exata fornecida nesta arquitetura de exemplo, você deve modificar as redes neste procedimento para corresponder ao seu ambiente. Também, cada nodo deve resolver os outros nodos pelo nome adicionalmente ao en-

dereço IP. Por exemplo, o nome de *controlador* deve resolver para `10.0.0.11`, o endereço IP da interface de gerenciamento no nodo controlador.

Atenção

A reconfiguração das intefaces de rede irá interromper a conectividade de rede. Recomendamos utilizar uma sessão de terminal local para estes procedimentos.

Nodo controlador

Para configurar a rede:

1. Configure a primeira interface como interface de gerenciamento:

   Endereço IP: 10.0.0.11

   Máscara de rede: 255.255.255.0 (ou /24)

   Gateway padrão: 10.0.0.1

2. Reinicialize o sistema para ativar as alterações.

Para configurar a resolução de nomes:

1. defina o hostname do nodo para *controller*.

2. Edite o arquivo `/etc/hosts` para conter o seguinte:

```
# controller
10.0.0.11       controller

# compute1
10.0.0.31       compute1
```

Atenção

Você deve remover ou comentar a linha começando com `127.0.1.1`.

Nodo de computação

Para configurar a rede:

1. Configure a primeira interface como interface de gerenciamento:

   Endereço IP: 10.0.0.31

   Máscara de rede: 255.255.255.0 (ou /24)

   Gateway padrão: 10.0.0.1

Nota

Nodos de Computação adicionais devem utilizar 10.0.0.32, 10.0.0.33, e assim por diante.

2. A interface externa utiliza uma configuração especial sem endereço IP atribuído a ela. Configure a segunda interface como interface externa:

   Substitua *INTERFACE_NAME* com o nome real da interface. Por exemplo, *eth1* ou *ens224*.

   - Edite o arquivo `/etc/network/interfaces` para conter o seguinte:

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

3. Reinicialize o sistema para ativar as alterações.

Para configurar a resolução de nomes:

1. Defina o hostname do nodo para `compute1`.

2. Edite o arquivo `/etc/hosts` para conter o seguinte:

   ```
   # compute1
   10.0.0.31       compute1

   # controller
   10.0.0.11       controller
   ```

Atenção

Você deve remover ou comentar a linha começando com `127.0.1.1`.

Verifique a conectividade

Recomendamos que você verifique a conectividade de rede à Internet e entre os nodos antes de prosseguir adiante.

1. Do nodo *controlador*, **ping** um site na 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
   ```

2. A partir do nodo *controlador*, **ping** a interface de gerenciamento no nodo de *computação*:

   ```
   # 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. A partir do nodo de *computação*, **ping** um site na 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. A partir do nodo de *computação*, **ping** a interface de gerenciamento no nodo *controlador*:

```
# ping -c 4 controlador
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
```

Network Time Protocol (NTP)

Você deve instalar o *NTP* para sincronizar adequadamente os serviços entre os nodos. recomendamos que você configure o nodo controlador para referenciar os servidores mais precisos (lower stratum) e os outros nodos para referenciarem o nodo controlador.

Nodo controlador

Para instalar o serviço NTP

- `# apt-get install ntp`

Para configurar o serviço NTP

Por padrão o nodo controlador sincroniza o tempo através de um conjunto de servidores públicos. Contudo, você pode opcionalmente editar o arquivo `/etc/ntp.conf` para configurar servidores alternativos tais como aqueles fornecidos por sua organização.

1. Edite o arquivo `/etc/ntp.conf` e adicione, altere ou remova as seguintes chaves conforme necessário para seu ambiente:

```
server NTP_SERVER iburst
restrict -4 default kod notrap nomodify
restrict -6 default kod notrap nomodify
```

Substitua *NTP_SERVER* com um hostname ou um endereço IP de um servidor NTP apropriado mais preciso (lower stratum). A configuração suporta múltiplas chaves `server`.

Nota

Para as chaves `restrict` , você essencialmente remove as opções `nopeer` e `noquery`.

Nota

Remova o arquivo `/var/lib/ntp/ntp.conf.dhcp` se ele existir.

2. Reinicie o serviço NTP:

```
# service ntp restart
```

Outros nodos

Para instalar o serviço NTP

-
```
# apt-get install ntp
```

Para configurar o serviço NTP

Configure os nodos de computação e de rede para referenciar o nodo controlador.

1. Edite o arquivo `/etc/ntp.conf`:

   Comente ou remova todas as chaves exceto a chave `server` e altere-a para referenciar o nodo controlador.

```
server controlador iburst
```

Nota

Remova o arquivo `/var/lib/ntp/ntp.conf.dhcp` se ele existir.

2. Reinicie o serviço NTP:

```
# service ntp restart
```

Verifique a operação

Recomendamos que você verifique a sincronização NTP antes de prosseguir. Alguns nodos, particularmente aqueles que referenciam o nodo controlador, podem levar vários minutos para sincronizar.

1. Execute esse comando no nodo *controlador*:

```
# ntpq -c peers
     remote           refid      st t when poll reach   delay   offset
 jitter
==============================================================================
====
*ntp-server1     192.0.2.11       2 u  169 1024  377    1.901   -0.611
 5.483
+ntp-server2     192.0.2.12       2 u  887 1024  377    0.922   -0.246
 2.864
```

O conteúdo na coluna *remote* deve indicar o hostname ou o endereço IP de um ou mais servidores NTP.

Nota

O conteúdo na coluna *refid* tipicamente referencia endereços IP de servidores upstream.

2. Execute esse comando no nodo *controlador*:

```
# ntpq -c assoc
ind assid status  conf reach auth condition  last_event cnt
===========================================================
  1 20487  961a   yes   yes  none  sys.peer    sys_peer  1
  2 20488  941a   yes   yes  none candidate    sys_peer  1
```

O conteúdo na coluna *condition* deve indicar o `sys.peer` para pelo menos um servidor.

3. Execute esse comando em *todos os outros* nodos:

```
# ntpq -c peers
     remote           refid      st t when poll reach   delay   offset
 jitter
==============================================================================
====
*controller      192.0.2.21       3 u   47   64   37    0.308   -0.251
 0.079
```

O conteúdo na coluna *remote* deve indicar o hostname do nodo controlador.

Nota

O conteúdo na coluna *refid* tipicamente referencia endereços IP de servidores upstream.

4. Execute esse comando em *todos os outros* nodos:

```
# ntpq -c assoc
ind assid status  conf reach auth condition  last_event cnt
===========================================================
  1 21181  963a   yes   yes  none  sys.peer    sys_peer  3
```

O conteúdo na coluna *condition* deve indicar o `sys.peer`.

Pacotes OpenStack

As distribuições lançam pacotes OPenStack como parte da distribuição ou utilizando outros métodos, por causa dos diferentes calendários de lançamentos. Execute esses procedimentos em todos os nodos.

Nota

Desabilite ou remova quaisquer serviços de atualização automática porque eles podem impactar seu ambiente OpenStack.

Habilitar o repositório OpenStack

- Instale o repositório e o arquivos de chaves do Ubuntu:

```
# apt-get install ubuntu-cloud-keyring
# echo "deb http://ubuntu-cloud.archive.canonical.com/ubuntu" \
  "trusty-updates/juno main" > /etc/apt/sources.list.d/cloudarchive-juno.
list
```

Para finalizar a instalação

- Atualize os pacotes em seu sistema:

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

Nota

Se o processo de atualização inclui um novo kernel, reinicialize seu sistema para ativá-lo.

Banco de dados

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.

Instalar e configurar o servidor de banco de dados

1. Instale os pacotes:

Nota

A biblioteca Python MySQL é compatível com o MariaDB.

```
# apt-get install mariadb-server python-mysqldb
```

2. Escolha uma senha adequada para a conta root do banco de dados.

3. Edite o arquivo `/etc/mysql/my.cnf` e complete as seguintes ações:

a. Na seção `[mysqld]`, defina a chave `bind-address` para o IP de gerenciamento do nodo controlador para habilitar o acesso pelos outros nodos via rede de gerenciamento:

```
[mysqld]
...
bind-address = 10.0.0.11
```

b. Na seção `[mysqld]`, defina as seguintes chaves para habilitar opções úteis e o conjunto de caracteres UTH-8:

```
[mysqld]
...
default-storage-engine = innodb
innodb_file_per_table
collation-server = utf8_general_ci
init-connect = 'SET NAMES utf8'
character-set-server = utf8
```

Para finalizar a instalação

1. Reinicie o serviço de banco de dados:

```
# service mysql restart
```

2. Proteja o serviço de banco de dados:

```
# mysql_secure_installation
```

Servidor de mensagens

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

- RabbitMQ
- Qpid
- ZeroMQ

Para instalar o serviço de intermediador de mensagem RabbitMQ

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

Par configurar o serviço de intermediador de mensagem

- The message broker creates a default account that uses `guest` for the username and password. To simplify installation of your test environment, we recommend that you use this account, but change the password for it.

  Execute o seguinte comando:

Substitua *RABBIT_PASS* com uma senha adequada.

```
# rabbitmqctl change_password guest RABBIT_PASS
Changing password for user "guest" ...
...done.
```

Você deve configurar a chave do `rabbit_password` no arquivo de configuração para cada serviço OpenStack que utiliza o intermediador de mensagem.

Nota

For production environments, you should create a unique account with suitable password. For more information on securing the message broker, see the documentation.

If you decide to create a unique account with suitable password for your test environment, you must configure the `rabbit_userid` and `rabbit_password` keys in the configuration file of each OpenStack service that uses the message broker.

Parabéns, agora você está pronto para instalar os serviços OpenStack!

Capítulo 3. Adicione o serviço de Identidade

Índice

OpenStack Identity concepts

The OpenStack*Identity Service* performs the following functions:

- Tracking users and their permissions.
- Providing a catalog of available services with their API endpoints.

When installing OpenStack Identity service, you must register each service in your OpenStack installation. Identity service can then track which OpenStack services are installed, and where they are located on the network.

To understand OpenStack Identity, you must understand the following concepts:

User — Digital representation of a person, system, or service who uses OpenStack cloud services. The Identity service validates that incoming requests are made by the user who claims to be making the call. Users have a login and may be assigned tokens to access resources. Users can be directly assigned to a particular tenant and behave as if they are contained in that tenant.

Credentials — Data that confirms the user's identity. For example: user name and password, user name and API key, or an authentication token provided by the Identity Service.

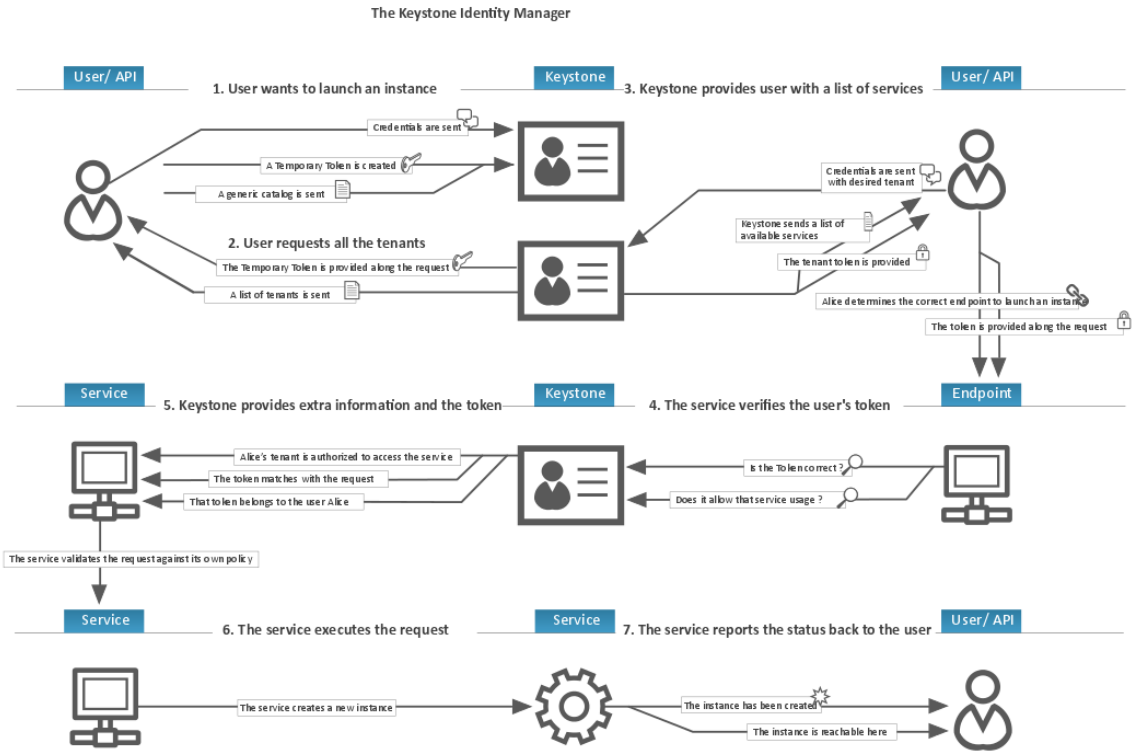
Authentication — The process of confirming the identity of a user. OpenStack Identity confirms an incoming request by validating a set of credentials supplied by the user.

These credentials are initially a user name and password, or a user name and API key. When user credentials are validated, OpenStack Identity issues an authentication token which the user provides in subsequent requests.

Token — An alpha-numeric string of text used to access OpenStack APIs and resources. A token may be revoked at any time and is valid for a finite duration.

	While OpenStack Identity supports token-based authentication in this release, the intention is to support additional protocols in the future. Its main purpose is to be an integration service, and not aspire to be a full-fledged identity store and management solution.
Tenant	A container used to group or isolate resources. Tenants also group or isolate identity objects. Depending on the service operator, a tenant may map to a customer, account, organization, or project.
Service	An OpenStack service, such as Compute (nova), Object Storage (swift), or Image Service (glance). It provides one or more endpoints in which users can access resources and perform operations.
Endpoint	A network-accessible address where you access a service, usually a URL address. If you are using an extension for templates, an endpoint template can be created, which represents the templates of all the consumable services that are available across the regions.
Role	A personality with a defined set of user rights and privileges to perform a specific set of operations. In the Identity service, a token that is issued to a user includes the list of roles. Services that are being called by that user determine how they interpret the set of roles a user has and to which operations or resources each role grants access.
Keystone Client	A command line interface for the OpenStack Identity API. For example, users can run the **keystone service-create** and **keystone endpoint-create** commands to register services in their OpenStack installations.

The following diagram shows the OpenStack Identity process flow:



Instalar e configurar

Esta seção descreve com instalar e configurar o serviço de Identidade do OpenStack no nó do controlador.

Para configurar pre-requisitos

Antes de você configurar o serviço de Identidade do OpenStack, você deve criar um banco de dados e um token de administração.

1. Para criar a base de dados, complete estes passos:

   a. Utilize o cliente de acesso a banco de dados para acessar o servidor de banco de dados como usuário `root`:

   ```
   $ mysql -u root -p
   ```

   b. Crie a base de dados do `keystone`:

   ```
   CREATE DATABASE keystone;
   ```

   c. Conceda os acessos adequados para a base de dados do `keystone`:

   ```
   GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' \
     IDENTIFIED BY 'KEYSTONE_DBPASS';
   GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' \
     IDENTIFIED BY 'KEYSTONE_DBPASS';
   ```

Substitua *KEYSTONE_DBPASS* com uma senha adequada.

d. Saia do cliente de acesso a banco de dados.

2. Gere um valor aleatório para utilizar como token de administração durante a configuração inicial:

```
# openssl rand -hex 10
```

Para instalar e configurar os componentes

1. Execute os seguintes comandos para instalar os pacotes:

```
# apt-get install keystone python-keystoneclient
```

2. Edite o arquivo `/etc/keystone/keystone.conf` e complete as seguintes ações:

a. Na seção `[DEFAULT]`, defina o valor do token de administração inicial:

```
[DEFAULT]
...
admin_token = ADMIN_TOKEN
```

Substitua *ADMIN_TOKEN* com o valor aleatório que você gerou no passo anterior.

b. Na seção `[database]`, configure o acesso ao banco de dados:

```
[database]
...
connection = mysql://keystone:KEYSTONE_DBPASS@controlador/keystone
```

Substitua *KEYSTONE_DBPASS* com a senha que você escolheu para o banco de dados.

c. Na seção `[token]`, configure o provedor do token UUID e o driver SQL:

```
[token]
...
provider = keystone.token.providers.uuid.Provider
driver = keystone.token.persistence.backends.sql.Token
```

d. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção `[DEFAULT]` :

```
[DEFAULT]
...
verbose = True
```

3. Popule a base de dados do Serviço de Identidade:

```
# su -s /bin/sh -c "keystone-manage db_sync" keystone
```

Para finalizar a instalação

1. Reinicie o serviço de Identidade:

```
# service keystone restart
```

2. Por padrão, os pacotes do Ubuntu criam um banco de dados SQLite.

   Devido esta configuração utilizar um servidor de banco de dados SQL, você pode remover o arquivo de banco de dados SQLite:

```
# rm -f /var/lib/keystone/keystone.db
```

3. Por padrão, o serviço de Identidade armazena os tokens expirados na base de dados indefinidamente. A acumulação de tokens expirados aumenta consideravelmente o tamanho da base de dados e pode degradar o desempenho do sistema, particularmente em ambientes com recursos limitados.

   Recomendamos que você utilize o `cron` para configurar uma tarefa periódica que limpa os tokens expirados de hora em hora:

```
# (crontab -l -u keystone 2>&1 | grep -q token_flush) || \
  echo '@hourly /usr/bin/keystone-manage token_flush >/var/log/keystone/
keystone-tokenflush.log 2>&1' \
  >> /var/spool/cron/crontabs/keystone
```

Criar tenants, usuários e papéis

Após instalar o serviço de Identidade, crie (projetos) *tenants*, *usuários*, e *papéis* para o seu ambiente. Você deve utilizar o token administrativo temporário que você criou em "Instalar e configurar" [32] e configurar manualmente a localização (endpoint) do serviço de Identidade antes de executar comandos **keystone** .

Você pode passar o valor do token de administração para o comando **keystone** com a opção `--os-token` ou definir a variável de ambiente temporária `OS_SERVICE_TOKEN`. Da mesma forma, você pode passar a localização do sistema de Identidade para o comando **keystone** com a opção **keystone** ou definir a variável de ambiente temporária `OS_SERVICE_ENDPOINT`. Este guia utiliza variáveis de ambiente para reduzir o comprimento do comando.

For more information, see the Operations Guide - Managing Project and Users.

Para configurar pre-requisitos

1. Configure o token de administração:

```
$ export OS_SERVICE_TOKEN=ADMIN_TOKEN
```

   Substitua cinder-common com o token de administração que você gerou em "Instalar e configurar" [32]. Por exemplo

```
$ export OS_SERVICE_TOKEN=294a4c8a8a475f9b9836
```

2. Configure o endpoint:

```
$ export OS_SERVICE_ENDPOINT=http://controlador:35357/v2.0
```

Para criar tenants, usuários, e papéis

1. Crie um tenant administrativo, um usuário, e um papel para operações administrativas em seu ambiente:

a. Crie o tenant `admin`:

```
$ keystone tenant-create --name admin --description "Admin Tenant"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |           Admin Tenant           |
|   enabled   |               True               |
|      id     | 6f4c1e4cbfef4d5a8a1345882fbca110 |
|     name    |              admin               |
+-------------+----------------------------------+
```

Nota

O OpenStack gera IDs dinâmicamente, portanto você deve ver diferentes valores a partir da saída de comandos.

b. Crie o usuário `admin`:

```
$ keystone user-create --name admin --pass ADMIN_PASS --
email EMAIL_ADDRESS
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |        admin@example.com         |
| enabled  |               True               |
|    id    | ea8c352d253443118041c9c8b8416040 |
|   name   |              admin               |
| username |              admin               |
+----------+----------------------------------+
```

Substitua *ADMIN_PASS* com uma senha adequada e *EMAIL_ADDRESS* com um endereço de e-mail adequado.

c. Crie o papel `admin`:

```
$ keystone role-create --name admin
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|    id    | bff3a6083b714fa29c9344bf8930d199 |
|   name   |              admin               |
+----------+----------------------------------+
```

d. Adicione o papel `admin` ao tenant e usuário `admin`:

```
$ keystone user-role-add --user admin --tenant admin --role admin
```

Nota

Este comando não retorna resultados.

Nota

Any roles that you create must map to roles specified in the `policy.json` file included with each OpenStack service. The default policy for most servi-

ces grants administrative access to the `admin` role. For more information, see the Operations Guide - Managing Projects and Users.

2. Crie um usuário e um tenant de demonstração para operações típicas em seu ambiente:

   a. Crie o tenant `demo`:

```
$ keystone tenant-create --name demo --description "Demo Tenant"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |           Demo Tenant            |
|   enabled   |               True               |
|      id     | 4aa51bb942be4dd0ac0555d7591f80a6 |
|     name    |               demo               |
+-------------+----------------------------------+
```

Nota

Não repita este passo quando estiver criando usuários adicionais para este tenant.

   b. Crie o usuário `demo` sob o tenant `demo`:

```
$ keystone user-create --name demo --tenant demo --pass DEMO_PASS --
email EMAIL_ADDRESS
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |         demo@example.com         |
| enabled  |               True               |
|    id    | 7004dfa0dda84d63aef81cf7f100af01 |
|   name   |               demo               |
| tenantId | 4aa51bb942be4dd0ac0555d7591f80a6 |
| username |               demo               |
+----------+----------------------------------+
```

Substitua *DEMO_PASS* com uma senha adequada e *EMAIL_ADDRESS* com um endereço de e-mail adequado.

Nota

Utilizando a opção `--tenant` automaticamente atribuirá o papel `_member_` a um usuário. Esta opção também irá criar o papel `_member_` se ele não existir.

Nota

Você pode repetir este procedimento para criar tenants e usuários adicionais.

3. OpenStack services also require a tenant, user, and role to interact with other services. Each service typically requires creating one or more unique users with the `admin` role under the `service` tenant.

- Crie o tenant `service`:

```
$ keystone tenant-create --name service --description "Service Tenant"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |          Service Tenant          |
|   enabled   |               True               |
|      id     | 6b69202e1bf846a4ae50d65bc4789122 |
|     name    |             service              |
+-------------+----------------------------------+
```

Criar a entidade de serviço e o endpoint de API

Depois de criar os tenants, usuários, e papéis, você deve criar a entidade de *serviço*.

Para configurar pre-requisitos

- Defina as variáveis de ambiente `OS_SERVICE_TOKEN` e `OS_SERVICE_ENDPOINT`, como descrito em "Criar tenants, usuários e papéis" [34].

Para criar a entidade de serviço e o endpoint de API

1. The Identity service manages a catalog of services in your OpenStack environment. Services use this catalog to locate other services in your environment.

   Criar a entidade de serviço para o Serviço de Identidade:

```
$ keystone service-create --name keystone --type identity \
  --description "OpenStack Identity"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |        OpenStack Identity        |
|   enabled   |               True               |
|      id     | 15c11a23667e427e91bc31335b45f4bd |
|     name    |             keystone             |
|     type    |             identity             |
+-------------+----------------------------------+
```

   Nota

   Devido ao OpenStack gerar IDs dinamicamente, você verá valores diferentes para a saída de comando deste exemplo.

2. The Identity service manages a catalog of API endpoints associated with the services in your OpenStack environment. Services use this catalog to determine how to communicate with other services in your environment.

   OpenStack provides three API endpoint variations for each service: admin, internal, and public. In a production environment, the variants might reside on separate networks that service different types of users for security reasons. Also, OpenStack supports multiple regions for scalability. For simplicity, this configuration uses the management network for all endpoint variations and the `regionOne` region.

Criar o endpoint de API para o serviço de Identidade:

```
$ keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ identity / {print $2}') \
  --publicurl http://controlador:5000/v2.0 \
  --internalurl http://controlador:5000/v2.0 \
  --adminurl http://controlador:35357/v2.0 \
  --region regionOne
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
|   adminurl  |   http://controller:35357/v2.0   |
|      id     | 11f9c625a3b94a3f8e66bf4e5de2679f |
| internalurl |   http://controller:5000/v2.0    |
|  publicurl  |   http://controller:5000/v2.0    |
|    region   |           regionOne              |
|  service_id | 15c11a23667e427e91bc31335b45f4bd |
+-------------+----------------------------------+
```

Nota

Este comando referencia o ID do serviço que você criou no passo anterior.

Nota

Each service that you add to your OpenStack environment requires adding information such as API endpoints to the Identity service. The sections of this guide that cover service installation include steps to add the appropriate information to the Identity service.

Verifique a operação

Esta seção descreve como verificar a operação do serviço de Identidade.

1. Limpe as variáveis temporárias de ambiente `OS_SERVICE_TOKEN` e`OS_SERVICE_ENDPOINT`:

   ```
   $ unset OS_SERVICE_TOKEN OS_SERVICE_ENDPOINT
   ```

2. Como usuário e tenant `admin`, requisite o token de autenticação:

   ```
   $ keystone --os-tenant-name admin --os-username admin --os-
   password ADMIN_PASS \
     --os-auth-url http://controller:35357/v2.0 token-get
   ```

   Substitua *ADMIN_PASS* com a senha que você escolheu para o usuário `admin` no serviço de Identidade. Você pode precisar utilizar aspas simples (') em torno de sua senha se ela incluir caracteres especiais.

   Saída longa que inclui um valor de token verifica operações para o tenant e o usuário `admin`.

3. Como tenant e usuário `admin`, liste os tenants para verificar que o tenant e usuário `admin` pode executar comandos CLI admin-only e que o serviço de Identidade contém os tenants que você criou em "Criar tenants, usuários e papéis" [34]:

```
$ keystone --os-tenant-name admin --os-username admin --os-
password ADMIN_PASS \
  --os-auth-url http://controller:35357/v2.0 tenant-list
+----------------------------------+----------+---------+
|                id                |   name   | enabled |
+----------------------------------+----------+---------+
| 6f4c1e4cbfef4d5a8a1345882fbca110 |  admin   |   True  |
| 4aa51bb942be4dd0ac0555d7591f80a6 |   demo   |   True  |
| 6b69202e1bf846a4ae50d65bc4789122 | service  |   True  |
+----------------------------------+----------+---------+
```

Nota

Devido ao OpenStack gerar IDs dinamicamente, você verá valores diferentes para a saída de comando deste exemplo.

4. Como usuário e tenant `admin`, liste os usuários para identificar que o Serviço de Identidade contém os usuários que você criou em "Criar tenants, usuários e papéis" [34]:

```
$ keystone --os-tenant-name admin --os-username admin --os-
password ADMIN_PASS \
  --os-auth-url http://controller:35357/v2.0 user-list
+----------------------------------+---------+---------
+---------------------+
|                id                |   name  | enabled |        email
   |
+----------------------------------+---------+---------
+---------------------+
| ea8c352d253443118041c9c8b8416040 |  admin  |   True  |  admin@example.
com  |
| 7004dfa0dda84d63aef81cf7f100af01 |   demo  |   True  |  demo@example.com
   |
+----------------------------------+---------+---------
+---------------------+
```

5. Como usuário e tenant `admin`, liste os papéis para verificar que o Serviço de Identidade contém o papel que você criou em "Criar tenants, usuários e papéis" [34]:

```
$ keystone --os-tenant-name admin --os-username admin --os-
password ADMIN_PASS \
  --os-auth-url http://controller:35357/v2.0 role-list
+----------------------------------+----------+
|                id                |   name   |
+----------------------------------+----------+
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_ |
| bff3a6083b714fa29c9344bf8930d199 |  admin   |
+----------------------------------+----------+
```

6. Como usuário e tenant `demo`, solicite um token de autenticação:

```
$ keystone --os-tenant-name demo --os-username demo --os-
password DEMO_PASS \
  --os-auth-url http://controller:35357/v2.0 token-get
+-----------+----------------------------------+
|  Property |              Value               |
+-----------+----------------------------------+
|  expires  |       2014-10-10T12:51:33Z       |
|     id    | 1b87ceae9e08411ba4a16e4dada04802 |
```

```
| tenant_id | 4aa51bb942be4dd0ac0555d7591f80a6 |
|  user_id  | 7004dfa0dda84d63aef81cf7f100af01 |
+-----------+----------------------------------+
```

Substitua *DEMO_PASS* com a senha que você escolheu para o usuário `demo` no Serviço de Identidade.

7. Como usuário e tenant literal>demo

```
$ keystone --os-tenant-name demo --os-username demo --os-
password DEMO_PASS \
  --os-auth-url http://controller:35357/v2.0 user-list
You are not authorized to perform the requested action, admin_required.
 (HTTP 403)
```

Nota

Each OpenStack service references a `policy.json` file to determine the operations available to a particular tenant, user, or role. For more information, see the Operations Guide - Managing Projects and Users.

Criar scripts de ambiente de cliente OpenStack

The previous section used a combination of environment variables and command options to interact with the Identity service via the **keystone** client. To increase efficiency of client operations, OpenStack supports simple client environment scripts also known as OpenRC files. These scripts typically contain common options for all clients, but also support unique options. For more information, see the OpenStack User Guide.

Para criar os scripts

Criar script de ambiente de cliente para os tenants e usuários `admin` e `demo`. Partes futuras deste guia referem-se a esses scripts para carregar apropriadadmente as credencilas para operações de cliente.

1. Edite o arquivo `admin-openrc.sh` e adicione o seguinte conteúdo:

```
export OS_TENANT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=ADMIN_PASS
export OS_AUTH_URL=http://controlador:35357/v2.0
```

Substitua ADMIN_PASS com a senha que você escolheu para o usuário `admin` no Serviço de Identidade.

2. Edite o arquivo `demo-openrc.sh` e adicione o seguinte conteúdo:

```
export OS_TENANT_NAME=demo
export OS_USERNAME=demo
export OS_PASSWORD=DEMO_PASS
export OS_AUTH_URL=http://controlador:5000/v2.0
```

Substitua DEMO_PASS com a senha que você escolheu para o usuário `demo` no Serviço de Identidade.

Portas de Identidade

Observe as duas diferentes portas utilizadas acima. A porta 35357 é utilizada para funcões administrativas somente. A porta 5000 é para funções normais de usuário e é a mais frequentemente utilizada.

Para carregar scripts de ambiente de cliente

- Para executar clientes como um certo tenant e usuário, você pode simplesmente carregar o script de ambiente de cliente associado, antes de executá-los. Por exempo, para carregar a localização do Serviço de Identidade e do tenant de `admin` e credenciais de usuário:

```
$ source admin-openrc.sh
```

Capítulo 4. Adicionar o Serviço de Imagem

Índice

The OpenStack Image Service (glance) enables users to discover, register, and retrieve virtual machine images. It offers a *REST* API that enables you to query virtual machine image metadata and retrieve an actual image. You can store virtual machine images made available through the Image Service in a variety of locations, from simple file systems to object-storage systems like OpenStack Object Storage.

Importante

For simplicity, this guide describes configuring the Image Service to use the `file` back end, which uploads and stores in a directory on the controller node hosting the Image Service. By default, this directory is `/var/lib/glance/images/`.

Before you proceed, ensure that the controller node has at least several gigabytes of space available in this directory.

For information on requirements for other back ends, see *Configuration Reference*.

OpenStack Image Service

The OpenStack Image Service is central to Infrastructure-as-a-Service (IaaS) as shown in Figura 1.1, “Arquitetura conceitual” [2]. It accepts API requests for disk or server images, and image metadata from end users or OpenStack Compute components. It also supports the storage of disk or server images on various repository types, including OpenStack Object Storage.

A number of periodic processes run on the OpenStack Image Service to support caching. Replication services ensure consistency and availability through the cluster. Other periodic processes include auditors, updaters, and reapers.

The OpenStack Image Service includes the following components:

`glance-api`	Accepts Image API calls for image discovery, retrieval, and storage.
`glance-registry`	Stores, processes, and retrieves metadata about images. Metadata includes items such as size and type.

Security note

The registry is a private internal service meant for use by OpenStack Image Service. Do not disclose it to users.

Database	Stores image metadata and you can choose your database depending on your preference. Most deployments use MySQL or SQLite.
Storage repository for image files	Various repository types are supported including normal file systems, Object Storage, RADOS block devices, HTTP, and Amazon S3. Note that some repositories will only support read-only usage.

Instalar e configurar

Esta seção descreve como instalar e configurar o Serviço de Imagem, apelidado de glance, no nodo controlador. Para simplificar, esta configuração armazena imagens no sistema de arquivo local.

Nota

Esta seção assume a instalação, configuração e operação correta do serviço de Identidade como descrito em "Instalar e configurar" [32] e "Verifique a operação" [38].

Para configurar pre-requisitos

Antes de você instalar e configurar o serviço de Imagem, você deve criar uma base de dados e credenciais do serviço de Identidade, incluindo os endpoints.

1. Para criar a base de dados, complete estes passos:

   a. Utilize o cliente de acesso a banco de dados para acessar o servidor de banco de dados como usuário `root`:

   ```
   $ mysql -u root -p
   ```

   b. Crie a base dados do `glance`:

   ```
   CREATE DATABASE glance;
   ```

   c. Conceda acessos apropriados à base de dados do `glance`:

   ```
   GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' \
     IDENTIFIED BY 'GLANCE_DBPASS';
   GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' \
     IDENTIFIED BY 'GLANCE_DBPASS';
   ```

   Substitua *GLANCE_DBPASS* com uma senha adequada.

   d. Saia do cliente de acesso a banco de dados.

2. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

```
$ source admin-openrc.sh
```

3. Para criar as credenciais do Serviço de Identidade, complete estes passos:

   a. Crie o usuário `glance`:

```
$ keystone user-create --name glance --pass GLANCE_PASS
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |                                  |
| enabled  |               True               |
|    id    | f89cca5865dc42b18e2421fa5f5cce66 |
|   name   |              glance              |
| username |              glance              |
+----------+----------------------------------+
```

   Substitua *GLANCE_PASS* com uma senha adequada.

   b. Ligue o usuário `glance` ao tenant de `serviço` e ao papel `admin`:

```
$ keystone user-role-add --user glance --tenant service --role admin
```

Nota

Este comando não retorna resultados.

   c. Crie o serviço `glance`:

```
$ keystone service-create --name glance --type image \
  --description "OpenStack Image Service"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |     OpenStack Image Service      |
|   enabled   |               True               |
|      id     | 23f409c4e79f4c9e9d23d809c50fbacf |
|     name    |              glance              |
|     type    |              image               |
+-------------+----------------------------------+
```

4. Crie os endpoints do Serviço de Identidade:

```
$ keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ image / {print $2}') \
  --publicurl http://controlador:9292 \
  --internalurl http://controlador:9292 \
  --adminurl http://controlador:9292 \
  --region regionOne
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
|   adminurl  |   http://controller:9292         |
|      id     | a2ee818c69cb475199a1ca108332eb35 |
| internalurl |   http://controller:9292         |
|  publicurl  |   http://controller:9292         |
```

```
|    region    |           regionOne              |
|  service_id | 23f409c4e79f4c9e9d23d809c50fbacf |
+-------------+----------------------------------+
```

Para instalar e configurar os componentes do Serviço de Imagem

1. Instale os pacotes:

```
# apt-get install glance python-glanceclient
```

2. Edite o arquivo `/etc/glance/glance-api.conf` e complete as seguintes ações:

   a. Na seção `[database]`, configure o acesso ao banco de dados:

```
[database]
...
connection = mysql://glance:GLANCE_DBPASS@controlador/glance
```

   Substitua *GLANCE_DBPASS* com a senha que você escolheu para a base de dados do Serviço de Imagem.

   b. Nas seções `[keystone_authtoken]` e `[paste_deploy]`, configure o acesso ao serviço de Identidade:

```
[keystone_authtoken]
...
auth_uri = http://controlador:5000/v2.0
identity_uri = http://controlador:35357
admin_tenant_name = service
admin_user = glance
admin_password = GLANCE_PASS

[paste_deploy]
...
flavor = keystone
```

   Substitua *GLANCE_PASS* com a senha que você escolheu para o usuário `glance` no serviço de Identidade.

Nota

Comente quaisquer opções `auth_host`, `auth_port`, e `auth_protocol`, porque a opção `identity_uri` as substitui.

   c. Na seção `[glance_store]`, configure o sistema de arquivo local de armazenamento e localização dos arquivos de Imagem:

```
[glance_store]
...
default_store = file
filesystem_store_datadir = /var/lib/glance/images/
```

   d. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção `[DEFAULT]` :

```
[DEFAULT]
...
verbose = True
```

3. Edite o arquivo `/etc/glance/glance-registry.conf` e complete as seguintes ações:

   a. Na seção `[database]`, configure o acesso ao banco de dados:

   ```
   [database]
   ...
   connection = mysql://glance:GLANCE_DBPASS@controlador/glance
   ```

   Substitua *GLANCE_DBPASS* com a senha que você escolheu para a base de dados do Serviço de Imagem.

   b. Nas seções `[keystone_authtoken]` e `[paste_deploy]`, configure o acesso ao serviço de Identidade:

   ```
   [keystone_authtoken]
   ...
   auth_uri = http://controlador:5000/v2.0
   identity_uri = http://controlador:35357
   admin_tenant_name = service
   admin_user = glance
   admin_password = GLANCE_PASS

   [paste_deploy]
   ...
   flavor = keystone
   ```

   Substitua *GLANCE_PASS* com a senha que você escolheu para o usuário `glance` no serviço de Identidade.

   Nota

   Comente quaisquer opções `auth_host`, `auth_port`, e `auth_protocol`, porque a opção `identity_uri` as substitui.

   c. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção `[DEFAULT]` :

   ```
   [DEFAULT]
   ...
   verbose = True
   ```

4. Popule a base de dados do Serviço de Imagem:

   ```
   # su -s /bin/sh -c "glance-manage db_sync" glance
   ```

Para finalizar a instalação

1. Reinicie os serviços do Serviço de Imagem:

   ```
   # service glance-registry restart
   # service glance-api restart
   ```

2. Por padrão, os pacotes do Ubuntu criam uma base de dados SQLite.

   Devido esta configuração utilizar um servidor de banco de dados SQL, você pode remover o arquivo de banco de dados SQLite:

```
# rm -f /var/lib/glance/glance.sqlite
```

Verifique a operação

This section describes how to verify operation of the Image Service using CirrOS, a small Linux image that helps you test your OpenStack deployment.

For more information about how to download and build images, see *OpenStack Virtual Machine Image Guide*. For information about how to manage images, see the *OpenStack User Guide*.

1. Crie e altere para um diretório local temporário:

```
$ mkdir /tmp/images
$ cd /tmp/images
```

2. Baixe a imagem para o diretório temporário local:

```
$ wget http://cdn.download.cirros-cloud.net/0.3.3/cirros-0.3.3-x86_64-
disk.img
```

3. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

```
$ source admin-openrc.sh
```

4. Envie a imagem para o Serviço de Imagem:

```
$ glance image-create --name "cirros-0.3.3-x86_64" --file cirros-0.3.3-
x86_64-disk.img \
  --disk-format qcow2 --container-format bare --is-public True --progress
[=============================>] 100%
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| checksum         | 133eae9fb1c98f45894a4e60d8736619     |
| container_format | bare                                 |
| created_at       | 2014-10-10T13:14:42                  |
| deleted          | False                                |
| deleted_at       | None                                 |
| disk_format      | qcow2                                |
| id               | acafc7c0-40aa-4026-9673-b879898e1fc2 |
| is_public        | True                                 |
| min_disk         | 0                                    |
| min_ram          | 0                                    |
| name             | cirros-0.3.3-x86_64                  |
| owner            | ea8c352d253443118041c9c8b8416040     |
| protected        | False                                |
| size             | 13200896                             |
| status           | active                               |
| updated_at       | 2014-10-10T13:14:43                  |
| virtual_size     | None                                 |
+------------------+--------------------------------------+
```

For information about the parameters for the **glance image-create** command, see Image Service command-line client in the *OpenStack Command-Line Interface Reference*.

For information about disk and container formats for images, see Disk and container formats for images in the *OpenStack Virtual Machine Image Guide*.

Nota

Como o ID de imagem retornado é gerado dinamicamente, sua implantação gera um ID diferente daquele mostrado neste exemplo.

5. Confirme o upload da imagem e valide os atributos:

```
$ glance image-list
+--------------------------------------+---------------------
+-------------+-----------------+----------+--------+
| ID                                   | Name                | Disk Format
 | Container Format | Size     | Status |
+--------------------------------------+---------------------
+-------------+-----------------+----------+--------+
| acafc7c0-40aa-4026-9673-b879898e1fc2 | cirros-0.3.3-x86_64 | qcow2
 | bare             | 13200896 | active |
+--------------------------------------+---------------------
+-------------+-----------------+----------+--------+
```

6. Remova o diretório local temporário:

```
$ rm -r /tmp/images
```

Capítulo 5. Adicione o serviço de Computação

Índice

OpenStack Compute

Use OpenStack Compute to host and manage cloud computing systems. OpenStack Compute is a major part of an Infrastructure-as-a-Service (IaaS) system. The main modules are implemented in Python.

OpenStack Compute interacts with OpenStack Identity for authentication, OpenStack Image Service for disk and server images, and OpenStack dashboard for the user and administrative interface. Image access is limited by projects, and by users; quotas are limited per project (the number of instances, for example). OpenStack Compute can scale horizontally on standard hardware, and download images to launch instances.

OpenStack Compute consists of the following areas and their components:

API

`nova-api service`
Accepts and responds to end user compute API calls. The service supports the OpenStack Compute API, the Amazon EC2 API, and a special Admin API for privileged users to perform administrative actions. It enforces some policies and initiates most orchestration activities, such as running an instance.

`nova-api-metadata` service
Accepts metadata requests from instances. The `nova-va-api-metadata` service is generally used when you run in multi-host mode with `nova-network` installations. For details, see Metadata service in the *OpenStack Cloud Administrator Guide*.

On Debian systems, it is included in the `nova-api` package, and can be selected through debconf.

Compute core

`nova-compute` service
A worker daemon that creates and terminates virtual machine instances through hypervisor APIs. For example:

	• XenAPI for XenServer/XCP • libvirt for KVM or QEMU • VMwareAPI for VMware Processing is fairly complex. Basically, the daemon accepts actions from the queue and performs a series of system commands such as launching a KVM instance and updating its state in the database.
`nova-scheduler` service	Takes a virtual machine instance request from the queue and determines on which compute server host it runs.
`nova-conductor` module	Mediates interactions between the `nova-compute` service and the database. It eliminates direct accesses to the cloud database made by the `nova-compute` service. The `nova-conductor` module scales horizontally. However, do not deploy it on nodes where the `nova-compute` service runs. For more information, see A new Nova service: nova-conductor.

Networking for VMs

`nova-network` worker daemon	Similar to the `nova-compute` service, accepts networking tasks from the queue and manipulates the network. Performs tasks such as setting up bridging interfaces or changing IPtables rules.

Console interface

nova-consoleauth daemon	Authorizes tokens for users that console proxies provide. See `nova-novncproxy` and `nova-xvpnvcproxy`. This service must be running for console proxies to work. You can run proxies of either type against a single `nova-consoleauth` service in a cluster configuration. For information, see About nova-consoleauth.
nova-novncproxy daemon	Provides a proxy for accessing running instances through a VNC connection. Supports browser-based novnc clients.
nova-spicehtml5proxy daemon	Provides a proxy for accessing running instances through a SPICE connection. Supports browser-based HTML5 client.
nova-xvpnvncproxy daemon	Provides a proxy for accessing running instances through a VNC connection. Supports an OpenStack-specific Java client.
nova-cert daemon	x509 certificates.

Image management (EC2 scenario)

nova-objectstore daemon	An S3 interface for registering images with the OpenStack Image Service. Used primarily for installations that must support euca2ools. The euca2ools tools talk to `nova-objectstore` in *S3 language*, and `nova-objectstore` translates S3 requests into Image Service requests.
euca2ools client	A set of command-line interpreter commands for managing cloud resources. Although it is not an OpenStack module, you can configure `nova-api` to support this EC2 interface. For more information, see the Eucalyptus 3.4 Documentation.

Command-line clients and other interfaces

nova client	Enables users to submit commands as a tenant administrator or end user.

Other components

The queue	A central hub for passing messages between daemons. Usually implemented with RabbitMQ, but can be implemented with an AMQP message queue, such as Apache Qpid or Zero MQ.
SQL database	Stores most build-time and run-time states for a cloud infrastructure, including: • Available instance types • Instances in use

- Available networks
- Projects

Theoretically, OpenStack Compute can support any database that SQL-Alchemy supports. Common databases are SQLite3 for test and development work, MySQL, and PostgreSQL.

Instalar e configurar o nodo controlador

Esta seção descreve como instalar e configurar o serviço de Computação, apelidado de nova, no nodo controlador.

Para configurar pre-requisitos

Antes de você instalar e configurar a Computação, você deve criar credenciais de banco de dados e de Serviço de Identidade, incluindo os enpoints.

1. Para criar a base de dados, complete estes passos:

   a. Utilize o cliente de acesso a banco de dados para acessar o servidor de banco de dados como usuário `root`:

   ```
   $ mysql -u root -p
   ```

   b. Crie a base de dados do `nova`:

   ```
   CREATE DATABASE nova;
   ```

   c. Conceda acesso apropriado à base de dados do `nova`:

   ```
   GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' \
     IDENTIFIED BY 'NOVA_DBPASS';
   GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' \
     IDENTIFIED BY 'NOVA_DBPASS';
   ```

   Substitua *NOVA_DBPASS* com uma senha adequada.

   d. Saia do cliente de acesso a banco de dados.

2. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

   ```
   $ source admin-openrc.sh
   ```

3. Para criar as credenciais do Serviço de Identidade, complete estes passos:

   a. Crie o usuário `nova`:

   ```
   $ keystone user-create --name nova --pass NOVA_PASS
   +----------+----------------------------------+
   | Property |              Value               |
   +----------+----------------------------------+
   |  email   |                                  |
   | enabled  |               True               |
   ```

```
|    id    | 387dd4f7e46d4f72965ee99c76ae748c |
|   name   |               nova               |
| username |               nova               |
+----------+----------------------------------+
```

Substitua *NOVA_PASS* com uma senha adequada.

b. Ligue o usuário `nova` ao `serviço` de tenant e ao papel `admin`:

```
$ keystone user-role-add --user nova --tenant service --role admin
```

Nota

Este comando não retorna resultados.

c. Crie o serviço `nova`:

```
$ keystone service-create --name nova --type compute \
  --description "OpenStack Compute"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |        OpenStack Compute         |
|   enabled   |               True               |
|      id     | 6c7854f52ce84db795557ebc0373f6b9 |
|     name    |               nova               |
|     type    |             compute              |
+-------------+----------------------------------+
```

4. Crie os endpoints do serviço de Computação:

```
$ keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ compute / {print $2}') \
  --publicurl http://controlador:8774/v2/%\(tenant_id\)s \
  --internalurl http://controlador:8774/v2/%\(tenant_id\)s \
  --adminurl http://controlador:8774/v2/%\(tenant_id\)s \
  --region regionOne
+-------------+-----------------------------------------+
|   Property  |                  Value                  |
+-------------+-----------------------------------------+
|   adminurl  | http://controller:8774/v2/%(tenant_id)s |
|      id     |    c397438bd82c41198ec1a9d85cb7cc74     |
| internalurl | http://controller:8774/v2/%(tenant_id)s |
|  publicurl  | http://controller:8774/v2/%(tenant_id)s |
|    region   |                regionOne                |
|  service_id |    6c7854f52ce84db795557ebc0373f6b9     |
+-------------+-----------------------------------------+
```

Para instalar e configurar os componentes do controlador de Computação

1. Instale os pacotes:

```
# apt-get install nova-api nova-cert nova-conductor nova-consoleauth \
  nova-novncproxy nova-scheduler python-novaclient
```

2. Edite o aqruivo `/etc/nova/nova.conf` e complete as seguintes ações:

a. Adicione uma seção `[database]`, e configure o acesso a banco de dados:

```
[database]
...
connection = mysql://nova:NOVA_DBPASS@controller/nova
```

Substitua *NOVA_DBPASS* com a senha que você escolheu para a base de dados da Computação.

b. Na seção `[DEFAULT]`, configure o acesso ao intermediador RabbitMQ:

```
[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controlador
rabbit_password = RABBIT_PASS
```

Substitua *RABBIT_PASS* com a senha que você escolheu para a conta `guest` no RabbitMQ.

c. Nas seções `[DEFAULT]` e `[keystone_authtoken]`, configure o acesso do Serviço de Identidade:

```
[DEFAULT]
...
auth_strategy = keystone

[keystone_authtoken]
...
auth_uri = http://controlador:5000/v2.0
identity_uri = http://controlador:35357
admin_tenant_name = service
admin_user = nova
admin_password = NOVA_PASS
```

Substitua *NOVA_PASS* com a senha que você escolheu para o usuário `nova` no serviço de Identidade.

Nota

Comente quaisquer opções `auth_host`, `auth_port`, e `auth_protocol`, porque a opção `identity_uri` as substitui.

d. Na seção `[DEFAULT]`, configure a opção `my_ip` para utilizar o endereço IP da interface de gerenciamento do nodo controlador:

```
[DEFAULT]
...
my_ip = 10.0.0.11
```

e. Na seção `[DEFAULT]`, configure o proxy VNC para utilizar o endereço IP da interface de gerenciamento do nodo controlador:

```
[DEFAULT]
...
vncserver_listen = 10.0.0.11
vncserver_proxyclient_address = 10.0.0.11
```

f. Na seção `[glance]`, configure a localização do Serviço de Imagem:

```
[glance]
...
host = controlador
```

g. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção [DEFAULT] :

```
[DEFAULT]
...
verbose = True
```

3. Popule a base de dados da Computação:

```
# su -s /bin/sh -c "nova-manage db sync" nova
```

Para finalizar a instalação

1. Reinicie o serviços de Computação:

```
# service nova-api restart
# service nova-cert restart
# service nova-consoleauth restart
# service nova-scheduler restart
# service nova-conductor restart
# service nova-novncproxy restart
```

2. Por padrão, os pacotes do Ubuntu criam uma base de dados SQLite.

   Devido esta configuração utilizar um servidor de banco de dados SQL, você pode remover o arquivo de banco de dados SQLite:

```
# rm -f /var/lib/nova/nova.sqlite
```

Instale e configure um nodo de Computação

This section describes how to install and configure the Compute service on a compute node. The service supports several *hypervisors* to deploy *instances* or *VMs*. For simplicity, this configuration uses the *QEMU* hypervisor with the *KVM* extension on compute nodes that support hardware acceleration for virtual machines. On legacy hardware, this configuration uses the generic QEMU hypervisor. You can follow these instructions with minor modifications to horizontally scale your environment with additional compute nodes.

Nota

This section assumes that you are following the instructions in this guide step-by-step to configure the first compute node. If you want to configure additional compute nodes, prepare them in a similar fashion to the first compute node in the example architectures section using the same networking service as your existing environment. For either networking service, follow the NTP configuration and OpenStack packages instructions. For OpenStack Networking (neutron), also follow the OpenStack Networking compute node instructions. For legacy networking (nova-network), also follow the legacy networking compute node instructions. Each additional compute node requires unique IP addresses.

Instalar e configurar os componentes de hypervisor de Computação

1. Instale os pacotes:

```
# apt-get install nova-compute sysfsutils
```

2. Edite o aqruivo `/etc/nova/nova.conf` e complete as seguintes ações:

   a. Na seção `[DEFAULT]`, configure o acesso ao intermediador RabbitMQ:

```
[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controlador
rabbit_password = RABBIT_PASS
```

   Substitua *RABBIT_PASS* com a senha que você escolheu para a conta `guest` no RabbitMQ.

   b. Nas seções `[DEFAULT]` e `[keystone_authtoken]`, configure o acesso do Serviço de Identidade:

```
[DEFAULT]
...
auth_strategy = keystone

[keystone_authtoken]
...
auth_uri = http://controlador:5000/v2.0
identity_uri = http://controlador:35357
admin_tenant_name = service
admin_user = nova
admin_password = NOVA_PASS
```

   Substitua *NOVA_PASS* com a senha que você escolheu para o usuário `nova` no serviço de Identidade.

Nota

Comente quaisquer opções `auth_host`, `auth_port`, e `auth_protocol`, porque a opção `identity_uri` as substitui.

   c. Na seção `[DEFAULT]`, configure a opção `my_ip`:

```
[DEFAULT]
...
my_ip = MANAGEMENT_INTERFACE_IP_ADDRESS
```

   Replace *MANAGEMENT_INTERFACE_IP_ADDRESS* with the IP address of the management network interface on your compute node, typically 10.0.0.31 for the first node in the example architecture.

   d. Na seção `[DEFAULT]`, habilite e configure o acesso de console remoto:

```
[DEFAULT]
...
vnc_enabled = True
vncserver_listen = 0.0.0.0
vncserver_proxyclient_address = MANAGEMENT_INTERFACE_IP_ADDRESS
novncproxy_base_url = http://controlador:6080/vnc_auto.html
```

The server component listens on all IP addresses and the proxy component only listens on the management interface IP address of the compute node. The base URL indicates the location where you can use a web browser to access remote consoles of instances on this compute node.

Replace *MANAGEMENT_INTERFACE_IP_ADDRESS* with the IP address of the management network interface on your compute node, typically 10.0.0.31 for the first node in the example architecture.

Nota

If the web browser to access remote consoles resides on a host that cannot resolve the `controller` hostname, you must replace *`controller`* with the management interface IP address of the controller node.

e. Na seção `[glance]`, configure a localização do Serviço de Imagem:

```
[glance]
...
host = controlador
```

f. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção `[DEFAULT]` :

```
[DEFAULT]
...
verbose = True
```

Para finalizar a instalação

1. Determine se seu nodo de Computação suporta aceleração de hardware para máquinas viruais:

```
$ egrep -c '(vmx|svm)' /proc/cpuinfo
```

Se este comando retorna um valor de *um ou maior*, seu nodo de Computação suporta aceleração de hardware a qual, tipicamente não requer configuração adicional.

Se este comando retorna um valor de *zero*, seu nodo de Computação não suporta aceleração de hardware e você deve configurar `libvirt` para utilizar QEMU em vez de KVM.

- Edite a seção `[libvirt]` nos arquivos `/etc/nova/nova-compute.conf`, como segue:

```
[libvirt]
...
virt_type = qemu
```

2. Reinicie o serviço de Computação:

```
# service nova-compute restart
```

3. Por padrão, os pacotes do Ubuntu criam uma base de dados SQLite.

   Devido esta configuração utilizar um servidor de banco de dados SQL, você pode remover o arquivo de banco de dados SQLite:

```
# rm -f /var/lib/nova/nova.sqlite
```

Verifique a operação

Esta seção descreve como verificar a operação do serviço de Computação.

Nota

Execute estes comandos no nodo controlador.

1. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

```
$ source admin-openrc.sh
```

2. Liste os componentes de serviço para verificar o lançamento com sucesso de cada processo:

```
$ nova service-list
+----+------------------+------------+----------+---------+-------
+----------------------------+-----------------+
| Id | Binary           | Host       | Zone     | Status  | State |
 Updated_at                 | Disabled Reason |
+----+------------------+------------+----------+---------+-------
+----------------------------+-----------------+
| 1  | nova-conductor   | controller | internal | enabled | up    |
 2014-09-16T23:54:02.000000 | -               |
| 2  | nova-consoleauth | controller | internal | enabled | up    |
 2014-09-16T23:54:04.000000 | -               |
| 3  | nova-scheduler   | controller | internal | enabled | up    |
 2014-09-16T23:54:07.000000 | -               |
| 4  | nova-cert        | controller | internal | enabled | up    |
 2014-09-16T23:54:00.000000 | -               |
| 5  | nova-compute     | compute1   | nova     | enabled | up    |
 2014-09-16T23:54:06.000000 | -               |
+----+------------------+------------+----------+---------+-------
+----------------------------+-----------------+
```

Nota

Esta saída deve indicar quatro componentes habilitados no nodo controlador e um componente habilitado no nodo de Computação.

3. Liste imagens no catálogo do Serviço de Imagem para verificar a conectividade com o serviço de Identidade e o serviço de Imagem:

```
$ nova image-list
+--------------------------------------+---------------------+--------
+--------+
| ID                                   | Name                | Status |
 Server |
+--------------------------------------+---------------------+--------
+--------+
| acafc7c0-40aa-4026-9673-b879898e1fc2 | cirros-0.3.3-x86_64 | ACTIVE |
     |
+--------------------------------------+---------------------+--------
+--------+
```

Capítulo 6. Adicione o componente de rede

Índice

This chapter explains how to install and configure either OpenStack Networking (neutron) or the legacy `nova-network` networking service. The `nova-network` service enables you to deploy one network type per instance and is suitable for basic network functionality. OpenStack Networking enables you to deploy multiple network types per instance and includes *plug-ins* for a variety of products that support *virtual networking*.

For more information, see the Networking chapter of the *OpenStack Cloud Administrator Guide*.

Rede OpenStack (neutron)

OpenStack Networking

OpenStack Networking allows you to create and attach interface devices managed by other OpenStack services to networks. Plug-ins can be implemented to accommodate different networking equipment and software, providing flexibility to OpenStack architecture and deployment.

It includes the following components:

`neutron-server`	Accepts and routes API requests to the appropriate OpenStack Networking plug-in for action.
OpenStack Networking plug-ins and agents	Plugs and unplugs ports, creates networks or subnets, and provides IP addressing. These plug-ins and agents differ depending on the vendor and technologies used in the particular cloud. OpenStack Networking ships with plug-ins and agents for Cisco virtual and physical switches, NEC OpenFlow products, Open vSwitch, Linux bridging, and the VMware NSX product. The common agents are L3 (layer 3), DHCP (dynamic host IP addressing), and a plug-in agent.
Messaging queue	Used by most OpenStack Networking installations to route information between the neutron-server and va-

rious agents, as well as a database to store networking state for particular plug-ins.

OpenStack Networking mainly interacts with OpenStack Compute to provide networks and connectivity for its instances.

Conceitos de rede

O OpenStack Networking (neutron) gerencia todas as facetas de rede para a Infraestrutura Virtual de Rede (VNI) e os aspectos da camada de acesso da Infraestrutura Física de Rede (PNI) no seu ambiente OpenStack. O OpenStack Networking permite os tenants criar topologias avançadas de rede virtual, incluindo serviços como *firewalls*, *load balancers*, e *virtual private networks (VPNs)*.

Networking fornece as abstrações de objetos de redes, sub-redes, e roteadores. Cada abstração possui funcionalidades que imitam o objeto físico equivalente: redes contém sub-redes, e roteadores roteiam o tráfego entre diferentes sub-redes e redes.

Cada roteador tem um gateway que conecta à rede, e várias interfaces conectadas a sub-redes. Sub-redes podem acessar máquinas em outras sub-redes conectadas ao mesmo roteador.

Qualquer configuração de rede dada tem ao menos uma rede externa. Diferente de outras redes, a rede externa não é meramente uma rede virtualmente definida. Em vez disso, ela representa uma visão dentro de uma fatia da rede física externa, acessível de fora da instalação do OpenStack. Os endereços IP na rede externa são acessíveis por qualquer um na rede de fora. Devido a rede externa meramente representar uma visão na rede de fora, o DHCP é desabilitado nessa rede.

Adicionalmente às redes externas, qualquer configuração de Rede tem uma ou mais redes internas. Essas redes definidas por software conectam diretamente às VMs. Somente as VMs em qualquer rede interna dada, ou aquelas em sub-redes conectadas através de interfaces para um roteador similar, podem acessar VMs conectadas à essa rede diretamente.

Para a rede externa acessar as VMs, e vice versa, são necessários roteadores entre as redes. Cada roteador tem um gateway conectado a uma rede e muitas interfaces que são conectadas a sub-redes. Como um roteador físico, as sub-redes podem acessar máquinas em outras sub-redes que estão conectadas ao mesmo roteador, e máquinas podem acessar a rede do lado de fora através do gateway para o roteador.

Adicionalmente, você pode alocar endereços IP nas redes externas para portas na rede interna. Quando algo está conectado a uma sub-rede, essa conexão é chamada uma porta. Você pode associar endereços IP de redes externas com portas para VMs. Desta forma, as entidades da rede do lado de fora podem acessar VMs.

Networking também suporta *grupos de segurança*. Grupos de segurança habilitam os administradores definir regras de firewall em grupos. Uma VM pode pertencer a um ou mais grupos de segurança, e o Networking aplica as regras nesses grupos de segurança para bloquear e desbloquear portas, intervalos de portas, ou tipos de tráfego para essa VM.

Cada plug-in que o Networking utiliza, tem seus próprios conceitos. Embora não seja vital para operar o VNI e o ambiente OpenStack, entender estes conceitos pode ajudá-lo a con-

figurar o Networking. Todas as instalações do Networking utilizam um plug-in central e um plug-in de grupo de segurança (ou simplesmente plug-in de grupo de segurança No-Op). Adicionalmente, os plug-ins de Firewall-as-a-Service (FWaaS) e Load-Balancer-as-a-Service (LBaaS) estão disponíveis.

Instalar e configurar o nodo controlador

Para configurar pre-requisitos

Antes de você configurar a Rede OpenStack (neutron), você deve criar uma base de dados e credenciais do Serviço de Identidade, incluindo os endpoints.

1. Para criar a base de dados, complete estes passos:

   a. Utilize o cliente de acesso a banco de dados para acessar o servidor de banco de dados como usuário `root`:

   ```
   $ mysql -u root -p
   ```

   b. Crie a base de dados `neutron`:

   ```
   CREATE DATABASE neutron;
   ```

   c. Conceda os acessos apropriados à base de dados `neutron`:

   ```
   GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost' \
     IDENTIFIED BY 'NEUTRON_DBPASS';
   GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%' \
     IDENTIFIED BY 'NEUTRON_DBPASS';
   ```

   Substitua *NEUTRON_DBPASS* com uma senha adequada.

   d. Saia do cliente de acesso a banco de dados.

2. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

   ```
   $ source admin-openrc.sh
   ```

3. Para criar as credenciais do Serviço de Identidade, complete estes passos:

   a. Crie o usuário `neutron`:

   ```
   $ keystone user-create --name neutron --pass NEUTRON_PASS
   +----------+----------------------------------+
   | Property |              Value               |
   +----------+----------------------------------+
   |  email   |                                  |
   | enabled  |               True               |
   |    id    | 7fd67878dcd04d0393469ef825a7e005 |
   |   name   |              neutron             |
   | username |              neutron             |
   +----------+----------------------------------+
   ```

   Substitua *NEUTRON_PASS* com uma senha adequada.

   b. Ligue o usuário `neutron` com o `serviço` de tenant e com o papel `admin`:

```
$ keystone user-role-add --user neutron --tenant service --role admin
```

Nota

Este comando não retorna resultados.

c. Crie o serviço `neutron`:

```
$ keystone service-create --name neutron --type network \
  --description "OpenStack Networking"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |       OpenStack Networking       |
|   enabled   |               True               |
|      id     | 6369ddaf99a447f3a0d41dac5e342161 |
|     name    |             neutron              |
|     type    |             network              |
+-------------+----------------------------------+
```

d. Crie os endpoints do Serviço de Identidade:

```
$ keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ network / {print $2}')
 \
  --publicurl http://controlador:9696 \
  --adminurl http://controlador:9696 \
  --internalurl http://controlador:9696 \
  --region regionOne
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
|   adminurl  |   http://controller:9696         |
|      id     | fa18b41938a94bf6b35e2c152063ee21 |
| internalurl |   http://controller:9696         |
|  publicurl  |   http://controller:9696         |
|    region   |            regionOne             |
|  service_id | 6369ddaf99a447f3a0d41dac5e342161 |
+-------------+----------------------------------+
```

Para instalar os componentes de rede

- `# apt-get install neutron-server neutron-plugin-ml2 python-neutronclient`

Para configurar o componente de servidor de Rede

A configuração dos componentes do servidor de Rede inclui base de dados, mecanismo de autenticação, intermediador de mensagens, notificações de alteração de topologia, e plug-in.

- Edite o arquivo `/etc/neutron/neutron.conf` e complete as seguintes ações:

  a. Na seção `[database]`, configure o acesso ao banco de dados:

```
[database]
...
connection = mysql://neutron:NEUTRON_DBPASS@controlador/neutron
```

Substitua *NEUTRON_DBPASS* com a senha que você escolheu para o banco de dados.

b. Na seção `[DEFAULT]`, configure o acesso ao intermediador RabbitMQ:

```
[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controlador
rabbit_password = RABBIT_PASS
```

Substitua *RABBIT_PASS* com a senha que você escolheu para a conta `guest` no RabbitMQ.

c. Nas seções `[DEFAULT]` e `[keystone_authtoken]`, configure o acesso do Serviço de Identidade:

```
[DEFAULT]
...
auth_strategy = keystone

[keystone_authtoken]
...
auth_uri = http://controlador:5000/v2.0
identity_uri = http://controlador:35357
admin_tenant_name = service
admin_user = neutron
admin_password = NEUTRON_PASS
```

Substitua *NEUTRON_PASS* com a senha que você escolheu para o usuário `neutron` no Serviço de Identidade.

Nota

Comente quaisquer opções `auth_host`, `auth_port`, e `auth_protocol`, porque a opção `identity_uri` as substitui.

d. Na seção `[DEFAULT]`, habilite o plug-in Modular Layer 2 (ML2), o serviço de roteamento, e os endereços IP que se sobrepõem:

```
[DEFAULT]
...
core_plugin = ml2
service_plugins = router
allow_overlapping_ips = True
```

e. Na seção `[DEFAULT]`, configure a Rede para notificar à Computação sobre alterações de topologia de rede:

```
[DEFAULT]
...
notify_nova_on_port_status_changes = True
notify_nova_on_port_data_changes = True
nova_url = http://controlador:8774/v2
nova_admin_auth_url = http://controlador:35357/v2.0
nova_region_name = regionOne
nova_admin_username = nova
nova_admin_tenant_id = SERVICE_TENANT_ID
nova_admin_password = NOVA_PASS
```

Substitua *SERVICE_TENANT_ID* com o identificador (id) do `serviço` de tenant no Serviço de Identidade e substitua *NOVA_PASS* com a senha que você escolheu para o usuário `nova` no serviço de Identidade.

Nota

Para obter o identificador (id) do `serviço` de tenant:

```
$ source admin-openrc.sh
$ keystone tenant-get service
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |          Service Tenant          |
|   enabled   |               True               |
|      id     | f727b5ec2ceb4d71bad86dfc414449bf |
|     name    |             service              |
+-------------+----------------------------------+
```

f. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção `[DEFAULT]` :

```
[DEFAULT]
...
verbose = True
```

Para configurar o plug-in Modular Layer 2 (ML2)

O plug-in ML2 utiliza o mecanismo (agente) *Open vSwitch (OVS)* para construir a estrutura virtual de rede para as instâncias. Contudo, o nodo controlador não precisa dos componentes OVS porque ele não lida com o tráfego de rede das instâncias.

- Edite o arquivo `/etc/neutron/plugins/ml2/ml2_conf.ini` e complete as seguintes ações:

  a. Na seção `[ml2]`, habilite os tipos de drivers *flat* e *generic routing encapsulation (GRE)*, as redes de tenant GRE, e o mecanismo de driver OVS:

```
[ml2]
...
type_drivers = flat,gre
tenant_network_types = gre
mechanism_drivers = openvswitch
```

Atenção

Um vez que você configura o plug-in ML2, esteja ciente de que desabilitando o driver de tipo de rede e habilitando-o novamente pode levar à inconsistências na base de dados.

b. Na seção `[ml2_type_gre]`, configure o identificador (id) da faixa do túnel:

```
[ml2_type_gre]
...
tunnel_id_ranges = 1:1000
```

c. Na seção `[securitygroup]`, habilite os grupos de segurança, habilite *ipset*, e configure o driver de firewall *iptables* do OVS:

```
[securitygroup]
...
enable_security_group = True
enable_ipset = True
firewall_driver = neutron.agent.linux.iptables_firewall.
OVSHybridIptablesFirewallDriver
```

Para configurar a Computação para utilizar a Rede

Por padrão, os pacotes de distribuição configuram a Computação para utilizar a rede legada. Você deve reconfigurar a Computação para gerenciar redes através do serviço de Rede.

- Edite o aqruivo `/etc/nova/nova.conf` e complete as seguintes ações:

a. Na seção `[DEFAULT]`, configure as *APIs* e os drivers:

```
[DEFAULT]
...
network_api_class = nova.network.neutronv2.api.API
security_group_api = neutron
linuxnet_interface_driver = nova.network.linux_net.
LinuxOVSInterfaceDriver
firewall_driver = nova.virt.firewall.NoopFirewallDriver
```

Nota

Por padrão, a Computação utiliza o serviço de firewall interno. Como o serviço de Rede inclui um serviço de firewall, você deve desabilitar o serviço de firewall da Computação utilizando o driver de firewall `nova.virt.firewall.NoopFirewallDriver`.

b. Na seção `[neutron]`, configure os parâmetros de acesso:

```
[neutron]
...
url = http://controlador:9696
auth_strategy = keystone
admin_auth_url = http://controlador:35357/v2.0
admin_tenant_name = service
admin_username = neutron
admin_password = NEUTRON_PASS
```

Substitua *NEUTRON_PASS* com a senha que você escolheu para o usuário `neutron` no Serviço de Identidade.

Para finalizar a instalação

1. Popule a base de dados:

```
# su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.
conf \
  --config-file /etc/neutron/plugins/ml2/ml2_conf.ini upgrade juno"
 neutron
```

Nota

O preenchimento da base de dados ocorre mais tarde para a Rede porque o script requer arquivos de configuração completos de servidor e de plug-in.

2. Reinicie o serviços de Computação:

```
# service nova-api restart
# service nova-scheduler restart
# service nova-conductor restart
```

3. Reinicie o serviço de Rede:

```
# service neutron-server restart
```

Verifique a operação

Nota

Execute estes comandos no nodo controlador.

1. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

```
$ source admin-openrc.sh
```

2. Liste as extensões carregadas para verificar a inicialização com sucesso, do processo do `neutron-server`:

```
$ neutron ext-list
+-----------------------+-----------------------------------------------+
| alias                 | name                                          |
+-----------------------+-----------------------------------------------+
| security-group        | security-group                                |
| l3_agent_scheduler    | L3 Agent Scheduler                            |
| ext-gw-mode           | Neutron L3 Configurable external gateway mode |
| binding               | Port Binding                                  |
| provider              | Provider Network                              |
| agent                 | agent                                         |
| quotas                | Quota management support                      |
| dhcp_agent_scheduler  | DHCP Agent Scheduler                          |
| l3-ha                 | HA Router extension                           |
| multi-provider        | Multi Provider Network                        |
```

```
| external-net          | Neutron external network                          |
| router                | Neutron L3 Router                                 |
| allowed-address-pairs | Allowed Address Pairs                             |
| extraroute            | Neutron Extra Route                               |
| extra_dhcp_opt        | Neutron Extra DHCP opts                           |
| dvr                   | Distributed Virtual Router                        |
+-----------------------+---------------------------------------------------+
```

Instalar e configurar o nodo de rede

O nodo de rede manipula primariamente o roteamento interno e externo e os serviços de *DHCP* para redes virtuais.

Para configurar pre-requisitos

Antes de você instalar e configurar a Rede OpenStack, você deve configurar certos parâmetros de rede do kernel.

1. Edite o arquivo `/etc/sysctl.conf` para conter os seguintes parâmetros:

```
net.ipv4.ip_forward=1
net.ipv4.conf.all.rp_filter=0
net.ipv4.conf.default.rp_filter=0
```

2. Implemente as alterações:

```
# sysctl -p
```

Para instalar os componentes de rede

- ```
  # apt-get install neutron-plugin-ml2 neutron-plugin-openvswitch-agent \
    neutron-l3-agent neutron-dhcp-agent
  ```

Para configurar os componentes comuns de Rede

A configuração dos componentes comuns de Rede incluem o mecanismo de autenticação, o intermediador de mensagens, e o plug-in.

- Edite o arquivo `/etc/neutron/neutron.conf` e complete as seguintes ações:

  a. Na seção `[database]`, comente quaisquer opções `connection` porque os nodos de rede não acessam a base de dados diretamente.

  b. Na seção `[DEFAULT]`, configure o acesso ao intermediador RabbitMQ:

  ```
  [DEFAULT]
  ...
  rpc_backend = rabbit
  rabbit_host = controlador
  rabbit_password = RABBIT_PASS
  ```

  Substitua *RABBIT_PASS* com a senha que você escolheu para a conta `guest` no RabbitMQ.

  c. Nas seções `[DEFAULT]` e `[keystone_authtoken]`, configure o acesso do Serviço de Identidade:
  ```

```
[DEFAULT]
...
auth_strategy = keystone

[keystone_authtoken]
...
auth_uri = http://controlador:5000/v2.0
identity_uri = http://controlador:35357
admin_tenant_name = service
admin_user = neutron
admin_password = NEUTRON_PASS
```

Substitua *NEUTRON_PASS* com a senha que você escolheu para o usuário `neutron` no Serviço de Identidade.

Nota

Comente quaisquer opções `auth_host`, `auth_port`, e `auth_protocol`, porque a opção `identity_uri` as substitui.

d. Na seção `[DEFAULT]`, habilite o plug-in Modular Layer 2 (ML2), o serviço de roteamento, e os endereços IP que se sobrepõem:

```
[DEFAULT]
...
core_plugin = ml2
service_plugins = router
allow_overlapping_ips = True
```

e. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção `[DEFAULT]` :

```
[DEFAULT]
...
verbose = True
```

Para configurar o plug-in Modular Layer 2 (ML2)

O plug-in ML2 utiliza o mecanismo (agente) *Open vSwitch (OVS)* para construir a estrutura de rede virtual para as instâncias.

- Edite o arquivo `/etc/neutron/plugins/ml2/ml2_conf.ini` e complete as seguintes ações:

  a. Na seção `[ml2]`, habilite os tipos de drivers *flat* e *generic routing encapsulation (GRE)*, as redes de tenant GRE, e o mecanismo de driver OVS:

```
[ml2]
...
type_drivers = flat,gre
tenant_network_types = gre
mechanism_drivers = openvswitch
```

  b. Na seção `[ml2_type_flat]`, configure o provedor externo de rede fixa:

```
[ml2_type_flat]
...
flat_networks = external
```

c. Na seção `[ml2_type_gre]`, configure o identificador (id) da faixa do túnel:

```
[ml2_type_gre]
...
tunnel_id_ranges = 1:1000
```

d. Na seção `[securitygroup]`, habilite os grupos de segurança, habilite *ipset*, e configure o driver de firewall *iptables* do OVS:

```
[securitygroup]
...
enable_security_group = True
enable_ipset = True
firewall_driver = neutron.agent.linux.iptables_firewall.
OVSHybridIptablesFirewallDriver
```

e. Na seção `[ovs]`, habilite os túneis, configure o endpoint do túnel local, e mapeie o provedor externo de rede fixa para a bridge de rede externa `br-ex`:

```
[ovs]
...
local_ip = INSTANCE_TUNNELS_INTERFACE_IP_ADDRESS
enable_tunneling = True
bridge_mappings = external:br-ex
```

Substitua *`INSTANCE_TUNNELS_INTERFACE_IP_ADDRESS`* com o endereço IP da interface de rede da instância de túneis em seu nodo de rede.

f. Na seção `[agent]`, habilite os túneis GRE:

```
[agent]
...
tunnel_types = gre
```

Para configurar o agente Camada-3 (L3)

O *agente Layer-3 (L3)* fornece serviços de roteamento para redes virtuais.

- Edite o arquivo `/etc/neutron/l3_agent.ini` e complete as seguintes ações:

a. Na seção `[DEFAULT]`, configure o driver, habilite *network namespaces*, e configure a bridge de rede externa:

```
[DEFAULT]
...
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
use_namespaces = True
external_network_bridge = br-ex
```

b. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção `[DEFAULT]` :

```
[DEFAULT]
...
verbose = True
```

Para configurar o agente DHCP

O *agente DHCP* fornece serviços DHCP para redes virtuais.

1. Edite o arquivo `/etc/neutron/dhcp_agent.ini` e complete as seguintes ações:

   a. Na seção `[DEFAULT]`, configure os drivers e habilite namespaces:

   ```
   [DEFAULT]
   ...
   interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
   dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
   use_namespaces = True
   ```

   b. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção `[DEFAULT]` :

   ```
   [DEFAULT]
   ...
   verbose = True
   ```

2. (Opcional)

   Tunneling protocols such as GRE include additional packet headers that increase overhead and decrease space available for the payload or user data. Without knowledge of the virtual network infrastructure, instances attempt to send packets using the default Ethernet *maximum transmission unit (MTU)* of 1500 bytes. *Internet protocol (IP)* networks contain the *path MTU discovery (PMTUD)* mechanism to detect end-to-end MTU and adjust packet size accordingly. However, some operating systems and networks block or otherwise lack support for PMTUD causing performance degradation or connectivity failure.

   Ideally, you can prevent these problems by enabling *jumbo frames* on the physical network that contains your tenant virtual networks. Jumbo frames support MTUs up to approximately 9000 bytes which negates the impact of GRE overhead on virtual networks. However, many network devices lack support for jumbo frames and OpenStack administrators often lack control over network infrastructure. Given the latter complications, you can also prevent MTU problems by reducing the instance MTU to account for GRE overhead. Determining the proper MTU value often takes experimentation, but 1454 bytes works in most environments. You can configure the DHCP server that assigns IP addresses to your instances to also adjust the MTU.

   Nota

   Algumas imagens de nuvem ignoram a opção de DHCP MTU, nesse caso você deve configurá-la utilizando metadados, script, ou outro método adequado.

   a. Edite o arquivo `/etc/neutron/dhcp_agent.ini` e complete as seguintes ações:

- Na seção [DEFAULT], habilite o arquivo de configuração *dnsmasq*:

```
[DEFAULT]
...
dnsmasq_config_file = /etc/neutron/dnsmasq-neutron.conf
```

b. Crie e edite o arquivo /etc/neutron/dnsmasq-neutron.conf e complete as seguintes ações:

- Habilite a opção (26) DHCP MTU e configure-a para 1454 bytes:

```
dhcp-option-force=26,1454
```

c. Mate qualquer processo dnsmasq existente:

```
# pkill dnsmasq
```

Para configurar o agente de metadado

O *agente de metadados* fornece informações de configuração tais como credenciais para as instâncias.

1. Edite o arquivo /etc/neutron/metadata_agent.ini e complete as seguintes ações:

a. Na seção [DEFAULT], configure os parâmetros de acesso:

```
[DEFAULT]
...
auth_url = http://controlador:5000/v2.0
auth_region = regionOne
admin_tenant_name = service
admin_user = neutron
admin_password = NEUTRON_PASS
```

Substitua *NEUTRON_PASS* com a senha que você escolheu para o usuário neutron no Serviço de Identidade.

b. Na seção [DEFAULT], configure o host de metadados:

```
[DEFAULT]
...
nova_metadata_ip = controlador
```

c. Na seção [DEFAULT], configure o shared secret do proxy de metadados:

```
[DEFAULT]
...
metadata_proxy_shared_secret = METADATA_SECRET
```

Substitua *METADATA_SECRET* com um secret adequado para o proxy de metadados.

d. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção [DEFAULT] :

```
[DEFAULT]
...
verbose = True
```

2. No nodo *controlador*, edite o arquivo `/etc/nova/nova.conf` e complete as seguintes ações:

   - Na seção `[neutron]`, habilite o proxy de metadados e configure o secret:

     ```
     [neutron]
     ...
     service_metadata_proxy = True
     metadata_proxy_shared_secret = METADATA_SECRET
     ```

     Substitua *METADATA_SECRET* com a senha que você escolheu para o proxy de metadado.

3. No nodo *controlador*, reinicie o serviço de *API* de Computação

   ```
   # service nova-api restart
   ```

Para configurar o serviço Open vSwitch (OVS)

The OVS service provides the underlying virtual networking framework for instances. The integration bridge `br-int` handles internal instance network traffic within OVS. The external bridge `br-ex` handles external instance network traffic within OVS. The external bridge requires a port on the physical external network interface to provide instances with external network access. In essence, this port connects the virtual and physical external networks in your environment.

1. Reinicie o serviço OVS:

   ```
   # service openvswitch-switch restart
   ```

2. Adicione a bridge externa:

   ```
   # ovs-vsctl add-br br-ex
   ```

3. Adicione uma porta à bridge externa que conecta à interface de rede física externa:

   Substitua *INTERFACE_NAME* com o nome atual da interface. Por exempo, *eth2* ou *ens256*.

   ```
   # ovs-vsctl add-port br-ex INTERFACE_NAME
   ```

Nota

Dependendo do seu driver de interface de rede, você poderá precisar desabilitar o *generic receive offload (GRO)* para alcançar a taxa de transferência desejada entre suas instâncias e a rede externa.

Para desabilitar o GRO na interface de rede externa enquanto testa seu ambiente:

```
# ethtool -K INTERFACE_NAME gro off
```

Para finalizar a instalação

- Reinicie os serviços de Rede:

```
# service neutron-plugin-openvswitch-agent restart
# service neutron-l3-agent restart
# service neutron-dhcp-agent restart
# service neutron-metadata-agent restart
```

Verifique a operação

Nota

Execute estes comandos no nodo controlador.

1. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

```
$ source admin-openrc.sh
```

2. Liste os agentes para verificar a inicialização com sucesso dos agentes do neutron:

```
$ neutron agent-list
+--------------------------------------+--------------------+---------
+-------+----------------+---------------------------+
| id                                   | agent_type         | host    |
 alive | admin_state_up | binary                    |
+--------------------------------------+--------------------+---------
+-------+----------------+---------------------------+
| 30275801-e17a-41e4-8f53-9db63544f689 | Metadata agent     | network
 | :-)   | True           | neutron-metadata-agent    |
| 4bd8c50e-7bad-4f3b-955d-67658a491a15 | Open vSwitch agent | network
 | :-)   | True           | neutron-openvswitch-agent |
| 756e5bba-b70f-4715-b80e-e37f59803d20 | L3 agent           | network
 | :-)   | True           | neutron-l3-agent          |
| 9c45473c-6d6d-4f94-8df1-ebd0b6838d5f | DHCP agent         | network
 | :-)   | True           | neutron-dhcp-agent        |
+--------------------------------------+--------------------+---------
+-------+----------------+---------------------------+
```

Instalar e configurar o nodo de Computação

O nodo de Computação trata da conectividade e dos *grupos de segurança* para as instâncias.

Para configurar pre-requisitos

Antes de você instalar e configurar a Rede OpenStack, você deve configurar certos parâmetros de rede do kernel.

1. Edite o arquivo `/etc/sysctl.conf` para conter os seguintes parâmetros:

```
net.ipv4.conf.all.rp_filter=0
net.ipv4.conf.default.rp_filter=0
```

2. Implemente as alterações:

```
# sysctl -p
```

Para instalar os componentes de rede

- `# apt-get install neutron-plugin-ml2 neutron-plugin-openvswitch-agent`

Para configurar os componentes comuns de Rede

A configuração dos componentes comuns de Rede incluem o mecanismo de autenticação, o intermediador de mensagens, e o plug-in.

- Edite o arquivo `/etc/neutron/neutron.conf` e complete as seguintes ações:

  a. Na seção `[database]`, comente quaisquer opções `connection` porque os nodos de Computação não acessam a base de dados diretamente.

  b. Na seção `[DEFAULT]`, configure o acesso ao intermediador RabbitMQ:

  ```
  [DEFAULT]
  ...
  rpc_backend = rabbit
  rabbit_host = controlador
  rabbit_password = RABBIT_PASS
  ```

  Substitua *RABBIT_PASS* com a senha que você escolheu para a conta `guest` no RabbitMQ.

  c. Nas seções `[DEFAULT]` e `[keystone_authtoken]`, configure o acesso do Serviço de Identidade:

  ```
  [DEFAULT]
  ...
  auth_strategy = keystone

  [keystone_authtoken]
  ...
  auth_uri = http://controlador:5000/v2.0
  identity_uri = http://controlador:35357
  admin_tenant_name = service
  admin_user = neutron
  admin_password = NEUTRON_PASS
  ```

  Substitua *NEUTRON_PASS* com a senha que você escolheu para o usuário `neutron` no Serviço de Identidade.

  Nota

  Comente quaisquer opções `auth_host`, `auth_port`, e `auth_protocol`, porque a opção `identity_uri` as substitui.

  d. Na seção `[DEFAULT]`, habilite o plug-in Modular Layer 2 (ML2), o serviço de roteamento, e os endereços IP que se sobrepõem:

  ```
  [DEFAULT]
  ...
  core_plugin = ml2
  service_plugins = router
  allow_overlapping_ips = True
  ```

e. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção `[DEFAULT]` :

```
[DEFAULT]
...
verbose = True
```

Para configurar o plug-in Modular Layer 2 (ML2)

O plug-in ML2 utiliza o mecanismo (agente) do Open vSwitch (OVS) para construir a estrutura de rede virtual para instâncias.

- Edite o arquivo `/etc/neutron/plugins/ml2/ml2_conf.ini` e complete as seguintes ações:

  a. Na seção `[ml2]`, habilite os tipos de drivers *flat* e *generic routing encapsulation (GRE)*, as redes de tenant GRE, e o mecanismo de driver OVS:

  ```
  [ml2]
  ...
  type_drivers = flat,gre
  tenant_network_types = gre
  mechanism_drivers = openvswitch
  ```

  b. Na seção `[ml2_type_gre]`, configure o identificador (id) da faixa do túnel:

  ```
  [ml2_type_gre]
  ...
  tunnel_id_ranges = 1:1000
  ```

  c. Na seção `[securitygroup]`, habilite os grupos de segurança, habilite *ipset*, e configure o driver de firewall *iptables* do OVS:

  ```
  [securitygroup]
  ...
  enable_security_group = True
  enable_ipset = True
  firewall_driver = neutron.agent.linux.iptables_firewall.
  OVSHybridIptablesFirewallDriver
  ```

  d. Na seção `[ovs]`, habilite os túneis e configure o endpoint do túnel local:

  ```
  [ovs]
  ...
  local_ip = INSTANCE_TUNNELS_INTERFACE_IP_ADDRESS
  enable_tunneling = True
  ```

  Substitua *`INSTANCE_TUNNELS_INTERFACE_IP_ADDRESS`* com o endereço IP da interface de rede dos túneis de instâncias no seu nodo de Computação.

  e. Na seção `[agent]`, habilite os túneis GRE:

  ```
  [agent]
  ...
  tunnel_types = gre
  ```

Para configurar o serviço Open vSwitch (OVS)

O serviço OVS fornece a estrutura de rede virtual subjacente para as instâncias.

- Reinicie o serviço OVS:

```
# service openvswitch-switch restart
```

Para configurar a Computação para utilizar a Rede

Por padrão, os pacotes de distribuição configuram a Computação para utilizar a rede legada. Você deve reconfigurar a Computação para gerenciar redes através do serviço de Rede.

- Edite o aqruivo `/etc/nova/nova.conf` e complete as seguintes ações:

  a. Na seção `[DEFAULT]`, configure as *APIs* e os drivers:

```
[DEFAULT]
...
network_api_class = nova.network.neutronv2.api.API
security_group_api = neutron
linuxnet_interface_driver = nova.network.linux_net.
LinuxOVSInterfaceDriver
firewall_driver = nova.virt.firewall.NoopFirewallDriver
```

Nota

Por padrão, a Computação utiliza o serviço de firewall interno. Como o serviço de Rede inclui um serviço de firewall, você deve desabilitar o serviço de firewall da Computação utilizando o driver de firewall `nova.virt.firewall.NoopFirewallDriver`.

  b. Na seção `[neutron]`, configure os parâmetros de acesso:

```
[neutron]
...
url = http://controlador:9696
auth_strategy = keystone
admin_auth_url = http://controlador:35357/v2.0
admin_tenant_name = service
admin_username = neutron
admin_password = NEUTRON_PASS
```

  Substitua *NEUTRON_PASS* com a senha que você escolheu para o usuário `neutron` no Serviço de Identidade.

Para finalizar a instalação

1. Reinicie o serviço de Computação:

```
# service nova-compute restart
```

2. Reinicie o agente Open vSwitch (OVS):

```
# service neutron-plugin-openvswitch-agent restart
```

Verifique a operação

Nota

Execute estes comandos no nodo controlador.

1. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

```
$ source admin-openrc.sh
```

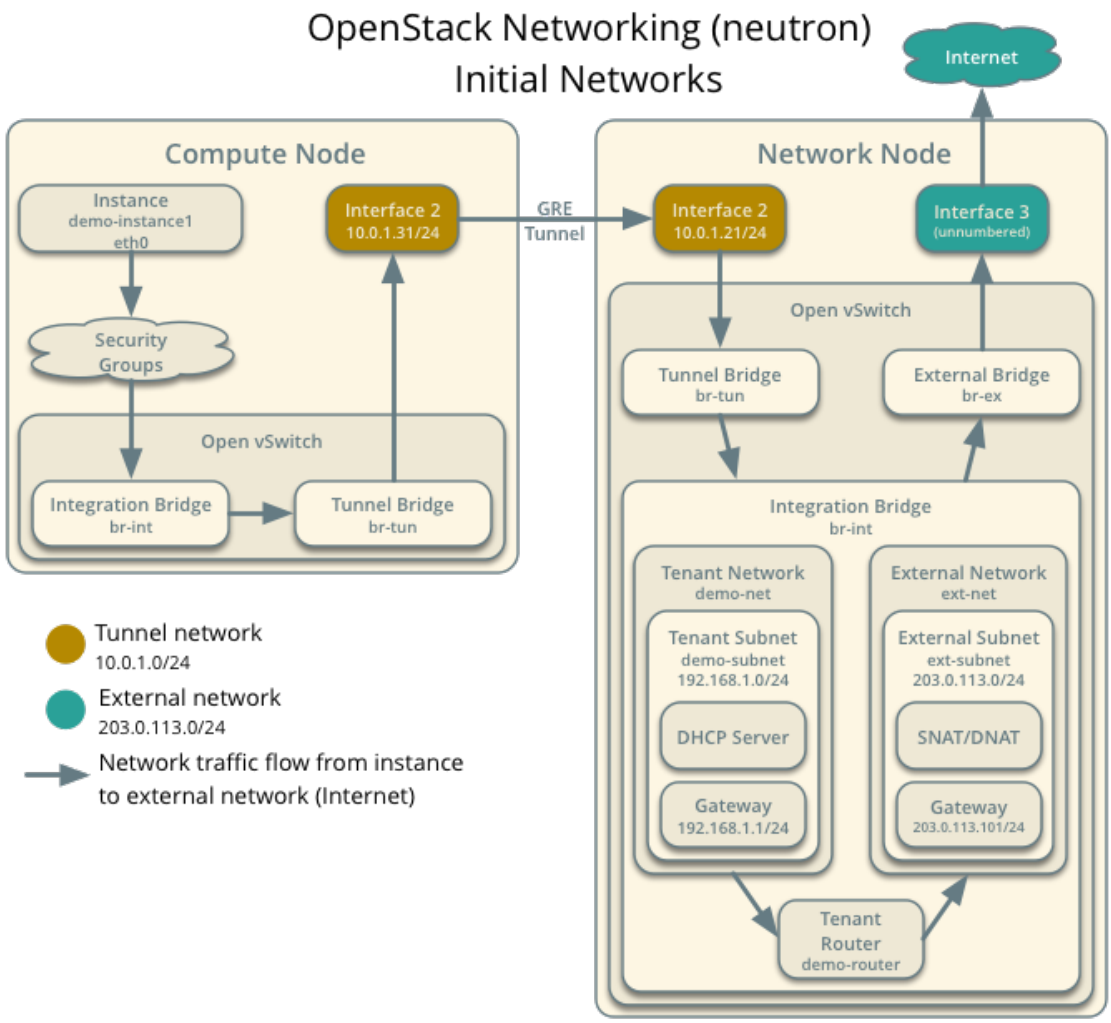
2. Liste os agentes para verificar a inicialização com sucesso dos agentes do neutron:

```
$ neutron agent-list
+--------------------------------------+--------------------+---------
+-------+----------------+---------------------------+
| id                                   | agent_type         | host    |
 alive | admin_state_up | binary                    |
+--------------------------------------+--------------------+---------
+-------+----------------+---------------------------+
...
| a5a49051-05eb-4b4f-bfc7-d36235fe9131 | Open vSwitch agent | compute1
 | :-)   | True           | neutron-openvswitch-agent |
+--------------------------------------+--------------------+---------
+-------+----------------+---------------------------+
```

Crie as redes iniciais

Antes de lançar sua primeira instância, você deve criar a estrutura de rede virtual necessária na qual a instância irá se conectar, incluindo a rede externa e a rede de tenant. Consulte Figura 6.1, “Redes iniciais” [79]. Após criar esta infraestrutura, recomendamos que você verifique a conectividade e resolva quaisquer problemas antes de prosseguir. Figura 6.1, “Redes iniciais” [79] fornece uma visão geral da arquitetura básica dos componentes que a Rede implementa para as redes iniciais e mostra como o tráfego de rede flui da instância para a rede externa ou a Internet.

Figura 6.1. Redes iniciais



Rede externa

The external network typically provides Internet access for your instances. By default, this network only allows Internet access *from* instances using *Network Address Translation (NAT)*. You can enable Internet access *to* individual instances using a *floating IP address* and suitable *security group* rules. The `admin` tenant owns this network because it provides external network access for multiple tenants.

Nota

Execute estes comandos no nodo controlador.

Para criar a rede externa

1. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

```
$ source admin-openrc.sh
```

2. Crie a rede

```
$ neutron net-create ext-net --router:external True \
  --provider:physical_network external --provider:network_type flat
Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 893aebb9-1c1e-48be-8908-6b947f3237b3 |
| name                      | ext-net                              |
| provider:network_type     | flat                                 |
| provider:physical_network | external                             |
| provider:segmentation_id  |                                      |
| router:external           | True                                 |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | 54cd044c64d5408b83f843d63624e0d8     |
+---------------------------+--------------------------------------+
```

Assim como uma rede física, uma rede virtual requer uma *subrede* atribuída a ela. A rede externa compartilha a mesma subrede e *gateway* associados com a rede física conectada à interface externa no nodo de rede. Você deve especificar uma faixa exclusiva desta subrede para o *roteador* e endereços IP flutuantes para evitar interferência com outros dispositivos na rede externa.

Para criar uma subrede na rede externa

- Crie a subrede:

```
$ neutron subnet-create ext-net --name ext-subnet \
  --allocation-pool start=FLOATING_IP_START,end=FLOATING_IP_END \
  --disable-dhcp --gateway EXTERNAL_NETWORK_GATEWAY EXTERNAL_NETWORK_CIDR
```

Substitua*FLOATING_IP_START* e *FLOATING_IP_END* com o primeiro e o último endereço IP da faixa que você quer alocar para os endereços IP flutuantes. Substitua *EXTERNAL_NETWORK_CIDR* com a subrede associada com a rede física. Substitua *EXTERNAL_NETWORK_GATEWAY* com o gateway associado com a rede física, tipicamente o endereço IP ".1" . Você deve desabilitar o *DHCP* nesta subrede porque as instâncias não se conectam diretamente à rede externa e os IPs flutuantes requerem atribuição manual.

Por exemplo, utilizando `203.0.113.0/24` com a faixa de IP flutuante de `203.0.113.101` até `203.0.113.200`:

```
$ neutron subnet-create ext-net --name ext-subnet \
  --allocation-pool start=203.0.113.101,end=203.0.113.200 \
  --disable-dhcp --gateway 203.0.113.1 203.0.113.0/24
Created a new subnet:
+------------------
+-----------------------------------------------------+
| Field             | Value
 |
```

```
+-------------------
+-------------------------------------------------------+
| allocation_pools  | {"start": "203.0.113.101", "end": "203.0.113.200"}
 |
| cidr              | 203.0.113.0/24
 |
| dns_nameservers   |
 |
| enable_dhcp       | False
 |
| gateway_ip        | 203.0.113.1
 |
| host_routes       |
 |
| id                | 9159f0dc-2b63-41cf-bd7a-289309da1391
 |
| ip_version        | 4
 |
| ipv6_address_mode |
 |
| ipv6_ra_mode      |
 |
| name              | ext-subnet
 |
| network_id        | 893aebb9-1c1e-48be-8908-6b947f3237b3
 |
| tenant_id         | 54cd044c64d5408b83f843d63624e0d8
 |
+-------------------
+-------------------------------------------------------+
```

Rede de tenant

A rede de tenant fornece acesso de rede interna para instâncias. A arquitetura isola esse tipo de rede dos outros tenants. O tenant demo detém essa rede porque ele fornece somente acesso de rede para instâncias dentro dele.

Nota

Execute estes comandos no nodo controlador.

Para criar a rede de tenant

1. Obtenha as credenciais demo para ter acesso aos comandos user-only CLI:

```
$ source demo-openrc.sh
```

2. Crie a rede

```
$ neutron net-create demo-net
Created a new network:
+-----------------+--------------------------------------+
| Field           | Value                                |
+-----------------+--------------------------------------+
| admin_state_up  | True                                 |
| id              | ac108952-6096-4243-adf4-bb6615b3de28 |
| name            | demo-net                             |
| router:external | False                                |
```

```
| shared          | False                                |
| status          | ACTIVE                               |
| subnets         |                                      |
| tenant_id       | cdef0071a0194d19ac6bb63802dc9bae     |
+-----------------+--------------------------------------+
```

Assim como a rede externa, sua rede de tenant também requer uma subrede anexada a ela. Você pode especificar qualquer subrede válida porque a arquitetura isola as redes de tenants. Por padrão, esta subrede irá utilizar DHCP para que suas instâncias possam obter endereços IP.

Para criar uma subrede na rede de tenant

- Crie a subrede:

```
$ neutron subnet-create demo-net --name demo-subnet \
  --gateway TENANT_NETWORK_GATEWAY TENANT_NETWORK_CIDR
```

Substitua *TENANT_NETWORK_CIDR* com a subrede que você quer associar com a rede de tenant e *TENANT_NETWORK_GATEWAY* com o gateway que você quer associar com ela, tipicamente o endereço IP ".1".

Exemplo utilizando `192.168.1.0/24`:

```
$ neutron subnet-create demo-net --name demo-subnet \
  --gateway 192.168.1.1 192.168.1.0/24
Created a new subnet:
+-------------------
+-----------------------------------------------------+
| Field             | Value
 |
+-------------------
+-----------------------------------------------------+
| allocation_pools  | {"start": "192.168.1.2", "end": "192.168.1.254"}
 |
| cidr              | 192.168.1.0/24
 |
| dns_nameservers   |
 |
| enable_dhcp       | True
 |
| gateway_ip        | 192.168.1.1
 |
| host_routes       |
 |
| id                | 69d38773-794a-4e49-b887-6de6734e792d
 |
| ip_version        | 4
 |
| ipv6_address_mode |
 |
| ipv6_ra_mode      |
 |
| name              | demo-subnet
 |
| network_id        | ac108952-6096-4243-adf4-bb6615b3de28
 |
| tenant_id         | cdef0071a0194d19ac6bb63802dc9bae
 |
```

```
+------------------
+-----------------------------------------------------+
```

Um roteador virtual passa o tráfego de rede entre duas ou mais redes virtuais. Cada roteador requer uma ou mais *interfaces* e/ou gateway que fornece acesso a redes específicas. Neste caso, você irá criar um roteador e conectar seu tenant e sua rede externa a ele.

Para criar um roteador na rede de tenant e conectar as redes externa e de tenant a ele

1. Criar o roteador:

```
$ neutron router-create demo-router
Created a new router:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 635660ae-a254-4feb-8993-295aa9ec6418 |
| name                  | demo-router                          |
| routes                |                                      |
| status                | ACTIVE                               |
| tenant_id             | cdef0071a0194d19ac6bb63802dc9bae     |
+-----------------------+--------------------------------------+
```

2. Conecte o roteador à subrede de tenant `demo`:

```
$ neutron router-interface-add demo-router demo-subnet
Added interface b1a894fd-aee8-475c-9262-4342afdc1b58 to router demo-
router.
```

3. Conecte o roteador à rede externa configurando-o como o gateway:

```
$ neutron router-gateway-set demo-router ext-net
Set gateway for router demo-router
```

Verifique a conectividade

Recomendamos que você verifique a conectividade de rede e resolva quaisquer problemas antes de prosseguir. Seguindo o exemplo da subrede externa utilizando `203.0.113.0/24`, o roteador de tenant deve ocupar o endereço IP mais baixo na faixa de endereços IP flutuantes, `203.0.113.101`. Se você configurou sua rede física externa e as redes virtuais, você deve ser capaz de **ping** este endereço IP de qualquer host em sua rede física externa.

Nota

Se você está construindo seus nodos OpenStack como máquinas virtuais, você deve configurar o hypervisor para permitir o modo promíscuo na rede externa.

Para verificar a conectividade de rede

- Ping o roteador gateway do tenant:

```
$ ping -c 4 203.0.113.101
PING 203.0.113.101 (203.0.113.101) 56(84) bytes of data.
```

```
64 bytes from 203.0.113.101: icmp_req=1 ttl=64 time=0.619 ms
64 bytes from 203.0.113.101: icmp_req=2 ttl=64 time=0.189 ms
64 bytes from 203.0.113.101: icmp_req=3 ttl=64 time=0.165 ms
64 bytes from 203.0.113.101: icmp_req=4 ttl=64 time=0.216 ms

--- 203.0.113.101 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
rtt min/avg/max/mdev = 0.165/0.297/0.619/0.187 ms
```

Rede legada (nova-network)

Configure o nodo controlador

Legacy networking primarily involves compute nodes. However, you must configure the controller node to use legacy networking.

Para configurar a rede legada

1. Edite o aqruivo `/etc/nova/nova.conf` e complete as seguintes ações:

   - Na seção `[DEFAULT]`, configure as APIs de rede e de grupo de segurança:

     ```
     [DEFAULT]
     ...
     network_api_class = nova.network.api.API
     security_group_api = nova
     ```

2. Reinicie o serviços de Computação:

   ```
   # service nova-api restart
   # service nova-scheduler restart
   # service nova-conductor restart
   ```

Configure o nodo de Computação

Esta seção cobre a implantação de uma simples *rede fixa* que fornece endereços IP para suas instâncias via *DHCP*. Se seu ambiente inclui múltiplos nodos de Computação, a característica de *multi-host* fornece redundância distribuindo funções de rede pelos nodos de Computação.

Para instalar componentes de rede legados

- ```
  # apt-get install nova-network nova-api-metadata
  ```

Para configurar a rede legada

1. Edite o aqruivo `/etc/nova/nova.conf` e complete as seguintes ações:

   - Na seção `[DEFAULT]`, configure os parâmetros de rede:

     ```
     [DEFAULT]
     ...
     network_api_class = nova.network.api.API
     security_group_api = nova
     firewall_driver = nova.virt.libvirt.firewall.IptablesFirewallDriver
     network_manager = nova.network.manager.FlatDHCPManager
     ```
     ```

```
network_size = 254
allow_same_net_traffic = False
multi_host = True
send_arp_for_ha = True
share_dhcp_address = True
force_dhcp_release = True
flat_network_bridge = br100
flat_interface = INTERFACE_NAME
public_interface = INTERFACE_NAME
```

Substitua *INTERFACE_NAME* pelo nome real da interface para a rede externa. Por exemplo, *eth1* ou *ens224*.

2. Reinicie os serviços:

```
# service nova-network restart
# service nova-api-metadata restart
```

Crie a rede inicial

Antes de lançar sua primeira instância, você deve criar a infraestrutura virtual de rede necessária, à qual a instância irá se conectar. Esta rede tipicamente fornece acesso Internet *a partir das*instâncias. Você pode habilitar o acesso Internet *para* instâncias individuais usando um *endereço IP flutuante* e regras adequadas de *grupo de segurança*. O tenant `admin` detém esta rede porque ele fornece acesso de rede externo para múltiplos tenants.

Esta rede compartilha a mesma *sub-rede* associada com a rede física conectada à *interface* no nodo de computação. Você deve especificar uma fatia exclusiva dessa sub-rede para prevenir interferência com outros dispositivos na rede externa.

Nota

Execute estes comandos no nodo controlador.

Para criar a rede

1. Adquira as credenciais do tenant `admin`:

```
$ source admin-openrc.sh
```

2. Crie a rede

   Substitua *`NETWORK_CIDRcom a sub-rede associada com a rede f`*

```
$ nova network-create demo-net --bridge br100 --multi-host T \
  --fixed-range-v4 NETWORK_CIDR
```

   Por exemplo, utilizando uma fatia exclusiva `203.0.113.0/24` com intervalo de endereços IP de `203.0.113.24` a `203.0.113.32`:

```
$ nova network-create demo-net --bridge br100 --multi-host T \
  --fixed-range-v4 203.0.113.24/29
```

Nota

Este comando não retorna resultados.

3. Verifique a criação da rede:

```
$ nova net-list
+--------------------------------------+----------+------------------+
| ID                                   | Label    | CIDR             |
+--------------------------------------+----------+------------------+
| 84b34a65-a762-44d6-8b5e-3b461a53f513 | demo-net | 203.0.113.24/29  |
+--------------------------------------+----------+------------------+
```

Próximos passos

Your OpenStack environment now includes the core components necessary to launch a basic instance. You can launch an instance or add more OpenStack services to your environment.

Capítulo 7. Adicione o dashboard

Índice

The OpenStack dashboard, also known as Horizon, is a Web interface that enables cloud administrators and users to manage various OpenStack resources and services.

O dashboard permite interações na web com o controlador de nuvem OpenStack Compute através das APIs do OpenStack.

O Horizon permite que você personalize a marca do painel.

O Horizon provê um conjunto de classes básicas, modelos reutilizáveis e ferramentas.

Este exemplo de implantação utiliza um servidor web Apache.

Requisitos de sistema

Antes de instalar o OpenStack dashboard, você deve satisfazer os seguintes requisitos de sistema:

- Instalação do OpenStack Compute. Habilite o serviço de Identidade para gerenciamento de usuário e projeto.

  Note as URLs do Serviço de Identidade e endpoints de Computação.

- Usuário do serviço de identidade com privilégios sudo. Como o Apache não serve o conteúdo via usuário root, os usuários devem executar o dashboard como um usuário do serviço de identidade com privilégios sudo.

- Python 2.7. A versão do Python deve suportar Django. A versão do Python deve rodar em qualquer sistema, incluindo Mac OS X. Os requisitos de instalação podem variar por plataforma.

Então, instale e configure a dashboard em um nodo que pode contactar o Identity Service.

Fornece aos usuários as seguintes informações para que eles possam acessar o dashboard por meio de um browser em sua máquina local:

- O endereço IP público a partir do qual eles podem acessar o dashboard.

- O usuário e a senha com os quais eles podem acessar o dashboard

O seu navegador e os de seus usuarios devem suportar HTML5 e ter cookies e o JavaScript habilitados

Nota

Para usar o cliente VNC com o dashboard, o browser deve suportar HTML5 Canvas e HTML5 WebSockets.

For details about browsers that support noVNC, see https://github.com/kanaka/noVNC/blob/master/README.md, and https://github.com/kanaka/noVNC/wiki/Browser-support, respectively.

Instalar e configurar

Esta seção descreve como instalar e configurar o dashboard no nodo controlador.

Before you proceed, verify that your system meets the requirements in "Requisitos de sistema" [87]. Also, the dashboard relies on functional core services including Identity, Image Service, Compute, and either Networking (neutron) or legacy networking (nova-network). Environments with stand-alone services such as Object Storage cannot use the dashboard. For more information, see the developer documentation.

Para instalar os componentes do dashboard

- Instale os pacotes:

```
# apt-get install openstack-dashboard apache2 libapache2-mod-wsgi
 memcached python-memcache
```

Nota

O Ubuntu instala o pacote openstack-dashboard-ubuntu-theme como uma dependência. Alguns usuários reportaram problemas com esse tema em versões anteriores. Se você encontrar problemas, remova este pacote para restaurar o tema original do OpenStack.

Para configurar o dashboard

- Edite o arquivo `/etc/openstack-dashboard/local_settings.py` e complete as seguintes ações:

  a. Configure o dashboard para utilizar os serviços OpenStack no nodo `controlador`:

  ```
  OPENSTACK_HOST = "controlador"
  ```

  b. Permita a todos os hosts acessarem o dashboard:

  ```
  ALLOWED_HOSTS = ['*']
  ```

  c. Configure a sessão de serviço de armazenamento do memcached:

```
CACHES = {
   'default': {
       'BACKEND': 'django.core.cache.backends.memcached.
MemcachedCache',
       'LOCATION': '127.0.0.1:11211',
   }
}
```

Nota

Comente qualquer outra sessão de configuração de armazenamento.

d. Opcionalmente, configure o fuso horário:

```
TIME_ZONE = "TIME_ZONE"
```

Replace *TIME_ZONE* with an appropriate time zone identifier. For more information, see the list of time zones.

Para finalizar a instalação

- Reinicie o servidor web e o serviço de armazenamento de sessão:

```
# service apache2 restart
# service memcached restart
```

Verifique a operação

Esta seção descreve como verificar a operação do dashboard.

1. Acesse o dashboard usando um navegador web: `http://controller/horizon`.
2. Autentique-se usando as credenciais de usuário `admin` ou `demo`.

Próximos passos

Seu ambiente OpenStack agora inclui o painel. Você pode lançar uma instância ou adicionar mais serviços ao seu ambiente nos capítulos seguintes.

Depois de instalar e configurar o painel, você pode completar as seguintes tarefas.

- Customize your dashboard. See section Customize the dashboard in the *OpenStack Cloud Administrator Guide* for information on setting up colors, logos, and site titles.
- Set up session storage. See section Set up session storage for the dashboard in the *OpenStack Cloud Administrator Guide* for information on user session data.

Capítulo 8. Adicione o serviço de Block Storage

Índice

The OpenStack Block Storage service provides block storage devices to instances using various backends. The Block Storage API and scheduler services run on the controller node and the volume service runs on one or more storage nodes. Storage nodes provide volumes to instances using local block storage devices or SAN/NAS backends with the appropriate drivers. For more information, see the *Configuration Reference*.

Nota

Este capítulo omite o gerenciador de backup porque ele depende do serviço de Object Storage.

OpenStack Block Storage

The OpenStack Block Storage service (cinder) adds persistent storage to a virtual machine. Block Storage provides an infrastructure for managing volumes, and interacts with OpenStack Compute to provide volumes for instances. The service also enables management of volume snapshots, and volume types.

The Block Storage service consists of the following components:

`cinder-api`	Accepts API requests, and routes them to the `cinder-volume` for action.
`cinder-volume`	Interacts directly with the Block Storage service, and processes such as the `cinder-scheduler`. It also interacts with these processes through a message queue. The `cinder-volume service` responds to read and write requests sent to the Block Storage service to maintain state. It can interact with a variety of storage providers through a driver architecture.
`cinder-scheduler` daemon	Selects the optimal storage provider node on which to create the volume. A similar component to the `nova-scheduler`.
Messaging queue	Routes information between the Block Storage processes.

Instalar e configurar o nodo controlador

Esta seção descreve como instalar e configurar o serviço de Block Storage, apelidado de cinder, no nodo controlador. Este serviço requer pelo menos um nodo de storage adicional que fornecerá volumes às instâncias.

Para configurar pre-requisitos

Antes de você instalar e configurar o serviço de Block Storage, você deve criar uma base de dados e as credenciais do Serviço de Identidade incluindo os endpoints.

1. Para criar a base de dados, complete estes passos:

   a. Utilize o cliente de acesso a banco de dados para acessar o servidor de banco de dados como usuário `root`:

   ```
   $ mysql -u root -p
   ```

   b. Crie a base de dados `cinder` :

   ```
   CREATE DATABASE cinder;
   ```

   c. Conceda os acessos adequados à base de dados `cinder`:

   ```
   GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' \
     IDENTIFIED BY 'CINDER_DBPASS';
   GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'%' \
     IDENTIFIED BY 'CINDER_DBPASS';
   ```

   Substitua *CINDER_DBPASS* com uma senha adequada:

   d. Saia do cliente de acesso a banco de dados.

2. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

   ```
   $ source admin-openrc.sh
   ```

3. Para criar as credenciais do Serviço de Identidade, complete estes passos:

   a. Crie um usuário `cinder`:

   ```
   $ keystone user-create --name cinder --pass CINDER_PASS
   +----------+----------------------------------+
   | Property |              Value               |
   +----------+----------------------------------+
   |  email   |                                  |
   | enabled  |               True               |
   |    id    | 881ab2de4f7941e79504a759a83308be |
   |   name   |              cinder              |
   | username |              cinder              |
   +----------+----------------------------------+
   ```

   Substitua *CINDER_PASS* com uma senha adequada.

   b. Ligue o usuário `cinder` ao `serviço de` tenant e ao papel `admin` :

```
$ keystone user-role-add --user cinder --tenant service --role admin
```

Nota

Este comando não retorna resultados.

c. Crie os serviços do `cinder`:

```
$ keystone service-create --name cinder --type volume \
  --description "OpenStack Block Storage"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |     OpenStack Block Storage      |
|   enabled   |               True               |
|      id     | 1e494c3e22a24baaafcaf777d4d467eb |
|     name    |              cinder              |
|     type    |              volume              |
+-------------+----------------------------------+
$ keystone service-create --name cinderv2 --type volumev2 \
  --description "OpenStack Block Storage"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |     OpenStack Block Storage      |
|   enabled   |               True               |
|      id     | 16e038e449c94b40868277f1d801edb5 |
|     name    |             cinderv2             |
|     type    |             volumev2             |
+-------------+----------------------------------+
```

Nota

O serviço de Block Storage requer dois diferentes serviços para suportar API versão 1 e 2.

d. Criar os endpoints do serviço de Block Storage:

```
$ keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ volume / {print $2}')
 \
  --publicurl http://controlador:8776/v1/%\(tenant_id\)s \
  --internalurl http://controlador:8776/v1/%\(tenant_id\)s \
  --adminurl http://controlador:8776/v1/%\(tenant_id\)s \
  --region regionOne
+-------------+-----------------------------------------+
|   Property  |                  Value                  |
+-------------+-----------------------------------------+
|   adminurl  | http://controller:8776/v1/%(tenant_id)s |
|      id     |    d1b7291a2d794e26963b322c7f2a55a4     |
| internalurl | http://controller:8776/v1/%(tenant_id)s |
|  publicurl  | http://controller:8776/v1/%(tenant_id)s |
|    region   |                regionOne                |
|  service_id |    1e494c3e22a24baaafcaf777d4d467eb     |
+-------------+-----------------------------------------+
$ keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ volumev2 / {print
 $2}') \
```

```
  --publicurl http://controlador:8776/v2/%\(tenant_id\)s \
  --internalurl http://controlador:8776/v2/%\(tenant_id\)s \
  --adminurl http://controlador:8776/v2/%\(tenant_id\)s \
  --region regionOne
+-------------+-----------------------------------------+
|   Property  |                  Value                  |
+-------------+-----------------------------------------+
|   adminurl  | http://controller:8776/v2/%(tenant_id)s |
|      id     |    097b4a6fc8ba44b4b10d4822d2d9e076     |
| internalurl | http://controller:8776/v2/%(tenant_id)s |
|  publicurl  | http://controller:8776/v2/%(tenant_id)s |
|    region   |                regionOne                |
|  service_id |    16e038e449c94b40868277f1d801edb5     |
+-------------+-----------------------------------------+
```

Nota

O serviço de Block Storage requer dois endpoints diferentes para suportar API versão 1 e 2.

Para instalar e configurar os componentes de controlador do Block Storage

1. Instale os pacotes:

```
# apt-get install cinder-api cinder-scheduler python-cinderclient
```

2. Edite o arquivo `/etc/cinder/cinder.conf` e complete as seguintes ações:

   a. Na seção `[database]`, configure o acesso ao banco de dados:

```
[database]
...
connection = mysql://cinder:CINDER_DBPASS@controller/cinder
```

   Substitua *CINDER_DBPASS* com a senha que você escolheu para a base de dados do Block Storage.

   b. Na seção `[DEFAULT]`, configure o acesso ao intermediador RabbitMQ:

```
[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controlador
rabbit_password = RABBIT_PASS
```

   Substitua *RABBIT_PASS* com a senha que você escolheu para a conta `guest` no RabbitMQ.

   c. Nas seções `[DEFAULT]` e `[keystone_authtoken]`, configure o acesso do Serviço de Identidade:

```
[DEFAULT]
...
auth_strategy = keystone

[keystone_authtoken]
...
auth_uri = http://controlador:5000/v2.0
identity_uri = http://controlador:35357
admin_tenant_name = service
admin_user = cinder
admin_password = CINDER_PASS
```

Substitua *CINDER_PASS* com a senha que você escolheu para o usuário `cinder` no serviço de Identidade.

Nota

Comente quaisquer opções `auth_host`, `auth_port`, e `auth_protocol`, porque a opção `identity_uri` as substitui.

d. Na seção `[DEFAULT]`, configure a opção `my_ip` para utilizar o endereço IP da interface de gerenciamento do nodo controlador:

```
[DEFAULT]
...
my_ip = 10.0.0.11
```

e. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção `[DEFAULT]` :

```
[DEFAULT]
...
verbose = True
```

3. Popule a base de dados do Block Storage:

```
# su -s /bin/sh -c "cinder-manage db sync" cinder
```

Para finalizar a instalação

1. Reinicie os serviços de Block Storage:

```
# service cinder-scheduler restart
# service cinder-api restart
```

2. Por padrão, os pacotes do Ubuntu criam uma base de dados SQLite.

   Devido esta configuração utilizar um servidor de banco de dados SQL, você pode remover o arquivo de banco de dados SQLite:

```
# rm -f /var/lib/cinder/cinder.sqlite
```

Instalar e configurar um nodo de storage

Esta seção descreve como instalar e configurar nodos de storage para o serviço de Block Storage. Para simplificar, esta configuração referencia um nodo de storage com um dispo-

sitivo vazio de armazenamento em bloco `/dev/sdb`, que contém uma tabela de partição adequada, com uma partição `/dev/sdb1` ocupando o dispositivo inteiro. O serviço provisiona volumes lógicos neste dispositivo utilizando o driver *LVM* e os fornece às instâncias via transporte *iSCSI*. Você pode seguir estas instruções com pequenas modificações para escalar horizontalmente seu ambiente com nodos de storage adicionais.

Para configurar pre-requisitos

Você deve configurar o nodo de storage antes de instalar e configurar o serviço de volume nele. Similar ao nodo de computação, o nodo de storage contém uma interface de rede na *rede de gerenciamento*. O nodo de storage também precisa de um dispositivo de blocos vazio de tamanho adequado para seu ambiente. Para mais informações, consulte Capítulo 2, Ambiente básico [11].

1. Configurar a interface de gerenciamento:

   Endereço IP: 10.0.0.41

   Máscara de rede: 255.255.255.0 (ou /24)

   Gateway padrão: 10.0.0.1

2. Defina o hostname do nodo para *`block1`*.

3. Copie o conteúdo do arquivo `/etc/hosts` do nodo controlador para o nodo de storage e adicione o seguinte a ele:

   ```
   # block1
   10.0.0.41       block1
   ```

   Adicione também este conteúdo ao arquivo `/etc/hosts` em todos os outros nodos em seu ambiente.

4. Instale e configure o *NTP* utilizando as instruções em "Outros nodos" [25].

5. Instalar os pacotes LVM:

   ```
   # apt-get install lvm2
   ```

   Nota

   Algumas distribuições incluem o LVM por padrão.

6. Crie o volume LVM físico `/dev/sdb1`:

   ```
   # pvcreate /dev/sdb1
     Physical volume "/dev/sdb1" successfully created
   ```

   Nota

   Se seu sistema utiliza um nome de dispositivo diferente, ajuste esses passos de acordo.

7. Crie o grupo de volume LVM `cinder-volumes`:

   ```
   # vgcreate cinder-volumes /dev/sdb1
   ```

```
  Volume group "cinder-volumes" successfully created
```

O serviço de Block Storage cria volumes lógicos neste grupo de volume.

8. Somente instâncias podem acessar os volumes do Block Storage. Contudo, o sistema operacional subjacente gerencia os dispositivos associados a estes volumes. Por padrão, a ferramenta de varredura de volume LVM varre o diretório `/dev` por dispositivos de armazenamento em blocos que contêm volumes. Se os tenants utilizam LVM em seus volumes, a ferramenta de varredura detecta estes volumes e tenta fazer cache deles, o que pode causar uma variedade de problemas com ambos, o sistema operacional subjacente e os volumes dos tenants. Você deve reconfigurar o LVM para varrer somente os dispositivos que contêm o grupo de volume `cinder-volume`. Edite o arquivo `/etc/lvm/lvm.conf` e complete as seguintes ações:

   - Na seção `devices`, adicione um filtro que aceite o dispositivo `/dev/sdb` e rejeite todos os outros dispositivos:

     ```
     devices {
     ...
     filter = [ "a/sdb/", "r/.*/"]
     ```

     Cada item na matriz de filtro começa com `a` para *aceitar* ou `r` para *rejeitar* e inclui uma expressão regular para o nome do dispositivo. A matriz deve finalizar com `r/.*/` para rejeitar qualquer dispositivo restante. Você pode escolher utilizar os comandos **vgs -vvvv** para testar os filtros.

     ### Atenção

     Se seus nodos de storage utilizam LVM no disco do sistema operacional, você deve também adicionar o dispositivo associado ao filtro. Por exemplo, se o dispositivo `/dev/sda` contém o sistema operacional:

     ```
     filter = [ "a/sda/", "a/sdb/", "r/.*/"]
     ```

     Da mesma forma, se seus nodos de computação utilizam LVM no disco do sistema operacional, você deve também modificar o filtro no arquivo `/etc/lvm/lvm.conf` nesses nodos para incluir somente o disco do sistema operacional. Por exemplo, se o dispositivo `/dev/sda` contém o sistema operacional:

     ```
     filter = [ "a/sda/", "r/.*/"]
     ```

Instalar e configurar os componentes de volume do Block Storage

1. Instale os pacotes:

   ```
   # apt-get install cinder-volume python-mysqldb
   ```

2. Edite o arquivo `/etc/cinder/cinder.conf` e complete as seguintes ações:

   a. Na seção `[database]`, configure o acesso ao banco de dados:

      ```
      [database]
      ...
      connection = mysql://cinder:CINDER_DBPASS@controlador/cinder
      ```

Substitua *CINDER_DBPASS* com a senha que você escolheu para a base de dados do Block Storage.

b. Na seção [DEFAULT], configure o acesso ao intermediador RabbitMQ:

```
[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controlador
rabbit_password = RABBIT_PASS
```

Substitua *RABBIT_PASS* com a senha que você escolheu para a conta guest no RabbitMQ.

c. Nas seções [DEFAULT] e [keystone_authtoken], configure o acesso do Serviço de Identidade:

```
[DEFAULT]
...
auth_strategy = keystone

[keystone_authtoken]
...
auth_uri = http://controlador:5000/v2.0
identity_uri = http://controlador:35357
admin_tenant_name = service
admin_user = cinder
admin_password = CINDER_PASS
```

Substitua *CINDER_PASS* com a senha que você escolheu para o usuário cinder no serviço de Identidade.

Nota

Comente quaisquer opções auth_host, auth_port, e auth_protocol, porque a opção identity_uri as substitui.

d. Na seção [DEFAULT], configure a opção my_ip:

```
[DEFAULT]
...
my_ip = MANAGEMENT_INTERFACE_IP_ADDRESS
```

Substitua *MANAGEMENT_INTERFACE_IP_ADDRESS* com o endereço IP da interface de gerenciamento de rede no seu nodo de storage, tipicamente 10.0.0.41 para o primeiro nodo na arquitetura de exemplo.

e. Na seção [DEFAULT], configure a localização do Serviço de Imagem:

```
[DEFAULT]
...
glance_host = controlador
```

f. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção [DEFAULT] :

```
[DEFAULT]
...
verbose = True
```

Para finalizar a instalação

1. Reinicie o serviço de volume do Block Storage incluindo suas dependências:

```
# service tgt restart
# service cinder-volume restart
```

2. Por padrão, os pacotes do Ubuntu criam uma base de dados SQLite. Como esta configuração utiliza um servidor de banco de dados SQL, remova o arquivo de base de dados SQLite.

```
# rm -f /var/lib/cinder/cinder.sqlite
```

Verifique a operação

Esta seção descreve como verificar a operação do serviço de Block Storage através da criação de um volume.

For more information about how to manage volumes, see the *OpenStack User Guide*.

Nota

Execute estes comandos no nodo controlador.

1. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

```
$ source admin-openrc.sh
```

2. Liste os componentes de serviço para verificar o lançamento com sucesso de cada processo:

```
$ cinder service-list
+------------------+------------+------+---------+-------
+----------------------------+-----------------+
|      Binary      |    Host    | Zone |  Status | State |
 Updated_at         | Disabled Reason |
+------------------+------------+------+---------+-------
+----------------------------+-----------------+
| cinder-scheduler | controller | nova | enabled |   up  |
 2014-10-18T01:30:54.000000 |       None      |
|  cinder-volume   |   block1   | nova | enabled |   up  |
 2014-10-18T01:30:57.000000 |       None      |
+------------------+------------+------+---------+-------
+----------------------------+-----------------+
```

3. Obtenha as credenciais do tenant `demo` para realizar os seguintes passos como um tenant não-administrativo:

```
$ source demo-openrc.sh
```

4. Crie um volume de 1 GB:

```
$ cinder create --display-name demo-volume1 1
+---------------------+--------------------------------------+
|       Property      |                Value                 |
+---------------------+--------------------------------------+
|     attachments     |                  []                  |
|  availability_zone  |                nova                  |
|       bootable      |               false                  |
|      created_at     |      2014-10-14T23:11:50.870239      |
| display_description |                None                  |
|     display_name    |             demo-volume1             |
|      encrypted      |               False                  |
|          id         | 158bea89-07db-4ac2-8115-66c0d6a4bb48 |
|       metadata      |                  {}                  |
|         size        |                  1                   |
|     snapshot_id     |                None                  |
|     source_volid    |                None                  |
|        status       |              creating                |
|     volume_type     |                None                  |
+---------------------+--------------------------------------+
```

5. Verifique a criação e a disponibilidade do volume:

```
$ cinder list
--------------------------------------+-----------+--------------+------
+-------------+----------+-------------+
|                  ID                  |   Status  | Display Name | Size |
 Volume Type | Bootable | Attached to |
+--------------------------------------+-----------+--------------+------
+-------------+----------+-------------+
| 158bea89-07db-4ac2-8115-66c0d6a4bb48 | available | demo-volume1 |  1   |
     None    |  false   |             |
+--------------------------------------+-----------+--------------+------
+-------------+----------+-------------+
```

Se o status não indicar `disponível`, cheque os logs no diretório `/var/log/cinder` nos nodos controlador e de volume para mais informações.

Nota

O capítulo lançar uma instância inclui instruções para conectar este volume a uma instância.

Próximos passos

Seu ambiente OpenStack agora inclui o Block Storage. Você pode lançar uma instância ou adicionar mais serviços ao seu ambiente nos capítulos seguintes.

Capítulo 9. Adicionar Object Storage

Índice

O serviços OpenStack de Object Storage (swift) trabalham em conjunto para fornecer armazenamento e recuperação de objeto através de uma API *REST*. Seu ambiente deve incluir, pelo menos, o serviço de Identidade (keystone) antes da implantação do Object Storage.

OpenStack Object Storage

The OpenStack Object Storage is a multi-tenant object storage system. It is highly scalable and can manage large amounts of unstructured data at low cost through a RESTful HTTP API.

It includes the following components:

Proxy servers (`swift-proxy-server`)	Accepts OpenStack Object Storage API and raw HTTP requests to upload files, modify metadata, and create containers. It also serves file or container listings to web browsers. To improve performance, the proxy server can use an optional cache that is usually deployed with memcache.
Account servers (`swift-account-server`)	Manages accounts defined with Object Storage.
Container servers (`swift-container-server`)	Manages the mapping of containers or folders, within Object Storage.
Object servers (`swift-object-server`)	Manages actual objects,such as files, on the storage nodes.
Various periodic processes	Performs housekeeping tasks on the large data store. The replication services ensure consistency and availability through the cluster. Other periodic processes include auditors, updaters, and reapers.
WSGI middleware	Handles authentication and is usually OpenStack Identity.

Install and configure the controller node

This section describes how to install and configure the proxy service that handles requests for the account, container, and object services operating on the storage nodes. For simplicity, this guide installs and configures the proxy service on the controller node. However, you can run the proxy service on any node with network connectivity to the storage nodes. Additionally, you can install and configure the proxy service on multiple nodes to increase performance and redundancy. For more information, see the Deployment Guide.

Para configurar pre-requisitos

The proxy service relies on an authentication and authorization mechanism such as the Identity service. However, unlike other services, it also offers an internal mechanism that allows it to operate without any other OpenStack services. However, for simplicity, this guide references the Identity service in Capítulo 3, Adicione o serviço de Identidade [30]. Before you configure the Object Storage service, you must create Identity service credentials including endpoints.

Nota

The Object Storage service does not use a SQL database on the controller node.

1. Para criar as credenciais do Serviço de Identidade, complete estes passos:

   a. Create a `swift` user:

```
$ keystone user-create --name swift --pass SWIFT_PASS
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |                                  |
| enabled  |               True               |
|    id    | d535e5cbd2b74ac7bfb97db9cced3ed6 |
|   name   |              swift               |
| username |              swift               |
+----------+----------------------------------+
```

   Replace *SWIFT_PASS* with a suitable password.

   b. Link the `swift` user to the `service` tenant and `admin` role:

```
$ keystone user-role-add --user swift --tenant service --role admin
```

Nota

Este comando não retorna resultados.

   c. Create the `swift` service:

```
$ keystone service-create --name swift --type object-store \
  --description "OpenStack Object Storage"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |     OpenStack Object Storage     |
```

```
|   enabled   |              True                |
|      id     | 75ef509da2c340499d454ae96a2c5c34 |
|     name    |              swift               |
|     type    |           object-store           |
+-------------+----------------------------------+
```

2. Crie os endpoints do Serviço de Identidade:

```
$ keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ object-store / {print
 $2}') \
  --publicurl 'http://controlador:8080/v1/AUTH_%(tenant_id)s' \
  --internalurl 'http://controlador:8080/v1/AUTH_%(tenant_id)s' \
  --adminurl http://controlador:8080 \
  --region regionOne
+-------------+-----------------------------------------------------+
|   Property  |                        Value                        |
+-------------+-----------------------------------------------------+
|   adminurl  |            http://controller:8080/                  |
|      id     |          af534fb8b7ff40a6acf725437c586ebe           |
| internalurl | http://controller:8080/v1/AUTH_%(tenant_id)s        |
|  publicurl  | http://controller:8080/v1/AUTH_%(tenant_id)s        |
|    region   |                      regionOne                      |
|  service_id |          75ef509da2c340499d454ae96a2c5c34           |
+-------------+-----------------------------------------------------+
```

To install and configure the controller node components

1. Instale os pacotes:

Nota

Complete OpenStack environments already include some of these packages.

```
# apt-get install swift swift-proxy python-swiftclient python-
keystoneclient \
  python-keystonemiddleware memcached
```

2. Create the `/etc/swift` directory.

3. Obtain the proxy service configuration file from the Object Storage source repository:

```
# curl -o /etc/swift/proxy-server.conf \
  https://raw.githubusercontent.com/openstack/swift/stable/juno/etc/proxy-
server.conf-sample
```

4. Edit the `/etc/swift/proxy-server.conf` file and complete the following actions:

   a. In the `[DEFAULT]` section, configure the bind port, user, and configuration directory:

```
[DEFAULT]
...
bind_port = 8080
user = swift
swift_dir = /etc/swift
```

   b. In the `[pipeline:main]` section, enable the appropriate modules:

```
[pipeline:main]
pipeline = authtoken cache healthcheck keystoneauth proxy-logging
 proxy-server
```

Nota

For more information on other modules that enable additional features, see the Deployment Guide.

c. In the `[app:proxy-server]` section, enable account management:

```
[app:proxy-server]
...
allow_account_management = true
account_autocreate = true
```

d. In the `[filter:keystoneauth]` section, configure the operator roles:

```
[filter:keystoneauth]
use = egg:swift#keystoneauth
...
operator_roles = admin,_member_
```

Nota

You might need to uncomment this section.

e. In the `[filter:authtoken]` section, configure Identity service access:

```
[filter:authtoken]
paste.filter_factory = keystonemiddleware.auth_token:filter_factory
...
auth_uri = http://controlador:5000/v2.0
identity_uri = http://controlador:35357
admin_tenant_name = service
admin_user = swift
admin_password = SWIFT_PASS
delay_auth_decision = true
```

Replace *SWIFT_PASS* with the password you chose for the `swift` user in the Identity service.

Nota

You might need to uncomment this section.

Nota

Comente quaisquer opções `auth_host`, `auth_port`, e `auth_protocol`, porque a opção `identity_uri` as substitui.

f. In the `[filter:cache]` section, configure the memcached location:

```
[filter:cache]
...
memcache_servers = 127.0.0.1:11211
```

Install and configure the storage nodes

This section describes how to install and configure storage nodes that operate the account, container, and object services. For simplicity, this configuration references two storage nodes, each containing two empty local block storage devices. Each of the devices, `/dev/sdb` and `/dev/sdc`, must contain a suitable partition table with one partition occupying the entire device. Although the Object Storage service supports any file system with *extended attributes (xattr)*, testing and benchmarking indicate the best performance and reliability on *XFS*. For more information on horizontally scaling your environment, see the Deployment Guide.

Para configurar pre-requisitos

You must configure each storage node before you install and configure the Object Storage service on it. Similar to the controller node, each storage node contains one network interface on the *management network*. Optionally, each storage node can contain a second network interface on a separate network for replication. For more information, see Capítulo 2, Ambiente básico [11].

1. Configure unique items on the first storage node:

   a. Configurar a interface de gerenciamento:

      IP address: 10.0.0.51

      Máscara de rede: 255.255.255.0 (ou /24)

      Gateway padrão: 10.0.0.1

   b. Set the hostname of the node to *`object1`*.

2. Configure unique items on the second storage node:

   a. Configurar a interface de gerenciamento:

      IP address: 10.0.0.52

      Máscara de rede: 255.255.255.0 (ou /24)

      Gateway padrão: 10.0.0.1

   b. Set the hostname of the node to *`object2`*.

3. Configure shared items on both storage nodes:

   a. Copy the contents of the `/etc/hosts` file from the controller node and add the following to it:

```
# object1
10.0.0.51       object1

# object2
10.0.0.52       object2
```

Adicione também este conteúdo ao arquivo `/etc/hosts` em todos os outros nodos em seu ambiente.

b. Instale e configure o *NTP* utilizando as instruções em "Outros nodos" [25].

c. Install the supporting utility packages:

```
# apt-get install xfsprogs rsync
```

d. Format the `/dev/sdb1` and `/dev/sdc1` partitions as XFS:

```
# mkfs.xfs /dev/sdb1
# mkfs.xfs /dev/sdc1
```

e. Create the mount point directory structure:

```
# mkdir -p /srv/node/sdb1
# mkdir -p /srv/node/sdc1
```

f. Edit the `/etc/fstab` file and add the following to it:

```
/dev/sdb1 /srv/node/sdb1 xfs noatime,nodiratime,nobarrier,logbufs=8 0
 2
/dev/sdc1 /srv/node/sdc1 xfs noatime,nodiratime,nobarrier,logbufs=8 0
 2
```

g. Mount the devices:

```
# mount /srv/node/sdb1
# mount /srv/node/sdc1
```

4. Edit the `/etc/rsyncd.conf` file and add the following to it:

```
uid = swift
gid = swift
log file = /var/log/rsyncd.log
pid file = /var/run/rsyncd.pid
address = MANAGEMENT_INTERFACE_IP_ADDRESS

[account]
max connections = 2
path = /srv/node/
read only = false
lock file = /var/lock/account.lock

[container]
max connections = 2
path = /srv/node/
read only = false
lock file = /var/lock/container.lock

[object]
max connections = 2
path = /srv/node/
read only = false
lock file = /var/lock/object.lock
```

Replace *MANAGEMENT_INTERFACE_IP_ADDRESS* with the IP address of the management network on the storage node.

Nota

The `rsync` service requires no authentication, so consider running it on a private network.

5. Edit the `/etc/default/rsync` file and enable the `rsync` service:

```
RSYNC_ENABLE=true
```

6. Inicie o serviço `rsync`:

```
# service rsync start
```

Install and configure storage node components

Nota

Perform these steps on each storage node.

1. Instale os pacotes:

```
# apt-get install swift swift-account swift-container swift-object
```

2. Obtain the accounting, container, and object service configuration files from the Object Storage source repository:

```
# curl -o /etc/swift/account-server.conf \
  https://raw.githubusercontent.com/openstack/swift/stable/juno/etc/
account-server.conf-sample
```

```
# curl -o /etc/swift/container-server.conf \
  https://raw.githubusercontent.com/openstack/swift/stable/juno/etc/
container-server.conf-sample
```

```
# curl -o /etc/swift/object-server.conf \
  https://raw.githubusercontent.com/openstack/swift/stable/juno/etc/
object-server.conf-sample
```

3. Edit the `/etc/swift/account-server.conf` file and complete the following actions:

   a. In the `[DEFAULT]` section, configure the bind IP address, bind port, user, configuration directory, and mount point directory:

```
[DEFAULT]
...
bind_ip = MANAGEMENT_INTERFACE_IP_ADDRESS
bind_port = 6002
user = swift
swift_dir = /etc/swift
devices = /srv/node
```

   Replace *MANAGEMENT_INTERFACE_IP_ADDRESS* with the IP address of the management network on the storage node.

b. In the `[pipeline:main]` section, enable the appropriate modules:

```
[pipeline:main]
pipeline = healthcheck recon account-server
```

Nota

For more information on other modules that enable additional features, see the Deployment Guide.

c. In the `[filter:recon]` section, configure the recon (metrics) cache directory:

```
[filter:recon]
...
recon_cache_path = /var/cache/swift
```

4. Edit the `/etc/swift/container-server.conf` file and complete the following actions:

a. In the `[DEFAULT]` section, configure the bind IP address, bind port, user, configuration directory, and mount point directory:

```
[DEFAULT]
...
bind_ip = MANAGEMENT_INTERFACE_IP_ADDRESS
bind_port = 6001
user = swift
swift_dir = /etc/swift
devices = /srv/node
```

Replace *MANAGEMENT_INTERFACE_IP_ADDRESS* with the IP address of the management network on the storage node.

b. In the `[pipeline:main]` section, enable the appropriate modules:

```
[pipeline:main]
pipeline = healthcheck recon container-server
```

Nota

For more information on other modules that enable additional features, see the Deployment Guide.

c. In the `[filter:recon]` section, configure the recon (metrics) cache directory:

```
[filter:recon]
...
recon_cache_path = /var/cache/swift
```

5. Edit the `/etc/swift/object-server.conf` file and complete the following actions:

a. In the `[DEFAULT]` section, configure the bind IP address, bind port, user, configuration directory, and mount point directory:

```
[DEFAULT]
...
bind_ip = MANAGEMENT_INTERFACE_IP_ADDRESS
bind_port = 6000
user = swift
swift_dir = /etc/swift
devices = /srv/node
```

Replace *MANAGEMENT_INTERFACE_IP_ADDRESS* with the IP address of the management network on the storage node.

b. In the `[pipeline:main]` section, enable the appropriate modules:

```
[pipeline:main]
pipeline = healthcheck recon object-server
```

Nota

For more information on other modules that enable additional features, see the Deployment Guide.

c. In the `[filter:recon]` section, configure the recon (metrics) cache directory:

```
[filter:recon]
...
recon_cache_path = /var/cache/swift
```

6. Ensure proper ownership of the mount point directory structure:

```
# chown -R swift:swift /srv/node
```

7. Create the `recon` directory and ensure proper ownership of it:

```
# mkdir -p /var/cache/swift
# chown -R swift:swift /var/cache/swift
```

Create initial rings

Before starting the Object Storage services, you must create the initial account, container, and object rings. The ring builder creates configuration files that each node uses to determine and deploy the storage architecture. For simplicity, this guide uses one region and zone with 2^10 (1024) maximum partitions, 3 replicas of each object, and 1 hour minimum time between moving a partition more than once. For Object Storage, a partition indicates a directory on a storage device rather than a conventional partition table. For more information, see the Deployment Guide.

Account ring

The account server uses the account ring to maintain lists of containers.

To create the ring

Nota

Perform these steps on the controller node.

1. Change to the `/etc/swift` directory.

2. Create the base `account.builder` file:

```
# swift-ring-builder account.builder create 10 3 1
```

3. Add each storage node to the ring:

```
# swift-ring-builder account.builder \
  add
 r1z1-STORAGE_NODE_MANAGEMENT_INTERFACE_IP_ADDRESS:6002/DEVICE_NAME DEVICE_WEIGHT
```

Replace *STORAGE_NODE_MANAGEMENT_INTERFACE_IP_ADDRESS* with the IP address of the management network on the storage node. Replace *DEVICE_NAME* with a storage device name on the same storage node. For example, using the first storage node in "Install and configure the storage nodes" [104] with the `/dev/sdb1` storage device and weight of 100:

```
# swift-ring-builder account.builder add r1z1-10.0.0.51:6002/sdb1 100
```

Repeat this command for each storage device on each storage node. The example architecture requires four variations of this command.

4. Verify the ring contents:

```
# swift-ring-builder account.builder
account.builder, build version 4
1024 partitions, 3.000000 replicas, 1 regions, 1 zones, 4 devices, 0.00
 balance
The minimum number of hours before a partition can be reassigned is 1
Devices:    id  region  zone      ip address  port  replication ip
 replication port      name weight partitions balance meta
             0       1     1       10.0.0.51  6002       10.0.0.51
       6002      sdb1 100.00        768    0.00
             1       1     1       10.0.0.51  6002       10.0.0.51
       6002      sdc1 100.00        768    0.00
             2       1     1       10.0.0.52  6002       10.0.0.52
       6002      sdb1 100.00        768    0.00
             3       1     1       10.0.0.52  6002       10.0.0.52
       6002      sdc1 100.00        768    0.00
```

5. Rebalance the ring:

```
# swift-ring-builder account.builder rebalance
```

Nota

This process can take a while.

Container ring

The container server uses the container ring to maintain lists of objects. However, it does not track object locations.

To create the ring

Nota

Perform these steps on the controller node.

1. Change to the `/etc/swift` directory.

2. Create the base `container.builder` file:

```
# swift-ring-builder container.builder create 10 3 1
```

3. Add each storage node to the ring:

```
# swift-ring-builder container.builder \
  add
 r1z1-STORAGE_NODE_MANAGEMENT_INTERFACE_IP_ADDRESS:6001/DEVICE_NAME DEVICE_WEIGHT
```

Replace *STORAGE_NODE_MANAGEMENT_INTERFACE_IP_ADDRESS* with the IP address of the management network on the storage node. Replace *DEVICE_NAME* with a storage device name on the same storage node. For example, using the first storage node in "Install and configure the storage nodes" [104] with the `/dev/sdb1` storage device and weight of 100:

```
# swift-ring-builder container.builder add r1z1-10.0.0.51:6001/sdb1 100
```

Repeat this command for each storage device on each storage node. The example architecture requires four variations of this command.

4. Verify the ring contents:

```
# swift-ring-builder container.builder
container.builder, build version 4
1024 partitions, 3.000000 replicas, 1 regions, 1 zones, 4 devices, 0.00
 balance
The minimum number of hours before a partition can be reassigned is 1
Devices:    id  region  zone      ip address  port  replication ip
 replication port      name weight partitions balance meta
             0       1     1       10.0.0.51  6001       10.0.0.51
       6001      sdb1 100.00        768    0.00
             1       1     1       10.0.0.51  6001       10.0.0.51
       6001      sdc1 100.00        768    0.00
             2       1     1       10.0.0.52  6001       10.0.0.52
       6001      sdb1 100.00        768    0.00
             3       1     1       10.0.0.52  6001       10.0.0.52
       6001      sdc1 100.00        768    0.00
```

5. Rebalance the ring:

```
# swift-ring-builder container.builder rebalance
```

Nota

This process can take a while.

Object ring

The object server uses the object ring to maintain lists of object locations on local devices.

To create the ring

Nota

Perform these steps on the controller node.

1. Change to the `/etc/swift` directory.

2. Create the base `object.builder` file:

```
# swift-ring-builder object.builder create 10 3 1
```

3. Add each storage node to the ring:

```
# swift-ring-builder object.builder \
  add
 r1z1-STORAGE_NODE_MANAGEMENT_INTERFACE_IP_ADDRESS:6000/DEVICE_NAME DEVICE_WEIGHT
```

Replace *STORAGE_NODE_MANAGEMENT_INTERFACE_IP_ADDRESS* with the IP address of the management network on the storage node. Replace *DEVICE_NAME* with a storage device name on the same storage node. For example, using the first storage node in "Install and configure the storage nodes" [104] with the `/dev/sdb1` storage device and weight of 100:

```
# swift-ring-builder object.builder add r1z1-10.0.0.51:6000/sdb1 100
```

Repeat this command for each storage device on each storage node. The example architecture requires four variations of this command.

4. Verify the ring contents:

```
# swift-ring-builder object.builder
object.builder, build version 4
1024 partitions, 3.000000 replicas, 1 regions, 1 zones, 4 devices, 0.00
 balance
The minimum number of hours before a partition can be reassigned is 1
Devices:    id  region  zone      ip address  port  replication ip
 replication port      name weight partitions balance meta
             0       1     1       10.0.0.51  6000       10.0.0.51
       6000      sdb1 100.00        768    0.00
             1       1     1       10.0.0.51  6000       10.0.0.51
       6000      sdc1 100.00        768    0.00
             2       1     1       10.0.0.52  6000       10.0.0.52
       6000      sdb1 100.00        768    0.00
             3       1     1       10.0.0.52  6000       10.0.0.52
       6000      sdc1 100.00        768    0.00
```

5. Rebalance the ring:

```
# swift-ring-builder object.builder rebalance
```

Nota

This process can take a while.

Distribute ring configuration files

Copy the `account.ring.gz`, `container.ring.gz`, and `object.ring.gz` files to the `/etc/swift` directory on each storage node and any additional nodes running the proxy service.

Finalize installation

Configure hashes and default storage policy

1. Obtain the `/etc/swift/swift.conf` file from the Object Storage source repository:

```
# curl -o /etc/swift/swift.conf \
  https://raw.githubusercontent.com/openstack/swift/stable/juno/etc/swift.
conf-sample
```

2. Edit the `/etc/swift/swift.conf` file and complete the following actions:

   a. In the `[swift-hash]` section, configure the hash path prefix and suffix for your environment.

```
[swift-hash]
...
swift_hash_path_suffix = HASH_PATH_PREFIX
swift_hash_path_prefix = HASH_PATH_SUFFIX
```

   Replace *HASH_PATH_PREFIX* and *HASH_PATH_SUFFIX* with unique values.

Atenção

Keep these values secret and do not change or lose them.

   b. In the `[storage-policy:0]` section, configure the default storage policy:

```
[storage-policy:0]
...
name = Policy-0
default = yes
```

3. Copy the `swift.conf` file to the `/etc/swift` directory on each storage node and any additional nodes running the proxy service.

4. On all nodes, ensure proper ownership of the configuration directory:

```
# chown -R swift:swift /etc/swift
```

5. On the controller node and any other nodes running the proxy service, restart the Object Storage proxy service including its dependencies:

```
# service memcached restart
# service swift-proxy restart
```

6. On the storage nodes, start the Object Storage services:

```
# swift-init all start
```

Nota

The storage node runs many Object Storage services and the **swift-init** command makes them easier to manage. You can ignore errors from services not running on the storage node.

Verifique a operação

This section describes how to verify operation of the Object Storage service.

Nota

Perform these steps on the controller node.

1. Obtenha as credenciais do tenant `demo` :

```
$ source demo-openrc.sh
```

2. Show the service status:

```
$ swift stat
Account: AUTH_11b9758b7049476d9b48f7a91ea11493
Containers: 0
   Objects: 0
     Bytes: 0
Content-Type: text/plain; charset=utf-8
X-Timestamp: 1381434243.83760
X-Trans-Id: txdcdd594565214fb4a2d33-0052570383
X-Put-Timestamp: 1381434243.83760
```

3. Upload a test file:

```
$ swift upload demo-container1 FILE
```

Replace *`FILE`* with the name of a local file to upload to the `demo-container1` container.

4. List containers:

```
$ swift list
demo-container1
```

5. Download a test file:

```
$ swift download demo-container1 FILE
```

Replace *`FILE`* with the name of the file uploaded to the `demo-container1` container.

Próximos passos

Seu ambiente OpenStack inclui agora o Block Storage. Você pode lançar uma instância ou adicionar mais serviços ao seu ambiente nos capítulos seguintes.

Capítulo 10. Adicione o módulo de Orquestração

Índice

O módulo de Orquestração (heat) utiliza o modelo heat orchestration (HOT) para criar e gerenciar recursos de nuvem.

Orchestration module concepts

The Orchestration module provides a template-based orchestration for describing a cloud application, by running OpenStack API calls to generate running cloud applications. The software integrates other core components of OpenStack into a one-file template system. The templates allow you to create most OpenStack resource types, such as instances, floating IPs, volumes, security groups and users. It also provides advanced functionality, such as instance high availability, instance auto-scaling, and nested stacks. This enables OpenStack core projects to receive a larger user base.

The service enables deployers to integrate with the Orchestration module directly or through custom plug-ins.

The Orchestration module consists of the following components:

`heat` command-line client — A CLI that communicates with the heat-api to run AWS CloudFormation APIs. End developers can directly use the Orchestration REST API.

`heat-api` component — An OpenStack-native REST API that processes API requests by sending them to the heat-engine over Remote Procedure Call (RPC).

`heat-api-cfn` component — An AWS Query API that is compatible with AWS CloudFormation. It processes API requests by sending them to the heat-engine over RPC.

`heat-engine` — Orchestrates the launching of templates and provides events back to the API consumer.

Instale e configure a Orquestração

Esta seção descreve como instalar e configurar o módulo de Orquestração, apelidado de heat, no nodo controlador.

Para configurar pre-requisitos

Antes de você instalar e configurar a Orquestração, você deve criar credenciais para base de dados e serviço de Identidade incluindo endpoints.

1. Para criar a base de dados, complete estes passos:

   a. Utilize o cliente de acesso a banco de dados para acessar o servidor de banco de dados como usuário `root`:

   ```
   $ mysql -u root -p
   ```

   b. Crie a base de dados `heat`:

   ```
   CREATE DATABASE heat;
   ```

   c. Conceda as pemissões apropriadas à base de dados do `heat`:

   ```
   GRANT ALL PRIVILEGES ON heat.* TO 'heat'@'localhost' \
     IDENTIFIED BY 'HEAT_DBPASS';
   GRANT ALL PRIVILEGES ON heat.* TO 'heat'@'%' \
     IDENTIFIED BY 'HEAT_DBPASS';
   ```

   Substitua *HEAT_DBPASS* com uma senha adequada.

   d. Saia do cliente de acesso a banco de dados.

2. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

   ```
   $ source admin-openrc.sh
   ```

3. Para criar as credenciais do Serviço de Identidade, complete estes passos:

   a. Crie o usuário `heat`:

   ```
   $ keystone user-create --name heat --pass HEAT_PASS
   +----------+----------------------------------+
   | Property |              Value               |
   +----------+----------------------------------+
   |  email   |                                  |
   | enabled  |               True               |
   |    id    | 7fd67878dcd04d0393469ef825a7e005 |
   |   name   |               heat               |
   | username |               heat               |
   +----------+----------------------------------+
   ```

   Substitua *HEAT_PASS* com uma senha adequada.

   b. Ligue o usuário `heat` ao tenant de `serviço` e ao papel `admin`:

   ```
   $ keystone user-role-add --user heat --tenant service --role admin
   ```

   Nota

   Este comando não retorna resultados.

   c. Create the `heat_stack_owner` role:

```
$ keystone role-create --name heat_stack_owner
```

d. Add the `heat_stack_owner` role to the `demo` tenant and user:

```
$ keystone user-role-add --user demo --tenant demo --role
 heat_stack_owner
```

Nota

You must add the `heat_stack_owner` role to users that manage stacks.

e. Create the `heat_stack_user` role:

```
$ keystone role-create --name heat_stack_user
```

Nota

The Orchestration service automatically assigns the `heat_stack_user` role to users that it creates during stack deployment. By default, this role restricts *API* operations. To avoid conflicts, do not add this role to users with the `heat_stack_owner` role.

f. Crie os serviços `heat` e `heat-cfn`:

```
$ keystone service-create --name heat --type orchestration \
  --description "Orchestration"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |          Orchestration           |
|   enabled   |               True               |
|      id     | 031112165cad4c2bb23e84603957de29 |
|     name    |               heat               |
|     type    |          orchestration           |
+-------------+----------------------------------+
$ keystone service-create --name heat-cfn --type cloudformation \
  --description "Orchestration"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |          Orchestration           |
|   enabled   |               True               |
|      id     | 297740d74c0a446bbff867acdccb33fa |
|     name    |             heat-cfn             |
|     type    |          cloudformation          |
+-------------+----------------------------------+
```

g. Crie os endpoints do Serviço de Identidade:

```
$ keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ orchestration / {print
 $2}') \
  --publicurl http://controlador:8004/v1/%\(tenant_id\)s \
  --internalurl http://controlador:8004/v1/%\(tenant_id\)s \
  --adminurl http://controlador:8004/v1/%\(tenant_id\)s \
  --region regionOne
```

```
+-------------+-----------------------------------------+
|   Property  |                  Value                  |
+-------------+-----------------------------------------+
|   adminurl  | http://controller:8004/v1/%(tenant_id)s |
|      id     |     f41225f665694b95a46448e8676b0dc2    |
| internalurl | http://controller:8004/v1/%(tenant_id)s |
|  publicurl  | http://controller:8004/v1/%(tenant_id)s |
|    region   |                regionOne                |
|  service_id |     031112165cad4c2bb23e84603957de29    |
+-------------+-----------------------------------------+
$ keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ cloudformation /
 {print $2}') \
  --publicurl http://controlador:8000/v1 \
  --internalurl http://controlador:8000/v1 \
  --adminurl http://controlador:8000/v1 \
  --region regionOne
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
|   adminurl  |    http://controller:8000/v1     |
|      id     | f41225f665694b95a46448e8676b0dc2 |
| internalurl |    http://controller:8000/v1     |
|  publicurl  |    http://controller:8000/v1     |
|    region   |            regionOne             |
|  service_id | 297740d74c0a446bbff867acdccb33fa |
+-------------+----------------------------------+
```

Para instalar e configurar os componentes da Orquestração

1. Execute os seguintes comandos para instalar os pacotes:

```
# apt-get install heat-api heat-api-cfn heat-engine python-heatclient
```

2. Edite o arquivo `/etc/heat/heat.conf` e complete as seguintes ações:

   a. Na seção `[database]`, configure o acesso ao banco de dados:

```
[database]
...
connection = mysql://heat:HEAT_DBPASS@controlador/heat
```

   Substitua *HEAT_DBPASS* com a senha que você escolheu para a base de dados da Orquestração.

   b. Na seção `[DEFAULT]`, configure o acesso ao intermediador RabbitMQ:

```
[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controlador
rabbit_password = RABBIT_PASS
```

   Substitua *RABBIT_PASS* com a senha que você escolheu para a conta `guest` no RabbitMQ.

   c. Nas seções `[keystone_authtoken]` e `[ec2authtoken]`, configure o acesso ao serviço de Identidade:

```
[keystone_authtoken]
...
auth_uri = http://controlador:5000/v2.0
identity_uri = http://controlador:35357
admin_tenant_name = service
admin_user = heat
admin_password = HEAT_PASS

[ec2authtoken]
...
auth_uri = http://controlador:5000/v2.0
```

Substitua *HEAT_PASS* com a senha que você escolheu para o usuário `heat` no serviço de Identidade.

Nota

Comente quaisquer opções `auth_host`, `auth_port`, e `auth_protocol`, porque a opção `identity_uri` as substitui.

d. Na seção `[DEFAULT]`, configure o metadado e as URLs de condição de espera:

```
[DEFAULT]
...
heat_metadata_server_url = http://controlador:8000
heat_waitcondition_server_url = http://controlador:8000/v1/
waitcondition
```

e. (Opcional) Para auxiliar com a solução de problemas, habilite o log detalhado na seção `[DEFAULT]` :

```
[DEFAULT]
...
verbose = True
```

3. Popule a base de dados de Orquestração:

```
# su -s /bin/sh -c "heat-manage db_sync" heat
```

Para finalizar a instalação

1. Reinicie os serviços de Orquestração:

```
# service heat-api restart
# service heat-api-cfn restart
# service heat-engine restart
```

2. Por padrão, os pacotes do Ubuntu criam um banco de dados SQLite.

   Devido esta configuração utilizar um servidor de banco de dados SQL, você pode remover o arquivo de banco de dados SQLite:

```
# rm -f /var/lib/heat/heat.sqlite
```

Verifique a operação

Esta seção descreve como verificar a operação do módulo de Orquestração (heat).

1. Obtenha as credenciais do tenant `demo` :

```
$ source demo-openrc.sh
```

2. The Orchestration module uses templates to describe stacks. To learn about the template language, see the Template Guide in the Heat developer documentation.

   Crie um modelo de teste no arquivo `test-stack.yml` com o seguinte conteúdo:

```
heat_template_version: 2013-05-23

description: Test Template

parameters:
  ImageID:
    type: string
    description: Image use to boot a server
  NetID:
    type: string
    description: Network ID for the server

resources:
  server1:
    type: OS::Nova::Server
    properties:
      name: "Test server"
      image: { get_param: ImageID }
      flavor: "m1.tiny"
      networks:
      - network: { get_param: NetID }

outputs:
  server1_private_ip:
    description: IP address of the server in the private network
    value: { get_attr: [ server1, first_address ] }
```

3. Utilize o comando **heat stack-create** para criar a pilha através do modelo:

```
$ NET_ID=$(nova net-list | awk '/ demo-net / { print $2 }')
$ heat stack-create -f test-stack.yml \
  -P "ImageID=cirros-0.3.3-x86_64;NetID=$NET_ID" testStack
+--------------------------------------+------------+--------------------
+----------------------+
| id                                   | stack_name | stack_status       |
 creation_time        |
+--------------------------------------+------------+--------------------
+----------------------+
| 477d96b4-d547-4069-938d-32ee990834af | testStack  | CREATE_IN_PROGRESS |
 2014-04-06T15:11:01Z |
+--------------------------------------+------------+--------------------
+----------------------+
```

4. Utilize o comando **heat stack-list** para verificar a criação com sucesso da pilha:

```
$ heat stack-list
+--------------------------------------+------------+-----------------
+----------------------+
| id                                   | stack_name | stack_status    |
 creation_time        |
```

```
+--------------------------------------+------------+-----------------
+----------------------+
| 477d96b4-d547-4069-938d-32ee990834af | testStack  | CREATE_COMPLETE |
 2014-04-06T15:11:01Z |
+--------------------------------------+------------+-----------------
+----------------------+
```

Próximos passos

Seu ambiente OpenStack agora inclui Orquestração. Você pode lançar uma instância ou adicionar mais serviços ao seu ambiente nos capítulos seguintes.

Capítulo 11. Adicionar o módulo de Telemetria

Índice

A Telemetria fornece uma estrutura para monitoramento e medição da nuvem OpenStack. Ela é conhecida como projeto ceilometer.

Telemetry module

The Telemetry module performs the following functions:

- Efficiently collects the metering data about the CPU and network costs.
- Collects data by monitoring notifications sent from services or by polling the infrastructure.
- Configures the type of collected data to meet various operating requirements. It accesses and inserts the metering data through the REST API.
- Expands the framework to collect custom usage data by additional plug-ins.
- Produces signed metering messages that cannot be repudiated.

The Telemetry module consists of the following components:

A compute agent (`ceilometer-agent-compute`) — Runs on each compute node and polls for resource utilization statistics. There may be other types of agents in the future, but for now our focus is creating the compute agent.

A central agent (`ceilometer-agent-central`) — Runs on a central management server to poll for resource utilization statistics for resources not tied to instances or compute nodes.

A notification agent (`ceilometer-agent-notification`) — Runs on a central management server to initiate alarm actions, such as calling out to a webhook with a description of the alarm state transition.

A collector (`ceilometer-collector`) — Runs on central management server(s) to monitor the message queues (for notifications and for metering da-

	ta coming from the agent). Notification messages are processed and turned into metering messages, which are sent to the message bus using the appropriate topic. Telemetry messages are written to the data store without modification.
An alarm evaluator (`ceilometer-alarm-evaluator`)	Runs on one or more central management servers to determine when alarms fire due to the associated statistic trend crossing a threshold over a sliding time window.
An alarm notifier (`ceilometer-alarm-notifier`)	Runs on one or more central management servers to allow alarms to be set based on the threshold evaluation for a collection of samples.
A data store	A database capable of handling concurrent writes (from one or more collector instances) and reads (from the API server).
An API server (`ceilometer-api`)	Runs on one or more central management servers to provide data access from the data store.

These services communicate by using the OpenStack messaging bus. Only the collector and API server have access to the data store.

Instalar e configurar o nodo controlador

This section describes how to install and configure the Telemetry module, code-named ceilometer, on the controller node. The Telemetry module uses separate agents to collect measurements from each OpenStack service in your environment.

Para configurar pre-requisitos

Before you install and configure Telemetry, you must install MongoDB, create a MongoDB database, and create Identity service credentials including endpoints.

1. Instale o pacote MongoDB:

```
# apt-get install mongodb-server
```

2. Edite o arquivo `/etc/mongodb.conf` e complete as seguintes ações:

   a. Configure a chave `bind_ip` para utilizar o endereço IP da interface de gerenciamento do nodo controlador.

   ```
   bind_ip = 10.0.0.11
   ```

   b. By default, MongoDB creates several 1GB journal files in the /var/lib/mongodb/journal directory. If you want to reduce the size of each journal file to 128MB and limit total journal space consumption to 512MB, assert the smallfiles key:

   ```
   smallfiles = true
   ```

   If you change the journaling configuration, stop the MongoDB service, remove the initial journal files, and start the service:

```
# service mongodb stop
# rm /var/lib/mongodb/journal/prealloc.*
# service mongodb start
```

You can also disable journaling. For more information, see the MongoDB manual.

c. Reinicie o serviço do MongoDB:

```
# service mongodb restart
```

3. Crie a base de dados do `ceilometer`:

```
# mongo --host controlador --eval '
  db = db.getSiblingDB("ceilometer");
  db.addUser({user: "ceilometer",
  pwd: "CEILOMETER_DBPASS",
  roles: [ "readWrite", "dbAdmin" ]})'
```

Substitua *CEILOMETER_DBPASS* com uma senha adequada.

4. Execute um source nas credenciais de `admin` para obter acesso aos comandos CLI admin-only.

```
$ source admin-openrc.sh
```

5. Para criar as credenciais do Serviço de Identidade:

a. Crie o usuário `ceilometer`:

```
$ keystone user-create --name ceilometer --pass CEILOMETER_PASS
```

Substitua *CEILOMETER_PASS* com uma senha adequada.

b. Ligue o usuário `ceilometer` ao `serviço` de tenant e ao papel `admin`:

```
$ keystone user-role-add --user ceilometer --tenant service --role
 admin
```

c. Crie o serviço `ceilometer`:

```
$ keystone service-create --name ceilometer --type metering \
  --description "Telemetry"
```

d. Crie os endpoints do Serviço de Identidade:

```
$ keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ metering / {print
$2}') \
  --publicurl http://controlador:8777 \
  --internalurl http://controlador:8777 \
  --adminurl http://controlador:8777 \
  --region regionOne
```

Para instalar e configurar os componentes do módulo de Telemetria

1. Instale os pacotes:

```
# apt-get install ceilometer-api ceilometer-collector ceilometer-agent-
central \
```

```
  ceilometer-agent-notification ceilometer-alarm-evaluator ceilometer-
alarm-notifier \
  python-ceilometerclient
```

2. Gere valores randômicos para utilizar como secret de medição:

```
# openssl rand -hex 10
```

3. Edite o arquivo `/etc/ceilometer/ceilometer.conf` e complete as seguintes ações:

   a. Na seção `[database]`, configure o acesso ao banco de dados:

```
[database]
...
connection = mongodb://ceilometer:CEILOMETER_DBPASS@controlador:27017/
ceilometer
```

   Substitua *CEILOMETER_DBPASS* com a senha que você escolheu para a base de dados do módulo de Telemetria.

   b. Na seção `[DEFAULT]`, configure o acesso ao intermediador RabbitMQ:

```
[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controlador
rabbit_password = RABBIT_PASS
```

   Substitua *RABBIT_PASS* com a senha que você escolheu para a conta `guest` no RabbitMQ.

   c. Nas seções `[DEFAULT]` e `[keystone_authtoken]`, configure o acesso do Serviço de Identidade:

```
[DEFAULT]
...
auth_strategy = keystone

[keystone_authtoken]
...
auth_uri = http://controlador:5000/v2.0
identity_uri = http://controlador:35357
admin_tenant_name = service
admin_user = ceilometer
admin_password = CEILOMETER_PASS
```

   Substitua *CEILOMETER_PASS* com a senha que você escolheu para o usuário `ceilometer` no Serviço de Identidade.

   Nota

   Comente quaisquer opções `auth_host`, `auth_port`, e `auth_protocol`, porque a opção `identity_uri` as substitui.

   d. Na seção `[service_credentials]`, configure as credenciais de serviço:

```
[service_credentials]
...
os_auth_url = http://controlador:5000/v2.0
os_username = ceilometer
os_tenant_name = service
os_password = CEILOMETER_PASS
```

Substitua *CEILOMETER_PASS* com a senha que você escolheu para o usuário `ceilometer` no Serviço de Identidade.

e. Na seção `[publisher]`, configure o secret de medição:

```
[publisher]
...
metering_secret = METERING_SECRET
```

Substitua *METERING_SECRET* com o valor randômico que você gerou no passo anterior.

f. Na seção `[DEFAULT]` , configure o diretório de log:

```
[DEFAULT]
...
log_dir = /var/log/ceilometer
```

Para finalizar a instalação

- Reinicie os serviços de Telemetria:

```
# service ceilometer-agent-central restart
# service ceilometer-agent-notification restart
# service ceilometer-api restart
# service ceilometer-collector restart
# service ceilometer-alarm-evaluator restart
# service ceilometer-alarm-notifier restart
```

Instale o agente de Computação para Telemetria

A Telemetria é composta de um serviço de API, um coletor e uma variedade de agentes diferentes. Esta seção explica como instalar e configurar o agente que roda no nodo de Computação.

Para configurar pre-requisitos

1. Instalar o pacote:

```
# apt-get install ceilometer-agent-compute
```

2. Edite o arquivo `/etc/nova/nova.conf` e adicione as seguintes linhas à seção `[DEFAULT]`:

```
[DEFAULT]
...
instance_usage_audit = True
instance_usage_audit_period = hour
notify_on_state_change = vm_and_task_state
notification_driver = nova.openstack.common.notifier.rpc_notifier
notification_driver = ceilometer.compute.nova_notifier
```

3. Reinicie o serviço de Computação:

```
# service nova-compute restart
```

Para configurar o agente de Computação para Telemetria

Edite o arquivo `/etc/ceilometer/ceilometer.conf` e complete as seguintes ações:

1. Na seção `[publisher]`, defina a chave secreta para os nodos do serviço de Telemetria.

```
[publisher]
# Secret value for signing metering messages (string value)
metering_secret = CEILOMETER_TOKEN
```

Substitua *`CEILOMETER_TOKEN`* com o token do ceilometer que você criou anteriormente.

2. Na seção `[DEFAULT]`, configure o acesso ao intermediador RabbitMQ.

```
[DEFAULT]
rabbit_host = controlador
rabbit_password = RABBIT_PASS
```

Substitua *`RABBIT_PASS`* com a senha que você escolheu para a conta guest no RabbitMQ.

3. Na seção `[keystone_authtoken]` , configure o acesso ao serviço de Identidade:

```
[keystone_authtoken]
auth_uri = http://controlador:5000/v2.0
identity_uri = http://controlador:35357
admin_tenant_name = service
admin_user = ceilometer
admin_password = CEILOMETER_PASS
```

Substitua *`CEILOMETER_PASS`* com a senha que você escolheu para a base de dados do módulo de Telemetria.

Nota

Comente as chaves `auth_host`, `auth_port`, e `auth_protocol`, uma vez que elas foram substituídas pelas chaves `identity_uri` e `auth_uri`.

4. Na seção `[service_credentials]`, configure as credenciais de serviço:

```
[service_credentials]
os_auth_url = http://controlador:5000/v2.0
os_username = ceilometer
os_tenant_name = service
os_password = CEILOMETER_PASS
os_endpoint_type = internalURL
```

Substitua *CEILOMETER_PASS* com a senha que você escolheu para o usuário ceilometer no Serviço de Identidade.

5. Na seção `[DEFAULT]` , configure o diretório de log:

```
[DEFAULT]
log_dir = /var/log/ceilometer
```

Para concluir a instalação

- Reinicie o serviço com as novas configurações:

```
# service ceilometer-agent-compute restart
```

Configure o Serviço de Imagem para Telemetria

1. Para obter amostras de imagens, você deve configurar o serviço de Imagem para enviar notificações ao barramento.

   Edite o arquivo `/etc/glance/glance-api.conf` e modifique a seção `[DEFAULT]`

```
notification_driver = messaging
rpc_backend = rabbit
rabbit_host = controlador
rabbit_password = RABBIT_PASS
```

2. Reinicie os Serviços de Imagem com suas novas configurações:

```
# service glance-registry restart
# service glance-api restart
```

Adicione o agente do serviço de Block Storage para Telemetria

1. Para recuperar exemplos de volume, você deve configurar o serviço de Block Storage para enviar notificações para o baramento.

   Edite `/etc/cinder/cinder.conf` e adicione na seção `[DEFAULT]` nos nodos controlador e volume:

```
control_exchange = cinder
notification_driver = cinder.openstack.common.notifier.rpc_notifier
```

2. Reinicie os serviços de Block Storage com suas novas configurações.

   No nodo do controlador:

```
# service cinder-api restart
# service cinder-scheduler restart
```

No nodo de armazenamento:

```
# service cinder-volume restart
```

3. If you want to collect OpenStack Block Storage notification on demand, you can use **cinder-volume-usage-audit** from OpenStack Block Storage. For more information, *Block Storage audit script setup to get notifications*.

Configure o servico de Object Storage para Telemetria.

1. Instale o pacote python-ceilometerclient em seu seu servidor proxy de Object Storage:

```
# apt-get install python-ceilometerclient
```

2. Para obter estatísticas de armazenamento de objeto, o serviço de Telemetria precisa acessar o Object Storage com o papel `ResellerAdmin`. Atribua este papel ao seu usuário `os_username` de tenant.

```
$ keystone role-create --name ResellerAdmin
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|    id    | 462fa46c13fd4798a95a3bfbe27b5e54 |
|   name   |          ResellerAdmin           |
+----------+----------------------------------+
```

```
$ keystone user-role-add --tenant service --user ceilometer \
      --role 462fa46c13fd4798a95a3bfbe27b5e54
```

3. Você deve adicionar o middleware de Telemetria ao Object Storage para lidar com o tráfego de entrada e saída. Adicione estas linhas ao arquivo `/etc/swift/proxy-server.conf`:

```
[filter:ceilometer]
use = egg:ceilometer#swift
```

4. Adicione `ceilometer` ao parâmetro `pipeline` desse mesmo arquivo:

```
[pipeline:main]
pipeline = healthcheck cache authtoken keystoneauth ceilometer proxy-
server
```

5. Adicione o usuário do sistema `swift` ao grupo de sistema `ceilometer` para dar acesso para o Object Storage ao arquivo `ceilometer.conf`.

```
# usermod -a -G ceilometer swift
```

6. Adicione `ResellerAdmin` ao parâmetro `operator_roles` do mesmo arquivo:

```
operator_roles = Member,admin,swiftoperator,_member_,ResellerAdmin
```

7. Reinicie o serviço com as novas configurações:

```
# service swift-proxy restart
```

Verifique a instalação da Telemetria

Para testar a instalação da Telemetria, baixe uma imagem do Serviço de Imagem, e utilize o comando **ceilometer** para mostrar estatísticas de uso.

1. Utilize o comando `ceilometer meter-list` para testar o acesso à Telemetria:

```
$ ceilometer meter-list
```

```
+------------+-------+-------+--------------------------------------
+---------+----------------------------------+
| Name       | Type  | Unit  | Resource ID                          | User
 ID | Project ID                       |
+------------+-------+-------+--------------------------------------
+---------+----------------------------------+
| image      | gauge | image | acafc7c0-40aa-4026-9673-b879898e1fc2 | None
    | efa984b0a914450e9a47788ad330699d |
| image.size | gauge | B     | acafc7c0-40aa-4026-9673-b879898e1fc2 | None
    | efa984b0a914450e9a47788ad330699d |
+------------+-------+-------+--------------------------------------
+---------+----------------------------------+
```

2. Baixe uma imagem a partir do serviço de Imagem:

```
$ glance image-download "cirros-0.3.3-x86_64" > cirros.img
```

3. Acione o comando `ceilometer meter-list` novamente para validar que o download foi detectado e armazenado pela Telemetria:

```
$ ceilometer meter-list
```

```
+----------------+-------+-------+--------------------------------------
+---------+----------------------------------+
| Name           | Type  | Unit  | Resource ID                          |
 User ID | Project ID                       |
+----------------+-------+-------+--------------------------------------
+---------+----------------------------------+
| image          | gauge | image | acafc7c0-40aa-4026-9673-b879898e1fc2 |
 None    | efa984b0a914450e9a47788ad330699d |
| image.download | delta | B     | acafc7c0-40aa-4026-9673-b879898e1fc2 |
 None    | efa984b0a914450e9a47788ad330699d |
| image.serve    | delta | B     | acafc7c0-40aa-4026-9673-b879898e1fc2 |
 None    | efa984b0a914450e9a47788ad330699d |
| image.size     | gauge | B     | acafc7c0-40aa-4026-9673-b879898e1fc2 |
 None    | efa984b0a914450e9a47788ad330699d |
+----------------+-------+-------+--------------------------------------
+---------+----------------------------------+
```

4. Agora você pode obter estatísticas de uso para os vários medidores:

```
$ ceilometer statistics -m image.download -p 60
```

```
+--------+---------------------+---------------------+-------
+------------+------------+------------+------------+----------
+---------------------------+---------------------------+
```

```
| Period | Period Start        | Period End          | Count | Min
 | Max        | Sum        | Avg        | Duration | Duration Start
      | Duration End               |
+--------+---------------------+---------------------+-------
+------------+------------+------------+------------+----------
+----------------------------+----------------------------+
| 60     | 2013-11-18T18:08:50 | 2013-11-18T18:09:50 | 1     | 13167616.0
 | 13167616.0 | 13167616.0 | 13167616.0 | 0.0      | 2013-11-18T18:09:05.
334000 | 2013-11-18T18:09:05.334000 |
+--------+---------------------+---------------------+-------
+------------+------------+------------+------------+----------
+----------------------------+----------------------------+
```

Próximos passos

Seu ambiente OpenStack agora inclui Telemetria. Você pode lançar uma instância ou adicionar mais serviços ao seu ambiente nos capítulos anteriores.

Capítulo 12. Adicione o serviço de Banco de Dados

Índice

Utilize o *módulo de Banco de Dados* para criar recursos de bases de dados na nuvem. O nome do projeto integrado é *trove*.

Atenção

Este capítulo é um trabalho em progresso. Ele pode conter informações incorretas, e será atualizado frequentemente.

Database service overview

The Database service provides scalable and reliable cloud provisioning functionality for both relational and non-relational database engines. Users can quickly and easily use database features without the burden of handling complex administrative tasks. Cloud users and database administrators can provision and manage multiple database instances as needed.

The Database service provides resource isolation at high performance levels, and automates complex administrative tasks such as deployment, configuration, patching, backups, restores, and monitoring.

Process flow example. This example is a high-level process flow for using Database services:

1. The OpenStack Administrator configures the basic infrastructure using the following steps:
   a. Install the Database service.
   b. Create an image for each type of database. For example, one for MySQL and one for MongoDB.
   c. Use the **trove-manage** command to import images and offer them to tenants.
2. The OpenStack end user deploys the Database service using the following steps:
   a. Create a Database service instance using the **trove create** command.
   b. Use the **trove list** command to get the ID of the instance, followed by the **trove show** command to get the IP address of it.

c. Access the Database service instance using typical database access commands. For example, with MySQL:

```
$ mysql -u myuser -p -h TROVE_IP_ADDRESS mydb
```

The Database service includes the following components:

`python-troveclient` command-line client	A CLI that communicates with the `trove-api` component.
`trove-api` component	Provides an OpenStack-native RESTful API that supports JSON to provision and manage Trove instances.
`trove-conductor` service	Runs on the host, and receives messages from guest instances that want to update information on the host.
`trove-taskmanager` service	Instruments the complex system flows that support provisioning instances, managing the lifecycle of instances, and performing operations on instances.
`trove-guestagent` service	Runs within the guest instance. Manages and performs operations on the database itself.

Instalar o serviço de Banco de Dados

Este procedimento instala o módulo de Banco de Dados no nodo controlador.

Pré-requisitos. Este capítulo assume que você já tem um ambiente OpenStack funcionando com pelo menos os seguintes componentes instalados: Computação, Serviço de Imagem, Identidade.

- Se você quiser fazer backup e restauração, você também precisará do Object Storage.
- Se você quiser provisionar datastores nos volumes de block-storage, você também precisará do Block Storage.

Para instalar o módulo de Banco de Dados no controlador:

1. Instale os pacotes requeridos:

```
# apt-get install python-trove python-troveclient \
  trove-common trove-api trove-taskmanager trove-conductor
```

2. Prepare o OpenStack:

   a. Obtenha o arquivo `admin-openrc.sh`.

```
$ source ~/admin-openrc.sh
```

   b. Crie um usuário `trove` que a Computação usa para autenticar-se com o Serviço de Identidade. Utilize o tenant de `serviço` e dê ao usuário o papel `admin`:

```
$ keystone user-create --name trove --pass TROVE_PASS
$ keystone user-role-add --user trove --tenant service --role admin
```

   Substitua *TROVE_PASS* com uma senha adequada.

3. Todos os arquivos de configuração devem sem colocados no diretório `/etc/trove`. Edite os seguintes arquivos de configuração, tomando as ações abaixo para cada arquivo:

   - `api-paste.ini`
   - `trove.conf`
   - `trove-taskmanager.conf`
   - `trove-conductor.conf`

   a. Você precisa pegar `api-paste.ini` e alterar o conteúdo abaixo nele:

   ```
   [composite:trove]
   auth_uri = http://controlador:5000/v2.0
   identity_uri = http://controlador:35357
   auth_host = controlador
   admin_tenant_name = service
   admin_user = trove
   admin_password = TROVE_PASS
   ```

   b. Edite a seção `[DEFAULT]` de cada arquivo (exceto de `api-paste.ini`) e defina valores apropriados para as URLs de serviço do OpenStack (pode ser manipulado pelo catálogo de serviço Keystone), configurações de log e mensagens, e conexões SQL:

   ```
   [DEFAULT]
   log_dir = /var/log/trove
   trove_auth_url = http://controlador:5000/v2.0
   nova_compute_url = http://controlador:8774/v2
   cinder_url = http://controlador:8776/v1
   swift_url = http://controlador:8080/v1/AUTH_
   sql_connection = mysql://trove:TROVE_DBPASS@controlador/trove
   notifier_queue_hostname = controlador
   ```

   c. Configure o módulo de Banco de Dados para utilizar o intermediador de mensagens RabbitMQ definindo as seguintes opções na configuração de grupo `[DEFAULT]` de cada arquivo:

   ```
   [DEFAULT]
   control_exchange = trove
   rabbit_host = controlador
   rabbit_userid = guest
   rabbit_password = RABBIT_PASS
   rabbit_virtual_host= /
   rpc_backend = trove.openstack.common.rpc.impl_kombu
   ```

4. Edite o arquivo `trove.conf` para que ele inclua valores apropriados para o datastore e o rótulo de rede regex como mostrado abaixo:

   ```
   [DEFAULT]
   # Config option for showing the IP address that nova doles out
   add_addresses = True
   network_label_regex = ^NETWORK_LABEL$
   control_exchange = trove
   ```

5. Edite o arquivo `trove-taskmanager.conf` para que ele inclua as configurações requeridas para conectar ao serviço de Computação do OpenStack como mostrado abaixo:

```
[DEFAULT]
# Configuration options for talking to nova via the novaclient.
# These options are for an admin user in your keystone config.
# It proxy's the token received from the user to send to nova via this
 admin users creds,
# basically acting like the client via that proxy token.
nova_proxy_admin_user = admin
nova_proxy_admin_pass = ADMIN_PASS
nova_proxy_admin_tenant_name = service
taskmanager_manager = trove.taskmanager.manager.Manager
log_file=trove-taskmanager.log
```

6. Prepare o banco de dados de administração do trove:

```
$ mysql -u root -p
mysql> CREATE DATABASE trove;
mysql> GRANT ALL PRIVILEGES ON trove.* TO trove@'localhost' \
IDENTIFIED BY 'TROVE_DBPASS';
mysql> GRANT ALL PRIVILEGES ON trove.* TO trove@'%' \
IDENTIFIED BY 'TROVE_DBPASS';
```

7. Prepare o serviço de banco de dados:

   a. Inicialize o banco de dados:

   ```
   # trove-manage db_sync
   ```

   b. Create a datastore. You need to create a separate datastore for each type of database you want to use, for example, MySQL, MongoDB, Cassandra. This example shows you how to create a datastore for a MySQL database:

   ```
   # su -s /bin/sh -c "trove-manage datastore_update mysql '" trove
   ```

8. Crie uma imagem do trove.

   Create an image for the type of database you want to use, for example, MySQL, MongoDB, Cassandra.

   Esta imagem deve ter o agente trove instalado, e deve ter o arquivo `trove-guestagent.conf` configurado para conectar-se ao seu ambiente OpenStack. Para configurar corretamente o arquivo `trove-guestagent.conf`, siga estes passos na instância guest que você está usando para contruir sua imagem:

   - Adicione as seguintes linha ao arquivo `trove-guestagent.conf`:

   ```
   rabbit_host = controlador
   rabbit_password = RABBIT_PASS
   nova_proxy_admin_user = admin
   nova_proxy_admin_pass = ADMIN_PASS
   nova_proxy_admin_tenant_name = service
   trove_auth_url = http://controlador:35357/v2.0
   log_file = trove-guestagent.log
   ```

9. Atualize o datastore e a versão para utilizar a imagem específica com o comando **trove-manage**.

```
#trove-manage datastore_update datastore_name datastore_version
      #trove-manage datastore_version_update datastore_name version_name \
      datastore_manager glance_image_id packages active
```

Este exemplo mostra a você como criar um datastore MySQL com versão 5.5:

```
#trove-manage datastore_update mysql ''
      #trove-manage datastore_version_update mysql 5.5
 mysql glance_image_ID mysql-server-5.5 1
      #trove-manage datastore_update mysql 5.5
```

Carregue as regras de validação de configuração de pós-provisionamento:

```
#trove-manage
 db_load_datastore_config_parameters datastore_name version_name \
          /etc/datastore_name/validation-rules.json
```

Exemplo para carregar regras para o datastore MySQL:

```
# trove-manage db_load_datastore_config_parameters \
          mysql 5.5 "$PYBASEDIR"/trove/templates/mysql/validation-rules.
json
```

10. Você deve registrar o módulo de Banco de Dados com o Serviço de Identidade para que outros serviços OpenStack possam localizá-lo. Registre o serviço e especifique o endpoint:

```
$ keystone service-create --name trove --type database \
  --description "OpenStack Database Service"
$ keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ trove / {print $2}') \
  --publicurl http://controlador:8779/v1.0/%\(tenant_id\)s \
  --internalurl http://controlador:8779/v1.0/%\(tenant_id\)s \
  --adminurl http://controlador:8779/v1.0/%\(tenant_id\)s \
  --region regionOne
```

11. Reinicie os serviços de Banco de Dados:

```
# service trove-api restart
# service trove-taskmanager restart
# service trove-conductor restart
```

Verifique a instalação do serviço de Banco de Dados

To verify that the Database service is installed and configured correctly, try executing a Trove command:

1. Source the `demo-openrc.sh` file.

```
$ source ~/demo-openrc.sh
```

2. Obtenha a lista de instâncias Trove:

```
$ trove list
```

Você deve ver uma saída semelhante a esta:

```
+----+------+-----------+-------------------+--------+-----------+------+
| id | name | datastore | datastore_version | status | flavor_id | size |
+----+------+-----------+-------------------+--------+-----------+------+
+----+------+-----------+-------------------+--------+-----------+------+
```

3. Assuming you have created an image for the type of database you want, and have updated the datastore to use that image, you can now create a Trove instance (database). To do this, use the trove **create** command.

   Este exemplo mostra a você como criar uma base de dados MySQL 5.5:

```
$ trove create nome 2 --size=2 --databases DBNAME \
  --users USER:PASSWORD --datastore_version mysql-5.5 \
  --datastore mysql
```

Capítulo 13. Adicionar o serviço de Processamento de Dados

Índice

O serviço de Processamento de Dados (sahara) habilita os usuários a fornecer uma pilha escalável de processamento de dados e interfaces de gerenciamento associadas. Isto inclui o provisionamento de clusters de processamento de dados, bem como agendamento e operação de tarefas de processamento de dados.

Atenção

Este capítulo é um trabalho em progresso. Ele pode conter informações incorretas, e será atualizado frequentemente.

Data processing service

The Data processing service for OpenStack (sahara) aims to provide users with simple means to provision data processing (Hadoop, Spark) clusters by specifying several parameters like Hadoop version, cluster topology, nodes hardware details and a few more. After user fills in all the parameters, the Data processing service deploys the cluster in a few minutes. Also sahara provides means to scale already provisioned clusters by adding/removing worker nodes on demand.

The solution addresses the following use cases:

- Fast provisioning of Hadoop clusters on OpenStack for development and QA.
- Utilization of unused compute power from general purpose OpenStack IaaS cloud.
- Analytics-as-a-Service for ad-hoc or bursty analytic workloads.

Key features are:

- Designed as an OpenStack component.
- Managed through REST API with UI available as part of OpenStack dashboard.
- Support for different Hadoop distributions:
  - Pluggable system of Hadoop installation engines.
  - Integration with vendor specific management tools, such as Apache Ambari or Cloudera Management Console.
- Predefined templates of Hadoop configurations with ability to modify parameters.

- User-friendly UI for ad-hoc analytics queries based on Hive or Pig.

Instale o serviço de processamento de Dados.

Este procedimento instala o serviço de processamento de Dados (sahara) no nodo controlador.

Para instalar o serviço de processamento de dados no controlador:

1. 

   Atenção

   Você precisa instalar os pacotes requeridos. No momento, o sahara não tem pacotes para Ubuntu. A Documentação será atualizada tão logo os pacotes estejam disponíveis. O restante deste documento assume que você tem os pacotes do serviço sahara instalados no sistema.

2. Edite o arquivo de configuração `/etc/sahara/sahara.conf`

   a. Primeiro, edite o parâmetro `connection` na seção `[database]`. A URL fornecida aqui deve apontar para uma base de dados vazia. Por exemplo, a string de conexão para a base de dados MySQL será:

   ```
   connection = mysql://sahara:SAHARA_DBPASS@controlador/sahara
   ```

   b. Alterne para a seção `[keystone_authtoken]`. O parâmetro `auth_uri` deve apontar para o endpoint público da API de Identidade. `identity_uri` deve apontar para o endpoint da API de Identidade de admin. Por exemplo:

   ```
   auth_uri = http://controlador:5000/v2.0
   identity_uri = http://controlador:35357
   ```

   c. A seguir, especifique `admin_user`, `admin_password` e `admin_tenant_name`. Esses parâmetros devem especificar um usuário de keystone que tem o papel de `admin` no tenant fornecido. Essas credenciais permitem o sahara autenticar e autorizar seus usuários.

   d. Alterne para a seção `[DEFAULT]`. Vá até os parâmetros de rede. Se você estiver usando Neutron para rede, então defina `use_neutron=true`. Do contrário, se você estiver usando `nova-network`, defina o parâmetro dado para `false`.

   e. Isso deveria ser suficiente para a primeira execução. Se você quiser aumentar o nível de log para solução de problemas, existem dois parâmetros na configuração: `verbose` e `debug`. Se o primeiro está definido para `true`, o sahara irá iniciar a escrever logs de nível `INFO` e acima. Se `debug` está definido para `true`, o sahara irá escrever todos os logs, incluindo aqueles de `DEBUG`.

3. Se você utiliza o serviço de processamento de Dados com o banco de dados MySQL, então para armazenar grandes tarefas binárias na base de dados interna do sahara você deve configurar o tamanho máximos de pacotes permitido. Edite o arquivo `my.cnf` e altere o parâmetro:

   ```
   [mysqld]
   max_allowed_packet = 256M
   ```

e reinicie o servidor MySQL.

4. Crie o esquema da base de dados:

```
# sahara-db-manage --config-file /etc/sahara/sahara.conf upgrade head
```

5. Você deve registrar o serviço de processamento de Dados com o serviço de Identidade para que outros serviços OpenStack possam localizá-lo. Registre o serviço e especifique o endpoint:

```
$ keystone service-create --name sahara --type data_processing \
  --description "Data processing service"
$ keystone endpoint-create \
  --service-id $(keystone service-list | awk '/ sahara / {print $2}') \
  --publicurl http://controlador:8386/v1.1/%\(tenant_id\)s \
  --internalurl http://controlador:8386/v1.1/%\(tenant_id\)s \
  --adminurl http://controlador:8386/v1.1/%\(tenant_id\)s \
  --region regionOne
```

6. Inicie o serviço sahara:

```
# systemctl start openstack-sahara-all
```

7. (Opcional) Habilite o serviço de processamento de Dados para iniciar no boot

```
# systemctl enable openstack-sahara-all
```

Verifique a instalação do serviço de processamento de Dados

Para verificar que o serviço de Processamento de Dados (sahara) está instalado e configurado corretamente, tente requisitar uma lista de clusters usando o cliente sahara.

1. Obtenha as credenciais do tenant `demo` :

```
$ source demo-openrc.sh
```

2. Obter a lista de clusters sahara:

```
$ sahara cluster-list
```

Você deve ver uma saída semelhante a esta:

```
+------+----+--------+------------+
| name | id | status | node_count |
+------+----+--------+------------+
+------+----+--------+------------+
```

Capítulo 14. Lançando uma instância

Índice

An instance is a VM that OpenStack provisions on a compute node. This guide shows you how to launch a minimal instance using the *CirrOS* image that you added to your environment in the Capítulo 4, Adicionar o Serviço de Imagem [42] chapter. In these steps, you use the command-line interface (CLI) on your controller node or any system with the appropriate OpenStack client libraries. To use the dashboard, see the *OpenStack User Guide*.

Launch an instance using OpenStack Networking (neutron) or legacy networking (nova-network) . For more information, see the *OpenStack User Guide*.

Nota

Esses passos referenciam componentes de exemplo criados no capitulo anterior. Você deve ajustar certos valores, tais como endereço IP para corresponder ao seu ambiente.

Lance uma instância com a Rede OpenStack (neutron)

Para gerar um par de chaves

A maioria da imagens de nuvem suportam *autenticação de chave pública* em vez de autenticação por usuário/senha. Antes de lançar uma instância, você deve gerar um par de chaves pública/privada utilizando **ssh-keygen** e adicionar a chave pública no seu ambiente OpenStack.

1. Obtenha as credenciais do tenant `demo` :

```
$ source demo-openrc.sh
```

2. Gere um par de chaves:

```
$ ssh-keygen
```

3. Adicione a chave pública ao seu ambiente OpenStack:

```
$ nova keypair-add --pub-key ~/.ssh/id_rsa.pub demo-key
```

Nota

Este comando não retorna resultados.

4. Verifique a adição da chave pública:

```
$ nova keypair-list
```

```
+----------+-------------------------------------------------+
| Name     | Fingerprint                                     |
+----------+-------------------------------------------------+
| demo-key | 6c:74:ec:3a:08:05:4e:9e:21:22:a6:dd:b2:62:b8:28 |
+----------+-------------------------------------------------+
```

Para lançar uma instância

Para lançar uma instância, você deve ao menos especificar o sabor, nome da imagem, grupo de segurança, chave, e o nome da instância.

1. Um sabor especifica um perfil de alocação de recurso virtual que inclui processador, memória, e armazenamento.

   Listar sabores disponíveis:

```
$ nova flavor-list
+----+-----------+-----------+------+-----------+------+-------
+-------------+-----------+
| ID | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs |
 RXTX_Factor | Is_Public |
+----+-----------+-----------+------+-----------+------+-------
+-------------+-----------+
| 1  | m1.tiny   | 512       | 1    | 0         |      | 1     | 1.0
   | True      |
| 2  | m1.small  | 2048      | 20   | 0         |      | 1     | 1.0
   | True      |
| 3  | m1.medium | 4096      | 40   | 0         |      | 2     | 1.0
   | True      |
| 4  | m1.large  | 8192      | 80   | 0         |      | 4     | 1.0
   | True      |
| 5  | m1.xlarge | 16384     | 160  | 0         |      | 8     | 1.0
   | True      |
+----+-----------+-----------+------+-----------+------+-------
+-------------+-----------+
```

   Sua primeira instância utiliza o sabor `m1.tiny`.

   Nota

   Você também pode referenciar um sabor pelo ID.

2. Listar imagens disponíveis:

```
$ nova image-list
+--------------------------------------+---------------------+--------
+--------+
| ID                                   | Name                | Status |
 Server |
+--------------------------------------+---------------------+--------
+--------+
| acafc7c0-40aa-4026-9673-b879898e1fc2 | cirros-0.3.3-x86_64 | ACTIVE |
     |
+--------------------------------------+---------------------+--------
+--------+
```

   Sua primeira instância utiliza a imagem `cirros-0.3.3-x86_64`.

3. Listar redes disponíveis:

```
$ neutron net-list
+--------------------------------------+----------
+------------------------------------------------------+
| id                                   | name     | subnets
                                |
+--------------------------------------+----------
+------------------------------------------------------+
| 3c612b5a-d1db-498a-babb-a4c50e344cb1 | demo-net | 20bcd3fd-5785-41fe-
ac42-55ff884e3180 192.168.1.0/24   |
| 9bce64a3-a963-4c05-bfcd-161f708042d1 | ext-net  | b54a8d85-b434-4e85-
a8aa-74873841a90d 203.0.113.0/24   |
+--------------------------------------+----------
+------------------------------------------------------+
```

Sua primeira instância utiliza a rede tenant `demo-net`. Contudo, você deve referenciar esta rede utilizando o ID em vez do nome.

4. Listar grupos de segurança disponíveis:

```
$ nova secgroup-list
+--------------------------------------+---------+-------------+
| Id                                   | Name    | Description |
+--------------------------------------+---------+-------------+
| ad8d4ea5-3cad-4f7d-b164-ada67ec59473 | default | default     |
+--------------------------------------+---------+-------------+
```

Your first instance uses the `default` security group. By default, this security group implements a firewall that blocks remote access to instances. If you would like to permit remote access to your instance, launch it and then configure remote access.

5. Lançar uma instância:

Substitua *`DEMO_NET_ID`* com o ID na rede tenant `demo-net`.

```
$ nova boot --flavor m1.tiny --image cirros-0.3.3-x86_64 --nic net-
id=DEMO_NET_ID \
  --security-group default --key-name demo-key demo-instance1
+--------------------------------------
+------------------------------------------------------------+
| Property                             | Value
                          |
+--------------------------------------
+------------------------------------------------------------+
| OS-DCF:diskConfig                    | MANUAL
                          |
| OS-EXT-AZ:availability_zone          | nova
                          |
| OS-EXT-STS:power_state               | 0
                          |
| OS-EXT-STS:task_state                | scheduling
                          |
| OS-EXT-STS:vm_state                  | building
                          |
| OS-SRV-USG:launched_at               | -
                          |
| OS-SRV-USG:terminated_at             | -
                          |
| accessIPv4                           |
                          |
```

```
| accessIPv6                           |
                          |
| adminPass                            | vFW7Bp8PQGNo
                          |
| config_drive                         |
                          |
| created                              | 2014-04-09T19:24:27Z
                          |
| flavor                               | m1.tiny (1)
                          |
| hostId                               |
                          |
| id                                   |
 05682b91-81a1-464c-8f40-8b3da7ee92c5                       |
| image                                | cirros-0.3.3-x86_64
 (acafc7c0-40aa-4026-9673-b879898e1fc2) |
| key_name                             | demo-key
                          |
| metadata                             | {}
                          |
| name                                 | demo-instance1
                          |
| os-extended-volumes:volumes_attached | []
                          |
| progress                             | 0
                          |
| security_groups                      | default
                          |
| status                               | BUILD
                          |
| tenant_id                            | 7cf50047f8df4824bc76c2fdf66d11ec
                          |
| updated                              | 2014-04-09T19:24:27Z
                          |
| user_id                              | 0e47686e72114d7182f7569d70c519c9
                          |
+--------------------------------------
+-------------------------------------------------------------+
```

6. Verifique o estado de sua instância:

```
$ nova list
+--------------------------------------+----------------+--------
+------------+-------------+-------------------------+
| ID                                   | Name           | Status | Task
 State | Power State | Networks                |
+--------------------------------------+----------------+--------
+------------+-------------+-------------------------+
| 05682b91-81a1-464c-8f40-8b3da7ee92c5 | demo-instance1 | ACTIVE | -
    | Running     | demo-net=192.168.1.3    |
+--------------------------------------+----------------+--------
+------------+-------------+-------------------------+
```

O estado muda de `BUILD` para `ACTIVE` quando sua instância termina o processo de construção.

Para acessar sua instância utilizando um console virtual

- Obtenha uma URL de sessão *Virtual Network Computing (VNC)* para sua instância e acesse-a a partir de um navegador web:

```
$ nova get-vnc-console demo-instance1 novnc
+-------
+-----------------------------------------------------------------------------------
+
| Type  | Url
                  |
+-------
+-----------------------------------------------------------------------------------
+
| novnc | http://controller:6080/vnc_auto.html?token=2f6dd985-f906-4bfc-
b566-e87ce656375b    |
+-------
+-----------------------------------------------------------------------------------
+
```

Nota

Se seu navegador web roda em um host que não pode resolver o nome de host do `controlador` , você pode substituir o `controlador` com o endereço IP da interface de gerenciamento no seu nodo controlador.

A imagem CirrOS inclui autenticação convencional de usuário/senha e fornece estas credenciais no prompt de login. Depois de fazer login na CirrOS, recomendamos que você verifique a conectividade de rede utilizando **ping**.

Verifique o gateway de rede do tenant `demo-net`:

```
$ ping -c 4 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_req=1 ttl=64 time=0.357 ms
64 bytes from 192.168.1.1: icmp_req=2 ttl=64 time=0.473 ms
64 bytes from 192.168.1.1: icmp_req=3 ttl=64 time=0.504 ms
64 bytes from 192.168.1.1: icmp_req=4 ttl=64 time=0.470 ms

--- 192.168.1.1 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
```

Verifique a rede externa `ext-net`:

```
$ 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
```

Para acessar sua instância remotamente

1. Adicione regras ao grupo de segurança `default`:

   a. Permita *ICMP* (ping):

```
$ nova secgroup-add-rule default icmp -1 -1 0.0.0.0/0
+-------------+-----------+---------+-----------+--------------+
| IP Protocol | From Port | To Port | IP Range  | Source Group |
+-------------+-----------+---------+-----------+--------------+
| icmp        | -1        | -1      | 0.0.0.0/0 |              |
+-------------+-----------+---------+-----------+--------------+
```

   b. Permita acesso ao shell seguro (SSH):

```
$ nova secgroup-add-rule default tcp 22 22 0.0.0.0/0
+-------------+-----------+---------+-----------+--------------+
| IP Protocol | From Port | To Port | IP Range  | Source Group |
+-------------+-----------+---------+-----------+--------------+
| tcp         | 22        | 22      | 0.0.0.0/0 |              |
+-------------+-----------+---------+-----------+--------------+
```

2. Crie um *endereço IP flutuante* na rede externa `ext-net`:

```
$ neutron floatingip-create ext-net
Created a new floatingip:
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| fixed_ip_address    |                                      |
| floating_ip_address | 203.0.113.102                        |
| floating_network_id | 9bce64a3-a963-4c05-bfcd-161f708042d1 |
| id                  | 05e36754-e7f3-46bb-9eaa-3521623b3722 |
| port_id             |                                      |
| router_id           |                                      |
| status              | DOWN                                 |
| tenant_id           | 7cf50047f8df4824bc76c2fdf66d11ec     |
+---------------------+--------------------------------------+
```

3. Associe o endereço IP flutuante com a sua instância:

```
$ nova floating-ip-associate demo-instance1 203.0.113.102
```

Nota

Este comando não retorna resultados.

4. Verifique o estado de seus endereços IP flutuantes:

```
$ nova list
+--------------------------------------+---------------+--------
+------------+-------------+-----------------------------------------+
| ID                                   | Name          | Status | Task
 State | Power State | Networks                                      |
+--------------------------------------+---------------+--------
+------------+-------------+-----------------------------------------+
| 05682b91-81a1-464c-8f40-8b3da7ee92c5 | demo-instance1 | ACTIVE | -
    | Running     | demo-net=192.168.1.3, 203.0.113.102     |
+--------------------------------------+---------------+--------
+------------+-------------+-----------------------------------------+
```

5. Verifique a conectividade de rede utilizando **ping** a partir do nodo controlador ou qualquer host na rede externa:

```
$ ping -c 4 203.0.113.102
PING 203.0.113.102 (203.0.113.112) 56(84) bytes of data.
64 bytes from 203.0.113.102: icmp_req=1 ttl=63 time=3.18 ms
64 bytes from 203.0.113.102: icmp_req=2 ttl=63 time=0.981 ms
64 bytes from 203.0.113.102: icmp_req=3 ttl=63 time=1.06 ms
64 bytes from 203.0.113.102: icmp_req=4 ttl=63 time=0.929 ms

--- 203.0.113.102 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
```

6. Acesse sua instância utilizando SSH a partir do nodo controlador ou qualquer host na rede externa:

```
$ ssh cirros@203.0.113.102
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.
$
```

Nota

Se seu host não contém o par de chave pública/privada criado no passo anterior, o SSH irá solicitar a senha padrão associada com o usuário `cirros`.

Para conectar um volume do Block Storage à sua instância

Se seu ambiente inclui o serviço Block Storage, você pode conectar um volume à instância.

1. Obtenha as credenciais do tenant `demo` :

```
$ source demo-openrc.sh
```

2. Listar volumes:

```
$ nova volume-list
+--------------------------------------+-----------+--------------+------
+-------------+-------------+
| ID                                   | Status    | Display Name | Size |
 Volume Type | Attached to |
+--------------------------------------+-----------+--------------+------
+-------------+-------------+
| 158bea89-07db-4ac2-8115-66c0d6a4bb48 | available | demo-volume1 | 1    |
 None        |             |
+--------------------------------------+-----------+--------------+------
+-------------+-------------+
```

3. Conectar o volume `demo-volume1` à instância `demo-instance1`:

```
$ nova volume-attach demo-instance1 158bea89-07db-4ac2-8115-66c0d6a4bb48
+----------+--------------------------------------+
| Property | Value                                |
```

```
+----------+--------------------------------------+
| device   | /dev/vdb                             |
| id       | 158bea89-07db-4ac2-8115-66c0d6a4bb48 |
| serverId | 05682b91-81a1-464c-8f40-8b3da7ee92c5 |
| volumeId | 158bea89-07db-4ac2-8115-66c0d6a4bb48 |
+----------+--------------------------------------+
```

Nota

Você deve referenciar volumes utilizando os IDs em vez de nomes.

4. Listar volumes:

```
$ nova volume-list
+--------------------------------------+-----------+--------------+------
+-------------+--------------------------------------+
| ID                                   | Status    | Display Name | Size |
 Volume Type | Attached to                          |
+--------------------------------------+-----------+--------------+------
+-------------+--------------------------------------+
| 158bea89-07db-4ac2-8115-66c0d6a4bb48 | in-use    | demo-volume1 | 1    |
 None        | 05682b91-81a1-464c-8f40-8b3da7ee92c5 |
+--------------------------------------+-----------+--------------+------
+-------------+--------------------------------------+
```

O estado do volume `demo-volume1` deve indicar `in-use` pelo ID da instância `demo-instance1`.

5. Acesse sua instância utilizando SSH a partir do nodo controlador ou qualquer host na rede externa e utilize o comando **fdisk** para verificar a presença do volume como um dispositivo de armazenamento de blocos `/dev/vdb`:

```
$ ssh cirros@203.0.113.102
$ 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
```

Nota

Você deve criar uma tabela de partição e um sistema de arquivos para utilizar o volume.

If your instance does not launch or seem to work as you expect, see the *OpenStack Operations Guide* for more information or use one of the many other options to seek assistance. We want your environment to work!

Lance uma instância com rede legada (nova-network)

Para gerar um par de chaves

A maioria da imagens de nuvem suportam *autenticação de chave pública* em vez de autenticação por usuário/senha. Antes de lançar uma instância, você deve gerar um par de chaves pública/privada utilizando **ssh-keygen** e adicionar a chave pública no seu ambiente OpenStack.

1. Obtenha as credenciais do tenant `demo` :

```
$ source demo-openrc.sh
```

2. Gere um par de chaves:

```
$ ssh-keygen
```

3. Adicione a chave pública ao seu ambiente OpenStack:

```
$ nova keypair-add --pub-key ~/.ssh/id_rsa.pub demo-key
```

Nota

Este comando não retorna resultados.

4. Verifique a adição da chave pública:

```
$ nova keypair-list
+----------+-------------------------------------------------+
| Name     | Fingerprint                                     |
+----------+-------------------------------------------------+
| demo-key | 6c:74:ec:3a:08:05:4e:9e:21:22:a6:dd:b2:62:b8:28 |
+----------+-------------------------------------------------+
```

Para lançar uma instância

Para lançar uma instância, você deve ao menos especificar o sabor, nome da imagem, grupo de segurança, chave, e o nome da instância.

1. Um sabor especifica um perfil de alocação de recurso virtual que inclui processador, memória, e armazenamento.

   Listar sabores disponíveis:

```
$ nova flavor-list
+----+-----------+-----------+------+-----------+------+-------
+-------------+-----------+
| ID | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs |
 RXTX_Factor | Is_Public |
+----+-----------+-----------+------+-----------+------+-------
+-------------+-----------+
```

```
| 1  | m1.tiny   | 512       | 1    | 0         |      | 1     | 1.0
   | True      |
| 2  | m1.small  | 2048      | 20   | 0         |      | 1     | 1.0
   | True      |
| 3  | m1.medium | 4096      | 40   | 0         |      | 2     | 1.0
   | True      |
| 4  | m1.large  | 8192      | 80   | 0         |      | 4     | 1.0
   | True      |
| 5  | m1.xlarge | 16384     | 160  | 0         |      | 8     | 1.0
   | True      |
+----+-----------+-----------+------+-----------+------+-------
+-------------+-----------+
```

Sua primeira instância utiliza o sabor `m1.tiny`.

Nota

Você também pode referenciar um sabor pelo ID.

2. Listar imagens disponíveis:

```
$ nova image-list
+--------------------------------------+--------------------+--------
+--------+
| ID                                   | Name               | Status |
 Server |
+--------------------------------------+--------------------+--------
+--------+
| acafc7c0-40aa-4026-9673-b879898e1fc2 | cirros-0.3.3-x86_64 | ACTIVE |
     |
+--------------------------------------+--------------------+--------
+--------+
```

Sua primeira instância utiliza a imagem `cirros-0.3.3-x86_64`.

3. Listar redes disponíveis:

Nota

Você deve obter as credenciais do tenant `admin` para este passo e então, obter as credenciais do tenant `demo` para os passos restantes.

```
$ source admin-openrc.sh
```

```
$ nova net-list
+--------------------------------------+----------+------------------+
| ID                                   | Label    | CIDR             |
+--------------------------------------+----------+------------------+
| 7f849be3-4494-495a-95a1-0f99ccb884c4 | demo-net | 203.0.113.24/29  |
+--------------------------------------+----------+------------------+
```

Sua primeira instância utiliza a rede tenant `demo-net`. Contudo, você deve referenciar esta rede utilizando o ID em vez do nome.

4. Listar grupos de segurança disponíveis:

```
$ nova secgroup-list
+--------------------------------------+---------+-------------+
```

```
| Id                                   | Name    | Description |
+--------------------------------------+---------+-------------+
| ad8d4ea5-3cad-4f7d-b164-ada67ec59473 | default | default     |
+--------------------------------------+---------+-------------+
```

Sua primeira instância utiliza o grupo de segurança `default`. Por padrão, este grupo de segurança implementa um firewall que bloqueia o acesso remoto às instâncias. Se você gostaria de permitir acesso remoto à sua instância, lance-a e então configure o acesso remoto.

5. Lançar uma instância:

   Substitua *`DEMO_NET_ID`* com o ID na rede tenant `demo-net`.

```
$ nova boot --flavor m1.tiny --image cirros-0.3.3-x86_64 --nic net-
id=DEMO_NET_ID \
  --security-group default --key-name demo-key demo-instance1
+--------------------------------------
+------------------------------------------------------------+
| Property                             | Value
                          |
+--------------------------------------
+------------------------------------------------------------+
| OS-DCF:diskConfig                    | MANUAL
                          |
| OS-EXT-AZ:availability_zone          | nova
                          |
| OS-EXT-STS:power_state               | 0
                          |
| OS-EXT-STS:task_state                | scheduling
                          |
| OS-EXT-STS:vm_state                  | building
                          |
| OS-SRV-USG:launched_at               | -
                          |
| OS-SRV-USG:terminated_at             | -
                          |
| accessIPv4                           |
                          |
| accessIPv6                           |
                          |
| adminPass                            | ThZqrg7ach78
                          |
| config_drive                         |
                          |
| created                              | 2014-04-10T00:09:16Z
                          |
| flavor                               | m1.tiny (1)
                          |
| hostId                               |
                          |
| id                                   | 45ea195c-
c469-43eb-83db-1a663bbad2fc                          |
| image                                | cirros-0.3.3-x86_64
 (acafc7c0-40aa-4026-9673-b879898e1fc2) |
| key_name                             | demo-key
                          |
| metadata                             | {}
                          |
```

```
| name                                 | demo-instance1
                          |
| os-extended-volumes:volumes_attached | []
                          |
| progress                             | 0
                          |
| security_groups                      | default
                          |
| status                               | BUILD
                          |
| tenant_id                            | 93849608fe3d462ca9fa0e5dbfd4d040
                          |
| updated                              | 2014-04-10T00:09:16Z
                          |
| user_id                              | 8397567baf4746cca7a1e608677c3b23
                          |
+--------------------------------------
+------------------------------------------------------------+
```

6. Verifique o estado de sua instância:

```
$ nova list
+--------------------------------------+----------------+--------
+------------+-------------+------------------------+
| ID                                   | Name           | Status | Task
 State | Power State | Networks               |
+--------------------------------------+----------------+--------
+------------+-------------+------------------------+
| 45ea195c-c469-43eb-83db-1a663bbad2fc | demo-instance1 | ACTIVE | -
    | Running     | demo-net=203.0.113.26  |
+--------------------------------------+----------------+--------
+------------+-------------+------------------------+
```

O estado muda de `BUILD` para `ACTIVE` quando sua instância termina o processo de construção.

Para acessar sua instância utilizando um console virtual

- Obtenha uma URL de sessão *Virtual Network Computing (VNC)* para sua instância e acesse-a a partir de um navegador web:

```
$ nova get-vnc-console demo-instance1 novnc
+-------
+-------------------------------------------------------------------------------------
+
| Type  | Url
                  |
+-------
+-------------------------------------------------------------------------------------
+
| novnc | http://controller:6080/vnc_auto.html?token=2f6dd985-f906-4bfc-
b566-e87ce656375b    |
+-------
+-------------------------------------------------------------------------------------
+
```

Nota

Se seu navegador web roda em um host que não pode resolver o nome de host do *controlador* , você pode substituir o *controlador* com o endereço IP da interface de gerenciamento no seu nodo controlador.

A imagem CirrOS inclui autenticação convencional de usuário/senha e fornece estas credenciais no prompt de login. Depois de fazer login na CirrOS, recomendamos que você verifique a conectividade de rede utilizando **ping**.

Verifique a rede `demo-net`:

```
$ 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
```

Para acessar sua instância remotamente

1. Adicione regras ao grupo de segurança `default`:

   a. Permita *ICMP* (ping):

   ```
   $ nova secgroup-add-rule default icmp -1 -1 0.0.0.0/0
   +-------------+-----------+---------+-----------+--------------+
   | IP Protocol | From Port | To Port | IP Range  | Source Group |
   +-------------+-----------+---------+-----------+--------------+
   | icmp        | -1        | -1      | 0.0.0.0/0 |              |
   +-------------+-----------+---------+-----------+--------------+
   ```

   b. Permita acesso ao shell seguro (SSH):

   ```
   $ nova secgroup-add-rule default tcp 22 22 0.0.0.0/0
   +-------------+-----------+---------+-----------+--------------+
   | IP Protocol | From Port | To Port | IP Range  | Source Group |
   +-------------+-----------+---------+-----------+--------------+
   | tcp         | 22        | 22      | 0.0.0.0/0 |              |
   +-------------+-----------+---------+-----------+--------------+
   ```

2. Verifique a conectividade de rede utilizando **ping** a partir do nodo controlador ou qualquer host na rede externa:

   ```
   $ ping -c 4 203.0.113.26
   PING 203.0.113.26 (203.0.113.26) 56(84) bytes of data.
   64 bytes from 203.0.113.26: icmp_req=1 ttl=63 time=3.18 ms
   64 bytes from 203.0.113.26: icmp_req=2 ttl=63 time=0.981 ms
   64 bytes from 203.0.113.26: icmp_req=3 ttl=63 time=1.06 ms
   64 bytes from 203.0.113.26: icmp_req=4 ttl=63 time=0.929 ms

   --- 203.0.113.26 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
```

3. Acesse sua instância utilizando SSH a partir do nodo controlador ou qualquer host na rede externa:

```
$ ssh cirros@203.0.113.26
The authenticity of host '203.0.113.26 (203.0.113.26)' 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 '203.0.113.26' (RSA) to the list of known
 hosts.
$
```

Nota

Se seu host não contém o par de chave pública/privada criado no passo anterior, o SSH irá solicitar a senha padrão associada com o usuário `cirros`.

Para conectar um volume do Block Storage à sua instância

Se seu ambiente inclui o serviço Block Storage, você pode conectar um volume à instância.

1. Obtenha as credenciais do tenant `demo` :

```
$ source demo-openrc.sh
```

2. Listar volumes:

```
$ nova volume-list
+--------------------------------------+-----------+--------------+------
+-------------+-------------+
| ID                                   | Status    | Display Name | Size |
 Volume Type | Attached to |
+--------------------------------------+-----------+--------------+------
+-------------+-------------+
| 158bea89-07db-4ac2-8115-66c0d6a4bb48 | available | demo-volume1 | 1    |
 None        |             |
+--------------------------------------+-----------+--------------+------
+-------------+-------------+
```

3. Conectar o volume `demo-volume1` à instância `demo-instance1`:

```
$ nova volume-attach demo-instance1 158bea89-07db-4ac2-8115-66c0d6a4bb48
+----------+--------------------------------------+
| Property | Value                                |
+----------+--------------------------------------+
| device   | /dev/vdb                             |
| id       | 158bea89-07db-4ac2-8115-66c0d6a4bb48 |
| serverId | 45ea195c-c469-43eb-83db-1a663bbad2fc |
| volumeId | 158bea89-07db-4ac2-8115-66c0d6a4bb48 |
+----------+--------------------------------------+
```

Nota

Você deve referenciar volumes utilizando os IDs em vez de nomes.

4. Listar volumes:

```
$ nova volume-list
+--------------------------------------+-----------+--------------+------
+-------------+--------------------------------------+
| ID                                   | Status    | Display Name | Size |
 Volume Type | Attached to                          |
+--------------------------------------+-----------+--------------+------
+-------------+--------------------------------------+
| 158bea89-07db-4ac2-8115-66c0d6a4bb48 | in-use    | demo-volume1 | 1    |
 None        | 45ea195c-c469-43eb-83db-1a663bbad2fc |
+--------------------------------------+-----------+--------------+------
+-------------+--------------------------------------+
```

O estado do volume `demo-volume1` deve indicar `in-use` pelo ID da instância `demo-instance1`.

5. Acesse sua instância utilizando SSH a partir do nodo controlador ou qualquer host na rede externa e utilize o comando **fdisk** para verificar a presença do volume como um dispositivo de armazenamento de blocos `/dev/vdb`:

```
$ ssh cirros@203.0.113.102
$ 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
```

Nota

Você deve criar uma tabela de partição e um sistema de arquivos para utilizar o volume.

If your instance does not launch or seem to work as you expect, see the *OpenStack Operations Guide* for more information or use one of the many other options to seek assistance. We want your environment to work!

Apêndice A. IDs de usuário reservados

OpenStack reserves certain user IDs to run specific services and own specific files. These user IDs are set up according to the distribution packages. The following table gives an overview.

Nota

Alguns pacotes do OpenStack geram e atribuem IDs de usuário automaticamente durante a instalação de pacotes. Nesses casos, o valor do ID de usuário não é importante. O que importa é a existência do ID.

Tabela A.1. IDs de usuário reservados

Nome	Descrição	ID
ceilometer	Daemons de Ceilometer do openStack	Atribuído durante a instalação de pacotes.
cinder	Daemons de Cinder do OpenStack	Atribuído durante a instalação de pacotes.
glance	Daemons de Glance do OpenStack	Atribuído durante a instalação de pacotes.
heat	Daemons de Heat do OpenStack	Atribuído durante a instalação de pacotes.
keystone	Daemons de Keystone do OpenStack	Atribuído durante a instalação de pacotes.
neutron	Daemons de Neutron do OpenStack	Atribuído durante a instalação de pacotes.
nova	Daemons de Nova do OpenStack	Atribuído durante a instalação de pacotes.
swift	Daemons de Swift do OpenStack	Atribuído durante a instalação de pacotes.
trove	OpenStack Trove Daemons	Atribuído durante a instalação de pacotes.

Cada usuário pertence a um grupo de usuários com o mesmo nome do usuário.

Apêndice B. Community support

Índice

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.

Documentation

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

To provide feedback on documentation, join and use the <openstack-docs@lists.openstack.org> mailing list at OpenStack Documentation Mailing List, or report a bug.

The following books explain how to install an OpenStack cloud and its associated components:

- *Installation Guide for Debian 7*
- *Installation Guide for openSUSE 13.1 and SUSE Linux Enterprise Server 11 SP3*
- *Installation Guide for Red Hat Enterprise Linux 7, CentOS 7, and Fedora 20*
- *Installation Guide for Ubuntu 14.04*

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

- *Architecture Design Guide*
- *Cloud Administrator Guide*
- *Configuration Reference*
- *Operations Guide*
- *High Availability Guide*
- *Security Guide*

- *Virtual Machine Image Guide*

The following books explain how to use the OpenStack dashboard and command-line clients:

- *API Quick Start*
- *End User Guide*
- *Admin User Guide*
- *Command-Line Interface Reference*

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

- OpenStack API Complete Reference (HTML)
- API Complete Reference (PDF)
- *OpenStack Block Storage Service API v2 Reference*
- *OpenStack Compute API v2 and Extensions Reference*
- *OpenStack Identity Service API v2.0 Reference*
- *OpenStack Image Service API v2 Reference*
- *OpenStack Networking API v2.0 Reference*
- *OpenStack Object Storage API v1 Reference*

The *Training Guides* offer software training for cloud administration and management.

ask.openstack.org

During the set up or testing of OpenStack, you might have questions about how a specific task is completed or be in a situation where a feature does not work correctly. Use the ask.openstack.org site to ask questions and get answers. When you visit the http://ask.openstack.org site, scan the recently asked questions to see whether your question has already been answered. If not, ask a new question. Be sure to give a clear, concise summary in the title and 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.

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 http://lists.openstack.org/cgi-bin/mailman/listinfo/openstack. You might be interested in the other mailing lists for specific projects or development, which you can find on the wiki. A description of all mailing lists is available at http://wiki.openstack.org/MailingLists.

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 nova, 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.

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 at https://launchpad.net/+login. 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, `"Juno 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: Bare Metal (ironic)
- Bugs: Data Processing Service (sahara)
- Bugs: Database Service (trove)

- Bugs: Orchestration (heat)
- Bugs: Telemetry (ceilometer)
- Bugs: Queue Service (marconi)
- Bugs: OpenStack API Documentation (developer.openstack.org)
- Bugs: OpenStack Documentation (docs.openstack.org)

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 http://webchat.freenode.net/. You can also use Colloquy (Mac OS X, http://colloquy.info/), mIRC (Windows, http://www.mirc.com/), 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 http://paste.openstack.org. 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 at https://wiki.openstack.org/wiki/IRC.

Documentation feedback

To provide feedback on documentation, join and use the `<openstack-docs@lists.openstack.org>` mailing list at OpenStack Documentation Mailing List, or report a bug.

OpenStack distribution packages

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

- **Debian:** http://wiki.debian.org/OpenStack
- **CentOS, Fedora, and Red Hat Enterprise Linux:** http://openstack.redhat.com/
- **openSUSE and SUSE Linux Enterprise Server:** http://en.opensuse.org/Portal:OpenStack
- **Ubuntu:** https://wiki.ubuntu.com/ServerTeam/CloudArchive

glossário