

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
1 组网	3
2 openstack 软件环境配置	8
2.1 主机网络配置.....	8
2.1.1 控制节点网络配置.....	8
2.1.2 计算节点网络配置.....	8
2.1.3 额外配置.....	8
2.1.4 连通性测试.....	9
2.2 各节点时间同步配置.....	9
2.2.1 控制节点.....	9
2.2.2 计算节点.....	9
2.2.3 时间同步验证.....	10
2.3 添加 openstack 软件源.....	10
2.4 安装 mysql 数据库.....	10
2.5 消息队列服务安装.....	11
2.6 安装 Memcached.....	11
3 认证服务——keystone 组件安装	12
3.1 创建 keystone 数据库.....	12
3.2 keystone 服务安装与配置.....	12
3.3 apache 服务器配置.....	13
3.4 创建服务实体和 API 端点.....	14
3.5 创建域、项目、用户和角色.....	15
3.6 验证操作.....	16
3.7 创建脚本.....	16
4 镜像服务——glance 组件安装	18
4.1 创建 glance 数据库.....	18
4.2 创建 glance 服务证书、服务实体和服务端点.....	18
4.3 安装并配置组件.....	19
4.4 验证操作.....	20
5 计算服务——nova 组件安装	22
5.1 创建 navo、nova_api 数据库.....	22
5.2 创建服务证书、服务实体和服务 API 端点.....	22
5.3 安装配置组件.....	23
5.3 安装配置计算节点.....	25
5.3 验证操作.....	26
6 网络服务——neutron 组件安装	28
6.1 创建 neutron 数据库.....	28

6.2 创建服务证书和服务 API 端点	28
6.3 控制节点安装与配置	29
6.4 计算节点安装与配置	32
6.5 验证操作	33
7 图形界面服务——horizon 组件安装	34
7.1 安装与配置	34
7.2 验证操作	36
7.3 可能出现的错误及解决方法	36
8 创建实例	36
8.1 创建提供者网络	36
8.2 创建实例	37
8.3 访问实例	38
8.4 添加私有网络	38
8.5 添加虚拟路由器	41

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。

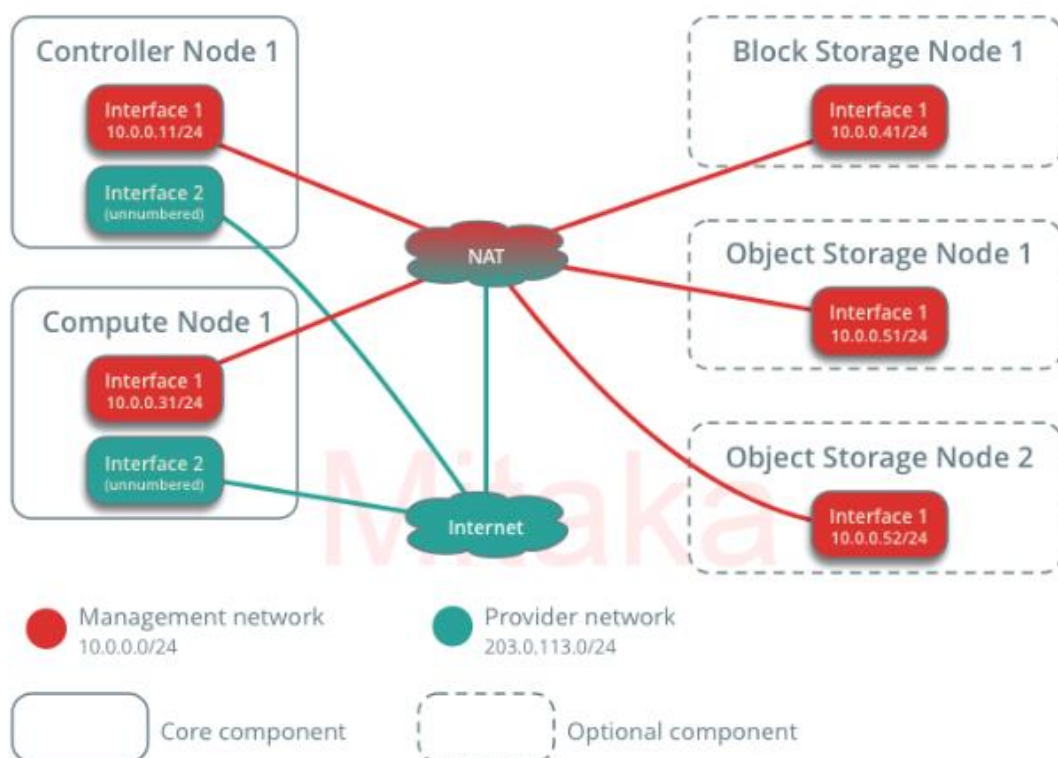


图 1.1 官方文档组网图

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。

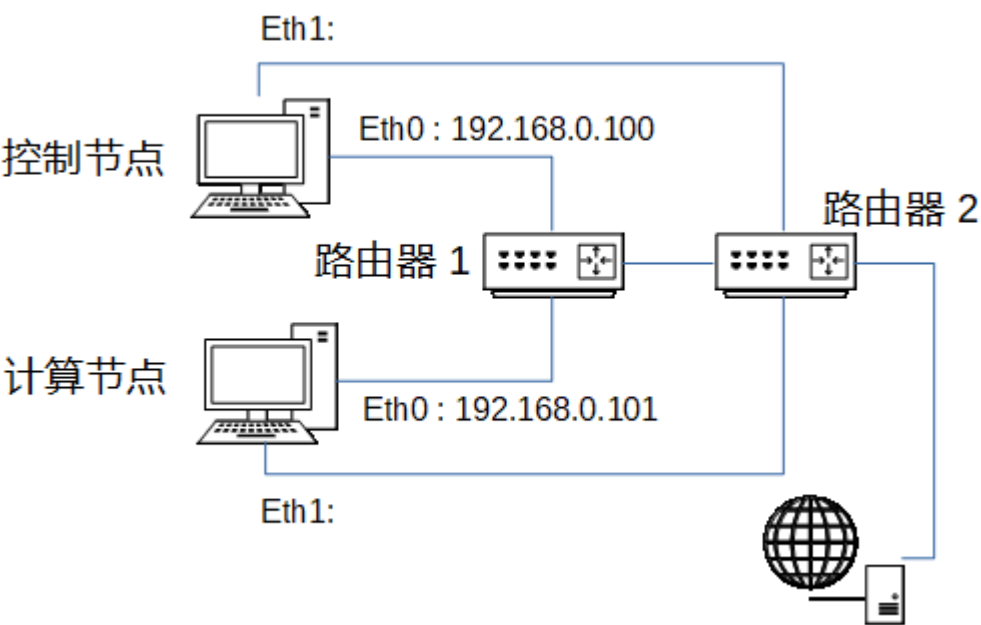


图 1.2 实现组网图

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。

IP与MAC绑定设置?

<input type="checkbox"/>	主机	MAC地址	IP地址	编辑
<input type="checkbox"/>	openstack1	44-8A-5B-C1-CF-7B	192.168.0.101	
<input type="checkbox"/>	openstack2	00-15-17-2D-52-08	192.168.0.100	

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）。

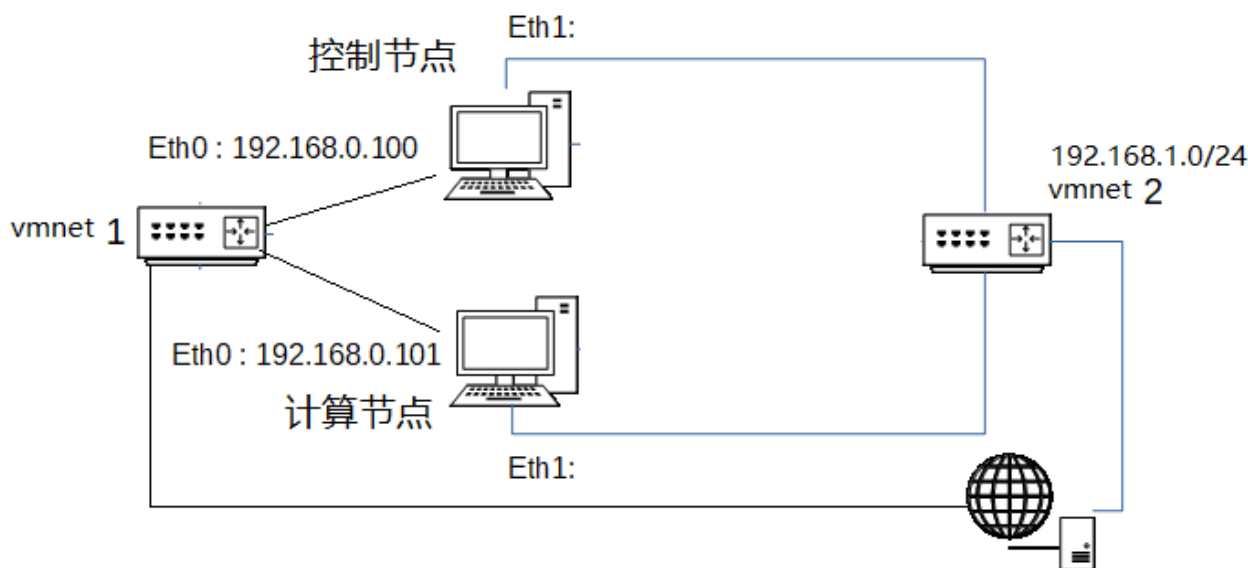


图 1.4 vmware 虚拟机组网图

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。

- (3) 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

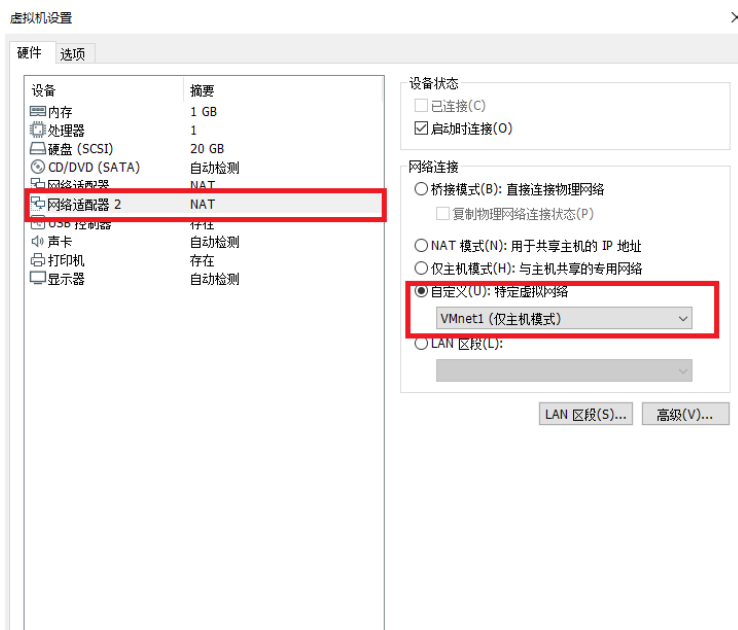


- (4) 为两台虚拟机添加第二个网卡，在两台虚拟机设置界面完成如下操作：

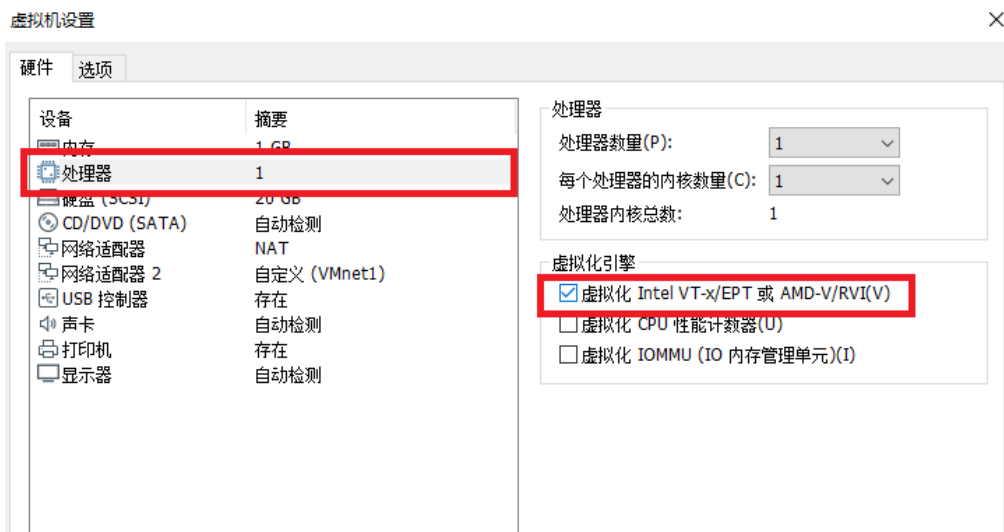
I、点击“编辑虚拟机设置”

II、点击“添加” -》 “网络适配器” -》 “完成”

III、 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



- (5) 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)



- (6) 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。

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto ens33
iface ens33 inet static
address 192.168.2.129
netmask 255.255.255.0
gateway 192.168.2.2
dns-nameserver 192.168.2.2
```

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
```

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

auto enp5s0
iface enp5s0 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
```

- (2) 编辑`` /etc/chrony/chrony.conf``文件

添加以下两行

```
allow 192.168.0.0/24  
server ntpl.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 ntpl.aliyun.com（阿里云服务器） 这个服务器作为时间同步服务器。

- (3) 重启 chrony 服务

```
# service chrony restart
```

2.2.2 计算节点

在计算节点上执行以下命令。

- (1) 安装 chrony

```
# apt install chrony
```

- (2) 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
```

- (3) 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。

```
root@ubuntu:~# chronyc sources
210 Number of sources = 5
MS Name/IP address         Stratum Poll Reach LastRx Last sample
=====
^- ntp8.flashdance.cx      2   7   337    64    +14ms[ +14ms] +/-  159ms
^- b.sin.pobot.net        2   7   377     5   -1352us[-1352us] +/-   67ms
^- makaki.miuku.net        2   7   377     3    +69ms[ +69ms] +/-  145ms
^- cn.ntp.faelix.net       2   7   377     5    +16ms[ +16ms] +/-  135ms
^* 120.25.115.20           2   7   377     5   -188us[-293us] +/- 3861us
root@ubuntu:~#
```

(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。

```
root@compute:~# chronyc sources
210 Number of sources = 1
MS Name/IP address         Stratum Poll Reach LastRx Last sample
=====
^* controller              3   6   377     2    -20us[ +131us] +/- 4933us
root@compute:~#
```

(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
```

- (4) 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
```

- (2) 添加 openstack 用户：

```
# rabbitmqctl add_user openstack 1234
```

```
Creating user "openstack" ...  
...done.
```

其中 1234 是为 openstack 用户设置的密码。

- (3) 给 ``openstack`` 用户配置写和读权限：

```
# rabbitmqctl set_permissions openstack ".*" ".*" ".*"
```

```
Setting permissions for user "openstack" in vhost "/" ...  
...done.
```

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
```

- (2) 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 。

- (3) 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
```

```
root@openstack2:~# cat token
f8a8bd287b0cb5d0bf0d
```

- (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>

```

- (4) 删除/etc/apache2/sites-enabled 文件夹下所有文件
rm /etc/apache2/sites-enabled/*
- (5) 开启认证服务虚拟主机:
ln -s /etc/apache2/sites-available/wsgi-keystone.conf \ /etc/apache2/sites-enabled
- (6) 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

Field	Value
description	OpenStack Identity
enabled	True
id	4ddaae90388b4ebc9d252ec2252d8d10
name	keystone
type	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
```

```
User Password:
Repeat User Password:
+-----+
| Field   | Value                                     |
+-----+
| domain_id | e0353a670a9e496da891347c589539e9 |
| enabled   | True                                   |
| id        | ac3377633149401296f6c0d92d79dc16 |
| name      | 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
```

```
User Password:
Repeat User Password:
+-----+
| Field   | Value                                     |
+-----+
| domain_id | e0353a670a9e496da891347c589539e9 |
| enabled   | True                                   |
| id        | 58126687cbcc4888bfa9ab73a2256f27 |
| name      | 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
```

```
Password:
+-----+
| Field      | Value                                                                 |
+-----+
| expires    | 2016-02-12T20:14:07.056119Z                                          |
| id         | gAAAAABWvi7_B8kKQD9wdXac8MoZiQldmjE0643d-e_j-XXq9AmIegIbA7UHGPv |
|            | atnN21qtOMjCFWx7BReJEQnVOAj3nc1RQgAYRsfsU_MrsuWb4EDtnjU7HEpoBb4 |
|            | o6ozsA_NmFWEpLeKy0uNn_WeKbAhYygrsmQGA49dclHVnz-OMVLiyM9ws        |
| project_id | 343d245e850143a096806dfaefa9afdc                                    |
| user_id    | ac3377633149401296f6c0d92d79dc16                                    |
+-----+
```

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
```

```
Password:
+-----+
| Field      | Value                                                                 |
+-----+
| expires    | 2016-02-12T20:15:39.014479Z                                          |
| id         | gAAAAABWvi9bsh7vkiby5BpCCnc-JkbGhm9wH3fabS_cY7uabOubesi-Me6IGWW |
|            | yQqNegDDZ5jw7grI26vvgy1J5nCVwZ_zFRqPiz_qhbq29mgbQLglbkq6FQvzBRQ |
|            | Jc0zq3uwhzNxsZJWmzGC7rJE_H0A_a3UFhqv8M4zMRYsSbS2YF0MyFmp_U      |
| project_id | ed0b60bf607743088218b0a533d5943f                                    |
| user_id    | 58126687cbcc4888bfa9ab73a2256f27                                    |
+-----+
```

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
```

Field	Value
expires	2016-02-12T20:44:35.659723Z
id	gAAAAABWvjYj-Zjfg8WxFaqnUd1DMYTbVrKw4h3fIagi5NoEmh21U72SrRv2tr1JWfYhLi2_uPR31Igf6A8mH2Rw9kv_bxNo1jbLNPLGzW_u5FC7InFqx0yYtTwa1eeq2b0f6-18KZyQhs7F3teAta143kJEWuNEYET-y7u29y0be1_64KYkM7E
project_id	343d245e850143a096806dfeafa9afdc
user_id	ac3377633149401296f6c0d92d79dc16

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`
- (2) 创建 glance 数据库：
`CREATE DATABASE glance;`
- (3) 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' ;`
- (4) 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`
- (2) 创建 glance 用户：
`# openstack user create --domain default --password-prompt glance`

```
User Password:
Repeat User Password:
+-----+
| Field | Value |
+-----+
| domain_id | e0353a670a9e496da891347c589539e9 |
| enabled | True |
| id | e38230eeff474607805b596c91fa15d9 |
| name | glance |
+-----+
```

需要为 glance 用户设置密码，统一设置为 1234。

- (3) 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)
- (4) 创建 ``glance`` 服务实体：
`# openstack service create --name glance --description "OpenStack Image" image`

- (5) 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
```

- (2) 编辑文件 `/etc/glance/glance-api.conf` 并完成如下动作:

I、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)

II、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。

III、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/
```

- (3) 编辑文件 ```/etc/glance/glance-registry.conf``` 并完成如下动作:

I、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)

II、 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
```

- (4) 写入镜像服务数据库：

```
# su -s /bin/sh -c "glance-manage db_sync" glance
```

- (5) 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
```

- (2) 下载源镜像：

```
# wget http://download.cirros-cloud.net/0.3.4/cirros-0.3.4-x86_64-disk.img
```

- (3) 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
```

Property	Value
checksum	133eae9fb1c98f45894a4e60d8736619
container_format	bare
created_at	2015-03-26T16:52:10Z
disk_format	qcow2
file	/v2/images/cc5c6982-4910-471e-b864-1098015901b5/file
id	cc5c6982-4910-471e-b864-1098015901b5
min_disk	0
min_ram	0
name	cirros
owner	ae7a98326b9c455588edd2656d723b9d
protected	False
schema	/v2/schemas/image
size	13200896
status	active
tags	
updated_at	2015-03-26T16:52:10Z
virtual_size	None
visibility	public

(4) 确认镜像的上传并验证属性:

`# openstack image list`

ID	Name	Status
38047887-61a7-41ea-9b49-27987d5e8bb9	cirros	active

5 计算服务——nova 组件安装

5.1 创建 nova、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
```

- (2) 创建 nova_api 和 nova 数据库：

```
CREATE DATABASE nova_api;
```

```
CREATE DATABASE nova;
```

- (3) 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。

- (4) 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

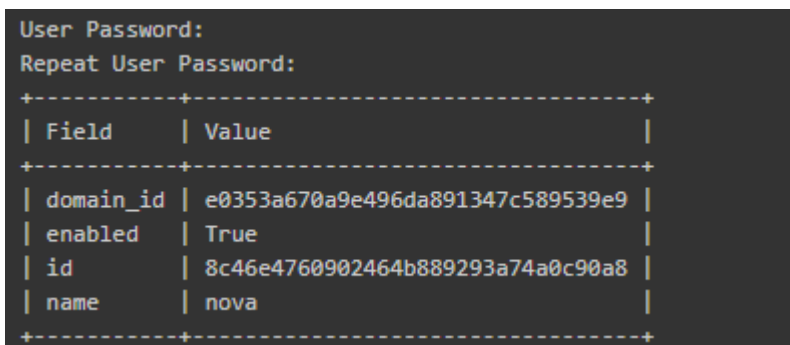
在控制节点执行以下操作。

- (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
```

- (2) 创建 nova 用户：

```
# openstack user create --domain default \
--password-prompt nova
```



```
User Password:
Repeat User Password:
+-----+
| Field      | Value                                     |
+-----+
| domain_id  | e0353a670a9e496da891347c589539e9 |
| enabled    | True                                 |
| id         | 8c46e4760902464b889293a74a0c90a8 |
| name       | nova                                 |
+-----+
```

- (3) 给 nova 用户添加 admin 角色：

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

(注：该命令无输出)

- (4) 创建 nova 服务实体：

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

- (5) 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
```

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
```

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

I、 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)

II、 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。

III、 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)

IV、 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。

V、 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。

VI、 在 [DEFAULT] 部分，使能 Networking 服务：

```
[DEFAULT]
```

```
...
```

```
use_neutron = True
```

```
firewall_driver = nova.virt.firewall.NoopFirewallDriver
```

VII、 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
```

VIII、 在 [glance] 区域，配置镜像服务 API 的位置：

```
[glance]
```

```
api_servers = http://controller:9292
```

IX、 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）

X、 从 [DEFAULT] 区域去除 log-dir 选项。

(3) 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
```

(4) 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
```

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

I、 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。

II、 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
```

III、 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。

IV、 在 [DEFAULT] 部分，使能 Networking 服务：

```
[DEFAULT]
```

```
...
```

```
use_neutron = True
```

```
firewall_driver = nova.virt.firewall.NoopFirewallDriver
```

V、 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
```

VI、 在 [glance] 区域，配置镜像服务 API 的位置：

```
[glance]
api_servers = http://controller:9292
```

VII、 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）

VIII、 从 [DEFAULT] 区域去除 log-dir 选项。

(3) 关于硬件加速

I、 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
```

II、 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
```

(4) 重启计算服务：

```
# service nova-compute restart
```

5.3 验证操作

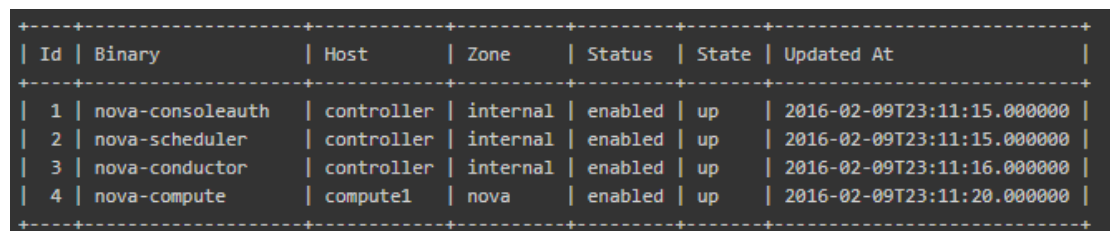
在控制节点执行以下操作。

(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
```

(2) 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
```



Id	Binary	Host	Zone	Status	State	Updated At
1	nova-consoleauth	controller	internal	enabled	up	2016-02-09T23:11:15.000000
2	nova-scheduler	controller	internal	enabled	up	2016-02-09T23:11:15.000000
3	nova-conductor	controller	internal	enabled	up	2016-02-09T23:11:16.000000
4	nova-compute	compute1	nova	enabled	up	2016-02-09T23:11:20.000000

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
```

- (2) 创建 ``neutron`` 数据库：

```
CREATE DATABASE neutron;
```

- (3) 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。

- (4) 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
```

- (2) 创建 ``neutron`` 用户：

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

```
User Password:
Repeat User Password:
+-----+-----+
| Field | Value |
+-----+-----+
| domain_id | e0353a670a9e496da891347c589539e9 |
| enabled | True |
| id | b20a6692f77b4258926881bf831eb683 |
| name | neutron |
+-----+-----+
```

需要为 neutron 用户设置密码，统一设置为 1234。

- (3) 添加 ``admin`` 角色到 ``neutron`` 用户：

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

(注：该命令无输出)

- (4) 创建 ``neutron`` 服务实体：

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

- (5) 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
```

(2) 编辑 ``/etc/neutron/neutron.conf`` 文件并完成如下操作：

I、 ``[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。)

II、 ``[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。

III、 “``[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。

IV、 “``[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。

V、 ``[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。

(3) 编辑 ``/etc/neutron/plugins/ml2/ml2_conf.ini`` 文件并完成以下操作：

I、 在 ``[ml2]`` 部分

```
[ml2]
```

```
...
```

```
type_drivers = flat,vlan,vxlan
```

```
tenant_network_types = vxlan
```

```
mechanism_drivers = linuxbridge,l2population
```

```
extension_drivers = port_security
```

II、 ``[ml2_type_flat]`` 部分

```
[ml2_type_flat]
```

```
...
```

```
flat_networks = provider
```

III、 在 ``[ml2_type_vxlan]`` 部分

```
[ml2_type_vxlan]
```

```
...
```

```
vni_ranges = 1:1000
```

IV、 ``[securitygroup]`` 部分

```
[securitygroup]
```

```
...
```

```
enable_ipset = True
```

(4) 编辑 ``/etc/neutron/plugins/ml2/linuxbridge_agent.ini`` 文件并且完成以下操作：

I、 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。

II、 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。

III、 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
```

(5) 编辑 ``/etc/neutron/l3_agent.ini`` 文件并完成以下操作：

I、 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
```

(6) 编辑 ``/etc/neutron/dhcp_agent.ini`` 文件并完成下面的操作：

I、 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
```

(7) 编辑 ``/etc/neutron/metadata_agent.ini`` 文件并完成以下操作：

I、 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。

(8) 编辑 ``/etc/nova/nova.conf`` 文件并完成以下操作：

I、 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
```

(9) 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
```

(10) 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
```

(2) 编辑 ``/etc/neutron/neutron.conf`` 文件并完成如下操作:

- I、 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。
- II、 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
```

- III、 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

```

- (3) 编辑 ``/etc/neutron/plugins/ml2/linuxbridge_agent.ini`` 文件并且完成以下操作：

I、 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。

II、 在 ``[vxlan]`` 部分，禁止 VXLAN 覆盖网络：

```

[vxlan]
enable_vxlan = True
local_ip = 192.168.0.101
l2_population = True

```

III、 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

```

- (4) 编辑 ``/etc/nova/nova.conf`` 文件并完成下面的操作：

I、 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

```

- (5) 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
```

- (2) 列出代理以验证启动 neutron 代理是否成功:

```
# neutron agent-list
```

```
$ neutron agent-list
```

id	agent_type	host	alive	admin_state_up	binary
08905043-5010-4b87-bba5-aedb1956e27a	Linux bridge agent	compute1	:-)	True	neutron-linuxbridg
27eee952-a748-467b-bf71-941e89846a92	Linux bridge agent	controller	:-)	True	neutron-linuxbridg
830344ff-dc36-4956-84f4-067af667a0dc	L3 agent	controller	:-)	True	neutron-l3-agent
dd3644c9-1a3a-435a-9282-eb306b4b0391	DHCP agent	controller	:-)	True	neutron-dhcp-agent
f49a4b81-afd6-4b3d-b923-66c8f0517099	Metadata agent	controller	:-)	True	neutron-metadata-a

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
```

- (2) 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)

- I、在 controller 节点上配置仪表盘以使用 OpenStack 服务:

```
OPENSTACK_HOST = "controller"
```

- II、允许所有主机访问仪表板:

```
ALLOWED_HOSTS = ['*', ]
```

- III、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。）

IV、 启用第 3 版认证 API：

```
OPENSTACK_KEYSTONE_URL = "http://%s:5000/v3" % OPENSTACK_HOST
```

V、 启用对域的支持

```
OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True
```

VI、 配置 API 版本：

```
OPENSTACK_API_VERSIONS = {
    "identity": 3,
    "image": 2,
    "volume": 2,
}
```

VII、 通过仪表盘创建用户时的默认域配置为 default：

```
OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = "default"
```

VIII、 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）

(3) 编辑文件 /etc/apache2/conf-available/openstack-dashboard.conf 并完成如下动作：

I、 在 WSGIProcessGroup horizon 一行下添加一行

```
WSGIApplicationGroup %{GLOBAL}
```

```
WSGIScriptAlias /horizon /usr/share/openstack-dashboard/openstack_dashboard/wsgi/django.wsgi process-group=horizon
WSGIDaemonProcess horizon user=horizon group=horizon processes=3 threads=10 display-name=%{GROUP}
WSGIProcessGroup horizon
WSGIApplicationGroup %{GLOBAL}

Alias /static /usr/share/openstack-dashboard/openstack_dashboard/static/
Alias /horizon/static /usr/share/openstack-dashboard/openstack_dashboard/static/

<Directory /usr/share/openstack-dashboard/openstack_dashboard/wsgi>
    Order allow,deny
    Allow from all
</Directory>
```

(4) 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
```

- (2) 创建网络：

```
# neutron net-create --shared --provider:physical_network provider \  
--provider:network_type flat provider
```

- (3) 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 \  
provider
```

```
--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。

DHCP服务器 ☒ ON ?

地址池开始地址	192.168.1.100
地址池结束地址	192.168.1.199

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
```

- (2) 生成一个键值对

I、 导入租户``demo``的凭证

```
# . demo-openrc
```

II、 生成和添加秘钥对：

```
# ssh-keygen -q -N ""
```

```
# openstack keypair create --public-key /root/.ssh/id_rsa.pub mykey
```

III、 验证公钥的添加是否成功：

```
# openstack keypair list
```

```
+-----+
| Name | Fingerprint |
+-----+
| mykey | ee:3d:2e:97:d4:e2:6a:54:6d:0d:ce:43:39:2c:ba:4d |
+-----+
```

- (3) 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
```

- (4) 列出可用网络：

```
# . demo-openrc
```

```
# openstack network list
```

```
+-----+-----+-----+
| ID | Name | Subnets |
+-----+-----+-----+
| 4716ddfe-6e60-40e7-b2a8-42e57bf3c31c | selfservice | 2112d5eb-f9d6-45fd-906e-7cabd38b7c7c |
| b5b6993c-ddf9-40e7-91d0-86806a42edb8 | provider | 310911f6-acf0-4a47-824e-3032916582ff |
+-----+-----+-----+
```

(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。)

(5) 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
```

(6) 检查实例的状态：

```
# openstack server list
```

ID	Name	Status	Networks
181c52ba-aebc-4c32-a97d-2e8e82e4eaf	provider-instance	ACTIVE	provider=203.0.113.103

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
```

其中 192.168.1.151 为实例的 ip 地址

(2) 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>

ubuntu[®] OpenStack Dashboard

Log in

Domain

default

用户名

demo

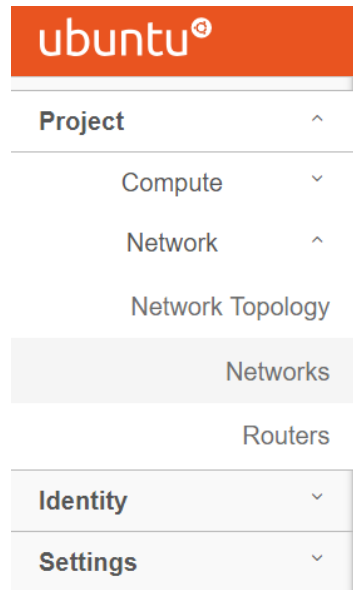
密码

.....

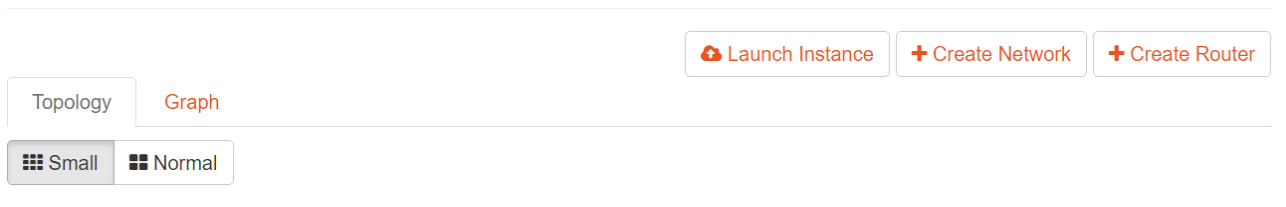
Connect

输入如图所示账号，密码为 1234。

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



(3) 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。



Create Network

Network

Subnet

Subnet Details

Network Name

net1

Admin State ?

UP

☐ Shared ?

☒ Create Subnet

Create a new network. In addition, a subnet associated with the network can be created in the following steps of this wizard.

Cancel

« Back

Next »

Create Network

Network

Subnet

Subnet Details

Subnet Name

net1sub

Network Address ?

192.168.5.0/24

IP Version

IPv4

Gateway IP ?

192.168.5.1

☐ Disable Gateway

Creates a subnet associated with the network. You need to enter a valid "Network Address" and "Gateway IP". If you did not enter the "Gateway IP", the first value of a network will be assigned by default. If you do not want gateway please check the "Disable Gateway" checkbox. Advanced configuration is available by clicking on the "Subnet Details" tab.

Cancel

« Back

Next »

Create Network

Network

Subnet

Subnet Details

☒ Enable DHCP

Specify additional attributes for the subnet.

Allocation Pools ?

192.168.5.2,192.168.5.254

DNS Name Servers ?

219.223.254.14

Cancel

« Back

Create

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

Network Topology

Topology

Graph

Launch Instance

Create Network

Create Router

Small

Normal

Create Router

Router Name

route1

Admin State

UP

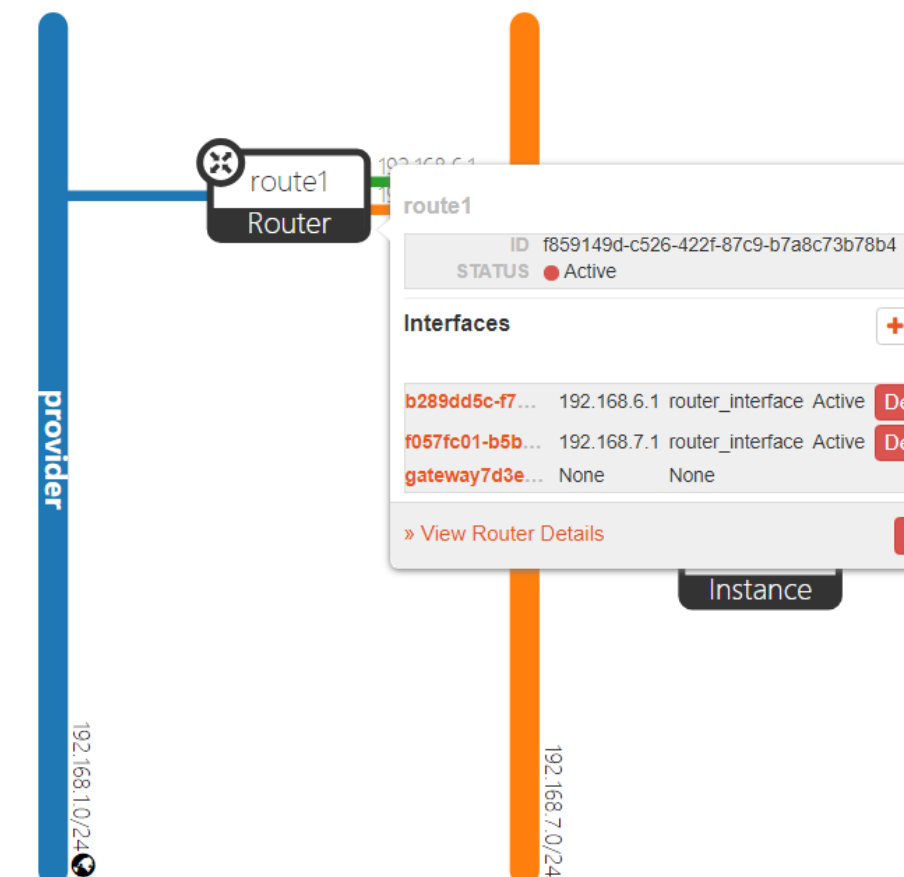
External Network

provider

Cancel

Create Router

- (2) 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 。



route1

Clear Gateway

Overview

Interfaces

Static Routes

+ Add Interface

🗑 Delete Interfaces

Add Interface



Subnet *

net1: 192.168.7.0/24 (net1sub)



IP Address (optional) ?

Router Name *

route1

Router ID *

f859149d-c526-422f-87c9-b7a8c73b78b4

Description:

You can connect a specified subnet to the router.

The default IP address of the interface created is a gateway of the selected subnet. You can specify another IP address of the interface here. You must select a subnet to which the specified IP address belongs to from the above list.

Cancel

Submit