一、 openstack 简介

OpenStack Mitaka是为建立云，第13个发行的配置开源软件，现在提供更高的可管理性和可扩展性以及更好的用户体验。2336个来自345个组织的开发者，操作 者，用户的国际组织设计和建立了Mitaka。OpenStack作为一个集成引擎，虚拟机和带单一的API集的容器orchestration框架。已 经成为企业和服务器提供商的云平台选择之一。

二、 openstack mitaka搭建

2.1 基础环境准备

2.1.1所有节点使用网页基础环境安装

controller node 127.0.0.1

compute node 127.0.0.1

2.1.2 环境准备

1)配置所有节点hostname

#vi /etc/hostname

controller(controller node)

compute(compute node)

2)配置/etc/hosts文件，添加：

127.0.0.1 controller

127.0.0.1 compute

3)配置NTP

#yum –y install chrony

所有节点执行

#vi /etc/chrony.conf

server controller iburst

时间服务器不需要添加

#systemctl enable chronyd.service

#systemctl start chronyd.service

添加开机自启动并启动服务

#chronyc sources

验证

4）关闭防火墙

#systemctl stop firewalld

#systemctl disable firewalld。service

5）关闭SElinux

#vi /etc/selinux/config

SElinux=disable

将enforcing替换为disable

2.1.2openstack安装包准备

1）安装centos提供的yum仓库

#yum install centos-release-openstack-mitaka

2 )更新

# yum upgrade

3）安装配置工具

#yum install python-openstackclient

openstackclient（又名OSC）是一种用于OpenStack， 带来集计算、身份、形象的命令的命令行客户端，对象存储和块存储API 一起在一个统一的指挥 结构单壳。

4）安装openstack-selinux

#yum install openstack-selinux

CentOS默认启用SELinux。安装openstack-selinux包自动管理安全策略OpenStack服务:

2.2 database service

2.2.1在controller节点安装mariadb

1) 安装mariadb数据库

# yum install mariadb mariadb-server python2-PyMySQL

2）创建/etc/my.cnf.d/openstack.cnf文件

# vi /etc/my.cnf.d/openstack.cnf

[mysqld]

bind-address = 127.0.0.1

default-storage-engine = innodb

innodb\_file\_per\_table

collation-server = utf8\_general\_ci

character-set-server = utf8

3）启动服务

# systemctl enable mariadb.service

# systemctl start mariadb.service

4）设置登录密码

# mysql\_secure\_installation

2.2.2 在controller安装消息队列服务

OpenStack使用消息队列来协调操作和状态信息等服务。消息队列服务通常控制器节点上运行。OpenStack支持多种消息队列服务,包括RabbitMQ Qpid,ZeroMQ。然而,大多数发行版包OpenStack支持一个特定消息队列服务。

1) 安装消息队列服务

#yum install rabbitmq-server

2）启动服务

# systemctl enable rabbitmq-server.service

# systemctl start rabbitmq-server.service

3）添加openstack用户

# rabbitmqctl add\_user openstack RABBIT\_PASS

4）配置openstack权限

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

2.2.3 安装Memcached缓存服务

identity服务令牌使用Memcached缓存。

1） 安装Memchched

#yum install memcached python-memcached

2） 启动服务

# systemctl enable memcached.service

# systemctl start memcached.service

2.3安装identity服务

做为 OpenStack 云系统的入口，Keystone 提供了云系统入口所需要的许多功能：

1）用户身份验证：系统得知道用户是不是合法的用户。为此，Keystone 需要对 user 进行管理和保存；管理用户相关的 tenant、role、group 和 domain等；用户 credential 的存放、验证、令牌管理等。

2） 服务目录列表：用户需要知道系统提供的服务目录。为此，Keystone 得提供服务目录的管理，包括 service、endpoint 等。

2.3.1创建keystone数据库服务

1）登陆数据库

# mysql –u root –p

2）创建keystone数据库

Mariadb> CREATE DATABASE keystone;

3） 创建并授权用户

Mariadb>GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'localhost'\

IDENTIFIED BY 'KEYSTONE\_DBPASS';

Mariadb>GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'%' \

IDENTIFIED BY 'KEYSTONE\_DBPASS';

2.3.2安装并配置keystone

1）安装keystone

# yum install openstack-keystone httpd mod\_wsgi

2）生成一个随机值作为配置期间的管理令牌

#openssl rand –hex 10

3）编辑/etc/keystone/keystone.conf

 在[DEFAULT]部分添加初始管理令牌的值

admin\_token = ADMIN\_TOKEN（将这部分替换为生成的随机值）

 在[database]部分添加数据库

connection = mysql+pymysql://keystone:11qqaazz@controller/keystone

 在[token]部分添加供应商

provider = fernet

4） 填充identity数据库

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

5） 初始化fernet密钥

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

2.3.3配置apache http服务

1）在/etc/httpd/conf/httpd.conf里添加

ServerName controller

2）创建/etc/httpd/conf.d/wsgi-keystone.conf

Listen 5000

Listen 35357

<VirtualHost \*:5000>

WSGIDaemonProcess keystone-public processes=5 threads=1 user=keystone group=keystone display-name=%{GROUP}

WSGIProcessGroup keystone-public

WSGIScriptAlias / /usr/bin/keystone-wsgi-public

WSGIApplicationGroup %{GLOBAL}

WSGIPassAuthorization On

ErrorLogFormat "%{cu}t %M"

ErrorLog /var/log/httpd/keystone-error.log

CustomLog /var/log/httpd/keystone-access.log combined

<Directory /usr/bin>

Require all granted

</Directory>

</VirtualHost>

<VirtualHost \*:35357>

WSGIDaemonProcess keystone-admin processes=5 threads=1 user=keystone group=keystone display-name=%{GROUP}

WSGIProcessGroup keystone-admin

WSGIScriptAlias / /usr/bin/keystone-wsgi-admin

WSGIApplicationGroup %{GLOBAL}

WSGIPassAuthorization On

ErrorLogFormat "%{cu}t %M"

ErrorLog /var/log/httpd/keystone-error.log

CustomLog /var/log/httpd/keystone-access.log combined

<Directory /usr/bin>

Require all granted

</Directory>

</VirtualHost>

6） 启动服务

# systemctl enable httpd.service

# systemctl start httpd.service

2.3.4 创建service entity和API endpoint

1）配置临时环境变量

 验证令牌

# export OS\_TOKEN=ADMIN\_TOKEN(随机值)

 配置endpoint URL

# export OS\_URL=http://controller:35357/v3

 配置API版本

# export OS\_IDENTITY\_API\_VERSION=3

2）创建service entity和API endpoint

 创建service entity

# openstack service create \

--name keystone --description "OpenStack Identity" identity

 创建service entity API endpoint

# openstack endpoint create --region RegionOne \

identity public http://controller:5000/v3

#openstack endpoint create --region RegionOne \

identity internal http://controller:5000/v3

# openstack endpoint create --region RegionOne \

identity admin http://controller:35357/v3

3）创建domain（域）、project（项目）、users（用户）、roles（角色）

 创建domain

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

 创建admin project

#openstack project create --domain default \

--description "Admin Project" admin

 创建admin role

# openstack user create --domain default \

--password-prompt admin

 创建admin role

#openstack role create admin

 将admin角色添加到admin项目和用户

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

4）创建service project

#openstack project create --domain default \

--description "Service Project" service

5）验证

 移除临时环境变量

# unset OS\_TOKEN OS\_URL

 使用admin用户进行身份验证

#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

6）创建客户端环境验证脚本admin-openrc并添加如下内容

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

将ADMIN\_PASS替换为设置好的密码

2.4image service

2.4.1创建数据库

1）登陆数据库

#mysql –u root –p

2）创建数据库

Mariadb>CREATE DATABASE glance;

3）创建用户并授权

Mariadb>GRANT ALL PRIVILEGES ON glance.\* TO 'glance'@'localhost' \

DENTIFIED BY '11qqaazz';

Mariadb>GRANT ALL PRIVILEGES ON glance.\* TO 'glance'@'%' \

IDENTIFIED BY '11qqaazz';

2.4.2 创建身份验证

1）验证环境变量

# . admin-openrc

2）创建服务凭证前

 创建glance用户

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

 将admin角色添加到glance用户和service项目

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

# openstack service create --name glance \

--description "OpenStack Image" image

--description "OpenStack Image" image

3）创建镜像服务和API endpoint

# 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

2.4.3安装并配置glance

1）安装glance

# yum install openstack-glance

2）编辑/etc/glance/glance-api.conf

 在[database]部分添加以下内容，提供数据库连接凭证

connection = mysql+pymysql://glance:11qqaazz@controller/glance

 添加keystone访问参数

[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 = 11qqaazz

[paste\_deploy]

...

flavor = keystone

 添加glance访问途径

[glance\_store]

...

stores = file,http

default\_store = file

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

3）编辑/etc/glance/glance-registry.conf

 添加数据库访问

[database]

...

connection = mysql+pymysql://glance:11qqaazz@controller/glance

 添加keystone访问参数

[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

 添加glance访问途径

[glance\_store]

...

stores = file,http

default\_store = file

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

4）编辑/etc/glance/glance-registry.conf

 添加数据库访问参数

connection = mysql+pymysql://glance:11qqaazz@controller/glance

 在[keystone\_authtoken] 和[paste\_deploy]部分添加

[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

5）填充数据库

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

6）启动服务

# systemctl enable openstack-glance-api.service \

openstack-glance-registry.service

# systemctl start openstack-glance-api.service \

openstack-glance-registry.service

2.4.4验证

1）验证环境变量

# . admin-openrc

2）下载测试镜像

# wget http://download.cirros-cloud.net/0.3.4/cirros-0.3.4-x86\_64-disk.img

3）上传镜像

# openstack image create "cirros" \

--file cirros-0.3.4-x86\_64-disk.img \

--disk-format qcow2 --container-format bare \

--public

4）查看结果

# openstack image list

2.5 compute service

2.5.1创建数据库

1）登陆数据库

#mysql –u root –p

2）创建数据库

Mariadb>CREATE DATABASE nova;

Mariadb>CREATE DATABASE nova\_api;

3）创建用户并授权

Mariadb>GRANT ALL PRIVILEGES ON nova\_api.\* TO 'nova'@'localhost' \

DENTIFIED BY '11qqaazz';

Mariadb>GRANT ALL PRIVILEGES ON nova\_api.\* TO 'nova'@'%' \

IDENTIFIED BY '11qqaazz';

Mariadb>GRANT ALL PRIVILEGES ON nova.\* TO 'nova'@'localhost' \

DENTIFIED BY '11qqaazz';

Mariadb>GRANT ALL PRIVILEGES ON nova.\* TO 'nova'@'%' \

IDENTIFIED BY '11qqaazz';

2.5.2创建身份验证

1）验证环境变量

# . admin\_openrc

2）创建nova用户

#openstack user create –domain default –password-prompt nova

3）将admin角色添加到service项目和nova用户

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

4）创建nova 服务凭证

#openstack service create --name nova \

--description “Openstack Compute” conpute

5）创建compute服务API endpoint

# openstack endpoint create --region RegionOne \

compute public http://controller:8774/v2.1/%\(tenant\_id\)s

# openstack endpoint create --region RegionOne \

compute public http://controller:8774/v2.1/%\(tenant\_id\)s

# openstack endpoint create --region RegionOne \

compute admin http://controller:8774/v2.1/%\(tenant\_id\)s

2.5.3安装并配置controller节点

1）安装

#yum install openstack-nova-api openstack-nova-conductor \

openstack-nova-console openstack-nova-novncproxy \

openstack-nova-scheduler

2）编辑 /etc/nova/nova.conf

 支持元数据和API

[DEFAULT]

...

enabled\_apis = osapi\_compute,metadata

 添加api\_database

[api\_database]

...

connection=mysql+pymysql://nova:11qqaazz@controller/nova\_api

[database]

...

connection = mysql+pymysql://nova:11qqaazz@controller/nova

 添加rabbit认证参数

[DEFAULT]

...

rpc\_backend = rabbit

[oslo\_messaging\_rabbit]

...

rabbit\_host = controller

rabbit\_userid = openstack

rabbit\_password = RABBIT\_PAS

 添加keystone访问参数

[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 = 11qqaazz

 [DEFAULT]部分添加变量IP

[DEFAULT]

...

my\_ip = 127.0.0.1

 在[DEFAULT]部分添加

use\_neutron = True

firewall\_driver = nova.virt.firewall.NoopFirewallDriver

 在[vnc]部分添加

use\_neutron = True

firewall\_driver = nova.virt.firewall.NoopFirewallDriver

 在[glance]部分添加

api\_servers = http://controller:9292

 在[oslo\_concurrency]部分添加

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

3）填充数据库

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

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

4）启动服务

# systemctl enable openstack-nova-api.service \

openstack-nova-consoleauth.service openstack-nova-scheduler.service \

openstack-nova-conductor.service openstack-nova-novncproxy.service

# systemctl start openstack-nova-api.service \

openstack-nova-consoleauth.service openstack-nova-scheduler.service \

openstack-nova-conductor.service openstack-nova-novncproxy.service

2.5.4安装并配置compute节点

1）安装nova。

# yum install openstack-nova-compute

2）编辑/etc/nova/nova.conf

 添加rabbit访问凭证

[DEFAULT]

...

rpc\_backend = rabbit

[oslo\_messaging\_rabbit]

...

rabbit\_host = controller

rabbit\_userid = openstack

rabbit\_password = 11qqaazz

 添加keystone访问参数

[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 = 11qqaazz

 添加ip变量

[DEFAULT]

...

my\_ip = 127.0.0.1

 支持防火墙

[DEFAULT]

...

use\_neutron = True

firewall\_driver = nova.virt.firewall.NoopFirewallDriver

 开启vnc

[vnc]

...

enabled = True

vncserver\_listen = 0.0.0.0

vncserver\_proxyclient\_address = $my\_ip

novncproxy\_base\_url = http://controller:6080/vnc\_auto.html

 添加lock路径

[oslo\_concurrency]

...

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

2）查看服务器是否支持虚拟化

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

如果返回值为0则需要编辑/etc/nova/nova.conf的[libvirt]部分，在这部分添加

[libvirt]

...

virt\_type = qemu

3）启动服务

# systemctl enable libvirtd.service openstack-nova-compute.service

# systemctl start libvirtd.service openstack-nova-compute.service

2.5.5 验证

1）验证环境变量

# . admin-openrc

2）检查是否出现compute

# openstack compute service list

2.6 neutron配置

2.6.1 在controller节点创建数据库

1）登陆数据库

#mysql –u root –p

2）创建数据库

mariadb>CREATE DATABASE neutron;

3）创建用户并授权

mariadb>GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'localhost' \

IDENTIFIED BY 'NEUTRON\_DBPASS';

mariadb>GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'%' \

IDENTIFIED BY 'NEUTRON\_DBPASS';

2.6.2 创建身份验证

1）刷新环境变量

#. admin-openrc

2）创建neutron用户

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

3）添加admin角色到neutron

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

4）创建neutron服务验证

# openstack service create --name neutron \

--description "OpenStack Networking" network

5）创建网络服务endpoint

#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

2.6.3配置neutron

按照官方文档neutron使用的是linuxbridge方式配置网络，本安装文档选择的是官网支持第三层网络的网络选项2，其他详情请查阅官方文档http://docs.openstack.org/mitaka/install-guide-rdo/neutron-controller-install-option2.html

1） 安装neutron

# yum install openstack-neutron openstack-neutron-ml2 \

openstack-neutron-linuxbridge ebtables

2） 配置/etc/neutron/neutron.conf文件

 在[database]部分添加数据库连接凭证

connection=mysql+pymysql://neutron:NEUTRON\_DBPASS@controller/neutron

 在[DEFAULT]部分添加支持ml2插件

[DEFAULT]

...

core\_plugin = ml2

service\_plugins = router

allow\_overlapping\_ips = True

 添加rabbit验证

[DEFAULT]

...

rpc\_backend = rabbit

[oslo\_messaging\_rabbit]

...

rabbit\_host = controller

rabbit\_userid = openstack

rabbit\_password = RABBIT\_PASS

 配置元数据主机和共享密钥

[DEFAULT]

...

nova\_metadata\_ip = controller

metadata\_proxy\_shared\_secret = METADATA\_SECRET

可以将METADATA\_SECRET替换为自定义，只要nova的配置里元数据密钥对应即可

 添加keystone认证

[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

 配置nova访问参数

[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

 配置lock路径

[oslo\_concurrency]

...

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

3） 配置ml2插件，编辑/etc/neutron/plugins/ml2/ml2\_conf.ini

 支持网络

[ml2]

...

type\_drivers = flat,vlan,vxlan

 配置租户网络

[ml2]

...

tenant\_network\_types = vxlan

 使用linuxbridge，l2population驱动

[ml2]

...

mechanism\_drivers = linuxbridge,l2population

 启用端口安全扩展

[ml2]

...

extension\_drivers = port\_security

 配置虚拟网络提供flat

[ml2\_type\_flat]

...

flat\_networks = provider

 配置vxlan标识比例

[ml2\_type\_vxlan]

...

vni\_ranges = 1:1000

 提高ipset效率配置

[securitygroup]

...

enable\_ipset = True

4）配置linuxbridge。/etc/neutron/plugins/ml2/linuxbridge\_agent.ini

 配置物理网络接口

[linux\_bridge]

physical\_interface\_mappings = provider:PROVIDER\_INTERFACE\_NAME

将PROVIDER\_INTERFACE\_NAME修改为物理网络接口

 配置vxlan第二层网络的接入物理地址

[vxlan]

enable\_vxlan = True

local\_ip = OVERLAY\_INTERFACE\_IP\_ADDRESS

l2\_population = True

 配置防火墙驱动

[securitygroup]

...

enable\_security\_group = True

firewall\_driver = neutron.agent.linux.iptables\_firewall.IptablesFirewallDriver

5）配置第三层网络代理

 配置Linux网桥接口驱动程序和外部网络桥:

[DEFAULT]

...

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

external\_network\_bridge =

7） 配置dhcp代理

 配置linux网桥接口驱动

[DEFAULT]

...

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

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

enable\_isolated\_metadata = True

8） 编辑/etc/nova/nova.conf

 配置neutron访问参数和共享密钥

[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

9） 创建ml2插件软链接

#ln -s /etc/neutron/plugins/ml2/ml2\_conf.ini /etc/neutron/plugin.ini

10） 填充数据库

# 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

11）重启计算服务

# systemctl restart openstack-nova-api.service

12）启动网络服务并添加开机自启动

# systemctl enable neutron-server.service \

neutron-linuxbridge-agent.service neutron-dhcp-agent.service \

neutron-metadata-agent.service

# systemctl start neutron-server.service \

neutron-linuxbridge-agent.service neutron-dhcp-agent.service \

neutron-metadata-agent.service

13）启动第三层网络服务

# systemctl enable neutron-l3-agent.service

# systemctl start neutron-l3-agent.service

2.6.4配置compute节点neutron

1）安装neutron

# yum install openstack-neutron-linuxbridge ebtables ipset

2）编辑/etc/neutron/neutron.conf

 配置rabbit认证

[DEFAULT]

...

rpc\_backend = rabbit

[oslo\_messaging\_rabbit]

...

rabbit\_host = controller

rabbit\_userid = openstack

rabbit\_password = RABBIT\_PASS

 配置keystone访问参数

[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

 配置lock路径

[oslo\_concurrency]

...

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

3）编辑/etc/neutron/plugins/ml2/linuxbridge\_agent.ini，配置linuxbridge

 配置物理网络接口

[linux\_bridge]

physical\_interface\_mappings = provider:PROVIDER\_INTERFACE\_NAME

将PROVIDER\_INTERFACE\_NAME替换为物理网卡

 配置vxlan接入第二层网络ip

[vxlan]

enable\_vxlan = True

local\_ip = OVERLAY\_INTERFACE\_IP\_ADDRESS

l2\_population = True

 配置防火墙驱动

[securitygroup]

...

enable\_security\_group = True

firewall\_driver = neutron.agent.linux.iptables\_firewall.IptablesFirewallDriver

4） 配置/etc/nova/nova.conf

 配置neutron访问参数

[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

5） 重启计算服务

# systemctl restart openstack-nova-compute.service

6） 启动linuxbridge服务

# systemctl enable neutron-linuxbridge-agent.service

# systemctl start neutron-linuxbridge-agent.service

2.7dashboard

1）安装dashboard

# yum install openstack-dashboard

2）编辑/etc/openstack-dashboard/local\_settings

 配置OPENSTACK\_HOST,将dashboard设置在controller节点上

OPENSTACK\_HOST = "controller"

 允许所有主机访问

ALLOWED\_HOSTS = ['\*', ]

 配置memcached会话存储服务:

SESSION\_ENGINE = 'django.contrib.sessions.backends.cache'

CACHES = {

'default': {

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

'LOCATION': 'controller:11211',

}

}

 配置身份验证版本为3

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

 让dashboard支持domain

OPENSTACK\_KEYSTONE\_MULTIDOMAIN\_SUPPORT = True

 配置API版本

OPENSTACK\_API\_VERSIONS = {

"identity": 3,

"image": 2,

"volume": 2,

}

 设置default为默认域

OPENSTACK\_KEYSTONE\_DEFAULT\_DOMAIN = "default"

 设置默认角色

OPENSTACK\_KEYSTONE\_DEFAULT\_ROLE = "user"

 如果你选择网络选项1,禁用支持第三层网络服务

OPENSTACK\_NEUTRON\_NETWORK = {

...

'enable\_router': False,

'enable\_quotas': False,

'enable\_distributed\_router': False,

'enable\_ha\_router': False,

'enable\_lb': False,

'enable\_firewall': False,

'enable\_vpn': False,

'enable\_fip\_topology\_check': False,

}

 设置时间域

TIME\_ZONE = "Asia/Shanghai"

3）重启服务

# systemctl restart httpd.service memcached.service