Documentação da nuvem computacional do LabEPI/ CERES baseada na documentação oficial do OpenStack Kilo para Ubuntu 14.04 LTS (a versão utilizada do Ubuntu é a Server 14.04.3 – Trusty Tahr), disponível em:

<http://docs.openstack.org/kilo/install-guide/install/apt/content/ch_preface.html>. Acesso em: 20 ago. 2015.

Considerações iniciais

1. Prompts de comando:

   `$` prompt – Qualquer usuário pode executar comandos prefixados com este prompt;

   `#` prompt – Apenas usuário `root` pode executar comandos prefixados com este prompt.

2. As palavras entre colchetes '[ ]' são seções de arquivos de configuração;
3. Observações gerais:
   - As reticências indicam que podem existir outras configurações nas seções dos arquivos de configuração;
   - O símbolo '*' indica que existe algo a ser considerado nas configurações realizadas, não faz parte do conteúdo do arquivo de configuração diretamente.
   - A configuração `verbose = True` é opcional.
4. O arquivo de configuração citado que não estiver presente no diretório indicado deve ser criado;
5. A seção de configuração que não estiver presente no arquivo de configuração deve ser criada com o nome conforme indicado.

Serviços OpenStack

Os serviços OpenStack instalados foram: Dashboard – Horizon, Compute – Nova, Networking – Neutron, Block Storage – Cinder, Serviço de Identidade – Keystone, Serviço de Imagem – Glance.

Arquitetura de Hardware

- Controller Node – 1 CPU (4 Cores); 8 GB RAM; 320 GB Storage; 1 NIC Gb/s, SO 64 bits;
- Compute Node – 4 CPU (4 Cores cada); 16 GB RAM; 500 + 250 GB Storage (250 GB disponíveis utilizando RAID 5); 2 NIC Gb/s, SO 64 bits;
- Network Node – 1 CPU; 2 GB RAM; 40 GB Storage; 3 NIC Gb/s; SO 64 bits;
- Block Storage Node – 1 CPU; 1, 5 GB RAM; 40 + 500 GB Storage; 1 NIC Gb/s, SO 32 bits.

Arquitetura de Rede (Utilizando o Neutron)

1. *Nó* Controller

   Interface de gerenciamento:

   ```
   IP address: 192.168.0.12
   Network mask: 255.255.255.0 (or /24)
   Default gateway: 192.168.0.254
   ```

2. *Nó* Network

Primeira interface (gerenciamento):

```
IP address: 192.168.0.14
Network mask: 255.255.255.0 (or /24)
Default gateway: 192.168.0.254
```

Segunda interface (criação de túneis de instância):

```
IP address: 192.168.1.14
Network mask: 255.255.255.0 (or /24)
```

Terceira interface (acesso externo):

```
auto INTERFACE_NAME
iface INTERFACE_NAME inet manual
        up ip link set dev $IFACE up
        down ip link set dev $IFACE down
```

*Observação: *INTERFACE_NAME* foi substituído pela identificação da terceira interface do *nó* compute

3. *Nó* Compute

   Interface de gerenciamento:

   ```
   IP address: 192.168.0.13
   Network mask: 255.255.255.0 (or /24)
   Default gateway: 192.168.0.254
   ```

   Segunda interface (criação de túneis de instância):

   ```
   IP address: 192.168.1.13
   Network mask: 255.255.255.0 (or /24)
   ```

4. *Nó* Block Storage

   Interface de gerenciamento:

   ```
   IP address: 192.168.0.15
   Network mask: 255.255.255.0 (or /24)
   Default gateway: 192.168.0.254
   ```

5. Configuração do arquivo `/etc/hosts` de cada *nó*:

   ```
   192.168.0.12     controller.labepi.ufrn.br      controller
   192.168.0.13     compute.labepi.ufrn.br         compute
   192.168.0.14     network.labepi.ufrn.br         network
   192.168.0.15     blockStorage.labepi.ufrn.br    blockStorage
   ```

   *Observação: A linha do arquivo que contem o endereço de loopback deve ser removida ou comentada.

6. A verificação da conectividade entre as máquinas foi feita através da utilização do comando *ping*.

Instalação e configuração do Network Time Protocol (NTP)

Instalação e configuração no *nó* controller

1. Instalação do serviço NTP:

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

2. Configuração do arquivo `/etc/ntp.conf`:

```
server 127.127.1.0
fudge 127.127.1.0 stratum 10

server 192.168.0.12 iburst
restrict -4 default kod notrap nomodify
restrict -6 default kod notrap nomodify
```

3. Linhas comentadas ou removidas do arquivo `/etc/ntp.conf`:

```
restrict -4 default kod notrap nomodify nopeer noquery
restrict -6 default kod notrap nomodify nopeer noquery
```

4. O arquivo `/var/lib/ntp/ntp.conf.dhcp` deve ser removido, caso exista.

5. Reinicialização do serviço NTP:

```
# service ntp restart
```

Nos demais *nós*

1. Instalação do serviço NTP:

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

2. Configuração do arquivo `/etc/ntp.conf`:

```
server controller iburst
```

3. Linhas comentadas ou removidas do arquivo `/etc/ntp.conf`:

```
server 0.ubuntu.pool.ntp.org
server 1.ubuntu.pool.ntp.org
server 2.ubuntu.pool.ntp.org
server 3.ubuntu.pool.ntp.org

server ntp.ubuntu.com

restrict -4 default kod notrap nomodify nopeer noquery
restrict -6 default kod notrap nomodify nopeer noquery
```

*Observações:

- O arquivo `/var/lib/ntp/ntp.conf.dhcp` deve ser removido, caso exista;
- Todos os nós devem referenciar apenas o *nó* controller como servidor NTP.

Verificação das operações do serviço NTP

1. Resultado da execução do comando `ntpq -c peers` no *nó* controller:

```
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
```

```
 c.st1.ntp.br    .INIT.          16 u    - 1024    0    0.000    0.000   0.000
 gps.ntp.br      .INIT.          16 u    - 1024    0    0.000    0.000   0.000
 vel.itat.io     .INIT.          16 u    - 1024    0    0.000    0.000   0.000
 santuario.pads. .INIT.          16 u    - 1024    0    0.000    0.000   0.000
 juniperberry.ca .INIT.          16 u    - 1024    0    0.000    0.000   0.000
*LOCAL(0)        .LOCL.          10 l    1   64  377    0.000    0.000   0.000
 controller.labe .INIT.          16 u    - 1024    0    0.000    0.000   0.000
```

2. Resultado da execução do comando `ntpq -c peers` nos demais *nós*:

a. *Nó* compute:

```
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*controller.labe LOCAL(0)        11 u   50   64  377    0.251   80.253   0.005
```

b. *Nó* network:

```
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*controller.labe LOCAL(0)        11 u   43   64  377    0.306   20.360   3.980
```

c. *Nó* blockStorage:

```
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*controller.labe LOCAL(0)        11 u   55   64  377    0.238    0.340   0.154
```

*Observação: Na coluna `remote`, observa-se que pelo menos o servidor local está disponível, isso é suficiente para as finalidades deste projeto.

3. Resultado da execução do comando `ntpq -c assoc` no *nó* controller:

```
ind assid status  conf reach auth condition  last_event cnt
===========================================================
  1 32036  8011   yes    no  none    reject    mobilize  1
  2 32037  8011   yes    no  none    reject    mobilize  1
  3 32038  8011   yes    no  none    reject    mobilize  1
  4 32039  8011   yes    no  none    reject    mobilize  1
  5 32040  8011   yes    no  none    reject    mobilize  1
  6 32041  963a   yes   yes  none  sys.peer    sys_peer  3
  7 32042  8011   yes    no  none    reject    mobilize  1
```

4. Resultado da execução do comando `ntpq -c assoc` nos demais *nós*:

a. *Nó* compute:

```
ind assid status  conf reach auth condition  last_event cnt
===========================================================
  1  2698  963a   yes   yes  none  sys.peer    sys_peer  3
```

b. *Nó* network:

```
ind assid status  conf reach auth condition  last_event cnt
===========================================================
  1 61148  965a   yes   yes  none  sys.peer    sys_peer  5
```

c. *Nó* blockStorage:

```
ind assid status  conf reach auth condition  last_event cnt
===========================================================
```

```
1  45597  963a   yes   yes  none  sys.peer    sys_peer  3
```

*Observação: Na coluna `condition`, observa-se que pelo menos o servidor local está sincronizando, isto é suficiente para as finalidades deste projeto.

Habilitação do repositório dos pacotes do OpenStack

1. Instalação dos arquivos chaves e repositório para a nuvem utilizando o Ubuntu (todos os *nós*):

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

2. Atualização dos pacotes e do sistema:

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

*Observações:

- As atualizações automáticas foram desabilitadas para evitar problemas futuros;
- Havendo necessidade, deve-se reinicializar o sistema.

Instalação do banco de dados MariaDB

Instalação e configuração do servidor de banco de dados no *nó* contoller

1. Instalação dos pacotes:

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

2. Configuração do arquivo `/etc/mysql/conf.d/mysqld_openstack.cnf`, após criado:

```
[mysqld]

bind-address = 192.168.0.12
default-storage-engine = innodb
innodb_file_per_table
collation-server = utf8_general_ci
init-connect = 'SET NAMES utf8'
character-set-server = utf8
```

*Observação: O arquivo `mysqld_openstack.cnf` foi criado no respectivo diretório citado acima.

3. Reinicialização do serviço de banco de dados:

```
# service mysql restart
```

4. Configuração de proteção do serviço de banco de dados:

```
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.
```

```
Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n] Y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] Y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] Y
 ... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

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

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] Y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```

Instalação e configuração do serviço de enfileiramento de mensagens

Instalação e configuração no *nó* controller

1. Instalação do pacote:

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

2. Adição do usuário `openstack`:

   ```
   # rabbitmqctl add_user openstack RABBIT_PASS
   ```

   *Observação: *`RABBIT_PASS`* foi substituído por uma senha adequada.

3. Configuração para permitir acesso à leitura e escrita ao usuário `openstack`:

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

Instalação e configuração do serviço de identidade do OpenStack – Keystone

1. Criação do banco de dados
   a. Acesso à base de dados como usuário `root`:

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

   b. Criação do banco de dados `keystone`:

   ```
   CREATE DATABASE keystone;
   ```

   c. Concessão de acesso ao banco de `keystone`:

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

   *Observação: *`KEYSTONE_DBPASS`* foi substituído por uma senha adequada.

2. Geração de um valor aleatório para ser usado como token de administração:

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

3. Instalação e configuração dos componentes do serviço de identidade
   a. Instalação dos pacotes:

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

4. Configuração do arquivo `/etc/keystone/keystone.conf`:

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

   [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

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

*Observações:

KEYSTONE_DBPASS foi substituído pela senha escolhida;

ADMIN_TOKEN foi substituído por um valor adequado gerado anteriormente (token de administração).

5. Povoamento do banco de dados `keystone`:

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

6. Configuração do servidor Apache HTTP
   a. Configuração do arquivo `/etc/apache2/sites-available/wsgi-keystone.conf`, após cria-lo:

```
Listen 5000
Listen 35357

<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>

<VirtualHost *:35357>
    WSGIDaemonProcess keystone-admin processes=5 threads=1 user=keystone display-
name=%{GROUP}
    WSGIProcessGroup keystone-admin
    WSGIScriptAlias / /var/www/cgi-bin/keystone/admin
    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>
```

   b. Habilitação do serviço de identidade para *nós* virtuais:

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

   c. Criação da estrutura de diretório para os componentes do WSGI:

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

   d. Comando executado para realizar a cópia dos componentes do repositório *online* do WSGI para o diretório criado:

```
# 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
```

e. Ajuste do proprietário e das permissões deste diretório e seus arquivos, respectivamente:

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

f. Remoção da opção de inicialização automática do serviço Keystone:

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

g. Reinicialização do servidor Apache:

```
# service apache2 restart
```

*Observação: Se necessário, o *nó* controller deve ser reinicializado neste passo.

7. Remova o arquivo do banco de dados SQLite (não utilizado):

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

8. Criação da entidade do serviço e endpoint da API

a. Configuração do token de administração:

```
$ export OS_TOKEN=ADMIN_TOKEN
```

*Observação: *ADMIN_TOKEN* foi substituído pelo token de administração gerado anteriormente.

b. Configuração do URL do endpoint:

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

c. Criação da entidade do serviço `keystone`:

```
$ openstack service create --name keystone --description \
"OpenStack Identity" identity
```

9. Criação do endpoint da API do serviço de identidade:

```
$ openstack endpoint create \
  --publicurl http://controller:5000/v2.0 \
  --internalurl http://controller:5000/v2.0 \
  --adminurl http://controller:35357/v2.0 \
  --region RegionOne \
  identity
```

10. Criação de projeto, usuário e papel para realização de operações administrativas no ambiente da nuvem

a. Criação do projeto `admin`:

```
$ openstack project create --description "Admin Project" admin
```

b. Criação do usuário `admin`:

```
$ openstack user create --password-prompt admin
```

c. Criação do papel `admin`:

```
$ openstack role create admin
```

d. Adição do papel `admin` ao projeto `admin` e ao usuário `admin`:

```
$ openstack role add --project admin --user admin admin
```

11. Criação do projeto `service`:

```
$ openstack project create --description "Service Project" service
```

12. Criação do projeto `demo` (não possui privilégios administrativos):

```
$ openstack project create --description "Demo Project" demo
```

13. Criação de projeto, usuário e papel não administrativo
    a. Criação do usuário `demo`:

```
$ openstack user create --password-prompt demo
```

    b. Criação do papel `user`:

```
$ openstack role create user
```

    c. Adicione o papel `user` ao projeto `demo` e usuário `demo`:

```
$ openstack role add --project demo --user demo user
```

14. Verificação da operação do serviço de identidade

    a. Desabilitação temporária do mecanismo de autenticação por token:
    Remova as *tags* `admin_token_auth` do arquivo `/etc/keystone/keystone-paste.ini`, nas seções `[pipeline:public_api]`, `[pipeline:admin_api]` e `[pipeline:api_v3]`

    b. Remoção dos valores das variáveis de ambiente temporárias `OS_TOKEN` e `OS_URL`:

```
$ unset OS_TOKEN OS_URL
```

    c. Requisição de um token de autenticação do serviço de identidade como usuário `admin`:

```
$ openstack --os-auth-url http://controller:35357 --os-project-name \
```

```
admin --os-username admin --os-auth-type password token issue
```

d. Requisição de um token de autenticação do serviço de identidade como usuário `admin` e utilizando as opções de domínio. Neste projeto específico a opção `domain` é `default`:

```
$ openstack --os-auth-url http://controller:35357 --os-project-domain-id \
default --os-user-domain-id default --os-project-name admin --os-username \
admin --os-auth-type password token issue
```

e. Exibição da lista dos projetos criados como usuário `admin`:

```
$ openstack --os-auth-url http://controller:35357 --os-project-name \
admin --os-username admin --os-auth-type password project list
```

f. Exibição da lista dos usuários criados como usuário `admin`:

```
$ openstack --os-auth-url http://controller:35357 --os-project-name \
admin --os-username admin --os-auth-type password user list
```

g. Exibição da lista dos papéis criados como usuário `admin`:

```
$ openstack --os-auth-url http://controller:35357 --os-project-name \
admin --os-username admin --os-auth-type password role list
```

h. Requisição de um token de autenticação do serviço de identidade como usuário `demo`:

```
$ openstack --os-auth-url http://controller:5000 --os-project-domain-id \
default --os-user-domain-id default --os-project-name demo --os-username \
demo --os-auth-type password token issue
```

i. Tentativa de exibição dos usuários criados como usuário `demo` (houve uma mensagem de erro, pois apenas o usuário `admin` tem esse privilégio):

```
$ openstack --os-auth-url http://controller:5000 --os-project-domain-id \
default --os-user-domain-id default --os-project-name demo --os-username \
demo --os-auth-type password user list
```

15. Criação de scripts de ambiente para clientes OpenStack

a. Configuração do arquivo `admin-openrc.sh`, após criado:

```
export OS_PROJECT_DOMAIN_ID=default
export OS_USER_DOMAIN_ID=default
export OS_PROJECT_NAME=admin
export OS_TENANT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=ADMIN_PASS
export OS_AUTH_URL=http://controller:35357/v3
export OS_REGION_NAME=RegionOne
```

*Observação: *`ADMIN_PASS`* foi substituído pela senha do usuário `admin` escolhida.

b. Configuração do arquivo `demo-openrc.sh`, após criado:

```
export OS_PROJECT_DOMAIN_ID=default
export OS_USER_DOMAIN_ID=default
export OS_PROJECT_NAME=demo
export OS_TENANT_NAME=demo
export OS_USERNAME=demo
export OS_PASSWORD=DEMO_PASS
export OS_AUTH_URL=http://controller:5000/v3
```

```
export OS_REGION_NAME=RegionOne
```

*Observação: *DEMO_PASS* foi substituído pela senha do usuário `demo` escolhida.

c. Comando para executar o script criado e carregar os valores nas variáveis de ambiente:

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

d. Requisição de um token de autenticação, após execução de um dos scripts criados:

```
$ openstack token issue
```

Instalação e configuração do serviço de imagem do OpenStack – Glance

1. Criação do banco de dados
   a. Acesso à base de dados como usuário `root`:

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

   b. Criação do banco de dados `glance`:

```
CREATE DATABASE glance;
```

   c. Concessão de acesso ao banco de `glance`:

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

*Observação: *GLANCE_DBPASS* foi substituído por uma senha adequada.

d. Obtenção das credenciais `admin`:

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

e. Criação do usuário `glance`:

```
$ openstack user create --password-prompt glance
```

f. Adição do papel `admin` ao usuário `glance`:

```
$ openstack role add --project service --user glance admin
```

g. Criação da entidade do serviço `glance`:

```
$ openstack service create --name glance --description "OpenStack \
Image service" image
```

h. Criação do endpoint da API do serviço de imagem:

```
$ openstack endpoint create \
  --publicurl http://controller:9292 \
  --internalurl http://controller:9292 \
  --adminurl http://controller:9292 \
  --region RegionOne \
```

image

2. Instalação e configuração dos componentes do serviço de imagem
   a. Instalação dos pacotes:

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

   b. Configuração do arquivo `/etc/glance/glance-api.conf`:

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

[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

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

[DEFAULT]
...
notification_driver = noop

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

   c. Configuração do arquivo `/etc/glance/glance-registry.conf`:

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

[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

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

*Observações:

GLANCE_DBPASS foi substituído pela senha escolhida;
GLANCE_PASS foi substituído pela senha escolhida;
Quaisquer outras configurações na seção `[keystone_authtoken]` devem ser comentadas ou removidas.

d. Povoamento do banco de dados `glance`:

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

3. Finalização da instalação

a. Reinicialização dos serviços do Glance:

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

b. Remova o arquivo do banco de dados SQLite (não utilizado):

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

4. Verificação da operação do serviço de imagem

a. Configuração para que o cliente do serviço de imagem utilize a versão 2.0 da API, em cada um dos scripts de ambiente criados:

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

b. Obtenção das credenciais `admin`:

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

c. Criação de um diretório local temporário:

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

d. Download da imagem do CirrOS:

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

e. Upload da imagem do CirrOS para o serviço de imagem:

```
$ 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 --visibility \
public --progress
```

f. Confirmação do upload da imagem e validação dos atributos:

```
$ glance image-list
```

g. Remoção do diretório local temporário e da imagem baixada:

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

Instalação e configuração do serviço de computação do OpenStack – Nova

Instalação e configuração no *nó* controller

1. Criação do banco de dados
   a. Acesso à base de dados como usuário `root`:

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

   b. Criação do banco de dados `nova`:

   ```
   CREATE DATABASE nova;
   ```

   c. Concessão de acesso ao banco de `nova`:

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

   *Observação: *NOVA_DBPASS* foi substituído por uma senha adequada.

2. Criação das credenciais do serviço

   a. Obtenção das credenciais `admin`:

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

   b. Criação do usuário `nova`:

   ```
   $ openstack user create --password-prompt nova
   ```

   c. Adição do papel `admin` ao usuário `nova`:

   ```
   $ openstack role add --project service --user nova admin
   ```

   d. Criação da entidade do serviço `nova`:

   ```
   $ openstack service create --name nova --description "OpenStack Compute" compute
   ```

3. Criação do endpoint da API do serviço de computação:

   ```
   $ openstack endpoint create \
     --publicurl http://controller:8774/v2/%\(tenant_id\)s \
     --internalurl http://controller:8774/v2/%\(tenant_id\)s \
   ```

```
--adminurl http://controller:8774/v2/%\(tenant_id\)s \
--region RegionOne \
compute
```

4. Instalação e configuração dos componentes para controle do serviço de computação
   a. Instalação dos pacotes:

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

   b. Configuração do arquivo `/etc/nova/nova.conf`:

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

[DEFAULT]
...
rpc_backend = rabbit
auth_strategy = keystone
my_ip = 192.168.0.12
vncserver_listen = 192.168.0.12
vncserver_proxyclient_address = 192.168.0.12
verbose = True

[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 = nova
password = NOVA_PASS

[glance]
...
host = controller

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

   *Observações:
   - *NOVA_DBPASS* foi substituído pela senha escolhida;
   - *RABBIT_PASS* foi substituído pela senha escolhida;
   - *NOVA_PASS* foi substituído pela senha escolhida;
   - Quaisquer outras configurações na seção `[keystone_authtoken]` devem ser comentadas ou removidas;

5. Povoamento do banco de dados `nova`:

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

6. Finalização da instalação

a. Reinicialização dos 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
```

b. Remova o arquivo do banco de dados SQLite (não utilizado):

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

Instalação e configuração no *nó* compute

1. Instalação e configuração dos componentes do hypervisor do serviço de computação

a. Instalação dos pacotes

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

b. Configuração do arquivo `/etc/nova/nova.conf`:

```
[DEFAULT]
...
rpc_backend = rabbit
auth_strategy = keystone
my_ip = 192.168.0.13
vnc_enabled = True
vncserver_listen = 0.0.0.0
vncserver_proxyclient_address = 192.168.0.13
novncproxy_base_url = http://controller:6080/vnc_auto.html
verbose = True

[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 = nova
password = NOVA_PASS

[glance]
```

```
...
host = controller

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

*Observações:

RABBIT_PASS foi substituído pela senha escolhida;
NOVA_PASS foi substituído pela senha escolhida;
Quaisquer outras configurações na seção `[keystone_authtoken]` devem ser comentadas ou removidas.

2. Finalização da instalação

a. Verificação se há suporte a aceleração de hardware para as máquinas virtuais:

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

*Observação: O valor apresentado após o comando executado está acima de zero, 16. O que indica que há suporte.

b. Reinicialização do serviço de computação:

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

c. Remova o arquivo do banco de dados SQLite (não utilizado):

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

3. Verificação da operação do serviço de computação (*nó* controller)

a. Obtenção das credenciais `admin`:

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

b. Comando para listagem dos componentes do serviço de computação instalados:

```
$ nova service-list
```

c. Comando para listagem dos endpoints das APIs no serviço de identidade:

```
$ nova endpoints
```

d. Comando para listagem das imagens no catálogo do serviço de imagem:

```
$ nova image-list
```

Instalação e configuração do serviço de rede do OpenStack – Neutron

Instalação e configuração no *nó* controller

1. Criação do banco de dados
   a. Acesso à base de dados como usuário `root`:
   ```
   $ mysql -u root -p
   ```

b. Criação do banco de dados `neutron`:

```
DATABASE neutron;
```

c. Concessão de acesso ao banco de dados `neutron`:

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

*Observação: *NEUTRON_DBPASS* foi substituído por uma senha adequada.

2. Criação das credenciais do serviço

a. Obtenção das credenciais admin:

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

b. Criação do usuário `neutron`:

```
$ openstack user create --password-prompt neutron
```

c. Adição do papel `admin` ao usuário `neutron`:

```
$ openstack role add --project service --user neutron admin
```

d. Criação da entidade do serviço `neutron`:

```
$ openstack service create --name neutron --description \
"OpenStack Networking" network
```

3. Criação do endpoint da API do serviço de rede:

```
$ openstack endpoint create \
  --publicurl http://controller:9696 \
  --adminurl http://controller:9696 \
  --internalurl http://controller:9696 \
  --region RegionOne \
  Network
```

4. Instalação e configuração dos componentes do serviço de rede

a. Instalação dos pacotes:

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

b. Configuração do arquivo `/etc/neutron/neutron.conf`:

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

[DEFAULT]
...
rpc_backend = rabbit

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

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

[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 = neutron
password = NEUTRON_PASS

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

[DEFAULT]
...
notify_nova_on_port_status_changes = True
notify_nova_on_port_data_changes = True
nova_url = http://controller:8774/v2

[nova]
...
auth_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
region_name = RegionOne
project_name = service
username = nova
password = NOVA_PASS

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

*Observações: *NEUTRON_DBPASS* foi substituído por uma senha adequada;

RABBIT_PASS foi substituído pela senha escolhida;

NEUTRON_PASS foi substituído pela senha escolhida;
Quaisquer outras configurações na seção [keystone_authtoken] devem ser comentadas ou removidas;

NOVA_PASS foi substituído pela senha escolhida.

5. Configuração do plug-in Modular Layer 2 (ML2)
    a. Configuração do arquivo `/etc/neutron/plugins/ml2/ml2_conf.ini`:

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

[ml2_type_gre]
...
tunnel_id_ranges = 1:1000

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

6. Configuração do serviço de computação para utilização do serviço de rede

    a. Configuração do arquivo `/etc/nova/nova.conf`:

```
[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

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

    *Observação: *NEUTRON_PASS* foi substituído pela senha escolhida.

7. Finalização da instalação

    a. Povoamento do banco de dados `neutron`:

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

    b. Reinicialização do serviço de computação:

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

    c. Reinicialização do serviço de rede:

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

8. Verificação da operação do serviço

a. Obtenção das credenciais de administrador:

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

b. Comando para listagem das extensões para verificação do pleno funcionamento do processo `neutron-server`:

```
$ neutron ext-list
```

Instalação e configuração do *nó* network

1. Configuração dos parâmetros do kernel do sistema operacional

   a. Configuração do arquivo `/etc/sysctl.conf`:

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

   b. Implementação das alterações:

   ```
   # sysctl -p
   ```

2. Instalação e configuração dos componentes do serviço de rede

   a. Instalação dos componentes:

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

   b. Configuração do arquivo `/etc/neutron/neutron.conf`:

   ```
   [DEFAULT]
   ...
   rpc_backend = rabbit

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

   [DEFAULT]
   ...
   auth_strategy = keystone
   [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 = neutron
   password = NEUTRON_PASS

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

   [DEFAULT]
   ```

```
...
verbose = True
```

*Observações: *RABBIT_PASS* foi substituído pela senha escolhida;
Na seção [database], comente qualquer opção de conexão para que o *nó* network não acesse o banco de dados diretamente;
NEUTRON_PASS foi substituído pela senha escolhida;
Quaisquer outras configurações na seção [keystone_authtoken] devem ser comentadas ou removidas.

3. Configuração do plug-in Modular Layer 2 (ML2):
   a. Configuração do arquivo /etc/neutron/plugins/ml2/ml2_conf.ini:

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

[ml2_type_flat]
...
flat_networks = external

[ml2_type_gre]
...
tunnel_id_ranges = 1:1000

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

[ovs]
...
local_ip = 192.168.1.14
bridge_mappings = external:br-ex

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

4. Configuração do agente Layer-3 (L3)
   a. Configuração do arquivo /etc/neutron/l3_agent.ini:

```
[DEFAULT]
...
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
external_network_bridge =
router_delete_namespaces = True

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

5. Configuração do agente DHCP
   a. Configuração do arquivo /etc/neutron/dhcp_agent.ini:

```
[DEFAULT]
```

```
...
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
dhcp_delete_namespaces = True

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

6. Configuração do serviço DHCP para ajuste do MTU
   a. Configuração do arquivo `/etc/neutron/dhcp_agent.ini`:
   ```
   [DEFAULT]
   ...
   dnsmasq_config_file = /etc/neutron/dnsmasq-neutron.conf
   ```

   b. Configuração do arquivo `/etc/neutron/dnsmasq-neutron.conf`, após ser criado no diretório indicado:

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

   c. Eliminação de quaisquer processos `dnsmasq` existente:
   ```
   # pkill dnsmasq
   ```

7. Configuração do agente metadata
   a. Configuração do arquivo `/etc/neutron/metadata_agent.ini`:
   ```
   [DEFAULT]
   ...
   auth_uri = http://controller:5000
   auth_url = http://controller:35357
   auth_region = RegionOne
   auth_plugin = password
   project_domain_id = default
   user_domain_id = default
   project_name = service
   username = neutron
   password = NEUTRON_PASS

   [DEFAULT]
   ...
   nova_metadata_ip = controller

   [DEFAULT]
   ...
   metadata_proxy_shared_secret = METADATA_SECRET

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

   b. Configuração do arquivo `/etc/nova/nova.conf` (no *nó* controller):
   ```
   [neutron]
   ...
   service_metadata_proxy = True
   metadata_proxy_shared_secret = METADATA_SECRET
   ```

   c. Reinicialização da API do serviço de computação (no *nó* controller):
   ```
   # service nova-api restart
   ```

*Observações: *NEUTRON_PASS* foi substituído pela senha escolhida;
METADATA_SECRET foi substituído por uma chave secreta adequada.

8. Configuração do serviço Open vSwitch (OVS)

    a. Renicialização do serviço OVS:
    ```
    # service openvswitch-switch restart
    ```

    b. Adição da bridge externa:
    ```
    # ovs-vsctl add-br br-ex
    ```

    c. Adição de uma porta para a bridge externa que se conecta a interface física da rede externa:
    ```
    # ovs-vsctl add-port br-ex eth2
    ```

    d. Desabilitação temporária da GRO da interface de rede externa:
    ```
    # ethtool –K eth2 gro off
    ```

9. Reinicialização dos componentes do serviço de rede:

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

10. Verificação da operação do serviço (no *nó* controller)

    a. Obtenção das credenciais de administrador:
    ```
    $ source admin-openrc.sh
    ```

    b. Comando para listagem dos agentes do serviço de rede instalados, para verificar o pleno funcionamento do neutron:
    ```
    $ neutron agent-list
    ```

Instalação e configuração no *nó* compute

1. Configuração dos parâmetros do kernel do sistema operacional

    a. Verificação do nome do módulo do kernel para ativar o modo bridge das interfaces de rede:
    ```
    $ egrep -r -i "nf-filter" /lib/modules/$(uname -r)/kernel/net/*
    ```

    b. Carregando o módulo `br-netfilter`, no nosso caso:
    ```
    # modprobe br-netfilter
    ```

    c. Configuração do arquivo `/etc/sysctl.conf`:

    ```
    net.ipv4.conf.all.rp_filter=0
    net.ipv4.conf.default.rp_filter=0
    net.bridge.bridge-nf-call-iptables=1
    net.bridge.bridge-nf-call-ip6tables=1
    ```

    d. Implementação das alterações:
    ```
    # sysctl –p
    ```

2. Instalação dos componentes do serviço de rede
```
# apt-get install neutron-plugin-ml2 neutron-plugin-openvswitch-agent
```

3. Configuração dos componentes comuns do serviço de rede
    a. Configuração do arquivo `/etc/neutron/neutron.conf`:

```
[DEFAULT]
...
rpc_backend = rabbit

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

[DEFAULT]
...
auth_strategy = keystone

[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 = neutron
password = NEUTRON_PASS

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

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

*Observações: *RABBIT_PASS* foi substituído pela senha escolhida;
Na seção `[database]`, comente qualquer opção de conexão para que o *nó* compute não acesse o banco de dados diretamente;
NEUTRON_PASS foi substituído pela senha escolhida;
Quaisquer outras configurações na seção `[keystone_authtoken]` devem ser comentadas ou removidas.

4. Configuração do plug-in Modular Layer 2 (ML2)

    a. Configuração do arquivo `/etc/neutron/plugins/ml2/ml2_conf.ini`:

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

[ml2_type_gre]
...
```

```
tunnel_id_ranges = 1:1000

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

[ovs]
...
local_ip = 192.168.1.13

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

5. Reinicialização do serviço do Open vSwitch (OVS):
   `# service openvswitch-switch restart`

6. Configuração do serviço de computação para utilizar o serviço de rede
   a. Configuração do arquivo `/etc/nova/nova.conf`:

```
[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

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

   *Observação: *NEUTRON_PASS* foi substituído pela senha escolhida.

7. Finalização da instalação
   a. Reinicialização do serviço de computação:
      `# service nova-compute restart`

   b. Reinicialização do agente Open vSwitch (OVS):
      `# service neutron-plugin-openvswitch-agent restart`

8. Verificação da operação do serviço (no *nó* controller)

   a. Obtenção das credenciais de administrador:
      `$ source admin-openrc.sh`

   b. Comando para listagem dos agentes do serviço de rede instalados, para verificar o pleno funcionamento do neutron:
      `$ neutron agent-list`

Criação das redes iniciais (no *nó* controller)

1. Criação da rede externa:
   a. Obtenção das credenciais de administrador:
      ```
      $ source admin-openrc.sh
      ```
   b. Criação da rede:
      ```
      $ neutron net-create ext-net --router:external --provider:physical_network \
      external --provider:network_type flat
      ```
   c. Criação de uma sub-rede na rede externa:
      ```
      $ neutron subnet-create ext-net 192.168.0.0/24 --name ext-subnet \
      --allocation-pool start=192.168.0.200,end=192.168.0.250 \
      --disable-dhcp --gateway 192.168.0.1
      ```
2. Criação da rede inquilino
   a. Obtenção das credencias de usuário:
      ```
      $ source demo-openrc.sh
      ```
   b. Criação da rede:
      ```
      $ neutron net-create demo-net
      ```
   c. Criação de uma sub-rede na rede inquilino:
      ```
      $ neutron subnet-create demo-net 172.16.0.0/24 \
        --name demo-subnet --gateway 172.16.0.1
      ```
3. Criação de um roteador na rede inquilo e anexando-o à rede externa para permitir a comunicação entre as mesmas
   a. Criação do roteador:
      ```
      $ neutron router-create demo-router
      ```
   b. Anexação do roteador à sub-rede da rede inquilino (`demo`):
      ```
      $ neutron router-interface-add demo-router demo-subnet
      ```
   c. Anexação do roteador à rede externa configurando-o como gateway:
      ```
      $ neutron router-gateway-set demo-router ext-net
      ```
   d. A verificação da conectividade foi feita através do comando *ping*.

Instalação e configuração da interface web (dashboard) para gerenciamento dos recursos do OpenStack – Horizon

Instalação e configuração no *nó* controller

1. Instalação dos pacotes dos componentes do dashboard:
   ```
   # apt-get install openstack-dashboard
   ```
2. Configuração do arquivo `/etc/openstack-dashboard/local_settings.py` :

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

```
OPENSTACK_KEYSTONE_DEFAULT_ROLE = "user"
TIME_ZONE = "UTC"
```

*Observação: Comente quaisquer outras configurações de armazenamento de sessão.

3. Finalização da instalação

a. Recarregamento das configurações do servidor web:
```
# service apache2 reload
```

4. Verificação da operação do dashboard

a. Acesso ao dashboard através de um navegador web:
http://192.168.0.12/horizon ou http://controller/horizon

b. Logo após, autentica – se utilizando as credenciais `admin` ou `demo`.

Instalação e configuração do serviço de armazenamento em blocos do OpenStack – Cinder

Instalação e configuração no *nó* controller

1. Criação do banco de dados
   a. Acesso à base de dados como usuário `root`:

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

   b. Criação do banco de dados `cinder`:
   ```
   DATABASE cinder;
   ```

   c. Concessão de acesso ao banco de `cinder`:
   ```
   GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'controller' IDENTIFIED  BY \
   'CINDER_DBPASS';
   GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'%' IDENTIFIED \
   BY 'CINDER_DBPASS';
   ```

   *Observação: *CINDER_DBPASS* foi substituído por uma senha adequada.

2. Criação das credenciais do serviço

   a. Obtenção das credenciais de administrador:
   ```
   $ source admin-openrc.sh
   ```

   b. Criação do usuário `cinder`:

   ```
   $ openstack user create --password-prompt cinder
   ```

   c. Adição do papel `admin` ao usuário `cinder`:

   ```
   $ openstack role add --project service --user cinder admin
   ```

   d. Criação das entidades do serviço `cinder`:

```
openstack service create --name cinder --description \
"OpenStack Block Storage" volume

openstack service create --name cinderv2 --description \
"OpenStack Block Storage" volumev2
```

3. Criação dos endpoints da API do serviço de armazenamento em bloco:

```
openstack endpoint create \
  --publicurl http://controller:8776/v2/%\(tenant_id\)s \
  --internalurl http://controller:8776/v2/%\(tenant_id\)s \
  --adminurl http://controller:8776/v2/%\(tenant_id\)s \
  --region RegionOne \
  volume

openstack endpoint create \
  --publicurl http://controller:8776/v2/%\(tenant_id\)s \
  --internalurl http://controller:8776/v2/%\(tenant_id\)s \
  --adminurl http://controller:8776/v2/%\(tenant_id\)s \
  --region RegionOne \
  volumev2
```

4. Instalação e configuração dos componentes para controle do serviço de armazenamento em bloco
   a. Instalação dos pacotes:

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

   b. Configuração do arquivo `/etc/cinder/cinder.conf`:

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


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

```
my_ip = 192.168.0.12

verbose = True


[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 = cinder

password = CINDER_PASS


[oslo_concurrency]

...

lock_path = /var/lock/cinder
```

*Observações:

CINDER_DBPASS foi substituído pela senha escolhida;
RABBIT_PASS foi substituído pela senha escolhida;
CINDER_PASS foi substituído pela senha escolhida;
Quaisquer outras configurações na seção `[keystone_authtoken]` devem ser comentadas ou removidas.

5. Povoamento do banco de dados `cinder`:

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

6. Finalização da instalação

   a. Reinicialização do serviço de armazenamento em bloco:

```
# service cinder-scheduler restart

# service cinder-api restart
```

b. Remoção do arquivo do banco de dados SQLite (não utilizado):

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

Instalação e configuração do *nó* de armazenamento

1. Instalações iniciais

   a. Instalação do pacote para suporte ao QEMU:
   ```
   # apt-get install quem
   ```

   b. Instalação dos pacotes LVM:
   ```
   # apt-get install lvm2
   ```

   c. Criação do volume físico LVM `/dev/sda1` (o nome do dispositivo foi ajustado de acordo com a instalação):
   ```
   # pvcreate /dev/sda1
   ```

   d. Criação do grupo de volume LVM `cinder-volumes`:
   ```
   # vgcreate cinder-volumes /dev/sda1
   ```

   e. Configuração do arquivo `/etc/lvm/lvm.conf` para escanear apenas o dispositivo `sda1`, que contém o grupo de volume `cinder-volume`:
   ```
   devices {
   ...
   filter = [ "a/sda/", "r/.*/"]
   ```

   *Observação: Nenhum outro dispositivo utiliza LVM no ambiente.

2. Instalação e configuração dos componentes para controle do serviço de armazenamento em bloco

   a. Instalação dos pacotes:
   ```
   # apt-get install cinder-volume python-mysqldb
   ```

   b. Configuração do arquivo `/etc/cinder/cinder.conf`:

   ```
   [database]

   ...

   connection = mysql://cinder:CINDER_DBPASS@controller/cinder

   [DEFAULT]

   ...

   verbose = True

   my_ip = 192.168.0.15

   rpc_backend = rabbit
   ```

```
auth_strategy = keystone

enabled_backends = lvm

glance_host = controller


[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 = cinder

password = CINDER_PASS


[lvm]

...

volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver

volume_group = cinder-volumes

iscsi_protocol = iscsi

iscsi_helper = tgtadm


[oslo_concurrency]

...

lock_path = /var/lock/cinder
```

*Observações:

CINDER_DBPASS foi substituído pela senha escolhida;

RABBIT_PASS foi substituído pela senha escolhida;
CINDER_PASS foi substituído pela senha escolhida;
Quaisquer outras configurações na seção `[keystone_authtoken]` devem ser comentadas ou removidas.

3. Finalização da instalação:

    a. Reinicialização do serviço de volume de armazenamento em bloco e suas dependências:

```
# service tgt restart

# service cinder-volume restart
```

    b. Remoção do arquivo do banco de dados SQLite (não utilizado):

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

4. Verificando a operação do serviço de armazenamento em bloco (*nó* controller)

    a. Configuração para que os clientes do serviço utilizem a versão 2.0 da API, em cada arquivo de script de ambiente para cliente:

```
$ echo "export OS_VOLUME_API_VERSION=2" | tee -a \

admin-openrc.sh demo-openrc.sh
```

    b. Obtenção das credencias `admin`:

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

    c. Listagem dos componentes dos serviços instalados:

```
$ cinder service-list
```

    d. Obtenção das credenciais `demo`:

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

    e. Criação de um volume com capacidade de 1 GB:

```
$ cinder create --name demo-volume1 1
```

    f. Verificação e disponibilidade do volume criado:

```
$ cinder list
```

POR FIM, CONSULTE O MANUAL DO OPENSTACK CITADO NO INÍCIO DESTA DOCUMENTAÇÃO PARA REALIZAR A INICIALIZAÇÃO DE UMA INSTÂNCIA NO AMBIENTE IMPLANTADO.