# Openstack平台搭建

（注2：建议配置过程中要设置的一切密码都设置成 1234 ）

目录

[Openstack平台搭建——私有网版 1](#_Toc526883784)

[**1 组网** 3](#_Toc526883785)

[**2 openstack软件环境配置** 8](#_Toc526883786)

[**2.1** **主机网络配置** 8](#_Toc526883787)

[2.1.1 控制节点网络配置 8](#_Toc526883788)

[2.1.2 计算节点网络配置 8](#_Toc526883789)

[2.1.3 额外配置 8](#_Toc526883790)

[2.1.4 连通性测试 9](#_Toc526883791)

[**2.2** **各节点时间同步配置** 9](#_Toc526883792)

[2.2.1 控制节点 9](#_Toc526883793)

[2.2.2 计算节点 9](#_Toc526883794)

[2.2.3 时间同步验证 10](#_Toc526883795)

[**2.3** **添加openstack软件源** 10](#_Toc526883796)

[**2.4** **安装mysql数据库** 10](#_Toc526883797)

[**2.5** **消息队列服务安装** 11](#_Toc526883798)

[**2.6** **安装Memcached** 11](#_Toc526883799)

[**3 认证服务——keystone组件安装** 12](#_Toc526883800)

[**3.1** **创建keystone数据库** 12](#_Toc526883801)

[**3.2** **keystone服务安装与配置** 12](#_Toc526883802)

[**3.3** **apache服务器配置** 13](#_Toc526883803)

[**3.4** **创建服务实体和API端点** 14](#_Toc526883804)

[**3.5** **创建域、项目、用户和角色** 15](#_Toc526883805)

[**3.6** **验证操作** 16](#_Toc526883806)

[**3.7** **创建脚本** 16](#_Toc526883807)

[**4 镜像服务——glance组件安装** 18](#_Toc526883808)

[**4.1** **创建glance数据库** 18](#_Toc526883809)

[**4.2** **创建glance服务证书、服务实体和服务端点** 18](#_Toc526883810)

[**4.3** **安装并配置组件** 19](#_Toc526883811)

[**4.4** **验证操作** 20](#_Toc526883812)

[**5 计算服务——nova组件安装** 22](#_Toc526883813)

[**5.1** **创建navo、nova\_api数据库** 22](#_Toc526883814)

[**5.2** **创建服务证书、服务实体和服务API端点** 22](#_Toc526883815)

[**5.3** **安装配置组件** 23](#_Toc526883816)

[**5.3** **安装配置计算节点** 25](#_Toc526883817)

[**5.3** **验证操作** 26](#_Toc526883818)

[**6 网络服务——neutron组件安装** 28](#_Toc526883819)

[**6.1** **创建neutron数据库** 28](#_Toc526883820)

[**6.2创建服务证书和服务API端点** 28](#_Toc526883821)

[**6.3 控制节点安装与配置** 29](#_Toc526883822)

[**6.4 计算节点安装与配置** 32](#_Toc526883823)

[**6.5 验证操作** 33](#_Toc526883824)

[**7 图形界面服务——horizon组件安装** 34](#_Toc526883825)

[**7.1 安装与配置** 34](#_Toc526883826)

[**7.2 验证操作** 36](#_Toc526883827)

[**7.3 可能出现的错误及解决方法** 36](#_Toc526883828)

[**8 创建实例** 36](#_Toc526883829)

[**8.1** **创建提供者网络** 36](#_Toc526883830)

[**8.2** **创建实例** 37](#_Toc526883831)

[**8.3** **访问实例** 38](#_Toc526883832)

[**8.4** **添加私有网络** 38](#_Toc526883833)

[**8.5** **添加虚拟路由器** 41](#_Toc526883834)

**1 组网**

本章给出物理环境下和VMware虚拟机环境下的组网方式。

**1.1 物理环境组网方式**

如图1.1官方文档给出的各节点网络拓扑图所示，右边虚框表示的节点为可选节点，而左边的节点为必要节点。因此，此次部署只部署一个控制节点，和一个计算节点。

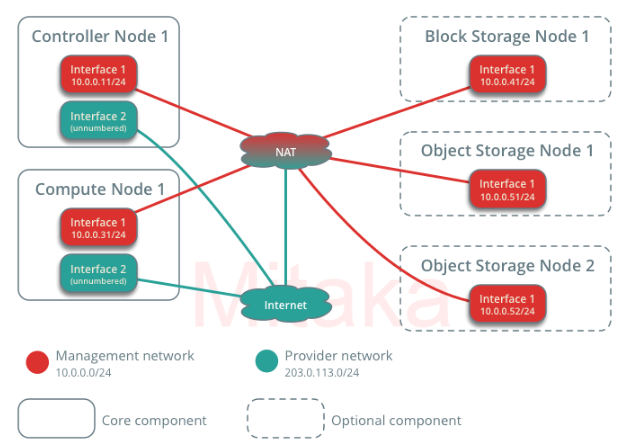


图1.1 官方文档组网图

综上所述，此次部署需要的硬件设备有：

1、 电脑两台：两个网卡，ubuntu16.04系统。最低要求，控制节点内存3G，计算节点内存1G。

2、 路由器两个

3、 网线若干

使用以上设备进行组网如图1.2所示。本文约定，使用eth0表示主机第一个网卡的名称，eth1表示第二张网卡的名称，网卡的实际名称可以在控制台通过ifconfig命令查看。

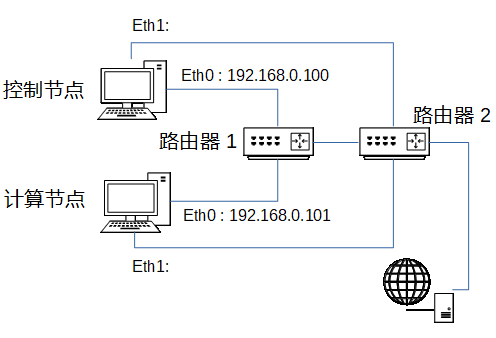


图1.2 实现组网图

图中控制节点与计算节点的第一个网卡分别接入路由器1，为了固定这两个网卡的ip需要配置为静态ip，也可以在路由器1中设置ip与主机MAC地址绑定。此次部署采取第二种方式，至于静态ip配置可自行百度。首先登录路由器1的管理页面，即在浏览器地址栏输入192.168.0.1，登录页面后，找到可以配置ip绑定的页面，设置ip绑定如图1.3所示。

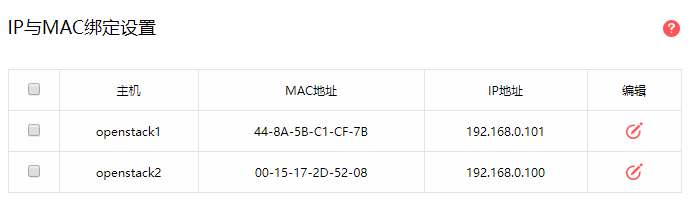


图1.3 路由器绑定IP

路由器2为两台主机分配的子网为192.168.1.0/24。将两个主机的第二个网卡接到路由器2，无需配置ip。

**1.2 VMware虚拟机环境组网方式**

使用VMware虚拟机组网图如图1.4所示。需要两个虚拟机，为每个虚拟机添加两个网卡。控制节点内存3G，计算节点内存1G，作者使用的是无图形界面的，若使用有图形界面的虚拟机，可以要适当调大内存（推荐各上调1G）。

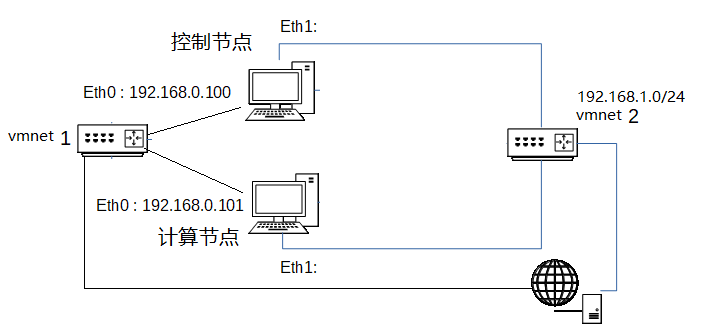


图1.4 vmware虚拟机组网图

组网过程如下：

1. 生成两台虚拟机，请使用ubuntu16.04版本系统。
2. 按以下步骤配置VMware虚拟网络：

I、点击 “编辑” –》 “虚拟网络编辑器”

可看到当前有的两个虚拟网络如图



II、点击“更改设置”，在弹出的界面中选中NAT模式的虚拟网卡，如下图中选择



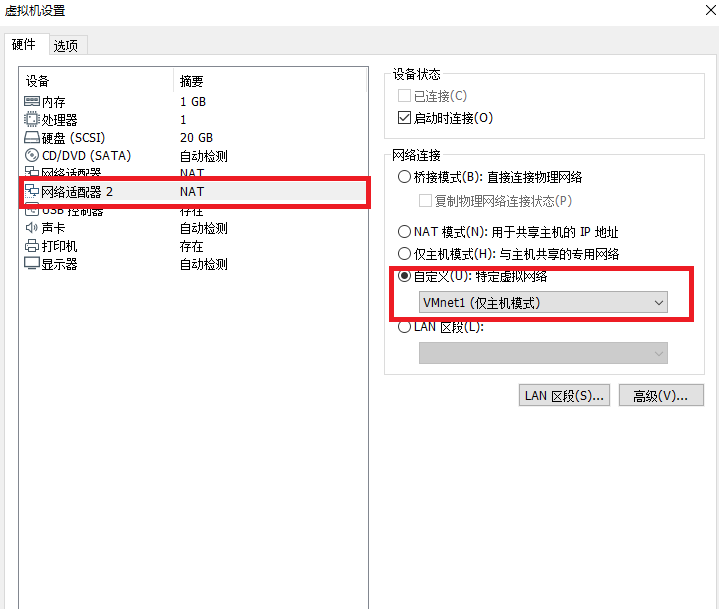
取消红框里的对勾。

1. 为作为控制节点的虚拟机设置3G内存，作为计算节点的设置1G内存：

选择如图所示“编辑虚拟机设置”，在打开页面中设置内存

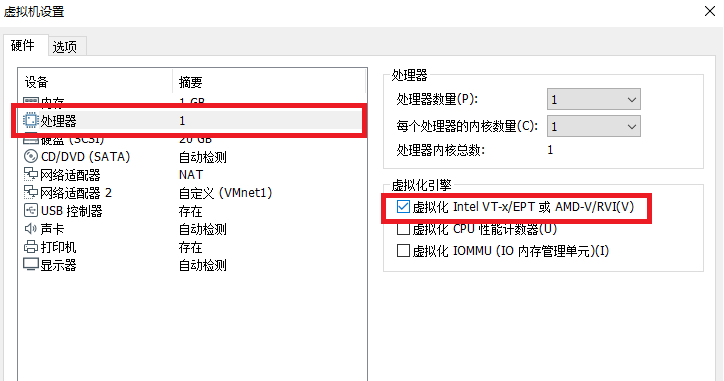


1. 为两台虚拟机添加第二个网卡，在两台虚拟机设置界面完成如下操作：
2. 点击“编辑虚拟机设置”
3. 点击“添加”－》 “网络适配器” －》 “完成”
4. 左边选择“网络适配器2” –》 右边选择“自定义(U):特定虚拟网络”并选择一个仅主机模式的网络，如下图选择vmnet1



1. 同样在虚拟机设置界面，为两台虚拟机配置如下

选择“处理器”－》 虚拟机 Intel VT-x/EPT 或AMD-V/RVI（V）



1. 启动两台虚拟机，并为虚拟机配置静态ip

编辑 ``/etc/network/interfaces`` 文件 ， 更改设置如下图所示

auto ens33

iface ens33 inet static

address 192.168.1.112

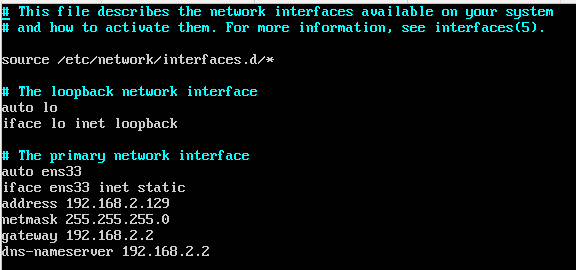
netmask 255.255.255.0

gateway 192.168.1.2

dns-nameserver 192.168.1.2

其中ens33是第一块网卡的名称，静态ip、网关请根据虚拟网络，具体情况具体分析。（提示：通过 ip link 命令可以查看所有网卡名称）。

保存文件，重启主机，用ifconfig查看是否生效。



**2 openstack软件环境配置**

对两台主机都执行以下命令

(1) 进入超级用户状态

$ sudo –s

**2.1 主机网络配置**

2.1.1 控制节点网络配置

(1) 编辑``/etc/network/interfaces``文件

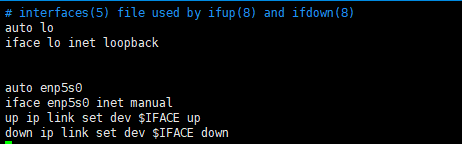
添加内容

auto eth1

iface eth1 inet manual

up ip link set dev $IFACE up

down ip link set dev $IFACE down



即对第二块网卡（eth1）进行配置。保存文件后，重启系统以激活修改。

(2) 编辑``/etc/hosts``文件

添加以下内容后保存文件

192.168.0.100 controller

192.168.0.101 compute

2.1.2 计算节点网络配置

计算节点配置与控制节点相同，需要注意的是主机的第二张网卡名称在实际中也许不同。

2.1.3 额外配置

计算如果控制节点与计算节点的主机名相同，需修改其中一个。修改主机名称的方法：

（1） 编辑`` /etc/hostname``文件

文件内容就是主机名，修改该文件内容，保存退出。（假如原文件内容为ubuntu，可将其改为openstack1）

（2） 编辑`` /etc/hosts``文件

将该文件中的原主机名改成与/etc/hostname文件内容一致。

（3）重启系统，使配置生效

2.1.4 连通性测试

（1） 在控制节点上执行

# ping openstack.org

# ping compute

（2） 在控制节点上执行

# ping openstack.org

# ping controller

保证4条ping命令都能ping通。

**2.2 各节点时间同步配置**

2.2.1 控制节点

在控制节点上执行以下命令。

1. 安装chrony

# apt install chrony

1. 编辑`` /etc/chrony/chrony.conf``文件

添加以下两行

allow 192.168.0.0/24

server ntp1.aliyun.com iburst

注： I、第一行表示允许192.168.0.0/24该子网内的主机访问本主机做时间同步。

II、第二行表示添加 ntp1.aliyun.com（阿里云服务器） 这个服务器作为时间同步服务器。

1. 重启chrony服务

# service chrony restart

2.2.2 计算节点

在计算结点上执行以下命令。

1. 安装chrony

# apt install chrony

1. 编辑`` /etc/chrony/chrony.conf``文件

注释或删除该文件中的所有行。

添加以下一行，使用控制节点为时间同步服务器。

server controller iburst

1. 重启chrony服务

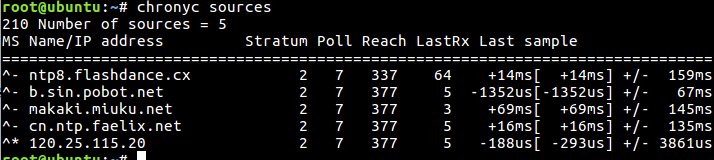
# service chrony restart

2.2.3 时间同步验证

（1）在控制节点上执行

# chronyc sources

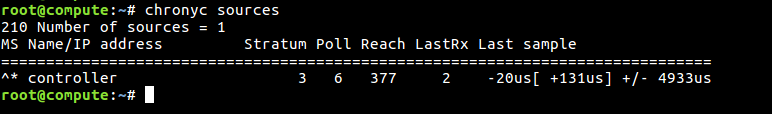
应出现如图所示。在120.25.115.20（这是ntp1.aliyun.com的ip地址）一行前应出现\*号，表示时间同步成功。



（2）在控制节点上执行

# chronyc sources

应出现如图所示。controller一行前有\*号，表示与控制节点时间同步成功。



（3） 如果没看到有\*号，请等待一会再查看，因为时间同步需要一小段时间。另外也可反复在各节点上执行service chrony restart命令。

**2.3 添加openstack软件源**

在所有节点执行以下命令。

（1） 添加软件源

# apt-get install software-properties-common

# add-apt-repository cloud-archive:newton

这里安装newton版本的openstack

（2）升级软件包

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

（3）重启系统

（4）安装 OpenStack 客户端：

# apt-get install python-openstackclient

**2.4 安装mysql数据库**

在控制节点执行以下命令：

（1）安装软件

# apt-get install mariadb-server python-pymysql

（2）创建并编辑文件``/etc/mysql/mariadb.conf.d/99-openstack.cnf``

写入以下内容。

[mysqld]

bind-address = 192.168.0.100

default-storage-engine = innodb

innodb\_file\_per\_table

max\_connections = 4096

collation-server = utf8\_general\_ci

character-set-server = utf8

其中192.168.0.100为控制节点在管理网络中的ip地址，请参考图1.2。

（3） 重启数据库服务：

# service mysql restart

1. 执行 mysql\_secure\_installation 脚本来对数据库进行安全加固。

# mysql\_secure\_installation

**2.5 消息队列服务安装**

在控制节点执行以下命令：

1. 安装软件

# apt-get install rabbitmq-server

1. 添加 openstack 用户：

# rabbitmqctl add\_user openstack 1234



其中1234是为openstack用户设置的密码。

1. 给``openstack``用户配置写和读权限：

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



**2.6 安装Memcached**

在控制节点执行以下命令：

1. 安装软件

# apt-get install memcached python-memcache

1. 编辑 /etc/memcached.conf 文件

把

-l 127.0.0.1

这一行改为

-l 192.168.0.100

其中192.168.0.100为控制节点在管理网络中的ip 。

1. 重启Memcached服务：

# service memcached restart

**3 认证服务——keystone组件安装**

**3.1 创建keystone数据库**

在控制节点执行以下命令。

（1）进入数据库

# mysql -u root -p

（2）创建 keystone 数据库：

CREATE DATABASE keystone;

（3）对``keystone``数据库授予恰当的权限：

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

IDENTIFIED BY '1234';

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

IDENTIFIED BY '1234';

其中1234为访问keystone数据库密码。

（4）退出数据库客户端。

exit

**3.2 keystone服务安装与配置**

在控制节点执行以下命令。

（1）安装后禁止keystone 服务自动启动：

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

（2）安装软件

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

（3）随机产生一个token

# openssl rand -hex 10 > token

保存在 token这个文件中备用，在下文中用ADMIN\_TOKEN指代，读者应注意替换 。

查看命令：

# cat token



（4）编辑文件 /etc/keystone/keystone.conf 并完成如下动作：

I、在``[DEFAULT]``部分，定义初始管理令牌的值：

[DEFAULT]

...

admin\_token = ADMIN\_TOKEN

使用前一步骤生成的随机数（该随机数保存在token文件中）替换``ADMIN\_TOKEN`` 。

（注：省略号表示该文件中原有部分应保留）

II、在 [database] 部分，配置数据库访问：

[database]

...

connection = mysql+pymysql://keystone:1234@controller/keystone

注意删除 [database] 部分原有的包含 connection的一行。

III、在``[token]``部分，配置Fernet UUID令牌的提供者。

[token]

...

provider = fernet

（5）初始化身份认证服务的数据库：

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

（6）初始化Fernet keys：

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

**3.3 apache服务器配置**

在控制节点执行以下命令。

（1）编辑``/etc/apache2/apache2.conf``文件，为控制节点配置``ServerName``选项：

在最后一行添加

ServerName controller

（2）使用下面的内容创建

``/etc/apache2/sites-available/wsgi-keystone.conf``文件：

Listen 5000

Listen 35357

<VirtualHost \*:5000>

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

WSGIProcessGroup keystone-public

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

WSGIApplicationGroup %{GLOBAL}

WSGIPassAuthorization On

ErrorLogFormat "%{cu}t %M"

ErrorLog /var/log/apache2/keystone.log

CustomLog /var/log/apache2/keystone\_access.log combined

<Directory /usr/bin>

Require all granted

</Directory>

</VirtualHost>

<VirtualHost \*:35357>

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

WSGIProcessGroup keystone-admin

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

WSGIApplicationGroup %{GLOBAL}

WSGIPassAuthorization On

ErrorLogFormat "%{cu}t %M"

ErrorLog /var/log/apache2/keystone.log

CustomLog /var/log/apache2/keystone\_access.log combined

<Directory /usr/bin>

Require all granted

</Directory>

</VirtualHost>

1. 删除/etc/apache2/sites-enabled文件夹下所有文件

# rm /etc/apache2/sites-enabled/\*

1. 开启认证服务虚拟主机：

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

/etc/apache2/sites-enabled

1. 重启服务器

# service apache2 restart

（6） 默认情况下，Ubuntu 上的安装包会自动创建一个 SQLite 数据库。

因为这里配置使用SQL 数据库服务器，所以你可以删除SQLite 服务库文件：

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

**3.4 创建服务实体和API端点**

（1） 配置必要环境变量：

# export OS\_TOKEN=ADMIN\_TOKEN

使用3.2中生成的认证令牌替代``ADMIN\_TOKEN``,使用cat token可以查看

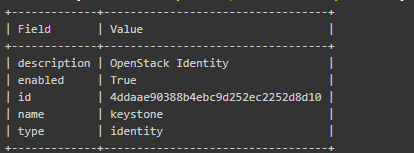
# export OS\_URL=http://controller:35357/v3

# export OS\_IDENTITY\_API\_VERSION=3

（2）创建服务实体

# openstack service create \

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



（3）创建API端点

# openstack endpoint create --region RegionOne identity public http://controller:5000/v3

# openstack endpoint create --region RegionOne identity internal http://controller:5000/v3

# openstack endpoint create --region RegionOne identity admin <http://controller:35357/v3>

**3.5 创建域、项目、用户和角色**

（1）创建域``default``：

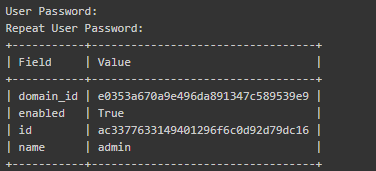
# openstack domain create --description "Default Domain" default

（2）创建 admin 项目：

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

（3）创建 admin 用户：

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



需要为用户设置密码，统一设置为1234。

（4）创建 admin 角色：

# openstack role create admin

（5）添加``admin`` 角色到 admin 项目和用户上：

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

（注：该命令无输出，即为执行成功）

（6）创建``service``项目：

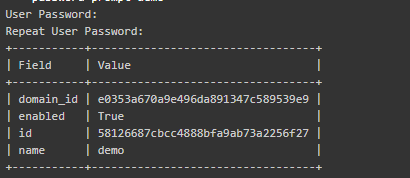
# openstack project create --domain default --description "Service Project" service

（7）创建``demo`` 项目：

# openstack project create --domain default --description "Demo Project" demo

（8）创建``demo`` 用户：

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



需要为用户设置密码，统一设置为1234。

（9）创建 user 角色：

# openstack role create user

（10）添加 user``角色到 ``demo 项目和用户：

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

（注：该命令无输出即为运行成功）

**3.6 验证操作**

在控制节点执行以下操作。

（1）编辑 /etc/keystone/keystone-paste.ini 文件，从``[pipeline:public\_api]``，[pipeline:admin\_api]``和``[pipeline:api\_v3]``部分删除``admin\_token\_auth`` 。

（2）重置``OS\_TOKEN``和``OS\_URL`` 环境变量：

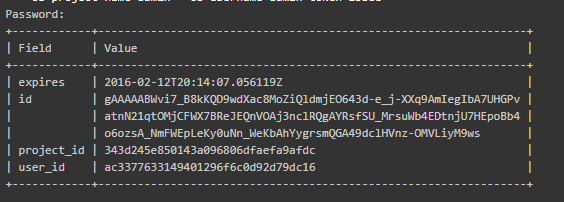
# unset OS\_TOKEN OS\_URL

（3）作为 admin 用户，请求认证令牌：

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

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

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



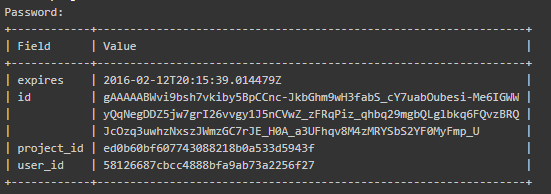
输入密码1234，并得到如图输出即为安装成功。

（4）作为``demo`` 用户，请求认证令牌：

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

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

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



输入密码1234，并得到如图输出即为安装成功。

**3.7 创建脚本**

在控制节点执行以下操作。

（1）编辑文件 admin-openrc 并添加如下内容：

export OS\_PROJECT\_DOMAIN\_NAME=default

export OS\_USER\_DOMAIN\_NAME=default

export OS\_PROJECT\_NAME=admin

export OS\_USERNAME=admin

export OS\_PASSWORD=1234

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

export OS\_IDENTITY\_API\_VERSION=3

export OS\_IMAGE\_API\_VERSION=2

其中1234为之前创建admin用户时设置的密码。

（2）编辑文件 demo-openrc 并添加如下内容：

export OS\_PROJECT\_DOMAIN\_NAME=default

export OS\_USER\_DOMAIN\_NAME=default

export OS\_PROJECT\_NAME=demo

export OS\_USERNAME=demo

export OS\_PASSWORD=1234

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

export OS\_IDENTITY\_API\_VERSION=3

export OS\_IMAGE\_API\_VERSION=2

其中1234为之前创建demo用户时设置的密码。

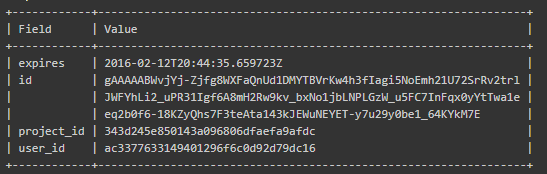
（3）试用脚本

I、加载``admin-openrc``文件来身份认证服务的环境变量位置和``admin``项目和用户证书：

# . admin-openrc

II、请求认证令牌:

# openstack token issue



**4 镜像服务——glance组件安装**

**4.1 创建glance数据库**

在控制节点执行以下操作。

1. 用数据库连接客户端以 root 用户连接到数据库服务器：

# mysql -u root -p

1. 创建 glance 数据库：

CREATE DATABASE glance;

1. 对``glance``数据库授予恰当的权限：

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

IDENTIFIED BY '1234' ;

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

IDENTIFIED BY '1234' ;

1. 退出数据库客户端。

exit

**4.2 创建glance服务证书、服务实体和服务端点**

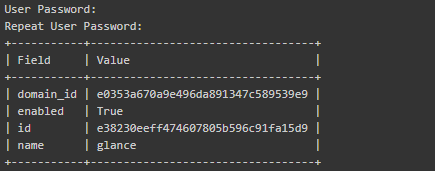
在控制节点执行以下操作。

1. 获得 admin 凭证来获取只有管理员能执行的命令的访问权限：

# . admin-openrc

1. 创建 glance 用户：

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



需要为glance 用户设置密码，统一设置为1234。

1. 添加 admin 角色到 glance 用户和 service 项目上。

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

（注：该命令无输出，即为执行成功）

1. 创建``glance``服务实体：

# openstack service create --name glance --description "OpenStack Image" image

1. 创建镜像服务的 API 端点：

# openstack endpoint create --region RegionOne image public http://controller:9292

# openstack endpoint create --region RegionOne image internal http://controller:9292

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

**4.3 安装并配置组件**

在控制节点执行以下操作。

1. 安装软件包：

# apt-get install glance

1. 编辑文件 /etc/glance/glance-api.conf 并完成如下动作：
2. 在 [database] 部分，配置数据库访问：

[database]

...

connection = mysql+pymysql://glance:1234@controller/glance

其中1234为3.1中设置的glance数据库访问密码。

（注：省略号表示保留该部分原有的配置）

1. 在 [keystone\_authtoken] 和 [paste\_deploy] 部分，配置认证服务访问：

[keystone\_authtoken]

auth\_uri = http://controller:5000

auth\_url = http://controller:35357

memcached\_servers = controller:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = glance

password = 1234

[paste\_deploy]

...

flavor = keystone

其中1234为3.2中设置的认证服务访问密码。

1. 在 [glance\_store] 部分，配置本地文件系统存储和镜像文件位置：

[glance\_store]

...

stores = file,http

default\_store = file

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

1. 编辑文件 ``/etc/glance/glance-registry.conf``并完成如下动作：
2. 在 [database] 部分，配置数据库访问：

[database]

...

connection = mysql+pymysql://glance:1234@controller/glance

其中1234为3.1中设置的glance数据库访问密码。

（注：省略号表示保留该部分原有的配置）

1. 在 [keystone\_authtoken] 和 [paste\_deploy] 部分，配置认证服务访问：

[keystone\_authtoken]

auth\_uri = http://controller:5000

auth\_url = http://controller:35357

memcached\_servers = controller:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = glance

password = 1234

[paste\_deploy]

...

flavor = keystone

1. 写入镜像服务数据库：

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

1. 重启镜像服务：

# service glance-registry restart

# service glance-api restart

**4.4 验证操作**

在控制节点执行以下操作。

1. 获得 admin 凭证来获取只有管理员能执行的命令的访问权限：

# . admin-openrc

1. 下载源镜像：

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

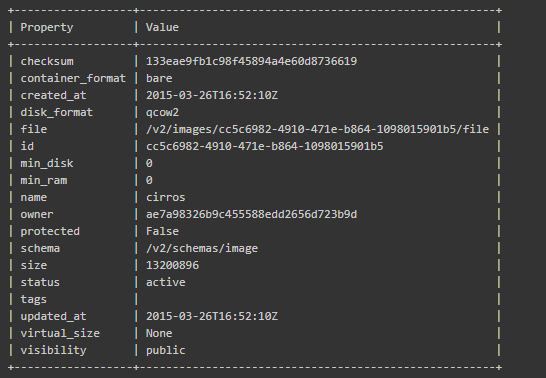
1. 使用 QCOW2 磁盘格式， bare 容器格式上传镜像到镜像服务并设置公共可见：

# openstack image create "cirros" \

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

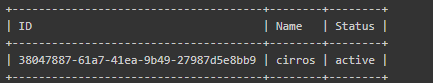
--disk-format qcow2 --container-format bare \

--public



1. 确认镜像的上传并验证属性：

# openstack image list



**5 计算服务——nova组件安装**

**5.1 创建navo、nova\_api数据库**

在控制节点执行以下操作。

1. 用数据库连接客户端以 root 用户连接到数据库服务器：

# mysql -u root -p

1. 创建 nova\_api 和 nova 数据库：

CREATE DATABASE nova\_api;

CREATE DATABASE nova;

1. 对数据库进行正确的授权：

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

IDENTIFIED BY '1234';

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

IDENTIFIED BY '1234';

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

IDENTIFIED BY '1234';

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

IDENTIFIED BY '1234';

其中1234为可自定义数据库访问密码。

1. 退出数据库客户端。

exit

**5.2 创建服务证书、服务实体和服务API端点**

在控制节点执行以下操作。

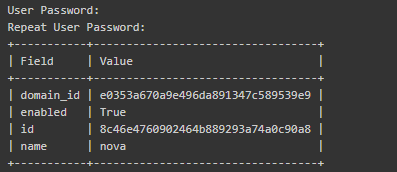
1. 获得 admin 凭证来获取只有管理员能执行的命令的访问权限：

# . admin-openrc

1. 创建 nova 用户：

# openstack user create --domain default \

--password-prompt nova



1. 给 nova 用户添加 admin 角色：

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

（注：该命令无输出）

1. 创建 nova 服务实体：

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

1. 创建 Compute 服务 API 端点 ：

# openstack endpoint create --region RegionOne \

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

# openstack endpoint create --region RegionOne \

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

# openstack endpoint create --region RegionOne \

compute admin [http://controller:8774/v2.1/%\(tenant\_id\)s](http://controller:8774/v2.1/%25\(tenant_id\)s)

**5.3 安装配置组件**

在控制节点执行以下操作。

1. 安装软件包：

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

nova-novncproxy nova-scheduler

1. 编辑``/etc/nova/nova.conf``文件并完成下面的操作：
2. 在``[DEFAULT]``部分，只启用计算和元数据API：

[DEFAULT]

...

enabled\_apis = osapi\_compute,metadata

这一行在原文件中应该是有的，应先注意查找。

（注：省略号表示其它东西应保留）

1. 在``[api\_database]``和``[database]``部分，配置数据库的连接：

[api\_database]

connection = mysql+pymysql://nova:1234@controller/nova\_api

[database]

connection = mysql+pymysql://nova:1234@controller/nova

其中1234为5.1中为数据库设置的密码。

1. 在 “[DEFAULT]” 和 “[oslo\_messaging\_rabbit]”部分，配置 “RabbitMQ” 消息队列访问：

[DEFAULT]

...

rpc\_backend = rabbit

[oslo\_messaging\_rabbit]

rabbit\_host = controller

rabbit\_userid = openstack

rabbit\_password = 1234

其中1234为2.5中设置的密码。

（注：[oslo\_messaging\_rabbit]这一部分在原文件中没有，应自行添加，下文中也有几个部分原文没有，需要读者自行添加）

1. 在 “[DEFAULT]” 和 “[keystone\_authtoken]” 部分，配置认证服务访问：

[DEFAULT]

...

auth\_strategy = keystone

[keystone\_authtoken]

auth\_uri = http://controller:5000

auth\_url = http://controller:35357

memcached\_servers = controller:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = nova

password = 1234

其中1234为5.2中设置的密码。

1. 在 [DEFAULT 部分，配置``my\_ip`` 来使用控制节点的管理接口的IP 地址。

[DEFAULT]

...

my\_ip = 192.168.0.100

其中192.168.0.100为控制节点在管理网络中的ip，参考图1.2。

1. 在 [DEFAULT] 部分，使能 Networking 服务：

[DEFAULT]

...

use\_neutron = True

firewall\_driver = nova.virt.firewall.NoopFirewallDriver

1. 在``[vnc]``部分，配置VNC代理使用控制节点的管理接口IP地址 ：

[vnc]

vncserver\_listen = $my\_ip

vncserver\_proxyclient\_address = $my\_ip

1. 在 [glance] 区域，配置镜像服务 API 的位置：

[glance]

api\_servers = http://controller:9292

1. 在 [oslo\_concurrency] 部分，配置锁路径：

[oslo\_concurrency]

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

（注意： [oslo\_concurrency]这一部分在原文件中是存在的，所以应修改原文，不是添加）

1. 从 [DEFAULT] 区域去除 log-dir 选项。
2. 同步Compute 数据库：

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

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

1. 重启 Compute 服务：

# service nova-api restart

# service nova-consoleauth restart

# service nova-scheduler restart

# service nova-conductor restart

# service nova-novncproxy restart

**5.3 安装配置计算节点**

在计算节点执行以下操作。

1. 安装软件包：

# apt-get install nova-compute

1. 编辑``/etc/nova/nova.conf``文件并完成下面的操作：
2. 在``[DEFAULT]`` 和 [oslo\_messaging\_rabbit]部分，配置``RabbitMQ``消息队列的连接：

[DEFAULT]

...

rpc\_backend = rabbit

[oslo\_messaging\_rabbit]

rabbit\_host = controller

rabbit\_userid = openstack

rabbit\_password = 1234

1234为2.5中设置的密码。

1. 在 “[DEFAULT]” 和 “[keystone\_authtoken]” 部分，配置认证服务访问：

[DEFAULT]

...

auth\_strategy = keystone

[keystone\_authtoken]

auth\_uri = http://controller:5000

auth\_url = http://controller:35357

memcached\_servers = controller:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = nova

password = 1234

1. 在 [DEFAULT] 部分，配置 my\_ip 选项：

[DEFAULT]

...

my\_ip = 192.168.0.101

其中192.168.0.101为计算节点在管理网络中的IP地址，参考图1.2。

1. 在 [DEFAULT] 部分，使能 Networking 服务：

[DEFAULT]

...

use\_neutron = True

firewall\_driver = nova.virt.firewall.NoopFirewallDriver

1. 在``[vnc]``部分，启用并配置远程控制台访问：

[vnc]

enabled = True

vncserver\_listen = 0.0.0.0

vncserver\_proxyclient\_address = $my\_ip

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

1. 在 [glance] 区域，配置镜像服务 API 的位置：

[glance]

api\_servers = <http://controller:9292>

1. 在[oslo\_concurrency] 部分，配置锁路径：

[oslo\_concurrency]

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

（注意： [oslo\_concurrency]这一部分在原文件中是存在的，所以应修改原文，不是添加）

1. 从 [DEFAULT] 区域去除 log-dir 选项。
2. 关于硬件加速
3. 确定您的计算节点是否支持虚拟机的硬件加速。

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

1. 如果这个命令返回了 one or greater 的值，那么你的计算节点支持硬件加速且不需要额外的配置。

如果这个命令返回了 zero 值，那么你的计算节点不支持硬件加速。你必须配置 libvirt 来使用 QEMU 去代替 KVM

在 /etc/nova/nova-compute.conf 文件的 [libvirt] 区域做出如下编辑：

[libvirt]

...

virt\_type = qemu

1. 重启计算服务：

# service nova-compute restart

**5.3 验证操作**

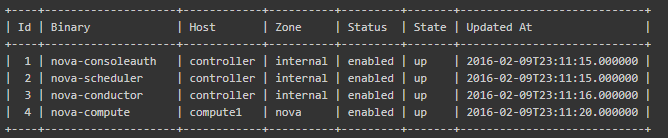
在控制节点执行以下操作。

1. 获得 admin 凭证来获取只有管理员能执行的命令的访问权限：

# . admin-openrc

1. 列出服务组件，以验证是否成功启动并注册了每个进程：

# openstack compute service list



该输出应该如图显示三个服务组件在控制节点上启用，一个服务组件在计算节点上启用。

**6 网络服务——neutron组件安装**

**6.1 创建neutron数据库**

在控制节点执行以下操作。

1. 用数据库连接客户端以 root 用户连接到数据库服务器：

# mysql -u root -p

1. 创建``neutron`` 数据库：

CREATE DATABASE neutron;

1. 对``neutron`` 数据库授予合适的访问权限：

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

IDENTIFIED BY '1234';

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

IDENTIFIED BY '1234';

其中1234是为neutron数据库设置的密码。

1. 退出数据库客户端。

exit

**6.2创建服务证书和服务API端点**

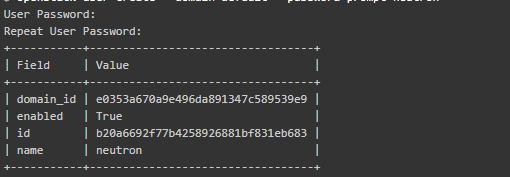
在控制节点执行以下操作。

1. 获得 admin 凭证来获取只有管理员能执行的命令的访问权限：

# . admin-openrc

1. 创建``neutron``用户：

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



需要为neutron用户设置密码，统一设置为1234。

1. 添加``admin`` 角色到``neutron`` 用户：

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

（注：该命令无输出）

1. 创建``neutron``服务实体：

# openstack service create --name neutron \

--description "OpenStack Networking" network

1. 创建网络服务API端点：

# openstack endpoint create --region RegionOne \

network public http://controller:9696

# openstack endpoint create --region RegionOne \

network internal http://controller:9696

# openstack endpoint create --region RegionOne \

network admin <http://controller:9696>

**6.3 控制节点安装与配置**

在控制节点执行以下操作。

1. 安装组件

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

neutron-linuxbridge-agent neutron-l3-agent neutron-dhcp-agent \

neutron-metadata-agent

1. 编辑``/etc/neutron/neutron.conf`` 文件并完成如下操作：
2. ``[DEFAULT]``部分

[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

（注意第一行在原文中是存在的。）

1. [database] 部分

[database]

...

connection = mysql+pymysql://neutron:1234@controller/neutron

（注意删除原文中包含connection的一行。）

其中1234为6.1中设置的database数据库密码。

1. “[oslo\_messaging\_rabbit]”部分

[oslo\_messaging\_rabbit]

...

rabbit\_host = controller

rabbit\_userid = openstack

rabbit\_password = 1234

其中1234为2.5中设置的密码。

1. “[keystone\_authtoken]” 部分

[keystone\_authtoken]

auth\_uri = http://controller:5000

auth\_url = http://controller:35357

memcached\_servers = controller:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = neutron

password = 1234

其中1234为6.2中设置的密码。

1. ``[nova]``部分

[nova]

...

auth\_url = http://controller:35357

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

region\_name = RegionOne

project\_name = service

username = nova

password = 1234

其中1234为5.2中设置的密码。

1. 编辑``/etc/neutron/plugins/ml2/ml2\_conf.ini``文件并完成以下操作：
2. 在``[ml2]``部分

[ml2]

...

type\_drivers = flat,vlan,vxlan

tenant\_network\_types = vxlan

mechanism\_drivers = linuxbridge,l2population

extension\_drivers = port\_security

1. ``[ml2\_type\_flat]``部分

[ml2\_type\_flat]

...

flat\_networks = provider

1. 在``[ml2\_type\_vxlan]``部分

[ml2\_type\_vxlan]

...

vni\_ranges = 1:1000

1. ``[securitygroup]``部分

[securitygroup]

...

enable\_ipset = True

1. 编辑``/etc/neutron/plugins/ml2/linuxbridge\_agent.ini``文件并且完成以下操作：
2. 在``[linux\_bridge]``部分，将公共虚拟网络和公共物理网络接口对应起来：

[linux\_bridge]

physical\_interface\_mappings = provider:eth1

其中eth1为控制节点主机的第二块网卡名，请参考图1.2。

1. 在``[vxlan]``部分，禁止VXLAN覆盖网络：

[vxlan]

enable\_vxlan = True

local\_ip = 192.168.0.100

l2\_population = True

其中，192.168.0.100为控制节点管理网络ip，参考图1.2。

1. 在 ``[securitygroup]``部分，启用安全组并配置 Linuxbridge iptables firewall driver:

[securitygroup]

...

enable\_security\_group = True

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

1. 编辑``/etc/neutron/l3\_agent.ini``文件并完成以下操作：
2. 在``[DEFAULT]``部分，配置Linuxbridge接口驱动和外部网络网桥：

[DEFAULT]

...

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

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

enable\_isolated\_metadata = True

1. 编辑``/etc/neutron/dhcp\_agent.ini``文件并完成下面的操作：
2. 在``[DEFAULT]``部分，配置Linuxbridge驱动接口，DHCP驱动并启用隔离元数据，这样在公共网络上的实例就可以通过网络来访问元数据

[DEFAULT]

...

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

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

enable\_isolated\_metadata = True

1. 编辑``/etc/neutron/metadata\_agent.ini``文件并完成以下操作：
2. 在``[DEFAULT]`` 部分，配置元数据主机以及共享密码：

[DEFAULT]

...

nova\_metadata\_ip = controller

metadata\_proxy\_shared\_secret = 1234

其中1234人为设置的密码。

1. 编辑``/etc/nova/nova.conf``文件并完成以下操作：
2. 在``[neutron]``部分，配置访问参数，启用元数据代理并设置密码：

[neutron]

url = http://controller:9696

auth\_url = http://controller:35357

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

region\_name = RegionOne

project\_name = service

username = neutron

password = 1234

service\_metadata\_proxy = True

metadata\_proxy\_shared\_secret = 1234

1. 同步数据库：

# 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

1. 重启服务：

# service nova-api restart

# service neutron-server restart

# service neutron-linuxbridge-agent restart

# service neutron-dhcp-agent restart

# service neutron-metadata-agent restart

# service neutron-l3-agent restart

（注：最后一条命令是重启L3服务，是私有网特有的）

**6.4 计算节点安装与配置**

在计算节点执行以下操作。

1. 安装组件

# apt-get install neutron-linuxbridge-agent

1. 编辑``/etc/neutron/neutron.conf`` 文件并完成如下操作：
2. 在``[database]`` 部分，注释或删除所有``connection`` 项，因为计算节点不直接访问数据库。
3. 在 “[DEFAULT]” 和 “[oslo\_messaging\_rabbit]”部分，配置 “RabbitMQ” 消息队列的连接：

[DEFAULT]

...

rpc\_backend = rabbit

[oslo\_messaging\_rabbit]

...

rabbit\_host = controller

rabbit\_userid = openstack

rabbit\_password = 1234

1. 在 “[DEFAULT]” 和 “[keystone\_authtoken]” 部分，配置认证服务访问：

[DEFAULT]

...

auth\_strategy = keystone

[keystone\_authtoken]

auth\_uri = http://controller:5000

auth\_url = http://controller:35357

memcached\_servers = controller:11211

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

project\_name = service

username = neutron

password = 1234

1. 编辑``/etc/neutron/plugins/ml2/linuxbridge\_agent.ini``文件并且完成以下操作：
2. 在``[linux\_bridge]``部分，将公共虚拟网络和公共物理网络接口对应起来：

[linux\_bridge]

physical\_interface\_mappings = provider:eth1

其中eth1为计算节点主机第二块网卡名称，请参考图1.2。

1. 在``[vxlan]``部分，禁止VXLAN覆盖网络：

[vxlan]

enable\_vxlan = True

local\_ip = 192.168.0.101

l2\_population = True

1. 在 ``[securitygroup]``部分，启用安全组并配置 Linuxbridge iptables firewall driver:

[securitygroup]

...

enable\_security\_group = True

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

1. 编辑``/etc/nova/nova.conf``文件并完成下面的操作：
2. 编辑``/etc/nova/nova.conf``文件并完成下面的操作：

[neutron]

url = http://controller:9696

auth\_url = http://controller:35357

auth\_type = password

project\_domain\_name = default

user\_domain\_name = default

region\_name = RegionOne

project\_name = service

username = neutron

password = 1234

1. 重启服务

# service nova-compute restart

# service neutron-linuxbridge-agent restart

**6.5 验证操作**

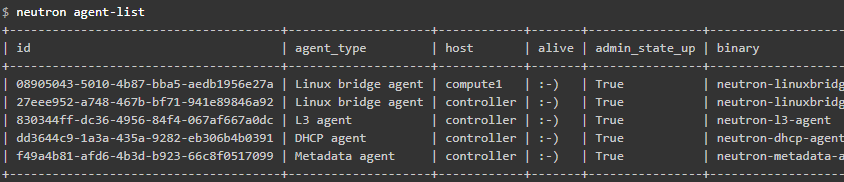
在控制节点执行以下操作。

1. 获得 admin 凭证来获取只有管理员能执行的命令的访问权限：

# . admin-openrc

1. 列出代理以验证启动 neutron 代理是否成功：

# neutron agent-list



输出结果应该包括控制节点上的四个代理和每个计算节点上的一个代理。注意alive一行应该是笑脸，不应该是xxx 。

**7 图形界面服务——horizon组件安装**

**7.1 安装与配置**

在控制节点执行以下操作。

1. 安装软件包：

# apt-get install openstack-dashboard

1. 编辑文件 /etc/openstack-dashboard/local\_settings.py 并完成如下动作：

（注：以下变量设置大多能在原文件中找到，请在原文件中修改，不要重复添加）

1. 在 controller 节点上配置仪表盘以使用 OpenStack 服务：

OPENSTACK\_HOST = "controller"

1. 允许所有主机访问仪表板：

ALLOWED\_HOSTS = ['\*', ]

1. 配置 memcached 会话存储服务：

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

CACHES = {

'default': {

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

'LOCATION': 'controller:11211',

}

}

（注：SESSION\_ENGINE这个变量在原文件中是没有的，但CACHES这个变量是有的。所以应在CACHES变量前添加SESSION\_ENGINE变量这一行，并修改CACHES变量。）

1. 启用第3版认证API:

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

1. 启用对域的支持

OPENSTACK\_KEYSTONE\_MULTIDOMAIN\_SUPPORT = True

1. 配置API版本:

OPENSTACK\_API\_VERSIONS = {

"identity": 3,

"image": 2,

"volume": 2,

}

1. 通过仪表盘创建用户时的默认域配置为 default :

OPENSTACK\_KEYSTONE\_DEFAULT\_DOMAIN = "default"

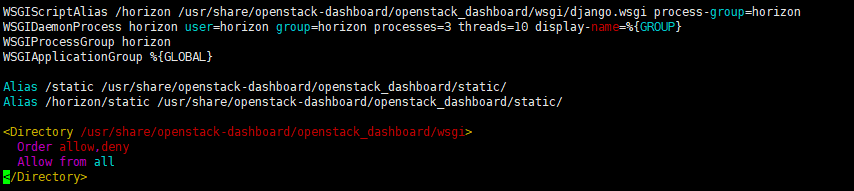
1. 通过仪表盘创建的用户默认角色配置为 user ：

OPENSTACK\_KEYSTONE\_DEFAULT\_ROLE = "user"

（注：这里是与公有网络选项即网络选项1的区别所在，此处不必配置禁用3层网络服务）

1. 编辑文件 /etc/apache2/conf-available/openstack-dashboard.conf 并完成如下动作：
2. 在WSGIProcessGroup horizon一行下添加一行

WSGIApplicationGroup %{GLOBAL}



1. 重新加载 web 服务器配置：

# service apache2 reload

**7.2 验证操作**

用可访问管理网络的浏览器访问<http://192.168.0.100/horizon>

**7.3 可能出现的错误及解决方法**

如果访问网址出现如下图的错误。

可尝试修改/etc/openstack-dashboard/local\_settings.py文件，将

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

改为

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



**8 创建实例**

**8.1 创建提供者网络**

在控制节点执行以下操作。

1. 在控制节点上，加载 admin 凭证来获取管理员能执行的命令访问权限：

# . admin-openrc

1. 创建网络：

# neutron net-create --shared --provider:physical\_network provider \

--provider:network\_type flat provider

1. 在网络上创建一个子网：

# neutron subnet-create --name provider \

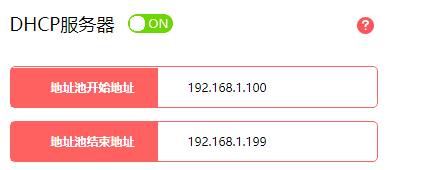
--allocation-pool start=192.168.1.150,end=192.168.1.199 \

--dns-nameserver DNS\_RESOLVER --gateway 192.168.1.1 \

provider 192.168.1.0/24

其中，192.168.1.0/24是路由器2的子网，请参考图1.2。start=192.168.1.150,end=192.168.1.199是为以后生成实例分配IP地址的范围。

设置开始和结束地址时，应先了解路由器的DHCP服务器的地址池范围，如下图所示。可以看到地址池是192.168.1.100到192.168.1.199，我们设置的地址范围应在地址池范围内。



**8.2 创建实例**

在控制节点执行以下操作。

1. 创建m1.nano规格的主机

# openstack flavor create --id 0 --vcpus 1 --ram 64 --disk 1 m1.nano

1. 生成一个键值对
2. 导入租户``demo``的凭证

# . demo-openrc

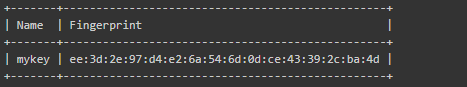
1. 生成和添加秘钥对：

# ssh-keygen -q -N ""

# openstack keypair create --public-key /root/.ssh/id\_rsa.pub mykey

1. 验证公钥的添加是否成功：

# openstack keypair list



1. 增加安全组规则，允许icmp和ssh

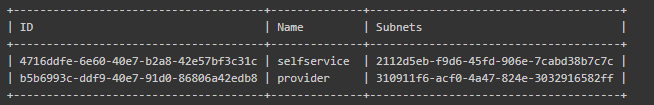
# openstack security group rule create --proto icmp default

# openstack security group rule create --proto tcp --dst-port 22 default

1. 列出可用网络：

# . demo-openrc

# openstack network list



（注：读者看到的输出应该只有provider一行，请注意provider这个网络ID在下面会用到。）

1. 启动实例：

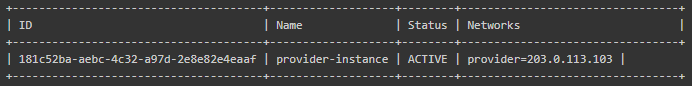
# openstack server create --flavor m1.nano --image cirros \

--nic net-id=b5b6993c-ddf9-40e7-91d0-86806a42edb8 \

--security-group default --key-name mykey provider-instance

1. 检查实例的状态：

# openstack server list



当构建过程完全成功后，状态会从 BUILD``变为``ACTIVE。

**8.3 访问实例**

在控制节点执行以下操作。

1. 使用 SSH远程访问实例：

# ssh [cirros@192.168.1.151](mailto:cirros@192.168.1.151)

其中192.168.1.151为实例的ip地址

1. 在实例上进行网络联通性测试

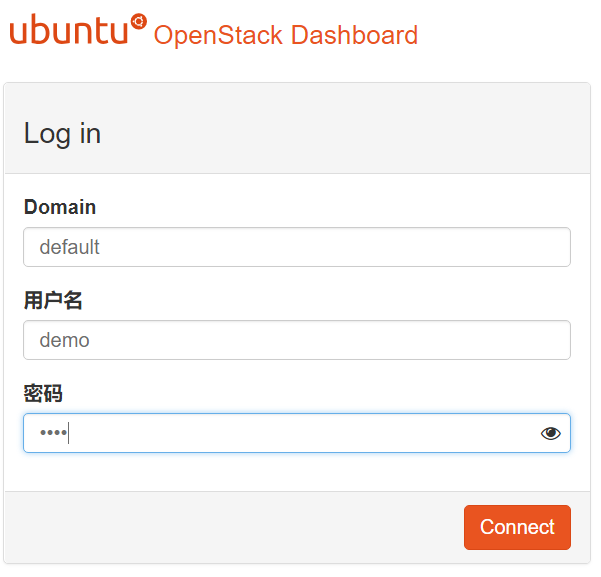
# ping openstack.org

# ping 192.168.1.1

**8.4 添加私有网络**

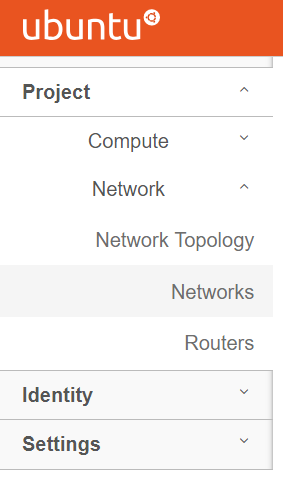
在能访问192.168.0.100的电脑上执行以下操作。这里使用图形界面创建私有网。

（1）在浏览器输入网址 <http://192.168.0.100/horizon>



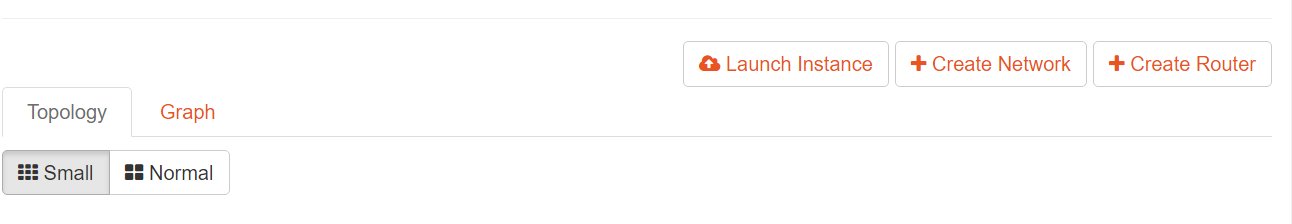
输入如图所示账号，密码为1234。

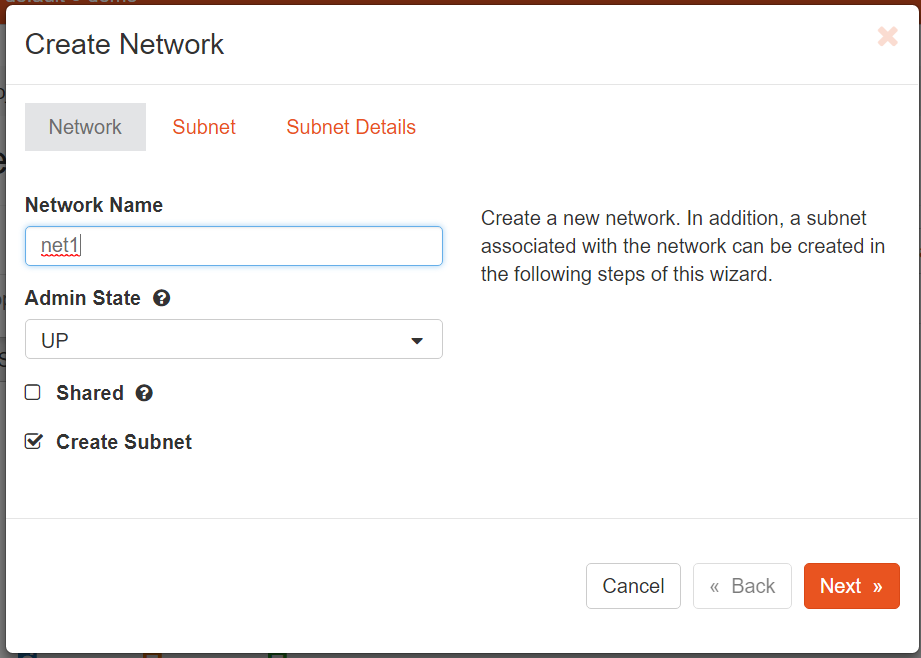
（2） 点击左边Project -> Network -> Network Topology

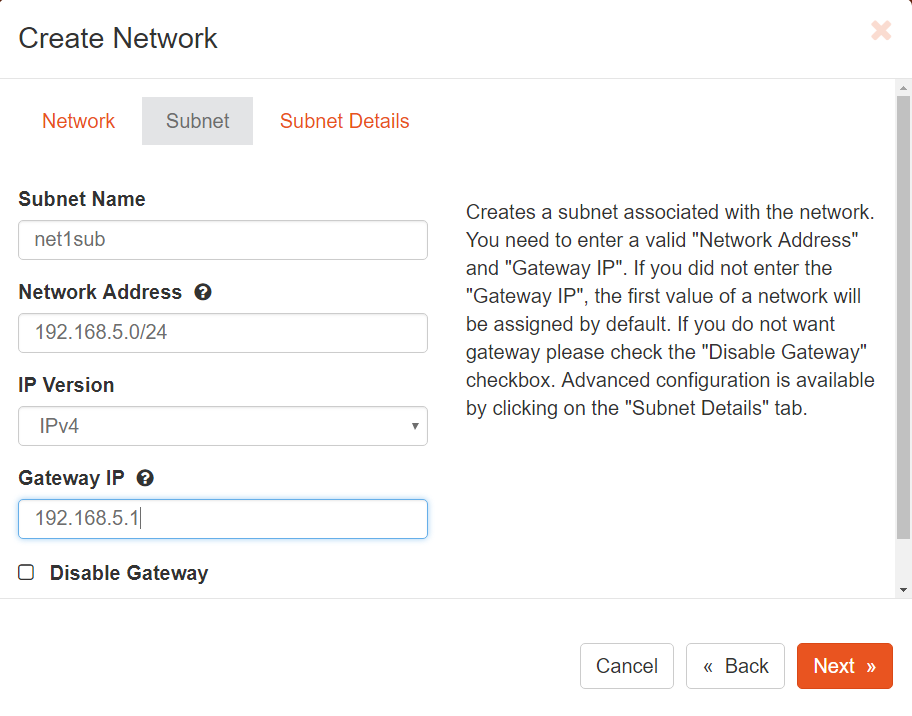


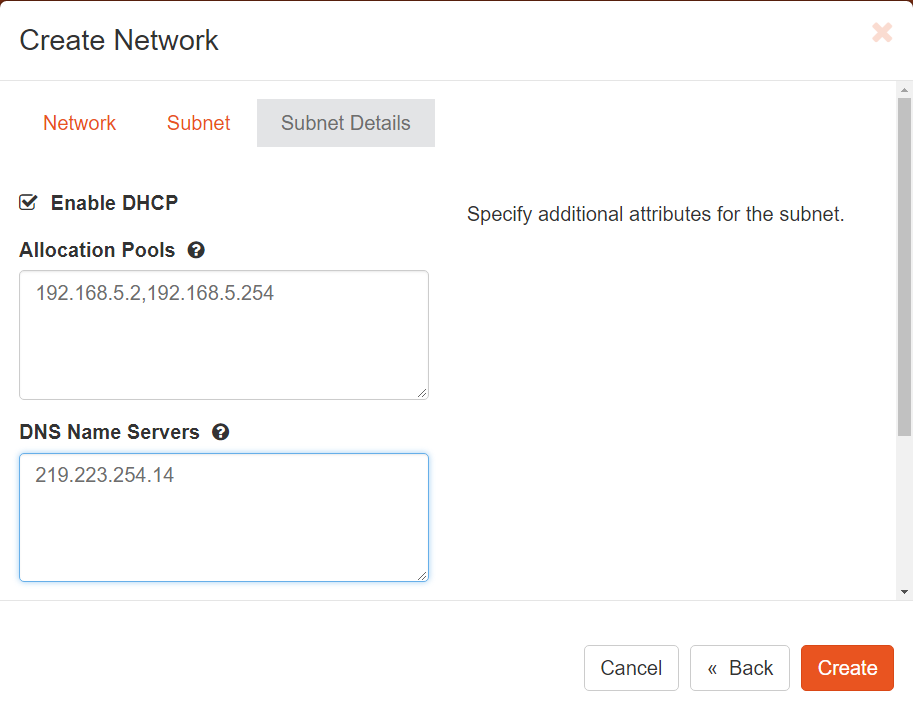
1. 点击Create Network -> 输入网络名net1 -> Next -> 输入子网名net1sub，Network Address 192.168.5.0/24 ， Gateway IP 192.168.5.1 –> Next -> 输入Allocation pools 192.168.5.2,192.168.5.254 ， DNS Name Server 219.223.254.14 -> Create

注：DNS服务器地址要设置一个控制节点和计算节点能连接上的。





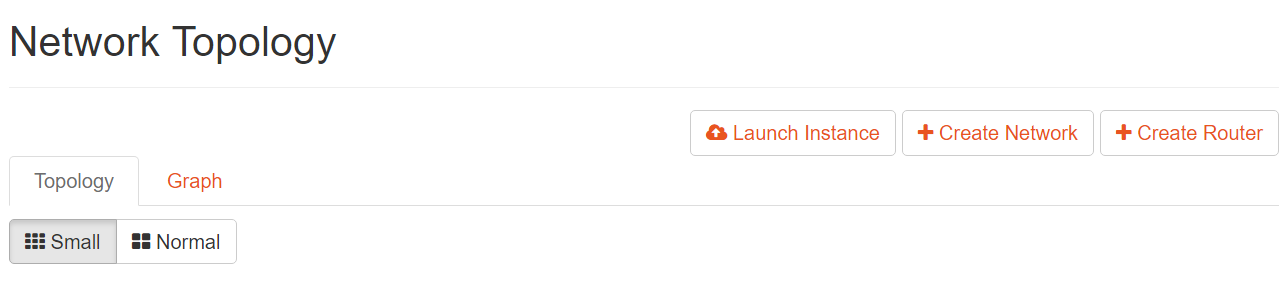


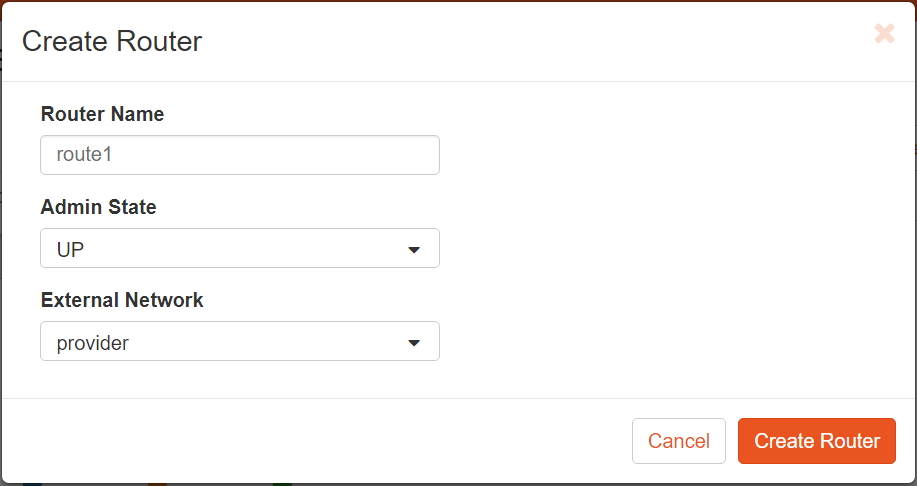


**8.5 添加虚拟路由器**

在添加私有网的页面，执行以下操作。

1. 点击Create Route -> 输入路由器名称route1 , 选择外部网络provider –> Create Router





1. 将鼠标悬浮到生成的路由器上-> 点击View Route Details -> Interfaces -> Add Interface -> subnet部分选择net1 –> Submit 。

