Installation Guide

Latest version: installation-openstack-ubuntu-note.md hosted on *Github.com/littlewey*

> ref: https://docs.openstack.org/install-guide

Ubuntu was chosen as host OS.

> "It's a good way to learn by installing it manually for as many services as you could :-) ."
>
> Wey Gu

Contents

Host networking

> ref: https://docs.openstack.org/install-guide/environment-networking.html

> ref: https://help.ubuntu.com/lts/serverguide/network-configuration.html

The example architectures assume use of the following networks:

- Management on 10.0.0.0/24 with gateway 10.0.0.1

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

- Provider on 203.0.113.0/24 with gateway 203.0.113.1

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

My network solution

```
Net0:
        Network name: VirtualBox  host-only Ethernet Adapter
        Purpose: administrator / management network
        IP block: 10.20.0.0/24
        DHCP: disable
        Linux device: eth0

Net1:
        Network name: VirtualBox  host-only Ethernet Adapter#2
        Purpose: Provider network
        DHCP: disable
        IP block: 172.16.0.0/24
        Linux device: eth1

Net2:
        Network name: VirtualBox  host-only Ethernet Adapter#3
        Purpose: Storage network
        DHCP: disable
        IP block: 192.168.199.0/24
        Linux device: eth2

Net3:
        Network name: VirtualBox  Bridged // for accessing network or remote access purpose
        Purpose: Internet
        DHCP: enable
        IP block: <depend on your network>
        Linux device: eth3
```

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

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

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

Base Machine

- download image from https://launchpad.net/ubuntu/+mirror/mirrors.neusoft.edu.cn-release
- Change root password

  ```
  $ sudo su
  # passwd
  ```

- Allow root ssh with password

```
# vi /etc/ssh/sshd_config

PermitRootLogin yes
```

- Check nic names

```
root@ubuntu:~# dmesg | grep rename
[    2.799294] e1000 0000:00:09.0 enp0s9: renamed from eth2
[    2.800192] e1000 0000:00:0a.0 enp0s10: renamed from eth3
[    2.801072] e1000 0000:00:08.0 enp0s8: renamed from eth1
[    2.804067] e1000 0000:00:03.0 enp0s3: renamed from eth0
```

- configure management network as a dummy one

```
# vi /etc/network/interfaces
auto enp0s3
iface enp0s3 inet static
address 10.20.0.11
netmask 255.255.255.0
```

- NTP
  - install chrony

    ```
    install chrony
    ```

  - Edit the `/etc/chrony/chrony.conf` file and add, change, or remove these keys as necessary for your environment:

    ```
    allow 10.20.0.0/24
    ```

  - restart service

    ```
    # service chrony restart
    ```

- Install OpenStack packages

  > ref : https://docs.openstack.org/install-guide/environment-packages.html

  Enable the OpenStack repository

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

  Upgrade the packages on all nodes:

  > Set apt proxy before doing that will help save your life

```
# vi /etc/apt/apt.conf.d/90proxy
Acquire::http::Proxy "http://www-proxy.exu.ericsson.se:8080";
Acquire::https::Proxy "http://www-proxy.exu.ericsson.se:8080";

# sed -i -e 's/cn/us/g' /etc/apt/sources.list
```

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

Install the OpenStack client:

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

Controller node actions

management network eth0 (enp0s3)

```
# vi /etc/network/interfaces

auto enp0s3
iface enp0s3 inet static
address 10.20.0.10
netmask 255.255.255.0

# ifup enp0s3
```

hostname and hosts

```
# echo controller > /etc/hostname
# echo 10.20.0.10    controller >> /etc/hosts
# echo 10.20.0.20    compute >> /etc/hosts
# hostname controller
```

SQL database

Install package

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

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

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

```
[mysqld]
bind-address = 10.20.0.10

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

restart database service

```
# service mysql restart
```

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

```
# mysql_secure_installation
```

Message queue

Install the package:

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

Add the `openstack` user:

```
# rabbitmqctl add_user openstack RABBIT_PASS

Creating user "openstack" ...
```

Replace `RABBIT_PASS` with a suitable password.

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

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

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

Memcached

Install the packages:

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

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

```
-l 10.20.0.10
```

> Change the existing line that had `-l 127.0.0.1`.

Restart the Memcached service:

```
# service memcached restart
```

Compute node actions

management network eth0 (enp0s3)

```
# vi /etc/network/interfaces

auto enp0s3
iface enp0s3 inet static
address 10.20.0.10
netmask 255.255.255.0

# ifup enp0s3
```

configure NTP by editing `/etc/chrony/chrony.conf`

```
server 10.20.0.10 iburst
```

change hostname and hosts

```
# echo conpute > /etc/hostname
# echo 10.20.0.10    controller >> /etc/hosts
# echo 10.20.0.20    compute >> /etc/hosts
# hostname compute
```

Keystone installation

> ref: https://docs.openstack.org/newton/install-guide-ubuntu/keystone.html

Before you configure the OpenStack Identity service, you must create a database and an administration token.

To create the database, complete the following actions:

- Use the database access client to connect to the database server as the `root` user:

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

  In 16.04 LTS local access need no user/psw

  ```
  # mysql
  ```

- Create the `keystone` database:

```
mysql> CREATE DATABASE keystone;
```

- Grant proper access to the `keystone` database:

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

  Replace `KEYSTONE_DBPASS` with a suitable password.

- Exit the database access client.

Run the following command to install the packages:

```
# apt install keystone -y
```

1. Edit the `/etc/keystone/keystone.conf` file and complete the following actions:
   - In the `[database]` section, configure database access:

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

     Replace `KEYSTONE_DBPASS` with the password you chose for the database.

     Comment out or remove any other `connection` options in the `[database]` section.
   - In the `[token]` section, configure the Fernet token provider:

     ```
     [token]
     ...
     provider = fernet
     ```

2. Populate the Identity service database:

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

3. Initialize Fernet key repositories:

```
# keystone-manage fernet_setup --keystone-user keystone --keystone-group keystone
# keystone-manage credential_setup --keystone-user keystone --keystone-group keystone
```

4. Bootstrap the Identity service:

```
# keystone-manage bootstrap --bootstrap-password ADMIN_PASS \
  --bootstrap-admin-url http://controller:35357/v3/ \
  --bootstrap-internal-url http://controller:35357/v3/ \
  --bootstrap-public-url http://controller:5000/v3/ \
  --bootstrap-region-id RegionOne
```

Replace `ADMIN_PASS` with a suitable password for an administrative user.

Configure the Apache HTTP server

1. Edit the `/etc/apache2/apache2.conf` file and configure the `ServerName` option to reference the controller node:

```
ServerName controller
```

Finalize the installation

1. Restart the Apache service and remove the default SQLite database:

```
# service apache2 restart
# rm -f /var/lib/keystone/keystone.db
```

1. Configure the administrative account

```
$ export OS_USERNAME=admin
$ export OS_PASSWORD=ADMIN_PASS
$ export OS_PROJECT_NAME=admin
$ export OS_USER_DOMAIN_NAME=Default
$ export OS_PROJECT_DOMAIN_NAME=Default
$ export OS_AUTH_URL=http://controller:35357/v3
$ export OS_IDENTITY_API_VERSION=3
```

Replace `ADMIN_PASS` with the password used in the `keystone-manage bootstrap` command from the section called Install and configure.

Create a domain, projects, users, and roles

The Identity service provides authentication services for each OpenStack service. The authentication service uses a combination of domains, projects, users, and roles.

1. This guide uses a service project that contains a unique user for each service that you add to your environment. Create the `service` project:

```
$ openstack project create --domain default \
  --description "Service Project" service

+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | Service Project                  |
| domain_id   | default                          |
| enabled     | True                             |
| id          | 24ac7f19cd944f4cba1d77469b2a73ed |
| is_domain   | False                            |
| name        | service                          |
| parent_id   | default                          |
+-------------+----------------------------------+
```

2. Regular (non-admin) tasks should use an unprivileged project and user. As an example, this guide creates the `demo` project and user.
   - Create the `demo` project:

     ```
     $ openstack project create --domain default \
       --description "Demo Project" demo

     +-------------+----------------------------------+
     | Field       | Value                            |
     +-------------+----------------------------------+
     | description | Demo Project                     |
     | domain_id   | default                          |
     | enabled     | True                             |
     | id          | 231ad6e7ebba47d6a1e57e1cc07ae446 |
     | is_domain   | False                            |
     | name        | demo                             |
     | parent_id   | default                          |
     +-------------+----------------------------------+
     ```

     Do not repeat this step when creating additional users for this project.

   - Create the `demo` user:

```
$ openstack user create --domain default \
  --password-prompt demo

User Password:
Repeat User Password:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | aeda23aa78f44e859900e22c24817832 |
| name                | demo                             |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

- Create the `user` role:

```
$ openstack role create user

+-----------+----------------------------------+
| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | None                             |
| id        | 997ce8d05fc143ac97d83fdfb5998552 |
| name      | user                             |
+-----------+----------------------------------+
```

- Add the `user` role to the `demo` project and user:

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

Verify operation

For security reasons, disable the temporary authentication token mechanism:

Edit the `/etc/keystone/keystone-paste.ini` file and remove `admin_token_auth` from the `[pipeline:public_api]`, `[pipeline:admin_api]`, and `[pipeline:api_v3]` sections.

Unset the temporary `OS_AUTH_URL` and `OS_PASSWORD` environment variable:

```
$ unset OS_AUTH_URL OS_PASSWORD
```

As the `admin` user, request an authentication token:

```
$ openstack --os-auth-url http://controller:35357/v3 \
  --os-project-domain-name Default --os-user-domain-name Default \
  --os-project-name admin --os-username admin token issue

Password:
+------------+-----------------------------------------------------------------+
| Field      | Value                                                           |
+------------+-----------------------------------------------------------------+
| expires    | 2017-08-12T20:14:07.056119Z                                     |
| id         | gAAAAABWvi7_B8kKQD9wdXac8MoZiQldmjEO643d-e_j-XXq9AmIegIbA7UHGPv |
|            | atnN21qtOMjCFWX7BReJEQnVOAj3nclRQgAYRsfSU_MrsuWb4EDtnjU7HEpoBb4 |
|            | o6ozsA_NmFWEpLeKy0uNn_WeKbAhYygrsmQGA49dclHVnz-OMVLiyM9ws       |
| project_id | 343d245e850143a096806dfaefa9afdc                                |
| user_id    | ac3377633149401296f6c0d92d79dc16                                |
+------------+-----------------------------------------------------------------+
```

This command uses the password for the `admin` user. As we gave above it's `ADMIN_PASS`.

As the `demo` user, request an authentication token:

```
$ openstack --os-auth-url http://controller:5000/v3 \
  --os-project-domain-name Default --os-user-domain-name Default \
  --os-project-name demo --os-username demo token issue

Password:
+------------+-----------------------------------------------------------------+
| Field      | Value                                                           |
+------------+-----------------------------------------------------------------+
| expires    | 2017-08-12T20:15:39.014479Z                                     |
| id         | gAAAAABWvi9bsh7vkiby5BpCCnc-JkbGhm9wH3fabS_cY7uabOubesi-Me6IGWW |
|            | yQqNegDDZ5jw7grI26vvgy1J5nCVwZ_zFRqPiz_qhbq29mgbQLglbkq6FQvzBRQ |
|            | JcOzq3uwhzNxszJWmzGC7rJE_H0A_a3UFhqv8M4zMRYSbS2YF0MyFmp_U       |
| project_id | ed0b60bf607743088218b0a533d5943f                                |
| user_id    | 58126687cbcc4888bfa9ab73a2256f27                                |
+------------+-----------------------------------------------------------------+
```

> This command uses the password for the `demo` user and API port 5000 which only allows regular (non-admin) access to the Identity service API.

Create OpenStack client environment scripts

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

Creating the scripts

Create client environment scripts for the `admin` and `demo` projects and users. Future portions of this guide reference these scripts to load appropriate credentials for client operations.

1. Edit the `admin-openrc` file and add the following content:

```
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=ADMIN_PASS
export OS_AUTH_URL=http://controller:35357/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
```

   Replace `ADMIN_PASS` with the password you chose for the `admin` user in the Identity service.

2. Edit the `demo-openrc` file and add the following content:

```
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=demo
export OS_USERNAME=demo
export OS_PASSWORD=demo
export OS_AUTH_URL=http://controller:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
```

   Replace `OS_PASSWORD=demo` with the password you chose for the `demo` user in the Identity service.

Using the scripts

To run clients as a specific project and user, you can simply load the associated client environment script prior to running them. For example:

1. Load the `admin-openrc` file to populate environment variables with the location of the Identity service and the `admin` project and user credentials:

```
$ . admin-openrc
```

2. Request an authentication token:

```
root@ubuntu:~# openstack token issue --max-width 70
+------------+------------------------------------------------------+
| Field      | Value                                                |
+------------+------------------------------------------------------+
| expires    | 2017-08-23T14:00:10+0000                             |
| id         | gAAAAABZnXxaKuuwh9Kw-dbY1mSn8LeNNQMKmIj2EW8jyjO0NSy5H |
|            | QPno4Tj6NqqSkumKhRZW8lPS1nZC2pm5fCuH5XMtVfJTu89RX6Sba |
|            | -vSv-OZl5uHvRY4KOK03WH15Dnp1XbWN97xY8tR_kAhc-69       |
|            | -WvDe1DLS6vKr-bKbYDVXLqlLshE8E                       |
| project_id | 78c9c849237649a3a8c4526167427589                     |
| user_id    | d8efd16c30904a7992010abe4bdb9a2b                     |
+------------+------------------------------------------------------+
```

Glance installation

ref: https://docs.openstack.org/newton/install-guide-ubuntu/glance.html

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

Before you proceed, ensure that the controller node has at least several gigabytes of space available in this directory. Keep in mind that since the `file` back end is often local to a controller node, it is not typically suitable for a multi-node glance deployment.

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

Install and configure

This section describes how to install and configure the Image service, code-named glance, on the controller node. For simplicity, this configuration stores images on the local file system.

Prerequisites

Before you install and configure the Image service, you must create a database, service credentials, and API endpoints.

1. To create the database, complete these steps:
   - Use the database access client to connect to the database server as the `root` user:

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

   - Create the `glance` database:

     ```
     mysql> CREATE DATABASE glance;
     ```

   - Grant proper access to the `glance` database:

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

     Replace `GLANCE_DBPASS` with a suitable password.
   - Exit the database access client.
2. Source the `admin` credentials to gain access to admin-only CLI commands:

   ```
   $ . admin-openrc
   ```

3. To create the service credentials, complete these steps:
   - Create the `glance` user:

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

User Password:
Repeat User Password:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | 3f4e777c4062483ab8d9edd7dff829df |
| name                | glance                           |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

- Add the `admin` role to the `glance` user and `service` project:

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

This command provides no output.

- Create the `glance` service entity:

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

+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | OpenStack Image                  |
| enabled     | True                             |
| id          | 8c2c7f1b9b5049ea9e63757b5533e6d2 |
| name        | glance                           |
| type        | image                            |
+-------------+----------------------------------+
```

4. Create the Image service API endpoints:

```
$ openstack endpoint create --region RegionOne \
  image public http://controller:9292

+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | 340be3625e9b4239a6415d034e98aace |
| interface    | public                           |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | 8c2c7f1b9b5049ea9e63757b5533e6d2 |
| service_name | glance                           |
| service_type | image                            |
| url          | http://controller:9292           |
+--------------+----------------------------------+

$ openstack endpoint create --region RegionOne \
  image internal http://controller:9292

+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | a6e4b153c2ae4c919eccfdbb7dceb5d2 |
| interface    | internal                         |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | 8c2c7f1b9b5049ea9e63757b5533e6d2 |
| service_name | glance                           |
| service_type | image                            |
| url          | http://controller:9292           |
+--------------+----------------------------------+

$ openstack endpoint create --region RegionOne \
  image admin http://controller:9292

+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | 0c37ed58103f4300a84ff125a539032d |
| interface    | admin                            |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | 8c2c7f1b9b5049ea9e63757b5533e6d2 |
| service_name | glance                           |
| service_type | image                            |
| url          | http://controller:9292           |
+--------------+----------------------------------+
```

Install and configure components

Install the packages:

```
# apt install glance -y
```

1. Edit the `/etc/glance/glance-api.conf` file and complete the following actions:
   - In the `[database]` section, configure database access:

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

     Replace `GLANCE_DBPASS` with the password you chose for the Image service database.
   - In the `[keystone_authtoken]` and `[paste_deploy]` sections, configure Identity service access:

     ```
     [keystone_authtoken]
     ...
     auth_uri = http://controller:5000
     auth_url = http://controller:35357
     memcached_servers = controller:11211
     auth_type = password
     project_domain_name = Default
     user_domain_name = Default
     project_name = service
     username = glance
     password = glance

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

     Replace `password = glance` with the password you chose for the `glance` user in the Identity service.

     Comment out or remove any other options in the `[keystone_authtoken]` section.
   - In the `[glance_store]` section, configure the local file system store and location of image files:

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

2. Edit the `/etc/glance/glance-registry.conf` file and complete the following actions:
   - In the `[database]` section, configure database access:

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

Replace `GLANCE_DBPASS` with the password you chose for the Image service database.

- In the `[keystone_authtoken]` and `[paste_deploy]` sections, configure Identity service access:

```
[keystone_authtoken]
...
auth_uri = http://controller:5000
auth_url = http://controller:35357
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = glance
password = glance

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

Replace `password = glance` with the password you chose for the `glance` user in the Identity service.

Comment out or remove any other options in the `[keystone_authtoken]` section.

Populate the Image service database:

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

Ignore any deprecation messages in this output.

Finalize installation

Restart the Image services:

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

Verify operation

Verify operation of the Image service using CirrOS, a small Linux image that helps you test your OpenStack deployment.

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

1. Source the `admin` credentials to gain access to admin-only CLI commands:

```
$ . admin-openrc
```

2. Download the source image:

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

> tip: add proxy to improve speed in office network
>
> ```
> $ export http_proxy=http://www-proxy.exu.ericsson.se:8080
>
> // after wget
>
> $ unset http_proxy
> ```

Install `wget` if your distribution does not include it.

3. Upload the image to the Image service using the QCOW2 disk format, bare container format, and public visibility so all projects can access it:

```
$ openstack image create "cirros" \
  --file cirros-0.3.4-x86_64-disk.img \
  --disk-format qcow2 --container-format bare \
  --public

+------------------+------------------------------------------------------+
| Field            | Value                                                |
+------------------+------------------------------------------------------+
| checksum         | 133eae9fb1c98f45894a4e60d8736619                     |
| container_format | bare                                                 |
| created_at       | 2015-03-26T16:52:10Z                                 |
| disk_format      | qcow2                                                |
| file             | /v2/images/cc5c6982-4910-471e-b864-1098015901b5/file |
| id               | cc5c6982-4910-471e-b864-1098015901b5                 |
| min_disk         | 0                                                    |
| min_ram          | 0                                                    |
| name             | cirros                                               |
| owner            | ae7a98326b9c455588edd2656d723b9d                     |
| protected        | False                                                |
| schema           | /v2/schemas/image                                    |
| size             | 13200896                                             |
| status           | active                                               |
| tags             |                                                      |
| updated_at       | 2015-03-26T16:52:10Z                                 |
| virtual_size     | None                                                 |
| visibility       | public                                               |
+------------------+------------------------------------------------------+
```

For information about the **openstack image create** parameters, see Create or update an image (glance) in the `OpenStack UserGuide`.

For information about disk and container formats for images, see Disk and container formats for images in the `OpenStack VirtualMachine Image Guide`.

OpenStack generates IDs dynamically, so you will see different values in the example command output.

4. Confirm upload of the image and validate attributes:

```
$ openstack image list

+--------------------------------------+--------+--------+
| ID                                   | Name   | Status |
+--------------------------------------+--------+--------+
| 38047887-61a7-41ea-9b49-27987d5e8bb9 | cirros | active |
+--------------------------------------+--------+--------+
```

Nova installation

> ref: https://docs.openstack.org/newton/install-guide-ubuntu/nova.html

Nova install and configure controller node

Prerequisites

Before you install and configure the Compute service, you must create databases, service credentials, and API endpoints.

1. To create the databases, complete these steps:
   - Use the database access client to connect to the database server as the `root` user:

     ```
     # mysql
     ```

- Create the `nova_api`, `nova`, and `nova_cell0` databases:

  ```
  MariaDB [(none)]> CREATE DATABASE nova_api;
  MariaDB [(none)]> CREATE DATABASE nova;
  MariaDB [(none)]> CREATE DATABASE nova_cell0;
  ```

- Grant proper access to the databases:

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova_api.* TO 'nova'@'localhost' \
  IDENTIFIED BY 'NOVA_DBPASS';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova_api.* TO 'nova'@'%' \
  IDENTIFIED BY 'NOVA_DBPASS';

MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' \
  IDENTIFIED BY 'NOVA_DBPASS';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' \
  IDENTIFIED BY 'NOVA_DBPASS';

MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova_cell0.* TO 'nova'@'localhost' \
  IDENTIFIED BY 'NOVA_DBPASS';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova_cell0.* TO 'nova'@'%' \
  IDENTIFIED BY 'NOVA_DBPASS';
```

Replace `NOVA_DBPASS` with a suitable password.

- Exit the database access client.

2. Source the `admin` credentials to gain access to admin-only CLI commands:

```
$ . admin-openrc
```

3. Create the Compute service credentials:
   - Create the `nova` user:

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

User Password:
Repeat User Password:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | 8a7dbf5279404537b1c7b86c033620fe |
| name                | nova                             |
| options             | {}                               |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

   - Add the `admin` role to the `nova` user:

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

     This command provides no output.

   - Create the `nova` service entity:

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

+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | OpenStack Compute                |
| enabled     | True                             |
| id          | 060d59eac51b4594815603d75a00aba2 |
| name        | nova                             |
| type        | compute                          |
+-------------+----------------------------------+
```

4. Create the Compute API service endpoints:

```
$ openstack endpoint create --region RegionOne \
  compute public http://controller:8774/v2.1

+--------------+------------------------------------------+
| Field        | Value                                    |
+--------------+------------------------------------------+
| enabled      | True                                     |
| id           | 3c1caa473bfe4390a11e7177894bcc7b         |
| interface    | public                                   |
| region       | RegionOne                                |
| region_id    | RegionOne                                |
| service_id   | 060d59eac51b4594815603d75a00aba2         |
| service_name | nova                                     |
| service_type | compute                                  |
| url          | http://controller:8774/v2.1              |
+--------------+------------------------------------------+

$ openstack endpoint create --region RegionOne \
  compute internal http://controller:8774/v2.1

+--------------+------------------------------------------+
| Field        | Value                                    |
+--------------+------------------------------------------+
| enabled      | True                                     |
| id           | e3c918de680746a586eac1f2d9bc10ab         |
| interface    | internal                                 |
| region       | RegionOne                                |
| region_id    | RegionOne                                |
| service_id   | 060d59eac51b4594815603d75a00aba2         |
| service_name | nova                                     |
| service_type | compute                                  |
| url          | http://controller:8774/v2.1              |
+--------------+------------------------------------------+

$ openstack endpoint create --region RegionOne \
  compute admin http://controller:8774/v2.1

+--------------+------------------------------------------+
| Field        | Value                                    |
+--------------+------------------------------------------+
| enabled      | True                                     |
| id           | 38f7af91666a47cfb97b4dc790b94424         |
| interface    | admin                                    |
| region       | RegionOne                                |
| region_id    | RegionOne                                |
| service_id   | 060d59eac51b4594815603d75a00aba2         |
| service_name | nova                                     |
| service_type | compute                                  |
| url          | http://controller:8774/v2.1              |
+--------------+------------------------------------------+
```

5. Create a Placement service user using your chosen `PLACEMENT_PASS`:

```
$ openstack user create --domain default --password-prompt placement

User Password:
Repeat User Password:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | fa742015a6494a949f67629884fc7ec8 |
| name                | placement                        |
| options             | {}                               |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

6. Add the Placement user to the service project with the admin role:

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

   This command provides no output.

7. Create the Placement API entry in the service catalog:

```
$ openstack service create --name placement --description "Placement API" placement
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | Placement API                    |
| enabled     | True                             |
| id          | 2d1a27022e6e4185b86adac4444c495f |
| name        | placement                        |
| type        | placement                        |
+-------------+----------------------------------+
```

8. Create the Placement API service endpoints:

```
$ openstack endpoint create --region RegionOne placement public http://controller:8778
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | 2b1b2637908b4137a9c2e0470487cbc0 |
| interface    | public                           |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | 2d1a27022e6e4185b86adac4444c495f |
| service_name | placement                        |
| service_type | placement                        |
| url          | http://controller:8778           |
+--------------+----------------------------------+

$ openstack endpoint create --region RegionOne placement internal http://controller:8778
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | 02bcda9a150a4bd7993ff4879df971ab |
| interface    | internal                         |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | 2d1a27022e6e4185b86adac4444c495f |
| service_name | placement                        |
| service_type | placement                        |
| url          | http://controller:8778           |
+--------------+----------------------------------+

$ openstack endpoint create --region RegionOne placement admin http://controller:8778
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | 3d71177b9e0f406f98cbff198d74b182 |
| interface    | admin                            |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | 2d1a27022e6e4185b86adac4444c495f |
| service_name | placement                        |
| service_type | placement                        |
| url          | http://controller:8778           |
+--------------+----------------------------------+
```

Install and configure components

Default configuration files vary by distribution. You might need to add these sections and options rather than modifying existing sections and options. Also, an ellipsis (`...`) in the configuration snippets indicates potential default configuration options that you should retain.

1. Install the packages:

```
# apt install nova-api nova-conductor nova-consoleauth \
  nova-novncproxy nova-scheduler nova-placement-api
```

1. Edit the `/etc/nova/nova.conf` file and complete the following actions:
   - In the `[api_database]` and `[database]` sections, configure database access:

     ```
     [api_database]
     # ...
     connection = mysql+pymysql://nova:NOVA_DBPASS@controller/nova_api

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

     Replace `NOVA_DBPASS` with the password you chose for the Compute databases.
   - In the `[DEFAULT]` section, configure `RabbitMQ` message queue access:

     ```
     [DEFAULT]
     # ...
     transport_url = rabbit://openstack:RABBIT_PASS@controller
     ```

     Replace `RABBIT_PASS` with the password you chose for the `openstack` account in `RabbitMQ`.
   - In the `[api]` and `[keystone_authtoken]` sections, configure Identity service access:

     ```
     [api]
     # ...
     auth_strategy = keystone

     [keystone_authtoken]
     # ...
     auth_uri = http://controller:5000
     auth_url = http://controller:35357
     memcached_servers = controller:11211
     auth_type = password
     project_domain_name = default
     user_domain_name = default
     project_name = service
     username = nova
     password = nova
     ```

     Replace `nova` with the password you chose for the `nova` user in the Identity service.

     Comment out or remove any other options in the `[keystone_authtoken]` section.
   - In the `[DEFAULT]` section, configure the `my_ip` option to use the management interface IP address of the controller node:

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

- In the `[DEFAULT]` section, enable support for the Networking service:

  ```
  [DEFAULT]
  # ...
  use_neutron = True
  firewall_driver = nova.virt.firewall.NoopFirewallDriver
  ```

  By default, Compute uses an internal firewall driver. Since the Networking service includes a firewall driver, you must disable the Compute firewall driver by using the `nova.virt.firewall.NoopFirewallDriver` firewall driver.

- In the `[vnc]` section, configure the VNC proxy to use the management interface IP address of the controller node:

  ```
  [vnc]
  enabled = true
  # ...
  vncserver_listen = $my_ip
  vncserver_proxyclient_address = $my_ip
  ```

- In the `[glance]` section, configure the location of the Image service API:

  ```
  [glance]
  # ...
  api_servers = http://controller:9292
  ```

- In the `[oslo_concurrency]` section, configure the lock path:

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

- Due to a packaging bug, remove the `log_dir` option from the `[DEFAULT]` section.

- In the `[placement]` section, configure the Placement API:

  ```
  [placement]
  # ...
  os_region_name = RegionOne
  project_domain_name = Default
  project_name = service
  auth_type = password
  user_domain_name = Default
  auth_url = http://controller:35357/v3
  username = placement
  password = PLACEMENT_PASS
  ```

Replace `PLACEMENT_PASS` with the password you choose for the `placement` user in the Identity service. Comment out any other options in the `[placement]` section.

1. Populate the nova-api database:

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

Ignore any deprecation messages in this output.

2. Register the `cell0` database:

```
# su -s /bin/sh -c "nova-manage cell_v2 map_cell0" nova
```

3. Create the `cell1` cell:

```
# su -s /bin/sh -c "nova-manage cell_v2 create_cell --name=cell1 --verbose" nova
109e1d4b-536a-40d0-83c6-5f121b82b650
```

4. Populate the nova database:

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

5. Verify nova cell0 and cell1 are registered correctly:

```
# nova-manage cell_v2 list_cells
+-------+--------------------------------------+
| Name  | UUID                                 |
+-------+--------------------------------------+
| cell1 | 109e1d4b-536a-40d0-83c6-5f121b82b650 |
| cell0 | 00000000-0000-0000-0000-000000000000 |
+-------+--------------------------------------+
```

Finalize installation

- Restart the Compute services:

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

Nova Install and configure a compute node

This section describes how to install and configure the Compute service on a compute node. The service supports several hypervisors to deploy instances or VMs. For simplicity, this configuration uses the QEMU hypervisor with the KVM extension on compute nodes that support hardware acceleration for virtual machines.

Install and configure components

1. Install the packages:

```
# apt install nova-compute
```

1. Edit the `/etc/nova/nova.conf` file and complete the following actions:
   - In the `[DEFAULT]` section, configure `RabbitMQ` message queue access:

     ```
     [DEFAULT]
     ...
     transport_url = rabbit://openstack:RABBIT_PASS@controller
     ```

     Replace `RABBIT_PASS` with the password you chose for the `openstack` account in `RabbitMQ`.
   - In the `[DEFAULT]` and `[keystone_authtoken]` sections, configure Identity service access:

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

     [keystone_authtoken]
     ...
     auth_uri = http://controller:5000
     auth_url = http://controller:35357
     memcached_servers = controller:11211
     auth_type = password
     project_domain_name = Default
     user_domain_name = Default
     project_name = service
     username = nova
     password = nova
     ```

     Replace `password = nova` with the password you chose for the `nova` user in the Identity service.

     Comment out or remove any other options in the `[keystone_authtoken]` section.

- In the `[DEFAULT]` section, configure the `my_ip` option:

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

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

here our compute is 10.20.0.20

- In the `[DEFAULT]` section, enable support for the Networking service:

```
[DEFAULT]
...
use_neutron = True
firewall_driver = nova.virt.firewall.NoopFirewallDriver
```

By default, Compute uses an internal firewall service. Since Networking includes a firewall service, you must disable the Compute firewall service by using the `nova.virt.firewall.NoopFirewallDriver` firewall driver.

- In the `[vnc]` section, enable and configure remote console access:

```
[vnc]
...
enabled = True
vncserver_listen = 0.0.0.0
vncserver_proxyclient_address = $my_ip
novncproxy_base_url = http://controller:6080/vnc_auto.html
```

The server component listens on all IP addresses and the proxy component only listens on the management interface IP address of the compute node.

The base URL indicates the location where you can use a web browser to access remote consoles of instances on this compute node.

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

- In the `[glance]` section, configure the location of the Image service API:

```
[glance]
...
api_servers = http://controller:9292
```

- In the `[oslo_concurrency]` section, configure the lock path:

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

- Due to a packaging bug, remove the `log-dir` option from the `[DEFAULT]` section.
- In the `[placement]` section, configure the Placement API:

```
[placement]
# ...
os_region_name = RegionOne
project_domain_name = Default
project_name = service
auth_type = password
user_domain_name = Default
auth_url = http://controller:35357/v3
username = placement
password = placement
```

Replace `placement` with the password you choose for the `placement` user in the Identity service. Comment out any other options in the `[placement]` section.

Finalize installation

Determine whether your compute node supports hardware acceleration for virtual machines:

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

If this command returns a value of `one or greater`, your compute node supports hardware acceleration which typically requires no additional configuration.

If this command returns a value of `zero`, your compute node does not support hardware acceleration and you must configure `libvirt` to use QEMU instead of KVM.

- Edit the `[libvirt]` section in the `/etc/nova/nova-compute.conf` file as follows:

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

Restart the Compute service:

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

Add the compute node to the cell database

Run the following commands on the **controller** node.

1. Source the admin credentials to enable admin-only CLI commands, then confirm there are compute hosts in the database:

```
$ . admin-openrc

$ openstack hypervisor list
+----+---------------------+-----------------+-----------+-------+
| ID | Hypervisor Hostname | Hypervisor Type | Host IP   | State |
+----+---------------------+-----------------+-----------+-------+
|  1 | compute1            | QEMU            | 10.0.0.31 | up    |
+----+---------------------+-----------------+-----------+-------+
```

2. Discover compute hosts:

```
# su -s /bin/sh -c "nova-manage cell_v2 discover_hosts --verbose" nova

Found 2 cell mappings.
Skipping cell0 since it does not contain hosts.
Getting compute nodes from cell 'cell1': ad5a5985-a719-4567-98d8-8d148aaae4bc
Found 1 computes in cell: ad5a5985-a719-4567-98d8-8d148aaae4bc
Checking host mapping for compute host 'compute': fe58ddc1-1d65-4f87-9456-bc040dc106b3
Creating host mapping for compute host 'compute': fe58ddc1-1d65-4f87-9456-bc040dc106b3
```

When you add new compute nodes, you must run `nova-manage cell_v2 discover_hosts` on the controller node to register those new compute nodes. Alternatively, you can set an appropriate interval in `/etc/nova/nova.conf`:

```
[scheduler]
discover_hosts_in_cells_interval = 300
```

Neutron installation

> ref: https://docs.openstack.org/newton/install-guide-ubuntu/neutron.html

This chapter explains how to install and configure the Networking service (neutron) using the provider networks.

For more information about the Networking service including virtual networking components, layout, and traffic flows, see the OpenStack Networking Guide.

Neutron Install and configure controller node

Prerequisites

Before you configure the OpenStack Networking (neutron) service, you must create a database, service credentials, and API endpoints.

1. To create the database, complete these steps:
   - Use the database access client to connect to the database server as the `root` user:

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

- Create the `neutron` database:

```
mysql> CREATE DATABASE neutron;
```

- Grant proper access to the `neutron` database, replacing `NEUTRON_DBPASS` with a suitable password:

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

- Exit the database access client.

2. Source the `admin` credentials to gain access to admin-only CLI commands:

```
$ . admin-openrc
```

3. To create the service credentials, complete these steps:
   - Create the `neutron` user:

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

User Password:
Repeat User Password:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | 319f34694728440eb8ffcb27b6dd8b8a |
| name                | neutron                          |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

   - Add the `admin` role to the `neutron` user:

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

     This command provides no output.

   - Create the `neutron` service entity:

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

+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | OpenStack Networking             |
| enabled     | True                             |
| id          | f71529314dab4a4d8eca427e701d209e |
| name        | neutron                          |
| type        | network                          |
+-------------+----------------------------------+
```

4. Create the Networking service API endpoints:

```
$ openstack endpoint create --region RegionOne \
  network public http://controller:9696

+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | 85d80a6d02fc4b7683f611d7fc1493a3 |
| interface    | public                           |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | f71529314dab4a4d8eca427e701d209e |
| service_name | neutron                          |
| service_type | network                          |
| url          | http://controller:9696           |
+--------------+----------------------------------+

$ openstack endpoint create --region RegionOne \
  network internal http://controller:9696

+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | 09753b537ac74422a68d2d791cf3714f |
| interface    | internal                         |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | f71529314dab4a4d8eca427e701d209e |
| service_name | neutron                          |
| service_type | network                          |
| url          | http://controller:9696           |
+--------------+----------------------------------+

$ openstack endpoint create --region RegionOne \
  network admin http://controller:9696

+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | 1ee14289c9374dffb5db92a5c112fc4e |
| interface    | admin                            |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | f71529314dab4a4d8eca427e701d209e |
| service_name | neutron                          |
| service_type | network                          |
| url          | http://controller:9696           |
+--------------+----------------------------------+
```

Configure networking options

You can deploy the Networking service using one of two architectures represented by options 1 and 2.

Option 1 deploys the simplest possible architecture that only supports attaching instances to provider (external) networks. No self-service (private) networks, routers, or floating IP addresses. Only the `admin` or other privileged user can manage provider networks.

- Networking Option 1: Provider networks

Here we choose Option 1.

Networking Option 1: Provider networks

Install and configure the Networking components on the *controller* node.

Install the components

```
# apt install neutron-server neutron-plugin-ml2 \
  neutron-linuxbridge-agent neutron-dhcp-agent \
  neutron-metadata-agent -y
```

Configure the server component

The Networking server component configuration includes the database, authentication mechanism, message queue, topology change notifications, and plug-in.

- Edit the `/etc/neutron/neutron.conf` file and complete the following actions:
  - In the `[database]` section, configure database access:

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

    Replace `NEUTRON_DBPASS` with the password you chose for the database.

    Comment out or remove any other `connection` options in the `[database]` section.
  - In the `[DEFAULT]` section, enable the Modular Layer 2 (ML2) plug-in and disable additional plug-ins:

    ```
    [DEFAULT]
    ...
    core_plugin = ml2
    service_plugins =
    ```

  - In the `[DEFAULT]` section, configure `RabbitMQ` message queue access:

    ```
    [DEFAULT]
    ...
    transport_url = rabbit://openstack:RABBIT_PASS@controller
    ```

    Replace `RABBIT_PASS` with the password you chose for the `openstack` account in RabbitMQ.

- In the `[DEFAULT]` and `[keystone_authtoken]` sections, configure Identity service access:

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

[keystone_authtoken]
...
auth_uri = http://controller:5000
auth_url = http://controller:35357
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = neutron
password = neutron
```

  Replace `password = neutron` with the password you chose for the `neutron` user in the Identity service.

  Comment out or remove any other options in the `[keystone_authtoken]` section.

- In the `[DEFAULT]` and `[nova]` sections, configure Networking to notify Compute of network topology changes:

```
[DEFAULT]
...
notify_nova_on_port_status_changes = True
notify_nova_on_port_data_changes = True

[nova]
...
auth_url = http://controller:35357
auth_type = password
project_domain_name = Default
user_domain_name = Default
region_name = RegionOne
project_name = service
username = nova
password = nova
```

  Replace `password = nova` with the password you chose for the `nova` user in the Identity service.

Configure the Modular Layer 2 (ML2) plug-in

The ML2 plug-in uses the Linux bridge mechanism to build layer-2 (bridging and switching) virtual networking infrastructure for instances.

- Edit the `/etc/neutron/plugins/ml2/ml2_conf.ini` file and complete the following actions:
  - In the `[ml2]` section, enable flat and VLAN networks:

```
[ml2]
...
type_drivers = flat,vlan
```

- In the `[ml2]` section, disable self-service networks:

```
[ml2]
...
tenant_network_types =
```

- In the `[ml2]` section, enable the Linux bridge mechanism:

```
[ml2]
...
mechanism_drivers = linuxbridge
```

  After you configure the ML2 plug-in, removing values in the `type_drivers` option can lead to database inconsistency.

- In the `[ml2]` section, enable the port security extension driver:

```
[ml2]
...
extension_drivers = port_security
```

- In the `[ml2_type_flat]` section, configure the provider virtual network as a flat network:

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

- In the `[securitygroup]` section, enable ipset to increase efficiency of security group rules:

```
[securitygroup]
...
enable_ipset = True
```

Configure the Linux bridge agent

The Linux bridge agent builds layer-2 (bridging and switching) virtual networking infrastructure for instances and handles security groups.

- Edit the `/etc/neutron/plugins/ml2/linuxbridge_agent.ini` file and complete the following actions:
  - In the `[linux_bridge]` section, map the provider virtual network to the provider physical network interface:

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

Replace `PROVIDER_INTERFACE_NAME` with the name of the underlying provider physical network interface. See Host networking for more information.

in our case it is: enp0s10, the bridged nic of controller network.

- In the `[vxlan]` section, disable VXLAN overlay networks:

```
[vxlan]
enable_vxlan = False
```

- In the `[securitygroup]` section, enable security groups and configure the Linux bridge iptables firewall driver:

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

Configure the DHCP agent

The DHCP agent provides DHCP services for virtual networks.

- Edit the `/etc/neutron/dhcp_agent.ini` file and complete the following actions:
  - In the `[DEFAULT]` section, configure the Linux bridge interface driver, Dnsmasq DHCP driver, and enable isolated metadata so instances on provider networks can access metadata over the network:

```
[DEFAULT]
...
interface_driver = neutron.agent.linux.interface.BridgeInterfaceDriver
dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
enable_isolated_metadata = True
```

Return to Networking controller node configuration.

Configure the metadata agent

The metadata agent provides configuration information such as credentials to instances.

- Edit the `/etc/neutron/metadata_agent.ini` file and complete the following actions:
  - In the `[DEFAULT]` section, configure the metadata host and shared secret:

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

Replace `METADATA_SECRET` with a suitable secret for the metadata proxy.

Configure the Compute service to use the Networking service

- Edit the `/etc/nova/nova.conf` file and perform the following actions:
  - In the `[neutron]` section, configure access parameters, enable the metadata proxy, and configure the secret:

    ```
    [neutron]
    ...
    url = http://controller:9696
    auth_url = http://controller:35357
    auth_type = password
    project_domain_name = Default
    user_domain_name = Default
    region_name = RegionOne
    project_name = service
    username = neutron
    password = neutron
    service_metadata_proxy = True
    metadata_proxy_shared_secret = METADATA_SECRET
    ```

    Replace `password = neutron` with the password you chose for the `neutron` user in the Identity service.

    Replace `METADATA_SECRET` with the secret you chose for the metadata proxy.

Finalize installation

1. Populate the database:

   ```
   # 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
   ```

   Database population occurs later for Networking because the script requires complete server and plug-in configuration files.

2. Restart the Compute API service:

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

3. Restart the Networking services.

   For both networking options:

   ```
   # service neutron-server restart
   # service neutron-linuxbridge-agent restart
   # service neutron-dhcp-agent restart
   # service neutron-metadata-agent restart
   ```

   For networking option 2, also restart the layer-3 service:

```
# service neutron-l3-agent restart
```

Verify operation

```
root@controller:~# openstack network agent list --max-width 50
+----------+------------+----------+-------------------+-------+-------+----------+
| ID       | Agent Type | Host     | Availability Zone | Alive | State | Binary   |
+----------+------------+----------+-------------------+-------+-------+----------+
| 1d661145 | Linux      | controll | None              | True  | UP    | neutron- |
| -0941    | bridge     | er       |                   |       |       | linuxbri |
| -411d-9b | agent      |          |                   |       |       | dge-     |
| 18-b3371 |            |          |                   |       |       | agent    |
| fe57c4b  |            |          |                   |       |       |          |
| 7502e1a3 | DHCP agent | controll | nova              | True  | UP    | neutron- |
| -998d-   |            | er       |                   |       |       | dhcp-    |
| 4aca-91e |            |          |                   |       |       | agent    |
| 4-ca17e1 |            |          |                   |       |       |          |
| b10c82   |            |          |                   |       |       |          |
| 7c47ac70 | Metadata   | controll | None              | True  | UP    | neutron- |
| -5de2-44 | agent      | er       |                   |       |       | metadata |
| 42-8fc1- |            |          |                   |       |       | -agent   |
| 91fe97ae |            |          |                   |       |       |          |
| 120f     |            |          |                   |       |       |          |
+----------+------------+----------+-------------------+-------+-------+----------+
```

Neutron Install and configure compute node

The compute node handles connectivity and [security groups] for instances.

Install the components

```
# apt install neutron-linuxbridge-agent -y
```

Configure the common component

The Networking common component configuration includes the authentication mechanism, message queue, and plug-in.

- Edit the `/etc/neutron/neutron.conf` file and complete the following actions:
  - In the `[database]` section, comment out any `connection` options because compute nodes do not directly access the database.
  - In the `[DEFAULT]` section, configure `RabbitMQ` message queue access:

    ```
    [DEFAULT]
    ...
    transport_url = rabbit://openstack:RABBIT_PASS@controller
    ```

    Replace `RABBIT_PASS` with the password you chose for the `openstack` account in RabbitMQ.

- In the `[DEFAULT]` and `[keystone_authtoken]` sections, configure Identity service access:

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

[keystone_authtoken]
...
auth_uri = http://controller:5000
auth_url = http://controller:35357
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = neutron
password = neutron
```

  Replace `password = neutron` with the password you chose for the `neutron` user in the Identity service.

  Comment out or remove any other options in the `[keystone_authtoken]` section.

Configure networking options

Choose the same networking option that you chose for the controller node to configure services specific to it. Afterwards, return here and proceed to Configure the Compute service to use the Networking service.

- Networking Option 1: Provider networks

Configure the Linux bridge agent

The Linux bridge agent builds layer-2 (bridging and switching) virtual networking infrastructure for instances and handles security groups.

- Edit the `/etc/neutron/plugins/ml2/linuxbridge_agent.ini` file and complete the following actions:
  - In the `[linux_bridge]` section, map the provider virtual network to the provider physical network interface:

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

    Replace `PROVIDER_INTERFACE_NAME` with the name of the underlying provider physical network interface. See Host networking for more information.

  - In the `[vxlan]` section, disable VXLAN overlay networks:

```
[vxlan]
enable_vxlan = False
```

  - In the `[securitygroup]` section, enable security groups and configure the Linux bridge iptables firewall driver:

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

Return to Networking compute node configuration.

Configure the Compute service to use the Networking service

- Edit the `/etc/nova/nova.conf` file and complete the following actions:
  - In the `[neutron]` section, configure access parameters:

```
[neutron]
...
url = http://controller:9696
auth_url = http://controller:35357
auth_type = password
project_domain_name = Default
user_domain_name = Default
region_name = RegionOne
project_name = service
username = neutron
password = neutron
```

    Replace `password = neutron` with the password you chose for the `neutron` user in the Identity service.

Finalize installation

1. Restart the Compute service:

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

2. Restart the Linux bridge agent:

```
# service neutron-linuxbridge-agent restart
```

Verify operation

Perform these commands on the controller node.

1. Source the `admin` credentials to gain access to admin-only CLI commands:

```
$ . admin-openrc
```

2. List loaded extensions to verify successful launch of the `neutron-server` process:

```
$ neutron ext-list

+---------------------------+-----------------------------------------------+
| alias                     | name                                          |
+---------------------------+-----------------------------------------------+
| default-subnetpools       | Default Subnetpools                           |
| network-ip-availability   | Network IP Availability                       |
| network_availability_zone | Network Availability Zone                     |
| auto-allocated-topology   | Auto Allocated Topology Services              |
| ext-gw-mode               | Neutron L3 Configurable external gateway mode |
| binding                   | Port Binding                                  |
| agent                     | agent                                         |
| subnet_allocation         | Subnet Allocation                             |
| l3_agent_scheduler        | L3 Agent Scheduler                            |
| tag                       | Tag support                                   |
| external-net              | Neutron external network                      |
| net-mtu                   | Network MTU                                   |
| availability_zone         | Availability Zone                             |
| quotas                    | Quota management support                      |
| l3-ha                     | HA Router extension                           |
| flavors                   | Neutron Service Flavors                       |
| provider                  | Provider Network                              |
| multi-provider            | Multi Provider Network                        |
| address-scope             | Address scope                                 |
| extraroute                | Neutron Extra Route                           |
| timestamp_core            | Time Stamp Fields addition for core resources |
| router                    | Neutron L3 Router                             |
| extra_dhcp_opt            | Neutron Extra DHCP opts                       |
| dns-integration           | DNS Integration                               |
| security-group            | security-group                                |
| dhcp_agent_scheduler      | DHCP Agent Scheduler                          |
| router_availability_zone  | Router Availability Zone                      |
| rbac-policies             | RBAC Policies                                 |
| standard-attr-description | standard-attr-description                     |
| port-security             | Port Security                                 |
| allowed-address-pairs     | Allowed Address Pairs                         |
| dvr                       | Distributed Virtual Router                    |
+---------------------------+-----------------------------------------------+
```

3. List agents to verify successful launch of the neutron agents:

```
$ openstack network agent list

root@controller:~# openstack network agent list --max-width 70
+----------+------------+----------+-------------------+-------+-------+------------+
| ID       | Agent Type | Host     | Availability Zone | Alive | State | Binary     |
+----------+------------+----------+-------------------+-------+-------+------------+
| 143d7731 | Linux      | compute  | None              | True  | UP    | neutron-li |
| -9227-4b | bridge     |          |                   |       |       | nuxbridge- |
| af-9052- | agent      |          |                   |       |       | agent      |
| 292d7aea |            |          |                   |       |       |            |
| 6992     |            |          |                   |       |       |            |
| 1d661145 | Linux      | controll | None              | True  | UP    | neutron-li |
| -0941    | bridge     | er       |                   |       |       | nuxbridge- |
| -411d-9b | agent      |          |                   |       |       | agent      |
| 18-b3371 |            |          |                   |       |       |            |
| fe57c4b  |            |          |                   |       |       |            |
| 7502e1a3 | DHCP agent | controll | nova              | True  | UP    | neutron-   |
| -998d-   |            | er       |                   |       |       | dhcp-agent |
| 4aca-91e |            |          |                   |       |       |            |
| 4-ca17e1 |            |          |                   |       |       |            |
| b10c82   |            |          |                   |       |       |            |
| 7c47ac70 | Metadata   | controll | None              | True  | UP    | neutron-   |
| -5de2-44 | agent      | er       |                   |       |       | metadata-  |
| 42-8fc1- |            |          |                   |       |       | agent      |
| 91fe97ae |            |          |                   |       |       |            |
| 120f     |            |          |                   |       |       |            |
+----------+------------+----------+-------------------+-------+-------+------------+
```

The output should indicate three agents on the controller node and one agent on each compute node.

Congratulations! Let's try booting an instance

Create provider network/subnetwork

> ref: https://docs.openstack.org/newton/install-guide-ubuntu/launch-instance-networks-provider.html

> The --provider:physical_network provider and --provider:network_type flat options connect the flat virtual network to the flat (native/untagged) physical network on the eth1 interface on the host

标注: 下边的创建网络里， 参数:

```
--provider-network-type flat \
--provider-physical-network provider
```

对应的是:

```
/etc/neutron/plugins/ml2/ml2_conf.ini
[ml2_type_flat]
flat_networks = provider


/etc/neutron/plugins/ml2/linuxbridge_agent.ini
[linux_bridge]
physical_interface_mappings = provider:enp0s10
```

```
root@controller:~# . admin-openrc
root@controller:~# openstack network create  --share --external \
>   --provider-physical-network provider \
>   --provider-network-type flat provider
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | UP                                   |
| availability_zone_hints   |                                      |
| availability_zones        |                                      |
| created_at                | 2017-08-23T17:14:21Z                 |
| description               |                                      |
| dns_domain                | None                                 |
| id                        | 2a33434f-ba29-4645-9b5d-24f1509066f1 |
| ipv4_address_scope        | None                                 |
| ipv6_address_scope        | None                                 |
| is_default                | None                                 |
| mtu                       | 1500                                 |
| name                      | provider                             |
| port_security_enabled     | True                                 |
| project_id                | 78c9c849237649a3a8c4526167427589     |
| provider:network_type     | flat                                 |
| provider:physical_network | provider                             |
| provider:segmentation_id  | None                                 |
| qos_policy_id             | None                                 |
| revision_number           | 4                                    |
| router:external           | External                             |
| segments                  | None                                 |
| shared                    | True                                 |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| updated_at                | 2017-08-23T17:14:21Z                 |
+---------------------------+--------------------------------------+
root@controller:~# neutron net-list
neutron CLI is deprecated and will be removed in the future. Use openstack CLI instead.
+--------------------------------------+----------+----------------------------------+----------+
| id                                   | name     | tenant_id                        | subnets |
+--------------------------------------+----------+----------------------------------+----------+
| 2a33434f-ba29-4645-9b5d-24f1509066f1 | provider | 78c9c849237649a3a8c4526167427589 |         |
+--------------------------------------+----------+----------------------------------+----------+
root@controller:~# openstack network list
+--------------------------------------+----------+---------+
| ID                                   | Name     | Subnets |
+--------------------------------------+----------+---------+
| 2a33434f-ba29-4645-9b5d-24f1509066f1 | provider |         |
+--------------------------------------+----------+---------+

root@controller:~# openstack subnet create --network provider \
>   --allocation-pool start=146.11.41.230,end=146.11.41.233 \
>   --dns-nameserver 147.128.5.12 --gateway 146.11.40.1 \
>   --subnet-range 146.11.40.1/23 provider
+-------------------+--------------------------------------+
| Field             | Value                                |
```

```
+-------------------+--------------------------------------+
| allocation_pools  | 146.11.41.230-146.11.41.233          |
| cidr              | 146.11.40.0/23                       |
| created_at        | 2017-08-23T17:17:54Z                 |
| description       |                                      |
| dns_nameservers   | 147.128.5.12                         |
| enable_dhcp       | True                                 |
| gateway_ip        | 146.11.40.1                          |
| host_routes       |                                      |
| id                | 9b118521-59b5-40ee-a439-9d59c3b392ea |
| ip_version        | 4                                    |
| ipv6_address_mode | None                                 |
| ipv6_ra_mode      | None                                 |
| name              | provider                             |
| network_id        | 2a33434f-ba29-4645-9b5d-24f1509066f1 |
| project_id        | 78c9c849237649a3a8c4526167427589     |
| revision_number   | 2                                    |
| segment_id        | None                                 |
| service_types     |                                      |
| subnetpool_id     | None                                 |
| updated_at        | 2017-08-23T17:17:54Z                 |
+-------------------+--------------------------------------+
```

Create flavor

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

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

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

Add security group rules

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

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

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

+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| created_at        | 2016-10-05T09:52:31Z                 |
| description       |                                      |
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| headers           |                                      |
| id                | 6ee8d630-9803-4d3d-9aea-8c795abbedc2 |
| port_range_max    | None                                 |
| port_range_min    | None                                 |
| project_id        | 77ae8d7104024123af342ffb0a6f1d88     |
| project_id        | 77ae8d7104024123af342ffb0a6f1d88     |
| protocol          | icmp                                 |
| remote_group_id   | None                                 |
| remote_ip_prefix  | 0.0.0.0/0                            |
| revision_number   | 1                                    |
| security_group_id | 4ceee3d4-d2fe-46c1-895c-382033e87b0d |
| updated_at        | 2016-10-05T09:52:31Z                 |
+-------------------+--------------------------------------+
```

- Permit secure shell (SSH) access:

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

+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| created_at        | 2016-10-05T09:54:50Z                 |
| description       |                                      |
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| headers           |                                      |
| id                | 3cd0a406-43df-4741-ab29-b5e7dcb7469d |
| port_range_max    | 22                                   |
| port_range_min    | 22                                   |
| project_id        | 77ae8d7104024123af342ffb0a6f1d88     |
| project_id        | 77ae8d7104024123af342ffb0a6f1d88     |
| protocol          | tcp                                  |
| remote_group_id   | None                                 |
| remote_ip_prefix  | 0.0.0.0/0                            |
| revision_number   | 1                                    |
| security_group_id | 4ceee3d4-d2fe-46c1-895c-382033e87b0d |
| updated_at        | 2016-10-05T09:54:50Z                 |
+-------------------+--------------------------------------+
```

Launch an instance

ref: Launch an instance on the provider network

Determine instance options

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

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

   ```
   $ . demo-openrc
   ```

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

   List available flavors:

   ```
   $ openstack flavor list

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

   You can also reference a flavor by ID.

3. List available images:

   ```
   $ openstack image list

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

   This instance uses the `cirros` image.

4. List available networks:

   ```
   $ openstack network list

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

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

5. List available security groups:

```
$ openstack security group list

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

This instance uses the `default` security group.

Launch the instance

1. Launch the instance:

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

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

```
root@controller:~# openstack server create --flavor m1.nano --image cirros \
>   --nic net-id=2a33434f-ba29-4645-9b5d-24f1509066f1 --security-group default provider-
instance
+-----------------------------+------------------------------------------------+
| Field                       | Value                                          |
+-----------------------------+------------------------------------------------+
| OS-DCF:diskConfig           | MANUAL                                         |
| OS-EXT-AZ:availability_zone |                                                |
| OS-EXT-STS:power_state      | NOSTATE                                        |
| OS-EXT-STS:task_state       | scheduling                                     |
| OS-EXT-STS:vm_state         | building                                       |
| OS-SRV-USG:launched_at      | None                                           |
| OS-SRV-USG:terminated_at    | None                                           |
| accessIPv4                  |                                                |
| accessIPv6                  |                                                |
| addresses                   |                                                |
| adminPass                   | MnjXdXf3qHia                                   |
| config_drive                |                                                |
| created                     | 2017-08-23T17:29:04Z                           |
| flavor                      | m1.nano (0)                                    |
| hostId                      |                                                |
| id                          | 02f54ef9-e867-4c1a-88f9-8eddd144da6f           |
| image                       | cirros (c17e391e-93e1-4480-9cf3-bf8623063e61) |
| key_name                    | None                                           |
| name                        | provider-instance                              |
| progress                    | 0                                              |
| project_id                  | cb015df53fb34d90b077e4c36ce35826               |
| properties                  |                                                |
| security_groups             | name='default'                                 |
| status                      | BUILD                                          |
| updated                     | 2017-08-23T17:29:05Z                           |
| user_id                     | cb98fad69e84459bb48f42130d5c0ce5               |
| volumes_attached            |                                                |
+-----------------------------+------------------------------------------------+
```

2. Check the status of your instance:

```
root@controller:~# nova list
+--------------------------------------+-------------------+--------+------------+----------
---+----------+
| ID                                   | Name              | Status | Task State | Power
State | Networks |
+--------------------------------------+-------------------+--------+------------+----------
---+----------+
| 02f54ef9-e867-4c1a-88f9-8eddd144da6f | provider-instance | BUILD  | scheduling | NOSTATE
|          |
+--------------------------------------+-------------------+--------+------------+----------
---+----------+

root@controller:~# openstack server list
+--------------------------------------+-------------------+--------+----------+------------
+
| ID                                   | Name              | Status | Networks | Image Name
|
+--------------------------------------+-------------------+--------+----------+------------
+
| 02f54ef9-e867-4c1a-88f9-8eddd144da6f | provider-instance | BUILD  |          | cirros
|
+--------------------------------------+-------------------+--------+----------+------------
+
```

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

Access the instance using the virtual console

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

```
$ openstack console url show provider-instance

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

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

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

2. Verify access to the provider physical network gateway:

```
$ ping -c 4 203.0.113.1

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

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

3. Verify access to the internet:

```
$ ping -c 4 openstack.org

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

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

Access the instance remotely

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

```
$ ping -c 4 203.0.113.103

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

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

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

```
$ ssh cirros@203.0.113.103

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

If your instance does not launch or seem to work as you expect, see the Instance Boot Failures section in OpenStack Operations Guide for more information or use one of the *many other options* to seek assistance. We want your first installation to work!

Return to Launch an instance.

[ISSUE] DHCP failure in VM troubleshooting

> ref: https://docs.openstack.org/neutron/pike/admin/intro-basic-networking.html

what it is like

in VM console (initial dhcp discover)

```
$ ifup eth0
udhcpc (v1.20.1) started
Sending discover...
Sending discover...
Sending discover...
Usage: /sbin/cirros-dhcpc <up|down>
No lease, failing
```

in controller console (monitor log)

```
root@controller:~# tail -f /var/log/syslog
Aug 24 03:06:30 controller dhclient[1166]: DHCPREQUEST of 146.11.41.129 on enp0s10 to
147.128.5.12 port 67 (xid=0x5d38ef7e)
Aug 24 03:06:30 controller dhclient[1166]: DHCPACK of 146.11.41.129 from 147.128.5.12
Aug 24 03:06:30 controller dhclient[1166]: Invalid domain list.
Aug 24 03:06:30 controller dhclient[1166]: suspect value in domain_search option - discarded
Aug 24 03:06:30 controller dhclient[1166]: Invalid domain list.
Aug 24 03:06:30 controller dhclient[1166]: suspect value in domain_search option - discarded
Aug 24 03:06:30 controller dhclient[1166]: Invalid domain list.
Aug 24 03:06:30 controller dhclient[1166]: bound to 146.11.41.129 -- renewal in 12824 seconds.
Aug 24 03:12:18 controller dnsmasq-dhcp[18894]: DHCPDISCOVER(ns-a6e0220e-ec) fa:16:3e:bb:c6:13
Aug 24 03:12:18 controller dnsmasq-dhcp[18894]: DHCPOFFER(ns-a6e0220e-ec) 146.11.41.232
fa:16:3e:bb:c6:13
Aug 24 03:13:19 controller dnsmasq-dhcp[18894]: DHCPDISCOVER(ns-a6e0220e-ec) fa:16:3e:bb:c6:13
Aug 24 03:13:19 controller dnsmasq-dhcp[18894]: DHCPOFFER(ns-a6e0220e-ec) 146.11.41.232
fa:16:3e:bb:c6:13
Aug 24 03:14:19 controller dnsmasq-dhcp[18894]: DHCPDISCOVER(ns-a6e0220e-ec) fa:16:3e:bb:c6:13
Aug 24 03:14:19 controller dnsmasq-dhcp[18894]: DHCPOFFER(ns-a6e0220e-ec) 146.11.41.232
fa:16:3e:bb:c6:13
```

By tcpdump from controllor bridge, it's found the DHCPOFFER was sent to VM:

```
root@controller:~# tcpdump -i  brq2a33434f-ba -vv port 67 or port 68 -e -n
tcpdump: listening on brq2a33434f-ba, link-type EN10MB (Ethernet), capture size 262144 bytes
04:04:22.830138 fa:16:3e:bb:c6:13 > ff:ff:ff:ff:ff:ff, ethertype IPv4 (0x0800), length 332: (tos
0x0, ttl 64, id 0, offset 0, flags [none], proto UDP (17), length 318)
    0.0.0.0.68 > 255.255.255.255.67: [udp sum ok] BOOTP/DHCP, Request from fa:16:3e:bb:c6:13,
length 290, xid 0x1fac2751, Flags [none] (0x0000)
          Client-Ethernet-Address fa:16:3e:bb:c6:13
          Vendor-rfc1048 Extensions
            Magic Cookie 0x63825363
            DHCP-Message Option 53, length 1: Discover
            Client-ID Option 61, length 7: ether fa:16:3e:bb:c6:13
            MSZ Option 57, length 2: 576
            Parameter-Request Option 55, length 9:
              Subnet-Mask, Default-Gateway, Domain-Name-Server, Hostname
              Domain-Name, MTU, BR, NTP
              Classless-Static-Route
            Vendor-Class Option 60, length 12: "udhcp 1.20.1"
            Hostname Option 12, length 6: "cirros"
04:04:22.831801 fa:16:3e:8b:53:5e > fa:16:3e:bb:c6:13, ethertype IPv4 (0x0800), length 370: (tos
0xc0, ttl 64, id 3044, offset 0, flags [none], proto UDP (17), length 356)
    146.11.41.230.67 > 146.11.41.232.68: [udp sum ok] BOOTP/DHCP, Reply, length 328, xid
0x1fac2751, Flags [none] (0x0000)
          Your-IP 146.11.41.232
          Server-IP 146.11.41.230
          Client-Ethernet-Address fa:16:3e:bb:c6:13
          Vendor-rfc1048 Extensions
            Magic Cookie 0x63825363
            DHCP-Message Option 53, length 1: Offer
            Server-ID Option 54, length 4: 146.11.41.230
            Lease-Time Option 51, length 4: 86400
            RN Option 58, length 4: 43200
            RB Option 59, length 4: 75600
            Subnet-Mask Option 1, length 4: 255.255.254.0
            BR Option 28, length 4: 146.11.41.255
            Domain-Name Option 15, length 14: "openstacklocal"
            Default-Gateway Option 3, length 4: 146.11.40.1
            Classless-Static-Route Option 121, length 14: (169.254.169.254/32:146.11.41.230),
(default:146.11.40.1)
            Domain-Name-Server Option 6, length 4: 147.128.5.12
            MTU Option 26, length 2: 1500
04:04:52.504181 08:2e:5f:5d:63:00 > ff:ff:ff:ff:ff:ff, ethertype IPv4 (0x0800), length 358: (tos
0x0, ttl 120, id 5320, offset 0, flags [DF], proto UDP (17), length 344)
```

While from compute , tcpdump the br-int bridge shows it's not received

```
root@compute:~# tcpdump -i brq2a33434f-ba -vv port 67 or port 68 -e -n
tcpdump: listening on brq2a33434f-ba, link-type EN10MB (Ethernet), capture size 262144 bytes
03:51:04.456668 fa:16:3e:bb:c6:13 > ff:ff:ff:ff:ff:ff, ethertype IPv4 (0x0800), length 332: (tos
0x0, ttl 64, id 0, offset 0, flags [none], proto UDP (17), length 318)
    0.0.0.0.68 > 255.255.255.255.67: [udp sum ok] BOOTP/DHCP, Request from fa:16:3e:bb:c6:13,
length 290, xid 0xe5e8f024, Flags [none] (0x0000)
          Client-Ethernet-Address fa:16:3e:bb:c6:13
          Vendor-rfc1048 Extensions
            Magic Cookie 0x63825363
            DHCP-Message Option 53, length 1: Discover
            Client-ID Option 61, length 7: ether fa:16:3e:bb:c6:13
            MSZ Option 57, length 2: 576
            Parameter-Request Option 55, length 9:
              Subnet-Mask, Default-Gateway, Domain-Name-Server, Hostname
              Domain-Name, MTU, BR, NTP
              Classless-Static-Route
            Vendor-Class Option 60, length 12: "udhcp 1.20.1"
            Hostname Option 12, length 6: "cirros"
03:51:30.022360 08:2e:5f:5d:63:00 > ff:ff:ff:ff:ff:ff, ethertype IPv4 (0x0800), length 358: (tos
0x0, ttl 120, id 29901, offset 0, flags [DF], proto UDP (17), length 344)
    147.128.5.12.67 > 255.255.255.255.68: [udp sum ok] BOOTP/DHCP, Reply, length 316, xid
0x4a42e788, Flags [Broadcast] (0x8000)
          Client-IP 146.11.40.250
          Gateway-IP 146.11.40.1
          Client-Ethernet-Address d0:bf:9c:df:7a:a5
          Vendor-rfc1048 Extensions
            Magic Cookie 0x63825363
            DHCP-Message Option 53, length 1: ACK
            Server-ID Option 54, length 4: 147.128.5.12
            Subnet-Mask Option 1, length 4: 255.255.254.0
            Vendor-Option Option 43, length 5: 220.3.78.65.80
            Domain-Name Option 15, length 18: "cn.ao.ericsson.se^@"
            Default-Gateway Option 3, length 4: 146.11.40.1
            Domain-Name-Server Option 6, length 12: 147.128.5.12,193.181.14.11,193.181.14.10
            Netbios-Name-Server Option 44, length 8: 146.11.115.50,146.11.116.30
            Netbios-Node Option 46, length 1: h-node
03:51:30.023490 2c:76:8a:1f:47:00 > ff:ff:ff:ff:ff:ff, ethertype IPv4 (0x0800), length 358: (tos
0x0, ttl 119, id 2187, offset 0, flags [DF], proto UDP (17), length 344)
```

Conclusion:

the DHCP offer was sent out from DHCP agent dnsmasq, but the package cannot be captured from host bridge connecting to vm eth0. the issue is located in the provider network router, the ECN router 146.11.40.1 in our office.

By searching online, there is a tech called DHCP snooping to prevent multiple dhcp server in one LAN from router, which makes sense.

Cinder

Cinder on controller

Here we provide a iSCSI driver backend cinder practice

> ref: https://docs.openstack.org/ocata/install-guide-ubuntu/cinder.html

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

The Block Storage service consists of the following components:

- cinder-api

  Accepts API requests, and routes them to the `cinder-volume` for action.

- cinder-volume

  Interacts directly with the Block Storage service, and processes such as the `cinder-scheduler`. It also interacts with these processes through a message queue. The `cinder-volume` service responds to read and write requests sent to the Block Storage service to maintain state. It can interact with a variety of storage providers through a driver architecture.

- cinder-scheduler daemon

  Selects the optimal storage provider node on which to create the volume. A similar component to the `nova-scheduler`.

- cinder-backup daemon

  The `cinder-backup` service provides backing up volumes of any type to a backup storage provider. Like the `cinder-volume` service, it can interact with a variety of storage providers through a driver architecture.

- Messaging queue

  Routes information between the Block Storage processes.

Install and configure controller node

This section describes how to install and configure the Block Storage service, code-named cinder, on the controller node. This service requires at least one additional storage node that provides volumes to instances.

Prerequisites

Before you install and configure the Block Storage service, you must create a database, service credentials, and API endpoints.

1. To create the database, complete these steps:
   - Use the database access client to connect to the database server as the `root` user:

     ```
     # mysql
     ```

- Create the `cinder` database:

  ```
  MariaDB [(none)]> CREATE DATABASE cinder;
  ```

- Grant proper access to the `cinder` database:

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' \
  IDENTIFIED BY 'CINDER_DBPASS';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'%' \
  IDENTIFIED BY 'CINDER_DBPASS';
```

Replace `CINDER_DBPASS` with a suitable password.

- Exit the database access client.

2. Source the `admin` credentials to gain access to admin-only CLI commands:

```
$ . admin-openrc
```

3. To create the service credentials, complete these steps:
   - Create a `cinder` user:

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

User Password:
Repeat User Password:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | 9d7e33de3e1a498390353819bc7d245d |
| name                | cinder                           |
| options             | {}                               |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

   - Add the `admin` role to the `cinder` user:

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

     This command provides no output.

   - Create the `cinderv2` and `cinderv3` service entities:

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

+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | OpenStack Block Storage          |
| enabled     | True                             |
| id          | eb9fd245bdbc414695952e93f29fe3ac |
| name        | cinderv2                         |
| type        | volumev2                         |
+-------------+----------------------------------+
```

```
$ openstack service create --name cinderv3 \
  --description "OpenStack Block Storage" volumev3

+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | OpenStack Block Storage          |
| enabled     | True                             |
| id          | ab3bbbef780845a1a283490d281e7fda |
| name        | cinderv3                         |
| type        | volumev3                         |
+-------------+----------------------------------+
```

The Block Storage services require two service entities.

4. Create the Block Storage service API endpoints:

```
$ openstack endpoint create --region RegionOne \
  volumev2 public http://controller:8776/v2/%\(project_id\)s

+--------------+----------------------------------------+
| Field        | Value                                  |
+--------------+----------------------------------------+
| enabled      | True                                   |
| id           | 513e73819e14460fb904163f41ef3759       |
| interface    | public                                 |
| region       | RegionOne                              |
| region_id    | RegionOne                              |
| service_id   | eb9fd245bdbc414695952e93f29fe3ac       |
| service_name | cinderv2                               |
| service_type | volumev2                               |
| url          | http://controller:8776/v2/%(project_id)s |
+--------------+----------------------------------------+

$ openstack endpoint create --region RegionOne \
  volumev2 internal http://controller:8776/v2/%\(project_id\)s

+--------------+----------------------------------------+
| Field        | Value                                  |
+--------------+----------------------------------------+
| enabled      | True                                   |
| id           | 6436a8a23d014cfdb69c586eff146a32       |
| interface    | internal                               |
| region       | RegionOne                              |
| region_id    | RegionOne                              |
| service_id   | eb9fd245bdbc414695952e93f29fe3ac       |
| service_name | cinderv2                               |
| service_type | volumev2                               |
| url          | http://controller:8776/v2/%(project_id)s |
+--------------+----------------------------------------+

$ openstack endpoint create --region RegionOne \
  volumev2 admin http://controller:8776/v2/%\(project_id\)s

+--------------+----------------------------------------+
| Field        | Value                                  |
+--------------+----------------------------------------+
| enabled      | True                                   |
| id           | e652cf84dd334f359ae9b045a2c91d96       |
| interface    | admin                                  |
| region       | RegionOne                              |
| region_id    | RegionOne                              |
| service_id   | eb9fd245bdbc414695952e93f29fe3ac       |
| service_name | cinderv2                               |
| service_type | volumev2                               |
| url          | http://controller:8776/v2/%(project_id)s |
+--------------+----------------------------------------+
```

```
$ openstack endpoint create --region RegionOne \
  volumev3 public http://controller:8776/v3/%\(project_id\)s

+--------------+-----------------------------------------+
| Field        | Value                                   |
+--------------+-----------------------------------------+
| enabled      | True                                    |
| id           | 03fa2c90153546c295bf30ca86b1344b        |
| interface    | public                                  |
| region       | RegionOne                               |
| region_id    | RegionOne                               |
| service_id   | ab3bbbef780845a1a283490d281e7fda        |
| service_name | cinderv3                                |
| service_type | volumev3                                |
| url          | http://controller:8776/v3/%(project_id)s |
+--------------+-----------------------------------------+

$ openstack endpoint create --region RegionOne \
  volumev3 internal http://controller:8776/v3/%\(project_id\)s

+--------------+-----------------------------------------+
| Field        | Value                                   |
+--------------+-----------------------------------------+
| enabled      | True                                    |
| id           | 94f684395d1b41068c70e4ecb11364b2        |
| interface    | internal                                |
| region       | RegionOne                               |
| region_id    | RegionOne                               |
| service_id   | ab3bbbef780845a1a283490d281e7fda        |
| service_name | cinderv3                                |
| service_type | volumev3                                |
| url          | http://controller:8776/v3/%(project_id)s |
+--------------+-----------------------------------------+

$ openstack endpoint create --region RegionOne \
  volumev3 admin http://controller:8776/v3/%\(project_id\)s

+--------------+-----------------------------------------+
| Field        | Value                                   |
+--------------+-----------------------------------------+
| enabled      | True                                    |
| id           | 4511c28a0f9840c78bacb25f10f62c98        |
| interface    | admin                                   |
| region       | RegionOne                               |
| region_id    | RegionOne                               |
| service_id   | ab3bbbef780845a1a283490d281e7fda        |
| service_name | cinderv3                                |
| service_type | volumev3                                |
| url          | http://controller:8776/v3/%(project_id)s |
+--------------+-----------------------------------------+
```

The Block Storage services require endpoints for each service entity.

Install and configure components

Install the packages:

```
# apt install cinder-api cinder-scheduler
```

Edit the `/etc/cinder/cinder.conf` file and complete the following actions:

- In the `[database]` section, configure database access:

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

  Replace `CINDER_DBPASS` with the password you chose for the Block Storage database.

- In the `[DEFAULT]` section, configure `RabbitMQ` message queue access:

  ```
  [DEFAULT]
  # ...
  transport_url = rabbit://openstack:RABBIT_PASS@controller
  ```

  Replace `RABBIT_PASS` with the password you chose for the `openstack` account in `RabbitMQ`.

- In the `[DEFAULT]` and `[keystone_authtoken]` sections, configure Identity service access:

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

  [keystone_authtoken]
  # ...
  auth_uri = http://controller:5000
  auth_url = http://controller:35357
  memcached_servers = controller:11211
  auth_type = password
  project_domain_name = default
  user_domain_name = default
  project_name = service
  username = cinder
  password = cinder
  ```

  Replace `password` with the password you chose for the `cinder` user in the Identity service.

  Comment out or remove any other options in the `[keystone_authtoken]` section.

- In the `[DEFAULT]` section, configure the `my_ip` option to use the management interface IP address of the controller node:

  ```
  [DEFAULT]
  # ...
  my_ip = 10.20.0.10
  ```

- In the `[oslo_concurrency]` section, configure the lock path:

```
[oslo_concurrency]
# ...
lock_path = /var/lib/cinder/tmp
```

Populate the Block Storage database:

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

Ignore any deprecation messages in this output.

Configure Compute to use Block Storage

- Edit the `/etc/nova/nova.conf` file and add the following to it:

```
[cinder]
os_region_name = RegionOne
```

Finalize installation

1. Restart the Compute API service:

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

2. Restart the Block Storage services:

```
# service cinder-scheduler restart
# service apache2 restart
```

Cinder on block storage node

configure storage network for compute

Check nic name

```
root@compute:~# dmesg | grep renamed
[    2.730898] e1000 0000:00:09.0 enp0s9: renamed from eth2
[    2.731826] e1000 0000:00:08.0 enp0s8: renamed from eth1
[    2.732819] e1000 0000:00:0a.0 enp0s10: renamed from eth3
[    2.735645] e1000 0000:00:03.0 enp0s3: renamed from eth0
```

`eth2` was named as `enp0s9`

Edit `/etc/network/interfaces`

```
# storage network eth2
auto enp0s9
iface enp0s9 inet static
address 192.168.199.20
netmask 255.255.255.0
```

```
# ifup enp0s9
```

Create cinder machine: storage

Storage node actions

> Clone it from base VM and add a virtual disk for storage vm

Management net eth0 (enp0s3) and storage net eth2 (enp0s9)

Edit `/etc/network/interfaces`

```
# management network eth0

auto enp0s3
iface enp0s3 inet static
address 10.20.0.30
netmask 255.255.255.0
```

```
# storage network eth2
auto enp0s9
iface enp0s9 inet static
address 192.168.199.30
netmask 255.255.255.0
```

```
//start the two nics
# ifup enp0s3
# ifup enp0s9
```

configure NTP by editing `/etc/chrony/chrony.conf`

```
server 10.20.0.10 iburst
```

change hostname and hosts

```
# echo storage > /etc/hostname
# echo 10.20.0.10    controller >> /etc/hosts
# echo 10.20.0.20    compute >> /etc/hosts
# hostname storage
```

Check new disk was there already

check by `fdisk -l` , /dev/sdb is there :-) .

```
root@storage:~# fdisk -l
Disk /dev/sda: 50 GiB, 53687091200 bytes, 104857600 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x3ce50a75

Device     Boot   Start       End   Sectors  Size Id Type
/dev/sda1  *       2048    999423    997376  487M 83 Linux
/dev/sda2       1001470 104855551 103854082 49.5G  5 Extended
/dev/sda5       1001472 104855551 103854080 49.5G 8e Linux LVM


Disk /dev/sdb: 50 GiB, 53687091200 bytes, 104857600 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes


Disk /dev/mapper/ubuntu--vg-root: 45.5 GiB, 48876224512 bytes, 95461376 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes


Disk /dev/mapper/ubuntu--vg-swap_1: 4 GiB, 4294967296 bytes, 8388608 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

Cinder on storage node

Install and configure a storage node

This section describes how to install and configure storage nodes for the Block Storage service. For simplicity, this configuration references one storage node with an empty local block storage device. The instructions use `/dev/sdb`, but you can substitute a different value for your particular node.

The service provisions logical volumes on this device using the LVM driver and provides them to instances via iSCSI transport. You can follow these instructions with minor modifications to horizontally scale your environment with additional storage nodes.

Prerequisites

Before you install and configure the Block Storage service on the storage node, you must prepare the storage device.

Perform these steps on the storage node.

1. Install the supporting utility packages:

   ```
   # apt install lvm2
   ```

   Some distributions include LVM by default.

2. Create the LVM physical volume `/dev/sdb`:

   ```
   # pvcreate /dev/sdb

   Physical volume "/dev/sdb" successfully created
   ```

3. Create the LVM volume group `cinder-volumes`:

   ```
   # vgcreate cinder-volumes /dev/sdb

   Volume group "cinder-volumes" successfully created
   ```

   The Block Storage service creates logical volumes in this volume group.

4. Only instances can access Block Storage volumes. However, the underlying operating system manages the devices associated with the volumes. By default, the LVM volume scanning tool scans the `/dev` directory for block storage devices that contain volumes. If projects use LVM on their volumes, the scanning tool detects these volumes and attempts to cache them which can cause a variety of problems with both the underlying operating system and project volumes. You must reconfigure LVM to scan only the devices that contain the `cinder-volumes` volume group. Edit the `/etc/lvm/lvm.conf` file and complete the following actions:

   - In the `devices` section, add a filter that accepts the `/dev/sdb` device and rejects all other devices:

     ```
     devices {
     ...
     filter = [ "a/sdb/", "r/.*/"]
     ```

     Each item in the filter array begins with `a` for **accept** or `r` for **reject** and includes a regular expression for the device name. The array must end with `r/.*/` to reject any remaining devices. You can use the **vgs -vvvv** command to test filters.

     If your storage nodes use LVM on the operating system disk, you must also add the associated device to the filter. For example, if the `/dev/sda` device contains the operating system:

     ```
     filter = [ "a/sda/", "a/sdb/", "r/.*/"]
     ```

     Similarly, if your compute nodes use LVM on the operating system disk, you must also modify the filter in the `/etc/lvm/lvm.conf` file on those nodes to include only the operating system disk. For example, if the `/dev/sda` device contains the operating system:

     ```
     filter = [ "a/sda/", "r/.*/"]
     ```

Install and configure components

Install the packages:

```
# apt install cinder-volume -y
```

1. Edit the `/etc/cinder/cinder.conf` file and complete the following actions:
   - In the `[database]` section, configure database access:

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

     Replace `CINDER_DBPASS` with the password you chose for the Block Storage database.
   - In the `[DEFAULT]` section, configure `RabbitMQ` message queue access:

     ```
     [DEFAULT]
     # ...
     transport_url = rabbit://openstack:RABBIT_PASS@controller
     ```

     Replace `RABBIT_PASS` with the password you chose for the `openstack` account in `RabbitMQ`.
   - In the `[DEFAULT]` and `[keystone_authtoken]` sections, configure Identity service access:

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

     [keystone_authtoken]
     # ...
     auth_uri = http://controller:5000
     auth_url = http://controller:35357
     memcached_servers = controller:11211
     auth_type = password
     project_domain_name = default
     user_domain_name = default
     project_name = service
     username = cinder
     password = cinder
     ```

     Replace `password` with the password you chose for the `cinder` user in the Identity service.

     Comment out or remove any other options in the `[keystone_authtoken]` section.
   - In the `[DEFAULT]` section, configure the `my_ip` option:

     ```
     [DEFAULT]
     # ...
     my_ip = STORAGE_INTERFACE_IP_ADDRESS
     ```

     Replace `STORAGE_INTERFACE_IP_ADDRESS` with the IP address of the storage network (eth2).

- In the `[lvm]` section, configure the LVM back end with the LVM driver, `cinder-volumes` volume group, iSCSI protocol, and appropriate iSCSI service:

```
[lvm]
# ...
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = cinder-volumes
iscsi_protocol = iscsi
iscsi_helper = tgtadm
```

- In the `[DEFAULT]` section, enable the LVM back end:

```
[DEFAULT]
# ...
enabled_backends = lvm
```

  Back-end names are arbitrary. As an example, this guide uses the name of the driver as the name of the back end.

- In the `[DEFAULT]` section, configure the location of the Image service API:

```
[DEFAULT]
# ...
glance_api_servers = http://controller:9292
```

- In the `[oslo_concurrency]` section, configure the lock path:

```
[oslo_concurrency]
# ...
lock_path = /var/lib/cinder/tmp
```

Finalize installation

1. Restart the Block Storage volume service including its dependencies:

```
# service tgt restart
# service cinder-volume restart
```

Verify operation

Verify operation of the Block Storage service.

Perform these commands on the controller node.

1. Source the `admin` credentials to gain access to admin-only CLI commands:

```
$ . admin-openrc
```

2. List service components to verify successful launch of each process:

```
root@controller:~# openstack volume service list
+------------------+-------------+------+---------+-------+----------------------------+
| Binary           | Host        | Zone | Status  | State | Updated At                 |
+------------------+-------------+------+---------+-------+----------------------------+
| cinder-scheduler | controller  | nova | enabled | up    | 2017-08-24T14:35:44.000000 |
| cinder-volume    | storage@lvm | nova | enabled | up    | 2017-08-24T14:35:39.000000 |
+------------------+-------------+------+---------+-------+----------------------------+
```

Let's try something on block storage!

Create a volume

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

```
$ . demo-openrc
```

2. Create a 1 GB volume:

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

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

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

```
root@controller:~# openstack volume list
+--------------------------------------+--------------+-----------+------+-------------+
| ID                                   | Display Name | Status    | Size | Attached to |
+--------------------------------------+--------------+-----------+------+-------------+
| 81ffed40-ed71-495d-bfa9-8fb8c72cf222 | volume1      | available |    1 |             |
+--------------------------------------+--------------+-----------+------+-------------+
```

4. check where it is?

```
root@storage:~# lvdisplay
  --- Logical volume ---
  LV Path                /dev/ubuntu-vg/root
  LV Name                root
  VG Name                ubuntu-vg
  LV UUID                NA7DgH-V0Sv-cH8E-wvej-aJmP-6EBO-joXO0C
  LV Write Access        read/write
  LV Creation host, time ubuntu, 2017-08-23 16:30:36 +0800
  LV Status              available
  # open                 1
  LV Size                45.52 GiB
  Current LE             11653
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:0

  --- Logical volume ---
  LV Path                /dev/ubuntu-vg/swap_1
  LV Name                swap_1
  VG Name                ubuntu-vg
  LV UUID                Vtixi8-qKcP-f1LH-bHqM-E73h-NN7z-eSD2zk
  LV Write Access        read/write
  LV Creation host, time ubuntu, 2017-08-23 16:30:36 +0800
  LV Status              available
  # open                 2
  LV Size                4.00 GiB
  Current LE             1024
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:1

  --- Logical volume ---
  LV Path                /dev/cinder-volumes/volume-81ffed40-ed71-495d-bfa9-8fb8c72cf222
  LV Name                volume-81ffed40-ed71-495d-bfa9-8fb8c72cf222
  VG Name                cinder-volumes
  LV UUID                6jbPGA-i3Eo-O4ng-8Mf3-IoeF-9WF7-g1DGEA
  LV Write Access        read/write
  LV Creation host, time storage, 2017-08-24 22:38:28 +0800
  LV Status              available
  # open                 0
  LV Size                1.00 GiB
  Current LE             256
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:2
```

Attach the volume to an instance

1. Attach a volume to an instance:

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

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

   Example

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

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

   This command provides no output.

2. List volumes:

```
root@controller:~# openstack volume list
+--------------------------------------+--------------+--------+------+--------------------
----------------------+
| ID                                   | Display Name | Status | Size | Attached to
|
+--------------------------------------+--------------+--------+------+--------------------
----------------------+
| 81ffed40-ed71-495d-bfa9-8fb8c72cf222 | volume1      | in-use |    1 | Attached to
provider-instance on /dev/vdb  |
+--------------------------------------+--------------+--------+------+--------------------
----------------------+

root@storage:~# lvdisplay
  --- Logical volume ---
  LV Path                /dev/ubuntu-vg/root
  LV Name                root
  VG Name                ubuntu-vg
  LV UUID                NA7DgH-V0Sv-cH8E-wvej-aJmP-6EBO-joXO0C
  LV Write Access        read/write
  LV Creation host, time ubuntu, 2017-08-23 16:30:36 +0800
  LV Status              available
  # open                 1
  LV Size                45.52 GiB
  Current LE             11653
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:0

  --- Logical volume ---
  LV Path                /dev/ubuntu-vg/swap_1
  LV Name                swap_1
  VG Name                ubuntu-vg
  LV UUID                Vtixi8-qKcP-f1LH-bHqM-E73h-NN7z-eSD2zk
  LV Write Access        read/write
  LV Creation host, time ubuntu, 2017-08-23 16:30:36 +0800
  LV Status              available
  # open                 2
  LV Size                4.00 GiB
  Current LE             1024
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:1

  --- Logical volume ---
  LV Path                /dev/cinder-volumes/volume-81ffed40-ed71-495d-bfa9-8fb8c72cf222
  LV Name                volume-81ffed40-ed71-495d-bfa9-8fb8c72cf222
  VG Name                cinder-volumes
  LV UUID                6jbPGA-i3Eo-O4ng-8Mf3-IoeF-9WF7-g1DGEA
  LV Write Access        read/write
```

```
  LV Creation host, time storage, 2017-08-24 22:38:28 +0800
  LV Status              available
  # open                 1
  LV Size                1.00 GiB
  Current LE             256
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:2
```

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

```
$ sudo fdisk -l

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

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

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

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

4. Check from storage node on iSCSI target point of view, it's found
    - Initiator: `iqn.1993-08.org.debian:01:e7b693dedcab alias: compute`
    - LUN 1: `Backing store path: /dev/cinder-volumes/volume-81ffed40-ed71-495d-bfa9-8fb8c72cf222`

```
root@storage:~# tgtadm --lld iscsi --op show --mode target
Target 1: iqn.2010-10.org.openstack:volume-81ffed40-ed71-495d-bfa9-8fb8c72cf222
    System information:
        Driver: iscsi
        State: ready
    I_T nexus information:
        I_T nexus: 1
            Initiator: iqn.1993-08.org.debian:01:e7b693dedcab alias: compute
            Connection: 0
                IP Address: 192.168.199.20
    LUN information:
        LUN: 0
            Type: controller
            SCSI ID: IET     00010000
            SCSI SN: beaf10
            Size: 0 MB, Block size: 1
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            SWP: No
            Thin-provisioning: No
            Backing store type: null
            Backing store path: None
            Backing store flags:
        LUN: 1
            Type: disk
            SCSI ID: IET     00010001
            SCSI SN: beaf11
            Size: 1074 MB, Block size: 512
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            SWP: No
            Thin-provisioning: No
            Backing store type: rdwr
            Backing store path: /dev/cinder-volumes/volume-81ffed40-ed71-495d-bfa9-
8fb8c72cf222
            Backing store flags:
    Account information:
        7jTnhhxXsVM4BwqxG979
    ACL information:
        ALL
```

5. Checking from compute via `virsh dumpxml <instance-id>`

   It's shown the device from initiator point of view:

```
    <disk type='block' device='disk'>
      <driver name='qemu' type='raw' cache='none' io='native'/>
      <source dev='/dev/disk/by-path/ip-192.168.199.30:3260-iscsi-iqn.2010-
10.org.openstack:volume-81ffed40-ed71-495d-bfa9-8fb8c72cf222-lun-1'/>
      <backingStore/>
      <target dev='vdb' bus='virtio'/>
      <serial>81ffed40-ed71-495d-bfa9-8fb8c72cf222</serial>
      <alias name='virtio-disk1'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x05' function='0x0'/>
    </disk>
```

Heat

Install and configure

This section describes how to install and configure the Orchestration service for Ubuntu 14.04 (LTS).

While our Ubuntu 16.04.3 LTS will be ok as well.

Prerequisites

Before you install and configure Orchestration, you must create a database, service credentials, and API endpoints. Orchestration also requires additional information in the Identity service.

1. To create the database, complete these steps:
   - Use the database access client to connect to the database server as the `root` user:

     ```
     $ mysql
     ```

   - Create the `heat` database:

     ```
     CREATE DATABASE heat;
     ```

   - Grant proper access to the `heat` database:

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

     Replace `HEAT_DBPASS` with a suitable password.

   - Exit the database access client.
2. Source the `admin` credentials to gain access to admin-only CLI commands:

```
$ . admin-openrc
```

3. To create the service credentials, complete these steps:

   - Create the `heat` user:

     ```
     $ openstack user create --domain default --password-prompt heat
     User Password:
     Repeat User Password:
     +-----------+----------------------------------+
     | Field     | Value                            |
     +-----------+----------------------------------+
     | domain_id | e0353a670a9e496da891347c589539e9 |
     | enabled   | True                             |
     | id        | ca2e175b851943349be29a328cc5e360 |
     | name      | heat                             |
     +-----------+----------------------------------+
     ```

   - Add the `admin` role to the `heat` user:

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

     This command provides no output.

   - Create the `heat` and `heat-cfn` service entities:

     ```
     $ openstack service create --name heat \
       --description "Orchestration" orchestration
     +-------------+----------------------------------+
     | Field       | Value                            |
     +-------------+----------------------------------+
     | description | Orchestration                    |
     | enabled     | True                             |
     | id          | 727841c6f5df4773baa4e8a5ae7d72eb |
     | name        | heat                             |
     | type        | orchestration                    |
     +-------------+----------------------------------+

     $ openstack service create --name heat-cfn \
       --description "Orchestration"  cloudformation
     +-------------+----------------------------------+
     | Field       | Value                            |
     +-------------+----------------------------------+
     | description | Orchestration                    |
     | enabled     | True                             |
     | id          | c42cede91a4e47c3b10c8aedc8d890c6 |
     | name        | heat-cfn                         |
     | type        | cloudformation                   |
     +-------------+----------------------------------+
     ```

4. Create the Orchestration service API endpoints:

```
$ openstack endpoint create --region RegionOne \
  orchestration public http://controller:8004/v1/%\(tenant_id\)s
+--------------+-----------------------------------------+
| Field        | Value                                   |
+--------------+-----------------------------------------+
| enabled      | True                                    |
| id           | 3f4dab34624e4be7b000265f25049609        |
| interface    | public                                  |
| region       | RegionOne                               |
| region_id    | RegionOne                               |
| service_id   | 727841c6f5df4773baa4e8a5ae7d72eb        |
| service_name | heat                                    |
| service_type | orchestration                           |
| url          | http://controller:8004/v1/%(tenant_id)s |
+--------------+-----------------------------------------+

$ openstack endpoint create --region RegionOne \
  orchestration internal http://controller:8004/v1/%\(tenant_id\)s
+--------------+-----------------------------------------+
| Field        | Value                                   |
+--------------+-----------------------------------------+
| enabled      | True                                    |
| id           | 9489f78e958e45cc85570fec7e836d98        |
| interface    | internal                                |
| region       | RegionOne                               |
| region_id    | RegionOne                               |
| service_id   | 727841c6f5df4773baa4e8a5ae7d72eb        |
| service_name | heat                                    |
| service_type | orchestration                           |
| url          | http://controller:8004/v1/%(tenant_id)s |
+--------------+-----------------------------------------+

$ openstack endpoint create --region RegionOne \
  orchestration admin http://controller:8004/v1/%\(tenant_id\)s
+--------------+-----------------------------------------+
| Field        | Value                                   |
+--------------+-----------------------------------------+
| enabled      | True                                    |
| id           | 76091559514b40c6b7b38dde790efe99        |
| interface    | admin                                   |
| region       | RegionOne                               |
| region_id    | RegionOne                               |
| service_id   | 727841c6f5df4773baa4e8a5ae7d72eb        |
| service_name | heat                                    |
| service_type | orchestration                           |
| url          | http://controller:8004/v1/%(tenant_id)s |
+--------------+-----------------------------------------+
```

```
$ openstack endpoint create --region RegionOne \
  cloudformation public http://controller:8000/v1
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | b3ea082e019c4024842bf0a80555052c |
| interface    | public                           |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | c42cede91a4e47c3b10c8aedc8d890c6 |
| service_name | heat-cfn                         |
| service_type | cloudformation                   |
| url          | http://controller:8000/v1        |
+--------------+----------------------------------+

$ openstack endpoint create --region RegionOne \
  cloudformation internal http://controller:8000/v1
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | 169df4368cdc435b8b115a9cb084044e |
| interface    | internal                         |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | c42cede91a4e47c3b10c8aedc8d890c6 |
| service_name | heat-cfn                         |
| service_type | cloudformation                   |
| url          | http://controller:8000/v1        |
+--------------+----------------------------------+

$ openstack endpoint create --region RegionOne \
  cloudformation admin http://controller:8000/v1
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | 3d3edcd61eb343c1bbd629aa041ff88b |
| interface    | internal                         |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | c42cede91a4e47c3b10c8aedc8d890c6 |
| service_name | heat-cfn                         |
| service_type | cloudformation                   |
| url          | http://controller:8000/v1        |
+--------------+----------------------------------+
```

5. Orchestration requires additional information in the Identity service to manage stacks. To add this information, complete these steps:
    - Create the `heat` domain that contains projects and users for stacks:

```
$ openstack domain create --description "Stack projects and users" heat
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | Stack projects and users         |
| enabled     | True                             |
| id          | 0f4d1bd326f2454dacc72157ba328a47 |
| name        | heat                             |
+-------------+----------------------------------+
```

- Create the `heat_domain_admin` user to manage projects and users in the `heat` domain:

  > here i gave password: heat_domain_admin

```
$ openstack user create --domain heat --password-prompt heat_domain_admin
User Password:
Repeat User Password:
+-----------+----------------------------------+
| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | 0f4d1bd326f2454dacc72157ba328a47 |
| enabled   | True                             |
| id        | b7bd1abfbcf64478b47a0f13cd4d970a |
| name      | heat_domain_admin                |
+-----------+----------------------------------+
```

- Add the `admin` role to the `heat_domain_admin` user in the `heat` domain to enable administrative stack management privileges by the `heat_domain_admin` user:

```
$ openstack role add --domain heat --user-domain heat --user heat_domain_admin admin
```

  This command provides no output.

- Create the `heat_stack_owner` role:

```
$ openstack role create heat_stack_owner
+-----------+----------------------------------+
| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | None                             |
| id        | 15e34f0c4fed4e68b3246275883c8630 |
| name      | heat_stack_owner                 |
+-----------+----------------------------------+
```

- Add the `heat_stack_owner` role to the `demo` project and user to enable stack management by the `demo` user:

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

  This command provides no output.

> You must add the `heat_stack_owner` role to each user that manages stacks.

- Create the `heat_stack_user` role:

```
$ openstack role create heat_stack_user
+-----------+----------------------------------+
| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | None                             |
| id        | 88849d41a55d4d1d91e4f11bffd8fc5c |
| name      | heat_stack_user                  |
+-----------+----------------------------------+
```

The Orchestration service automatically assigns the `heat_stack_user` role to users that it creates during stack deployment. By default, this role restricts API operations. To avoid conflicts, do not add this role to users with the `heat_stack_owner` role.

Install and configure components

Install the packages:

```
# apt-get install heat-api heat-api-cfn heat-engine
```

1. Edit the `/etc/heat/heat.conf` file and complete the following actions:
   - In the `[database]` section, configure database access:

```
[database]
...
connection = mysql+pymysql://heat:HEAT_DBPASS@controller/heat
```

Replace `HEAT_DBPASS` with the password you chose for the Orchestration database.

   - In the `[DEFAULT]` section, configure `RabbitMQ` message queue access:

```
[DEFAULT]
...
transport_url = rabbit://openstack:RABBIT_PASS@controller
```

Replace `RABBIT_PASS` with the password you chose for the `openstack` account in `RabbitMQ`.

   - In the `[keystone_authtoken]`, `[trustee]` and `[clients_keystone]` sections, configure Identity service access:

```
[keystone_authtoken]
...
auth_uri = http://controller:5000
auth_url = http://controller:35357
memcached_servers = controller:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = heat
password = HEAT_PASS

[trustee]
...
auth_type = password
auth_url = http://controller:35357
username = heat
password = heat
user_domain_name = default

[clients_keystone]
...
auth_uri = http://controller:5000
```

Replace `password` with the password you chose for the `heat` user in the Identity service.

- In the `[DEFAULT]` section, configure the metadata and wait condition URLs:

```
[DEFAULT]
...
heat_metadata_server_url = http://controller:8000
heat_waitcondition_server_url = http://controller:8000/v1/waitcondition
```

- In the `[DEFAULT]` section, configure the stack domain and administrative credentials:

```
[DEFAULT]
...
stack_domain_admin = heat_domain_admin
stack_domain_admin_password = heat_domain_admin
stack_user_domain_name = heat
```

Replace `heat_domain_admin` with the password you chose for the `heat_domain_admin` user in the Identity service.

2. Populate the Orchestration database:

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

Ignore any deprecation messages in this output.

Finalize installation

1. Restart the Orchestration services:

```
# service heat-api restart
# service heat-api-cfn restart
# service heat-engine restart
```

Verify operation

Verify operation of the Orchestration service.

Perform these commands on the controller node.

1. Source the `admin` tenant credentials:

```
$ . admin-openrc
```

2. List service components to verify successful launch and registration of each process:

```
$ openstack orchestration service list
+------------+-------------+--------------------------------------+------------+--------+---
------------------------+--------+
| hostname   | binary      | engine_id                            | host       | topic  |
updated_at                | status |
+------------+-------------+--------------------------------------+------------+--------+---
------------------------+--------+
| controller | heat-engine | 3e85d1ab-a543-41aa-aa97-378c381fb958 | controller | engine |
2015-10-13T14:16:06.000000 | up     |
| controller | heat-engine | 45dbdcf6-5660-4d5f-973a-c4fc819da678 | controller | engine |
2015-10-13T14:16:06.000000 | up     |
| controller | heat-engine | 51162b63-ecb8-4c6c-98c6-993af899c4f7 | controller | engine |
2015-10-13T14:16:06.000000 | up     |
| controller | heat-engine | 8d7edc6d-77a6-460d-bd2a-984d76954646 | controller | engine |
2015-10-13T14:16:06.000000 | up     |
+------------+-------------+--------------------------------------+------------+--------+---
------------------------+--------+
```

   This output should indicate four `heat-engine` components (default to 4 or number of CPUs on the host, whichever is greater) on the controller node.

[ISSUE] list heat service failure

error occurred during `openstack orchestration service list`

the initial output is very nonsense as below:

```
# openstack orchestration service list
ERROR: None
```

give `--debug` to have detailed information

```
REQ: curl -g -i -X GET http://controller:8004/v1/78c9c849237649a3a8c4526167427589/services -H
"User-Agent: python-heatclient" -H "Accept: application/json" -H "X-Auth-Token:
{SHA1}d4b406278269babd78368ed572cbe50382938cb6"
Starting new HTTP connection (1): controller
http://controller:8004 "GET /v1/78c9c849237649a3a8c4526167427589/services HTTP/1.1" 503 170
RESP: [503] Content-Length: 170 Content-Type: application/json; charset=UTF-8 X-Openstack-
Request-Id: req-e38d405a-776d-44fa-bb31-e180d62c42e0 Date: Fri, 25 Aug 2017 06:05:45 GMT
Connection: keep-alive
RESP BODY: {"message": "The server is currently unavailable. Please try again at a later time.<br
/><br />\n\n\n", "code": "503 Service Unavailable", "title": "Service Unavailable"}

GET call to orchestration for http://controller:8004/v1/78c9c849237649a3a8c4526167427589/services
used request id req-e38d405a-776d-44fa-bb31-e180d62c42e0
ERROR: None
Traceback (most recent call last):
  File "/usr/lib/python2.7/dist-packages/cliff/app.py", line 400, in run_subcommand
    result = cmd.run(parsed_args)
  File "/usr/lib/python2.7/dist-packages/osc_lib/command/command.py", line 41, in run
    return super(Command, self).run(parsed_args)
  File "/usr/lib/python2.7/dist-packages/cliff/display.py", line 112, in run
    column_names, data = self.take_action(parsed_args)
  File "/usr/lib/python2.7/dist-packages/heatclient/osc/v1/service.py", line 37, in take_action

  File "/usr/lib/python2.7/dist-packages/heatclient/v1/services.py", line 33, in list
    return self._list(url, "services")
  File "/usr/lib/python2.7/dist-packages/heatclient/common/base.py", line 114, in _list

  File "/usr/lib/python2.7/dist-packages/keystoneauth1/adapter.py", line 217, in get
    return self.request(url, 'GET', **kwargs)
  File "/usr/lib/python2.7/dist-packages/heatclient/common/http.py", line 318, in request
    raise exc.from_response(resp)
HTTPServiceUnavailable: ERROR: None
clean_up ListService: ERROR: None
Traceback (most recent call last):
  File "/usr/lib/python2.7/dist-packages/osc_lib/shell.py", line 135, in run
    ret_val = super(OpenStackShell, self).run(argv)
  File "/usr/lib/python2.7/dist-packages/cliff/app.py", line 279, in run
    result = self.run_subcommand(remainder)
  File "/usr/lib/python2.7/dist-packages/osc_lib/shell.py", line 180, in run_subcommand
    ret_value = super(OpenStackShell, self).run_subcommand(argv)
  File "/usr/lib/python2.7/dist-packages/cliff/app.py", line 400, in run_subcommand
    result = cmd.run(parsed_args)
  File "/usr/lib/python2.7/dist-packages/osc_lib/command/command.py", line 41, in run
    return super(Command, self).run(parsed_args)
  File "/usr/lib/python2.7/dist-packages/cliff/display.py", line 112, in run
    column_names, data = self.take_action(parsed_args)
  File "/usr/lib/python2.7/dist-packages/heatclient/osc/v1/service.py", line 37, in take_action
    services = heat_client.services.list()
  File "/usr/lib/python2.7/dist-packages/heatclient/v1/services.py", line 33, in list
    return self._list(url, "services")
  File "/usr/lib/python2.7/dist-packages/heatclient/common/base.py", line 114, in _list
    body = self.client.get(url).json()
  File "/usr/lib/python2.7/dist-packages/keystoneauth1/adapter.py", line 217, in get
```

```
    return self.request(url, 'GET', **kwargs)
  File "/usr/lib/python2.7/dist-packages/heatclient/common/http.py", line 318, in request
    raise exc.from_response(resp)
HTTPServiceUnavailable: ERROR: None

END return value: 1
```

We could see it got `503` when performing api call in `8004` port

```
REQ: curl -g -i -X GET http://controller:8004/v1/78c9c849237649a3a8c4526167427589/services -H
"User-Agent: python-heatclient" -H "Accept: application/json" -H "X-Auth-Token:
{SHA1}d4b406278269babd78368ed572cbe50382938cb6"
Starting new HTTP connection (1): controller
http://controller:8004 "GET /v1/78c9c849237649a3a8c4526167427589/services HTTP/1.1" 503 170
RESP: [503] Content-Length: 170 Content-Type: application/json; charset=UTF-8 X-Openstack-
Request-Id: req-e38d405a-776d-44fa-bb31-e180d62c42e0 Date: Fri, 25 Aug 2017 06:05:45 GMT
Connection: keep-alive
RESP BODY: {"message": "The server is currently unavailable. Please try again at a later time.<br
/><br />\n\n\n", "code": "503 Service Unavailable", "title": "Service Unavailable"}
```

it's checked to be heat wsgi service

```
# openstack endpoint list | grep 8004
| 1d6153de2a7a4bfc8f5277b360d5b695 | RegionOne | heat         | orchestration  | True    |
internal  | http://controller:8004/v1/%(tenant_id)s   |
| 38dd2f686f5840e6ae9381b6df1076d8 | RegionOne | heat         | orchestration  | True    | admin
| http://controller:8004/v1/%(tenant_id)s   |
| 88b0e0667eb149efa337e6e0428c98ad | RegionOne | heat         | orchestration  | True    | public
| http://controller:8004/v1/%(tenant_id)s   |
```

Then let's check `503` in `/var/log/heat/heat-api.log`

```
2017-08-25 14:24:48.962 2778 WARNING keystonemiddleware.auth_token [-] Identity response:
{"error": {"message": "The request you have made requires authentication.", "code": 401, "title":
"Unauthorized"}}
```

it's keystone `401` meaning the credential is with issues when requesting token from keystone, let us check keystone logs:

```
# grep "Authorization failed" /var/log/apache2/keyston*.log

/var/log/apache2/keystone.log:2017-08-25 14:24:48.954127 2017-08-25 14:24:48.953 3388 WARNING
keystone.common.wsgi [req-fc1e6351-ab7a-40e3-967d-4e7376ced9d9 - - - - -] Authorization failed.
The request you have made requires authentication. from 10.20.0.10
```

Solution

Where the credential for heat keystone call was configured? `/etc/heat/heat.conf` , it turned out we set wrong password for keystone, the one we set was `heat` while HEAT_PASS was configured, changed it as below and restart services will solve the issue.

```
[keystone_authtoken]
...
password = heat

[trustee]
...
password = heat
```

restart services to make it work

```
# service heat-api restart
# service heat-api-cfn restart
# service heat-engine restart
```

and verify it:

```
root@controller:/var/log# openstack orchestration service list --max-width 85
+------------+------------+------------+------------+--------+-------------+--------+
| Hostname   | Binary     | Engine ID  | Host       | Topic  | Updated At  | Status |
+------------+------------+------------+------------+--------+-------------+--------+
| controller | heat-      | 07ff1da7   | controller | engine | 2017-08-25T | up     |
|            | engine     | -a77b-4d52 |            |        | 06:49:36.00 |        |
|            |            | -95e6-cb53 |            |        | 0000        |        |
|            |            | 89a106dc   |            |        |             |        |
|            | engine     | de8-45eb-9 |            |        | 06:24:52.00 |        |
|            |            | 839-4d038d |            |        | 0000        |        |
|            |            | a4a330     |            |        |             |        |
| controller | heat-      | 279303d6-2 | controller | engine | 2017-08-25T | up     |
|            | engine     | d06-4e55-b |            |        | 06:49:36.00 |        |
|            |            | 931-2c8ad0 |            |        | 0000        |        |
|            |            | eadca7     |            |        |             |        |
| controller | heat-      | 2cd3a832-d | controller | engine | 2017-08-25T | up     |
|            | engine     | c21-4cfa-  |            |        | 06:49:36.00 |        |
|            |            | 8a4b-bddd8 |            |        | 0000        |        |
|            |            | dadba12    |            |        |             |        |
| controller | heat-      | e1b05787   | controller | engine | 2017-08-25T | up     |
|            | engine     | -9eda-46c3 |            |        | 06:49:36.00 |        |
|            |            | -acca-b1f9 |            |        | 0000        |        |
|            |            | bcf7b653   |            |        |             |        |
|            | engine     | 1fa-4844-9 |            |        | 06:24:52.00 |        |
|            |            | f68-688486 |            |        | 0000        |        |
|            |            | f4ccfb     |            |        |             |        |
|            | engine     | 023-4d10   |            |        | 06:24:52.00 |        |
|            |            | -89ea-30af |            |        | 0000        |        |
|            |            | f5430d25   |            |        |             |        |
|            | engine     | b86-41af-  |            |        | 06:24:52.00 |        |
|            |            | 895a-0c43a |            |        | 0000        |        |
|            |            | f016950    |            |        |             |        |
+------------+------------+------------+------------+--------+-------------+--------+
```

Start Orchestration! let's start from a single instance HOT

> ref: https://docs.openstack.org/heat/latest/install/launch-instance.html

In environments that include the Orchestration service, you can create a stack that launches an instance.

Create a template

The Orchestration service uses templates to describe stacks. To learn about the template language, see the Template Guide in the Heat developer documentation.

- Create the `HOT-demo.yml` file with the following content:

```
heat_template_version: 2015-10-15
description: Launch a basic instance with CirrOS image using the
             ``m1.nano`` flavor,  and one network.

parameters:
  NetID:
    type: string
    description: Network ID to use for the instance.

resources:
  server:
    type: OS::Nova::Server
    properties:
      image: cirros
      flavor: m1.nano
      networks:
      - network: { get_param: NetID }

outputs:
  instance_name:
    description: Name of the instance.
    value: { get_attr: [ server, name ] }
  instance_ip:
    description: IP address of the instance.
    value: { get_attr: [ server, first_address ] }
```

Create a stack

Create a stack using the `demo-template.yml` template.

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

```
$ . demo-openrc
```

2. Determine available networks.

```
root@controller:~# openstack network list
+--------------------------------------+----------+--------------------------------------+
| ID                                   | Name     | Subnets                              |
+--------------------------------------+----------+--------------------------------------+
| 2a33434f-ba29-4645-9b5d-24f1509066f1 | provider | 9b118521-59b5-40ee-a439-9d59c3b392ea |
+--------------------------------------+----------+--------------------------------------+
```

This output may differ from your environment.

3. Set the `NET_ID` environment variable to reflect the ID of a network. For example, using the provider network:

```
$ export NET_ID=$(openstack network list | awk '/ provider / { print $2 }')
```

4. Create a stack of one CirrOS instance on the provider network:

```
$ openstack stack create -t HOT-demo.yml --parameter "NetID=$NET_ID" stack
+--------------------------------------+------------+--------------------+------------------
---+--------------+
| ID                                   | Stack Name | Stack Status       | Creation Time
| Updated Time |
+--------------------------------------+------------+--------------------+------------------
---+--------------+
| dbf46d1b-0b97-4d45-a0b3-9662a1eb6cf3 | stack      | CREATE_IN_PROGRESS | 2015-10-
13T15:27:20 | None         |
+--------------------------------------+------------+--------------------+------------------
---+--------------+
```

5. After a short time, verify successful creation of the stack:

```
$ openstack stack list
+--------------------------------------+------------+-----------------+--------------------
+--------------+
| ID                                   | Stack Name | Stack Status    | Creation Time
| Updated Time |
+--------------------------------------+------------+-----------------+--------------------
+--------------+
| dbf46d1b-0b97-4d45-a0b3-9662a1eb6cf3 | stack      | CREATE_COMPLETE | 2015-10-13T15:27:20
| None         |
+--------------------------------------+------------+-----------------+--------------------
+--------------+
```

6. Show the name and IP address of the instance and compare with the output of the OpenStack client:

```
root@controller:~# openstack stack output show --all stack
+---------------+-------------------------------------------------+
| Field         | Value                                           |
+---------------+-------------------------------------------------+
| instance_name | {                                               |
|               |   "output_value": "stack-server-bigm7yoguexa",  |
|               |   "output_key": "instance_name",                |
|               |   "description": "Name of the instance."        |
|               | }                                               |
| instance_ip   | {                                               |
|               |   "output_value": "146.11.41.233",              |
|               |   "output_key": "instance_ip",                  |
|               |   "description": "IP address of the instance."  |
|               | }                                               |
+---------------+-------------------------------------------------+
```

```
root@controller:~# openstack server list
+--------------------------------------+--------------------------+---------+-------------
----------+------------+
| ID                                   | Name                     | Status  | Networks
| Image Name |
+--------------------------------------+--------------------------+---------+-------------
----------+------------+
| a81fc7b9-b5c0-4a0f-85d3-3ff739f8e5ce | stack-server-bigm7yoguexa | ACTIVE  |
provider=146.11.41.233 | cirros     |
| e73c64ac-3af3-47fd-abfe-e138e40f0a40 | provider-instance         | SHUTOFF |
provider=146.11.41.232 | cirros     |
+--------------------------------------+--------------------------+---------+-------------
----------+------------+
```

7. Delete the stack.

```
$ openstack stack delete --yes stack
```

How about Design a fake vAPG VNF and instantiate it?

> ref: https://docs.openstack.org/heat/latest/template_guide/index.html

HOT

```
heat_template_version: 2015-10-15
description: Fake vAPG with CirrOS image using the
             ``m1.nano`` flavor,  and one network.

parameters:
  NetID:
    type: string
    description: Network ID to use for the instance.

resources:
  nodeA:
    type: OS::Nova::Server
    properties:
      image: cirros
      flavor: m1.nano
      networks:
      - network: { get_param: NetID }
  nodeB:
    type: OS::Nova::Server
    properties:
      image: cirros
      flavor: m1.nano
      networks:
      - network: { get_param: NetID }
  diskA:
    type: OS::Cinder::Volume
    properties:
      size: 1
  diskB:
    type: OS::Cinder::Volume
    properties:
      size: 1
  NodeAvolume_attachment:
    type: OS::Cinder::VolumeAttachment
    properties:
      volume_id: { get_resource: diskA }
      instance_uuid: { get_resource: nodeA }
  NodeBvolume_attachment:
    type: OS::Cinder::VolumeAttachment
    properties:
      volume_id: { get_resource: diskB }
      instance_uuid: { get_resource: nodeB }
outputs:
  nodeA:
    description: Name of the instance.
    value: { get_attr: [ nodeA, name ] }
  nodeA_ip:
    description: IP address of the instance.
    value: { get_attr: [ nodeA, first_address ] }
  nodeB:
    description: Name of the instance.
    value: { get_attr: [ nodeB, name ] }
  nodeB_ip:
```

```
    description: IP address of the instance.
    value: { get_attr: [ nodeB, first_address ] }
```

Instantiation vAPG

For simplicity we reused the existed network

```
# . demo-openrc
# openstack stack create -t HOT-vAPG.yml --parameter "NetID=$NET_ID" vAPG

// monitoring

# openstack stack output show --all vAPG
```

Print outs during/after instantiation

```
root@controller:~# openstack stack list
+--------------------------------------+------------+--------------------+----------------------
+--------------+
| ID                                   | Stack Name | Stack Status       | Creation Time        |
Updated Time |
+--------------------------------------+------------+--------------------+----------------------
+--------------+
| 7c4f5cda-a943-4d64-be23-72402829c62f | vAPG       | CREATE_IN_PROGRESS | 2017-08-25T07:38:55Z |
None         |
+--------------------------------------+------------+--------------------+----------------------
+--------------+
root@controller:~#
root@controller:~#
root@controller:~# openstack stack list
+--------------------------------------+------------+-----------------+----------------------+---
-----------+
| ID                                   | Stack Name | Stack Status    | Creation Time        |
Updated Time |
+--------------------------------------+------------+-----------------+----------------------+---
-----------+
| 7c4f5cda-a943-4d64-be23-72402829c62f | vAPG       | CREATE_COMPLETE | 2017-08-25T07:38:55Z |
None         |
+--------------------------------------+------------+-----------------+----------------------+---
-----------+
root@controller:~# openstack stack output show --all vAPG
+----------+-------------------------------------------------+
| Field    | Value                                           |
+----------+-------------------------------------------------+
| nodeA    | {                                               |
|          |   "output_value": "vAPG-nodeA-slvg7ccfel4j",    |
|          |   "output_key": "nodeA",                        |
|          |   "description": "Name of the instance."        |
|          | }                                               |
| nodeB_ip | {                                               |
|          |   "output_value": "146.11.41.231",              |
|          |   "output_key": "nodeB_ip",                     |
|          |   "description": "IP address of the instance." |
|          | }                                               |
| nodeA_ip | {                                               |
|          |   "output_value": "146.11.41.233",              |
|          |   "output_key": "nodeA_ip",                     |
|          |   "description": "IP address of the instance." |
|          | }                                               |
| nodeB    | {                                               |
|          |   "output_value": "vAPG-nodeB-rjame5ftjcrf",    |
|          |   "output_key": "nodeB",                        |
|          |   "description": "Name of the instance."        |
|          | }                                               |
+----------+-------------------------------------------------+
root@controller:~# openstack server list
+--------------------------------------+------------------------+---------+---------------------
---+------------+
| ID                                   | Name                   | Status  | Networks
```

```
| Image Name |
+--------------------------------------+------------------------+---------+--------------------
---+------------+
| 3eef32d9-f885-48ca-a038-365233ed300f | vAPG-nodeB-rjame5ftjcrf | ACTIVE  |
provider=146.11.41.231 | cirros     |
| 2a2709f2-9d77-42e1-be36-fd51c33b30de | vAPG-nodeA-slvg7ccfel4j | ACTIVE  |
provider=146.11.41.233 | cirros     |
| e73c64ac-3af3-47fd-abfe-e138e40f0a40 | provider-instance       | SHUTOFF |
provider=146.11.41.232 | cirros     |
+--------------------------------------+------------------------+---------+--------------------
---+------------+
```

Check resources afterwards

```
root@controller:~# openstack volume list
+--------------------------------+------------------------+-----------+------+------------------
--------------+
| ID                             | Display Name           | Status    | Size | Attached to
|
+--------------------------------+------------------------+-----------+------+------------------
--------------+
| 9a93c5aa-56ed-                 | vAPG-diskA-ff253tzmwhuv | in-use    |    1 | Attached to vAPG-
nodeA-        |
| 42e0-b0a0-e0e617b533e1         |                        |           |      | slvg7ccfel4j on
/dev/vdb       |
| 988f5003-2a3f-                 | vAPG-diskB-xpzh5o7pw5po | in-use    |    1 | Attached to vAPG-
nodeB-        |
| 4eeb-b836-6be6f64b0b32         |                        |           |      | rjame5ftjcrf on
/dev/vdb       |
+--------------------------------+------------------------+-----------+------+------------------
--------------+
root@controller:~# openstack server list
+--------------------------------------+------------------------+---------+--------------------
---+------------+
| ID                                   | Name                   | Status  | Networks
| Image Name |
+--------------------------------------+------------------------+---------+--------------------
---+------------+
| 3eef32d9-f885-48ca-a038-365233ed300f | vAPG-nodeB-rjame5ftjcrf | ACTIVE  |
provider=146.11.41.231 | cirros     |
| 2a2709f2-9d77-42e1-be36-fd51c33b30de | vAPG-nodeA-slvg7ccfel4j | ACTIVE  |
provider=146.11.41.233 | cirros     |
+--------------------------------------+------------------------+---------+--------------------
---+------------+

root@controller:~# openstack port list
+--------------------------------------+------+-------------------+------------------------------
----------+--------+
| ID                                   | Name | MAC Address       | Fixed IP Addresses
| Status |
+--------------------------------------+------+-------------------+------------------------------
----------+--------+
| 0e54bc09-29d2-4774-bd71-413739b7bffe |      | fa:16:3e:0c:8a:66 | ip_address='146.11.41.233',
subnet_id= | ACTIVE |
|                                      |      |                   | '9b118521-59b5-40ee-a439-
9d59c3b392ea' |        |
| 2aa35c9c-2b50-40cd-8971-2056fb4cf04d |      | fa:16:3e:bb:c6:13 | ip_address='146.11.41.232',
subnet_id= | ACTIVE |
|                                      |      |                   | '9b118521-59b5-40ee-a439-
9d59c3b392ea' |        |
+--------------------------------------+------+-------------------+------------------------------
----------+--------+
```

Horizon dashboard

The Dashboard (horizon) is a web interface that enables cloud administrators and users to manage various OpenStack resources and services.

This example deployment uses an Apache web server.

> it's actually a Django based openstack client front-end

Install and configure

Install the packages:

```
# apt install openstack-dashboard
```

1. Edit the `/etc/openstack-dashboard/local_settings.py` file and complete the following actions:
   - Configure the dashboard to use OpenStack services on the `controller` node:

     ```
     OPENSTACK_HOST = "controller"
     ```

   - In the Dashboard configuration section, allow your hosts to access Dashboard:

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

     - Do not edit the `ALLOWED_HOSTS` parameter under the Ubuntu configuration section.
     - `ALLOWED_HOSTS` can also be `['*']` to accept all hosts. This may be useful for development work, but is potentially insecure and should not be used in production. See the Django documentation for further information.

   - Configure the `memcached` session storage service:

     ```
     SESSION_ENGINE = 'django.contrib.sessions.backends.cache'

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

```

```

```
Comment out any other session storage configuration.
```

- Enable the Identity API version 3:

  ```
  OPENSTACK_KEYSTONE_URL = "http://%s:5000/v3" % OPENSTACK_HOST
  ```

- Enable support for domains:

```
OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True
```

- Configure API versions:

```
OPENSTACK_API_VERSIONS = {
    "identity": 3,
    "image": 2,
    "volume": 2,
}
```

- Configure `Default` as the default domain for users that you create via the dashboard:

```
OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = "Default"
```

- Configure `user` as the default role for users that you create via the dashboard:

```
OPENSTACK_KEYSTONE_DEFAULT_ROLE = "user"
```

- If you chose networking option 1, disable support for layer-3 networking services:

```
OPENSTACK_NEUTRON_NETWORK = {
    ...
    'enable_router': False,
    'enable_quotas': False,
    'enable_ipv6': False,
    'enable_distributed_router': False,
    'enable_ha_router': False,
    'enable_lb': False,
    'enable_firewall': False,
    'enable_vpn': False,
    'enable_fip_topology_check': False,
}
```

- Optionally, configure the time zone:

```
TIME_ZONE = "CN"
```

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

Finalize installation

- Reload the web server configuration:

```
# service apache2 reloadetc
```

Verify operation

Verify operation of the dashboard.

Access the dashboard using a web browser at `http://controller/horizon`.

Authenticate using `admin` or `demo` user and `default` domain credentials.

> Monitoring the how process by `tail -f /var/log/apache2/*.log`

[ISSUE] Horizon `500` internal error

Fault reproduce in cli

```
# curl http://10.20.0.10/horizon
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>500 Internal Server Error</title>
</head><body>
<h1>Internal Server Error</h1>
<p>The server encountered an internal error or
misconfiguration and was unable to complete
your request.</p>
<p>Please contact the server administrator at
 webmaster@localhost to inform them of the time this error occurred,
 and the actions you performed just before this error.</p>
<p>More information about this error may be available
in the server error log.</p>
<hr>
<address>Apache/2.4.18 (Ubuntu) Server at 10.20.0.10 Port 80</address>
</body></html>
```

It's apache2 based web service, error could be fond in `/var/log/apache2/error.log`

```
[Fri Aug 25 16:22:13.681394 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248] mod_wsgi (pid=9401): Target WSGI script '/usr/share/openstack-
dashboard/openstack_dashboard/wsgi/django.wsgi' cannot be loaded as Python module.
[Fri Aug 25 16:22:13.681453 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248] mod_wsgi (pid=9401): Exception occurred processing WSGI script
'/usr/share/openstack-dashboard/openstack_dashboard/wsgi/django.wsgi'.
[Fri Aug 25 16:22:13.681498 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248] Traceback (most recent call last):
[Fri Aug 25 16:22:13.681531 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]   File "/usr/share/openstack-dashboard/openstack_dashboard/wsgi/django.wsgi",
line 16, in <module>[Fri Aug 25 16:22:13.681621 2017] [wsgi:error] [pid 9401:tid 140055623386880]
[remote 10.20.0.10:18248]     application = get_wsgi_application()
[Fri Aug 25 16:22:13.681635 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]   File "/usr/lib/python2.7/dist-packages/django/core/wsgi.py", line 14, in
get_wsgi_application
[Fri Aug 25 16:22:13.681672 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]     django.setup()
[Fri Aug 25 16:22:13.681682 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]   File "/usr/lib/python2.7/dist-packages/django/__init__.py", line 17, in setup
[Fri Aug 25 16:22:13.681713 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]     configure_logging(settings.LOGGING_CONFIG, settings.LOGGING)
[Fri Aug 25 16:22:13.681726 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]   File "/usr/lib/python2.7/dist-packages/django/conf/__init__.py", line 48, in
__getattr__
[Fri Aug 25 16:22:13.681806 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]     self._setup(name)
[Fri Aug 25 16:22:13.681886 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]   File "/usr/lib/python2.7/dist-packages/django/conf/__init__.py", line 44, in
_setup
[Fri Aug 25 16:22:13.681909 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]     self._wrapped = Settings(settings_module)
[Fri Aug 25 16:22:13.681916 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]   File "/usr/lib/python2.7/dist-packages/django/conf/__init__.py", line 92, in
__init__
[Fri Aug 25 16:22:13.681927 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]     mod = importlib.import_module(self.SETTINGS_MODULE)
[Fri Aug 25 16:22:13.681934 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]   File "/usr/lib/python2.7/importlib/__init__.py", line 37, in import_module
[Fri Aug 25 16:22:13.681974 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]     __import__(name)
[Fri Aug 25 16:22:13.681984 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]   File "/usr/share/openstack-dashboard/openstack_dashboard/settings.py", line
335, in <module>
[Fri Aug 25 16:22:13.682112 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]     from local.local_settings import *  # noqa
[Fri Aug 25 16:22:13.682123 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]   File "/usr/share/openstack-
dashboard/openstack_dashboard/local/local_settings.py", line 131, in <module>
[Fri Aug 25 16:22:13.682586 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]     SECRET_KEY = secret_key.generate_or_read_from_file('/var/lib/openstack-
dashboard/secret_key') [Fri Aug 25 16:22:13.682611 2017] [wsgi:error] [pid 9401:tid
140055623386880] [remote 10.20.0.10:18248]   File "/usr/share/openstack-
dashboard/horizon/utils/secret_key.py", line 70, in generate_or_read_from_file
```

```
[Fri Aug 25 16:22:13.682688 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]     key = read_from_file(key_file)
[Fri Aug 25 16:22:13.682699 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]   File "/usr/share/openstack-dashboard/horizon/utils/secret_key.py", line 52,
in read_from_file
[Fri Aug 25 16:22:13.682714 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248]     with open(key_file, 'r') as f:
[Fri Aug 25 16:22:13.682746 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
10.20.0.10:18248] IOError: [Errno 13] Permission denied: '/var/lib/openstack-
dashboard/secret_key'
```

beautify it as below

```
 mod_wsgi (pid=9401): Target WSGI script '/usr/share/openstack-
dashboard/openstack_dashboard/wsgi/django.wsgi' cannot be loaded as Python module.
 mod_wsgi (pid=9401): Exception occurred processing WSGI script '/usr/share/openstack-
dashboard/openstack_dashboard/wsgi/django.wsgi'.
 Traceback (most recent call last):
   File "/usr/share/openstack-dashboard/openstack_dashboard/wsgi/django.wsgi", line 16, in
<module>[Fri Aug 25 16:22:13.681621 2017] [wsgi:error] [pid 9401:tid 140055623386880] [remote
application = get_wsgi_application()
   File "/usr/lib/python2.7/dist-packages/django/core/wsgi.py", line 14, in get_wsgi_application
     django.setup()
   File "/usr/lib/python2.7/dist-packages/django/__init__.py", line 17, in setup
     configure_logging(settings.LOGGING_CONFIG, settings.LOGGING)
   File "/usr/lib/python2.7/dist-packages/django/conf/__init__.py", line 48, in __getattr__
     self._setup(name)
   File "/usr/lib/python2.7/dist-packages/django/conf/__init__.py", line 44, in _setup
     self._wrapped = Settings(settings_module)
   File "/usr/lib/python2.7/dist-packages/django/conf/__init__.py", line 92, in __init__
     mod = importlib.import_module(self.SETTINGS_MODULE)
   File "/usr/lib/python2.7/importlib/__init__.py", line 37, in import_module
     __import__(name)
   File "/usr/share/openstack-dashboard/openstack_dashboard/settings.py", line 335, in <module>
     from local.local_settings import *  # noqa
   File "/usr/share/openstack-dashboard/openstack_dashboard/local/local_settings.py", line 131,
in <module>
     SECRET_KEY = secret_key.generate_or_read_from_file('/var/lib/openstack-
dashboard/secret_key') [Fri Aug 25 16:22:13.682611 2017] [wsgi:error] [pid 9401:tid    File
"/usr/share/openstack-dashboard/horizon/utils/secret_key.py", line 70, in
generate_or_read_from_file
     key = read_from_file(key_file)
   File "/usr/share/openstack-dashboard/horizon/utils/secret_key.py", line 52, in read_from_file
     with open(key_file, 'r') as f:
 IOError: [Errno 13] Permission denied: '/var/lib/openstack-dashboard/secret_key'
```

The log shows that the file cannot be accessed by horizon. Check its ownership and permission

```
root@controller:~# ll /var/lib/openstack-dashboard/secret_key
-rw------- 1 root root 64 Aug 25 15:49 /var/lib/openstack-dashboard/secret_key
```

try chown to `horizon:horizon`:

```
# chown -R horizon:horizon /var/lib/openstack-dashboard/secret_key
# ll /var/lib/openstack-dashboard/secret_key
-rw------- 1 horizon:horizon 64 Aug 25 15:49 /var/lib/openstack-dashboard/secret_key
# service apache2 reload
```

retry with `curl`, still got `500`,and with same error on apache error log

```
# curl http://10.20.0.10/horizon
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>500 Internal Server Error</title>
</head><body>
<h1>Internal Server Error</h1>
<p>The server encountered an internal error or
misconfiguration and was unable to complete
your request.</p>
<p>Please contact the server administrator at
 webmaster@localhost to inform them of the time this error occurred,
 and the actions you performed just before this error.</p>
<p>More information about this error may be available
in the server error log.</p>
<hr>
<address>Apache/2.4.18 (Ubuntu) Server at 10.20.0.10 Port 80</address>
</body></html>
```

try change as `777` while it cannot pass horizon permission policy with error as below:

```
# chmod 777 /var/lib/openstack-dashboard/secret_key
# service apache2 reload
# curl http://10.20.0.10/horizon

# less /var/log/apache2/error.log
...
[Fri Aug 25 16:20:38.336541 2017] [wsgi:error] [pid 9110:tid 140055698921216] [remote
10.20.0.10:18504]     SECRET_KEY = secret_key.generate_or_read_from_file('/var/lib/openstack-
dashboard/secret_key') [Fri Aug 25 16:20:38.336553 2017] [wsgi:error] [pid 9110:tid
140055698921216] [remote 10.20.0.10:18504]   File "/usr/share/openstack-
dashboard/horizon/utils/secret_key.py", line 70, in generate_or_read_from_file
[Fri Aug 25 16:20:38.336595 2017] [wsgi:error] [pid 9110:tid 140055698921216] [remote
10.20.0.10:18504]     key = read_from_file(key_file)
[Fri Aug 25 16:20:38.336603 2017] [wsgi:error] [pid 9110:tid 140055698921216] [remote
10.20.0.10:18504]   File "/usr/share/openstack-dashboard/horizon/utils/secret_key.py", line 51,
in read_from_file
[Fri Aug 25 16:20:38.336612 2017] [wsgi:error] [pid 9110:tid 140055698921216] [remote
10.20.0.10:18504]     os.path.abspath(key_file))
[Fri Aug 25 16:20:38.336628 2017] [wsgi:error] [pid 9110:tid 140055698921216] [remote
10.20.0.10:18504] FilePermissionError: Insecure permissions on key file /var/lib/openstack-
dashboard/secret_key, should be 0600.
...
```

We should identify the process owner and change to it accordingly.

By checking horizon wsgi process it's found the user is www-data (the one for apache2):

```
root@controller:~# ps -aux | grep horizon
www-data 10299  0.0  0.2 251032  8292 ?        Sl   16:26   0:00 (wsgi:horizon)    -k start
www-data 10300  0.0  0.2 251024  8292 ?        Sl   16:26   0:00 (wsgi:horizon)    -k start
www-data 10301  0.3  4.1 550344 169144 ?       Sl   16:26   0:02 (wsgi:horizon)    -k start
```

Solution

Change owner to `www-data`

```
# chown www-data /var/lib/openstack-dashboard/secret_key
# service apache2 reload
```

Retry with `curl http://10.20.0.10/horizon` no error came out :-).