**:**

**ODPF3.0版部署手册V1.0**

PVC

2016-5-6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 修订内容 | 日期 | 版本号 | 修订人 | 批准人 |
| 1 | 初稿 | 2016/5/6 | V 1.0 | 杨海利/赵进勇 |  |
| 2 | 修订 | 2016/07/08 | V1.1 | 姜旭 |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

目录

[1 引言 4](#_Toc456017598)

[1.1编写目的 4](#_Toc456017599)

[1.2背景 4](#_Toc456017600)

[1.3参考资料 5](#_Toc456017601)

[2 环境搭建 5](#_Toc456017602)

[2.1 Security OpenSSL 5](#_Toc456017603)

[2.2 Host networking 5](#_Toc456017604)

[2.3 etc/hosts互信 5](#_Toc456017605)

[2.4 selinux设置为disable 6](#_Toc456017606)

[2.5 关闭防火墙 6](#_Toc456017607)

[2.6 配置时间同步 6](#_Toc456017608)

[2.7 安装openstack源和必要的安装包 #yum install centos-release-openstack-mitaka #yum upgrade #yum install openstack-selinux python-openstackclient 7](#_Toc456017609)

[2.8 初始化数据库 (控制节点) 7](#_Toc456017610)

[2.9 RabbitMQ (控制节点) 8](#_Toc456017611)

[2.10 Memcached (控制节点) 8](#_Toc456017612)

[3 Identity服务（控制节点） 9](#_Toc456017613)

[3.1 创建keystone数据库 9](#_Toc456017614)

[3.2 安装keystone 10](#_Toc456017615)

[3.3 配置Apache HTTP服务 12](#_Toc456017616)

[3.4 创建服务实体和API端点 16](#_Toc456017617)

[3.5 创建域、租户、用户和角色 18](#_Toc456017618)

[3.6 验证操作 21](#_Toc456017619)

[3.7 创建OpenStack客户端环境脚本 22](#_Toc456017620)

[4 Image服务（控制节点） 24](#_Toc456017621)

[4.1 创建Image服务数据库 24](#_Toc456017622)

[4.2 运行admin环境变量脚本 25](#_Toc456017623)

[4.3 创建服务证书 25](#_Toc456017624)

[4.4 创建Image服务API端点 26](#_Toc456017625)

[4.5 安装并配置openstack-glance 28](#_Toc456017626)

[4.6 填充image服务数据 32](#_Toc456017627)

[4.7 启动Image服务并设为开机启动 33](#_Toc456017628)

[4.8 验证操作 34](#_Toc456017629)

[5 Compute服务 35](#_Toc456017630)

[5.1 控制节点部分: 35](#_Toc456017631)

[5.2 计算节点部分: 45](#_Toc456017632)

[6 Networking服务 49](#_Toc456017633)

[6.1 控制节点部分 49](#_Toc456017634)

[6.2 计算节点部分 63](#_Toc456017635)

[7 Dashboard（控制节点） 70](#_Toc456017636)

[7.1 安装配置 70](#_Toc456017637)

[7.2 验证操作 73](#_Toc456017638)

[8 Telemetry服务（控制节点） 74](#_Toc456017639)

[8.1 安装NoSql数据库MongoDB 74](#_Toc456017640)

[8.2 准备 75](#_Toc456017641)

[8.3 安装和配置 78](#_Toc456017642)

[8.4 启用Image服务meters 82](#_Toc456017643)

[8.5 启用Compute服务meters 85](#_Toc456017644)

[8.6 启用块存储meters 89](#_Toc456017645)

[8.7 启用对象存储meters 89](#_Toc456017646)

[8.8 报警服务 89](#_Toc456017647)

[8.9 验证操作 96](#_Toc456017648)

[9 Shared File Systems服务 99](#_Toc456017649)

[9.1 安装配置 (控制节点 node163) 99](#_Toc456017650)

[9.1.1 登录数据库 99](#_Toc456017651)

[9.1.2 创建manila服务 99](#_Toc456017652)

[9.1.3 安装配置组件 102](#_Toc456017653)

[9.2 安装配置share node (暂时放到控制节点 node163) 103](#_Toc456017654)

[9.2.1 安装包 103](#_Toc456017655)

[9.2.2 编辑配置文件 104](#_Toc456017656)

[9.2.3 配置Driver support for share servers management 105](#_Toc456017657)

[9.2.4 启动manila-share服务 106](#_Toc456017658)

[9.2.5 验证是否成功 106](#_Toc456017659)

# 引言

## 1.1编写目的

构建OpenStack M版的集群

## 1.2背景

机器:

172.28.217.163 控制节点

172.28.217.49 计算节点

172.28.217.53 计算节点

操作系统: CentOS 7.2

内核: 3.10.0-327.13.1.el7.x86\_64

虚拟化部分的版本:

1. Libvirt 1.2.17-13.el7\_2.4.x86\_64
2. qemu 2.3.0-31.el7\_2.10.1.x86\_64

# 1.3参考资料

http://docs.openstack.org/mitaka/install-guide-rdo/

# 环境搭建

注意：黄色背景的字体需要根据情况替换！！

## Security OpenSSL

## Host networking

## /etc/hosts互信

编辑/etc/hosts文件，添加如下内容

172.28.217.49 node49

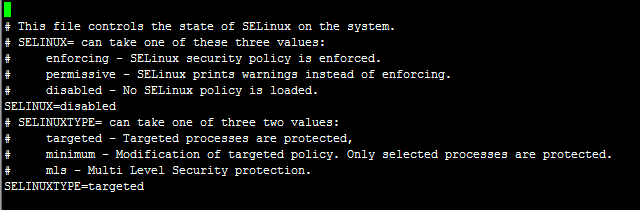
172.28.217.163 node163

172.28.217.53 node53

## selinux设置为disable

编辑/etc/selinux/config 设置SELINUX=disabled 重启生效

#vi /etc/selinux/config



设置及时生效

# setenforce 0

## 关闭防火墙

停止firewall

#systemctl stop firewalld.service

禁止firewall开机启动

#systemctl disable firewalld.service

## 配置时间同步

#yum install chrony

注：控制节点作为时间同步的主节点，计算节点作为从节点与控制节点做时间同步  
客户端49和53(计算节点)上需要如下配置编辑

#vi /etc/chrony.conf  
#server node163 (指定主机名) iburst

在chrony作server的节点(ip地址根据实际修改):

allow 10.0.0.0/24

添加到开机启动  
#systemctl enable chronyd.service

启动服务  
#systemctl start chronyd.service

加定时任务:

1.配置crontab，每五分钟运行一次时间同步   
**echo "\*/5 \* \* \* \* /sbin/service chronyd restart"|crontab**  
crontab -l 可以查看添加好的内容  
2.设置crontab开机启动  
chkconfig --level 35 crond on

## 安装openstack源和必要的安装包 #yum install centos-release-openstack-mitaka #yum upgrade（don’t execute） #yum install openstack-selinux python-openstackclient

## 初始化数据库 (控制节点)

#yum install mariadb mariadb-server python2-PyMySQL

创建并配置客户端文件openstack.cnf

#cd /etc/ my.cnf.d/

#vi openstack.cnf

添加如下信息

[mysqld]

bind-address = 172.28.217.163

default-storage-engine = innodb

innodb\_file\_per\_table

collation-server = utf8\_general\_ci

character-set-server = utf8

设置开机启动

#systemctl enable mariadb.service

启动服务

#systemctl start mariadb.service

数据库的root账户设置密码

#mysql\_secure\_installation

## \*RabbitMQ (控制节点)

安装RabbitMQ

#yum install rabbitmq-server

设置开机启动  
#systemctl enable rabbitmq-server.service

启动RabbitMQ  
#systemctl start rabbitmq-server.service  
添加openstack账户(重起后消失？？)  
#rabbitmqctl add\_user openstack openstack  
权限配置(重起后消失？？)  
#rabbitmqctl set\_permissions openstack ".\*" ".\*" ".\*"

## Memcached (控制节点)

安装Memcached

#yum install memcached python-memcached

设置开机启动  
#systemctl enable memcached.service

启动Memcached  
#systemctl start memcached.service

# Identity服务（控制节点）

## 创建keystone数据库

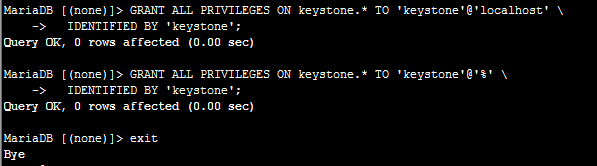
1. 登陆数据库：

#**mysql –u root –p**

1. 创建keystone数据库：**create database keystone;**
2. 分配访问权限

**GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'localhost' IDENTIFIED BY 'keystone';**

**GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'%' IDENTIFIED BY 'keystone';**

****

添加主机node163的访问权限，否则后面创建sever Entity时会报错，

**GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'node163' IDENTIFIED BY 'keystone';**

1. 创建admin-token：4650290a545d9f07b438

**#openssl rand –hex 10**



## 安装keystone

1. Yum安装

**#yum install openstack-keystone httpd mod\_wsgi**



2.修改配置文件：**vi /etc/keystone/keystone.conf**

**[DEFAULT]**

**…**

**admin\_token =** **4650290a545d9f07b438**

**…**

**[database]**

**…**

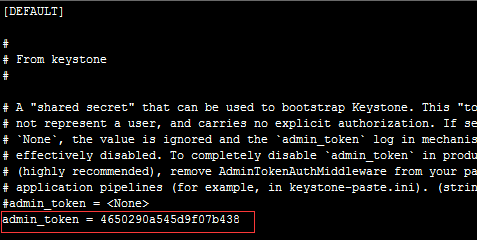
**connection = mysql+pymysql://keystone:keystone@node163/keystone**

**…**

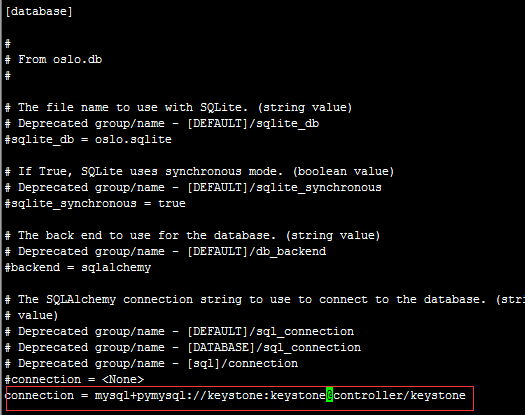
**[token]**

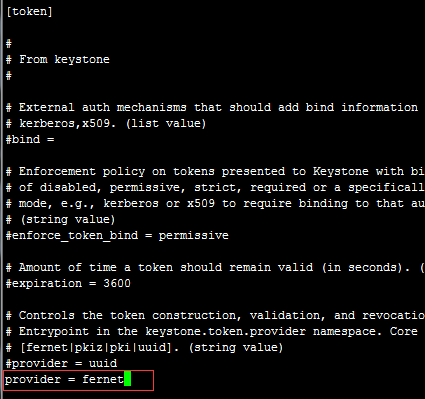
**…**

**provider = fernet**

****

**Admin\_token为2.1中创建的**





3、填充身份数据库：

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



4、初始化Fernet keys:

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



## 配置Apache HTTP服务

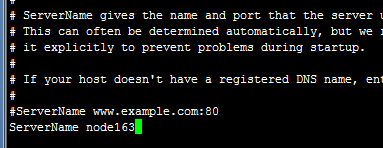
1. 修改配置文件：

#**vi /etc/httpd/conf/httpd.conf**

****

配置ServerName为node163（node163要与hosts文件中配置一致）

**ServerName node163**

****

1. 创建keystone配置文件：

**#vi /etc/httpd/conf.d/wsgi-keystone.conf**

**输入如下内容**

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

<IfVersion >= 2.4>

ErrorLogFormat "%{cu}t %M"

</IfVersion>

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

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

<Directory /usr/bin>

<IfVersion >= 2.4>

Require all granted

</IfVersion>

<IfVersion < 2.4>

Order allow,deny

Allow from all

</IfVersion>

</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

<IfVersion >= 2.4>

ErrorLogFormat "%{cu}t %M"

</IfVersion>

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

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

<Directory /usr/bin>

<IfVersion >= 2.4>

Require all granted

</IfVersion>

<IfVersion < 2.4>

Order allow,deny

Allow from all

</IfVersion>

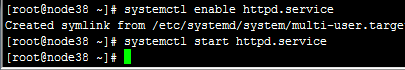
</Directory>

</VirtualHost>

3、开启apache HTTP服务并配置开机启动

**# systemctl enable httpd.service**

**# systemctl start httpd.service**



## 创建服务实体和API端点

1. 创建权限token

**#export OS\_TOKEN=4650290a545d9f07b438**



1. 配置endpoint URL

**#export OS\_URL=http://node163:35357/v3**



1. 配置Identity API的版本

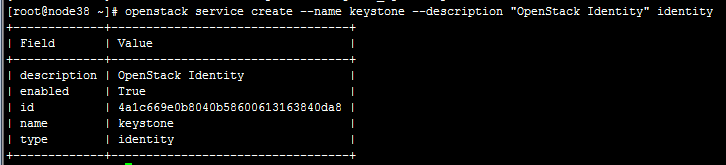
**#export OS\_IDENTITY\_API\_VERSION=3**



1. 创建服务实体身份服务

**#openstack service create \**

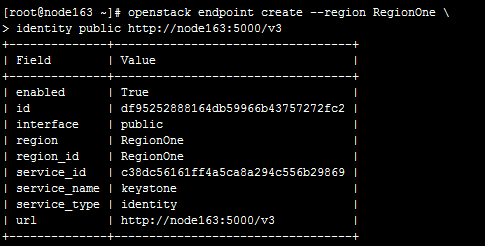
**--name keystone --description "OpenStack Identity" identity**



1. 创建身份服务API端点

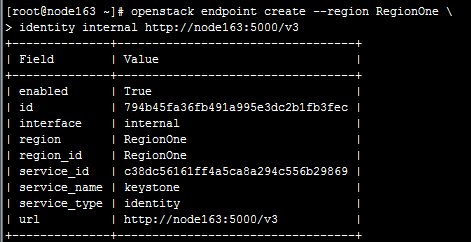
**#openstack endpoint create --region RegionOne \**

**identity public http://node163:5000/v3**



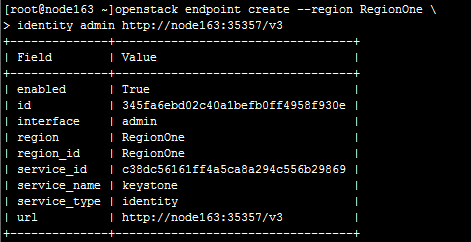
**#openstack endpoint create --region RegionOne \**

**identity internal http://node163:5000/v3**



**#openstack endpoint create --region RegionOne \**

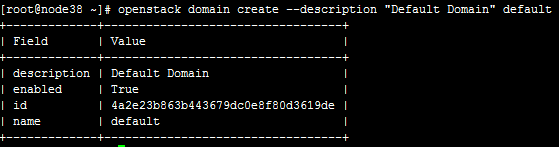
**identity admin http://node163:35357/v3**



## 创建域、租户、用户和角色

1. 创建域“default”

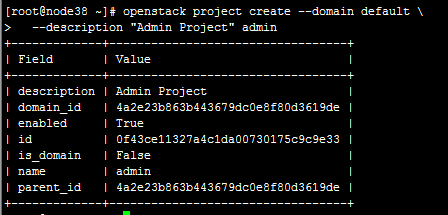
**#openstack domain create --description "Default Domain" default**



1. 创建租户“admin”

**#openstack project create --domain default \**

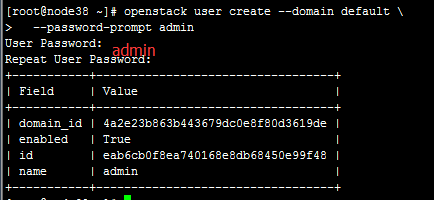
**--description "Admin Project" admin**



1. 创建用户“admin”

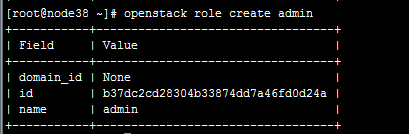
**#openstack user create --domain default \**

**--password-prompt admin**



1. 创建角色“admin”

**#openstack role create admin**



1. 添加“admin”角色到“admin”租户和用户

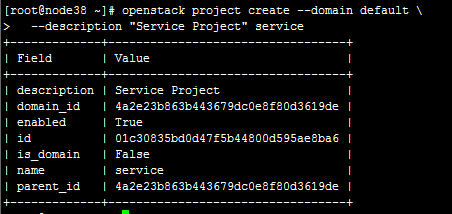
**#openstack role add --project admin --user admin admin**



1. 创建一个Server租户

**#openstack project create --domain default \**

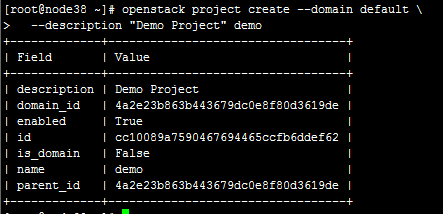
**--description "Service Project" service**



1. 创建一个租户“demo”

**#openstack project create --domain default \**

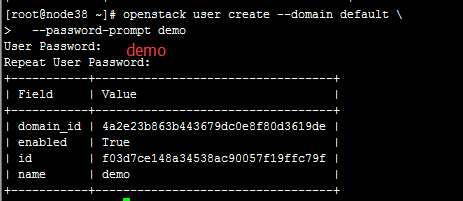
**--description "Demo Project" demo**



1. 创建一个用户“demo”

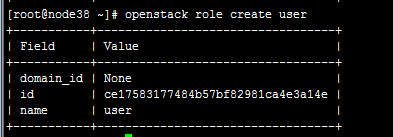
**#openstack user create --domain default \**

**--password-prompt demo**



1. 创建一个角色“user”

**#openstack role create user**



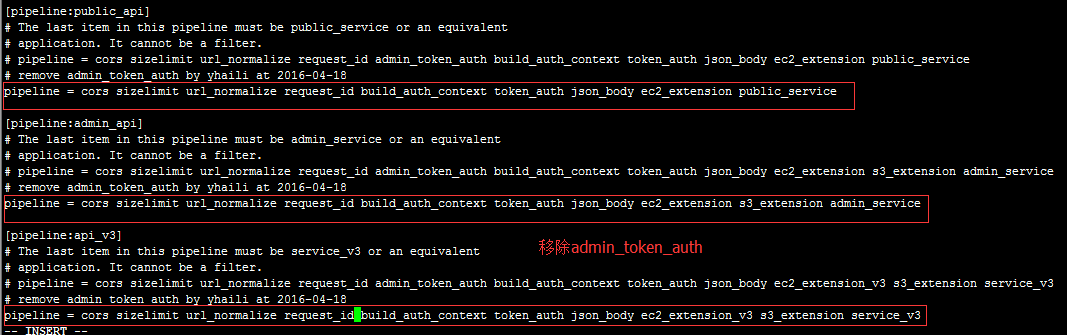
1. 添加“user”角色到“demo”租户和用户

**#openstack role add --project demo --user demo user**



## 验证操作

1. 编辑文件/etc/keystone/keystone-paste.ini 并从[pipeline:public\_api],[pipeline:admin\_api], 和[pipeline:api\_v3]中移除”admin\_token\_auth”



1. 设置临时OS\_TOKEN OS\_URL环境变量

**#unset OS\_TOKEN OS\_URL**

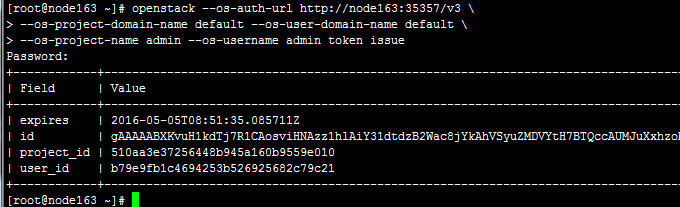


1. 使用管理用户,请求身份验证令牌:

**#openstack --os-auth-url http://node163:35357/v3 \**

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

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

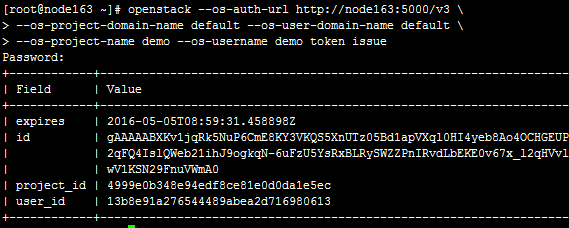


1. 使用demo用户,请求身份验证令牌

**#openstack --os-auth-url http://node163:5000/v3 \**

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

**--os-project-name demo --os-username demo token issue**



## 创建OpenStack客户端环境脚本

1. 创建admin-openrc 脚本，编辑内容如下:

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

export OS\_AUTH\_URL=http://node163:35357/v3

export OS\_IDENTITY\_API\_VERSION=3

export OS\_IMAGE\_API\_VERSION=2

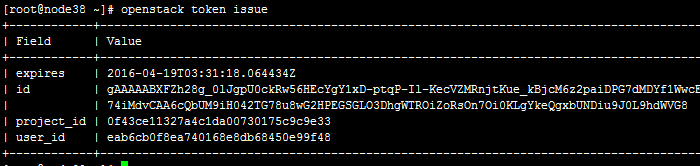
使用脚本：

**#. admin-openrc**



请求身份验证令牌：

**#openstack token issue**



1. 创建demo-openrc脚本，编辑内容如下:

export OS\_PROJECT\_DOMAIN\_NAME=default

export OS\_USER\_DOMAIN\_NAME=default

export OS\_PROJECT\_NAME=demo

export OS\_USERNAME=demo

export OS\_PASSWORD=demo

export OS\_AUTH\_URL=http://node163:5000/v3

export OS\_IDENTITY\_API\_VERSION=3

export OS\_IMAGE\_API\_VERSION=2

# Image服务（控制节点）

## 创建Image服务数据库

1. root用户登陆mysql:

#**mysql -u root -p**



1. 创建glance数据库

**#Create database glance;**

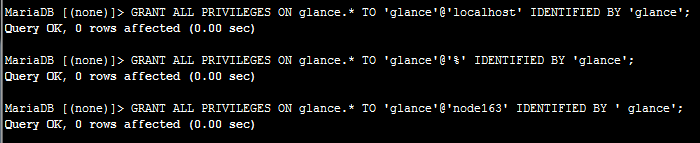


1. 分配访问权限

**#GRANT ALL PRIVILEGES ON glance.\* TO 'glance'@'localhost' IDENTIFIED BY 'glance';**

**#GRANT ALL PRIVILEGES ON glance.\* TO 'glance'@'%' IDENTIFIED BY 'glance';**

**#GRANT ALL PRIVILEGES ON glance.\* TO 'glance'@' node163' IDENTIFIED BY 'glance';**



1. 退出数据库：

#**exit;**



## 运行admin环境变量脚本

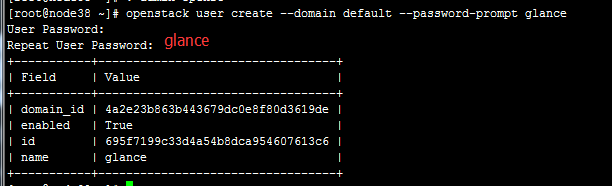
**#. admin-openrc**



## 创建服务证书

1. 创建用户“glance”

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



1. 添加“admin”角色到“glance”用户和“service”租户

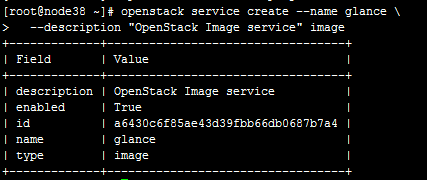
**#openstack role add --project service --user glance admin**



1. 创建“glance”服务实体

**#openstack service create --name glance \**

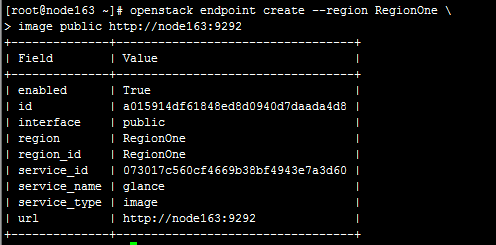
**--description "OpenStack Image" image**



## 创建Image服务API端点

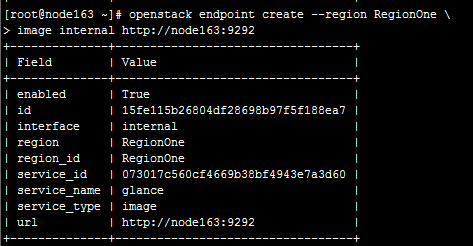
**#openstack endpoint create --region RegionOne \**

**image public http://node163:9292**



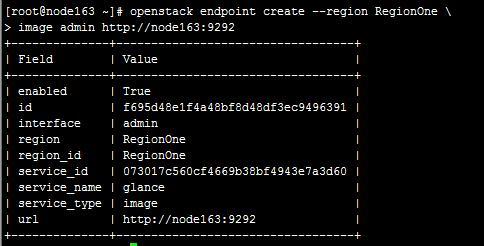
**#openstack endpoint create --region RegionOne \**

**image internal http://node163:9292**



**#openstack endpoint create --region RegionOne \**

**image admin http://node163:9292**



## 安装并配置openstack-glance

1. yum安装

**#yum install openstack-glance**



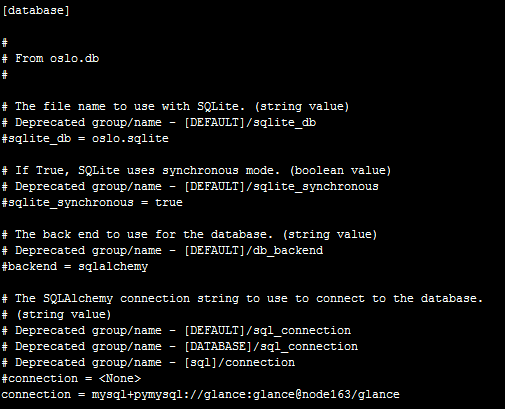
1. 修改配置

#vi /etc/glance/glance-api.conf（添加红框内的配置）

**[database]**

**...**

**connection = mysql+pymysql://glance:glance@node163/glance**



**[keystone\_authtoken]**

**...**

**auth\_uri = http://node163:5000**

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

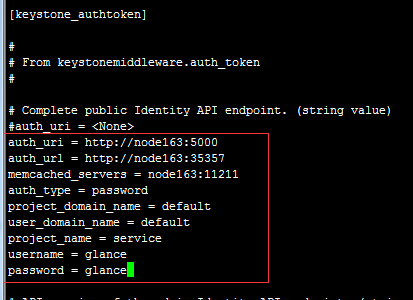
**project\_domain\_name = default**

**user\_domain\_name = default**

**project\_name = service**

**username = glance**

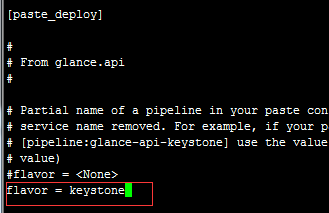
**password = glance**



**[paste\_deploy]**

**...**

**flavor = keystone**



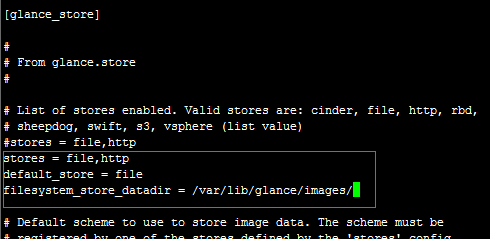
**[glance\_store]**

**...**

**stores = file,http**

**default\_store = file**

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



1. 修改配置

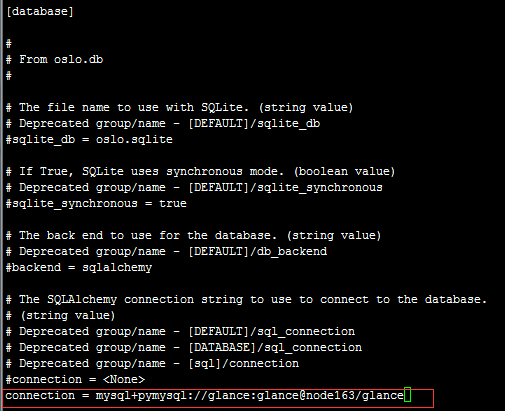
#vi /etc/glance/glance-registry.conf

（添加红框内配置）

**[database]**

**...**

**connection = mysql+pymysql://glance:glance@node163/glance**



**[keystone\_authtoken]**

**...**

**auth\_uri = http://node163:5000**

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

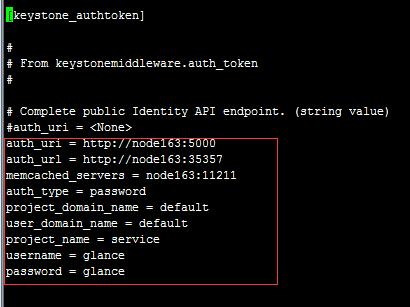
**project\_domain\_name = default**

**user\_domain\_name = default**

**project\_name = service**

**username = glance**

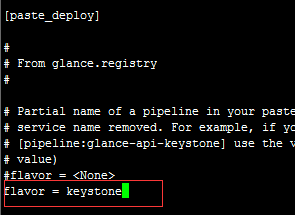
**password = glance**



**[paste\_deploy]**

**...**

**flavor = keystone**



## 填充image服务数据

[root@node163 ~]# **su -s /bin/sh -c "glance-manage db\_sync" glance**

Option "verbose" from group "DEFAULT" is deprecated for removal. Its value may be silently ignored in the future.

/usr/lib/python2.7/site-packages/oslo\_db/sqlalchemy/enginefacade.py:1056: OsloDBDeprecationWarning: EngineFacade is deprecated; please use oslo\_db.sqlalchemy.enginefacade

expire\_on\_commit=expire\_on\_commit, \_conf=conf)

/usr/lib/python2.7/site-packages/pymysql/cursors.py:146: Warning: Duplicate index 'ix\_image\_properties\_image\_id\_name' defined on the table 'glance.image\_properties'. This is deprecated and will be disallowed in a future release.

result = self.\_query(query)

以上信息为填充image服务数据时返回的信息，暂未发现有何影响

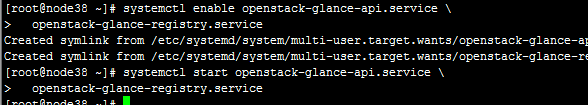
## 启动Image服务并设为开机启动

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

**openstack-glance-registry.service**

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

**openstack-glance-registry.service**



## 验证操作

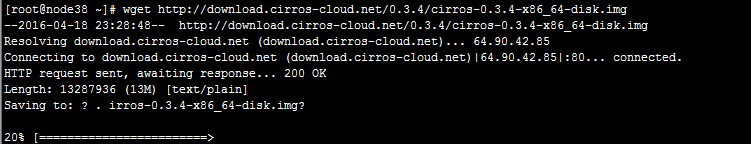
1. 使用admin环境变量脚本

# . admin-openrc



1. 下载源Image

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



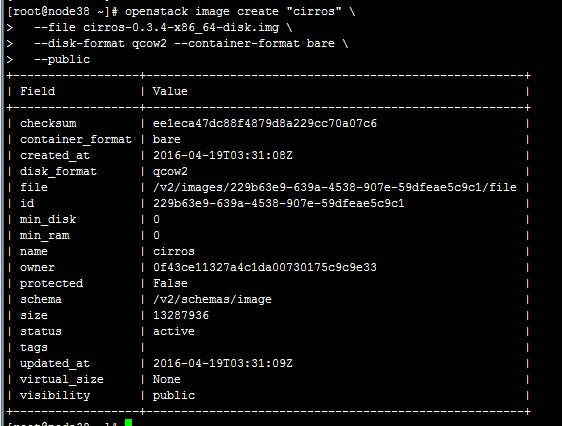
1. 上传Image到Image服务

**#openstack image create "cirros" \**

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

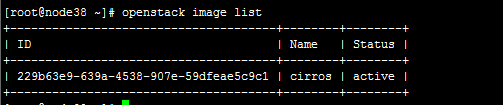
**--disk-format qcow2 --container-format bare \**

**--public**



1. 确认上传的Image并校验属性

**#openstack image list**



# \*Compute服务

**注意:橙色字体的内容需按实际情况修改**

## 控制节点部分:

1控制节点上进行数据库授权

root用户登陆mysql:

#**mysql -u root -p**

创建nova\_api数据库

#**CREATE DATABASE nova\_api;**

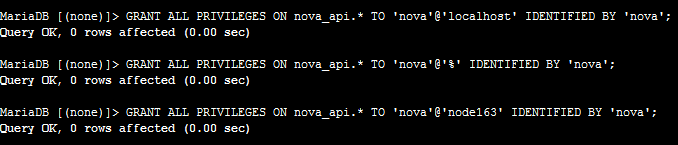


分配访问权限

**#GRANT ALL PRIVILEGES ON nova\_api.\* TO 'nova'@'localhost' IDENTIFIED BY 'nova';**

**#GRANT ALL PRIVILEGES ON nova\_api.\* TO 'nova'@'%' IDENTIFIED BY 'nova';**

**#GRANT ALL PRIVILEGES ON nova\_api.\* TO 'nova'@'node163' IDENTIFIED BY 'nova';**



创建nova数据库

#**CREATE DATABASE nova;**

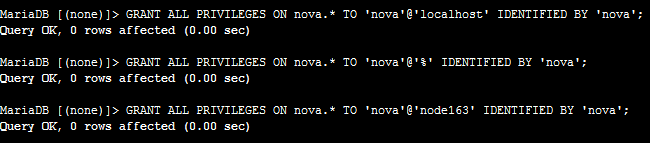


分配访问权限

**#GRANT ALL PRIVILEGES ON nova.\* TO 'nova'@'localhost' IDENTIFIED BY 'nova';**

**#GRANT ALL PRIVILEGES ON nova.\* TO 'nova'@'%' IDENTIFIED BY 'nova';**

**#GRANT ALL PRIVILEGES ON nova.\* TO 'nova'@'node163' IDENTIFIED BY 'nova';**

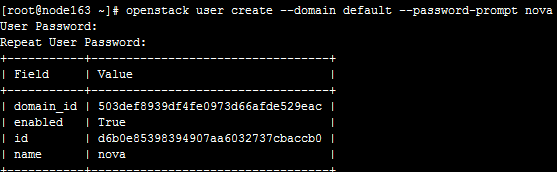


退出数据库



2 控制节点上创建nova用户,并设置密码为nova

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



nova

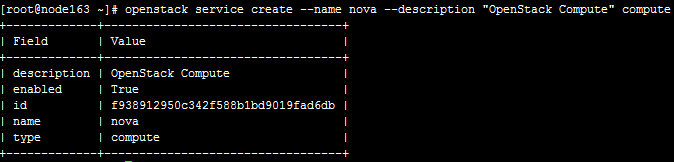
3 控制节点上将admin角色添加到nova用户中

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

****

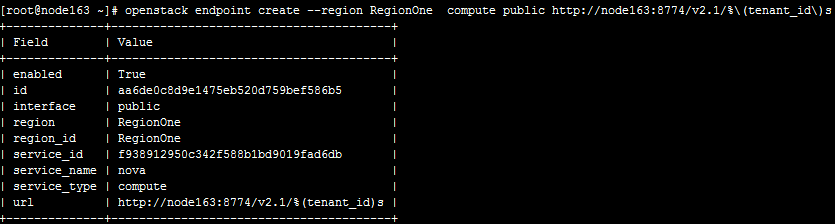
4 控制节点上创建nova服务的实体

**#openstack service create --name nova --description "OpenStack Compute" compute**

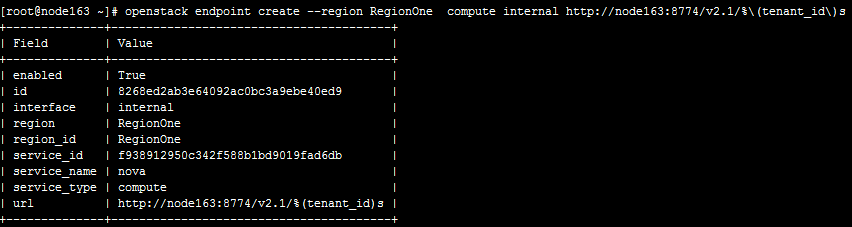


5 控制节点上创建计算服务API终端

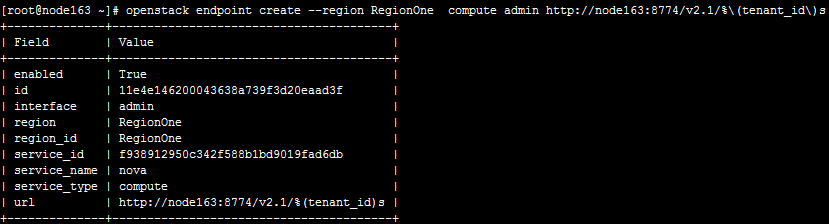
**#openstack endpoint create --region RegionOne compute public** [**http://node163:8774/v2.1/%\(tenant\_id\)s**](http://node163:8774/v2.1/%25\(tenant_id\)s)



**#openstack endpoint create --region RegionOne compute internal http://node163:8774/v2.1/%\(tenant\_id\)s**



**#openstack endpoint create --region RegionOne compute admin** [**http://node163:8774/v2.1/%\(tenant\_id\)s**](http://node163:8774/v2.1/%25\(tenant_id\)s)

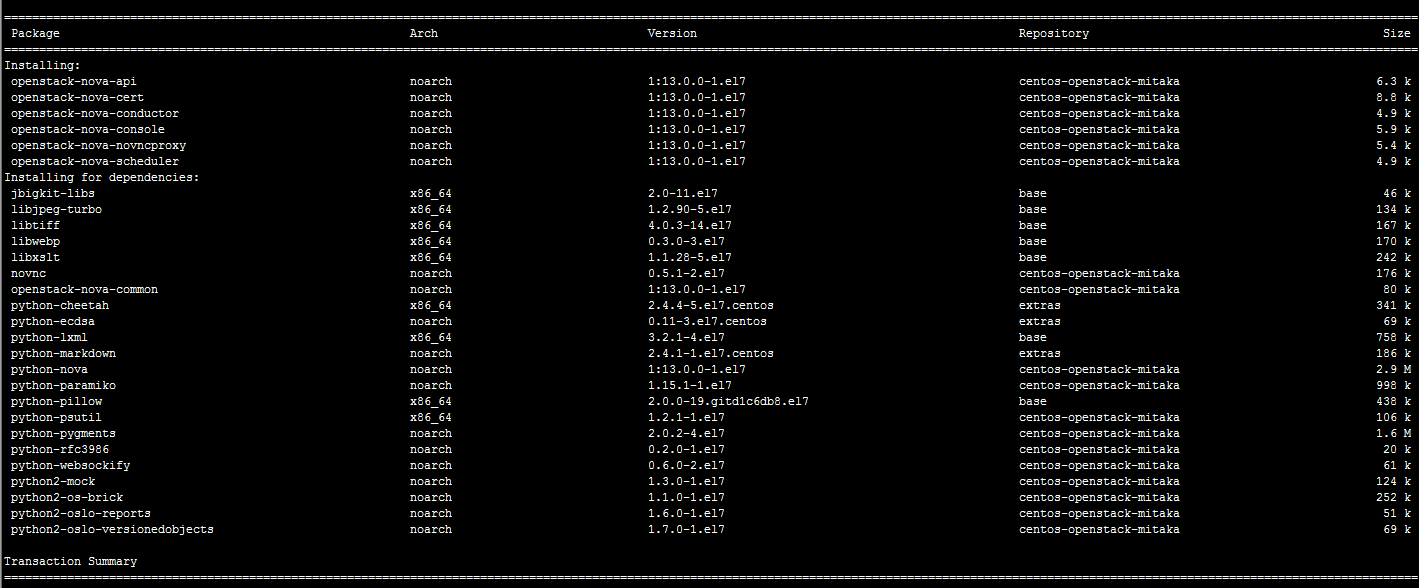


6 在控制节点上yum安装包

**yum install openstack-nova-api openstack-nova-cert \**

**openstack-nova-conductor openstack-nova-console \**

**openstack-nova-novncproxy openstack-nova-scheduler**



7 在控制节点上编辑

#vi /etc/nova/nova.conf

添加如下内容

**enabled\_apis = osapi\_compute,metadata**

****

[api\_database]

**connection = mysql+pymysql://nova:nova@node163/nova\_api**

****

[database]

**connection = mysql+pymysql://nova:nova@node163/nova**



[DEFAULT]

**rpc\_backend = rabbit**

**auth\_strategy = keystone**

**use\_neutron = True**

**firewall\_driver = nova.virt.firewall.NoopFirewallDriver**

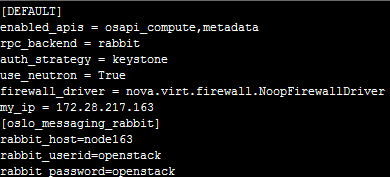
**my\_ip = 172.28.217.163**

**[oslo\_messaging\_rabbit]**

**rabbit\_host=node163**

**rabbit\_userid=openstack**

**rabbit\_password=openstack**



[keystone\_authtoken]

**auth\_uri = http://node163:5000**

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

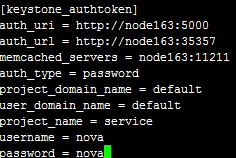
**project\_domain\_name = default**

**user\_domain\_name = default**

**project\_name = service**

**username = nova**

**password = nova**



[vnc]

**vncserver\_listen = $my\_ip**

**vncserver\_proxyclient\_address = $my\_ip**



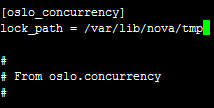
[glance]

**api\_servers = http://node163:9292**



[oslo\_concurrency]

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

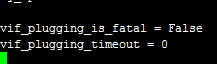


**注：为防止虚拟机在创建过程中出现Failed to allocate the network(s), not rescheduling的情况，需要补充如下配置:**

[DEFAULT]

**vif\_plugging\_is\_fatal = False**

**vif\_plugging\_timeout = 0**



8 在控制节点上填充计算服务的数据库

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

****

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

****

9 在控制节点上启动计算服务并配置开机启动

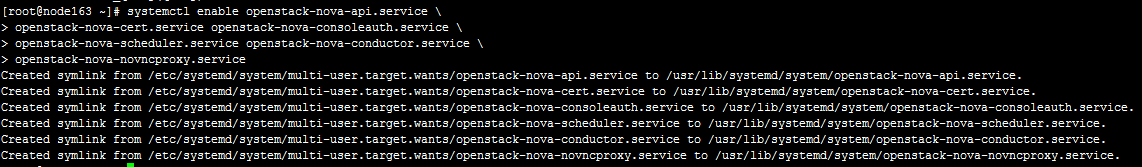
开机启动:

**#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**



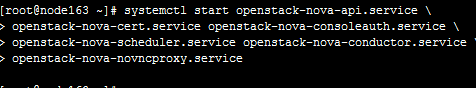
服务启动:

**#systemctl start openstack-nova-api.service \**

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

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

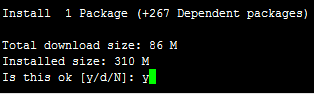
**openstack-nova-novncproxy.service**



## \*计算节点部分:

1 yum安装包

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



2 编辑

#vi /etc/nova/nova.conf

在指定位置添加如下配置

[DEFAULT]

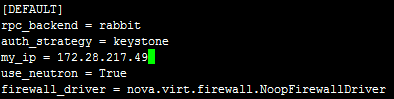
**rpc\_backend = rabbit**

**auth\_strategy = keystone**

**my\_ip = 172.28.217.49**

**use\_neutron = True**

**firewall\_driver = nova.virt.firewall.NoopFirewallDriver**

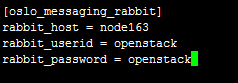
****

[oslo\_messaging\_rabbit]

**rabbit\_host = node163**

**rabbit\_userid = openstack**

**rabbit\_password = openstack**



[keystone\_authtoken]

**auth\_uri = http://node163:5000**

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

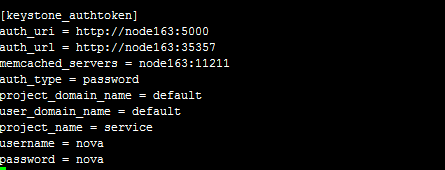
**project\_domain\_name = default**

**user\_domain\_name = default**

**project\_name = service**

**username = nova**

**password = nova**



[vnc]

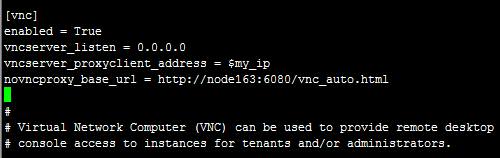
**enabled = True**

**vncserver\_listen = 0.0.0.0**

**vncserver\_proxyclient\_address = $my\_ip**

**#novncproxy\_base\_url =** [**http://node163:6080/vnc\_auto.html**](http://node163:6080/vnc_auto.html)

**novncproxy\_base\_url =** [**http://172.28.217.163:6080/vnc\_auto.html**](http://172.28.217.163:6080/vnc_auto.html)



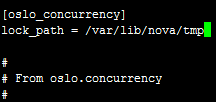
[glance]

**api\_servers =** [**http://node163:9292**](http://node163:9292)



[oslo\_concurrency]

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



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

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

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

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

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

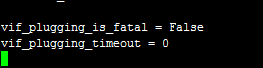
virt\_type = qemu

**注：为防止虚拟机在创建过程中出现Failed to allocate the network(s), not rescheduling的情况，需要补充如下配置:**

[DEFAULT]

**vif\_plugging\_is\_fatal = False**

**vif\_plugging\_timeout = 0**



3启动和注册启动nova服务

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

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



# Networking服务

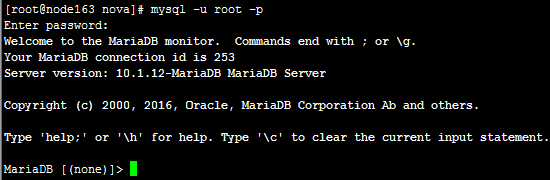
**注意:橙色字体的内容需按实际情况修改**

## 控制节点部分

1 控制节点上进行数据库授权

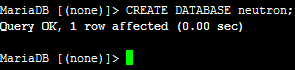
root用户登陆mysql:

# **mysql -u root -p**



创建neutron数据库

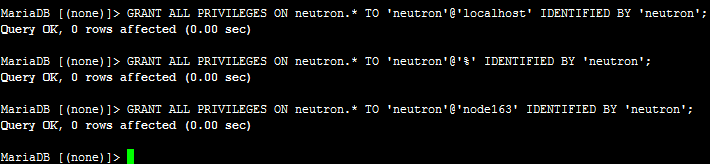
# **CREATE DATABASE neutron;**



分配访问权限

**# GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'localhost' IDENTIFIED BY 'neutron';**

**# GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'%' IDENTIFIED BY 'neutron';**



退出数据库



2 创建neutron用户,密码设置为neutron

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



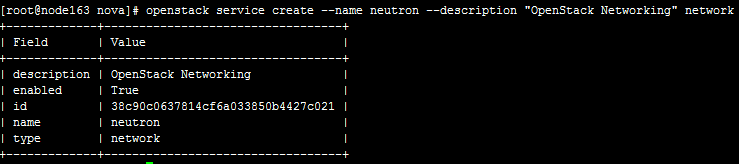
3 添加管理员角色到neutron用户中

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



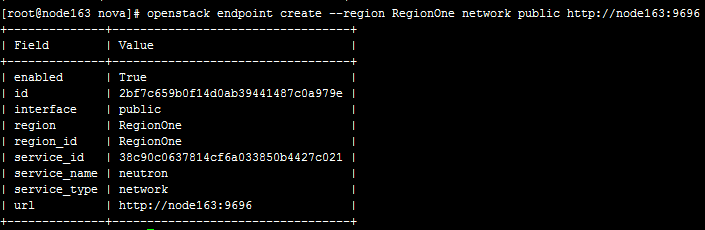
4 创建neutron服务的实体

**# openstack service create --name neutron --description "OpenStack Networking" network**

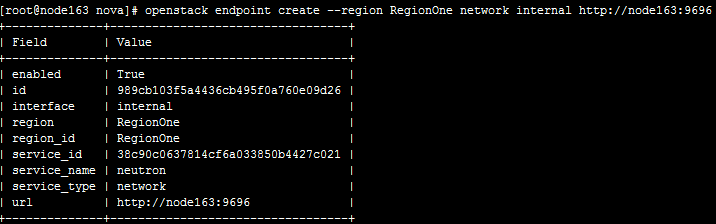


5 创建网络服务API终端

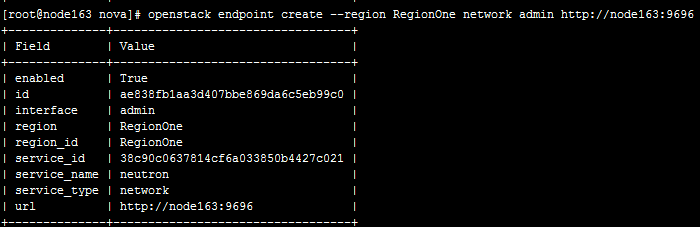
**# openstack endpoint create --region RegionOne network public** [**http://node163:9696**](http://node163:9696)

****

**#openstack endpoint create --region RegionOne network internal** [**http://node163:9696**](http://node163:9696)

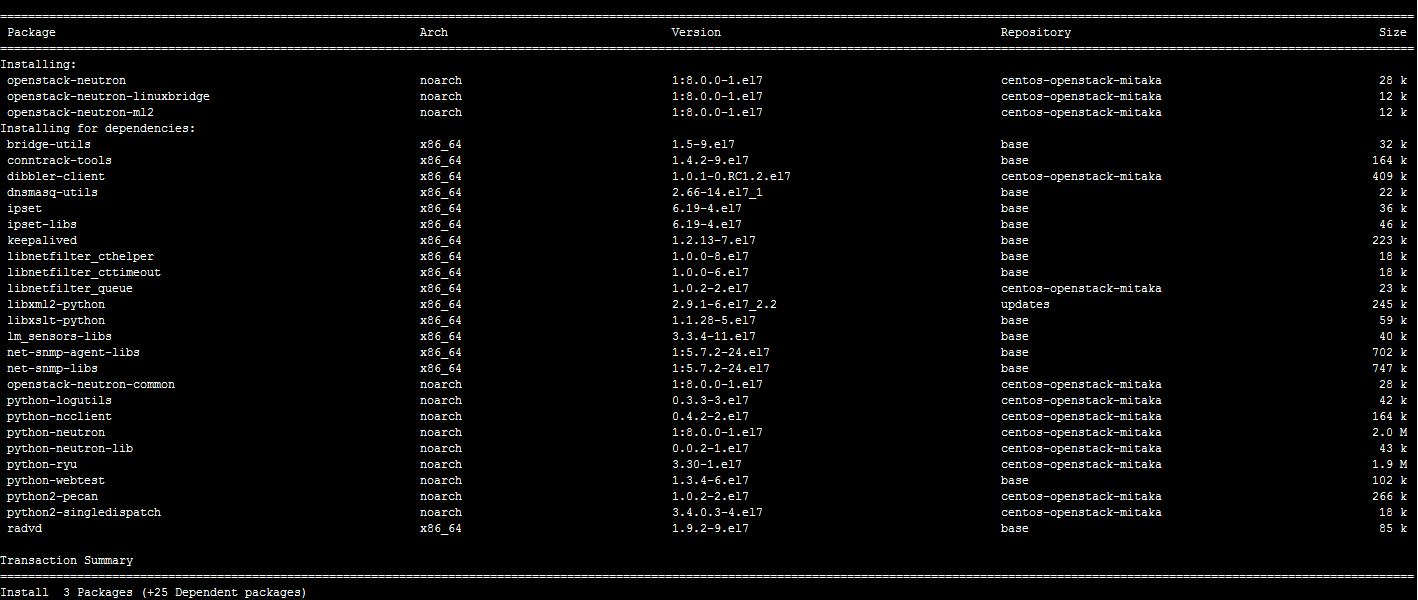
****

**#openstack endpoint create --region RegionOne network admin http://node163:9696**



6 yum安装对应的包

**# yum install openstack-neutron openstack-neutron-ml2 openstack-neutron-linuxbridge ebtables**



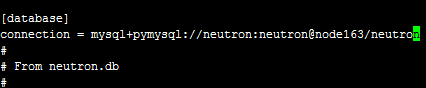
7 修改配置文件

# vi /etc/neutron/neutron.conf

添加如下配置

[database]

**connection = mysql+pymysql://neutron:neutron@node163/neutron**



[DEFAULT]

**core\_plugin = ml2**

**service\_plugins = router**

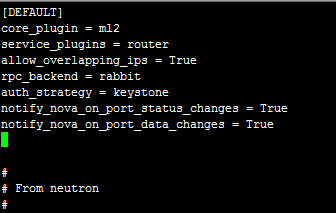
**allow\_overlapping\_ips = True**

**rpc\_backend = rabbit**

**auth\_strategy = keystone**

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

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

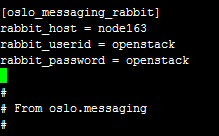
****

[oslo\_messaging\_rabbit]

**rabbit\_host = node163**

**rabbit\_userid = openstack**

**rabbit\_password = openstack**

****

[keystone\_authtoken]

**auth\_uri =** [**http://node163:5000**](http://node163:5000)

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

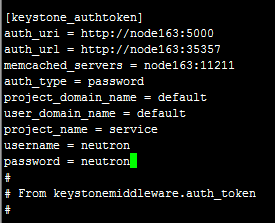
**project\_domain\_name = default**

**user\_domain\_name = default**

**project\_name = service**

**username = neutron**

**password = neutron**

****

[nova]

**auth\_url = http://node163:35357**

**auth\_type = password**

**project\_domain\_name = default**

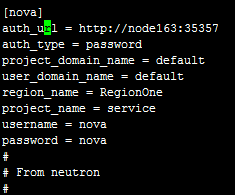
**user\_domain\_name = default**

**region\_name = RegionOne**

**project\_name = service**

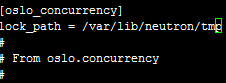
**username = nova**

**password = nova**



[oslo\_concurrency]

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



8 修改配置文件

# vi /etc/neutron/plugins/ml2/ml2\_conf.ini

添加如下内容

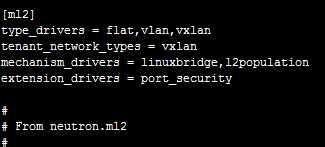
[ml2]

**type\_drivers = flat,vlan,vxlan**

**tenant\_network\_types = vxlan**

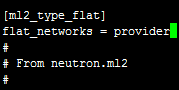
**mechanism\_drivers = linuxbridge,l2population**

**extension\_drivers = port\_security**

****

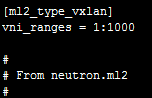
[ml2\_type\_flat]

**flat\_networks = provider**



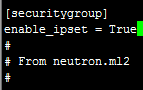
[ml2\_type\_vxlan]

**vni\_ranges = 1:1000**



[securitygroup]

**enable\_ipset = True**

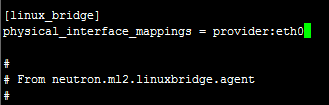


9 配置Linux bridge agent

vi /etc/neutron/plugins/ml2/linuxbridge\_agent.ini

[linux\_bridge]

**physical\_interface\_mappings = provider:eth0**

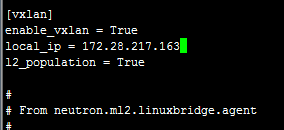


[vxlan]

**enable\_vxlan = True**

**local\_ip = 172.28.217.163**

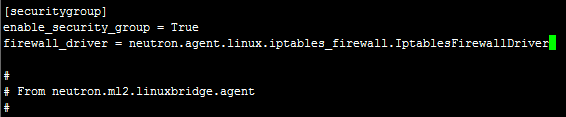
**l2\_population = True**



[securitygroup]

**enable\_security\_group = True**

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



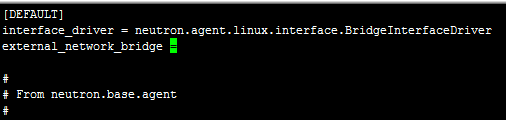
10 配置layer-3 agent

vi /etc/neutron/l3\_agent.ini

[DEFAULT]

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

**external\_network\_bridge =**

****

11 配置DHCP agent

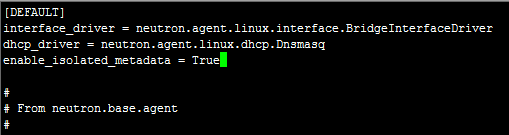
vi /etc/neutron/dhcp\_agent.ini

[DEFAULT]

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

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

**enable\_isolated\_metadata = True**



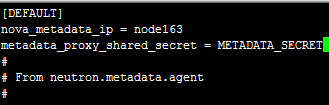
12 配置metadata agent 修改配置文件

# vi /etc/neutron/metadata\_agent.ini

[DEFAULT]

**nova\_metadata\_ip = node163**

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



13 配置nova和neutron结合部分 修改配置文件

#vi /etc/nova/nova.conf

[neutron]

**url = http://node163:9696**

**auth\_url = http://node163:35357**

**auth\_type = password**

**project\_domain\_name = default**

**user\_domain\_name = default**

**region\_name = RegionOne**

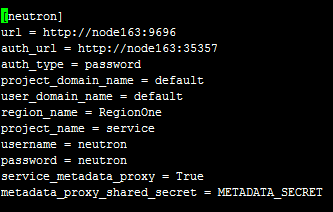
**project\_name = service**

**username = neutron**

**password = neutron**

**service\_metadata\_proxy = True**

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



13 创建软链接

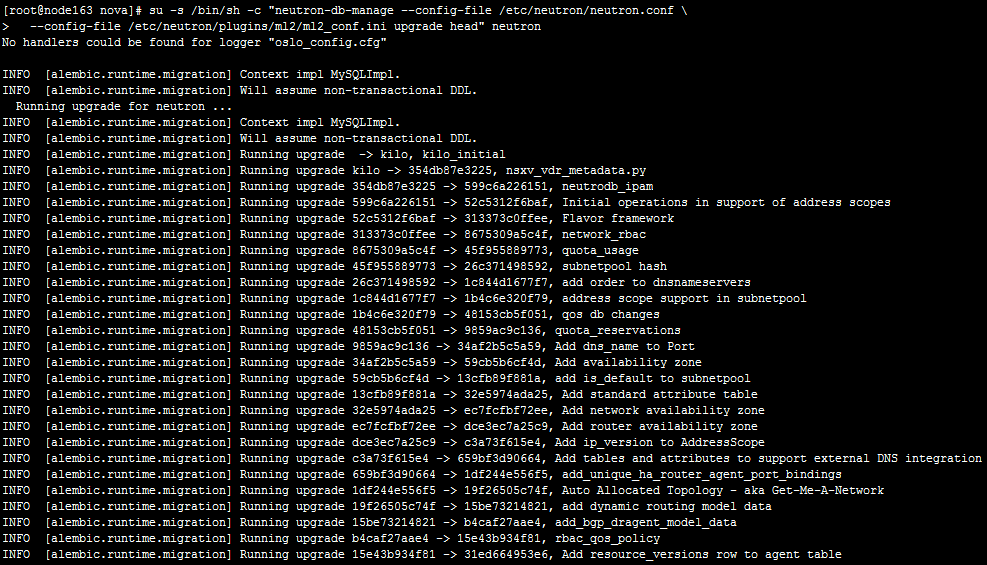
**ln -s /etc/neutron/plugins/ml2/ml2\_conf.ini /etc/neutron/plugin.ini**



14 填充neutron服务的数据库

**#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**



15 重新启动nova的api服务

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

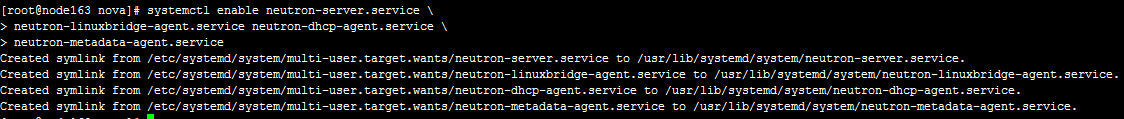


16 配置neutron开机启动并启动neutron服务

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

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

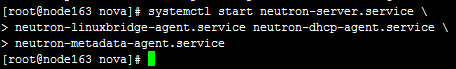
**neutron-metadata-agent.service neutron-l3-agent.service**



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

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

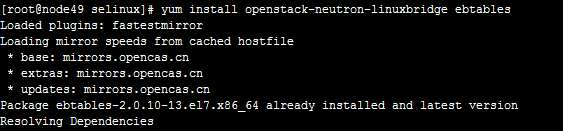
**neutron-metadata-agent.service neutron-l3-agent.service**

****

## 计算节点部分

1 yum安装包

**# yum install openstack-neutron-linuxbridge ebtables ipset**



2 修改配置文件

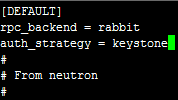
# vi /etc/neutron/neutron.conf

添加如下配置

[DEFAULT]

**rpc\_backend = rabbit**

**auth\_strategy = keystone**

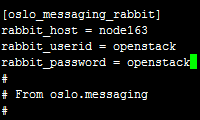


[oslo\_messaging\_rabbit]

**rabbit\_host = node163**

**rabbit\_userid = openstack**

**rabbit\_password = openstack**



[keystone\_authtoken]

**auth\_uri = http://node163:5000**

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

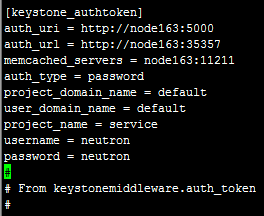
**project\_domain\_name = default**

**user\_domain\_name = default**

**project\_name = service**

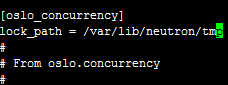
**username = neutron**

**password = neutron**



[oslo\_concurrency]

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

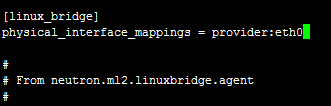


3 配置Linux bridge agent

vi /etc/neutron/plugins/ml2/linuxbridge\_agent.ini

[linux\_bridge]

**physical\_interface\_mappings = provider:eth0**

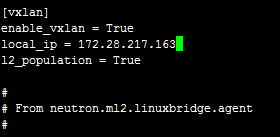


[vxlan]

**enable\_vxlan = True**

**local\_ip = 172.28.217.163**

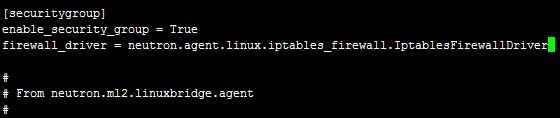
**l2\_population = True**



[securitygroup]

**enable\_security\_group = True**

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



4 修改配置文件 /etc/nova/nova.conf 添加如下配置

[neutron]

**url = http://node163:9696**

**auth\_url = http://node163:35357**

**auth\_type = password**

**project\_domain\_name = default**

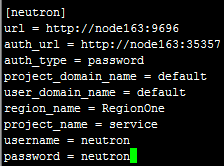
**user\_domain\_name = default**

**region\_name = RegionOne**

**project\_name = service**

**username = neutron**

**password = neutron**



5 重新启动nova计算服务

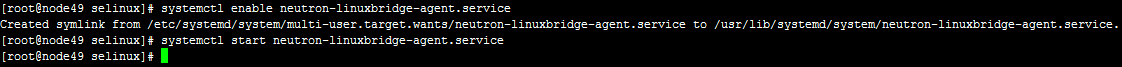
**# systemctl restart openstack-nova-compute.service**



6添加Linux bridge agent为开机启动，并且启动该服务

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

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

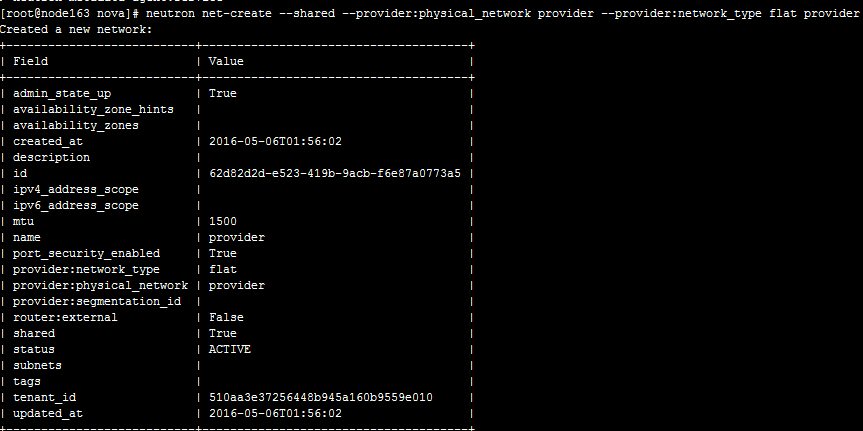
****

**控制节点部分**：

案例中使用的是provider网络，所以下面的网络参数为provider

1 创建虚拟网络

**# neutron net-create --shared --provider:physical\_network provider --provider:network\_type flat provider**

****

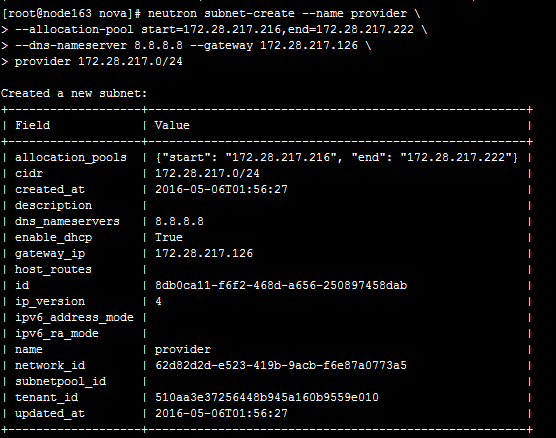
2 创建子网

**# neutron subnet-create --name provider \**

**--allocation-pool start=172.28.217.216,end=172.28.217.222 \**

**--dns-nameserver 8.8.8.8 --gateway 172.28.217.126 \**

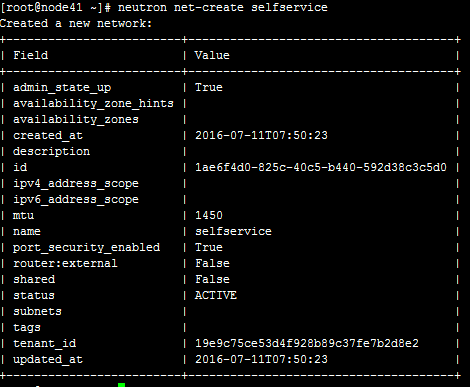
**provider 172.28.217.0/24**



3 创建私有网络

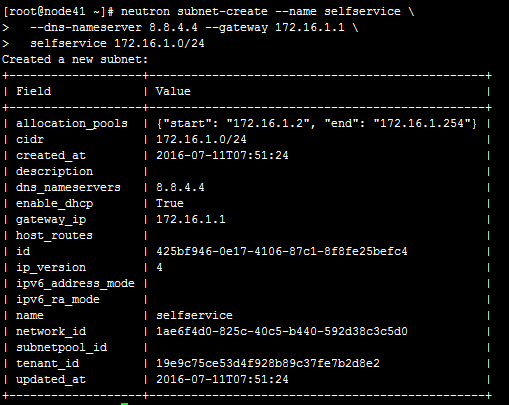
#source demo-openrc

# neutron net-create selfservice



4 创建私有网络子网

# neutron subnet-create --name selfservice --dns-nameserver 8.8.4.4 --gateway 172.16.1.1 selfservice 172.16.1.0/24



5 创建路由

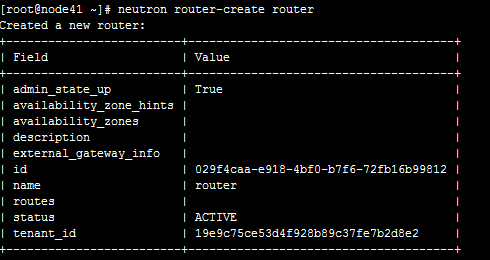
# source admin-openrc

# neutron net-update provider --router:external



# source demo-openrc

# neutron router-create router



# neutron router-interface-add router selfservice



# neutron router-gateway-set router provider



# Networking（DVR）服务

**注意:橙色字体的内容需按实际情况修改**

## 升级内核(所有节点)

**此过程单节点需要三小时左右，请提前安排好时间**

**此过程需要使用root帐户执行，请提前获取root帐户**

1 安装编译内核依赖软件:

**yum install gcc ncurses ncurses-devel bc perl**

2 下载内核

从自建库或者官网下载内核源码包

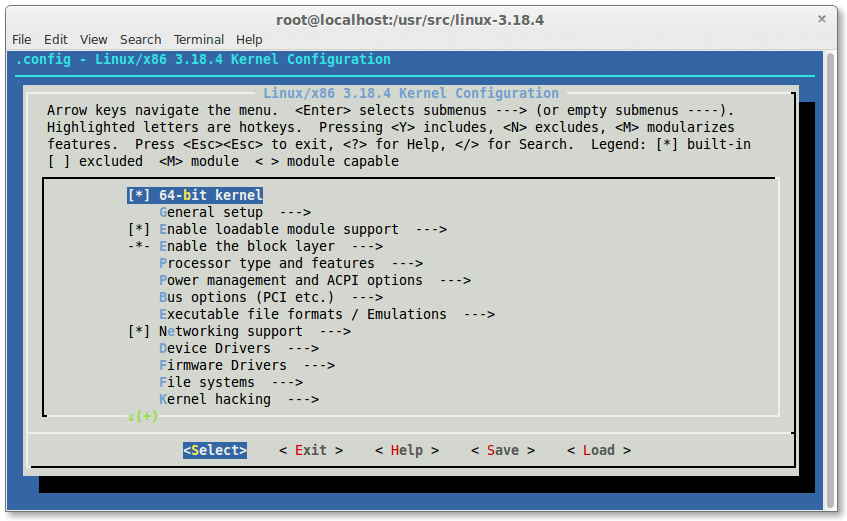
wget <https://www.kernel.org/pub/linux/kernel/v3.x/linux-3.18.4.tar.xz>

3 配置内核

tar -xf linux-3.18.4.tar.xz -C /usr/src/

cd /usr/src/linux-3.18.4/

make menuconfig



在出来此界面后直接保存退出即可。

4 编译内核

make

如下则为成功:

IHEX firmware/mts\_edge.fw

H16TOFW firmware/edgeport/boot.fw

H16TOFW firmware/edgeport/boot2.fw

H16TOFW firmware/edgeport/down.fw

H16TOFW firmware/edgeport/down2.fw

IHEX firmware/edgeport/down3.bin

IHEX2FW firmware/whiteheat\_loader.fw

IHEX2FW firmware/whiteheat.fw

IHEX2FW firmware/keyspan\_pda/keyspan\_pda.fw

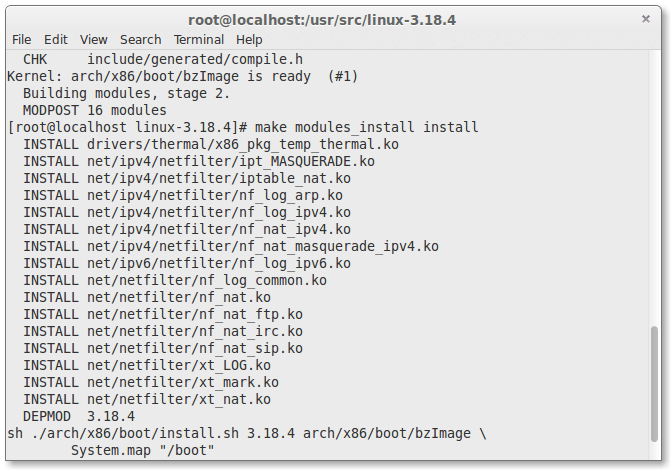
IHEX2FW firmware/keyspan\_pda/xircom\_pgs.fw

You have new mail in /var/spool/mail/root

5 安装内核

make modules\_install install

如下:



6 更改新内核为开机启动默认内核

cat /boot/grub2/grub.cfg |grep 3.18

grub2-set-default 'CentOS Linux (3.18.4) 7 (Core)'

备注:标黄参数由第一条指令返回结果中的entry得到

[root@node65 ~]# cat /boot/grub2/grub.cfg |grep 3.18

menuentry 'CentOS Linux (3.18.4) 7 (Core)' --class centos --class gnu-linux --class gnu --class os --unrestricted $menuentry\_id\_option 'gnulinux-3.10.0-327.el7.x86\_64-advanced-13a1ad74-6624-4bc4-afce-5f7aad9a507f' {

linux16 /vmlinuz-3.18.4 root=/dev/mapper/centos-root ro crashkernel=auto rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet LANG=en\_US.UTF-8

initrd16 /initramfs-3.18.4.img

6 重起机器并察看内核

reboot 0

uname -r

如下:



## 编译安装openvswitch

1 安装编译依赖软件

yum -y install make gcc openssl-devel autoconf automake rpm-build redhat-rpm-config python-devel openssl-devel kernel-devel kernel-debug-devel libtool wget python-twisted-core python-zope-interface PyQt4 desktop-file-utils groff graphviz selinux-policy-devel libcap-ng-devel

2 编译准备

mkdir -p ~/rpmbuild/SOURCES

wget <http://openvswitch.org/releases/openvswitch-2.5.0.tar.gz>

cp openvswitch-2.5.0.tar.gz ~/rpmbuild/SOURCES/  
tar xfz openvswitch-2.5.0.tar.gz

3 编译openvswitch

#rpmbuild -bb --nocheck --nocheck -D "kversion 3.18.4" ~/openvswitch-2.5.0/rhel/openvswitch-kmod-fedora.spec

#rpmbuild -bb --nocheck ~/openvswitch-2.5.0/rhel/openvswitch-fedora.spec

4 安装

yum install ~/rpmbuild/RPMS/x86\_64/openvswitch-kmod-2.5.0-1.el7.centos.x86\_64.rpm

yum install ~/rpmbuild/RPMS/noarch/python-openvswitch-2.5.0-1.el7.centos.noarch.rpm

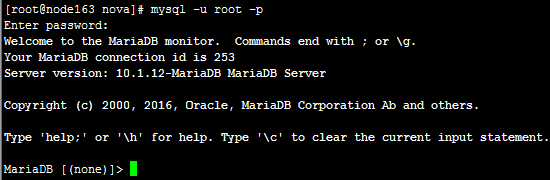
yum install ~/rpmbuild/RPMS/x86\_64/openvswitch-2.5.0-1.el7.centos.x86\_64.rpm

## 控制节点部分

1 控制节点上进行数据库授权

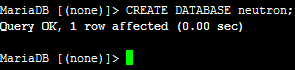
root用户登陆mysql:

# **mysql -u root -p**



创建neutron数据库

# **CREATE DATABASE neutron;**

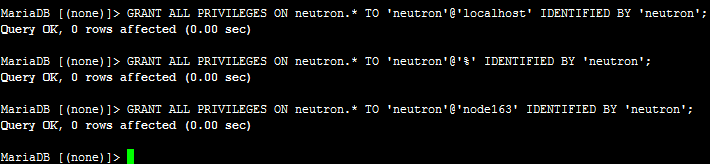


分配访问权限

**# GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'localhost' IDENTIFIED BY 'neutron';**

**# GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'%' IDENTIFIED BY 'neutron';**

**# GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'node163' IDENTIFIED BY 'neutron';**



退出数据库



2 创建neutron用户,密码设置为neutron

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



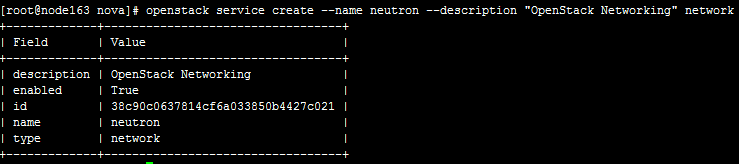
3 添加管理员角色到neutron用户中

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



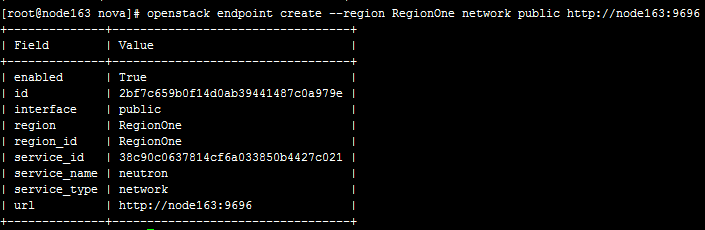
4 创建neutron服务的实体

**# openstack service create --name neutron --description "OpenStack Networking" network**

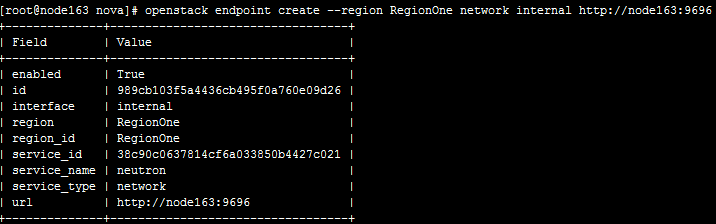


5 创建网络服务API终端

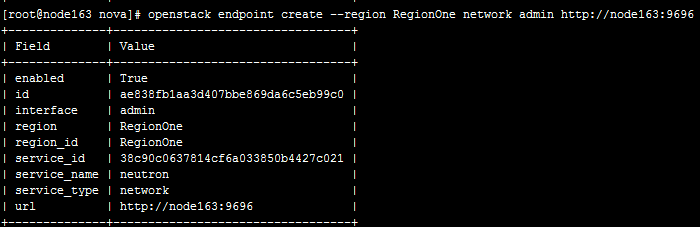
**# openstack endpoint create --region RegionOne network public** [**http://node163:9696**](http://node163:9696)

****

**#openstack endpoint create --region RegionOne network internal** [**http://node163:9696**](http://node163:9696)

****

**#openstack endpoint create --region RegionOne network admin http://node163:9696**



## 控制节点网络部分+网络节点（位于同一主机）

6 yum安装对应的包

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

7 修改配置文件

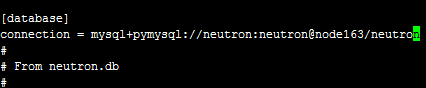
# vi /etc/neutron/neutron.conf

添加如下配置

[database]

**connection = mysql+pymysql://neutron:neutron@node163/neutron**

如下:



[DEFAULT]

**core\_plugin = ml2**

**service\_plugins = router**

**allow\_overlapping\_ips = True**

**rpc\_backend = rabbit**

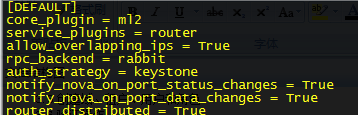
**auth\_strategy = keystone**

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

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

**router\_distributed = True**

如下:

****

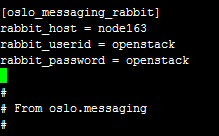
[oslo\_messaging\_rabbit]

**rabbit\_host = node163**

**rabbit\_userid = openstack**

**rabbit\_password = openstack**

如下:

****

[keystone\_authtoken]

**auth\_uri =** [**http://node163:5000**](http://node163:5000)

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

**project\_domain\_name = default**

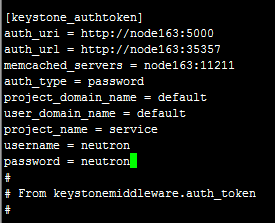
**user\_domain\_name = default**

**project\_name = service**

**username = neutron**

**password = neutron**

如下：

****

[nova]

**auth\_url = http://node163:35357**

**auth\_type = password**

**project\_domain\_name = default**

**user\_domain\_name = default**

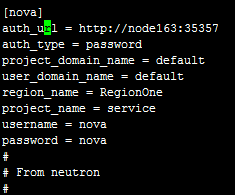
**region\_name = RegionOne**

**project\_name = service**

**username = nova**

**password = nova**

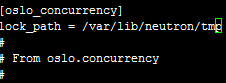
如下：



[oslo\_concurrency]

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

如下：



8 修改配置文件

# vi /etc/neutron/plugins/ml2/ml2\_conf.ini

添加如下内容

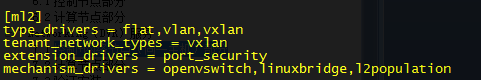
[ml2]

type\_drivers = flat,vlan,vxlan

tenant\_network\_types = vxlan

extension\_drivers = port\_security  
mechanism\_drivers = openvswitch,linuxbridge,l2population

如下:



[ml2\_type\_flat]

flat\_networks = external

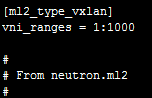
如下:



[ml2\_type\_vxlan]

**vni\_ranges = 1:1000**

如下:



[securitygroup]

firewall\_driver = iptables\_hybrid

如下:

****

9 配置openvswitch\_agent

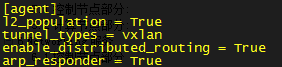
vi /etc/neutron/plugins/ml2/openvswitch\_agent.ini

[agent]

l2\_population = True  
tunnel\_types = vxlan  
enable\_distributed\_routing = True

arp\_responder = True

 如下:



[ovs]

local\_ip = TUNNELS\_IP  
bridge\_mappings = external:br-ex

如下:



[securitygroup]

firewall\_driver = iptables\_hybrid

如下:



10 配置layer-3 agent

vi /etc/neutron/l3\_agent.ini

[DEFAULT]

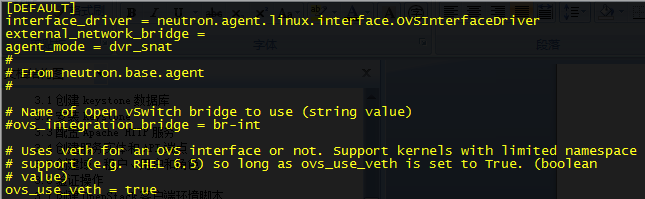
interface\_driver = neutron.agent.linux.interface.OVSInterfaceDriver

external\_network\_bridge =

agent\_mode = dvr\_snat

ovs\_use\_veth = true

如下:



11 配置DHCP agent

vi /etc/neutron/dhcp\_agent.ini

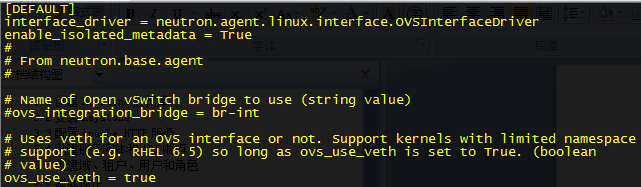
[DEFAULT]

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

enable\_isolated\_metadata = True

ovs\_use\_veth = true

如下:



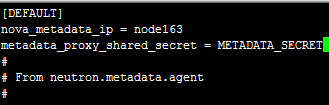
12 配置metadata agent 修改配置文件

# vi /etc/neutron/metadata\_agent.ini

[DEFAULT]

**nova\_metadata\_ip = node163**

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



13 配置nova和neutron结合部分 修改配置文件

#vi /etc/nova/nova.conf

[neutron]

**url = http://node163:9696**

**auth\_url = http://node163:35357**

**auth\_type = password**

**project\_domain\_name = default**

**user\_domain\_name = default**

**region\_name = RegionOne**

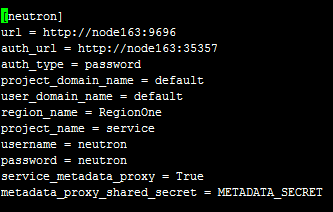
**project\_name = service**

**username = neutron**

**password = neutron**

**service\_metadata\_proxy = True**

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



13 创建软链接

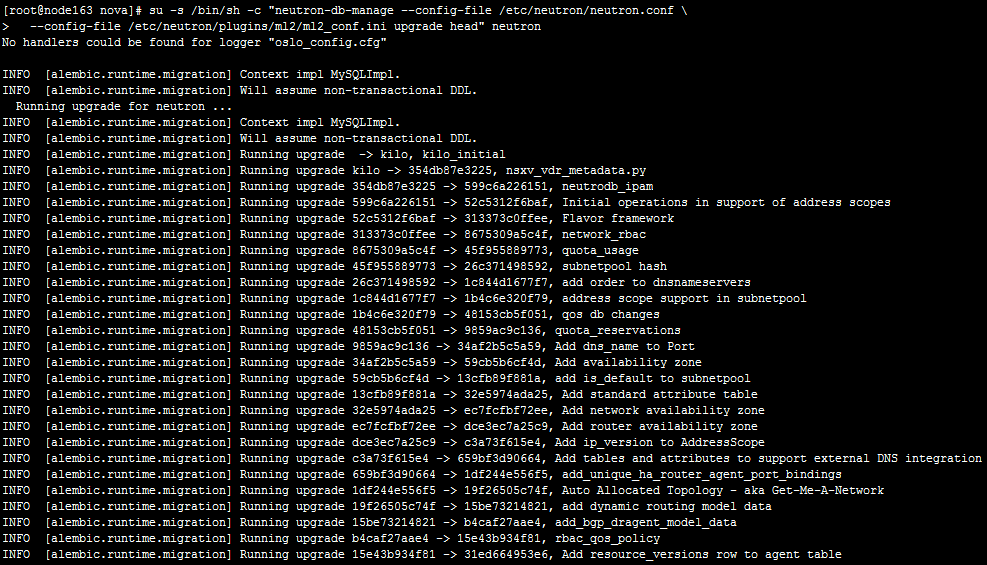
**ln -s /etc/neutron/plugins/ml2/ml2\_conf.ini /etc/neutron/plugin.ini**



14 填充neutron服务的数据库

**#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**



14 建立openvswitch网桥br-ex

The OVS service provides the underlying virtual networking framework for instances. The  
integration bridge br-int handles internal instance network traffic within OVS. The external bridge br-ex handles external instance network traffic within OVS. The external bridge  
requires a port on the physical external network interface to provide instances with external network access. In essence, this port connects the virtual and physical external networks  
in your environment.  
1. Start the OVS service and configure it to start when the system boots:  
# systemctl enable openvswitch.service  
# systemctl start openvswitch.service

**2.** Create an external network bridge (br-ex) and add an associated port (eth1) to it:

Create the external bridge in */etc/sysconfig/network-scripts/ifcfg-br-ex*:

DEVICE=br-ex  
TYPE=OVSBridge  
DEVICETYPE=ovs  
ONBOOT=yes  
NM\_CONTROLLED=no  
BOOTPROTO=none

In */etc/sysconfig/network-scripts/ifcfg-eth1*, configure eth1 to connect to br-ex:

DEVICE=eth1  
TYPE=OVSPort  
DEVICETYPE=ovs  
OVS\_BRIDGE=br-ex  
ONBOOT=yes  
NM\_CONTROLLED=no  
BOOTPROTO=none

Reboot the node or restart the network service for the changes to take effect.

Note  
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**

15 重新启动nova的api服务

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



16 配置neutron开机启动并启动neutron服务

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

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

**neutron-metadata-agent.service neutron-l3-agent.service \**

**neutron-openvswitch-agent.service**

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

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

**neutron-metadata-agent.service neutron-l3-agent.service \**

**neutron-openvswitch-agent.service**

17 启动br-tun,br-int

**yum install net-tools**

**ifconfig br-tun up**

**ifconfig br-int up**

**创建定时启动脚本:**

**vi /etc/startbr.sh:**

**#!/bin/bash**

**for iter in {1..12};**

**do**

**brtun=$(/usr/sbin/ifconfig br-tun|grep UP|awk '{print $NF}');**

**brint=$(/usr/sbin/ifconfig br-int|grep UP|awk '{print $NF}');**

**brtunexisted=$(/usr/sbin/ifconfig br-tun|awk '{print $NF}'|head -1);**

**brintexisted=$(/usr/sbin/ifconfig br-int|awk '{print $NF}'|head -1);**

**if [ -z $brint ] && [ ! -z $brintexisted ];then /usr/sbin/ifconfig br-int up; fi;**

**if [ -z $brtun ] && [ ! -z $brtunexisted ];then /usr/sbin/ifconfig br-tun up; fi;**

**sleep 5s**

**done**

加执行权限:

chmod +x /etc/startbr.sh

**添加到定时任务:**

1.配置crontab，每五分钟运行一次时间同步

**echo "\*/1 \* \* \* \* /bin/bash /etc/startbr.sh"|crontab**

crontab -l 可以查看添加好的内容

2.设置crontab开机启动

chkconfig --level 35 crond on

## 计算节点部分

1 yum安装包

**yum install openstack-neutron-linuxbridge ebtables ipset**

**yum install openstack-neutron openstack-neutron-ml2**

**yum install openstack-neutron-openvswitch**

2 修改配置文件

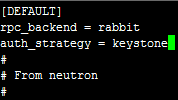
# vi /etc/neutron/neutron.conf

添加如下配置

[DEFAULT]

**rpc\_backend = rabbit**

**auth\_strategy = keystone**

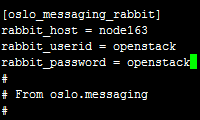


[oslo\_messaging\_rabbit]

**rabbit\_host = node163**

**rabbit\_userid = openstack**

**rabbit\_password = openstack**



[keystone\_authtoken]

**auth\_uri = http://node163:5000**

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

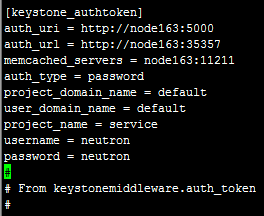
**project\_domain\_name = default**

**user\_domain\_name = default**

**project\_name = service**

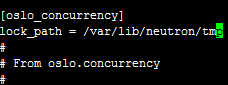
**username = neutron**

**password = neutron**



[oslo\_concurrency]

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



3 配置openvswitch\_agent

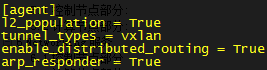
#vi /etc/neutron/plugins/ml2/openvswitch\_agent.ini

[agent]

l2\_population = True  
tunnel\_types = vxlan  
enable\_distributed\_routing = True

arp\_responder = True

如下:



[ovs]

local\_ip = TUNNELS\_IP  
bridge\_mappings = external:br-ex

如下:



[securitygroup]

firewall\_driver = iptables\_hybrid

如下:



4 配置l3\_agent

vi /etc/neutron/l3\_agent.ini:

[DEFAULT]

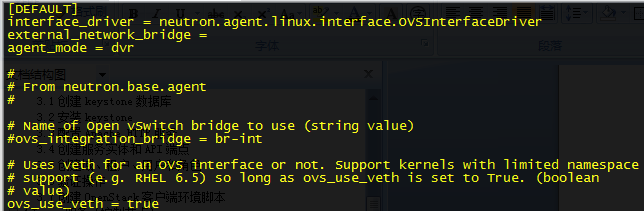
interface\_driver = neutron.agent.linux.interface.OVSInterfaceDriver

external\_network\_bridge =

agent\_mode = dvr

ovs\_use\_veth = true

如下:



5 配置metadata\_agent

vi /etc/neutron/metadata\_agent.ini:

[DEFAULT]

nova\_metadata\_ip = controller

metadata\_proxy\_shared\_secret = METADATA\_SECRET

如下:



6 修改配置文件 /etc/nova/nova.conf 添加如下配置

[neutron]

**url = http://node163:9696**

**auth\_url = http://node163:35357**

**auth\_type = password**

**project\_domain\_name = default**

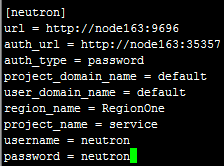
**user\_domain\_name = default**

**region\_name = RegionOne**

**project\_name = service**

**username = neutron**

**password = neutron**



7建立openvswitch网桥br-ex

The OVS service provides the underlying virtual networking framework for instances. The  
integration bridge br-int handles internal instance network traffic within OVS. The external bridge br-ex handles external instance network traffic within OVS. The external bridge  
requires a port on the physical external network interface to provide instances with external network access. In essence, this port connects the virtual and physical external networks  
in your environment.  
1. Start the OVS service and configure it to start when the system boots:  
# systemctl enable openvswitch.service  
# systemctl start openvswitch.service

**2.** Create an external network bridge (br-ex) and add an associated port (eth1) to it:

Create the external bridge in */etc/sysconfig/network-scripts/ifcfg-br-ex*:

DEVICE=br-ex  
TYPE=OVSBridge  
DEVICETYPE=ovs  
ONBOOT=yes  
NM\_CONTROLLED=no  
BOOTPROTO=none

In */etc/sysconfig/network-scripts/ifcfg-eth1*, configure eth1 to connect to br-ex:

DEVICE=eth1  
TYPE=OVSPort  
DEVICETYPE=ovs  
OVS\_BRIDGE=br-ex  
ONBOOT=yes  
NM\_CONTROLLED=no  
BOOTPROTO=none

Reboot the node or restart the network service for the changes to take effect.

Note  
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**

8 重新启动nova计算服务

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



9启动neutron相关服务并设置开机启动

**chkconfig neutron-openvswitch-agent on**

**chkconfig neutron-metadata-agent on**

**chkconfig neutron-l3-agent on**

**service neutron-openvswitch-agent start**

**service neutron-metadata-agent start**

**service neutron-l3-agent start**

10 启动br-tun,br-int

**yum install net-tools**

**ifconfig br-tun up**

**ifconfig br-int up**

**创建定时启动脚本:**

**vi /etc/startbr.sh:**

**#!/bin/bash**

**for iter in {1..12};**

**do**

**brtun=$(/usr/sbin/ifconfig br-tun|grep UP|awk '{print $NF}');**

**brint=$(/usr/sbin/ifconfig br-int|grep UP|awk '{print $NF}');**

**brtunexisted=$(/usr/sbin/ifconfig br-tun|awk '{print $NF}'|head -1);**

**brintexisted=$(/usr/sbin/ifconfig br-int|awk '{print $NF}'|head -1);**

**if [ -z $brint ] && [ ! -z $brintexisted ];then /usr/sbin/ifconfig br-int up; fi;**

**if [ -z $brtun ] && [ ! -z $brtunexisted ];then /usr/sbin/ifconfig br-tun up; fi;**

**sleep 5s**

**done**

加执行权限:

chmod +x /etc/startbr.sh

**添加到定时任务:**

1.配置crontab，每五分钟运行一次时间同步

**echo "\*/1 \* \* \* \* /bin/bash /etc/startbr.sh"|crontab**

crontab -l 可以查看添加好的内容

2.设置crontab开机启动

chkconfig --level 35 crond on

## 验证安装

### neutron节点察看服务

网络+控制节点应该有Metadata agent，Open vSwitch agent，DHCP agent，L3 agent

计算节点应该有Metadata agent，Open vSwitch agent， L3 agent

$ neutron agent-list

+--------------------------------------+--------------------+----------+-------+----------------+---------------------------+

| id | agent\_type | host | alive | admin\_state\_up | binary |

+--------------------------------------+--------------------+----------+-------+----------------+---------------------------+

| 10b084e5-4ab8-43d6-9b04-6d56f27f9cd4 | Metadata agent | network1 | :-) | True | neutron-metadata-agent |

| 2f90ef81-3eed-4ecf-b6b9-2d2c21dda85c | Open vSwitch agent | compute2 | :-) | True | neutron-openvswitch-agent |

| 319563ac-88f9-4352-b63e-e55beb673372 | DHCP agent | network1 | :-) | True | neutron-dhcp-agent |

| 3345723e-16e8-4b74-9d15-d7f1f977a3bd | Open vSwitch agent | compute1 | :-) | True | neutron-openvswitch-agent |

| 4643c811-a54a-41da-91a8-c2328bcaeea3 | Open vSwitch agent | network1 | :-) | True | neutron-openvswitch-agent |

| 5ad81671-efc3-4acc-9d5d-030a1c4f6a25 | L3 agent | compute1 | :-) | True | neutron-l3-agent |

| 641337fa-99c2-468d-8d7e-89277d6ba144 | Metadata agent | compute1 | :-) | True | neutron-metadata-agent |

| 9372e008-bd29-4436-8e01-8ddfd50d2b74 | L3 agent | network1 | :-) | True | neutron-l3-agent |

| af9d1169-1012-4440-9de2-778c8fce21b9 | L3 agent | compute2 | :-) | True | neutron-l3-agent |

| ee59e3ba-ee3c-4621-b3d5-c9d8123b6cc5 | Metadata agent | compute2 | :-) | True | neutron-metadata-agent |

+--------------------------------------+--------------------+----------+-------+----------------+---------------------------+

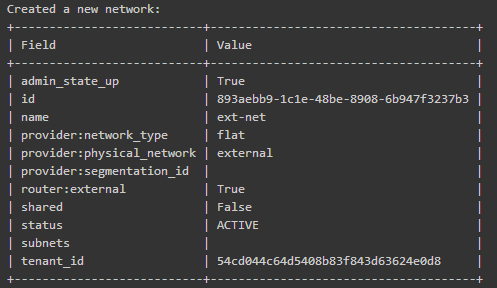
### 创建初始网络

This example creates a flat external network and a VXLAN project network.

1. Source the administrative project credentials.
2. Create the external network:

neutron net-create ext-net --router:external \

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

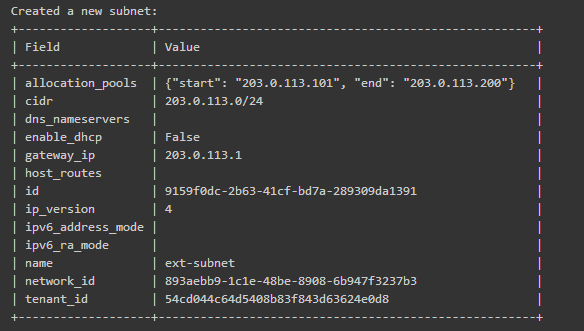


1. Create a subnet on the external network:

neutron subnet-create ext-net 203.0.113.0/24 --allocation-pool \

start=203.0.113.101,end=203.0.113.200 --disable-dhcp \

--gateway 203.0.113.1

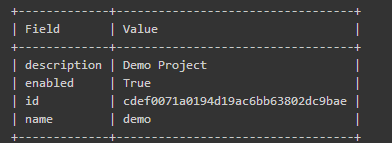


Note

The example configuration contains vlan as the first project network type. Only a privileged user can create other types of networks such as GRE or VXLAN. The following commands use the admin project credentials to create a VXLAN project network.

1. Obtain the ID of a regular project. For example, using the demo project:

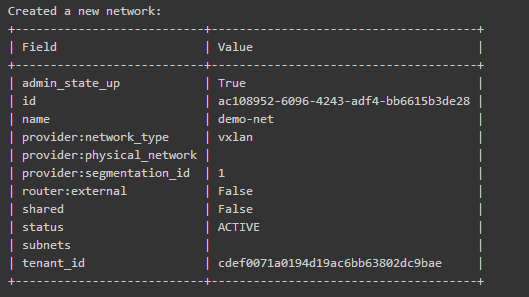
$ openstack project show demo



1. Create the project network:

$ neutron net-create demo-net --tenant-id cdef0071a0194d19ac6bb63802dc9bae \

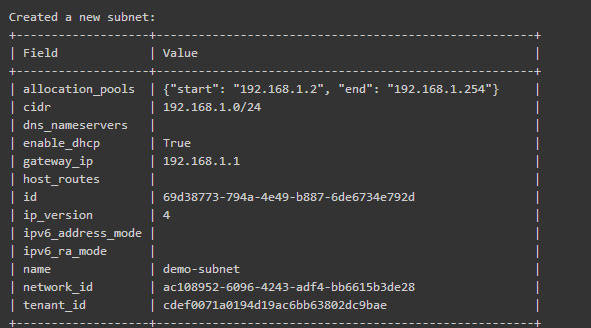
--provider:network\_type vxlan



1. Source the regular project credentials.
2. Create a subnet on the project network:

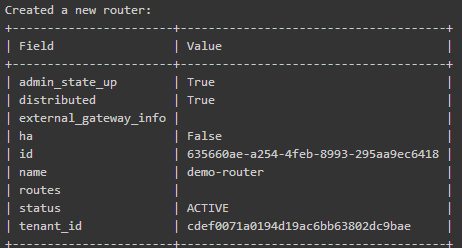
$ neutron subnet-create demo-net --name demo-subnet --gateway 192.168.1.1 \

192.168.1.0/24



1. Create a distributed project router:

$ neutron router-create demo-router



Note

Default policy might prevent the ‘distributed` flag from appearing in the command output for non-privileged users.

1. Attach the project network to the router:

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

Added interface b1a894fd-aee8-475c-9262-4342afdc1b58 to router demo-router.

1. Add a gateway to the external network for the project network on the router:

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

Set gateway for router demo-router

### 验证网络操作

1. On the network node, verify creation of the snat, qrouter, and qdhcp namespaces:

$ ip netns

snat-4d7928a0-4a3c-4b99-b01b-97da2f97e279

qrouter-4d7928a0-4a3c-4b99-b01b-97da2f97e279

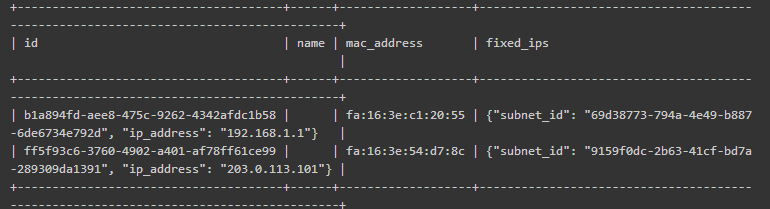
qdhcp-353f5937-a2d3-41ba-8225-fa1af2538141

Note

One or more namespaces might not exist until launching an instance.

1. Source the administrative project credentials.
2. Determine the external network gateway IP address for the project network on the router, typically the lowest IP address in the external subnet IP allocation range:

$ neutron router-port-list demo-router



1. On the controller node or any host with access to the external network, ping the external network gateway IP address on the project router:

$ ping -c 4 203.0.113.101

PING 203.0.113.101 (203.0.113.101) 56(84) bytes of data.

64 bytes from 203.0.113.101: icmp\_req=1 ttl=64 time=0.619 ms

64 bytes from 203.0.113.101: icmp\_req=2 ttl=64 time=0.189 ms

64 bytes from 203.0.113.101: icmp\_req=3 ttl=64 time=0.165 ms

64 bytes from 203.0.113.101: icmp\_req=4 ttl=64 time=0.216 ms

--- 203.0.113.101 ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 2999ms

rtt min/avg/max/mdev = 0.165/0.297/0.619/0.187 ms

1. Source the regular project credentials.
2. Launch an instance with an interface on the project network.
3. On the compute node with the instance, verify creation of the qrouter namespace:

$ ip netns

qrouter-4d7928a0-4a3c-4b99-b01b-97da2f97e279

1. Obtain console access to the instance.
   1. Test connectivity to the project router:

$ ping -c 4 192.168.1.1

PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.

64 bytes from 192.168.1.1: icmp\_req=1 ttl=64 time=0.357 ms

64 bytes from 192.168.1.1: icmp\_req=2 ttl=64 time=0.473 ms

64 bytes from 192.168.1.1: icmp\_req=3 ttl=64 time=0.504 ms

64 bytes from 192.168.1.1: icmp\_req=4 ttl=64 time=0.470 ms

--- 192.168.1.1 ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 2998ms

rtt min/avg/max/mdev = 0.357/0.451/0.504/0.055 ms

* 1. Test connectivity to the Internet:

$ ping -c 4 openstack.org

PING openstack.org (174.143.194.225) 56(84) bytes of data.

64 bytes from 174.143.194.225: icmp\_req=1 ttl=53 time=17.4 ms

64 bytes from 174.143.194.225: icmp\_req=2 ttl=53 time=17.5 ms

64 bytes from 174.143.194.225: icmp\_req=3 ttl=53 time=17.7 ms

64 bytes from 174.143.194.225: icmp\_req=4 ttl=53 time=17.5 ms

--- openstack.org ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 3003ms

rtt min/avg/max/mdev = 17.431/17.575/17.734/0.143 ms

1. Create the appropriate security group rules to allow ping and SSH access to the instance. For example:

$ nova secgroup-add-rule default icmp -1 -1 0.0.0.0/0

+-------------+-----------+---------+-----------+--------------+

| IP Protocol | From Port | To Port | IP Range | Source Group |

+-------------+-----------+---------+-----------+--------------+

| icmp | -1 | -1 | 0.0.0.0/0 | |

+-------------+-----------+---------+-----------+--------------+

$ nova secgroup-add-rule default tcp 22 22 0.0.0.0/0

+-------------+-----------+---------+-----------+--------------+

| IP Protocol | From Port | To Port | IP Range | Source Group |

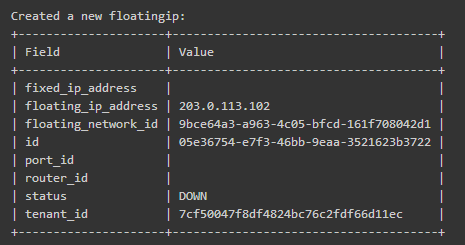
+-------------+-----------+---------+-----------+--------------+

| tcp | 22 | 22 | 0.0.0.0/0 | |

+-------------+-----------+---------+-----------+--------------+

1. Create a floating IP address on the external network:

$ neutron floatingip-create ext-net



1. Associate the floating IP address with the instance:

$ nova floating-ip-associate demo-instance1 203.0.113.102

1. Verify addition of the floating IP address to the instance:

$ nova list

+--------------------------------------+----------------+--------+------------+-------------+-----------------------------------------+

| ID | Name | Status | Task State | Power State | Networks |

+--------------------------------------+----------------+--------+------------+-------------+-----------------------------------------+

| 05682b91-81a1-464c-8f40-8b3da7ee92c5 | demo-instance1 | ACTIVE | - | Running | demo-net=192.168.1.3, 203.0.113.102 |

+--------------------------------------+----------------+--------+------------+-------------+-----------------------------------------+

1. On the compute node with the instance, verify creation of the fip namespace:

$ ip netns

fip-2c7bd9c2-8ab0-46ef-b7c1-023ce0452c24

1. On the controller node or any host with access to the external network, ping the floating IP address associated with the instance:

$ ping -c 4 203.0.113.102

PING 203.0.113.102 (203.0.113.112) 56(84) bytes of data.

64 bytes from 203.0.113.102: icmp\_req=1 ttl=63 time=3.18 ms

64 bytes from 203.0.113.102: icmp\_req=2 ttl=63 time=0.981 ms

64 bytes from 203.0.113.102: icmp\_req=3 ttl=63 time=1.06 ms

64 bytes from 203.0.113.102: icmp\_req=4 ttl=63 time=0.929 ms

--- 203.0.113.102 ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 3002ms

rtt min/avg/max/mdev = 0.929/1.539/3.183/0.951 ms

# Networking（DVR+VRRP+DHCPHA）服务

**注意:橙色字体的内容需按实际情况修改**

## 升级内核(所有节点)

**此过程单节点需要三小时左右，请提前安排好时间**

**此过程需要使用root帐户执行，请提前获取root帐户**

1 安装编译内核依赖软件:

**yum install gcc ncurses ncurses-devel bc perl**

2 下载内核

从自建库或者官网下载内核源码包

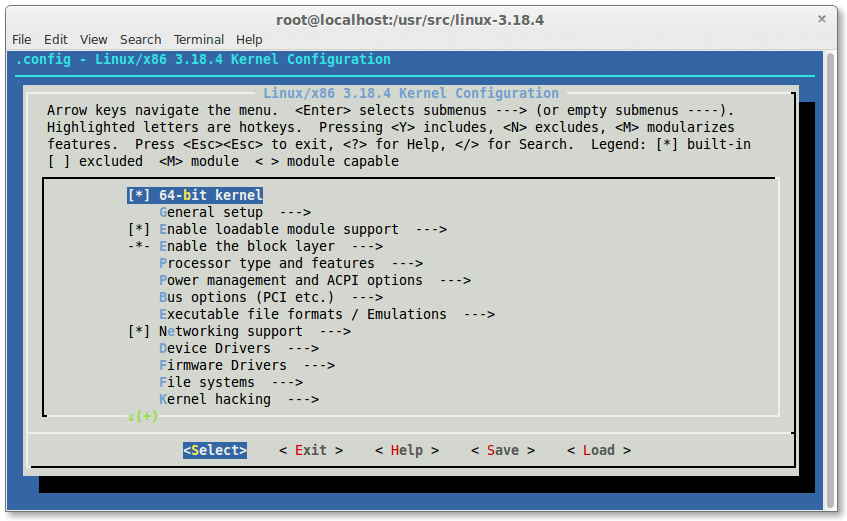
wget <https://www.kernel.org/pub/linux/kernel/v3.x/linux-3.18.4.tar.xz>

3 配置内核

tar -xf linux-3.18.4.tar.xz -C /usr/src/

cd /usr/src/linux-3.18.4/

make menuconfig



在出来此界面后直接保存退出即可。

4 编译内核

make

如下则为成功:

IHEX firmware/mts\_edge.fw

H16TOFW firmware/edgeport/boot.fw

H16TOFW firmware/edgeport/boot2.fw

H16TOFW firmware/edgeport/down.fw

H16TOFW firmware/edgeport/down2.fw

IHEX firmware/edgeport/down3.bin

IHEX2FW firmware/whiteheat\_loader.fw

IHEX2FW firmware/whiteheat.fw

IHEX2FW firmware/keyspan\_pda/keyspan\_pda.fw

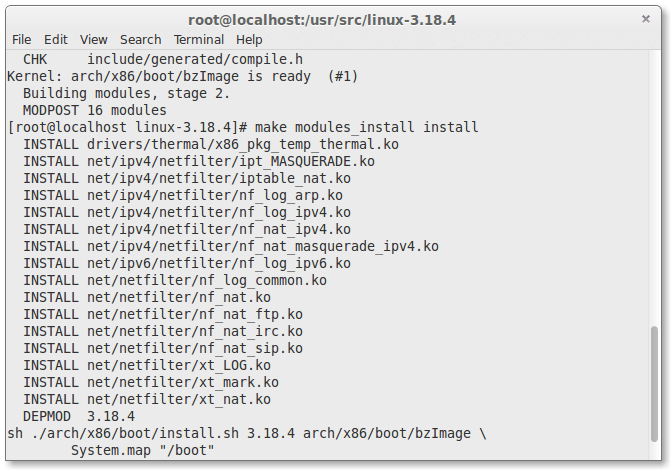
IHEX2FW firmware/keyspan\_pda/xircom\_pgs.fw

You have new mail in /var/spool/mail/root

5 安装内核

make modules\_install install

如下:



6 更改新内核为开机启动默认内核

cat /boot/grub2/grub.cfg |grep 3.18

grub2-set-default 'CentOS Linux (3.18.4) 7 (Core)'

备注:标黄参数由第一条指令返回结果中的entry得到

[root@node65 ~]# cat /boot/grub2/grub.cfg |grep 3.18

menuentry 'CentOS Linux (3.18.4) 7 (Core)' --class centos --class gnu-linux --class gnu --class os --unrestricted $menuentry\_id\_option 'gnulinux-3.10.0-327.el7.x86\_64-advanced-13a1ad74-6624-4bc4-afce-5f7aad9a507f' {

linux16 /vmlinuz-3.18.4 root=/dev/mapper/centos-root ro crashkernel=auto rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet LANG=en\_US.UTF-8

initrd16 /initramfs-3.18.4.img

6 重起机器并察看内核

reboot

uname -r

如下:



## 编译安装openvswitch

1 安装编译依赖软件

yum -y install make gcc openssl-devel autoconf automake rpm-build redhat-rpm-config python-devel openssl-devel kernel-devel kernel-debug-devel libtool wget python-twisted-core python-zope-interface PyQt4 desktop-file-utils groff graphviz selinux-policy-devel libcap-ng-devel

2 编译准备

mkdir -p ~/rpmbuild/SOURCES

wget <http://openvswitch.org/releases/openvswitch-2.5.0.tar.gz>

cp openvswitch-2.5.0.tar.gz ~/rpmbuild/SOURCES/  
tar xfz openvswitch-2.5.0.tar.gz

3 编译openvswitch

#rpmbuild -bb --nocheck --nocheck -D "kversion 3.18.4" ~/openvswitch-2.5.0/rhel/openvswitch-kmod-fedora.spec

#rpmbuild -bb --nocheck ~/openvswitch-2.5.0/rhel/openvswitch-fedora.spec

4 安装

yum install ~/rpmbuild/RPMS/x86\_64/openvswitch-kmod-2.5.0-1.el7.centos.x86\_64.rpm

yum install ~/rpmbuild/RPMS/noarch/python-openvswitch-2.5.0-1.el7.centos.noarch.rpm

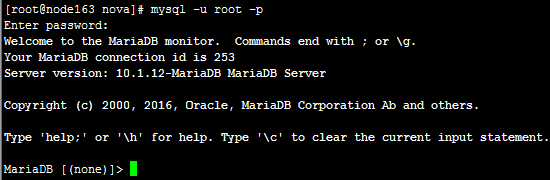
yum install ~/rpmbuild/RPMS/x86\_64/openvswitch-2.5.0-1.el7.centos.x86\_64.rpm

## 控制节点基础部分

1 控制节点上进行数据库授权

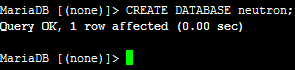
root用户登陆mysql:

# **mysql -u root -p**



创建neutron数据库

# **CREATE DATABASE neutron;**

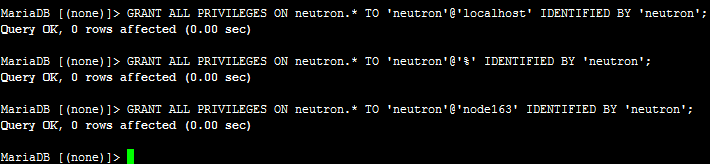


分配访问权限

**# GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'localhost' IDENTIFIED BY 'neutron';**

**# GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'%' IDENTIFIED BY 'neutron';**

**# GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'node163' IDENTIFIED BY 'neutron';**



退出数据库



2 创建neutron用户,密码设置为neutron

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



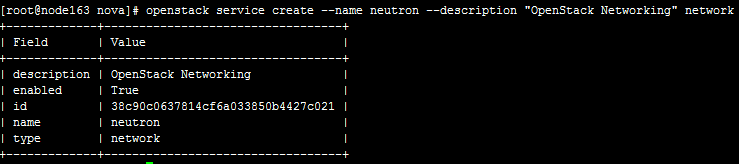
3 添加管理员角色到neutron用户中

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



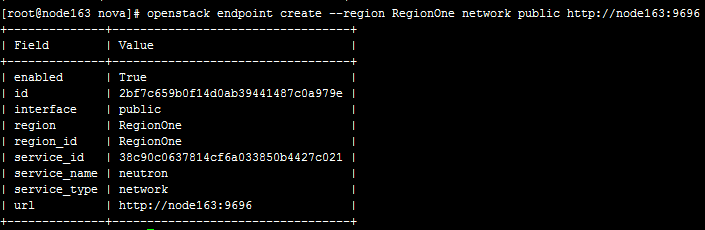
4 创建neutron服务的实体

**# openstack service create --name neutron --description "OpenStack Networking" network**

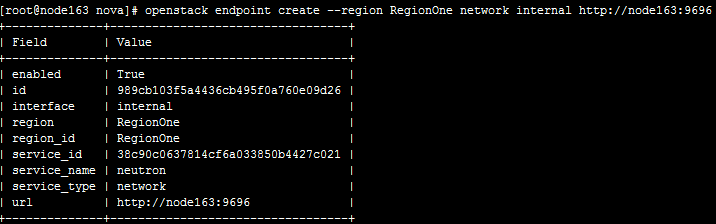


5 创建网络服务API终端

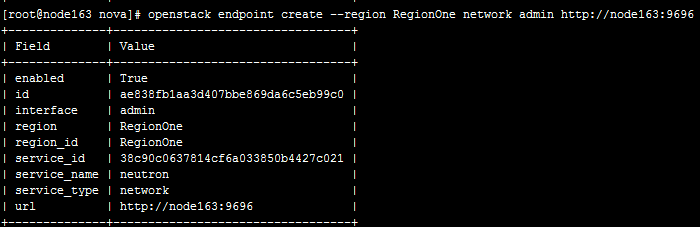
**# openstack endpoint create --region RegionOne network public** [**http://node163:9696**](http://node163:9696)

****

**#openstack endpoint create --region RegionOne network internal** [**http://node163:9696**](http://node163:9696)

****

**#openstack endpoint create --region RegionOne network admin http://node163:9696**



## 控制节点网络部分+网络节点（位于同一主机）

1 yum安装对应的包

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

2 修改配置文件

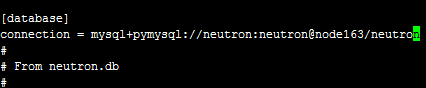
# vi /etc/neutron/neutron.conf

添加如下配置

[database]

**connection = mysql+pymysql://neutron:neutron@node163/neutron**

如下:



[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**

**router\_distributed = True**

l3\_ha = True

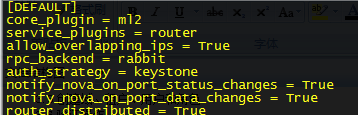
dhcp\_agents\_per\_network = 2

max\_l3\_agents\_per\_router = 3

min\_l3\_agents\_per\_router = 2

#l3\_ha\_net\_cidr = 169.254.192.0/18

如下:

****

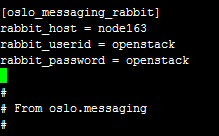
[oslo\_messaging\_rabbit]

**rabbit\_host = node163**

**rabbit\_userid = openstack**

**rabbit\_password = openstack**

如下:

****

[keystone\_authtoken]

**auth\_uri =** [**http://node163:5000**](http://node163:5000)

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

**project\_domain\_name = default**

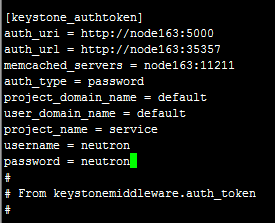
**user\_domain\_name = default**

**project\_name = service**

**username = neutron**

**password = neutron**

如下：

****

[nova]

**auth\_url = http://node163:35357**

**auth\_type = password**

**project\_domain\_name = default**

**user\_domain\_name = default**

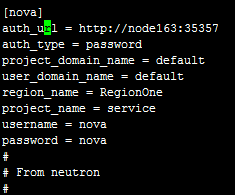
**region\_name = RegionOne**

**project\_name = service**

**username = nova**

**password = nova**

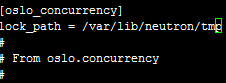
如下：



[oslo\_concurrency]

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

如下：



3 修改配置文件

# vi /etc/neutron/plugins/ml2/ml2\_conf.ini

添加如下内容

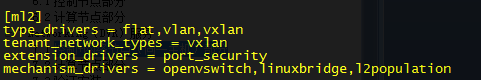
[ml2]

type\_drivers = flat,vlan,vxlan

tenant\_network\_types = vxlan

extension\_drivers = port\_security  
mechanism\_drivers = openvswitch,linuxbridge,l2population

如下:



[ml2\_type\_flat]

flat\_networks = external

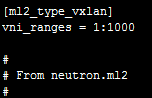
如下:



[ml2\_type\_vxlan]

**vni\_ranges = 1:1000**

如下:



[securitygroup]

firewall\_driver = iptables\_hybrid

如下:

****

4 配置openvswitch\_agent

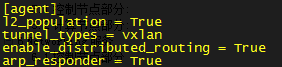
vi /etc/neutron/plugins/ml2/openvswitch\_agent.ini

[agent]

l2\_population = True  
tunnel\_types = vxlan  
enable\_distributed\_routing = True

arp\_responder = True

 如下:



[ovs]

local\_ip = TUNNELS\_IP  
bridge\_mappings = external:br-ex

如下:



[securitygroup]

firewall\_driver = iptables\_hybrid

如下:



5 配置layer-3 agent

vi /etc/neutron/l3\_agent.ini

[DEFAULT]

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

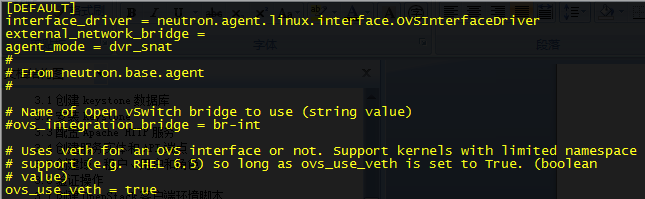
external\_network\_bridge =

agent\_mode = dvr\_snat

ovs\_use\_veth = true

#ha\_vrrp\_auth\_password = password

如下:



6 配置DHCP agent

vi /etc/neutron/dhcp\_agent.ini

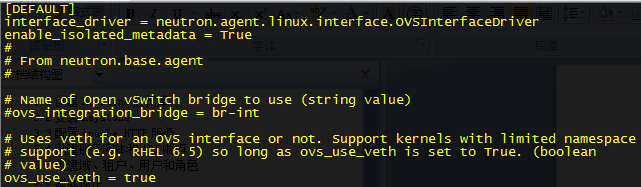
[DEFAULT]

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

enable\_isolated\_metadata = True

ovs\_use\_veth = true

如下:



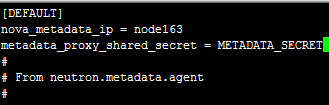
7 配置metadata agent 修改配置文件

# vi /etc/neutron/metadata\_agent.ini

[DEFAULT]

**nova\_metadata\_ip = node163**

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



8 配置nova和neutron结合部分 修改配置文件

#vi /etc/nova/nova.conf

[neutron]

**url = http://node163:9696**

**auth\_url = http://node163:35357**

**auth\_type = password**

**project\_domain\_name = default**

**user\_domain\_name = default**

**region\_name = RegionOne**

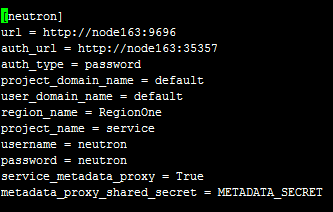
**project\_name = service**

**username = neutron**

**password = neutron**

**service\_metadata\_proxy = True**

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



9 创建软链接

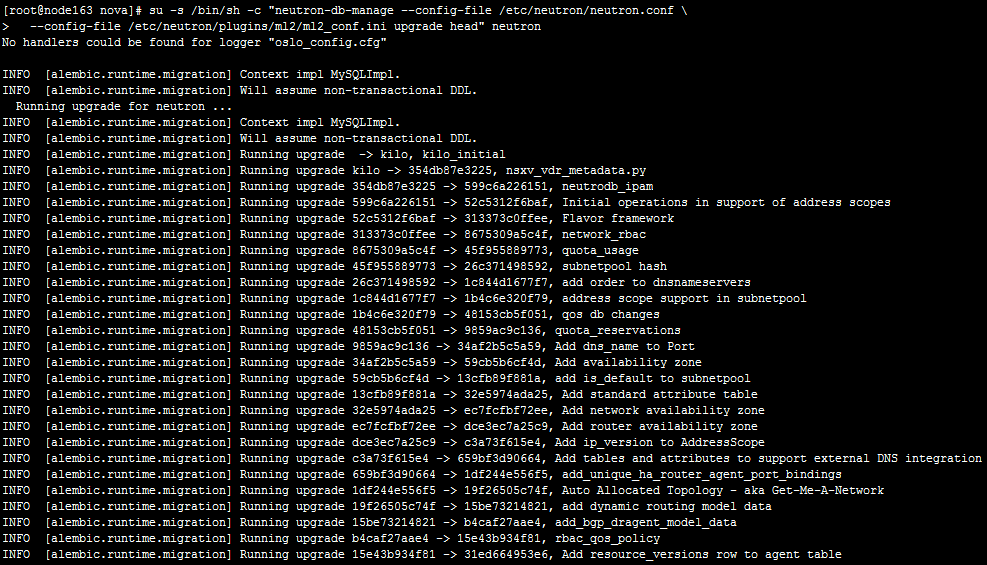
**ln -s /etc/neutron/plugins/ml2/ml2\_conf.ini /etc/neutron/plugin.ini**



10 填充neutron服务的数据库

**#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 建立openvswitch网桥br-ex

The OVS service provides the underlying virtual networking framework for instances. The  
integration bridge br-int handles internal instance network traffic within OVS. The external bridge br-ex handles external instance network traffic within OVS. The external bridge  
requires a port on the physical external network interface to provide instances with external network access. In essence, this port connects the virtual and physical external networks  
in your environment.  
1. Start the OVS service and configure it to start when the system boots:  
# systemctl enable openvswitch.service  
# systemctl start openvswitch.service

**2.** Create an external network bridge (br-ex) and add an associated port (eth1) to it:

Create the external bridge in */etc/sysconfig/network-scripts/ifcfg-br-ex*:

DEVICE=br-ex  
TYPE=OVSBridge  
DEVICETYPE=ovs  
ONBOOT=yes  
NM\_CONTROLLED=no  
BOOTPROTO=none

In */etc/sysconfig/network-scripts/ifcfg-eth1*, configure eth1 to connect to br-ex:

DEVICE=eth1  
TYPE=OVSPort  
DEVICETYPE=ovs  
OVS\_BRIDGE=br-ex  
ONBOOT=yes  
NM\_CONTROLLED=no  
BOOTPROTO=none

Reboot the node or restart the network service for the changes to take effect.

Note  
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**

12 重新启动nova的api服务

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



13 配置neutron开机启动并启动neutron服务

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

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

**neutron-metadata-agent.service neutron-l3-agent.service \**

**neutron-openvswitch-agent.service**

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

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

**neutron-metadata-agent.service neutron-l3-agent.service \**

**neutron-openvswitch-agent.service**

14 启动br-tun,br-int

**yum install net-tools**

**ifconfig br-tun up**

**ifconfig br-int up**

**创建定时启动脚本:**

**vi /etc/startbr.sh:**

**#!/bin/bash**

**for iter in {1..12};**

**do**

**brtun=$(/usr/sbin/ifconfig br-tun|grep UP|awk '{print $NF}');**

**brint=$(/usr/sbin/ifconfig br-int|grep UP|awk '{print $NF}');**

**brtunexisted=$(/usr/sbin/ifconfig br-tun|awk '{print $NF}'|head -1);**

**brintexisted=$(/usr/sbin/ifconfig br-int|awk '{print $NF}'|head -1);**

**if [ -z $brint ] && [ ! -z $brintexisted ];then /usr/sbin/ifconfig br-int up; fi;**

**if [ -z $brtun ] && [ ! -z $brtunexisted ];then /usr/sbin/ifconfig br-tun up; fi;**

**sleep 5s**

**done**

加执行权限:

chmod +x /etc/startbr.sh

**添加到定时任务:**

1.配置crontab，每五分钟运行一次时间同步

**echo "\*/1 \* \* \* \* /bin/bash /etc/startbr.sh"|crontab**

crontab -l 可以查看添加好的内容

2.设置crontab开机启动

chkconfig --level 35 crond on

## (初稿未验证)独立控制节点网络部分

1 yum安装对应的包

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

2 修改配置文件

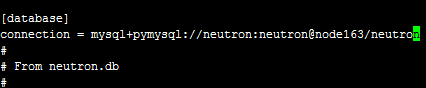
# vi /etc/neutron/neutron.conf

添加如下配置

[database]

**connection = mysql+pymysql://neutron:neutron@node163/neutron**

如下:



[DEFAULT]

**core\_plugin = ml2**

**service\_plugins = router**

**allow\_overlapping\_ips = True**

**rpc\_backend = rabbit**

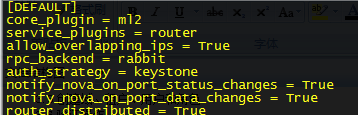
**auth\_strategy = keystone**

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

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

**router\_distributed = True**

如下:

****

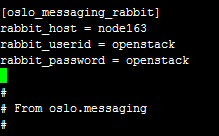
[oslo\_messaging\_rabbit]

**rabbit\_host = node163**

**rabbit\_userid = openstack**

**rabbit\_password = openstack**

如下:

****

[keystone\_authtoken]

**auth\_uri =** [**http://node163:5000**](http://node163:5000)

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

**project\_domain\_name = default**

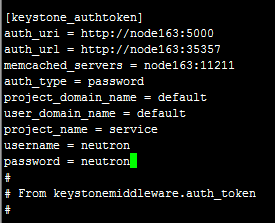
**user\_domain\_name = default**

**project\_name = service**

**username = neutron**

**password = neutron**

如下：

****

[nova]

**auth\_url = http://node163:35357**

**auth\_type = password**

**project\_domain\_name = default**

**user\_domain\_name = default**

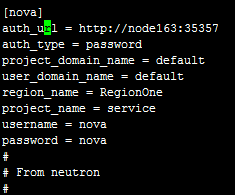
**region\_name = RegionOne**

**project\_name = service**

**username = nova**

**password = nova**

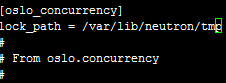
如下：



[oslo\_concurrency]

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

如下：



3 修改配置文件

# vi /etc/neutron/plugins/ml2/ml2\_conf.ini

添加如下内容

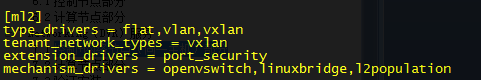
[ml2]

type\_drivers = flat,vlan,vxlan

tenant\_network\_types = vxlan

extension\_drivers = port\_security  
mechanism\_drivers = openvswitch,linuxbridge,l2population

如下:



[ml2\_type\_flat]

flat\_networks = external

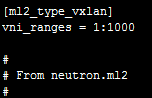
如下:



[ml2\_type\_vxlan]

**vni\_ranges = 1:1000**

如下:



[securitygroup]

firewall\_driver = iptables\_hybrid

如下:

****

4 配置nova和neutron结合部分 修改配置文件

#vi /etc/nova/nova.conf

[neutron]

**url = http://node163:9696**

**auth\_url = http://node163:35357**

**auth\_type = password**

**project\_domain\_name = default**

**user\_domain\_name = default**

**region\_name = RegionOne**

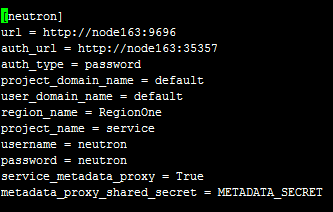
**project\_name = service**

**username = neutron**

**password = neutron**

**service\_metadata\_proxy = True**

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



5 创建软链接

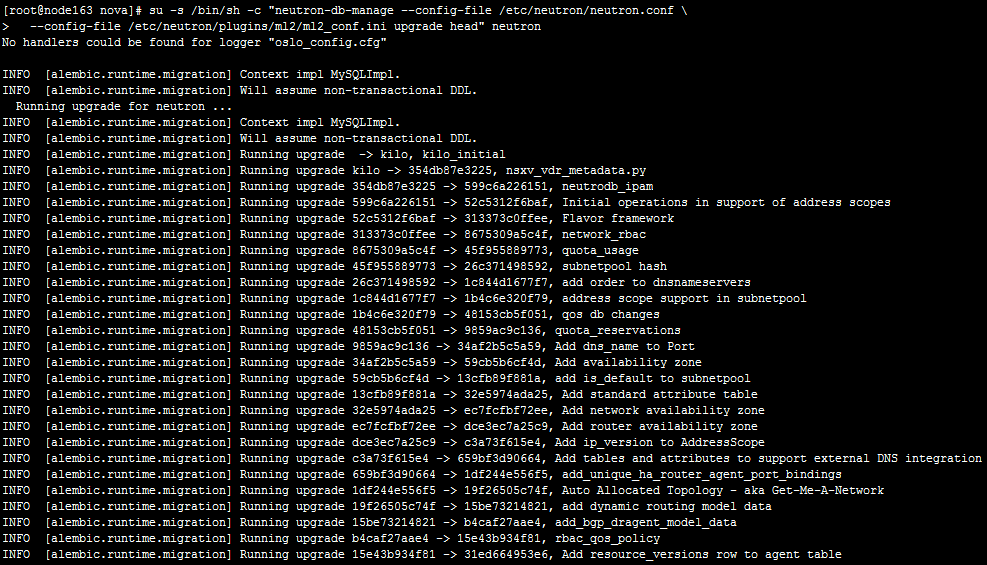
**ln -s /etc/neutron/plugins/ml2/ml2\_conf.ini /etc/neutron/plugin.ini**



6 填充neutron服务的数据库

**#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**



7 重新启动nova的api服务

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



8 配置neutron开机启动并启动neutron服务

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

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

9 启动br-tun,br-int

**yum install net-tools**

**ifconfig br-tun up**

**ifconfig br-int up**

**创建定时启动脚本:**

**vi /etc/startbr.sh:**

**#!/bin/bash**

**for iter in {1..12};**

**do**

**brtun=$(/usr/sbin/ifconfig br-tun|grep UP|awk '{print $NF}');**

**brint=$(/usr/sbin/ifconfig br-int|grep UP|awk '{print $NF}');**

**brtunexisted=$(/usr/sbin/ifconfig br-tun|awk '{print $NF}'|head -1);**

**brintexisted=$(/usr/sbin/ifconfig br-int|awk '{print $NF}'|head -1);**

**if [ -z $brint ] && [ ! -z $brintexisted ];then /usr/sbin/ifconfig br-int up; fi;**

**if [ -z $brtun ] && [ ! -z $brtunexisted ];then /usr/sbin/ifconfig br-tun up; fi;**

**sleep 5s**

**done**

加执行权限:

chmod +x /etc/startbr.sh

**添加到定时任务:**

1.配置crontab，每五分钟运行一次时间同步

**echo "\*/1 \* \* \* \* /bin/bash /etc/startbr.sh"|crontab**

crontab -l 可以查看添加好的内容

2.设置crontab开机启动

chkconfig --level 35 crond on

## 独立网络节点部分

1 yum安装对应的包

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

2 修改配置文件

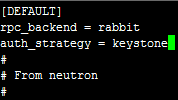
# vi /etc/neutron/neutron.conf

添加如下配置

[DEFAULT]

**rpc\_backend = rabbit**

**auth\_strategy = keystone**

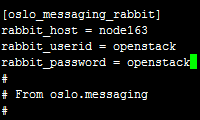


[oslo\_messaging\_rabbit]

**rabbit\_host = node163**

**rabbit\_userid = openstack**

**rabbit\_password = openstack**



[keystone\_authtoken]

**auth\_uri = http://node163:5000**

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

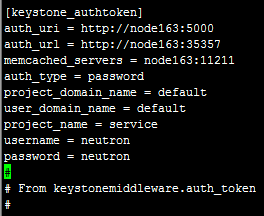
**project\_domain\_name = default**

**user\_domain\_name = default**

**project\_name = service**

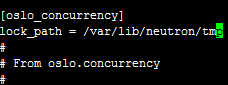
**username = neutron**

**password = neutron**



[oslo\_concurrency]

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



3 配置openvswitch\_agent

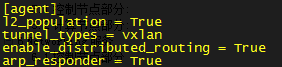
vi /etc/neutron/plugins/ml2/openvswitch\_agent.ini

[agent]

l2\_population = True  
tunnel\_types = vxlan  
enable\_distributed\_routing = True

arp\_responder = True

 如下:



[ovs]

local\_ip = TUNNELS\_IP  
bridge\_mappings = external:br-ex

如下:



[securitygroup]

firewall\_driver = iptables\_hybrid

如下:



4 配置layer-3 agent

vi /etc/neutron/l3\_agent.ini

[DEFAULT]

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

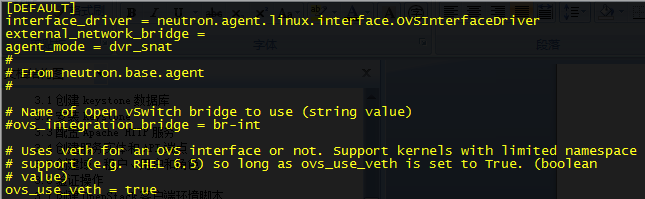
external\_network\_bridge =

agent\_mode = dvr\_snat

ovs\_use\_veth = true

#ha\_vrrp\_auth\_password = password

如下:



5 配置DHCP agent

vi /etc/neutron/dhcp\_agent.ini

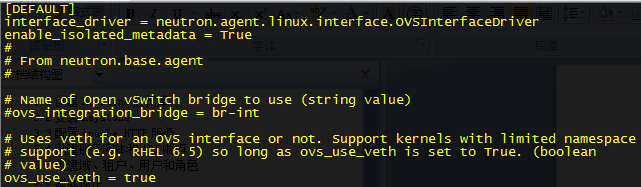
[DEFAULT]

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

enable\_isolated\_metadata = True

ovs\_use\_veth = true

如下:



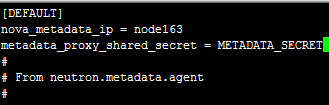
6 配置metadata agent 修改配置文件

# vi /etc/neutron/metadata\_agent.ini

[DEFAULT]

**nova\_metadata\_ip = node163**

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



7 建立openvswitch网桥br-ex

The OVS service provides the underlying virtual networking framework for instances. The  
integration bridge br-int handles internal instance network traffic within OVS. The external bridge br-ex handles external instance network traffic within OVS. The external bridge  
requires a port on the physical external network interface to provide instances with external network access. In essence, this port connects the virtual and physical external networks  
in your environment.  
1. Start the OVS service and configure it to start when the system boots:  
# systemctl enable openvswitch.service  
# systemctl start openvswitch.service

**2.** Create an external network bridge (br-ex) and add an associated port (eth1) to it:

Create the external bridge in */etc/sysconfig/network-scripts/ifcfg-br-ex*:

DEVICE=br-ex  
TYPE=OVSBridge  
DEVICETYPE=ovs  
ONBOOT=yes  
NM\_CONTROLLED=no  
BOOTPROTO=none

In */etc/sysconfig/network-scripts/ifcfg-eth1*, configure eth1 to connect to br-ex:

DEVICE=eth1  
TYPE=OVSPort  
DEVICETYPE=ovs  
OVS\_BRIDGE=br-ex  
ONBOOT=yes  
NM\_CONTROLLED=no  
BOOTPROTO=none

Reboot the node or restart the network service for the changes to take effect.

Note  
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**

8 配置neutron开机启动并启动neutron服务

**# systemctl enable neutron-dhcp-agent.service \**

**neutron-metadata-agent.service neutron-l3-agent.service \**

**neutron-openvswitch-agent.service**

**# systemctl start neutron-dhcp-agent.service \**

**neutron-metadata-agent.service neutron-l3-agent.service \**

**neutron-openvswitch-agent.service**

8 启动br-tun,br-int

**yum install net-tools**

**ifconfig br-tun up**

**ifconfig br-int up**

**创建定时启动脚本:**

**vi /etc/startbr.sh:**

**#!/bin/bash**

**for iter in {1..12};**

**do**

**brtun=$(/usr/sbin/ifconfig br-tun|grep UP|awk '{print $NF}');**

**brint=$(/usr/sbin/ifconfig br-int|grep UP|awk '{print $NF}');**

**brtunexisted=$(/usr/sbin/ifconfig br-tun|awk '{print $NF}'|head -1);**

**brintexisted=$(/usr/sbin/ifconfig br-int|awk '{print $NF}'|head -1);**

**if [ -z $brint ] && [ ! -z $brintexisted ];then /usr/sbin/ifconfig br-int up; fi;**

**if [ -z $brtun ] && [ ! -z $brtunexisted ];then /usr/sbin/ifconfig br-tun up; fi;**

**sleep 5s**

**done**

加执行权限:

chmod +x /etc/startbr.sh

**添加到定时任务:**

1.配置crontab，每五分钟运行一次时间同步

**echo "\*/1 \* \* \* \* /bin/bash /etc/startbr.sh"|crontab**

crontab -l 可以查看添加好的内容

2.设置crontab开机启动

chkconfig --level 35 crond on

## 计算节点部分

1 yum安装包

**yum install openstack-neutron-linuxbridge ebtables ipset**

**yum install openstack-neutron openstack-neutron-ml2**

**yum install openstack-neutron-openvswitch**

2 修改配置文件

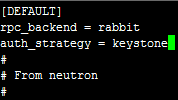
# vi /etc/neutron/neutron.conf

添加如下配置

[DEFAULT]

**rpc\_backend = rabbit**

**auth\_strategy = keystone**

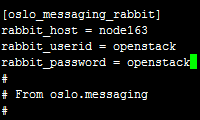


[oslo\_messaging\_rabbit]

**rabbit\_host = node163**

**rabbit\_userid = openstack**

**rabbit\_password = openstack**



[keystone\_authtoken]

**auth\_uri = http://node163:5000**

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

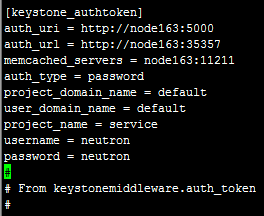
**project\_domain\_name = default**

**user\_domain\_name = default**

**project\_name = service**

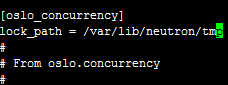
**username = neutron**

**password = neutron**



[oslo\_concurrency]

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



3 配置openvswitch\_agent

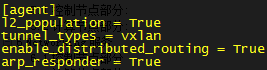
#vi /etc/neutron/plugins/ml2/openvswitch\_agent.ini

[agent]

l2\_population = True  
tunnel\_types = vxlan  
enable\_distributed\_routing = True

arp\_responder = True

如下:



[ovs]

local\_ip = TUNNELS\_IP  
bridge\_mappings = external:br-ex

如下:



[securitygroup]

firewall\_driver = iptables\_hybrid

如下:



4 配置l3\_agent

vi /etc/neutron/l3\_agent.ini:

[DEFAULT]

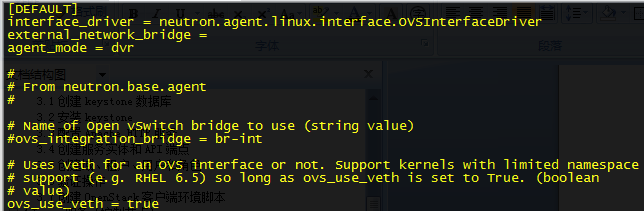
interface\_driver = neutron.agent.linux.interface.OVSInterfaceDriver

external\_network\_bridge =

agent\_mode = dvr

ovs\_use\_veth = true

如下:



5 配置metadata\_agent

vi /etc/neutron/metadata\_agent.ini:

[DEFAULT]

nova\_metadata\_ip = controller

metadata\_proxy\_shared\_secret = METADATA\_SECRET

如下:



6 修改配置文件 /etc/nova/nova.conf 添加如下配置

[neutron]

**url = http://node163:9696**

**auth\_url = http://node163:35357**

**auth\_type = password**

**project\_domain\_name = default**

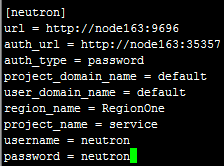
**user\_domain\_name = default**

**region\_name = RegionOne**

**project\_name = service**

**username = neutron**

**password = neutron**



7建立openvswitch网桥br-ex

The OVS service provides the underlying virtual networking framework for instances. The  
integration bridge br-int handles internal instance network traffic within OVS. The external bridge br-ex handles external instance network traffic within OVS. The external bridge  
requires a port on the physical external network interface to provide instances with external network access. In essence, this port connects the virtual and physical external networks  
in your environment.  
1. Start the OVS service and configure it to start when the system boots:  
# systemctl enable openvswitch.service  
# systemctl start openvswitch.service

**2.** Create an external network bridge (br-ex) and add an associated port (eth1) to it:

Create the external bridge in */etc/sysconfig/network-scripts/ifcfg-br-ex*:

DEVICE=br-ex  
TYPE=OVSBridge  
DEVICETYPE=ovs  
ONBOOT=yes  
NM\_CONTROLLED=no  
BOOTPROTO=none

In */etc/sysconfig/network-scripts/ifcfg-eth1*, configure eth1 to connect to br-ex:

DEVICE=eth1  
TYPE=OVSPort  
DEVICETYPE=ovs  
OVS\_BRIDGE=br-ex  
ONBOOT=yes  
NM\_CONTROLLED=no  
BOOTPROTO=none

Reboot the node or restart the network service for the changes to take effect.

Note  
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**

8 重新启动nova计算服务

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



9启动neutron相关服务并设置开机启动

**chkconfig neutron-openvswitch-agent on**

**chkconfig neutron-metadata-agent on**

**chkconfig neutron-l3-agent on**

**service neutron-openvswitch-agent start**

**service neutron-metadata-agent start**

**service neutron-l3-agent start**

10 启动br-tun,br-int

**yum install net-tools**

**ifconfig br-tun up**

**ifconfig br-int up**

**创建定时启动脚本:**

**vi /etc/startbr.sh:**

**#!/bin/bash**

**for iter in {1..12};**

**do**

**brtun=$(/usr/sbin/ifconfig br-tun|grep UP|awk '{print $NF}');**

**brint=$(/usr/sbin/ifconfig br-int|grep UP|awk '{print $NF}');**

**brtunexisted=$(/usr/sbin/ifconfig br-tun|awk '{print $NF}'|head -1);**

**brintexisted=$(/usr/sbin/ifconfig br-int|awk '{print $NF}'|head -1);**

**if [ -z $brint ] && [ ! -z $brintexisted ];then /usr/sbin/ifconfig br-int up; fi;**

**if [ -z $brtun ] && [ ! -z $brtunexisted ];then /usr/sbin/ifconfig br-tun up; fi;**

**sleep 5s**

**done**

加执行权限:

chmod +x /etc/startbr.sh

**添加到定时任务:**

1.配置crontab，每五分钟运行一次时间同步

**echo "\*/1 \* \* \* \* /bin/bash /etc/startbr.sh"|crontab**

crontab -l 可以查看添加好的内容

2.设置crontab开机启动

chkconfig --level 35 crond on

## 验证安装

### neutron节点察看服务

网络节点应该有Metadata agent，Open vSwitch agent，DHCP agent，L3 agent

计算节点应该有Metadata agent，Open vSwitch agent， L3 agent

$ neutron agent-list

+--------------------------------------+--------------------+----------+-------+----------------+---------------------------+

| id | agent\_type | host | alive | admin\_state\_up | binary |

+--------------------------------------+--------------------+----------+-------+----------------+---------------------------+

| 10b084e5-4ab8-43d6-9b04-6d56f27f9cd4 | Metadata agent | network1 | :-) | True | neutron-metadata-agent |

| 2f90ef81-3eed-4ecf-b6b9-2d2c21dda85c | Open vSwitch agent | compute2 | :-) | True | neutron-openvswitch-agent |

| 319563ac-88f9-4352-b63e-e55beb673372 | DHCP agent | network1 | :-) | True | neutron-dhcp-agent |

| 3345723e-16e8-4b74-9d15-d7f1f977a3bd | Open vSwitch agent | compute1 | :-) | True | neutron-openvswitch-agent |

| 4643c811-a54a-41da-91a8-c2328bcaeea3 | Open vSwitch agent | network1 | :-) | True | neutron-openvswitch-agent |

| 5ad81671-efc3-4acc-9d5d-030a1c4f6a25 | L3 agent | compute1 | :-) | True | neutron-l3-agent |

| 641337fa-99c2-468d-8d7e-89277d6ba144 | Metadata agent | compute1 | :-) | True | neutron-metadata-agent |

| 9372e008-bd29-4436-8e01-8ddfd50d2b74 | L3 agent | network1 | :-) | True | neutron-l3-agent |

| af9d1169-1012-4440-9de2-778c8fce21b9 | L3 agent | compute2 | :-) | True | neutron-l3-agent |

| ee59e3ba-ee3c-4621-b3d5-c9d8123b6cc5 | Metadata agent | compute2 | :-) | True | neutron-metadata-agent |

+--------------------------------------+--------------------+----------+-------+----------------+---------------------------+

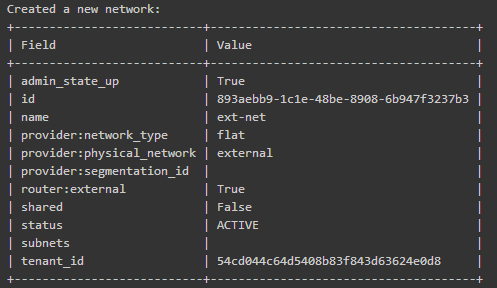
### 创建初始网络

This example creates a flat external network and a VXLAN project network.

1. Source the administrative project credentials.
2. Create the external network:

neutron net-create ext-net --router:external \

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

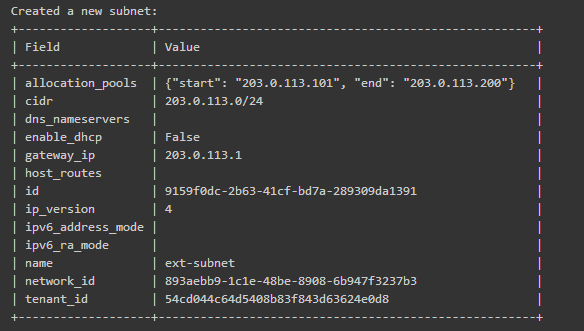


1. Create a subnet on the external network:

neutron subnet-create ext-net 203.0.113.0/24 --allocation-pool \

start=203.0.113.101,end=203.0.113.200 --disable-dhcp \

--gateway 203.0.113.1

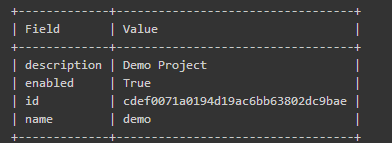


Note

The example configuration contains vlan as the first project network type. Only a privileged user can create other types of networks such as GRE or VXLAN. The following commands use the admin project credentials to create a VXLAN project network.

1. Obtain the ID of a regular project. For example, using the demo project:

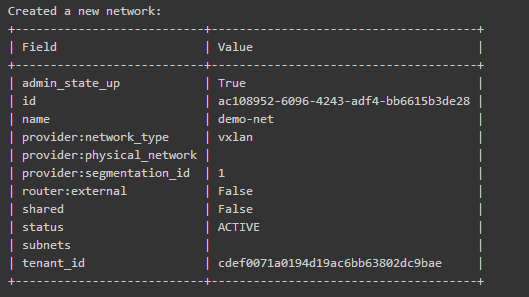
$ openstack project show demo



1. Create the project network:

$ neutron net-create demo-net --tenant-id cdef0071a0194d19ac6bb63802dc9bae \

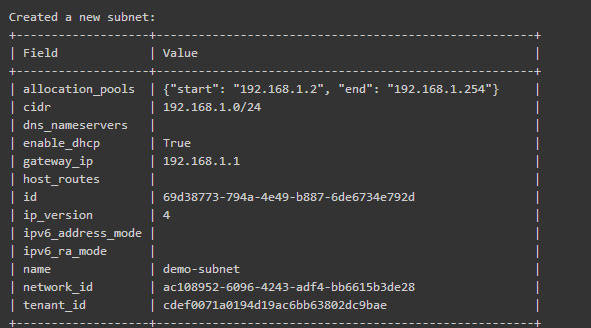
--provider:network\_type vxlan



1. Source the regular project credentials.
2. Create a subnet on the project network:

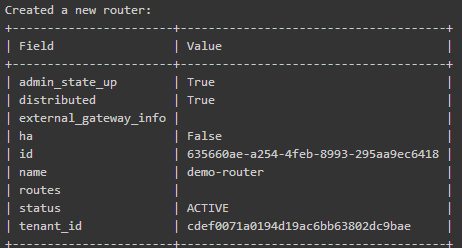
$ neutron subnet-create demo-net --name demo-subnet --gateway 192.168.1.1 \

192.168.1.0/24



1. Create a distributed project router:

$ neutron router-create demo-router



Note

Default policy might prevent the ‘distributed` flag from appearing in the command output for non-privileged users.

1. Attach the project network to the router:

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

Added interface b1a894fd-aee8-475c-9262-4342afdc1b58 to router demo-router.

1. Add a gateway to the external network for the project network on the router:

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

Set gateway for router demo-router

### 验证DVR网络操作

1. On the network node, verify creation of the snat, qrouter, and qdhcp namespaces:

$ ip netns

snat-4d7928a0-4a3c-4b99-b01b-97da2f97e279

qrouter-4d7928a0-4a3c-4b99-b01b-97da2f97e279

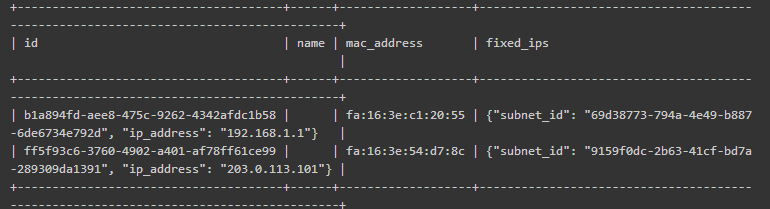
qdhcp-353f5937-a2d3-41ba-8225-fa1af2538141

Note

One or more namespaces might not exist until launching an instance.

1. Source the administrative project credentials.
2. Determine the external network gateway IP address for the project network on the router, typically the lowest IP address in the external subnet IP allocation range:

$ neutron router-port-list demo-router



1. On the controller node or any host with access to the external network, ping the external network gateway IP address on the project router:

$ ping -c 4 203.0.113.101

PING 203.0.113.101 (203.0.113.101) 56(84) bytes of data.

64 bytes from 203.0.113.101: icmp\_req=1 ttl=64 time=0.619 ms

64 bytes from 203.0.113.101: icmp\_req=2 ttl=64 time=0.189 ms

64 bytes from 203.0.113.101: icmp\_req=3 ttl=64 time=0.165 ms

64 bytes from 203.0.113.101: icmp\_req=4 ttl=64 time=0.216 ms

--- 203.0.113.101 ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 2999ms

rtt min/avg/max/mdev = 0.165/0.297/0.619/0.187 ms

1. Source the regular project credentials.
2. Launch an instance with an interface on the project network.
3. On the compute node with the instance, verify creation of the qrouter namespace:

$ ip netns

qrouter-4d7928a0-4a3c-4b99-b01b-97da2f97e279

1. Obtain console access to the instance.
   1. Test connectivity to the project router:

$ ping -c 4 192.168.1.1

PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.

64 bytes from 192.168.1.1: icmp\_req=1 ttl=64 time=0.357 ms

64 bytes from 192.168.1.1: icmp\_req=2 ttl=64 time=0.473 ms

64 bytes from 192.168.1.1: icmp\_req=3 ttl=64 time=0.504 ms

64 bytes from 192.168.1.1: icmp\_req=4 ttl=64 time=0.470 ms

--- 192.168.1.1 ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 2998ms

rtt min/avg/max/mdev = 0.357/0.451/0.504/0.055 ms

* 1. Test connectivity to the Internet:

$ ping -c 4 openstack.org

PING openstack.org (174.143.194.225) 56(84) bytes of data.

64 bytes from 174.143.194.225: icmp\_req=1 ttl=53 time=17.4 ms

64 bytes from 174.143.194.225: icmp\_req=2 ttl=53 time=17.5 ms

64 bytes from 174.143.194.225: icmp\_req=3 ttl=53 time=17.7 ms

64 bytes from 174.143.194.225: icmp\_req=4 ttl=53 time=17.5 ms

--- openstack.org ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 3003ms

rtt min/avg/max/mdev = 17.431/17.575/17.734/0.143 ms

1. Create the appropriate security group rules to allow ping and SSH access to the instance. For example:

$ nova secgroup-add-rule default icmp -1 -1 0.0.0.0/0

+-------------+-----------+---------+-----------+--------------+

| IP Protocol | From Port | To Port | IP Range | Source Group |

+-------------+-----------+---------+-----------+--------------+

| icmp | -1 | -1 | 0.0.0.0/0 | |

+-------------+-----------+---------+-----------+--------------+

$ nova secgroup-add-rule default tcp 22 22 0.0.0.0/0

+-------------+-----------+---------+-----------+--------------+

| IP Protocol | From Port | To Port | IP Range | Source Group |

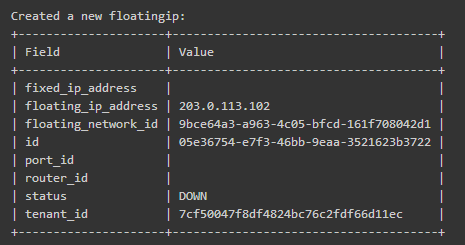
+-------------+-----------+---------+-----------+--------------+

| tcp | 22 | 22 | 0.0.0.0/0 | |

+-------------+-----------+---------+-----------+--------------+

1. Create a floating IP address on the external network:

$ neutron floatingip-create ext-net



1. Associate the floating IP address with the instance:

$ nova floating-ip-associate demo-instance1 203.0.113.102

1. Verify addition of the floating IP address to the instance:

$ nova list

+--------------------------------------+----------------+--------+------------+-------------+-----------------------------------------+

| ID | Name | Status | Task State | Power State | Networks |

+--------------------------------------+----------------+--------+------------+-------------+-----------------------------------------+

| 05682b91-81a1-464c-8f40-8b3da7ee92c5 | demo-instance1 | ACTIVE | - | Running | demo-net=192.168.1.3, 203.0.113.102 |

+--------------------------------------+----------------+--------+------------+-------------+-----------------------------------------+

1. On the compute node with the instance, verify creation of the fip namespace:

$ ip netns

fip-2c7bd9c2-8ab0-46ef-b7c1-023ce0452c24

1. On the controller node or any host with access to the external network, ping the floating IP address associated with the instance:

$ ping -c 4 203.0.113.102

PING 203.0.113.102 (203.0.113.112) 56(84) bytes of data.

64 bytes from 203.0.113.102: icmp\_req=1 ttl=63 time=3.18 ms

64 bytes from 203.0.113.102: icmp\_req=2 ttl=63 time=0.981 ms

64 bytes from 203.0.113.102: icmp\_req=3 ttl=63 time=1.06 ms

64 bytes from 203.0.113.102: icmp\_req=4 ttl=63 time=0.929 ms

--- 203.0.113.102 ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 3002ms

rtt min/avg/max/mdev = 0.929/1.539/3.183/0.951 ms

### 验证VRRP网络操作

# Dashboard（控制节点）

## 安装配置

1. yum安装

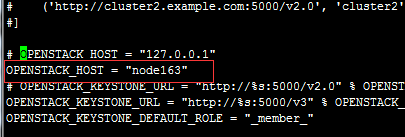
**#yum install openstack-dashboard**



1. 修改文件：

# vi /etc/openstack-dashboard/local\_settings

* Configure the dashboard to use OpenStack services on the node163 node

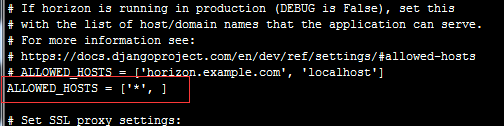


把以下配置

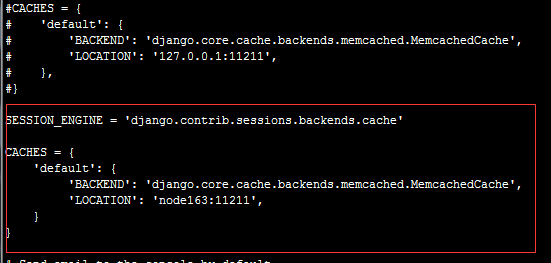
ALLOWED\_HOSTS = ['horizon.example.com','localhost']

改为：

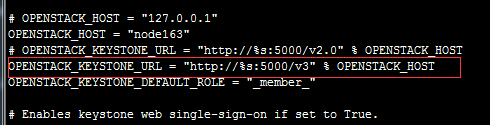
ALLOWED\_HOSTS = ['\*']



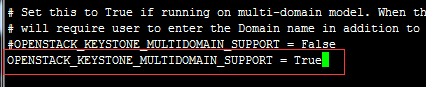
* configure the memcached session storage service:



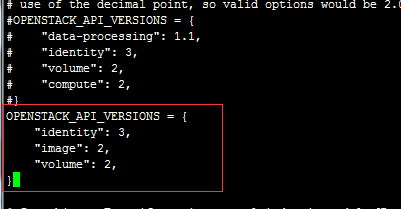
* Enable the Identity API version 3



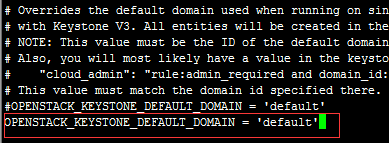
* Enable support for domains



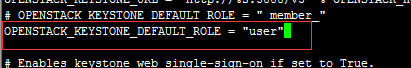
* Configure API versions



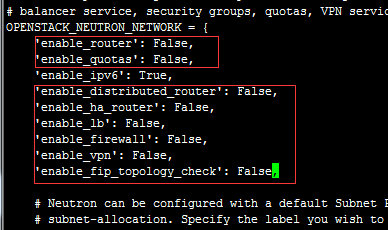
* Configure default as the default domain for users that you create via the dashboard



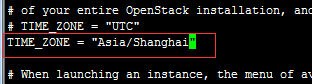
* Configure user as the default role for users that you create via the dashboard



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

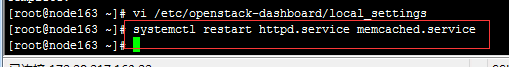


* Optionally, configure the time zone



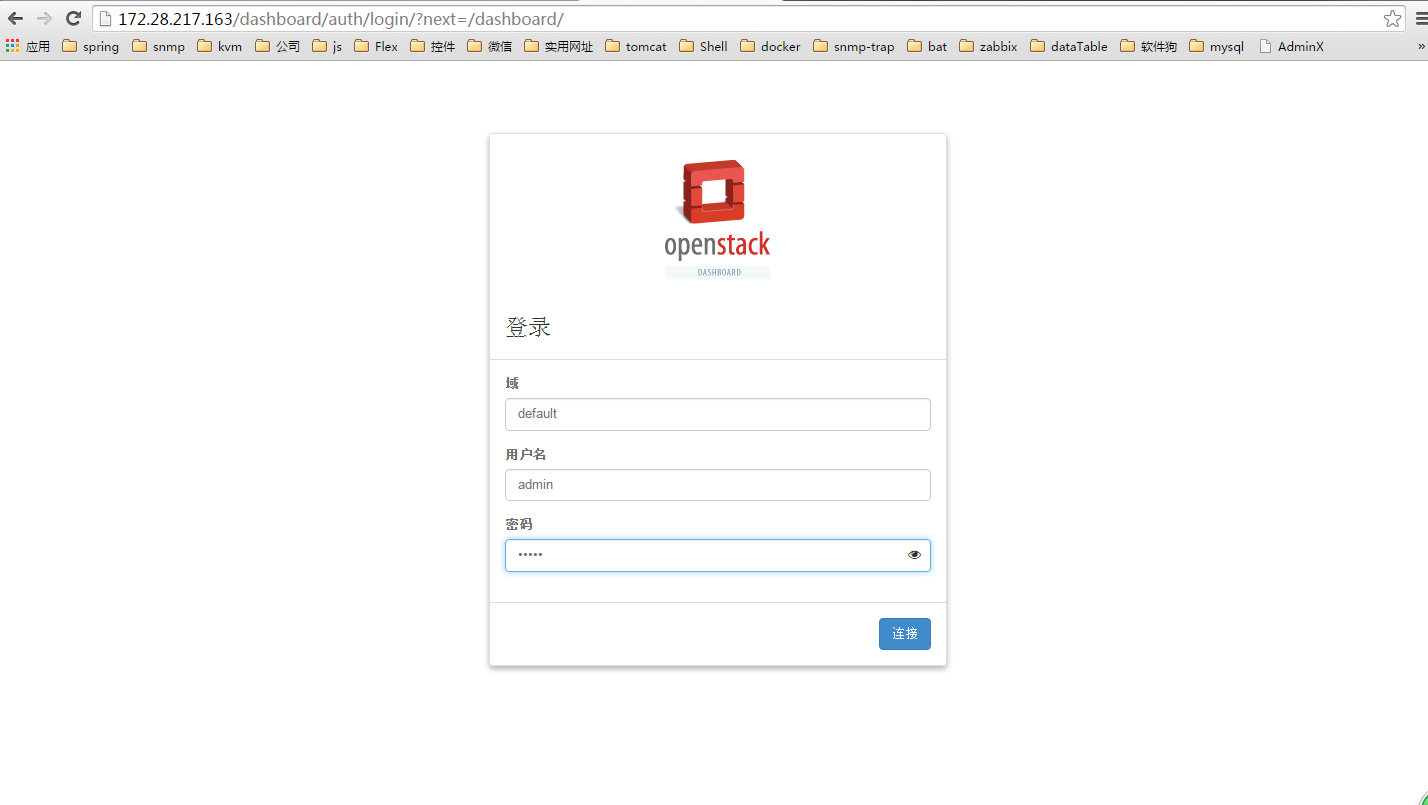
1. Restart the web server and session storage service:

**systemctl restart httpd.service memcached.service**

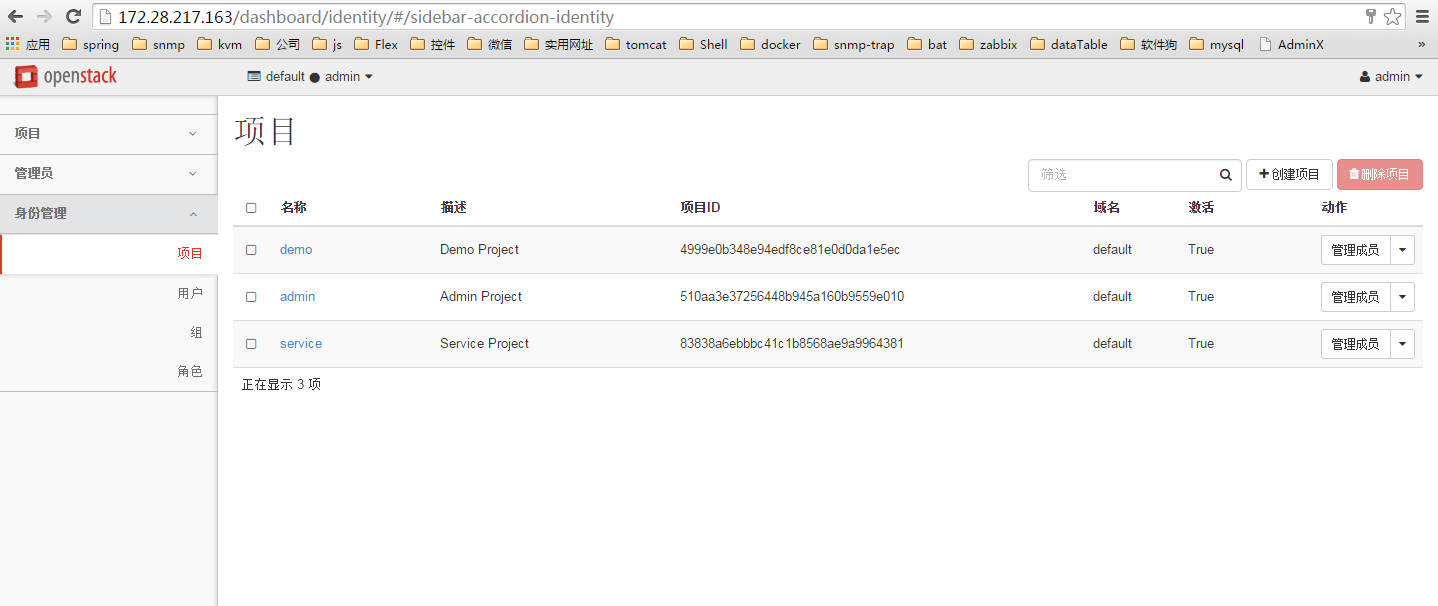
****

## 验证操作

浏览器访问：<http://172.28.217.163/dashboard>



登陆成功：



# Telemetry服务（控制节点）

## 安装NoSql数据库MongoDB

1. Yum安装

**yum install mongodb-server mongodb**

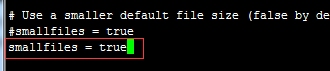
****

1. 修改配置：/etc/mongod.conf

* 配置bind\_ip为控制节点的ip地址



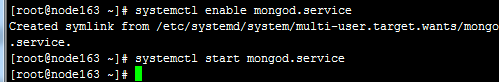
* 默认情况下,MongoDB /var/lib/mongodb/journal目录中创建几个1 GB的日志文件。如果你想减少每个日志文件的大小为128 MB,限制总日志空间消费为512 MB,可以配置smallfiles



1. 启动MongoDB服务并配置为开机启动

**# systemctl enable mongod.service**

**# systemctl start mongod.service**

****

## 准备

1. 创建ceilometer数据库

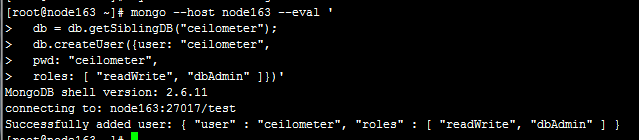
**mongo --host node163 --eval '**

**db = db.getSiblingDB("ceilometer");**

**db.createUser({user: "ceilometer",**

**pwd: "ceilometer",**

**roles: [ "readWrite", "dbAdmin" ]})'**



1. 使用admin-openrc的环境变量

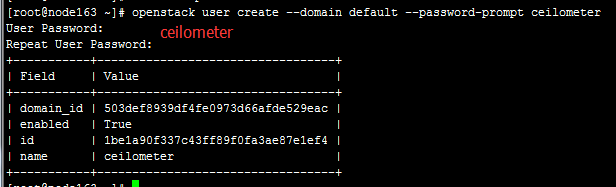
**. admin-openrc**

****

1. 创建服务凭证

* 创建ceilometer用户

**openstack user create --domain default --password-prompt ceilometer**



* 添加admin角色到ceilometer用户

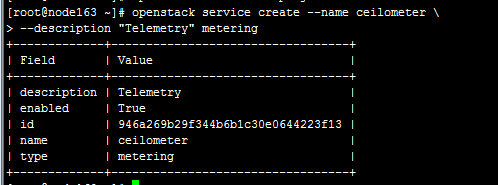
**openstack role add --project service --user ceilometer admin**

****

* 创建ceilometer服务实体

**openstack service create --name ceilometer \**

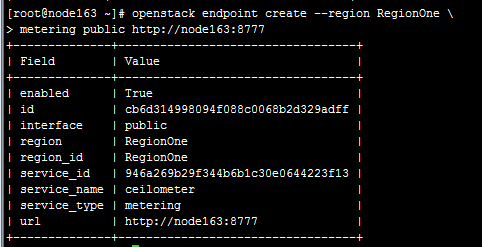
**--description "Telemetry" metering**

****

1. 创建Telemetry服务API端点

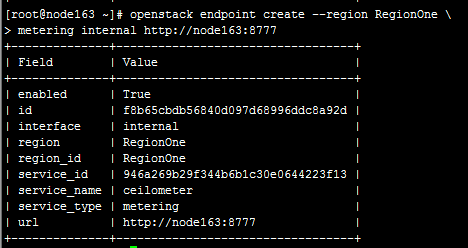
**openstack endpoint create --region RegionOne \**

**metering public http://node163:8777**



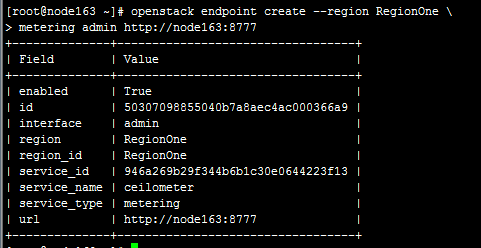
**openstack endpoint create --region RegionOne \**

**metering internal** [**http://node163:8777**](http://node163:8777)

****

**openstack endpoint create --region RegionOne \**

**metering admin** [**http://node163:8777**](http://node163:8777)

****

## 安装和配置

1. Yum安装

**yum install openstack-ceilometer-api \**

**openstack-ceilometer-collector openstack-ceilometer-notification \**

**openstack-ceilometer-central python-ceilometerclient**

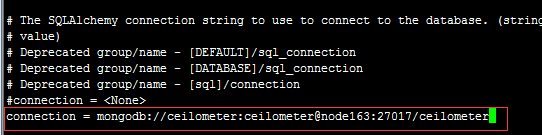
****

1. 修改配置/etc/ceilometer/ceilometer.conf

**[database]**

**...**

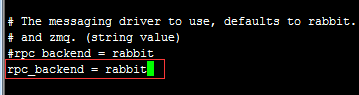
**connection = mongodb://ceilometer:ceilometer@node163:27017/ceilometer**

****

**[DEFAULT]**

**...**

**rpc\_backend = rabbit**

****

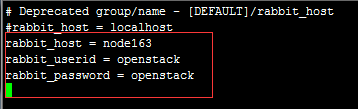
**[oslo\_messaging\_rabbit]**

**...**

**rabbit\_host = node163**

**rabbit\_userid = openstack**

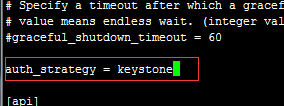
**rabbit\_password = openstack**

****

**[DEFAULT]**

**...**

**auth\_strategy = keystone**

****

**[keystone\_authtoken]**

**...**

**auth\_uri = http://node163:5000**

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

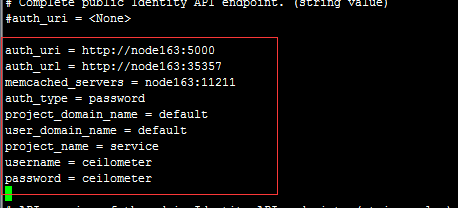
**project\_domain\_name = default**

**user\_domain\_name = default**

**project\_name = service**

**username = ceilometer**

**password = ceilometer**

****

**[service\_credentials]**

**...**

**os\_auth\_url = http://node163:5000/v2.0**

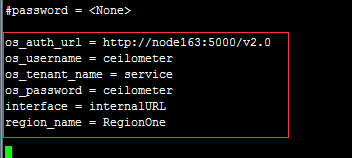
**os\_username = ceilometer**

**os\_tenant\_name = service**

**os\_password = ceilometer**

**interface = internalURL**

**region\_name = RegionOne**

****

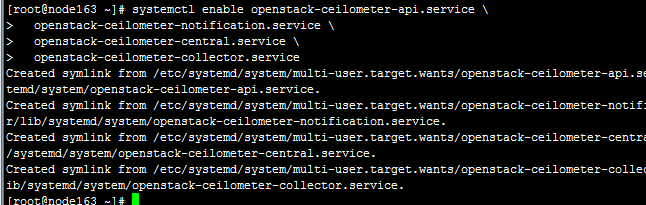
1. 开启Telemetry服务并配置开机启动

**systemctl enable openstack-ceilometer-api.service \**

**openstack-ceilometer-notification.service \**

**openstack-ceilometer-central.service \**

**openstack-ceilometer-collector.service**

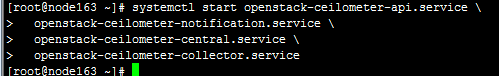
****

**systemctl start openstack-ceilometer-api.service \**

**openstack-ceilometer-notification.service \**

**openstack-ceilometer-central.service \**

**openstack-ceilometer-collector.service**

****

## 启用Image服务meters

1. 配置Image服务来使用Telemetry

* 修改配置：/etc/glance/glance-api.conf

**[DEFAULT]**

**...**

**rpc\_backend = rabbit**

**[oslo\_messaging\_notifications]**

**...**

**driver = messagingv2**

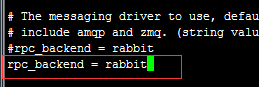
**[oslo\_messaging\_rabbit]**

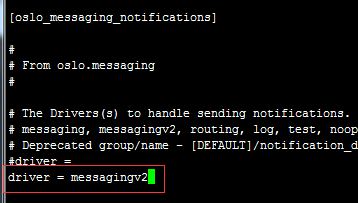
**...**

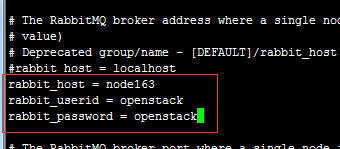
**rabbit\_host = node163**

**rabbit\_userid = openstack**

**rabbit\_password = openstack**







* 修改配置：/etc/glance/glance-registry.conf

**[DEFAULT]**

**...**

**rpc\_backend = rabbit**

**[oslo\_messaging\_notifications]**

**...**

**driver = messagingv2**

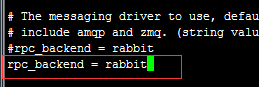
**[oslo\_messaging\_rabbit]**

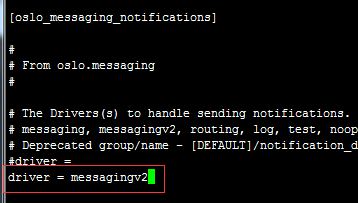
**...**

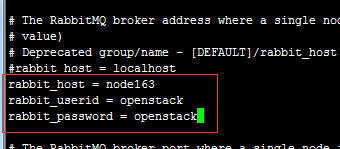
**rabbit\_host = node163**

**rabbit\_userid = openstack**

**rabbit\_password = openstack**







1. 重启Image服务

systemctl restart openstack-glance-api.service openstack-glance-registry.service



## 启用Compute服务meters

1. Yum安装相关组件

**yum install openstack-ceilometer-compute python-ceilometerclient python-pecan**

****

1. 修改配置：/etc/ceilometer/ceilometer.conf

* 配置RabbitMQ消息队列访问

**[DEFAULT]**

**...**

**rpc\_backend = rabbit**

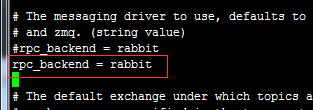
**[oslo\_messaging\_rabbit]**

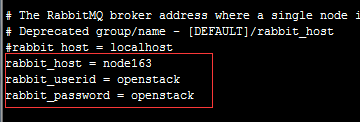
**...**

**rabbit\_host = node163**

**rabbit\_userid = openstack**

**rabbit\_password = openstack**

****

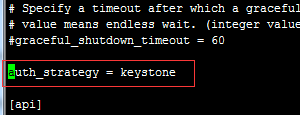
****

* 配置Identity服务访问

**[DEFAULT]**

**...**

**auth\_strategy = keystone**

****

**[keystone\_authtoken]**

**...**

**auth\_uri = http://node163:5000**

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

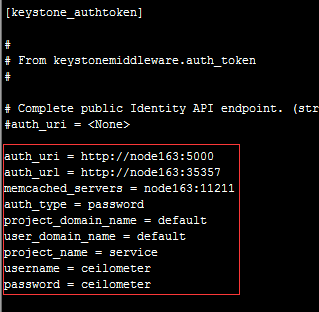
**project\_domain\_name = default**

**user\_domain\_name = default**

**project\_name = service**

**username = ceilometer**

**password = ceilometer**

****

* 配置服务凭证

**[service\_credentials]**

**...**

**os\_auth\_url = http://node163:5000/v2.0**

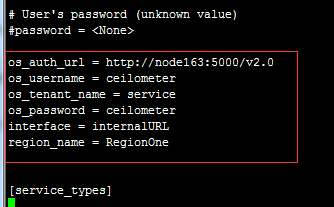
**os\_username = ceilometer**

**os\_tenant\_name = service**

**os\_password = ceilometer**

**interface = internalURL**

**region\_name = RegionOne**

****

1. 修改/etc/nova/nova.conf配置通知(**计算结点node49**)

**[DEFAULT]**

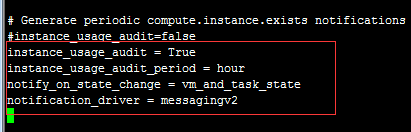
**...**

**instance\_usage\_audit = True**

**instance\_usage\_audit\_period = hour**

**notify\_on\_state\_change = vm\_and\_task\_state**

**notification\_driver = messagingv2**

****

1. 开启agent并设为开机启动

**systemctl enable openstack-ceilometer-compute.service**

****

**systemctl start openstack-ceilometer-compute.service**

****

1. 重启Compute服务（**计算结点node49**）

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

****

## 启用块存储meters

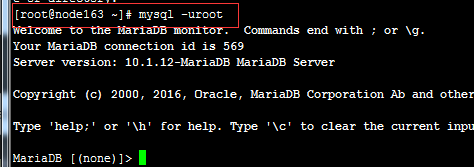
## 启用对象存储meters

## 报警服务

1. 创建数据库aodh

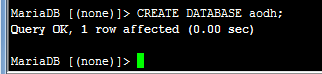
* Root用户登陆

**mysql –uroot**



* 创建aodh库

**CREATE DATABASE aodh;**

****

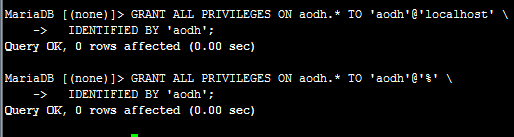
* 给aodh分配访问权限

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

**IDENTIFIED BY 'aodh';**

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

**IDENTIFIED BY 'aodh';**

****

1. 使用admin-openrc环境变量

**. admin-openrc**

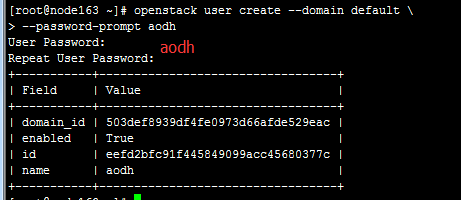
****

1. 创建服务凭证

* 创建aodh用户

**openstack user create --domain default \**

**--password-prompt aodh**

****

* 添加admin角色到aodh用户

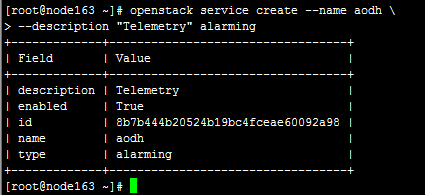
**openstack role add --project service --user aodh admin**

****

* 创建aodh服务实体

**openstack service create --name aodh \**

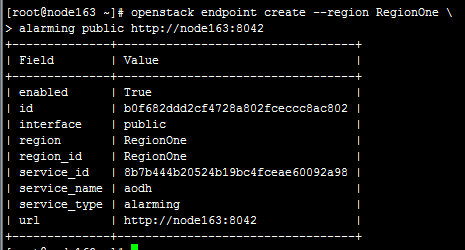
**--description "Telemetry" alarming**

****

1. 创建报警服务API端点

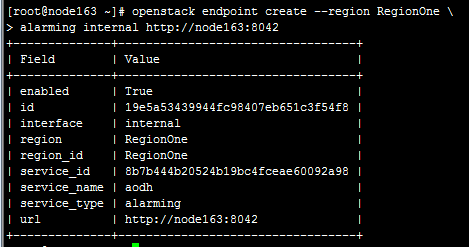
**openstack endpoint create --region RegionOne \**

**alarming public http://node163:8042**

****

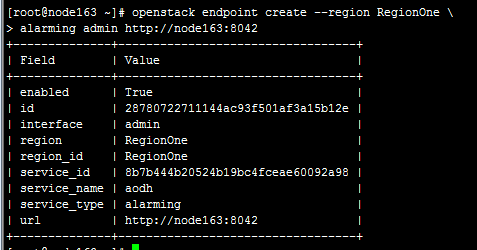
**openstack endpoint create --region RegionOne \**

**alarming internal http://node163:8042**

****

**openstack endpoint create --region RegionOne \**

**alarming admin** [**http://node163:8042**](http://node163:8042)

****

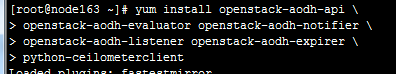
1. Yum安装组件

**yum install openstack-aodh-api \**

**openstack-aodh-evaluator openstack-aodh-notifier \**

**openstack-aodh-listener openstack-aodh-expirer \**

**python-ceilometerclient**

****

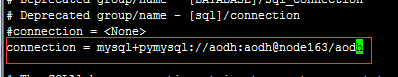
1. 修改配置：/etc/aodh/aodh.conf

* 配置数据库访问

**[database]**

**...**

**connection = mysql+pymysql://aodh:aodh@node163/aodh**

****

* 配置RabbitMQ消息队列访问

**[DEFAULT]**

**...**

**rpc\_backend = rabbit**

**[oslo\_messaging\_rabbit]**

**...**

**rabbit\_host = node163**

**rabbit\_userid = openstack**

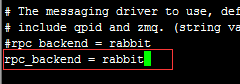
**rabbit\_password = openstack**

* 配置Identity服务访问

**[DEFAULT]**

**...**

**auth\_strategy = keystone**

****

**[keystone\_authtoken]**

**...**

**auth\_uri = http://node163:5000**

**auth\_url = http://node163:35357**

**memcached\_servers = node163:11211**

**auth\_type = password**

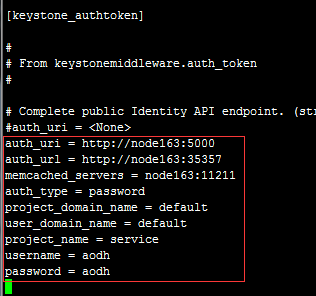
**project\_domain\_name = default**

**user\_domain\_name = default**

**project\_name = service**

**username = aodh**

**password = aodh**



* 配置服务凭证

**[service\_credentials]**

**...**

**os\_auth\_url = http://node163:5000/v2.0**

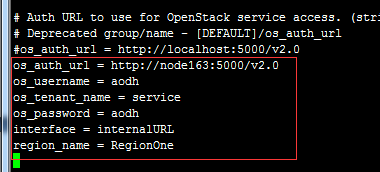
**os\_username = aodh**

**os\_tenant\_name = service**

**os\_password = aodh**

**interface = internalURL**

**region\_name = RegionOne**



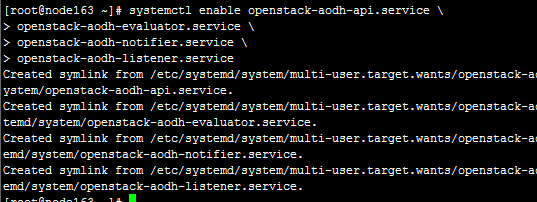
1. 开启报警服务并配置开机启动

**systemctl enable openstack-aodh-api.service \**

**openstack-aodh-evaluator.service \**

**openstack-aodh-notifier.service \**

**openstack-aodh-listener.service**

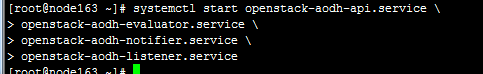
****

**systemctl start openstack-aodh-api.service \**

**openstack-aodh-evaluator.service \**

**openstack-aodh-notifier.service \**

**openstack-aodh-listener.service**

****

## 验证操作

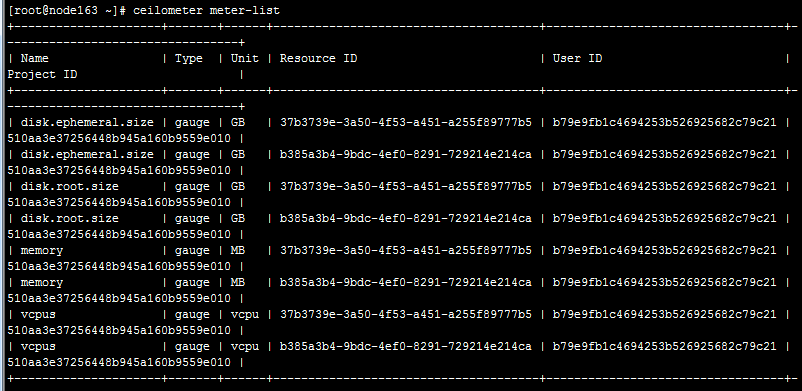
1. 使用admin的凭证进行下面命令操作

**. admin-openrc**

****

1. 展示可用的meters

**ceilometer meter-list**

****

1. 从Image服务下载cirros镜像

**IMAGE\_ID=$(glance image-list | grep 'cirros' | awk '{ print $2 }')**

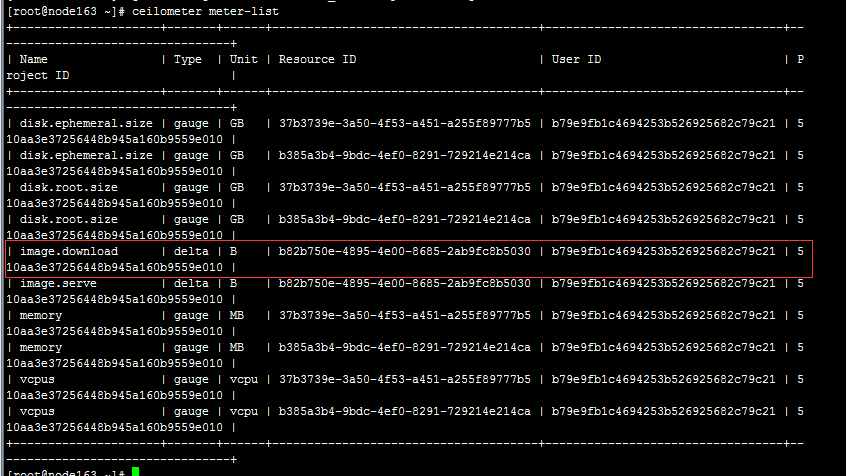
****

**glance image-download $IMAGE\_ID > /tmp/cirros.img**

****

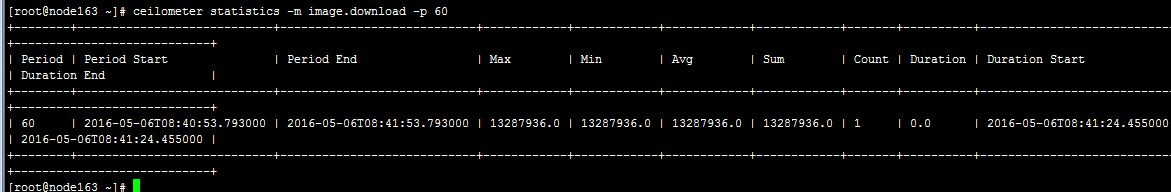
1. 再次展示可用的meters来校验检测镜像下载

**ceilometer meter-list**



1. 从image.download meters中检索使用统计数据

**ceilometer statistics -m image.download -p 60**

****

1. 移除之前下载的镜像文件 /tmp/cirros.img

**rm /tmp/cirros.img**

****

# Shared File Systems服务

## 安装配置 (控制节点 node163)

### 登录数据库

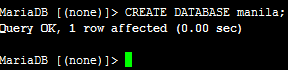
登录数据库，并授权

mysql -u root -p



创建数据库

CREATE DATABASE manila;

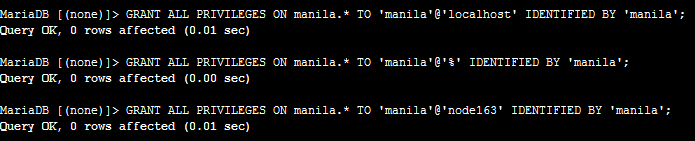


数据库授权

GRANT ALL PRIVILEGES ON manila.\* TO 'manila'@'localhost' IDENTIFIED BY 'manila';

GRANT ALL PRIVILEGES ON manila.\* TO 'manila'@'%' IDENTIFIED BY 'manila';

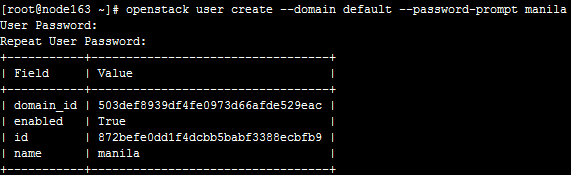
GRANT ALL PRIVILEGES ON manila.\* TO 'manila'@'node163' IDENTIFIED BY 'manila';



### 创建manila服务

创建manila用户，密码为manila

openstack user create --domain default --password-prompt manila

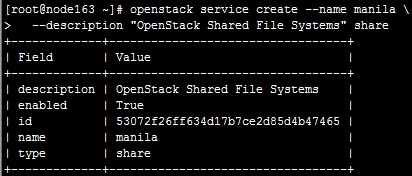
**

添加admin角色到manila用户

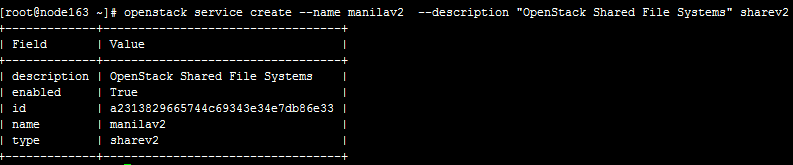


创建manila和manilav2服务实体

openstack service create --name manila --description "OpenStack Shared File Systems" share

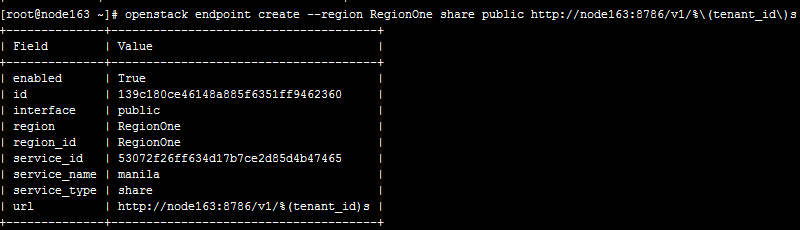


openstack service create --name manilav2 --description "OpenStack Shared File Systems" sharev2

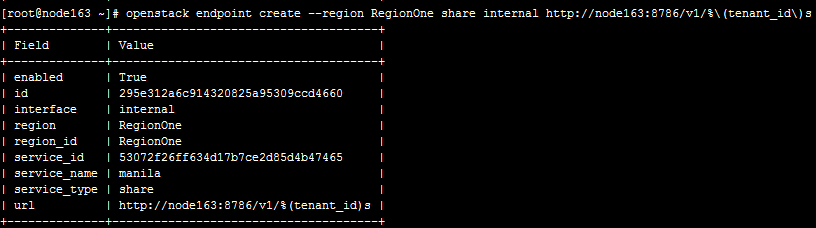


创建Shared File Systems服务的API终端:

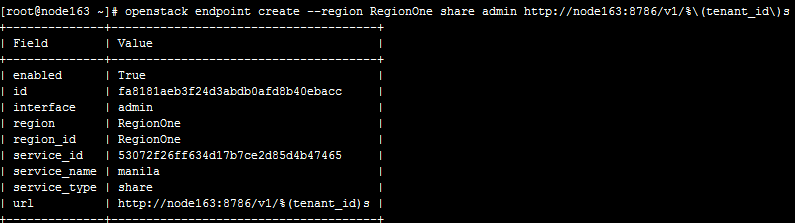
openstack endpoint create --region RegionOne share public http://**node163**:8786/v1/%\(tenant\_id\)s



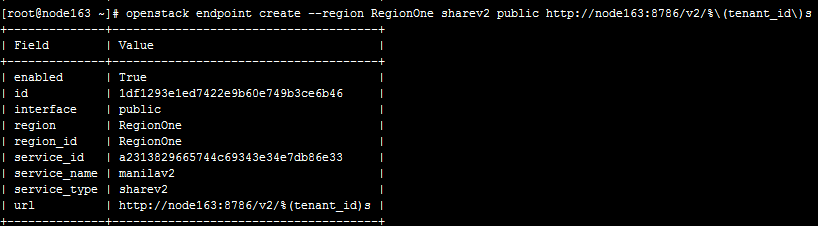
openstack endpoint create --region RegionOne share internal http://**node163**:8786/v1/%\(tenant\_id\)s



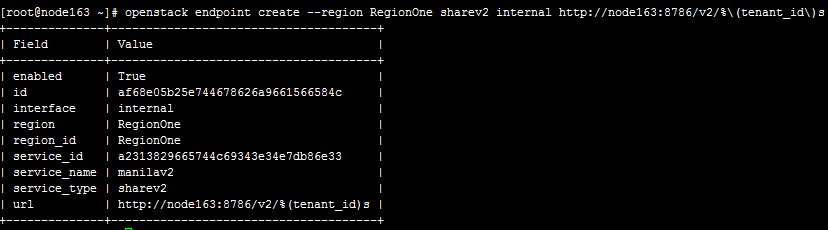
openstack endpoint create --region RegionOne share admin http://**node163**:8786/v1/%\(tenant\_id\)s



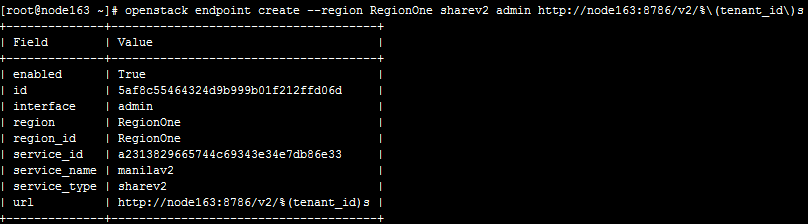
openstack endpoint create --region RegionOne sharev2 public http://**node163**:8786/v2/%\(tenant\_id\)s



openstack endpoint create --region RegionOne sharev2 internal http://**node163**:8786/v2/%\(tenant\_id\)s



openstack endpoint create --region RegionOne sharev2 admin [http://**node163**:8786/v2/%\(tenant\_id\)s](http://node163:8786/v2/%25\(tenant_id\)s)



### 安装配置组件

安装相关包

yum install openstack-manila python-manilaclient



编辑/etc/manila/manila.conf，添加如下配置

[database]

connection = mysql+pymysql://manila:manila@node163/manila

[DEFAULT]

rpc\_backend = rabbit

default\_share\_type = default\_share\_type

rootwrap\_config = /etc/manila/rootwrap.conf

auth\_strategy = keystone

my\_ip = 172.28.217.163

[oslo\_messaging\_rabbit]

rabbit\_host = node163

rabbit\_userid = openstack

rabbit\_password = openstack

[keystone\_authtoken]

memcached\_servers = node163:11211

auth\_uri = http://node163:5000

auth\_url = http://node163:35357

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = manila

password = manila

[oslo\_concurrency]

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

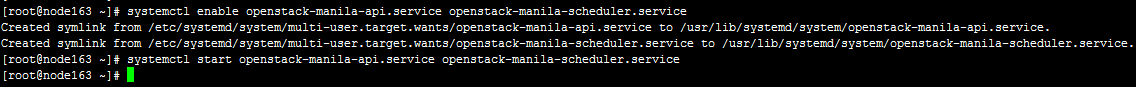
填充Share File System数据库

su -s /bin/sh -c "manila-manage db sync" manila

启动服务并设置开机启动

systemctl enable openstack-manila-api.service openstack-manila-scheduler.service

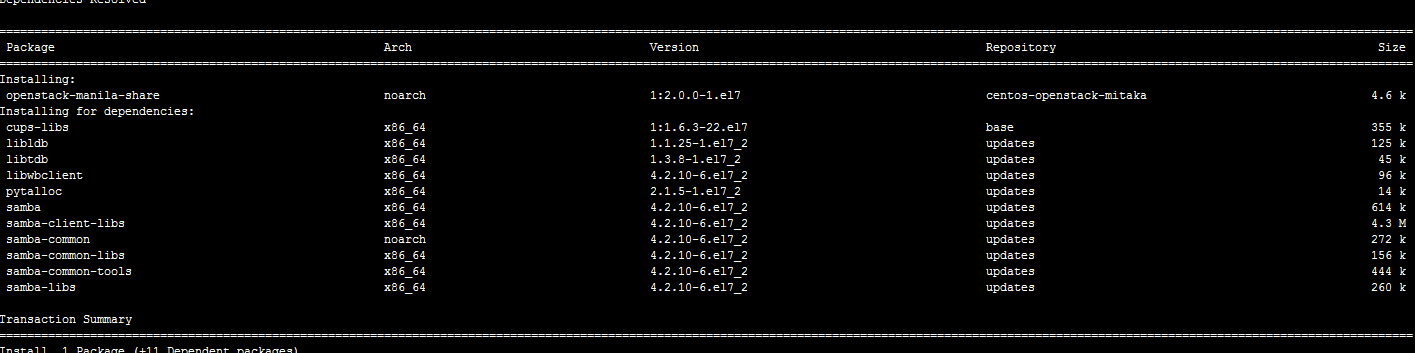
systemctl start openstack-manila-api.service openstack-manila-scheduler.service



## 安装配置share node (暂时放到控制节点 node163)

### 安装包

yum install openstack-manila-share python2-PyMySQL



### 编辑配置文件

编辑/etc/manila/manila.conf 添加以下内容:

[database]

connection = mysql://manila:manila@node163/manila

[DEFAULT]

rpc\_backend = rabbit

default\_share\_type = default\_share\_type

rootwrap\_config = /etc/manila/rootwrap.conf

auth\_strategy = keystone

my\_ip = 172.28.217.163

[oslo\_messaging\_rabbit]

rabbit\_host = node163

rabbit\_userid = openstack

rabbit\_password = openstack

[keystone\_authtoken]

memcached\_servers = node163:11211

auth\_uri = http://node163:5000

auth\_url = http://node163:35357

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = manila

password = manila

[oslo\_concurrency]

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

### 配置Driver support for share servers management

yum安装相关包

yum install openstack-neutron openstack-neutron-linuxbridge ebtables

编辑配置文件/etc/manila/manila.conf 添加如下配置

[DEFAULT]

enabled\_share\_backends = generic

enabled\_share\_protocols = NFS,CIFS

[neutron]

url = http://node163:9696

auth\_uri = http://node163:5000

auth\_url = http://node163:35357

memcached\_servers = node163:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

region\_name = RegionOne

project\_name = service

username = neutron

password = neutron

[nova]

auth\_uri = http://node163:5000

auth\_url = http://node163:35357

memcached\_servers = node163:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

region\_name = RegionOne

project\_name = service

username = nova

password = nova

[cinder]

auth\_uri = http://node163:5000

auth\_url = http://node163:35357

memcached\_servers = node163:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

region\_name = RegionOne

project\_name = service

username = cinder

password = cinder

[generic]

share\_backend\_name = GENERIC

share\_driver = manila.share.drivers.generic.GenericShareDriver

driver\_handles\_share\_servers = True

service\_instance\_flavor\_id = 100

service\_image\_name = manila-service-image

service\_instance\_user = manila

service\_instance\_password = manila

interface\_driver = manila.network.linux.interface.BridgeInterfaceDriver

### 启动manila-share服务

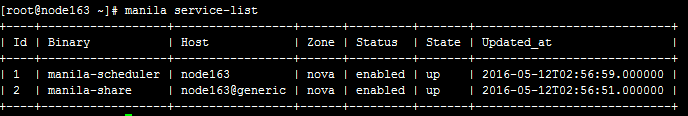
启动manila-share并配置开机启动

systemctl enable openstack-manila-share.service

systemctl start openstack-manila-share.service

### 验证是否成功

运行manila service-list



Scheduler和share服务均已启动