Assignment 3 – Deploying a Mini-cloud via OpenStack

(Due by Midnight, Tuesday, December 5, 10% Grade)

1. Summary

In this assignment, we will deploy a mini cloud using OpenStack [2]. Due to limited resources, we will only explore a single node deployment. During this process, we will get familiar with the basic process of deploying a cloud environment and the key components of OpenStack.

2 Get to Know the Tools

In this assignment, we will use VMs from Watson School. You can access your VM via this document [1]. It's better to use SSH to access your VM when you install OpenStack, as the GUI is not working well. Please let TAs know if you have questions about accessing your VM.

3 OpenStack Deployment

An installation reference can be found here [2]. You will notice from the "Example Architecture" that, there are many "Nodes" for different services. We will install all the services within a single VM.. Before the installation, we should pay attention that:

(1) We will provide passwords to different services. For ease of this demo deployment, we could choose one simple password (e.g. "123456") for all services. Notice that, in a real deployment, we need to provide complex passwords to different services.
(2) In the end of each service deployment, there is a "verification" section. Make sure that you pass all verifications, and then move to the next service deployment. Otherwise, it will block further services.
(3) For each service deployment, we have to prepare a lot of "configurations", either in the database or in the configuration files. Make sure all the configurations are CORRECT. Only one small misconfiguration could block the whole system.
(4) In the interest of time, this document has provided the main steps involved in deploying the main services of OpenStack (e.g., identity service, image service, compute service, network service, dashboard). However, more detailed guidance and explanation are included in the installation reference [2].
(5) In the end of each service deployment, we will bring up that service. If you cannot pass the verification tasks. You could refer to the "logs" information for possible errors. You can find all log files under "/var/log/[service name]", e.g., "/var/log/nova".
(6) Try to master as many "openstack" commands as possible for you to debug the system. Use "openstack –help" to check all the commands.

3.1 Environment Preparation

First of all, ssh to your VM by following [1].

Add "your_vm_IP_address controller" to /etc/hosts
e.g., "192.168.0.143 controller"
We will use this IP information multiple times in this document. Please replace it (i.e., "192.168.0.143") with your VM's IP.

a. Install OpenStack packages:

Refer to: https://docs.openstack.org/ocata/install-guide-ubuntu/environment-packages.html

```
$sudo apt install software-properties-common
$sudo add-apt-repository cloud-archive:ocata
$sudo apt update && sudo apt dist-upgrade
$sudo apt install python-openstackclient
```

b. Install a database

Openstack stores information in a database.
(Refer to https://docs.openstack.org/ocata/install-guide-ubuntu/environment-sql-database.html)

$sudo apt install mariadb-server python-pymysql

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 (e.g., 192.168.0.143). Notice using your VM's IP address

```
[mysqld]
bind-address = 192.168.0.143

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

Restart the database and initialize the database:
$sudo service mysql restart
$mysql_secure_installation

For this step, you have to choose a password for root.
Also for "Disallow root login remotely?" choose no.

After this step, you can use the following commands to test your database:
$sudo mysql -uroot -p123456 (e.g., "123456" is my database's password)
If successful, you can login into your database.

Grant remote access privilege:
$mysql

```
GRANT ALL PRIVILEGES
ON *.*
TO 'root'@'%'
IDENTIFIED BY '123456'; (Your password)
```

After this, you can test using:
mysql -h192.168.0.143 -uroot -p123456

You should access your database.

c. Install a message queue

The components of Openstack use a message queue to communicate messages with each other.
Refer to https://docs.openstack.org/ocata/install-guide-ubuntu/environment-messaging.html.

Install the rabbitmy message queue
$sudo apt install rabbitmq-server
$rabbitmqctl add_user openstack RABBIT_PASS (i.e., your password)

Grant access permissions:
$sudo rabbitmqctl set_permissions openstack ".*" ".*" ".*"

d. Install Memcached

The Identity service authentication mechanism for services uses Memcached to cache tokens.
Please refer to https://docs.openstack.org/ocata/install-guide-ubuntu/environment-memcached.html.

$sudo 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.

```
-l 192.168.0.143  (notice, your ip)
```

Restart the memcached service.
$sudo service memcached restart

3.2 Identity Service

Please refer to this document for a good overview of Identity service:
https://docs.openstack.org/ocata/install-guide-ubuntu/common/get-started-identity.html
For the installation, please refer to this document:
https://docs.openstack.org/ocata/install-guide-ubuntu/keystone-install.html

a. Prepare database

Log to your database:
$sudo mysql

create keystone database:

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

Grant proper access to the keystone database:

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

Notice, replace 'KEYSTONE_DBPASS' with your own database password.
Exit the database access client.

b. Install packages:

$sudo apt install keystone

c. 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
```

Notice, replace KEYSTONE_DBPASS with your own password. Remove other connections.

In the `[token]` section, configure the Fernet token provider:

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

Populate the Identity service database:

```
$sudo keystone-manage db_sync
```

Initialize Fernet key repositories:

```
$sudo keystone-manage fernet_setup --keystone-user keystone --keystone-group keystone
$sudo keystone-manage credential_setup --keystone-user keystone --keystone-group keystone
```

Bootstrap the Identity service:

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

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

d. Finalize the installation

Remove the default SQLite database:

```
$sudo rm -f /var/lib/keystone/keystone.db
```

Configure the administrative account. You will use this account to run openstack command.
Create a "admin-openrc.sh" file (e.g., with name admin-openrc.sh)

```
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.

e. Create a domain, projects, users, and roles

Please https://docs.openstack.org/ocata/install-guide-ubuntu/keystone-users.html

Source the `admin` credentials to gain access to admin-only CLI commands. What does it mean?

$. admin-openrc.sh

This guide uses a service project that contains a unique user for each (OpenStack) service that you add to your environment.

create the `service` project:

```
$sudo apt-get install python-openstackclient
$openstack project create --domain default \
  --description "Service Project" service
```

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
```

Create the `demo` user:

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

Create the `user` role:

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

f. Verification

https://docs.openstack.org/ocata/install-guide-ubuntu/keystone-verify.html

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
```

No errors should expect.

Similarly, 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
```

3.3 Image service

Please refer to this document for an overview of image service:
https://docs.openstack.org/ocata/install-guide-ubuntu/common/get-started-image-service.html

a. Prepare database

$sudo mysql

Create the `glance` database:

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

Grant proper access to the `glance` database:

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

Replace GLANCE_DBPASS with your preferred password.
Exit the database access client.

Source the `admin` credentials to gain access to admin-only CLI commands:
$. admin-openrc.sh (created in step 1.d)

b. Create glance user

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

User Password:
Repeat User Password:

It will show something like:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | 3f4e777c4062483ab8d9edd7dff829df |
| name                | glance                           |
| options             | {}                               |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

Notice that again, for this demo installation, it's better to use the same password throughput the whole installation. (But in practice, it's better to use different, complex password).

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

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

Create the `glance` service entity:

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

Create the Image service API endpoints:

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

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

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

c. Install packages

$sudo apt install glance

d. 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 your database password.
Remove the sqlite_db connection, if it exists under [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_PASS

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

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

e. 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 your database password.

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_PASS

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

Replace GLANCE_DBPASS with your database password.

f. Populate the Image service database:

$sudo glance-manage db_sync

g. Restart the Image services:

$sudo service glance-registry restart
$sudo service glance-api restart

h. Verify installation

Refer to https://docs.openstack.org/ocata/install-guide-ubuntu/glance-verify.html

Download the source image:
wget http://download.cirros-cloud.net/0.3.5/cirros-0.3.5-x86_64-disk.img

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-0.3.5-x86_64-disk.img \
  --disk-format qcow2 --container-format bare \
  --public
```

Confirm upload of the image and validate attributes:
$openstack image list

The output should look like this:

```
+--------------------------------------+--------+--------+
| ID                                   | Name   | Status |
+--------------------------------------+--------+--------+
| 6f00b69a-a4bd-4b5b-9528-373ba2e085eb | cirros | active |
+--------------------------------------+--------+--------+
```

3.4 Compute service

Please refer to this document for an overview of the compute service:
https://docs.openstack.org/ocata/install-guide-ubuntu/common/get-started-compute.html

a. Install the controller part of compute service

1. **Prepare database**
   $sudo 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
```

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

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

   Create the `nova` service entity:

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

   Create the Compute API service endpoints:

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

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

$openstack endpoint create --region RegionOne \
```

```
  compute admin http://controller:8774/v2.1
```

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

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

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

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

Create the Placement API entry in the service catalog:

```
$openstack service create --name placement --description "Placement API" placement
```

Create the Placement API service endpoints:

```
$openstack endpoint create --region RegionOne placement public
http://controller:8778

$openstack endpoint create --region RegionOne placement internal
http://controller:8778

$openstack endpoint create --region RegionOne placement admin
http://controller:8778
```

4. **Install packages:**

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

5. **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_PASS
```

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

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

```
[DEFAULT]
# ...
my_ip = 192.168.0.143
```

Notice, this should be your IP address.

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

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

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.

6. **Populate the nova-api database:**
   $sudo nova-manage api_db sync

7. **Register the `cell0` database:**
   $sudo nova-manage cell_v2 map_cell0

8. **Create the `cell1` cell:**
   $sudo nova-manage cell_v2 create_cell --name=cell1 --verbose

9. **Populate the nova database:**
   $sudo nova-manage db sync

10. **Verify nova cell0 and cell1 are registered correctly:**

```
$sudo nova-manage cell_v2 list_cells
```

You should expect something like:

```
+-------+--------------------------------------+
|  Name |                 UUID                 |
+-------+--------------------------------------+
| cell0 | 00000000-0000-0000-0000-000000000000 |
| cell1 | 6878c89f-1c6b-45f6-abe8-c7c7fbfba475 |
+-------+--------------------------------------+
```

11. **Finalize installation**

```
$sudo service nova-api restart
$sudo service nova-consoleauth restart
$sudo service nova-scheduler restart
$sudo service nova-conductor restart
$sudo service nova-novncproxy restart
```

b. Install the compute part of compute service

As we have one single machine, we install the compute part on the same machine.
https://docs.openstack.org/ocata/install-guide-ubuntu/nova-compute-install.html

1. **Prepare your compute node:**
   Add "your_vm_IP_address controller" to /etc/hosts
   e.g., "192.168.0.143 controller"

   Install OpenStack packages:
   https://docs.openstack.org/ocata/install-guide-ubuntu/environment-packages.html
   sudo apt install software-properties-common
   sudo add-apt-repository cloud-archive:ocata
   sudo apt update && sudo apt dist-upgrade
   sudo apt install python-openstackclient

2. **Install packages**
   $sudo apt install nova-compute

3. **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 `[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_PASS
```

   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 (e.g., 192.168.0.143). You can check this IP address using “ifconfig”.

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

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

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
```

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.

Finalize the installation on the compute node

```
$sudo service nova-compute restart
```

c. Add the compute node to the cell database

1. $. admin-openrc.sh

2. $openstack hypervisor list
   You should observe compute node:

```
+----+-------------------------+-------+---------+
| ID | Hypervisor hostname     | State | Status  |
+----+-------------------------+-------+---------+
| 1  | CS480-580-huilu.csvb.local | up    | enabled |
+----+-------------------------+-------+---------+
```

3. nova-manage cell_v2 discover_hosts --verbose

```
Found 2 cell mappings.
Skipping cell0 since it does not contain hosts.
Getting compute nodes from cell 'cell1': 6878c89f-1c6b-45f6-abe8-c7c7fbfba475
Found 1 computes in cell: 6878c89f-1c6b-45f6-abe8-c7c7fbfba475
Checking host mapping for compute host 'CS480-580-huilu': 6a158d51-12a7-4877-977f-1b6e3c1cf39a
Creating host mapping for compute host 'CS480-580-huilu': 6a158d51-12a7-4877-977f-1b6e3c1cf39a
```

d. Verify the installation

Refer to https://docs.openstack.org/ocata/install-guide-ubuntu/nova-verify.html

First run ". admin-openrc"
Then run "$openstack compute service list"
You should observe the following results:

```
+----+----------------+----------------+----------+---------+-------+----------------------------+
| ID | Binary         | Host           | Zone     | Status  | State | Updated At                 |
+----+----------------+----------------+----------+---------+-------+----------------------------+
|  3 | nova-consoleauth | ubuntu         | internal | enabled | up    | 2017-11-25T23:19:13.000000 |
|  4 | nova-scheduler   | ubuntu         | internal | enabled | up    | 2017-11-25T23:19:05.000000 |
|  5 | nova-conductor   | ubuntu         | internal | enabled | up    | 2017-11-25T23:19:07.000000 |
|  6 | nova-compute     | CS480-580-huilu | nova     | enabled | up    | 2017-11-25T23:19:09.000000 |
+----+----------------+----------------+----------+---------+-------+----------------------------+
```

Other tests:

1. List API endpoints in the Identity service to verify connectivity with the Identity service:
   $openstack catalog list

2. List images in the Image service to verify connectivity with the Image service:

$openstack image list

3. Check the cells and placement API are working successfully:
   $nova-status upgrade check

3.5 Network service

Please refer to the document for detailed overview: https://docs.openstack.org/ocata/install-guide-ubuntu/common/get-started-networking.html

a. Install the controller part of network service

1. Prepare database

$sudo mysql

Create the `neutron` database:

```
MariaDB [(none)] CREATE DATABASE neutron;
```

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

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

2. $. admin-openrc

3. Create the `neutron` user:

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

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

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

Create the `neutron` service entity:

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

Create the Networking service API endpoints:

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

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

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

4. There are many network options. For this task, we only choose 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. Please refer to this document for provider network information: https://docs.openstack.org/ocata/install-guide-ubuntu/neutron-controller-install-option1.html

5. **Install packages:**

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

6. **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.

Note: 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_PASS
```

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

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_PASS
```

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

7. 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
   ```

   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
```

8. 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.

   Notice that, as we only have one "physical" network interface, which has the IP of 192.168.0.143, we will create a "fake" network interface as the provider network interface (e.g., eth10):

   (a) sudo lsmod | grep dummy (make sure dummy is loaded)
   (b) sudo modprobe dummy (load it if it does not exist)
   (c) sudo lsmod | grep dummy (you should see the output)
   (d) sudo ip link set name eth10 dev dummy0 (create the fake interface)
   (e) ip link show eth10 (confirm it)

   The replace PROVIDER_INTERFACE_NAME with eth10.

   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
```

9. Edit the /etc/neutron/dhcp_agent.ini
   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 = linuxbridge
dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
enable_isolated_metadata = true
```

10. Edit the `/etc/neutron/metadata_agent.ini`

    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 (e.g., same password as other services) for the metadata proxy.

    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_PASS
service_metadata_proxy = true
metadata_proxy_shared_secret = METADATA_SECRET
```

    Replace `NEUTRON_PASS` 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.

11. Populate the database:
    sudo neutron-db-manage --config-file /etc/neutron/neutron.conf --config-file /etc/neutron/plugins/ml2/ml2_conf.ini upgrade head

12. Restart the Compute API service:

```
$sudo service nova-api restart
```

13. Restart the Networking services.

```
$sudo service neutron-server restart
$sudo service neutron-linuxbridge-agent restart
$sudo service neutron-dhcp-agent restart
$sudo service neutron-metadata-agent restart
```

b. Install the compute part of the network service

Because we use a single node. We will stay on the same machine.

1. **Install packages**

```
$sudo apt install neutron-linuxbridge-agent
```

2. Edit the `/etc/neutron/neutron.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. (Skip it if we have it)

   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_PASS
```

   Replace `NEUTRON_PASS` with the password you chose for the `neutron` user in the Identity service. (Skip it if we have)

3. Edit the `/etc/neutron/plugins/ml2/linuxbridge_agent.ini`

   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. (Skip it if we have)

   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
```

4. 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_PASS
```

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

5. Restart compute service:
   $sudo service nova-compute restart

6. Restart network bridge agent:
   sudo service neutron-linuxbridge-agent restart

c. Verify installation

Please refer to: https://docs.openstack.org/ocata/install-guide-ubuntu/neutron-verify.html

1. . admin-openrc
2. openstack extension list --network

You should receive a lot of information after a while.

3.6 Dashboard

Please refer to the document for an overview:
https://docs.openstack.org/ocata/install-guide-ubuntu/horizon.html

a. Install packages:

$sudo apt install openstack-dashboard

b. 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"
```

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',
    }
}
```

Note that 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"
```

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,
}
```

c. $sudo service apache2 reload

d. You may need to fixed permission issue:

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

e. Verify installation

Please refer to: https://docs.openstack.org/ocata/install-guide-ubuntu/horizon-verify.html

Switch to your GUI (refer to [1]). Access the dashboard using a web browser at http://controller/horizon (login into your native machine and open firefox). Authenticate using admin or demo user and 'default' domain credentials.

Notice that, the dashboard may not work due to some issues. Please report your situations.

3.7 Spawn an Instance

a. Create Network

Please refer to: https://docs.openstack.org/ocata/install-guide-ubuntu/launch-instance-networks-provider.html

1. Create a network

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

2. Create a subnetwork associated with the above nework

```
$openstack subnet create --network provider \
  --allocation-pool start=192.168.1.101,end=192.168.1.250 \
  --gateway 192.168.1.1 \
  --subnet-range 192.168.1.0/24 provider
```

3. Verify it

```
$openstack network list
```

b. Create a flavor

```
$openstack flavor create --id 0 --vcpus 1 --ram 256 --disk 1 m1.nano
$oepnstack flavor list
```

c. Generate a key pair

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

d. Add security group rules

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

e. Permit secure shell (SSH) access:

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

f. Launch an instance (without network connection)

$openstack server create --flavor m1.nano --image cirros --security-group default --key-name mykey provider-instance

Notice that you may need to do the following things to solve the "no valid hosts" issue.

```
rmmod kvm_intel
rmmod kvm
modprobe kvm
modprobe kvm_intel
```

a. Launch an instance with network

$openstack server create --flavor m1.nano --image cirros --nic net-id=55c58406-6664-43e8-81a2-3a52b5ac2f19 --security-group default --key-name mykey provider-instance

b. Verify success of instance launching

$nova list

Or

$openstack server list

4 Submission & Grading

You will demo your OpenStack deployment in person with TAs. We will announce the time later. Make sure you pass all verification tasks throughout this documentation.

• Identify service – 10 points.

• Image service – 10 points.

• Compute service – 20 points.

• Network service – 20 points.

• Dashboard service – 10 points.

• Spawn an instance – 30 points.

• Bonus – 10 points. You can get the bonus points, if you deploy block service (Cinder), and demo its correctness – 10 points

References

[1] How to access Watson School's VM. http://www.cs.binghamton.edu/~huilu/assignments/watsonschoolvm.pdf

[2] OpenStack Installation Guidance. https://docs.openstack.org/ocata/install-guide-ubuntu/overview.html#example-architecture