Capítulo 1

Introducao ao Openstack

1.1 O que e Openstack

Em um primeiro momento poderíamos definir vagamente OpenStack como um software de cloud computing. Sendo um pouco mais específico: um Sistema Operacional projetado para nuvem capaz de controlar uma enorme quantidade de recursos computacionais (processamento, armazenamento e rede) em um Datacenter, ou seja, OpenStack é um projeto aberto para a criação de nuvens privadas e públicas, um sistema operacional para gestão e controle nuvem. Um cenário típico de um ambiente de cloud.

O termo projeto aberto neste contexto merece destaque, pois implica que o projeto nasceu com o objetivo de ser transparente e independente de fabricante. O site do projeto destaca os seguintes valores:**open source, open design, open development and an open community**.

Como "plataforma base" poderíamos empregar o software OpenStack, por exemplo, para construção de uma solução de Infraestutura como Serviço (Infrastructure as a Service – IaaS).

Basicamente uma plataforma de IaaS fornece meios para criação/provisionamento de máquinas virtuais (ou nós computacionais) sob demanda. Permitindo ainda que sua infraestrutura de servidores possa expandir ou encolher de forma elástica de acordo com a necessidade da sua aplicação.

1.1.1 Arquitetura

Como uma plataforma de computação em nuvem o OpenStack foi projetado para ser extremamente escalável e flexível. A plataforma é composta por vários "sub projetos" que juntos formam o seu núcleo (core). Sua arquitetura é modular formada por vários componentes que juntos implementam as funcionalidades três pilares que sustentam uma infraestrutura de nuvem: Processamento (compute), Rede (networking) e Armazenamento (storing).

Estruturas de processamento	
Nome	Funcao
nova	Gerencia o ciclo de vida das instâncias de computação em um ambiente OpenStack. As responsabilidades incluem a geração, programação desmantelamento de máquinas virtuais sob demanda.

Estruturas de networking	
Nome	Funcao
Neutron(nova-compute)	Permite Network-Conectividade-as-a-Service para outros serviços OpenStack, como OpenStack Compute. Fornece uma API para que os usuários definam as redes e os anexos neles. Tem uma arquitetura conectável que suporta muitos fornecedores de redes populares e tecnologias.

Estruturas de storage	
Nome	Funcao
Swift	Armazena e recupera objetos de dados não estruturados arbitrárias através de um HTTP baseado API RESTful. É altamente tolerante a falhas com a sua replicação de dados e arquitetura scale-out. A sua implementação não é como um servidor de arquivos com diretórios montáveis. Neste caso, ele grava objetos e arquivos para várias unidades, garantindo que os dados são replicados em um cluster de servidor
Cinder	Fornece armazenamento em bloco persistente para instâncias em execução. Sua arquitetura condutora conectável facilita a criação e gestão de dispositivos de armazenamento de bloco.

Além dos componentes que formam o core do OpenStack a plataforma conta com um conjunto de serviços que integra cada componente para fornecer uma plataforma de IaaS completa. Essa integração é possível porque cada componente disponibiliza um conjunto APIs que permite o acesso às suas funcionalidades. Os serviços são os seguintes:

Estruturas de processamento	
Serviço	Funcao
Keystone	Fornece um serviço de autenticação e autorização para outros serviços OpenStack. Fornece um catálogo de pontos de extremidade para todos os serviços OpenStack.
Glance	Armazena e recupera imagens de disco de máquina virtual. OpenStack Compute faz uso deste durante o exemplo de provisionamento.
Ceilometer	Monitora e faz medidas do OpenStack nuvem para o faturamento, o benchmarking, escalabilidade e fins estatísticos.
Heat	Orquestra múltiplas aplicações em nuvem compostas usando o formato de modelo HOT nativo ou o formato de modelo AWS CloudFormation, tanto através de uma API REST OpenStack-nativo e uma consulta API CloudFormation-compatível.
Trove	Fornece dados-as-a-Service funcionalidade escalável e confiável nuvem para ambos os motores de banco de dados relacionais e não-relacionais.

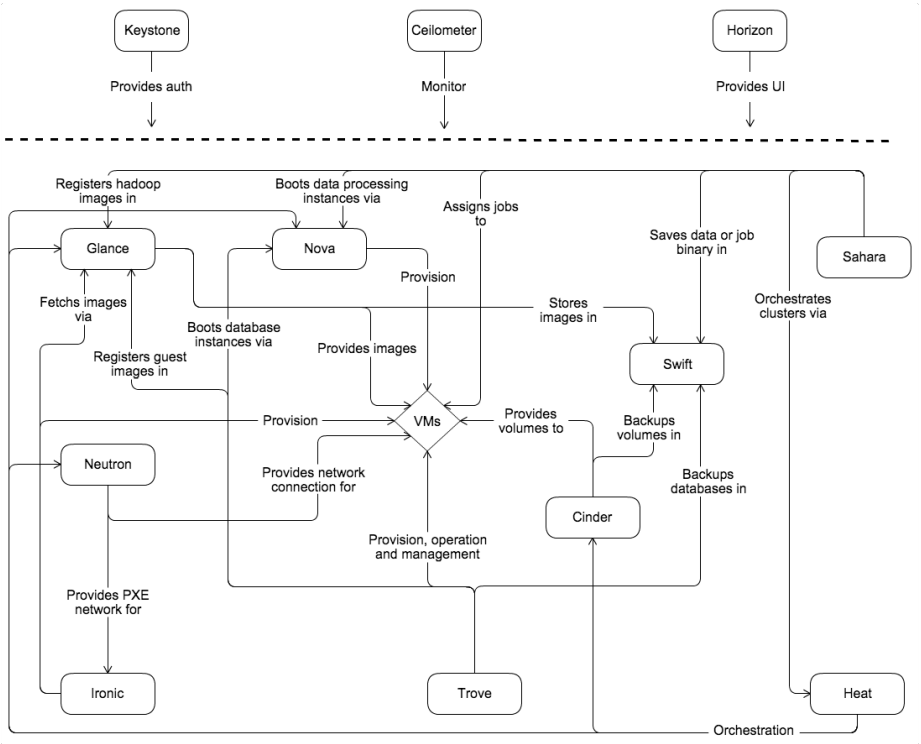


Figura 1.1: Arquitetura do OpenStack

1.1.2 Exemplo de Arquitetura

OpenStack é altamente configurável para atender às necessidades diferentes, com vários nós Compute, rede e opções de armazenamento. Neste tutorial será utilizado as seguintes configurações:

- O nó *Controller* executa o serviço de Identidade, serviço de imagem, parte do Compute gestão, e o painel de instrumentos. Ele também inclui suporte de serviços como um banco de dados SQL, fila de mensagens, e Network Time Protocol (NTP).

- O nó *Compute* executa parte da do Hypervisor do *Compute* que opera os "tenants"das máquinas virtuais, ou instancias. Por padrão, Compute usa KVM como o hypervisor.Nó *Compute*também dispõe "tenants"de rede fornece firewall (grupos de segurança) serviços.Pode-se executar mais de um nó.*Compute*

Opcionalmente, o nó Compute executa um agente de telemetria para recolher metros. Além disso, ele pode conter uma terceira interface de rede em uma rede de armazenamento separado para melhorar o desempenho dos serviços de armazenamento.

- O nó *Block Storage* opcional contém os discos que o serviço Block Storage provisões para instâncias de máquinas virtuais inquilino. Pode-se executar mais de um desses nós;

- O nó *Object Storage* contêm o discos que o serviço de armazenamento objeto usa para conta de armazenamento, recipiente e objeto. A arquitetura minima são dois nós.

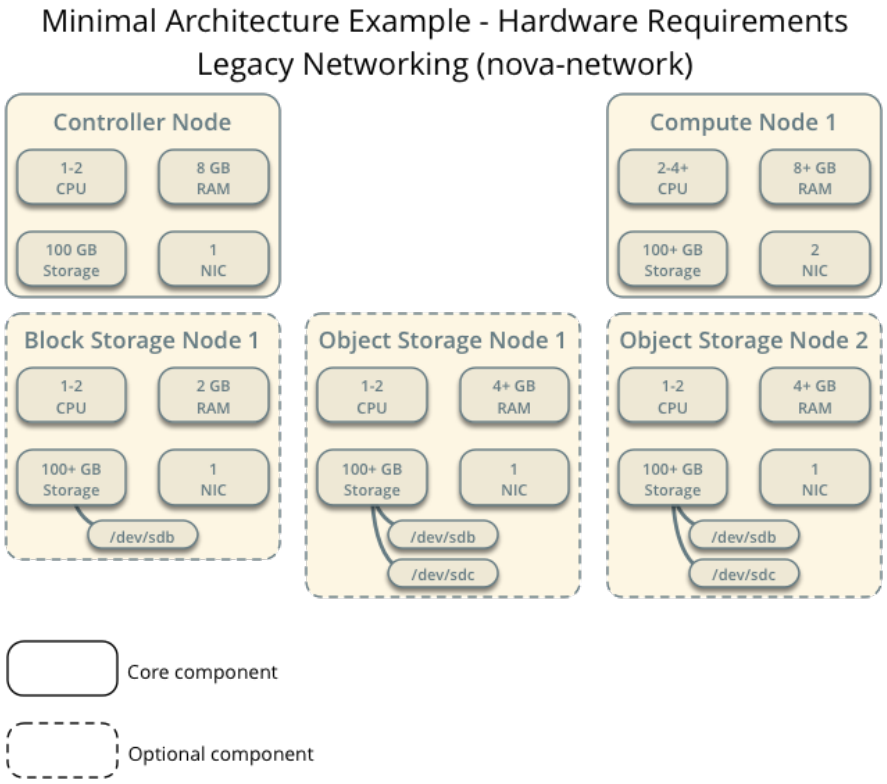


Figura 1.2: Arquitetura minima de hardware para suportar OpenStack

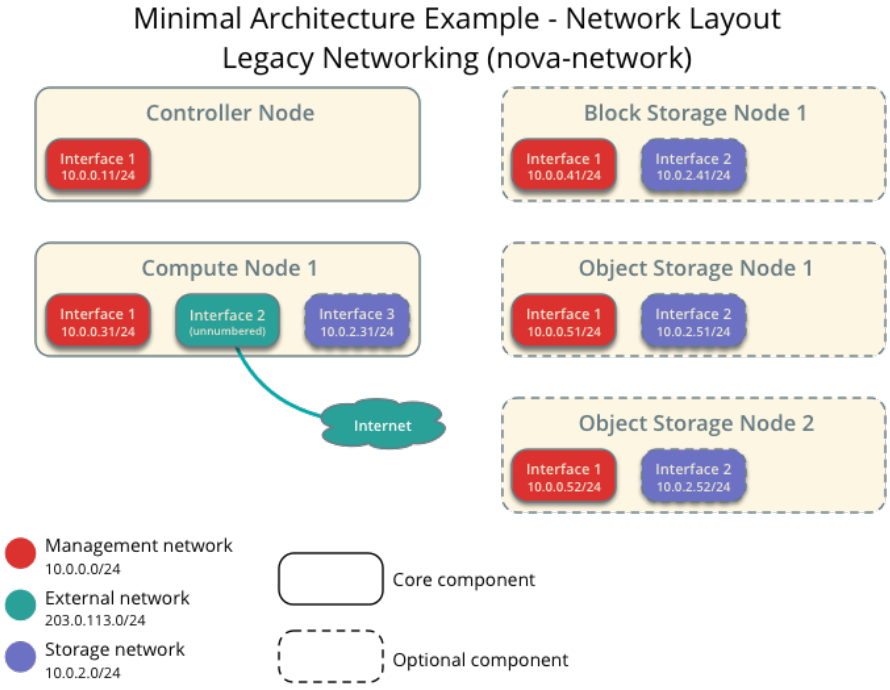


Figura 1.3: Arquitetura de rede para o OpenStack

Capítulo 2

Ambiente Basico

2.1 Passos iniciais de instalacao

Para melhor performance, é recomendado que o ambiente seja parecido ou exceda os requisitos de hardware na figura 1.2 do capítulo anterior. No entanto, OpenStack não requer uma quantidade significativa de recursos e os seguintes requisitos mínimos devem suportar um ambiente de "proof-of-concept"com os serviços centrais e várias instâncias CirrOS

- *nó Controller*:1 processador, 2GB de memória e 10GB de armazenamento
- *nó Compute*: 1 processador, 2GB de memória e 20GB de armazenamento

Para poupar espaço em disco e fornecer mais recursos para OpenStack, recomendamos uma instalação mínima de sua distribuição Linux. Além disso, é altamente recomendável que você instalar uma versão de sua distribuição de 64 bits em, pelo menos, o nó *Compute*. Se você instalar uma versão de sua distribuição no nó *Compute* de 32 bits, a tentativa de iniciar uma instância usando uma imagem de 64 bits falhará. Muitos usuários construir seus ambientes de teste em máquinas virtuais (VMs). Os principais benefícios de VMs incluem o seguinte:

- Um servidor físico pode suportar vários nós, cada um com praticamente qualquer número de interfaces de rede.
- Capacidade de tomar "snap shots"periódicas durante todo o processo de instalação e "voltar"para uma configuração de trabalho, no caso de um problema.

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

Para as senhas dos serviços do OpenStack, use $SERVICE\_PASS$ e para as senhas de banco de dados DB_PASS.

Nome da Senha	Descricao
ADMIN_PASS	Senha do usuario *admin*
CEILOMETER_DBPASS	
CEILOMETER_PASS	
CINDER_DBPASS	
CINDER_PASS	
DEMO_PASS	Senha do usuario *demo*
GLANCE_DBPASS	Senha do banco de dados para o servico de Imagem
GLANCE_PASS	Senha do usuario de servico de Imagem *glance*
KEYSTONE_DBPASS	Senha do banco de dados para o servico de autenticacao
NOVA_DBPASS	Senha do banco de dados para o servico *Compute*
NOVA_PASS	Senha do usuario de servico de *Compute*
RABBIT_PASS	Senha do usuario convidado do *RabbitMQ*

Neste guia está sendo utilizado "*openstack*"como senha para o banco de dados e para os serviços

OpenStack e serviços de suporte exigem privilégios administrativos durante a instalação e operação. Em alguns casos, os serviços de realizar modificações para o host que pode interferir com ferramentas de automação de implantação, tais como Ansible, Chef e Puppet. Por exemplo, alguns serviços OpenStack adicionar um envoltório de "root"para sudo que podem interferir com as políticas de segurança.

Depois de instalar o sistema operacional em cada nó para a arquitetura que optar por utilizar, é necessário configurar as interfaces de rede.É recomendado que desativar todas as ferramentas de gerenciamento de rede automatizados e editar manualmente os arquivos de configuração apropriados para distribuição.

2.2 Distribuicao da Rede no modo Legacy (nova-network)

O exemplo da arquitetura com a rede no modo Legacy requer um nó controlador e pelo menos um nó *Compute*.O nó controlador contém uma interface de rede na rede de gestão.O nó *Compute* contém uma interface de rede na rede de gestão e um na *rede externa*.

A arquitetura em exemplo assume o uso da seguintes redes:

- Gerenciador de rede:
- Rede externa:

2.2.1 Controller Node

Configuracao da Rede

- Endereco de Ip:10.129.64.51
- Mascara de Rede: 255.255.255.0 (ou /24)

- Gateway:

Reinicie o sistema para ativar as mudancas

Configurar nome da Rede

- Configurar o nome do computador (hostname) do nó para Controller
- Editar o arquivo */etc/hosts* com as seguintes configuracoes

```
#controller

10.129.64.51 controller

#compute1#

10.129.64.61 compute1
```

OBS: Algumas distribuições adicionam uma entrada estranha no arquivo /etc/hosts que resolve o nome da máquina para outro endereço IP de auto-retorno, como 127.0.1.1. Você deve comentar ou remover esta entrada para evitar problemas de resolução de nomes.

2.2.2 Nó Compute

Configuracao da Rede

Configurar a primeira interface como interface de gerenciamento

- Endereco de Ip:10.129.64.61
- Mascara de rede: 255.255.255.0 (ou /24)
- Gateway:

Configurar a segunda interface como externa utilizando uma configuracao especial sem um endereco de Ip assinalado.

Reescreva $NOME_INTERFACE$ com o nome da interface atual. Exemplo: eth1, ens224

- Edite o arquivo */etc/network/interfaces* com as seguintes configuracoes:

```
#Ainterfaceexternadarede
auto NOME_INTERFACE
iface NOME_INTERFACE inet manual
        up ip link set dev $IFACE up
        down ip link set dev $IFACE down
```

Reinicie o sistema para ativar as mudancas

Configurar nome da rede

- Coloque o nome do computador (hostname) do no como *compute1*
- Edite o arquivo */etc/hosts* com as seguintes configuracoes

```
#controller

10.129.64.51 controller

#compute1#

10.129.64.61 compute1
```

2.3 Pacotes OpenStack

Distribuições liberam pacotes OpenStack como parte da distribuição ou usando outros métodos devido às diferentes datas de lançamento. Realizar esses procedimentos em todos os nós

OBS: Desativar ou remover qualquer servico de atualizacao automatica, pois pode modificar o ambiente OpenStack

Permitir OpenStack Repositorio

- Instalar o arquivo "keyring"Ubuntu Cloud e repositorio

```
# apt-get install ubuntu-cloud-keyring
#echo "deb http://ubuntu-cloud.archive.canonical.com/ubuntutrusty-updates/kilo
main">/etc/apt/sources.list.d/cloudaarchive-kilo.list
```

Finalizar Instalacao

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

OBS: Se a atualizacao inclui um novo kernel, reiniciar o sistema para ativar

2.4 Banco de dados SQL

A maioria dos servicos OpenStack usa um banco de dados SQL para armazenar informacao. O banco de dados tipicamente executa no nó Controller. O procedimento nesta documentação usa o MariaDB ou o MySQL dependendo da distribuicao. Os servicos do OpenStack também suporta outros bancos de dados SQL.

Instalar e configurar o banco de dados do Servidor

1. Instalar pacotes

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

2. Escolhe uma senha para a conta de super-usuario

3. Criar e editar o arquivo */etc/mysql/conf.d/mysqld_openstack.cnf* com as seguintes configuracoes:

```
[mysqld]
...
bind-address = 10.129.64.51 default-storage-engine = innodb
innodb_file_per_table
collation-server = utf8_general_ci
init-connect = 'SET NAMES utf8'
character-set-server = utf8
```

Finalizar Instalação

1. Reiniciar o banco de dados

```
service mysql restart
```

2. Guardar o banco de dados

```
# mysql_secure_installation ...

Set root password? [Y/N] ...

Remove anonymous users? [Y/N] y
... Sucess!

...

Disallow root login remotely? [Y/N] y
... Sucess!

Remove test database and access to it? [Y/N] Y
- Dropping test database ...
... Sucess!
- Removing privileges on test database ...
... Success!

...

Reload privilege tables now? [Y/N] Y
... Sucess!
```

```
...

Thanks for using MariaDB!
```

2.5 Fila de Mensagens (Message QUEUE)

OpenStack usa o *message queue* para coordernar operações e status da informacao através dos serviços. O servico *message queue* tipicamente é executado no nó Controller. OpenStack suporta vários servicos de fila de mensagem, incluindp RabbitMQ,Qpid e ZeroMQ. Contudo, a maioria das distribuições que o pacote OpenStack suporta um particular servico de fila de mensagens. Esta documentação implementa o RabbitMQ servico de *message queue*, pois a maioria das distribuições o suportam.

2.5.1 Instalar o pacote de fila de mensagens

- Instalar o pacote

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

2.5.2 Configurar o serviço de fila de mensagens

1. Adicionar o usuário *openstack*

```
# rabbitmqctl add_user openstack RABBIT_PASS
Creating user "openstack" ...
... done.
```

2. Permitir configuração, escrita, e acesso a leitura for *user*

```
# rabbitmqctl set_permissions openstack ".* " ".* " ".*"
Setting permissions for user "openstack" vhost "/"
```

Capítulo 3

Serviço de Autenticação

3.1 Conceitos da Autenticação no Openstack

O serviço de autenticação do OpenStack (*Openstack Identity service*) tem as seguintes funções :

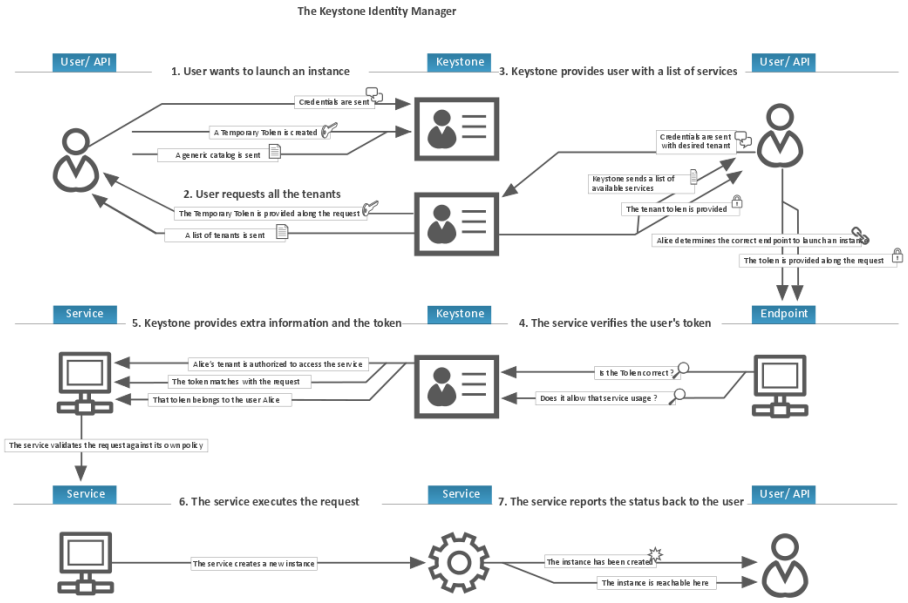
- Localizar usuarios e suas permissões

- Prover um catálogo dos serviços com seus terminais

Quando está instalando o serviço de autenticação do Openstack, é necessário registrar cada servico na instalação do Openstack. O serviço de autenticação pode localizar qual serviço do OpenStack estão instalados, e onde eles estão localizados na rede

Para entender o *OpenStack Identity*, deve-se entender os seguintes conceitos:

Usuario	Representação Digital de uma pessoa, sistema ou serviço de quem usa os serviços OpenStack cloud.Usuários possuem login e podem ser associado tokens para acesso de recursos
Credenciais	Dado que confirma a autenticação do usuário. Exemplo: nome de usuário e chave de API, ou token de autenticação provido pelo servico de autenticação
Autenticação	Processo que confirma a identidade de usuário. O *OpenStack Identity* confirma que uma requisição por validação de um conjunto de credenciais foi fornecido pelo usuário
Token	Uma string alfa-numérico usado para acessar as APIs do OpenStack e recursos. Um token pode ser cancelar em qualquer tempo e é valido por um tempo finito.Seu principal objetivo é ser um serviço de integração, e não aspirar a ser um armazenamento de identidade de pleno direito e solução de gerenciamento.
Tenant	Um container usado para agrupar ou isolar recursos. *Tenants* também agrupa ou isola objetos de autenticação. Dependendo do operador de serviço, um *tenant* pode mapear conta, organização ou projeto.
Serviço	Um serviço do OpenStack, assim como Compute(Nova) ou serviço de Imagem(Glance). Provê ou mais "*endpoints*" no qual usuários podem acessar recursos e executar operações.
Endpoint	Um endereço de rede acessível onde se pode acessar o servico, geralmente com endereço de URL.
Regra	Uma característica com um conjunto definido de direitos do usuário e privilégios para executar um conjunto específico de operações
Keystone Cliente	Uma interface linha de comando para a API de autenticação do OpenStack, exemplo: usuário executa os comandos **keystone service-create** e **keystone endpoint-create** para o registrar serviços na instalação do OpenStack



3.2 Instalando e Configurando

Todos os comandos de instalação do Openstack serão feitos no nó controlador. Por questão de performance, esta configuração implanta o *Apache HTTP server* para tratar requisições e *Memcached* para armazenar os tokens no lugar de um banco de dados SQL.

3.2.1 Configurando os requisitos

Antes de iniciar o serviço de autenticação, é necessário criar uma base de dados e um token de administração.

1. Para criar a base de dados, executar as seguintes instruções:

   - Acessar o banco de dados de acesso cliente para conectar ao servidor da base de dados como *root* usuário:

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

   - Criar a base de dados *keystone*

     ```
     CREATE DATABASE keystone;

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

     GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' IDEN-
     TIFIED BY 'KEYSTONE_DBPASS'
     ```

- Sair do MariaDB

3.2.2 Instalando os componentes de autenticação

1. Desativar o serviço *keystone* de iniciar automaticamente após a instalação

```
# echo "manual">/etc/init/keystone.override
```

2. Executar o seguinte comando para instalar os pacotes

```
# apt-get install keystone python-openstackclient apache2 libapache2-mod-wsgi memcached python-memcache
```

3. Editar o arquivo */etc/keystone/keystone.conf* e realize as seguintes modificações:

```
[DEFAULT]
...
admin token = ADMIN_TOKEN

...
verbose = true ...



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



[memcache]
...
servers = localhost:11211

...



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



[revoke]
...
driver = keystone.contrib.revoke.backends.sql.Revoke
```

4. Sincronizar a base de dados do serviço de autenticação

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

Configurar o servidor Apache HTTP

1. Editar o arquivo */etc/apache2/apach2.conf* com os seguintes dados

```
ServerName controller
```

2. Criar arquivo */etc/apache2/sites-available/wsgi-keystone.conf* com as seguintes configurações:

```
<VirtualHost *:5000 >
                WSGIDaemonProcess keystone-public processes=5 threads=1
user=keystone display-name=%{GROUP}
                WSGIProcessGroup keystone-public
                WSGIScriptAlias / /var/www/cgi-bin/keystone/main
                WSGIApplicationGroup %{GLOBAL}
                WSGIPassAuthorization On
                <IfVersion >= 2.4 >
                        ErrorLogFormat "%{cu}t %M"
                </IfVersion >                   LogLevel info
                ErrorLog /var/log/apache2/keystone-error.log
                CustomLog /var/log/apache2/keystone-access.log combined
<VirtualHost *:5000 >


<VirtualHost *:35357 >
                WSGIDaemonProcess keystone-public processes=5 threads=1
user=keystone display-name=%{GROUP}
                WSGIProcessGroup keystone-public
                WSGIScriptAlias / /var/www/cgi-bin/keystone/main
                WSGIApplicationGroup %{GLOBAL}
                WSGIPassAuthorization On
                <IfVersion >= 2.4 >
                        ErrorLogFormat "%{cu}t %M"
                </IfVersion >                   LogLevel info
                ErrorLog /var/log/apache2/keystone-error.log
                CustomLog /var/log/apache2/keystone-access.log combined
<VirtualHost *:35357 >
```

3. Permitir o *virtual host* do serviço de autenticação

```
# ln -s /etc/apache2/sites-available/wsgi-keystone.conf /etc/apache2/sites-
enabled
```

4. Criar a estrutura diretório para os componentes WSGI:

```
# mkdir -p /var/www/cgi-bin/keystone
```

5. Copiar os componentes WSGI do repositório no diretório

```
# curl http://git.openstack.org/cgit/openstack/keystone/plain/httpd/ keystone.py?h=stable/kilo \
— tee /var/www/cgi-bin/keystone/main /var/www/cgi-bin/keystone/admin
```

6. Ajustar ao proprietário permissoes neste diretório nos arquivos dentro dele:

```
# chown -R keystone:keystone /var/www/cgi-bin/keystone
# chmod 755 /var/www/cgi-bin/keystone/*
```

Finalizar instalação

1. Reiniciar o servidor Apache HTTP

```
# service apache2 restart
```

2. Por padrão, os pacotes Ubuntu cria um banco de dados SQLite. Por conta desta configuracao utilizar o servidor banco de dados SQL, remover a base de dados do arquivo SQLite:

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

3.3 Criar serviços de entrada e API endpoint

O serviço de autenticação fornece um catálogo de serviços e suas localizações. Cada serviço que adicionar ao ambiente OpenStack requer uma entidade de serviço e vários endpoint API no catálogo.

É necessário passar o valor de token autenticação para o comando **Openstack** com parametro *–os-token* ou cojunto de variável ambiente OS_TOKEN. Similarmente, é necessário passar o valor da URL do serviço de autenticação para o comando **openstack** com *–os-url* ou conjunto de variável ambiente OS_URL

1. Configurar o token de autenticação

```
$ export OS_TOKEN=ADMIN_TOKEN
```

2. Configurar o endpoint URL

```
$ export OS_URL=http://controller:35357/v2.0
```

3.3.1 Criar o entidade serviço e API endpoint

1. Criar a entidade de serviço para API endpoint

```
$ keystone tenant-create –name admin –description ”Admin Tenant”
```

2. Criar API serviço de autenticação endpoint

```
$ keystone endpoint-create
–service-id $(keystone service-list — awk '/ identity / print $2')
–publicurl http://controller:5000/v2.0
–internalurl http://controller:5000/v2.0
–adminurl http://controller:35357/v2.0
–region regionOne
```

3.4 Criar projetos,usuários e regras

1. Criar *projeto* admin

```
$ keystone tenant-create –name admin –description ”Admin Tenant”
```

2. Criar usuário *admin*

```
$ keystone user-create –name admin –pass $USER_ADMIN_PASSWORD
–email $EMAIL_ADMIN
```

3. Criar regra *admin*

```
$ keystone role-create –name admin
```

4. Adicionar a regra *admin* para o projeto e usuário *admin*

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

Tarefas regulares (nao-administrador) deve usar projeto e usuários nao privilegiados.

1. Criar *demo* Projeto

```
$ tenant-create –name demo –description ”Demo Tenant”
```

2. Criar usuário *demo*

```
$ keystone user-create –name demo –tenant demo –pass USER_DEMO_PASSWORD
–email EMAIL_DEMO
```

3. Criar regra *usuário*

```
$ keystone user-role-add –user=demo –role=_member_ –tenant=demo
```

3.5 Verificar Operação

1. Por razões de segurança, desativar temporariamente mecanismo de autenticação de token editando o arquivo */etc/keystone/keystone-paste.ini* e remover a linha *admin_token_auth* nas seções *[pipeline:public_api]*, *[pipeline:admin_api]* e *[pipeline:api_v3]*.

2. Retirar temporariamente as variaveis de ambiente *OS_TOKEN* e *OS_URL*

```
$ unset OS_TOKEN OS_URL
```

Como usuário administrador, requisitar um token de autenticação da API 2.0 de autenticação

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

Como *administrador*, listar projetos, usuarios, regras que verificam que o *administrador* pode executar apenas comandos de administrador e esse serviço de autenticacao contem os projetos criados nas duas seções anteriores

```
$ keystone –os-tenant-name admin –os-username admin –os-password
$USER_ADMIN_PASSWORD
–os-auth-url ”http://controller:35357/v2.0”user-list
```

```
$ keystone –os-tenant-name admin –os-username admin –os-password
$USER_ADMIN_PASSWORD –os-auth-url ”http://controller:35357/v2.0”role-
list
```

Como *demo* usuário, requisitar um token de identificação da API 2.0 de autenticação e uma lista de usuários

```
$ keystone –os-tenant-name demo –os-username demo –os-password
$USER_DEMO_PASSWORD –os-auth-url ”http://controller:35357/v2.0”token-
get
```

```
$ keystone –os-tenant-name demo –os-username demo –os-password
$USER_DEMO_PASSWORD –os-auth-url ”http://controller:35357/v2.0”user-
list
```

3.6 Criar scripts de ambiente para cliente OpenStack

A seção anterior usou uma combinação de variáveis de ambiente e opções de comandos para interagir com o serviço de Identidade através do cliente OpenStack. Para aumentar a eficiência das operações do cliente, OpenStack suporta scripts simples ambiente cliente também conhecidos como arquivos OpenRC.

1. Crie e edite o arquivo *admin-openrc.sh* com os seguintes dados

```
export OS_TENANT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=ADMIN_PASS
export OS_AUTH_URL=http://controller:35357/v3
```

2. Crie e edite o arquivo *demo-openrc.sh* com os seguintes dados

```
export OS_TENANT_NAME=demo
export OS_USERNAME=demo
export OS_PASSWORD=ADMIN_PASS
export OS_AUTH_URL=http://controller:35357/v3
```

Capítulo 4

Serviço de Imagem

O serviço de imagem do OpenStack (glance) permite que os usuários para descobrir, registrar e recuperar imagens de máquinas virtuais. Ele oferece uma API REST que permite consultar metadados imagem de máquina virtual e recuperar uma imagem real. pode armazenar imagens de máquinas virtuais disponibilizados através do serviço de imagem em uma variedade de locais, de sistemas de arquivos simples de se opor de armazenamento de sistemas como o Object Storage OpenStack.

4.1 Serviço de Imagem Openstack

O serviço de imagem do OpenStack aceita solicitações de API para disco ou imagens de servidor, e metadados de imagens dos usuários finais ou componentes OpenStack Compute. Ele também suporta o armazenamento de imagens de disco ou servidor em vários tipos de repositório, incluindo o OpenStack Object Storage

Um número de processos periódicos executado no serviço de imagem OpenStack para suportar o armazenamento em cache. serviços de replicação de assegurar a consistência e disponibilidade por meio do cluster. Outros processos periódicos incluem auditores, atualizadores e ceifeiros.

O serviço de imagem do OpenStack possui os seguintes Componentes

glance-api	Aceita chamadas Imagem de API para a descoberta de imagem, recuperação e armazenamento.
glance-registry	Armazena, processos e recupera metadados sobre imagens. Os metadados incluem itens como tamanho e tipo.
Banco de Dados	Armazena imagem metadados e pode ser escolhido o seu banco de dados, dependendo da preferência.
Repositório de armazenamento para arquivos de imagem	Vários tipos de repositórios são suportados, incluindo sistemas de arquivos normais,dispositivos de bloco RADOS, HTTP e Amazon S3. Alguns destes repositórios são somente de leitura.

4.2 Instalando e configurando os pacotes

4.2.1 Configurando pre-requisitos

Antes de instalar e configurar o serviço de imagem, deve ser criado um banco de dados, as credenciais de serviço, e ponto final API.

1. Acessar o banco de dados de acesso cliente para conectar ao servidor da base de dados como *root* usuário:

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

2. Para criar o banco de dados, executar as seguintes instruçoes

```
CREATE DATABASE glance;

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

GRANT ALL PRIVILEGES ON glance.* TO 'glance'@"%' IDENTIFIED BY 'GLANCE_DBPASS'
```

3. De origem as credenciais de administrador para ter acesso ao admin-somente comandos CLI:

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

4. Para criar as credenciais de serviço, execute as seguintes funcoes

   - Criar usuario glance

     ```
     $ keystone user-create –name glance –pass $KEYSTONE_USER_GLANCE_PASSWORD –email $EMAIL_GLANCE
     ```

   - Adicionar a regra *admin* para o usuário *glance* e o projeto de serviço

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

   - Criar a entidade de serviço *glance*

     ```
     keystone service-create –name glance –type image –description "OpenStack Image Service"
     ```

   - Criar API endpoint do serviço de imagem

     ```
     $ keystone endpoint-create \
     –service-id $(keystone service-list |awk '|image / print $2') \
     –publicurl http://controller:9292 \
     –internalurl http://controller:9292 \
     –adminurl http://controller:9292 \
     –region regionOne
     ```

4.2.2 Instalar e configurar componentes

1. Instalar pacotes

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

2. Editar o arquivo */etc/glance/glance-api.conf* e editar os seguintes dados

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

...



[keystone_authtoken]
...
auth_uri = http://controller:5000
auth4_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = glance
password = GLANCE_PASS

...



[paste_deploy]
...
flavor = keystone

...



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

...



[default]
...
notification_driver = noop
...
verbose = True
```

3. Editar arquivo */etc/glance/glance-registry.conf* com as seguintes configurações

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

...



[keystone_authtoken]
...
auth_uri = http://controller:5000
auth_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = glance
password = GLANCE_PASS

...



[paste_deploy]
...
flavor = keystone

...



[default]
...
notification_driver = noop
...
verbose = True
```

4. Popular o banco de dados do serviço de imagem

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

4.2.3 Finalizando a Instalação

1. Reiniciar os serviços do serviço de imagem

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

2. Por padrão, os pacotes Ubuntu cria um banco de dados SQLite. Por conta desta configuracao utilizar o servidor banco de dados SQL, remover a base de dados do arquivo SQLite:

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

4.3 Verificar Operação

Verificar o funcionamento do serviço de imagem usando CirrOS, que é uma imagem pequena Linux que ajuda a testar a implantação OpenStack.

1. Em cada ambiente de script de cliente, configurar o cliente serviço de imagem para usar a versão da API 2.0

```
$ echo "export OS_IMAGE_API_VERSION=2"|tee -a admin-openrc.sh demo-openrc.sh
```

2. De origem as credenciais de administrador para ter acesso ao admin-somente comandos CLI:

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

3. Criar um diretório temporário local

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

4. Faça o download da imagem dentro do diretório

```
$ wget -P /tmp/images http://download.cirros-cloud.net/0.3.4/cirros-0.3.4-x86_64-disk.img
```

5. Faça o Upload da imagem para o serviço de imagem usando o formato de disco QCOW2 , formato de container *bare*, visibilidade publica,ou seja, todos os projetos podem acessá-lo

```
$ glance image-create –name "cirros-0.3.4-x86_64--file /tmp/images/ cirros-0.3.4-x86_64-disk.img \
–disk-format qcow2 –container-format bare –is-public True –progress <cirros-0.3.2-x86_64-disk.img progress
```

6. Confirmar o upload da imagem e validar atributos

```
$ glance image-list
```

7. Remover o diretório temporário local e seus arquivos

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

Capítulo 5

Serviço Compute

Usa-se o OpenStack Compute para hospedar e gerenciar sistemas de computação nas nuvens. OpenStack Compute é parte principal de um sistema IaaS (Infrastructure-as-a-Service). Os módulos principais são implementados em Phyton.

OpenStack Compute interage com o OpenStack Identity para autenticação, com o serviço OpenStack Image para disco e servidor de imagens. Acesso a imagem é limitado por projetos e usuários; parte é limitada para projeto (numero de instâncias, por exemplo). OpenStack Compute pode escalar horizontalmente os padrões de software fazer o download de imagens e executar instâncias.

	API
Serviço nova-api	Aceita e reponde para o usuário final computar a API de chamadas. Faz cumprir algumas políticas e inicia a execução de ima instância.
Servico nova-api-metadata	Aceita requisições de metadados vindos das intâncias. O serviço *nova-api-metadata* é geralmente utilizado quando executa-se em modo multi-host a instalação *nova-network*

Nucleo Compute	
Serviço nova-compute	Um daemon operário que cria e termina as instâncias de máquinas virtuais através da API do hypervisor . Como exemplo, libvirt para KVM ou QEMU, VMwareAPI para VMware. O processo e complexo. Basicamente, o daemon aceita ações vindos da fila e realiza uma série de comandos de sistema assim como lançar uma instância API KVM e atualiza o estados na base de dados
Serviço nova-scheduler	Tira uma requisição da instância da máquina virtual da fila e determina em qual servidor compute hospedeiro ele executa
Modulo nova-conductor	Media as interações entre o serviço *nova-compute* e o banco de dados. Elimina o acesso direto para o banco de dados nas nuvens feito pelo serviço *nova-compute*. O módulo *nova-conductor* escala horizontalmente . Contudo, ele nao implanta dentro dos nós onde o s erviço *nova-compute* executa.
Modulo nova-cert	Um servidor daemon que serve o serviço Nova Cert. Utilizado para gerar certificados para ***euca-bundle-image***
Redes para as máquinas virtuais	
Daemon nova-network operário	Similar ao serviço *nova-compute*, aceita as tarefas da rede vindos da fila e manipula a rede. Realiza tarefas assim como ajusta e levanta as interfaces ou muda as regras IPTables
Interface de console	
daemon nova-consoleauth	Autoriza tokens para usuários que consoles de proxy fornecem. Este serviço deve estar executando para os consoles de proxy trabalharem.
daemon nova-novncproxy	Fornece um proxy para acessar instâncias em execução através de uma conexão VNC. Suporta clientes novnc browser baseado.
	Outr
A fila	Um hub daemons.
Banco de dados SQL	Armazena maioria do tempo construção tempo de execução estados para

5.1 Instalando e Configurando no nó Controllador

Configurando os pre-requisitos

Antes de instalar e configurar o serviço Compute, deve ser criado um banco de dados, as credenciais de serviço, e ponto final API.

1. Acessar o banco de dados de acesso cliente para conectar ao servidor da base de dados como *root* usuário:

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

2. Para criar o banco de dados, executar as seguintes instruçoes

```
CREATE DATABASE nova;

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

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

3. De origem as credenciais de administrador para ter acesso ao admin-somente comandos CLI:

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

4. Para criar as credenciais de serviço, execute as seguintes funcoes

   - Criar usuario nova

   ```
   $ keystone user-create –name nova –pass $KEYSTONE_USER_NOVA_PASSWORD –email $EMAIL_NOVA
   ```

   - Adicionar a regra *admin* para o usuário *nova* e o projeto de serviço

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

   - Criar a entidade de serviço *nova*

   ```
   keystone service-create –name nova –type compute–description "OpenStack Compute"
   ```

   - Criar API endpoint do serviço de imagem

   ```
   $ keystone endpoint-create \
   –service-id $(keystone service-list |awk '|image / print $2') \
   –publicurl http://controller:9292/v2/%\(tenant_id\)s \
   –internalurl http://controller:9292/v2/%\(tenant_id\)s \
   –adminurl http://controller:9292/v2/%\(tenant_id\)s \
   –region regionOne
   ```

5.1.1 Instalando e Configurando os componentes do nó controlador

1. Instalar os pacotes

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

2. Editar o arquivo */etc/nova/nova.conf* com os seguintes dados

```
[default]
...
verbose = true
rpc_backend = rabbit
auth_strategy = keystone
my_ip= 10.129.64.51 vncserver_listen = 10.129.64.51 vncserver_proxyclient_address
= 10.129.64.51
...



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



...



[oslo_messaging_rabbit]
... rabbit_host = controller
rabbit_userid = openstack
rabbit_password = RABBIT_PASS

...



[keystone_authtoken]
...
auth_uri = http://controller:5000
auth_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = glance
password = •NOVAPASS

...



[glance]
...
```

```
host = controller

...



[oslo_concurrency]
...
lock_path = /var/lib/nova/tmp
```

3. Popular a base de dados do Compute

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

5.1.2 Finalizando a instalação

1. Reiniciar os serviços Compute

```
# 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 Ubuntu cria um banco de dados SQLite. Por conta desta configuracao utilizar o servidor banco de dados SQL, remover a base de dados do arquivo SQLite:

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

5.2 Instalando e Configurando no nó Compute

Instalando e configurando os componentes do hypervisor no Compute

1. Instalar os pacotes

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

2. Editar o arquivo */etc/nova/nova.conf* e inserir os seguintes dados

```
[default]
...
verbose = true
rpc_backend = rabbit
auth_estrategy = keystone
my_ip= 10.129.64.61 vnc_enabled = true vncserver_listen = 0.0.0.0 vncserver_proxyclient_address = 10.129.64.61 novncproxy_base_url = http://controller:6080/vnc_auto.html
...
```

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



...



[oslo_messaging_rabbit]
... rabbit_host = controller
rabbit_userid = openstack
rabbit_password = RABBIT_PASS

...



[keystone_authtoken]
...
auth_uri = http://controller:5000
auth_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = glance
password = NOVAPASS

...



[glance]
...
host = controller

...



[oslo_concurrency]
...
lock_path = /var/lib/nova/tmp
```

5.2.1 Finalizando a instalação

1. Determinar se o nó compute suporta aceleração de hardware para maquinas virtuais

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

Se o comando retornar valor maior ou igual a 1, o nó compute suporta aceleração de hardware. Se o comando retornar valor igual a 0, o nó compute nao suporta aceleração de hardware e é então necessário configurar *libvirt* para usar QEMU no lugar de KVM

- Edite a seção *[libvirt]* em */etc/nova/nova-compute.conf* como mostra a seguir

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

2. Reiniciar o serviço Compute

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

3. Por padrão, os pacotes Ubuntu cria um banco de dados SQLite. Por conta desta configuracao utilizar o servidor banco de dados SQL, remover a base de dados do arquivo SQLite:

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

5.3 Verificar a Operação

1. De origem as credenciais de administrador para ter acesso ao admin-somente comandos CLI:

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

2. Listar os componentes de serviço para verificar se executou com sucesso e registrar cada processo

```
$ nova service-list
```

3. Listar os nós extremos da API no serviço de autenticação para verificar conectividade com o serviço de autenticação

```
$ nova endpoints
```

4. Listar as imagens no catálogo de serviço de imagens para verificar a conectividade com o serviço de imagens

```
$ nova image-list
```

Capítulo 6

Adicionando a rede

Redes Legacy envolve principalmente nós de computação. No entanto, é necessário configurar o nó controlador para usar a rede Legacy

6.1 Configurando o nó controlador

1. Editar o arquivo */etc/nova/nova.conf* e completar as seguintes ações

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

2. Reiniciar os serviços Compute

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

6.2 Configurando o nó Compute

Esta seção implantação de uma rede plana simples que fornece endereços IP a instâncias via DHCP.

6.2.1 Instalar componentes de rede Legacy

- Instalar o seguinte pacote

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

6.2.2 Configurando a rede Legacy

1. Editar o arquivo */etc/nova/nova.conf* e colocar os seguintes dados

```
[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
```

2. Reiniciar os serviços

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

6.3 Criando uma rede inicial

Antes de lançar a primeira instância, deve-se criar a infra-estrutura de rede virtual necessária para que a instância irá se conectar. Esta rede normalmente fornece acesso à Internet a partir de instâncias. Pode permitir o acesso à Internet para instâncias individuais usando um IP flutuante endereço e regras do grupo de segurança adequadas. O inquilino administrador detém esta rede, pois fornece acesso à rede externa para vários inquilinos.

Esta rede tem a mesma sub-rede associado com a rede física ligada à interface externa no nó de computação.Deve ser especificado uma fatia exclusivo deste sub-rede para evitar a interferência com outros dispositivos na rede externa.

OBS: Executar os comandos no nó controlador

Criando a rede

1. De origem as credenciais de administrador para ter acesso ao admin-somente comandos CLI:

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

2. Criar a rede

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

3. Verificar a criação da rede

```
$ nova net-list
```