**OpenStack(Juno) 安装**

目录

[**1.** **安装环境** 1](#_Toc406508555)

[**2.** **网卡信息：** 1](#_Toc406508556)

[3. **总体架构** 2](#_Toc406508557)

[**4.** **NTP 安装** 3](#_Toc406508558)

[**5.** **系统更新** 4](#_Toc406508559)

[**6.** **安装认证服务** 5](#_Toc406508560)

[**7.** **安装image 服务** 9](#_Toc406508561)

[**8.** **安装计算服务** 10](#_Toc406508562)

[**9.** **配置网络节点(nova-network)** 14](#_Toc406508563)

[**10.** **配置dashboard** 14](#_Toc406508564)

[**11.** **添加块存储** 15](#_Toc406508565)

1. **安装环境**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 操作系统 | 节点类型 | 内存 | 磁盘 | 网卡eth0 ip | 网卡eth1 ip | 说明 |
| 1 | Ubuntu14.04 | Controller | 1600M | 30GB | 192.168.244.200 | 不需要网卡 | 网卡1桥接 |
| 2 | Ubuntu14.04 | computer | 3000M | 45G+8G | 192.168.244.201 | 无ip 地址 | 网卡2桥接 |
| 3 | Virtual box4.3 |  |  |  |  |  |  |

1. **网卡信息：**

控制节点

192.168.244.200

vim /etc/network/interfaces

# interfaces(5) file used by ifup(8) and ifdown(8)

auto lo

iface lo inet loopback

auto eth0

iface eth0 inet static

address 192.168.244.200

netmask 255.255.255.0

gateway 192.168.244.254

dns-nameservers 202.106.0.20 202.106.46.151

主机名信息：

cat /etc/hosts

192.168.244.200 controller

192.168.244.201 computer

计算节点

192.168.244.201

vim /etc/network/interfaces

# interfaces(5) file used by ifup(8) and ifdown(8)

auto lo

iface lo inet loopback

auto eth0

iface eth0 inet static

address 192.168.244.201

netmask 255.255.255.0

gateway 192.168.244.254

dns-nameservers 202.106.0.20 202.106.46.151

主机名字信息：

cat /etc/hosts

192.168.244.200 controller

192.168.244.201 computer

# 3. 总体架构



1. **NTP 安装**
   1. **apt-get install ntp**
   2. **修改配置文件 /etc/ntp.conf**

server *ntp.ubuntu.com* iburst

restrict -4 default kod notrap nomodify

restrict -6 default kod notrap nomodify

删除/var/lib/ntp/ntp.conf.dhcp 文件， 如果此文件存在的话

**service ntp restart**

* 1. **其他节点**

**apt-get install ntp**

**修改/etc/ntp.conf**

server *controller* iburst

删除/var/lib/ntp/ntp.conf.dhcp 文件， 如果此文件存在的话

**service ntp restart**

* 1. **验证**

**在控制节点运行 命令： ntpq -c peers**

remote refid st t when poll reach delay offset

jitter

==========================================================================

====

\*ntp-server1 192.0.2.11 2 u 169 1024 377 1.901 -0.611

5.483

+ntp-server2 192.0.2.12 2 u 887 1024 377 0.922 -0.246

在控制节点运行：**ntpq -c assoc**

# **ntpq -c assoc**

ind assid status conf reach auth condition last\_event cnt

===========================================================

1 20487 961a yes yes none sys.peer sys\_peer 1

2 20488 941a yes yes none candidate sys\_peer 1

运行下面命令在其他节点

# **ntpq -c peers**

remote refid st t when poll reach delay offset

jitter

==========================================================================

====

\*controller 192.0.2.21 3 u 47 64 37 0.308 -0.251

0.079

# **ntpq -c assoc**

ind assid status conf reach auth condition last\_event cnt

===========================================================

1 21181 963a yes yes none sys.peer sys\_peer 3

1. **系统更新**

**apt-get install python-software-properties**

apt-add-repository ppa:ubuntu-cloud-archive/juno-staging

**apt-get update && apt-get dist-upgrade**

在控制节点安装数据库

**apt-get –y install mariadb-server python-mysqldb**

**修改数据库配置文件**

**/etc/mysql/my.cnf**

[mysqld]

bind-address = 0.0.0.0

default-storage-engine = innodb

innodb\_file\_per\_table

collation-server = utf8\_general\_ci

init-connect = 'SET NAMES utf8'

character-set-server = utf8

重新启动mysql 服务

**service mysql restart**

**配置mysql 密码**

**mysql\_secure\_installation**

**在控制节点安装消息队列**

**apt-get –y install rabbitmq-server**

**修改消息队列guest 密码**

**rabbitmqctl change\_password guest 123456**

**启动rabbitmq WEB管理终端**

**rabbitmq-plugins enable rabbitmq\_management**

**service rabbitmq-server restart**

**登录web 管理：**

[**http://192.168.244.200:55672**](http://192.168.244.200:55672)

**用户名： guest**

**密码： 123456**

1. **安装认证服务**

KeyStone 图解



建立keystone 数据库

mysql -uroot -p

**CREATE DATABASE keystone;**

**授权：**

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

**IDENTIFIED BY '123456';**

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

**IDENTIFIED BY '123456';**

**产生token**

# **openssl rand -hex 10**

**安装配置组件**

# **apt-get install keystone python-keystoneclient**

**修改**/etc/keystone/keystone.conf

[DEFAULT]

admin\_token = *ADMIN\_TOKEN* # ADMIN\_TOKEN 修改为你生成的token

verbose = True

[database]

connection = mysql://keystone:123456@*controller*/keystone

[token]

provider = keystone.token.providers.uuid.Provider

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

产生认证数据库,表信息

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

重启keystone 服务

**service keystone restart**

删除SQLlite db

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

**添加任务计划**

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

**echo '@hourly /usr/bin/keystone-manage token\_flush >/var/log/keystone/**

**keystone-tokenflush.log 2>&1' \**

**>> /var/spool/cron/crontabs/keystone**

**配置环境变量：**

**export OS\_SERVICE\_TOKEN=294a4c8a8a475f9b9836**

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

**创建租户， 用户， 角色**

1. **创建admin 租户**

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

1. **创建用户**

**keystone user-create --name admin --pass 123456--**

**email  *[admin@email.com](mailto:admin@email.com)***

1. 创建角色

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

d．添加admin 租户， 用户到admin 角色

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

**e． 创建member 角色 ， 限制访问dashboard**

**keystone role-create --name \_member\_**

**f. 添加admin 租户和用户到member 角色**

**keystone user-role-add --tenant admin --user admin --role \_member\_**

**创建demo 用户， 租户等**

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

**keystone user-create --name demo --pass 123456--**

**email** [***demo@email.com***](mailto:demo@email.com)

**keystone user-role-add --tenant demo --user demo --role \_member\_**

**创建服务租户**

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

**创建服务条目和endpoint**

**为认证服务 创建服务条目**

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

**--description "OpenStack Identity"**

**创建API endpoint 为认证服务**

**keystone endpoint-create \**

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

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

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

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

**--region regionOne**

**验证操作**

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

**获取token**

**keystone --os-tenant-name admin --os-username admin –-os-password**

***123456* \**

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

**获取租户**

**keystone --os-tenant-name admin --os-username admin --os-password**

***123456* \**

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

获取用户

**keystone --os-tenant-name admin --os-username admin --os-password**

***123456* \**

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

**获取角色**

**keystone --os-tenant-name admin --os-username admin --os-password**

***123456* \**

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

**创建OpenStack客户端环境脚本 admin-openrc.sh, 内容如下**

export OS\_TENANT\_NAME=admin

export OS\_USERNAME=admin

export OS\_PASSWORD=123456

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

创建demo 租户脚本 demo-openrc.sh

export OS\_TENANT\_NAME=demo

export OS\_USERNAME=demo

export OS\_PASSWORD=*123456*

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

加载环境脚本

**source admin-openrc.sh**

1. **安装image 服务**
   1. **配置数据库**

**mysql –uroot –p**

**CREATE DATABASE glance;**

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

**IDENTIFIED BY '*123456*';**

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

**IDENTIFIED BY '*123456*';**

* 1. **创建glance 账户， 角色， 服务，租户**

**source admin-openrc.sh**

**keystone user-create --name glance --pass *123456***

**keystone user-role-add --user glance --tenant service --role admin**

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

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

**7.3 创建ednpoint**

**keystone endpoint-create \**

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

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

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

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

**--region regionOne**

**7.4 安装配置image配置服务组件**

**apt-get install glance python-glanceclient**

**修改** /etc/glance/glance-api.conf

[database]

connection = mysql://glance:123456@*controller*/glance

[keystone\_authtoken]

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

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

admin\_tenant\_name = service

admin\_user = glance

admin\_password = 123456

[paste\_deploy]

flavor = keystone

[glance\_store]

default\_store = file

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

[DEFAULT]

verbose = True

修改 /etc/glance/glance-registry.conf

[database]

connection = mysql://glance:123456@*controller*/glance

[keystone\_authtoken]

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

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

admin\_tenant\_name = service

admin\_user = glance

admin\_password = 123456

[paste\_deploy]

flavor = keystone

[DEFAULT]

verbose = True

生成image 服务数据库

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

**重启image 服务**

# **service glance-registry restart**

# **service glance-api restart**

**删除 sqllite 文件**

# **rm -f /var/lib/glance/glance.sqlite**

**验证操作**

**wget http://download.cirros-cloud.net/0.3.3/cirros-0.3.3-x86\_64-**

**disk.img**

**source admin-openrc.sh**

**glance image-create --name "cirros-0.3.3-x86\_64" --file cirros-0.3.3-**

**x86\_64-disk.img \**

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

**获取镜像列表**

**glance image-list**

1. **安装计算服务**
   1. **在控制节点安装**

**本节点以FlatDHCPManager 方式安装**

1. **配置mysql**

**mysql -u root –p**

**CREATE DATABASE nova;**

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

**IDENTIFIED BY '123456';**

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

**IDENTIFIED BY '123456';**

1. **配置租户，用户， endpoint等**

**source admin-openrc.sh**

**keystone user-create --name nova --pass *123456***

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

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

**--description "OpenStack Compute"**

**keystone endpoint-create \**

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

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

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

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

**--region regionOne**

1. **安装配置控制节点组件**

**apt-get install nova-api nova-cert nova-conductor nova-consoleauth \**

**nova-novncproxy nova-scheduler python-novaclient nova-network**

**d． 修改配置文件** /etc/nova/nova.conf

[database]

connection = mysql://nova:123456@controller/nova

[DEFAULT]

rpc\_backend = rabbit

rabbit\_host = *controller*

rabbit\_password = *123456*

[DEFAULT]

auth\_strategy = keystone

my\_ip = 192.168.244.200

vncserver\_listen = 192.168.244.200

vncserver\_proxyclient\_address = 192.168.244.200

verbose = True

[keystone\_authtoken]

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

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

admin\_tenant\_name = service

admin\_user = nova

admin\_password = *123456*

[glance]

host = *controller*

*e．* 生成compute 数据库

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

**f. 重启服务**

# **service nova-api restart**

# **service nova-cert restart**

# **service nova-consoleauth restart**

# **service nova-scheduler restart**

# **service nova-conductor restart**

# **service nova-novncproxy restart**

**# service nova-network restart**

# **rm -f /var/lib/nova/nova.sqlite**

**备注： 由于controller 节点没有安装nova-network 服务， 所以在horizon 界面看到的服务**

**有状态时停止的。 安装nova-network 服务后， 就处于运行状态**

* 1. **安装配置计算节点**

# **apt-get install nova-compute sysfsutils**

1. **修改：**/etc/nova/nova.conf

**root@computer-VirtualBox:~# cat /etc/nova/nova.conf**

**[DEFAULT]**

**dhcpbridge\_flagfile=/etc/nova/nova.conf**

**dhcpbridge=/usr/bin/nova-dhcpbridge**

**logdir=/var/log/nova**

**state\_path=/var/lib/nova**

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

**force\_dhcp\_release=True**

**libvirt\_use\_virtio\_for\_bridges=True**

**verbose=True**

**ec2\_private\_dns\_show\_ip=True**

**api\_paste\_config=/etc/nova/api-paste.ini**

**enabled\_apis=ec2,osapi\_compute,metadata**

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

**security\_group\_api = nova**

**firewall\_driver = nova.virt.libvirt.firewall.IptablesFirewallDriver**

**network\_manager = nova.network.manager.FlatDHCPManager**

**network\_size = 254**

**#flat\_network\_dhcp\_start=10.108.110.20**

**#flat\_network\_dhcp\_start= 10.0.0.100**

**allow\_same\_net\_traffic = False**

**multi\_host = True**

**send\_arp\_for\_ha = True**

**share\_dhcp\_address = True**

**force\_dhcp\_release = True**

**flat\_network\_bridge = br100**

**flxed\_range = 192.168.233.0/24**

**# available ip: 192.168.244.65~192.168.244.94**

**#floating\_range=192.168.244.64/27**

**flat\_interface = eth1**

**public\_interface = eth0**

**start\_guests\_on\_host\_boot=true**

**resume\_guests\_state\_on\_host\_boot=true**

**rpc\_backend = rabbit**

**rabbit\_host = controller**

**rabbit\_password = 123456**

**auth\_strategy = keystone**

**my\_ip = 192.168.244.201**

**vnc\_enabled = True**

**vncserver\_listen = 0.0.0.0**

**vncserver\_proxyclient\_address = 192.168.244.201**

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

**[keystone\_authtoken]**

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

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

**admin\_tenant\_name = service**

**admin\_user = nova**

**admin\_password = 123456**

**[glance]**

**host = controller**

**验证：是否支持虚拟化**

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

如果结果是0， 则不支持kvm， 需要qemu

修改/etc/nova/nova-compute.conf

[libvirt]

virt\_type = qemu

重启计算服务

# **service nova-compute restart**

# **rm -f /var/lib/nova/nova.sqlite**

**B．验证操作**

**source admin-openrc.sh**

**nova service-list**

**nova image-list**

1. **配置网络节点(nova-network)**

**由于neutron 复杂， 没有配置**

1. **在控制节点**

**修改** /etc/nova/nova.conf

[DEFAULT]

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

security\_group\_api = nova

**重启服务**

# **service nova-api restart**

# **service nova-scheduler restart**

# **service nova-conductor restart**

**b． 在计算节点**

# **apt-get install nova-network nova-api-metadata**

**修改/etc/nova/nova.conf 见compute 节点nova.conf 配置**

**重启服务**

# **service nova-network restart**

# **service nova-api-metadata restart**

**创建网络**

**source admin-openrc.sh**

**nova network-create demo-net --bridge br100 --multi-host T \**

**--fixed-range-v4 *192.168.233.0/24***

**nova net-list**

1. **配置dashboard**
2. **在控制节点**

**apt-get install openstack-dashboard apache2 libapache2-mod-wsgi**

**memcached python-memcache**

**b． 修改配置文件** /etc/openstack-dashboard/local\_settings.py

OPENSTACK\_HOST = "*controller*"

ALLOWED\_HOSTS = ['\*']

CACHES = {

'default': {

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

MemcachedCache',

'LOCATION': '127.0.0.1:11211',

}

}

TIME\_ZONE = "*CN*"

**C．重启服务**

# **service apache2 restart**

# **service memcached restart**

**d． 验证**

<http://192.168.244.200/horizon>

用户名:admin

密码： 123456

E．创建floating ip

nova-manage floating create 192.168.244.64/27

1. **添加块存储**
2. **在控制节点**

**数据库操作**

$ **mysql -u root –p**

**CREATE DATABASE cinder;**

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

**IDENTIFIED BY '123456';**

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

**IDENTIFIED BY '123456';**

**租户，服务等**

**source admin-openrc.sh**

**keystone user-create --name cinder --pass *123456***

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

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

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

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

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

**keystone endpoint-create \**

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

**\**

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

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

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

**--region regionOne**

**keystone endpoint-create \**

**--service-id $(keystone service-list | awk '/ volumev2 / {print**

**$2}') \**

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

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

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

**--region regionOne**

**安装组件**

**apt-get install cinder-api cinder-scheduler python-cinderclient**

**配置** /etc/cinder/cinder.conf

**[DEFAULT]**

**rootwrap\_config = /etc/cinder/rootwrap.conf**

**api\_paste\_confg = /etc/cinder/api-paste.ini**

**iscsi\_helper = tgtadm**

**volume\_name\_template = volume-%s**

**volume\_group = cinder-volumes**

**verbose = True**

**auth\_strategy = keystone**

**state\_path = /var/lib/cinder**

**lock\_path = /var/lock/cinder**

**volumes\_dir = /var/lib/cinder/volumes**

**my\_ip = 192.168.244.200**

**rpc\_backend = rabbit**

**rabbit\_host = controller**

**rabbit\_password = 123456**

**[database]**

**connection = mysql://cinder:123456@controller/cinder**

**[keystone\_authtoken]**

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

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

**admin\_tenant\_name = service**

**admin\_user = cinder**

**admin\_password = 123456**

**产生storage 数据库**

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

**重启服务**

# **service cinder-scheduler restart**

# **service cinder-api restart**

# **rm -f /var/lib/cinder/cinder.sqlite**

1. **在computer 上配置存储节点**

**apt-get install lvm2**

**fdisk /dev/sdb**

**t 选择类型（lvm）**

**pvcreate /dev/sdb1**

**vgcreate cinder-volumes /dev/sdb1**

**安装块存储组件**

**apt-get install cinder-volume python-mysqldb**

**修改**/etc/cinder/cinder.conf

**[DEFAULT]**

**rootwrap\_config = /etc/cinder/rootwrap.conf**

**api\_paste\_confg = /etc/cinder/api-paste.ini**

**iscsi\_helper = tgtadm**

**volume\_name\_template = volume-%s**

**volume\_group = cinder-volumes**

**verbose = True**

**auth\_strategy = keystone**

**state\_path = /var/lib/cinder**

**lock\_path = /var/lock/cinder**

**volumes\_dir = /var/lib/cinder/volumes**

**my\_ip = 192.168.244.201**

**glance\_host = controller**

**rpc\_backend = rabbit**

**rabbit\_host = controller**

**rabbit\_password = 123456**

**[database]**

**connection = mysql://cinder:123456@controller/cinder**

**[keystone\_authtoken]**

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

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

**admin\_tenant\_name = service**

**admin\_user = cinder**

**admin\_password = 123456**

**重启服务**

# **service tgt restart**

# **service cinder-volume restart**

# **rm -f /var/lib/cinder/cinder.sqlite**

**验证操作**

**source admin-openrc.sh**

**cinder service-list**

**source demo-openrc.sh**

**创建1G卷， 名字是demo-volume1**

**cinder create --display-name demo-volume1 1**

**cinder list**