**Network Time Protocol (NTP)**

* + - Install Chrony packages:

# apt-get install chrony

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

# server $NTP\_server\_IP iburst

* + - Start NTP service and configure it to start when the system boots:

# service chrony restart

* + - Verify NTP synchronization:

# chronyc sources

**OpenStack packages**

* + - Enable the OpenStack repository

# apt-get install software-properties-common

# add-apt-repository cloud-archive:mitaka

* + - Upgrade the packages:

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

* + - Install the OpenStack client

# apt-get install python-openstackclient

**MySQL database**

* + - Install the packages:

# apt-get install mariadb-server python-pymysql

* + - Configure the /etc/mysql/conf.d/openstack.cnf with the following:

[mysqld]

bind-address = $Server\_IP\_or\_Hostname

default-storage-engine = innodb

innodb\_file\_per\_table = on

max\_connections = 4096

collation-server = utf8\_general\_ci

character-set-server = utf8

* + - Start the database service and configure it to start when the system boots:

# service mysql restart

* + - Secure the database service by running the mysql\_secure\_installation script. Choose a suitable password for the database root account:

# mysql\_secure\_installation

**Message queue**

* + - Install the packages:

# apt-get install rabbitmq-server

* + - Create the openstack user:

# rabbitmqctl add\_user openstack $your\_password\_here

* + - Permit configuration, write, and read access for the openstack user:

# rabbitmqctl set\_permissions openstack ".\*" ".\*" ".\*"

**Memcached**

* + - Install the packages:

# apt-get install memcached python-memcached

* + - Edit the /etc/sysconfig/memcached file and configure the service to use the management IP address of the controller:

-l $server\_IP\_or\_Hostname (change existing)

* + - Start the Memcached service and configure it to start when the system boots:

# service memcached restart

**Identity service**

* + - Create the keystone database and grab proper access:

mysql -u root –p

MariaDB [(none)]> CREATE DATABASE keystone;

MariaDB [(none)]> GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'localhost' \

IDENTIFIED BY '$password\_here';

MariaDB [(none)]> GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'%' \

IDENTIFIED BY '$password\_here';

* Disable the keystone service from starting automatically after installation:

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

* + - Install the packages:

# apt-get install keystone apache2 libapache2-mod-wsgi

* + - Edit the /etc/keystone/keystone.conf file and complete the following actions:

[DEFAULT]

admin\_token = $token\_password (your choice or create a random one)

[database]

connection = mysql+pymysql://keystone:$password\_here@$controller\_IP\_or\_hostname/keystone

[token]

provider = fernet

* + - Populate the Identity service database:

# su -s /bin/sh -c "keystone-manage db\_sync" keystone

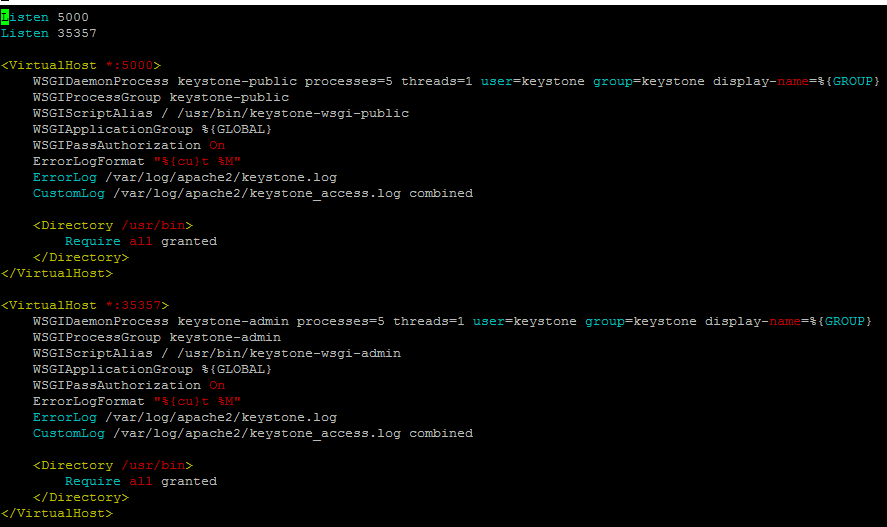
* + - Initialize Fernet key repositories:

# keystone-manage fernet\_setup --keystone-user keystone --keystone-group keystone

* + - Edit the /etc/apache2/apache2.conf file and add the ServerName option to reference the controller node:

ServerName $controller\_IP\_or\_Hostname

* Create the /etc/apache2/sites-available/wsgi-keystone.conf file with the following content:



* + - Enable the Identity service virtual hosts:

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

* + - Start the Apache HTTP service and configure it to start when the system boots:

# service apache2 restart

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

* + - Configure the administrative account:

# export OS\_TOKEN=$token\_password (as you set before)

# export OS\_URL=http://$controller\_IP\_or\_Hostname:35357/v3

# export OS\_IDENTITY\_API\_VERSION=3

* + - Create the service project. It contains a unique user for each service that you add to your environment:

# openstack project create --domain default --description "Service Project" service

* + - Create the Identity service API endpoints:

# openstack endpoint create --region RegionOne identity public\ <http://$controller_IP_or_Hostname:5000/v3>

# openstack endpoint create --region RegionOne identity internal\ <http://$controller_IP_or_Hostname:5000/v3>

# openstack endpoint create --region RegionOne identity admin\ <http://$controller_IP_or_Hostname:5000/v3>

* + - Create the default domain:

# openstack domain create --description "Default Domain" default

* + - Create the admin project:

# openstack project create --domain default --description "Admin Project" admin

* + - Create the admin user and set a password of your choice:

# openstack user create --domain default --password-prompt admin

* + - Create the admin role:

# openstack role create admin

* + - Add the admin role to the admin project and user:

# openstack role add --project admin --user admin admin

* + - Create the service project:

# openstack project create --domain default --description "Service Project" service

* + - 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\_IP\_or\_Hostname:35357/v3 \

--os-project-domain-name default --os-user-domain-name default \

--os-project-name admin --os-username admin token issue

* + - Create and 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=$password\_here (from section 1)

export OS\_AUTH\_URL=http://$controller\_IP\_or\_Hostname:35357/v3

export OS\_IDENTITY\_API\_VERSION=3

export OS\_IMAGE\_API\_VERSION=2

* + - Using the script:

# source admin-openrc

# openstack token issue

**Image Service**

* + - Create the glance database and grab proper access:

# mysql -u root –p

MariaDB [(none)]> CREATE DATABASE glance;

MariaDB [(none)]> GRANT ALL PRIVILEGES ON glance.\* TO 'glance'@'localhost' \

IDENTIFIED BY '$glance\_db\_password';

MariaDB [(none)]> GRANT ALL PRIVILEGES ON glance.\* TO 'glance'@'%' \

IDENTIFIED BY '$glance\_db\_password';

* + - Source the admin credentials:

# source admin-openrc

* + - Create the glance user and set a password of your choice:

# openstack user create --domain default --password-prompt glance

* + - 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_IP_or_Hostname:9292>

# openstack endpoint create --region RegionOne image internal <http://$controller_IP_or_Hostname:9292>

# openstack endpoint create --region RegionOne image admin <http://$controller_IP_or_Hostname:9292>

* + - Install the packages:

# apt-get install glance

* + - Edit the /etc/glance/glance-api.conf file and complete the following actions:

[database]

connection = mysql+pymysql://glance:$glance\_db\_password@$controller\_IP\_or\_Hostname/glance

[keystone\_authtoken]

auth\_uri = http://$controller\_IP\_or\_Hostname:5000

auth\_url = http://$controller\_IP\_or\_Hostname:35357

memcached\_servers = $controller\_IP\_or\_Hostname:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = glance

password = GLANCE\_PASS

[paste\_deploy]

flavor = keystone

[glance\_store]

stores = file,http

default\_store = file

filesystem\_store\_datadir = /var/lib/glance/images/

* + - Edit the /etc/glance/glance-registry.conf file and complete the following actions:

[database]

connection = mysql+pymysql://glance:$glance\_db\_password@$controller\_IP\_or\_Hostname/glance

[keystone\_authtoken]

auth\_uri = http://$controller\_IP\_or\_Hostname:5000

auth\_url = http://$controller\_IP\_or\_Hostname:35357

memcached\_servers = $controller\_IP\_or\_Hostname:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = glance

password = $glance\_user\_password

[paste\_deploy]

flavor = keystone

* + - Populate the Image service database:

# su -s /bin/sh -c "glance-manage db\_sync" glance

* + - Start the Image services and configure them to start when the system boots:

# service glance-registry restart

# service glance-api restart

* + - Verify operation

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

**Compute service**

* + - Create the compute database and grab proper access:

# mysql -u root –p

MariaDB [(none)]> CREATE DATABASE nova\_api;

MariaDB [(none)]> CREATE DATABASE nova;

MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova\_api.\* TO 'nova'@'localhost' \

IDENTIFIED BY '$nova\_db\_password';

MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova\_api.\* TO 'nova'@'%' \

IDENTIFIED BY '$nova\_db\_password';

MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova.\* TO 'nova'@'localhost' \

IDENTIFIED BY '$nova\_db\_password';

MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova.\* TO 'nova'@'%' \

IDENTIFIED BY '$nova\_db\_password';

* + - Source the admin credentials:

# source admin-openrc

* + - Create the nova user and set a password of your choice:

# 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\_IP\_or\_Hostname: 8774/v2.1/%\(tenant\_id\)s](http://$controller_IP_or_Hostname:8774/v2.1)

# openstack endpoint create --region RegionOne compute internal [http://$controller\_IP\_or\_Hostname: 8774/v2.1/%\(tenant\_id\)s](http://$controller_IP_or_Hostname:8774/v2.1)

# openstack endpoint create --region RegionOne compute admin [http://$controller\_IP\_or\_Hostname: 8774/v2.1/%\(tenant\_id\)s](http://$controller_IP_or_Hostname:8774/v2.1)

* + - Install the packages:

# apt-get install nova-api nova-conductor nova-consoleauth nova-novncproxy \

nova-scheduler nova-compute

* + - Edit the /etc/nova/nova.conf file and complete the following actions:

[DEFAULT]

enabled\_apis = osapi\_compute,metadata

[api\_database]

connection = mysql+pymysql://nova:$nova\_db\_password@$controller\_IP\_or Hostname/nova\_api

[database]

connection = mysql+pymysql://nova:$nova\_db\_password@$controller\_IP\_or\_Hostname/nova

[DEFAULT]

auth\_strategy = keystone

rpc\_backend = rabbit

my\_ip = $controller\_IP\_or\_Hostname

use\_neutron = True

firewall\_driver = nova.virt.firewall.NoopFirewallDriver

[oslo\_messaging\_rabbit]

rabbit\_host = $controller\_IP\_or\_Hostname

rabbit\_userid = openstack

rabbit\_password = $rabbit\_user\_password

[keystone\_authtoken]

auth\_uri = http://$controller\_IP\_or\_Hostname:5000

auth\_url = http://$controller\_IP\_or\_Hostname:35357

memcached\_servers = $controller\_IP\_or\_Hostname:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = nova

password = $nova\_user\_password

[vnc]

enabled = true

vncserver\_listen = 0.0.0.0

vncserver\_proxyclient\_address = $my\_ip

novncproxy\_base\_url = http://$controller\_IP\_or\_Hostname:6080/vnc\_auto.html

[glance]

api\_servers = <http://$controller_IP_or_Hostname:9292>

[oslo\_concurrency]

lock\_path = /var/lib/nova/tmp

* + - Populate the compute databases:

# su -s /bin/sh -c "nova-manage api\_db sync" nova

# su -s /bin/sh -c "nova-manage api\_db sync" nova

* + - Start the Compute services and configure them to start when the system boots:

# service nova-api restart

# service nova-consoleauth restart

# service nova-scheduler restart

# service nova-conductor restart

# service nova-novncproxy restart

# service nova-compute restart

* + - Verify operation:

# openstack compute service list

**Network service**

* + - Create the glance database and grab proper access:

# mysql -u root –p

MariaDB [(none)]> CREATE DATABASE neutron;

MariaDB [(none)]> GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'localhost' \

IDENTIFIED BY '$neutron\_db\_password';

MariaDB [(none)]> GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'%' \

IDENTIFIED BY '$neutron\_db\_password';

* + - Create the neutron user and set a password of your choice:

# 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 neutro --description "OpenStack Networking"\ network

* + - Create the Networking service API endpoints:

# openstack endpoint create --region RegionOne network public\ <http://$controller_IP_or_Hostname:9696>

# openstack endpoint create --region RegionOne network internal\ <http://$controller_IP_or_Hostname:9696>

# openstack endpoint create --region RegionOne network admin\ <http://$controller_IP_or_Hostname:9696>

* + - Install and configure the Networking components:

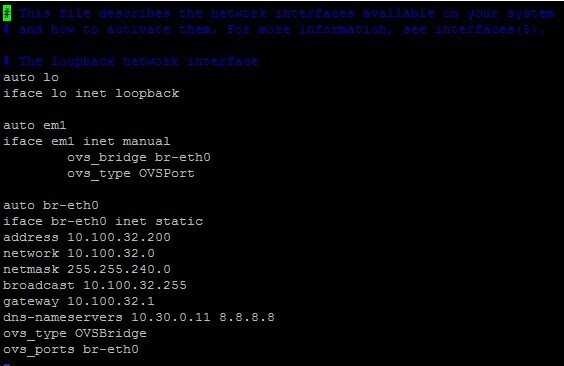
# apt-get install neutron-server neutron-plugin-ml2 \

neutron-openvswitch-agent neutron-l3-agent neutron-dhcp-agent \

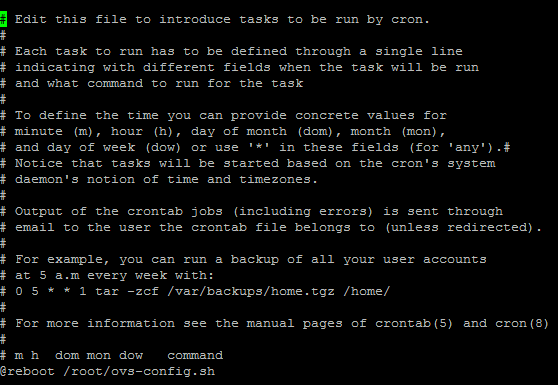
neutron-metadata-agent

* + - **HERE YOU RUN THE SCRIPT WE HAVE CREATED (ovs-config.sh). ADD IT TO RUN ON BOOT**. Follow the following steps:

1) set the /etc/network/interfaces file as bellow (using your network IPs):

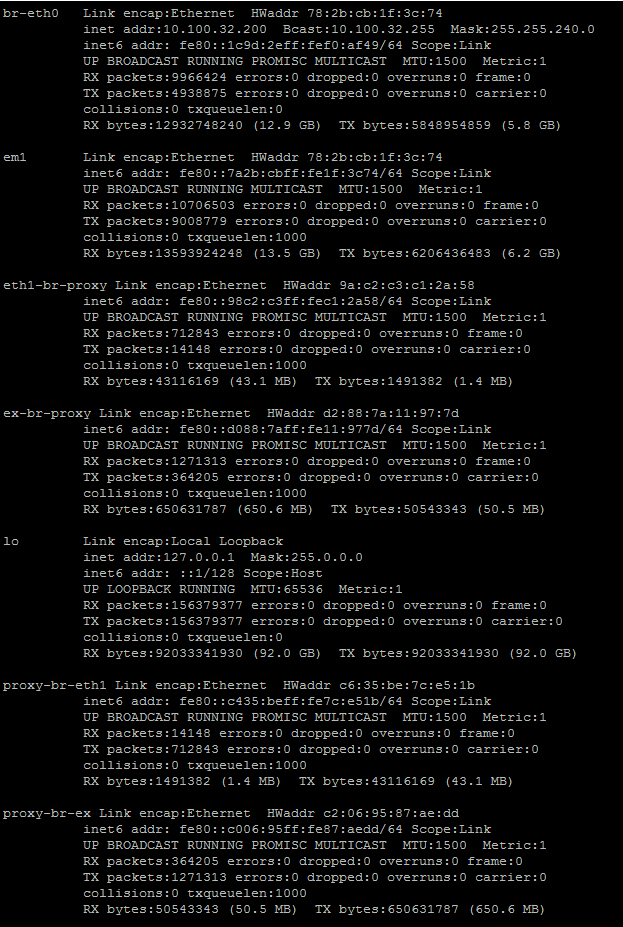


2) Add the script file in crontab in order to run in on boot



3) Reboot the PC. The script should run every time you reboot the PC.

The output of ifconfig should be as follows:



* + - Edit the /etc/neutron/neutron.conf file:

[database]

connection = mysql+pymysql://neutron:$neutron\_db\_password@$controller\_IP\_or\_Hostname/neutron

[DEFAULT]

core\_plugin = ml2

service\_plugins = router

allow\_overlapping\_ips = True

rpc\_backend = rabbit

auth\_strategy = keystone

notify\_nova\_on\_port\_status\_changes = True

notify\_nova\_on\_port\_data\_changes = True

[oslo\_messaging\_rabbit]

rabbit\_host = $controller\_IP\_or\_Hostname

rabbit\_userid = openstack

rabbit\_password = $rabbit\_user\_password

[keystone\_authtoken]

auth\_uri = http://$controller\_IP\_or\_Hostname:5000

auth\_url = http://$controller\_IP\_or\_Hostname:35357

memcached\_servers = $controller\_IP\_or\_Hostname:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = neutron

password = $neutron\_user\_password

[nova]

auth\_url = http://$controller\_IP\_or\_Hostname:35357

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

region\_name = RegionOne

project\_name = service

username = nova

password = $nova\_user\_password

* + - Edit the /etc/neutron/plugins/ml2/ml2\_conf.ini:

[ml2]

type\_drivers = flat,vlan

tenant\_network\_types = vlan,flat

mechanism\_drivers = openvswitch

extension\_drivers = port\_security

[ml2\_type\_flat]

flat\_networks = External

[ml2\_type\_vlan]

network\_vlan\_ranges = Intnet1:100:200

* + - Edit the /etc/neutron/plugins/ml2/openvswitch\_agent.ini

[ovs]

local\_ip = $controller\_IP\_or\_Hostname

bridge\_mappings = External:br-ex,Intnet1:br-eth1

[securitygroup]

firewall\_driver = neutron.agent.firewall.NoopFirewallDriver

enable\_security\_group = False

enable\_ipset = true

* + - Edit the /etc/neutron/l3\_agent.ini:

[DEFAULT]

use\_namespaces = True

interface\_driver = neutron.agent.linux.interface.OVSInterfaceDriver

* + - Edit the /etc/neutron/dhcp\_agent.ini:

[DEFAULT]

use\_namespaces = True

interface\_driver = neutron.agent.linux.interface.OVSInterfaceDriver

dhcp\_driver = neutron.agent.linux.dhcp.Dnsmasq

* + - Edit the /etc/neutron/metadata\_agent.ini:

[DEFAULT]

nova\_metadata\_ip = $controller\_IP\_or\_Hostname

metadata\_proxy\_shared\_secret = $password\_of\_your\_choice

* + - Edit the /etc/nova/nova.conf and add the following lines:

[neutron]

url = http://$controller\_IP\_or\_Hostname:9696

auth\_url = http://$controller\_IP\_or\_Hostname:35357

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

region\_name = RegionOne

project\_name = service

username = neutron

password = $neutron\_user\_password

service\_metadata\_proxy = True

metadata\_proxy\_shared\_secret = $password\_as\_above

* + - 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

# service nova-api restart

# service neutron-server restart

# service neutron-linuxbridge-agent restart

# service neutron-dhcp-agent restart

# service neutron-metadata-agent restart

# service neutron-l3-agent restart

* + - Verify operation

# neutron ext-list

# neutron agent-list

* + - Dashboard

# apt-get install openstack-dashboard

* + - Edit the /etc/openstack-dashboard/local\_settings.py

OPENSTACK\_HOST = "$controller\_IP\_or\_Hostname"

ALLOWED\_HOSTS = ['\*', ]

CACHES = {

'default': {

'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',

'LOCATION': '$controller\_IP\_or\_Hostname:11211',

}

}

OPENSTACK\_KEYSTONE\_URL = "http://%s:5000/v3" % OPENSTACK\_HOST

OPENSTACK\_KEYSTONE\_MULTIDOMAIN\_SUPPORT = True

OPENSTACK\_API\_VERSIONS = {

"identity": 3,

"image": 2,

"volume": 2,

}

OPENSTACK\_KEYSTONE\_DEFAULT\_DOMAIN = "default"

OPENSTACK\_KEYSTONE\_DEFAULT\_ROLE = "user"

Optionally, configure the time zone. Replace TIME\_ZONE with an appropriate time zone identifier:

TIME\_ZONE = "TIME\_ZONE"

* + - Reload the web server configuration:

# service apache2 reload

* + - Verify operation

Access the dashboard using a web browser at <http://$controller_IP_or_Hostname/horizon>

Authenticating using admin user and default domain credentials

**Orchestration service**

* + - Create the glance database and grab proper access:

# mysql -u root –p

MariaDB [(none)]> CREATE DATABASE heat;

MariaDB [(none)]> GRANT ALL PRIVILEGES ON heat.\* TO 'heat'@'localhost' \

IDENTIFIED BY '$heat\_db\_password';

MariaDB [(none)]> GRANT ALL PRIVILEGES ON heat.\* TO 'heat'@'%' \

IDENTIFIED BY '$heat\_db\_password';

* + - Create the heat user and set a password of your choice:

# openstack user create --domain default --password-prompt heat

* + - Add the admin role to the heat user:

# openstack role add --project service --user heat admin

* + - Create the heat and heat-cfn service entities:

# openstack service create --name heat --description "Orchestration" orchestration

# openstack service create --name heat-cfn --description "Orchestration" \ cloudformation

* + - Create the Orchestration service API endpoints:

# openstack endpoint create --region RegionOne \

orchestration public [http://$controller\_IP\_or\_Hostname:8004/v1/%\(tenant\_id\)s](http://$controller_IP_or_Hostname:8004/v1/%25\(tenant_id\)s)

# openstack endpoint create --region RegionOne \

Orchestration internal [http://$controller\_IP\_or\_Hostname:8004/v1/%\(tenant\_id\)s](http://$controller_IP_or_Hostname:8004/v1/%25\(tenant_id\)s)

# openstack endpoint create --region RegionOne \

orchestration admin http://$controller\_IP\_or\_Hostname:8004/v1/%\(tenant\_id\)s

# openstack endpoint create --region RegionOne cloudformation public\ <http://$controller_IP_or_Hostname:8000/v1>

# openstack endpoint create --region RegionOne cloudformation internal\ <http://$controller_IP_or_Hostname:8000/v1>

# openstack endpoint create --region RegionOne cloudformation admin\ <http://$controller_IP_or_Hostname:8000/v1>

* + - Create the heat domain that contains projects and users for stacks:

# openstack domain create --description "Stack projects and users" heat

* + - Create the heat\_domain\_admin user to manage projects and users in the heat domain. Set a password of your choice:

# openstack user create --domain heat --password-prompt 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

* + - Create the heat\_stack\_owner role:

# openstack role create heat\_stack\_owner

* + - Create the heat\_stack\_user role:

# openstack role create heat\_stack\_user

* + - Install the packages:

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

* + - Edit the /etc/heat/heat.conf:

[database]

connection = mysql+pymysql://heat:$heat\_db\_pass@controller\_IP\_or\_Hostname/heat

[DEFAULT]

rpc\_backend = rabbit

heat\_metadata\_server\_url = http:// $controller\_IP\_or\_Hostname:8000

heat\_waitcondition\_server\_url = <http://$controller_IP_or_Hostname:8000/v1/waitcondition>

stack\_domain\_admin = heat\_domain\_admin

stack\_domain\_admin\_password = $heat\_domain\_admin\_user\_password

stack\_user\_domain\_name = heat

[oslo\_messaging\_rabbit]

rabbit\_host = controller\_IP\_or\_Hostname

rabbit\_userid = openstack

rabbit\_password = $rabbit\_user\_password

[keystone\_authtoken]

auth\_uri = http:// $controller\_IP\_or\_Hostname:5000

auth\_url = http:// $controller\_IP\_or\_Hostname:35357

memcached\_servers = $controller\_IP\_or\_Hostname:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = heat

password = $heat\_user\_password

[trustee]

auth\_plugin = password

auth\_url = http://$controller\_IP\_or\_Hostname:35357

username = heat

password = $heat\_user\_password

user\_domain\_name = default

[clients\_keystone]

auth\_uri = <http://$controller_IP_or_Hostname:35357>

[ec2authtoken]

auth\_uri = <http://$controller_IP_or_Hostname:5000/v2.0>

* + - Populate the Orchestration database:

# su -s /bin/sh -c "heat-manage db\_sync" heat

* + - Restart the Orchestration services:

# service heat-api restart

# service heat-api-cfn restart

# service heat-engine restart

* + - Verify operation

# openstack orchestration service list

**Launch an Instance**

* + - Using Dashboard or cli

Firstly, you need to create an external and at least an internal network.

**External network creation:**

# neutron net-create --shared --provider:physical\_network External \ --provider:network\_type flat $external\_network\_name

neutron subnet-create --name $external\_subnet\_name --allocation-pool\ start=$start\_ip\_address,end=$end\_ip\_address --dns-nameserver $DNS\_RESOLVER \ --gateway $external\_network\_gateway $external\_network\_name \ $your\_external\_network\_CIDR

If we assume that you use a 192.168.0.0/24?(as external network), the gateway should be 192.168.0.1 (the connection to your router).

**Internal network creation:**

neutron net-create $network\_name

neutron subnet-create --name $subnet\_name --dns-nameserver $DNS\_RESOLVER \ --gateway $internal\_network\_gateway $network\_name $internal\_network\_CIDR

**Router creation:**

Add the router: external option to the external network:

# neutron net-update External --router:external

Create the router:

# neutron router-create $router\_name

Add the internal network subnet as an interface on the router:

# neutron router-interface-add $router\_name $internal\_subnet\_name

Set a gateway on the External network on the router:

# neutron router-gateway-set $router\_name $external\_network\_name

After you finish with the above creations, you are ready to launch instances, either via dashboard or cli.

* + - Using Orchestration service refer to the following ling:

https://docs.openstack.org/mitaka/install-guide-ubuntu/launch-instance-heat.html