CentOS7 + OpenStack + KVM + vlan文档

[CentOS7 + OpenStack + KVM + vlan文档 1](#_Toc437344570)

[1. Openstack概念图 3](#_Toc437344571)

[2. 本次试验架构图 4](#_Toc437344572)

[2.1 本次采用三个节点 4](#_Toc437344573)

[3. keystone认证服务 7](#_Toc437344574)

[3.1 定义认证用户，租户，角色 9](#_Toc437344575)

[4. glance镜像服务 10](#_Toc437344576)

[4.1 创建数据库并授权 11](#_Toc437344577)

[4.2 安装 11](#_Toc437344578)

[4.3 配置数据库 11](#_Toc437344579)

[4.4 生成数据库表文件 13](#_Toc437344580)

[4.8 验证glance服务 13](#_Toc437344581)

[5 Nova计算服务 14](#_Toc437344582)

[5.1 Nova控制节点 14](#_Toc437344583)

[5.2 Nova计算节点【安装配置计算节点】 16](#_Toc437344584)

[6. Network网络服务 18](#_Toc437344585)

[6.1 控制节点 18](#_Toc437344586)

[6.2 配置计算节点 20](#_Toc437344587)

[7. Dashboard Web服务 31](#_Toc437344588)

[7.1 软件包安装 31](#_Toc437344589)

[8. Cinder服务 32](#_Toc437344590)

[8.1 Add the Block Storage service 32](#_Toc437344591)

[8.2 创建数据库 32](#_Toc437344592)

[8.3 创建对应账户 32](#_Toc437344593)

[8.4 安装相应软件或者插件 32](#_Toc437344594)

[8.5 修改配置文件 33](#_Toc437344595)

[8.6 存储节点 34](#_Toc437344596)

[9. Heat服务 35](#_Toc437344597)

[9.1 软件包安装 35](#_Toc437344598)

[9.2 配置数据库文件 35](#_Toc437344599)

[9.3 创建数据库并赋予相应权限 36](#_Toc437344600)

[9.4 生成数据库表 36](#_Toc437344601)

[9.5 配置Heat配置文件 36](#_Toc437344602)

[9.6 注册heat服务 36](#_Toc437344603)

[9.7 创建heat账户 37](#_Toc437344604)

[9.8 启动heat服务并设置开机启动 37](#_Toc437344605)

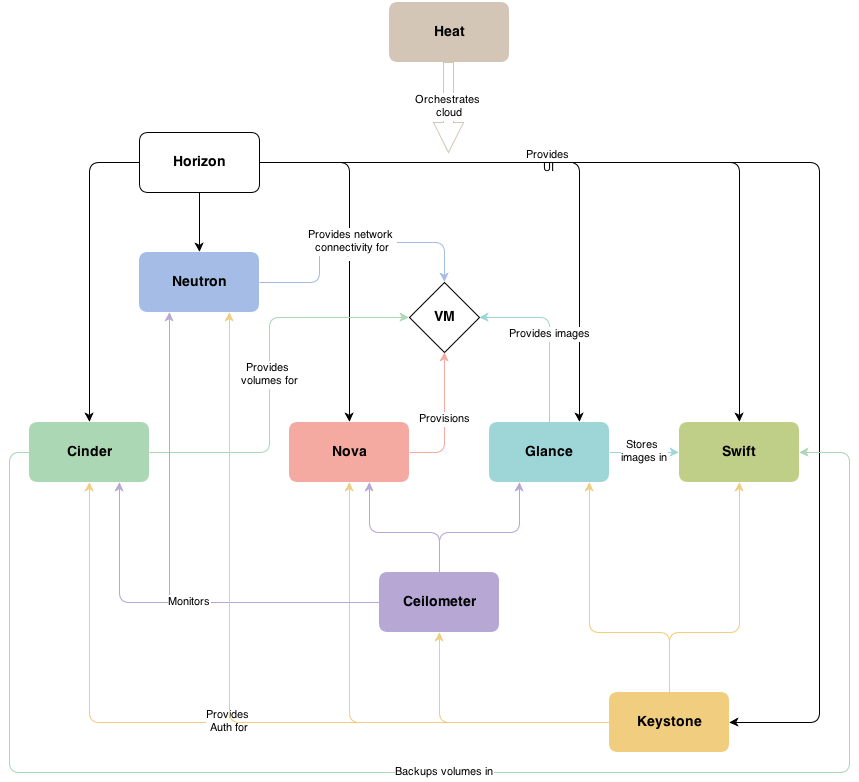
[9.9 验证 37](#_Toc437344606)

[10. 热迁移 38](#_Toc437344607)

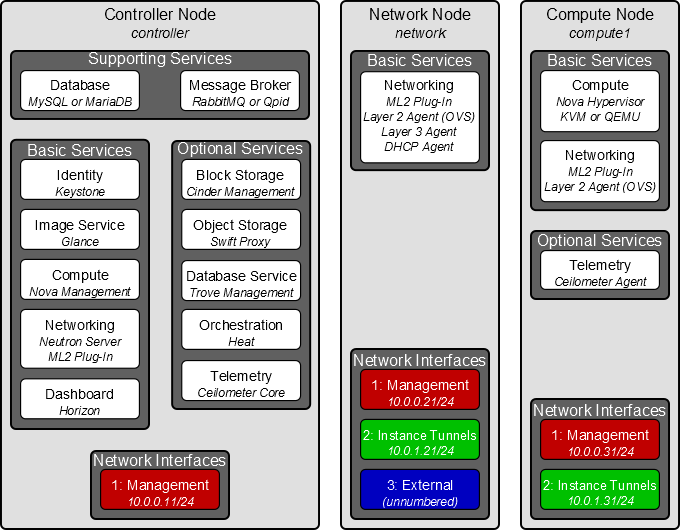
[10.1. Openstack热迁移计算节点 39](#_Toc437344608)

[Q&A 43](#_Toc437344609)

## Openstack概念图



## 本次试验架构图



### 2.1 本次采用三个节点

控制节点（**虚机**）

管理网(Management)：192.168.5.228 ---- eth0

外部网(External)：111.204.241.184 ---- eth1

网络节点（**虚机**）

管理网(Management)：192.168.5.76 ---- eth0

外部网(External)：111.204.241.209 ---- eth1

通道网(Tunnels)：111.204.241.209 ---- eth1（可以使用另外网段）

计算节点（**实机**）

管理网(Management)：192.168.5.220 ---- eth0

通道网(Tunnels)：111.204.241.187 ---- eth1（可以使用另外网段）

外部网(External)：111.204.241.187 ---- eth1

#####**PS：管理网、外部网、通道网必须在同一网段，且在对应网卡**

对应系统最低需求：

* Controller Node: 1 processor, 2 GB memory, and 5 GB storage
* Network Node: 1 processor, 512 MB memory, and 5 GB storage
* Compute Node: 1 processor, 2 GB memory, and 10 GB storage

操作系统版本：CentOS6.5 x64

配置前的准备工作：

* # service NetworkManager stop
* # service network start
* # chkconfig NetworkManager off
* # chkconfig network on
* # service firewalld stop
* # service iptables start
* # chkconfig firewalld off
* # chkconfig iptables on

网络环境准备：

在对应的网卡上配置相应的IP地址

【PS：为了具有更好的扩展性，修改hosts文件。将各个节点的主机名和IP添加进去。在三个节点都需要操作同样的事情。便于以后直接使用域名】

# controller

192.168.5.228 controller

# network

192.168.5.76 network

# compute1

192.168.5.220 compute1

在三个节点都对应配置好之后，请相互之间ping测试一下。【必须操作】

**安装时间同步ntp 服务**。

#yum install ntp

然后到计划任务中添加以下几行

# Puppet Name: time sync

\*/5 1-23 \* \* \* (/usr/sbin/ntpdate time.soufun.com)

# Puppet Name: time sync 40

40 0 \* \* \* (/usr/sbin/ntpdate time.soufun.com)

# Puppet Name: time sync 30

30 0 \* \* \* (/usr/sbin/ntpdate time.soufun.com)

# Puppet Name: time sync 50

50 0 \* \* \* (/usr/sbin/ntpdate time.soufun.com)

数据库—MySQL

**控制节点：**

安装mysql数据库

**# yum install mysql mysql-server MySQL-python 【现在这里转由数据库处理】**

然后再/etc/my.cnf配置文件中，增加一下几个参数

**bind-address = 0.0.0.0**  ##推荐修改为控制节点的IP，192.168.5.228

**default-storage-engine = innodb**

**innodb\_file\_per\_table**

**collation-server = utf8\_general\_ci**

**init-connect = 'SET NAMES utf8'**

**character-set-server = utf8**

然后重启服务，并删除anonymous账户

# mysql\_install\_db ###生成数据库表

# mysql\_secure\_installation ###安全操作，基本上可以直接Yes到底

**其它节点**

# yum install MySQL-python

Openstack软件包准备并更新系统环境和kernel内核

【PS：我还是推荐把163源添加进去，到163源官网可以直接下载使用，或者从现有机器上拷贝直接可以使用】

# yum install yum-plugin-priorities

**yum install http://dl.fedoraproject.org/pub/epel/7/x86\_64/e/epelrelease-**

**7-5.noarch.rpm**

# **yum install** **http://rdo.fedorapeople.org/openstack-juno/rdo-release-juno.**

**Rpm**

# yum upgrade

# reboot

# yum install openstack-utils

# yum install openstack-selinux

消息队列服务，在控制节点上安装，并作为消息队列服务的服务器

**yum install rabbitmq-server**

启动服务，并设置为开机启动

**# systemctl enable rabbitmq-server.service**

# **systemctl start rabbitmq-server.service**

配置rabbit账户和密码【如果使用了非guest账户，需要在每一个使用rabbit服务的openstack节点上都做相应的修改配置】

**rabbitmqctl change\_password guest *RABBIT\_PASS***

# The RabbitMQ userid. (string value)

#rabbit\_userid=guest

#rabbit\_password=guest

检测rabbitMQ服务状态

# **rabbitmqctl status | grep rabbit**

/etc/rabbitmq/rabbitmq.config

[{rabbit, [{loopback\_users, []}]}].

**systemctl restart rabbitmq-server.service**

## keystone认证服务

认证服务是联系所有服务。任何操作都必须经过认证才能够操作。

1. 安装认证服务

# yum install openstack-keystone python-keystoneclient

认证服务中的认证信息都是存储在数据库中，所以必须配置数据库配置文件，如下：

# openstack-config --set /etc/keystone/keystone.conf database connection mysql://keystone:*KEYSTONE\_DBPASS*@*controller*/keystone

【KEYSTONE\_DBPASS为你自己定义的数据库连接密码

Controller为你的控制节点，可以直接使用IP地址】

然后登陆数据库，创建数据库，并授予相应权限。

$ mysql -u root -p

mysql> CREATE DATABASE keystone;

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*';

mysql> exit

根据配置文件，生成数据库表

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

【操作完这一步之后，一定记得去数据库中看一下，是否有表生成，如果没有，接下来的操作会失败。关于这一步失败的原因在最后的QA中有分析】

生成对应的认证文件，并写入配置文件中

随机生成密码命令：

**openssl rand -base64 12**

**openssl rand -hex 12**

**安装相应的组件**

# **yum install openstack-keystone python-keystoneclient**

**编辑**/etc/keystone/keystone.conf

[DEFAULT]

...

admin\_token = ADMIN\_TOKEN ###修改为上面随机生成的密码

连接数据库

[database]

...

connection = mysql://keystone:*KEYSTONE\_DBPASS*@*controller*/keystone

配置mysql驱动

[token]

...

provider = keystone.token.providers.uuid.Provider

driver = keystone.token.persistence.backends.sql.Token

配置sql revoke驱动

[revoke]

...

driver = keystone.contrib.revoke.backends.sql.Revoke

[DEFAULT]

...

verbose = True

对认证文件进行授权操作

# **keystone-manage pki\_setup --keystone-user keystone --keystone-group**

**keystone**

# **chown -R keystone:keystone /var/log/keystone**

# **chown -R keystone:keystone /etc/keystone/ssl**

# chown -R keystone:keystone /etc/keystone/ssl

# chmod -R o-rwx /etc/keystone/ssl

生成数据文件

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

设置开机启动

# **systemctl enable openstack-keystone.service**

# **systemctl start openstack-keystone.service**

这一步，非必须操作，但是推荐。【这一步作用是用于清空日志，因为日志生成太快了，当然可以使用 cat /dev/null>/var/log/keystone/keystone-tokenflush.log】

# (crontab -l -u keystone 2>&1 | grep -q token\_flush) || \

echo '@hourly /usr/bin/keystone-manage token\_flush >/var/log/keystone/keystone-tokenflush.log 2>&1' >> /var/spool/cron/keystone

\*/30 \* \* \* \* /usr/bin/keystone-manage token\_flush >/var/log/keystone/keystone-tokenflush.log 2>&1

### 3.1 定义认证用户，租户，角色

$ export OS\_SERVICE\_TOKEN=*ADMIN\_TOKEN* ##ADMIN\_TOKEN就是上面随机生成的密码

$ export OS\_SERVICE\_ENDPOINT=http://controller:35357/v2.0

创建admin租户

$ **keystone tenant-create --name admin --description "Admin Tenant"**

创建admin用户

$ **keystone user-create --name admin --pass *ADMIN\_PASS* --**

**email *EMAIL\_ADDRESS***

*创建admin角色*

$ **keystone role-create --name admin**

将admin角色加入admin租户中

$ **keystone user-role-add --user admin --tenant admin --role admin**

Link the admin user, \_member\_ role, and admin tenant:

**keystone tenant-create --name demo --description "Demo Tenant"**

$ **keystone user-create --name demo --tenant demo --pass *DEMO\_PASS* --**

**email *EMAIL\_ADDRESS***

$ **keystone tenant-create --name service --description "Service Tenant"**

$ **keystone service-create --name keystone --type identity \**

**--description "OpenStack Identity"**

创建一个对应的认证服务

$ keystone service-create --name=keystone --type=identity

--description="OpenStack Identity"

创建对应API接口

$ keystone endpoint-create \

--service-id=$(keystone service-list | awk '/ identity / {print $2}') \

--publicurl=http://*controller*:5000/v2.0 \

--internalurl=http://*controller*:5000/v2.0 \

--adminurl=http://*controller*:35357/v2.0 \

**--region regionOne**

验证操作

第一步：取消认证

**unset OS\_SERVICE\_TOKEN OS\_SERVICE\_ENDPOINT**

$ **keystone --os-tenant-name admin --os-username admin –os-password *ADMIN\_PASS* \**

**--os-auth-url http://controller:35357/v2.0 token-get**

$ **keystone --os-tenant-name admin --os-username admin –os-password *ADMIN\_PASS* \**

**--os-auth-url http://controller:35357/v2.0 tenant-list**

$ **keystone --os-tenant-name admin --os-username admin –os-password *ADMIN\_PASS* \**

**--os-auth-url http://controller:35357/v2.0 user-list**

$ **keystone --os-tenant-name admin --os-username admin –os-password *ADMIN\_PASS* \**

**--os-auth-url http://controller:35357/v2.0 role-list**

$ **keystone --os-tenant-name demo --os-username demo –os-password *DEMO\_PASS* \**

**--os-auth-url http://controller:35357/v2.0 token-get**

**keystone --os-tenant-name demo --os-username demo –os-password *DEMO\_PASS* \**

**--os-auth-url http://controller:35357/v2.0 user-list**

为了便于以后操作，切换，请创建admin-openrc.sh，作用是切换到admin环境。使用操作：source admin-openrc.sh

export OS\_USERNAME=admin

export OS\_PASSWORD=ADMIN\_PASS

export OS\_TENANT\_NAME=admin

export OS\_AUTH\_URL=http://controller:35357/v2.0

创建demo环境，创建demo-openrc.sh,作用是切换到demo环境。使用操作：source demo-openrc.sh

**export OS\_TENANT\_NAME=demo**

**export OS\_USERNAME=demo**

**export OS\_PASSWORD=*DEMO\_PASS***

**export OS\_AUTH\_URL=http://*controller*:5000/v2.0**

查看现有用户命令：

Keystone user-list

keystone user-role-list --user admin --tenant admin

## glance镜像服务

* glance-api. Accepts Image API calls for image discovery, retrieval, and storage.
* glance-registry. Stores, processes, and retrieves metadata about images. Metadata includes items such as size and type.
* Database. Stores image metadata. You can choose your database depending on your preference. Most deployments use MySQL or SQlite.
* Storage repository for image files. The Image Service supports a variety of repositories including normal file systems, Object Storage, RADOS block devices, HTTP, and Amazon S3. Some types of repositories support only read-only usage.

### 4.1 创建数据库并授权

***$ mysql -u root -p***

***mysql> CREATE DATABASE glance;***

***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';***

创建keystone用户

$ keystone user-create --name=glance --pass=*GLANCE\_PASS* \

//可以不写 --email=*glance@example.com*

$ keystone user-role-add --user=glance --tenant=service --role=admin

$ **keystone service-create --name glance --type image \**

**--description "OpenStack Image Service"**

$ **keystone endpoint-create \**

**--service-id $(keystone service-list | awk '/ image / {print $2}') \**

**--publicurl http://*controller*:9292 \**

**--internalurl http://*controller*:9292 \**

**--adminurl http://*controller*:9292 \**

**--region regionOne**

### 4.2 安装

# yum install openstack-glance python-glanceclient

### 配置数据库

Edit the /etc/glance/glance-api.conf

**[database]**

**...**

**connection = mysql://glance:*GLANCE\_DBPASS*@*controller*/glance**

**[keystone\_authtoken]**

**...**

**auth\_uri = http://*controller*:5000/v2.0**

**identity\_uri = http://*controller*:35357**

**admin\_tenant\_name = service**

**admin\_user = glance**

**admin\_password = *GLANCE\_PASS***

**[paste\_deploy]**

**...**

**flavor = keystone**

Comment out any auth\_host, auth\_port, and auth\_protocol

options because the identity\_uri option replaces them.

**[glance\_store]**

**...**

**default\_store = file**

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

**[DEFAULT]**

**...**

**notification\_driver = noop**

**[DEFAULT]**

**...**

**verbose = True**

Edit the /etc/glance/glance-registry.conf

**[database]**

**...**

**connection = mysql://glance:*GLANCE\_DBPASS*@*controller*/glance**

**[keystone\_authtoken]**

**...**

**auth\_uri = http://*controller*:5000/v2.0**

**identity\_uri = http://*controller*:35357**

**admin\_tenant\_name = service**

**admin\_user = glance**

**admin\_password = *GLANCE\_PASS***

**[paste\_deploy]**

**...**

**flavor = keystone**

**[DEFAULT]**

**...**

**notification\_driver = noop**

**[DEFAULT]**

**...**

**verbose = True**

### 4.4 生成数据库表文件

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

# **systemctl enable openstack-glance-api.service openstack-glance-registry.service**

# **systemctl start openstack-glance-api.service openstack-glance-registry.service**

### 4.8 验证glance服务

#### 4.8.1 下载glance镜像

$ mkdir /tmp/images

$ cd /tmp/images/

$ wget http://cdn.download.cirros-cloud.net/0.3.2/cirros-0.3.2-x86\_64-disk.img

$ **wget -P /tmp/images http://download.cirros-cloud.net/0.3.3/cirros-0.3.3-x86\_64-disk.img**

$ **source admin-openrc.sh**

$ **glance image-create --name "cirros-0.3.3-x86\_64" --file /tmp/images/cirros-0.3.3-x86\_64-disk.img \**

**--disk-format qcow2 --container-format bare --is-public True --progress**

$ **glance image-list**

#### 4.8.2 上传镜像到glance服务中存储

$ mkdir /tmp/images

$ cd /tmp/images/

$ wget http://cdn.download.cirros-cloud.net/0.3.2/cirros-0.3.2-x86\_64-disk.img

可以通过以下命令查看文件信息

$ file cirros-0.3.2-x86\_64-disk.img

cirros-0.3.2-x86\_64-disk.img: **QEMU** QCOW Image (v2), 41126400 bytes

#### 4.8.3 通过命令，将img制作成glance镜像

$ source admin-openrc.sh

$ glance image-create --name "cirros-0.3.2-x86\_64" --disk-format qcow2 --container-format bare --is-public True --progress < cirros-0.3.2-x86\_64-disk.img

【扩展：如果镜像是在网上，可以通过url地址使用，可用户跨机房使用的情况】

$ glance image-create --name="cirros-0.3.2-x86\_64" --disk-format=qcow2 \

--container-format=bare --is-public=true \

--copy-from http://cdn.download.cirros-cloud.net/0.3.2/cirros-0.3.2-x86\_64-disk.img

#### 4.8.4 验证glance镜像

$ glance image-list

## Nova计算服务

### 5.1 Nova控制节点

创建Nova数据库

$ **mysql -u root –p**

**CREATE DATABASE nova;**

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

**IDENTIFIED BY '*NOVA\_DBPASS*';**

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

**IDENTIFIED BY '*NOVA\_DBPASS*';**

**切换到admin环境**

$ **source admin-openrc.sh**

$ **keystone user-create --name nova --pass *NOVA\_PASS***

$ **keystone user-role-add --user nova --tenant service --role admin**

$ **keystone service-create --name nova --type compute \**

**--description "OpenStack Compute"**

$ **keystone endpoint-create \**

**--service-id $(keystone service-list | awk '/ compute / {print $2}') \**

**--publicurl http://*controller*:8774/v2/%\(tenant\_id\)s \**

**--internalurl http://*controller*:8774/v2/%\(tenant\_id\)s \**

**--adminurl http://*controller*:8774/v2/%\(tenant\_id\)s \**

**--region regionOne**

#### 5.1.1 安装openstack nova软件包

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

openstack-nova-console openstack-nova-novncproxy openstack-nova-scheduler \

python-novaclient

#### 5.1.2 配置数据库配置文件

/etc/nova/nova.conf

**[database]**

**...**

**connection = mysql://nova:*NOVA\_DBPASS*@controller/nova**

**[DEFAULT]**

**...**

**rpc\_backend = rabbit**

**rabbit\_host = *controller***

**rabbit\_password = *RABBIT\_PASS***

**[DEFAULT]**

**...**

**auth\_strategy = keystone**

**[keystone\_authtoken]**

**...**

**auth\_uri = http://*controller*:5000/v2.0**

**identity\_uri = http://*controller*:35357**

**admin\_tenant\_name = service**

**admin\_user = nova**

**admin\_password = *NOVA\_PASS***

**[DEFAULT]**

**...**

**my\_ip = 10.0.0.11**

**[DEFAULT]**

**...**

**vncserver\_listen = 10.0.0.11**

**vncserver\_proxyclient\_address = 10.0.0.11**

**[glance]**

**...**

**host = *controller***

**[DEFAULT]**

**...**

**verbose = True**

**生成数据库文件**

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

# **systemctl enable openstack-nova-api.service openstack-nova-cert.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-cert.service \**

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

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

#### 5.1.11 验证nova服务

$ nova image-list

### 5.2 Nova计算节点【安装配置计算节点】

#### 5.2.1 安装nova软件

# **yum install openstack-nova-compute sysfsutils**

【如果多安装了其他服务，只要不启动服务，不会影响结果】

Edit the /etc/nova/nova.conf

**[DEFAULT]**

**...**

**rpc\_backend = rabbit**

**rabbit\_host = *controller***

**rabbit\_password = *RABBIT\_PASS***

**[DEFAULT]**

**...**

**auth\_strategy = keystone**

**[keystone\_authtoken]**

**...**

**auth\_uri = http://*controller*:5000/v2.0**

**identity\_uri = http://*controller*:35357**

**admin\_tenant\_name = service**

**admin\_user = nova**

**admin\_password = *NOVA\_PASS***

**[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.

**[DEFAULT]**

**...**

**vnc\_enabled = True**

**vncserver\_listen = 0.0.0.0**

**vncserver\_proxyclient\_address = *MANAGEMENT\_INTERFACE\_IP\_ADDRESS***

**novncproxy\_base\_url =** [**http://*controller*:6080/vnc\_auto.html**](http://controller:6080/vnc_auto.html)

**[glance]**

**...**

**host = *controller***

**[DEFAULT]**

**...**

**verbose = True**

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

**启动相应服务**

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

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

$ **source admin-openrc.sh**

$ **nova service-list**

$ **nova image-list**

##### 5.2.5.1 查看计算节点是否支持虚拟化技术

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

如果返回为0，表示不支持，修虚拟化技术修改为qemu,如果非0，表示支持虚拟化技术。则修改为kvm，当然，如果使用的是其他虚拟化机器，请对应修改。

# openstack-config --set /etc/nova/nova.conf libvirt virt\_type qemu

## 6. Network网络服务

### 6.1 控制节点

#### 6.1.2 生成数据库

$ **mysql -u root –p**

**CREATE DATABASE neutron;**

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

**IDENTIFIED BY '*NEUTRON\_DBPASS*';**

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

**IDENTIFIED BY '*NEUTRON\_DBPASS*';**

#### 6.1.3创建对应账户

$ **source admin-openrc.sh**

$ **keystone user-create --name neutron --pass *NEUTRON\_PASS***

$ **keystone user-role-add --user neutron --tenant service --role admin**

$ **keystone service-create --name neutron --type network \**

**--description "OpenStack Networking"**

$ **keystone endpoint-create \**

**--service-id $(keystone service-list | awk '/ network / {print $2}') \**

**--publicurl http://*controller*:9696 \**

**--adminurl http://*controller*:9696 \**

**--internalurl http://*controller*:9696 \**

**--region regionOne**

#### 6.1.4 安装软件

# **yum install openstack-neutron openstack-neutron-ml2 python-neutronclient which**

#### 6.1.5 修改配置

Edit the /etc/neutron/neutron.conf

[database]

...

connection = mysql://neutron:*NEUTRON\_DBPASS*@*controller*/neutron

[DEFAULT]

...

rpc\_backend = rabbit

rabbit\_host = *controller*

rabbit\_password = *RABBIT\_PASS*

[DEFAULT]

...

auth\_strategy = keystone

[keystone\_authtoken]

...

auth\_uri = http://*controller*:5000/v2.0

identity\_uri = http://*controller*:35357

admin\_tenant\_name = service

admin\_user = neutron

admin\_password = *NEUTRON\_PASS*

[DEFAULT]

...

core\_plugin = ml2

service\_plugins = router

allow\_overlapping\_ips = True

[DEFAULT]

...

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

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

nova\_url = http://*controller*:8774/v2

nova\_admin\_auth\_url = http://*controller*:35357/v2.0

nova\_region\_name = regionOne

nova\_admin\_username = nova

nova\_admin\_tenant\_id = *SERVICE\_TENANT\_ID*

nova\_admin\_password = *NOVA\_PASS*

**通过以下命令获取service\_tenant\_id**

$ **source admin-openrc.sh**

$ **keystone tenant-get service**

[DEFAULT]

...

verbose = True

#### 6.1.6 配置ML2

**To configure the Modular Layer 2 (ML2) plug-in**

Edit the /etc/neutron/plugins/ml2/ml2\_conf.ini

[ml2]

...

type\_drivers = flat,vlan

tenant\_network\_types = vlan

mechanism\_drivers = openvswitch

[ml2\_type\_vlan]

network\_vlan\_ranges = physnet1:900:910

[securitygroup]

...

enable\_security\_group = True

enable\_ipset = True

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

### 6.2 配置计算节点

**To configure Compute to use Networking**

Edit the /etc/nova/nova.conf file on the controller node and complete the following actions:

[DEFAULT]

...

network\_api\_class = nova.network.neutronv2.api.API

security\_group\_api = neutron

linuxnet\_interface\_driver = nova.network.linux\_net.LinuxOVSInterfaceDriver

firewall\_driver = nova.virt.firewall.NoopFirewallDriver

[neutron]

...

url = http://*controller*:9696

auth\_strategy = keystone

admin\_auth\_url = http://*controller*:35357/v2.0

admin\_tenant\_name = service

admin\_username = neutron

admin\_password = *NEUTRON\_PASS*

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

# **su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf \**

**--config-file /etc/neutron/plugins/ml2/ml2\_conf.ini upgrade juno" neutron**

# **systemctl restart openstack-nova-api.service openstack-nova-scheduler.service \**

**openstack-nova-conductor.service**

# **systemctl enable neutron-server.service**

# **systemctl start neutron-server.service**

#### 6.2.1 验证

$ **source admin-openrc.sh**

$ **neutron ext-list**

#### 6.2.2 安装和配置网络节点

**首先配置内核**

Edit the /etc/sysctl.conf

net.ipv4.ip\_forward=1

net.ipv4.conf.all.rp\_filter=0

net.ipv4.conf.default.rp\_filter=0

# **sysctl –p**

# **yum install openstack-neutron openstack-neutron-ml2 openstack-neutronopenvswitch**

Edit the /etc/neutron/neutron.conf

In the [database] section, comment out any connection options because network

nodes do not directly access the database.

[DEFAULT]

...

rpc\_backend = rabbit

rabbit\_host = *controller*

rabbit\_password = *RABBIT\_PASS*

[DEFAULT]

...

auth\_strategy = keystone

[keystone\_authtoken]

...

auth\_uri = http://*controller*:5000/v2.0

identity\_uri = http://*controller*:35357

admin\_tenant\_name = service

admin\_user = neutron

admin\_password = *NEUTRON\_PASS*

[DEFAULT]

...

core\_plugin = ml2

service\_plugins = router

allow\_overlapping\_ips = True

[DEFAULT]

...

verbose = True

Edit the /etc/neutron/plugins/ml2/ml2\_conf.ini

[ml2]

...

type\_drivers = flat,vlan

tenant\_network\_types = vlan

mechanism\_drivers = openvswitch

[ml2\_type\_flat]

...

flat\_networks = external

[securitygroup]

...

enable\_security\_group = True

enable\_ipset = True

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

In the [ovs] section, enable tunnels, configure the local tunnel endpoint, and

map the external flat provider network to the br-ex external network bridge:

[ovs]

...

local\_ip = *INSTANCE\_TUNNELS\_INTERFACE\_IP\_ADDRESS*

local\_ip = 106.39.78.53

tenant\_network\_type = vlan

integration\_bridge = br-int

network\_vlan\_ranges = physnet1:900:910

bridge\_mappings = physnet1:br-ex

**To configure the Layer-3 (L3) agent**

Edit the /etc/neutron/l3\_agent.ini

[DEFAULT]

...

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

use\_namespaces = True

external\_network\_bridge = br-ex

router\_delete\_namespaces = True

[DEFAULT]

...

verbose = True

**To configure the DHCP agent**

Edit the /etc/neutron/dhcp\_agent.ini

[DEFAULT]

...

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

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

use\_namespaces = True

dhcp\_delete\_namespaces = True

[DEFAULT]

...

verbose = True

Edit the /etc/neutron/dhcp\_agent.ini

[DEFAULT]

...

dnsmasq\_config\_file = /etc/neutron/dnsmasq-neutron.conf

Create and edit the /etc/neutron/dnsmasq-neutron.conf

dhcp-option-force=26,1454 ### Enable the DHCP MTU option (26) and configure it to 1454 bytes

Kill any existing dnsmasq processes:

# **pkill dnsmasq**

**To configure the metadata agent**

Edit the /etc/neutron/metadata\_agent.ini

[DEFAULT]

...

auth\_url = http://*controller*:5000/v2.0

auth\_region = regionOne

admin\_tenant\_name = service

admin\_user = neutron

admin\_password = *NEUTRON\_PASS*

[DEFAULT]

...

nova\_metadata\_ip = *controller*

[DEFAULT]

...

metadata\_proxy\_shared\_secret = *METADATA\_SECRET*

[DEFAULT]

...

verbose = True

**On the *controller* node**, edit the /etc/nova/nova.conf file and complete the following action:

[neutron]

...

service\_metadata\_proxy = True

metadata\_proxy\_shared\_secret = *METADATA\_SECRET*

On the *controller* node, restart the Compute *API* service:

# **systemctl restart openstack-nova-api.service**

# **systemctl enable openvswitch.service**

# **systemctl start openvswitch.service**

# **ovs-vsctl add-br br-ex**

Add a port to the external bridge that connects to the physical external network interface:

Replace *INTERFACE\_NAME* with the actual interface name. For example, *eth2* or *ens256*.

# **ovs-vsctl add-port br-ex *INTERFACE\_NAME***

Depending on your network interface driver, you may need to disable

*generic receive offload (GRO)* to achieve suitable throughput between your

instances and the external network.

To temporarily disable GRO on the external network interface while testing

your environment:

# **ethtool -K *INTERFACE\_NAME* gro off**

**To finalize the installation**

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

# **cp /usr/lib/systemd/system/neutron-openvswitch-agent.service \**

**/usr/lib/systemd/system/neutron-openvswitch-agent.service.orig**

# **sed -i 's,plugins/openvswitch/ovs\_neutron\_plugin.ini,plugin.ini,g' \**

**/usr/lib/systemd/system/neutron-openvswitch-agent.service**

# **systemctl enable neutron-openvswitch-agent.service neutron-l3-agent.service \**

**neutron-dhcp-agent.service neutron-metadata-agent.service \**

**neutron-ovs-cleanup.service**

# **systemctl start neutron-openvswitch-agent.service neutron-l3-agent.service \**

**neutron-dhcp-agent.service neutron-metadata-agent.service**

**Note**

Do not explictly start the neutron-ovs-cleanup service.

$ **source admin-openrc.sh**

$ **neutron agent-list**

**Install and configure compute node**

The compute node handles connectivity and *security groups* for instances.

1. Edit the /etc/sysctl.conf file to contain the following parameters:

net.ipv4.conf.all.rp\_filter=0

net.ipv4.conf.default.rp\_filter=0

2. Implement the changes:

# **sysctl –p**

# **yum install openstack-neutron-ml2 openstack-neutron-openvswitch**

Edit the /etc/neutron/neutron.conf

[DEFAULT]

...

rpc\_backend = rabbit

rabbit\_host = *controller*

rabbit\_password = *RABBIT\_PASS*

[DEFAULT]

...

auth\_strategy = keystone

[keystone\_authtoken]

...

auth\_uri = http://*controller*:5000/v2.0

identity\_uri = http://*controller*:35357

admin\_tenant\_name = service

admin\_user = neutron

admin\_password = *NEUTRON\_PASS*

[DEFAULT]

...

core\_plugin = ml2

service\_plugins = router

allow\_overlapping\_ips = True

[DEFAULT]

...

verbose = True

**To configure the Modular Layer 2 (ML2) plug-in**

The ML2 plug-in uses the Open vSwitch (OVS) mechanism (agent) to build the virtual networking

framework for instances.

Edit the /etc/neutron/plugins/ml2/ml2\_conf.ini

[ml2]

...

type\_drivers = flat,vlan

tenant\_network\_types = vlan

mechanism\_drivers = openvswitch

[securitygroup]

...

enable\_security\_group = True

enable\_ipset = True

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

**[ovs]**

**#local\_ip = 106.39.78.56**

**#enable\_tunneling = True**

**local\_ip = 106.39.78.56**

**#enable\_tunneling = True**

**tenant\_network\_type = vlan**

**integration\_bridge = br-int**

**network\_vlan\_ranges = physnet1:900:910**

**bridge\_mappings = physnet1:br-ex**

**To configure the Open vSwitch (OVS) service**

# **systemctl enable openvswitch.service**

# **systemctl start openvswitch.service**

**To configure Compute to use Networking**

By default, distribution packages configure Compute to use legacy networking. You must

reconfigure Compute to manage networks through Networking

Edit the /etc/nova/nova.conf

[DEFAULT]

...

network\_api\_class = nova.network.neutronv2.api.API

security\_group\_api = neutron

linuxnet\_interface\_driver = nova.network.linux\_net.LinuxOVSInterfaceDriver

firewall\_driver = nova.virt.firewall.NoopFirewallDriver

[neutron]

...

url = http://*controller*:9696

auth\_strategy = keystone

admin\_auth\_url = http://*controller*:35357/v2.0

admin\_tenant\_name = service

admin\_username = neutron

admin\_password = *NEUTRON\_PASS*

**To finalize the installation**

The Networking service initialization scripts expect a symbolic link /etc/neutron/

plugin.ini pointing to the ML2 plug-in configuration file, /etc/neutron/plugins/ml2/ml2\_conf.ini. If this symbolic link does not exist, create it using the following command

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

# **cp /usr/lib/systemd/system/neutron-openvswitch-agent.service \**

**/usr/lib/systemd/system/neutron-openvswitch-agent.service.orig**

# **sed -i 's,plugins/openvswitch/ovs\_neutron\_plugin.ini,plugin.ini,g' \**

**/usr/lib/systemd/system/neutron-openvswitch-agent.service**

# **systemctl restart openstack-nova-compute.service**

# **systemctl enable neutron-openvswitch-agent.service**

# **systemctl start neutron-openvswitch-agent.service**

$ **source admin-openrc.sh**

$ **neutron agent-list**

**创建网络**

**[root@jyw-o-control01 ~]# neutron net-create --provider:physical\_network=physnet1 --provider:network\_type=vlan --provider:segmentation\_id=901 --shared INSIDE\_NET**

**[root@jyw-o-control01 ~]# neutron net-create --provider:physical\_network=physnet1 --provider:network\_type=vlan --provider:segmentation\_id=902 --shared DMZ\_NET**

**# 创建子网**

**neutron subnet-create INSIDE\_NET 10.10.10.0/24 --name INSIDE\_SUBNET --no-gateway --host-route destination=0.0.0.0/0,nexthop=10.10.10.1 --allocation-pool start=10.10.10.10,end=10.10.10.254 --dns-nameservers list=true 8.8.8.7 8.8.8.8**

**[root@jyw-o-control01 ~]# neutron subnet-create INSIDE\_NET 10.10.10.0/24 --name INSIDE\_SUBNET --no-gateway --host-route destination=0.0.0.0/0,nexthop=10.10.10.1 --allocation-pool start=10.10.10.10,end=10.10.10.254 --dns-nameservers list=true 8.8.8.7 8.8.8.8**

**# 创建子网**

**neutron subnet-create DMZ\_NET 172.16.0.0/24 --name DMZ\_SUBNET --no-gateway --host-route destination=0.0.0.0/0,nexthop=172.16.0.1 --allocation-pool start=172.16.0.100,end=172.16.0.199 --dns-nameservers list=true 8.8.8.7 8.8.8.8**

**[root@jyw-o-control01 ~]# neutron subnet-create DMZ\_NET 172.16.0.0/24 --name DMZ\_SUBNET --no-gateway --host-route destination=0.0.0.0/0,nexthop=172.16.0.1 --allocation-pool start=172.16.0.100,end=172.16.0.199 --dns-nameservers list=true 8.8.8.7 8.8.8.8**

**ip netns exec qdhcp-24124257-2628-4ac2-ae49-45e364ce4add ping 10.10.10.10**

#######创建外部网络

$ **source admin-openrc.sh**

$ **neutron net-create ext-net --router:external True \**

**--provider:physical\_network external --provider:network\_type flat**

$ **neutron subnet-create ext-net --name ext-subnet \**

**--allocation-pool start=*FLOATING\_IP\_START*,end=*FLOATING\_IP\_END* \**

**--disable-dhcp --gateway *EXTERNAL\_NETWORK\_GATEWAY EXTERNAL\_NETWORK\_CIDR***

$ **neutron subnet-create ext-net --name ext-subnet \**

**--allocation-pool start=203.0.113.101,end=203.0.113.200 \**

**--disable-dhcp --gateway 203.0.113.1 203.0.113.0/24**

$ **source demo-openrc.sh**

$ **neutron net-create demo-net**

$ **neutron subnet-create demo-net --name demo-subnet \**

**--gateway *TENANT\_NETWORK\_GATEWAY TENANT\_NETWORK\_CIDR***

$ **neutron subnet-create demo-net --name demo-subnet \**

**--gateway 192.168.1.1 192.168.1.0/24**

$ **neutron router-create demo-router**

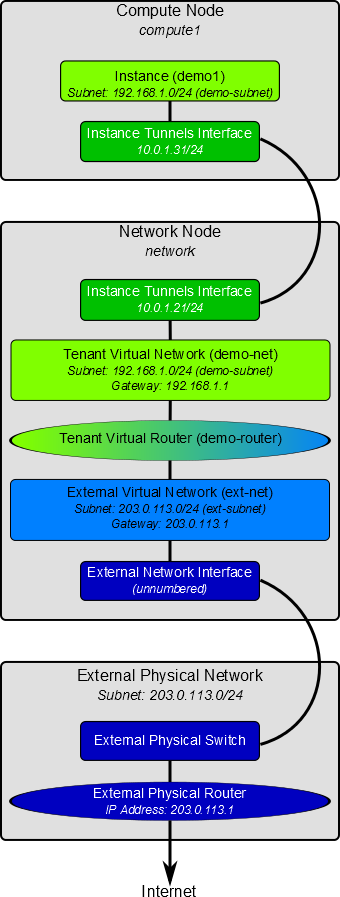
$ **neutron router-interface-add demo-router demo-subnet**

$ **neutron router-gateway-set demo-router ext-net**

**检测到路由网关是否通**

$ **ping -c 4 203.0.113.101**

#### 6.2.3 贴出实例到外网网络流程图



## 7. Dashboard Web服务

多说一句，Dashboard是基于python下的框架Django。所以如果想要使用这个功能，python版本必须支持，推荐使用Python2.6，因为Openstack JUNO官方使用的python版本也是Python2.6。

### 7.1 软件包安装

在控制节点

# **yum install openstack-dashboard httpd mod\_wsgi memcached pythonmemcached**

Edit the /etc/openstack-dashboard/local\_settings

OPENSTACK\_HOST = "*controller*"

ALLOWED\_HOSTS = ['\*']

CACHES = {

'default': {

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

'LOCATION': '127.0.0.1:11211',

}

}

TIME\_ZONE = "*TIME\_ZONE*"

# **setsebool -P httpd\_can\_network\_connect on**

# **chown -R apache:apache /usr/share/openstack-dashboard/static**

# **systemctl enable httpd.service memcached.service**

# **systemctl start httpd.service memcached.service**

接下来，就可以通过以下URL地址访问了

 http://controller/dashboard

## 8. Cinder服务

### 8.1 Add the Block Storage service

### 8.2 创建数据库

$ **mysql -u root –p**

**CREATE DATABASE cinder;**

**GRANT ALL PRIVILEGES ON cinder.\* TO 'cinder'@'localhost' \**

**IDENTIFIED BY '*CINDER\_DBPASS*';**

**GRANT ALL PRIVILEGES ON cinder.\* TO 'cinder'@'%' \**

**IDENTIFIED BY '*CINDER\_DBPASS*';**

### 8.3 创建对应账户

$ **source admin-openrc.sh**

$ **keystone user-create --name cinder --pass *CINDER\_PASS***

$ **keystone user-role-add --user cinder --tenant service --role admin**

$ **keystone service-create --name cinder --type volume \**

**--description "OpenStack Block Storage"**

$ **keystone service-create --name cinderv2 --type volumev2 \**

**--description "OpenStack Block Storage"**

$ **keystone endpoint-create \**

**--service-id $(keystone service-list | awk '/ volume / {print $2}') \**

**--publicurl http://*controller*:8776/v1/%\(tenant\_id\)s \**

**--internalurl http://*controller*:8776/v1/%\(tenant\_id\)s \**

**--adminurl http://*controller*:8776/v1/%\(tenant\_id\)s \**

**--region regionOne**

$ **keystone endpoint-create \**

**--service-id $(keystone service-list | awk '/ volumev2 / {print $2}') \**

**--publicurl http://*controller*:8776/v2/%\(tenant\_id\)s \**

**--internalurl http://*controller*:8776/v2/%\(tenant\_id\)s \**

**--adminurl http://*controller*:8776/v2/%\(tenant\_id\)s \**

**--region regionOne**

### 8.4 安装相应软件或者插件

# **yum install openstack-cinder python-cinderclient python-oslo-db**

### 8.5 修改配置文件

Edit the /etc/cinder/cinder.conf

[database]

...

connection = mysql://cinder:*CINDER\_DBPASS*@controller/cinder

[DEFAULT]

...

rpc\_backend = rabbit

rabbit\_host = *controller*

rabbit\_password = *RABBIT\_PASS*

[DEFAULT]

...

auth\_strategy = keystone

[keystone\_authtoken]

...

auth\_uri = http://*controller*:5000/v2.0

identity\_uri = http://*controller*:35357

admin\_tenant\_name = service

admin\_user = cinder

admin\_password = *CINDER\_PASS*

[DEFAULT]

...

my\_ip = 10.0.0.11

[DEFAULT]

...

verbose = True

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

# **systemctl enable openstack-cinder-api.service openstack-cinderscheduler.**

**service**

# **systemctl start openstack-cinder-api.service openstack-cinder-scheduler.**

**Service**

**Install and configure a storage node**

### 8.6 存储节点

#### 8.6.1 安装软件

# **yum install openstack-cinder targetcli python-oslo-db MySQL-python**

#### 8.6.2 配置配置文件

[database]

...

connection = mysql://cinder:*CINDER\_DBPASS*@*controller*/cinder

[DEFAULT]

...

rpc\_backend = rabbit

rabbit\_host = *controller*

rabbit\_password = *RABBIT\_PASS*

[DEFAULT]

...

auth\_strategy = keystone

[keystone\_authtoken]

...

auth\_uri = http://*controller*:5000/v2.0

identity\_uri = http://*controller*:35357

admin\_tenant\_name = service

admin\_user = cinder

admin\_password = *CINDER\_PASS*

[DEFAULT]

...

my\_ip = *MANAGEMENT\_INTERFACE\_IP\_ADDRESS*

[DEFAULT]

...

glance\_host = *controller*

[DEFAULT]

...

iscsi\_helper = lioadm

[DEFAULT]

...

verbose = True

# **systemctl enable openstack-cinder-volume.service target.service**

# **systemctl start openstack-cinder-volume.service target.service**

$ **source admin-openrc.sh**

$ **cinder service-list**

## 9. Heat服务

### 9.1 软件包安装

# yum install openstack-heat-api openstack-heat-engine \

openstack-heat-api-cfn

### 9.2 配置数据库文件

# yum install openstack-heat-api openstack-heat-engine \

openstack-heat-api-cfn

### 9.3 创建数据库并赋予相应权限

$ mysql -u root -p

mysql> CREATE DATABASE heat;

mysql> GRANT ALL PRIVILEGES ON heat.\* TO 'heat'@'localhost' IDENTIFIED BY '*HEAT\_DBPASS*';

mysql> GRANT ALL PRIVILEGES ON heat.\* TO 'heat'@'%' \

IDENTIFIED BY '*HEAT\_DBPASS*';

### 9.4 生成数据库表

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

### 9.5 配置Heat配置文件

# openstack-config --set /etc/heat/heat.conf DEFAULT qpid\_hostname *controller*

$ keystone user-create --name=heat --pass=*HEAT\_PASS* \

--email=*heat@example.com*

$ keystone user-role-add --user=heat --tenant=service --role=admin

# openstack-config --set /etc/heat/heat.conf keystone\_authtoken \

auth\_uri http://*controller*:5000/v2.0

# openstack-config --set /etc/heat/heat.conf keystone\_authtoken \

auth\_port 35357

# openstack-config --set /etc/heat/heat.conf keystone\_authtoken \

auth\_protocol http

# openstack-config --set /etc/heat/heat.conf keystone\_authtoken \

admin\_tenant\_name service

# openstack-config --set /etc/heat/heat.conf keystone\_authtoken \

admin\_user heat

# openstack-config --set /etc/heat/heat.conf keystone\_authtoken \

admin\_password*HEAT\_PASS*

# openstack-config --set /etc/heat/heat.conf ec2authtoken \

auth\_uri http://*controller*:5000/v2.0

### 9.6 注册heat服务

$ keystone service-create --name=heat --type=orchestration \

--description="Orchestration"

$ keystone endpoint-create \

--service-id=$(keystone service-list | awk '/ orchestration / {print $2}') \

--publicurl=http://*controller*:8004/v1/%\(tenant\_id\)s \

--internalurl=http://*controller*:8004/v1/%\(tenant\_id\)s \

--adminurl=http://*controller*:8004/v1/%\(tenant\_id\)s

$ keystone service-create --name=heat-cfn --type=cloudformation \

--description="Orchestration CloudFormation"

$ keystone endpoint-create \

--service-id=$(keystone service-list | awk '/ cloudformation / {print $2}') \

--publicurl=http://*controller*:8000/v1 \

--internalurl=http://*controller*:8000/v1 \

--adminurl=http://*controller*:8000/v1

### 9.7 创建heat账户

$ keystone role-create --name heat\_stack\_user

# openstack-config --set /etc/heat/heat.conf \

DEFAULT heat\_metadata\_server\_url http://*10.0.0.11*:8000

# openstack-config --set /etc/heat/heat.conf \

DEFAULT heat\_waitcondition\_server\_url http://*10.0.0.11*:8000/v1/waitcondition

### 9.8 启动heat服务并设置开机启动

# service openstack-heat-api start

# service openstack-heat-api-cfn start

# service openstack-heat-engine start

# chkconfig openstack-heat-api on

# chkconfig openstack-heat-api-cfn on

# chkconfig openstack-heat-engine on

### 9.9 验证

Heat需要事先创建模板文件，官方格式为yaml格式。如下：

heat\_template\_version: 2013-05-23

description: Test Template

parameters:

ImageID:

type: string

description: Image use to boot a server

NetID:

type: string

description: Network ID for the server

resources:

server1:

type: OS::Nova::Server

properties:

name: "Test server"

image: { get\_param: ImageID }

flavor: "m1.tiny"

networks:

- network: { get\_param: NetID }

outputs:

server1\_private\_ip:

description: IP address of the server in the private network

value: { get\_attr: [ server1, first\_address ] }

如上所示，为一个创建实例的yaml模块文件

$ NET\_ID=$(nova net-list | awk '/ demo-net / { print $2 }')

$ heat stack-create -f test-stack.yml \

-P "ImageID=cirros-0.3.2-x86\_64;NetID=$NET\_ID" testStack

然后通过指定网段和相应的镜像文件以及模板文件既可以创建相应的实例

## 10. 热迁移

热迁移配置很简单，只需要了解了，就OK了

和上一样，我们采用的是 两个计算节点+一个控制节点（热迁移不涉及的，没有列出）

采用热迁移【共享存储：即将所有的计算节点都挂载在NFS共享磁盘上】

为什么采用共享存储热迁移？

采用共享存储热迁移技术，迁移的时候，只需要将内存迁移到另外一台KVM机器上即可。系统文件不需要从一台机器拷贝到另外一台机器。避免因为网络，磁盘IO等各种其他因素的干扰。

计算节点1：compute1: 111.204.241.187 - 192.168.5.220

计算节点2：compute2:111.204.241.240 - 192.168.5.209

我将NFS服务器放在了计算节点1上。【生成环境，需要单独放到一台NFS服务器上】

【**PS:nfs配置，在此不再阐述，google一下，需要注意，挂载的时需要加上nolock 参数**】

并注意权限设置问题

### 10.1. Openstack热迁移计算节点

#### 10.1.1 配置如下：

首先需要将实例存储位置都指向NFS上的相同位置，这里采用的是/data1

[root@jyw-w-kvmtest2 ~]# df -Th

Filesystem Type Size Used Avail Use% Mounted on

…..

192.168.5.220:/ nfs4 276G 2.9G 259G 2% /data1

[root@jyw-w-kvmtest2 ~]# ls /data1/nfs/lib/nova/ ##需要将/var/lib/nova下的全部拷贝到这里

buckets images instances keys networks tmp

Vim /etc/nova/nova.conf [两台计算节点都需要，如果有多台，多台都需要]

state\_path=/data1/nfs/lib/nova ##必须，否则不成功

instances\_path=/data1/nfs/lib/nova/instances ###必须，否则不成功

vncserver\_listen = 0.0.0.0 ###必须，否则不成功

###########

在各个主机直接，需要配置DNS服务，当然可以直接修改hosts文件，让个机器之间，可以通过主机名互通

例如：

192.168.5.220 jyw-o-kvmtest.light.soufun.com jyw-o-kvmtest

192.168.5.209 jyw-w-kvmtest2.light.soufun.com jyw-w-kvmtest2

#### 10.1.2 /etc/export

/data1 192.168.4.0/22(rw,sync,fsid=0,no\_all\_squash,no\_root\_squash) 111.204.241.128/25(rw,sync,fsid=0,no\_all\_squash,no\_root\_squash)

针对/data1/nfs下面的文件，我设置权限为777,上线的话，可以将权限设置小一点。

至少需要执行权限，倒不如设置777，重启nfs服务

#### 10.1.3. 修改/etc/nova/nova.conf

修改/etc/nova/nova.conf配置文件，让他支持热迁移，默认是不支持，需要额外配置

| **Configuration option = Default value** | **Description** |
| --- | --- |
| **[DEFAULT]** | |
| live\_migration\_retry\_count = *30* | (IntOpt) Number of 1 second retries needed in live\_migration |
| **[libvirt]** | |
| live\_migration\_bandwidth = *0* | (IntOpt) Maximum bandwidth to be used during migration, in Mbps |
| live\_migration\_flag = *VIR\_MIGRATE\_UNDEFINE\_SOURCE, VIR\_MIGRATE\_PEER2PEER, VIR\_MIGRATE\_LIVE, VIR\_MIGRATE\_TUNNELLED* | (StrOpt) Migration flags to be set for live migration |
| live\_migration\_uri = *qemu+tcp://%s/system* | (StrOpt) Migration target URI (any included "%s" is replaced with the migration target hostname) |

列出：

live\_migration\_retry\_count = *30*

live\_migration\_bandwidth = *0*

live\_migration\_flag = *VIR\_MIGRATE\_UNDEFINE\_SOURCE, VIR\_MIGRATE\_PEER2PEER, VIR\_MIGRATE\_LIVE, VIR\_MIGRATE\_TUNNELLED*

live\_migration\_uri = *qemu+tcp://%s/system*

live\_migration\_flag=VIR\_MIGRATE\_UNDEFINE\_SOURCE,VIR\_MIGRATE\_PEER2PEER,VIR\_MIGRATE\_LIVE, VIR\_MIGRATE\_TUNNELLED ###需要修改这条的默认

#### 10.1.4. 修改libvirtd

cat /etc/libvirt/libvirtd.conf |egrep -v "^#|^$"

listen\_tls = 0

listen\_tcp = 1

auth\_tcp = "none"

vim /etc/sysconfig/libvirtd ##开启监听libvirtd，需要放开16509默认端口，可以修改

LIBVIRTD\_ARGS="--listen"

#### 10.1.5 块迁移

##### 首先必须配置无密码访问

将nova账户设置为可登陆，修改/etc/passwd nova /bin/bash

然后设置nova账户公私钥，ssh-keygen

然后传到所有的计算节点，因为在openstack中，各个机器迁移的时候，是随机的，更具各个机器的负载情况，然后nova调度器进行调度。

ssh-copy-cp –I .ssh/id\_rsa.pub nova@$ip

##### 修改一下的配置文件

/etc/libvirt/libvirtd.conf

listen\_tls = 0

listen\_tcp = 1

auth\_tcp = "none"

$ nova live-migration --block\_migrate med server-01

/etc/nova/nova.conf中下面配置，也可以使用系统默认的

block\_migration\_flag=VIR\_MIGRATE\_UNDEFINE\_SOURCE,VIR\_MIGRATE\_PEER2PEER,VIR\_MIGRATE\_NON\_SHARED\_INC

##### 修改数据库，设置多一些数据库连接数

当控制节点中提示如下错误时，是数据库连接数做了限制，解决方案在/etc/my.cnf文件里面添加如下信息：

##connection###########

max\_connections = 1700

max\_connect\_errors = 10000

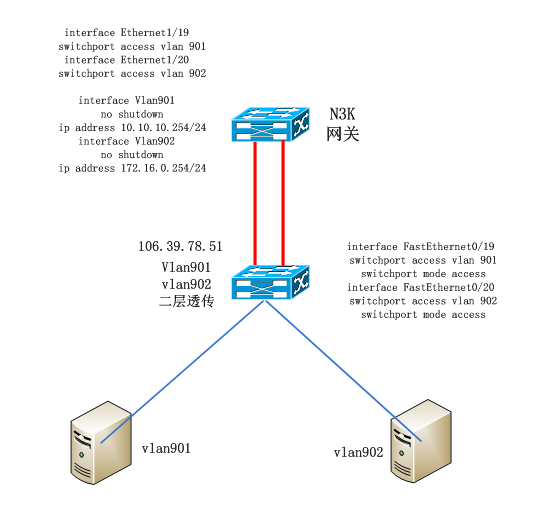
back\_log = 1000

interactive\_timeout = 150

wait\_timeout = 150

## 11 交换机上配置如下

### 11.1 架构以及配置如下图所示



## Q&A

1. 在# su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf \

--config-file /etc/neutron/plugin.ini upgrade head" neutron时，在日志中找到ImportError: Class ml2 cannot be found ，解决办法如下：

# openstack-config --set /etc/neutron/neutron.conf DEFAULT \

core\_plugin neutron.plugins.ml2.plugin.Ml2Plugin

# openstack-config --set /etc/neutron/neutron.conf DEFAULT \

service\_plugins neutron.services.l3\_router.l3\_router\_plugin.L3RouterPlugin

2. 在配置好了所有的服务之后，通过$ip/dashboard访问的时候，提示没有权限，需要修改一下文件

/etc/openstack-dashboard/local\_settings 中添加本机的IP地址即可，如果不用考虑安全问题，直接使用”\*”,这种方式不推荐，不够安全

ALLOWED\_HOSTS = ['localhost', 'my-desktop', '111.204.241.184']

3. 如果使用虚拟机做环境，必须将网络模式修改为混杂模式，即允许不是自己的数据包通过。

4. OpenStack中虚拟化技术在哪里配置

主要两个文件：nova.conf nova-compute.conf。

其中的虚拟技术应该一致，比如都为kvm，或者都为qemu。

5. noVNC不能连接到实例，提示“Connection Refuesd”，什么原因  
可能是控制节点在收到vnc请求的时候，无法解析计算节点的主机名，从而无法和计算节点上的实例建立连接。  
另外可能是，当前浏览器不支持或者不能访问，将计算节点的ip和主机名的对应关系加入到控制节点的/etc/hosts文件中

6. 如果创建实例之后，不能够获取分配的IP地址，请查看是否已经启动了对应的DNSMASQ服务进程。dnsmasq服务是监控dhcp服务进程状态的。可以通过查看计算节点的/var/log/message的信息找到相应的错误

7. su -s /bin/sh -c "keystone-manage db\_sync" keystone 这里，如果在数据库中没有生成表，则是keystone.conf文件中的mysql配置错误

8. 在日志中找到显示binding-failed，表示使用的虚拟化技术错误，或者计算节点没有启动，导致服务创建实例。错误： 创建实例 "dhcp-cirros" 失败: 请稍后再试 [错误: No valid host was found. ].

9. /usr/lib64/python2.6/site-packages/Crypto/Util/number.py:57: PowmInsecureWarning: Not using mpz\_powm\_sec. You should rebuild using libgmp >= 5 to avoid timing attack vulnerability.

\_warn("Not using mpz\_powm\_sec. You should rebuild using libgmp >= 5 to avoid timing attack vulnerability.", PowmInsecureWarning，此提示随不是错误，但是感觉就是不爽，其实就是版本问题

10. # cinder create --display-name myVolume 1

ERROR: Unauthorized (HTTP 401)

顾名思义，肯定是认证问题，去查看cinder的认证配置是否正确

11. Failed to schedule\_create\_volume: No valid host was found

###权限不够，使用admin权限即可

12. ERROR: Policy doesn't allow volume\_extension:services to be performed. (HTTP 403)

###权限不够，使用admin权限即可

13. NAT表记录满，导致漫天的流量

/var/log/message日志信息如下：Apr 20 03:27:44 jyw-o-kvmtest kernel: \_\_ratelimit: 3712 callbacks suppressed

Apr 20 03:27:44 jyw-o-kvmtest kernel: nf\_conntrack: table full, dropping packet.

解决办法：

sysctl –w net.netfilter.nf\_conntrack\_max = 2097152

<http://www.cnblogs.com/higkoo/articles/iptables_tunning_for_conntrack.html>

14 dnsmasq **版本不支持，下载dnsmasp2.68**

**15. Command: ['sudo', 'neutron-rootwrap', '/etc/neutron/rootwrap.conf', 'kill', '-9', '19885']**

[**https://wiki.openstack.org/wiki/Rootwrap**](https://wiki.openstack.org/wiki/Rootwrap)

参考文档：

<http://docs.openstack.org/icehouse/install-guide/install/yum/content/install_dashboard.html>

<http://www.openstack.cn/>

<https://ask.openstack.org/zh/questions/>

<https://wiki.openstack.org/wiki/Main_Page>

<http://docs.openstack.org/cli-reference/content/>