# （一）Horizon服务

## 1.概述

OpenStack仪表板Dashboard服务的项目名称是Horizon，它所需的唯一服务是身份服务keystone，开发语言是python的web框架Django。

## 2.语言环境

Python 2.7、3.6或3.7

Django 1.11、2.0和2.2

Django 2.0和2.2支持在Train版本中处于试验阶段。

Ussuri发行版（Train发行版之后的下一个发行版）将使用Django 2.2作为主要的Django版本。Django 2.0支持将被删除。

## 3.可访问的keystone endpoint

keystone的endpoint是可以访问的。

## 4.其他服务都是可选的

从Stein版本开始，Horizon支持以下服务：

cinder：块状存储

glance：镜像管理

neutron：网络

nova：计算

swift：对象存储

如果已配置好服务keystone的endpoint，那么Horizon将对其进行检测并自动启用其支持。 Horizon还通过插件支持许多其他OpenStack服务。 在

# （二）Horizon服务安装

## 1.安装节点选择

计算节点（compute）上安装仪表板服务horizon 由于horizon运行需要apache，为了不影响控制节点上的keystone等其他服务使用的apache，故在计算节点上安装。安装之前确认以前安装的服务是否正常启动。

## 2.软件安装

安装dashboard服务

[root@compute ~]# yum install openstack-dashboard -y

安装httpd服务

yum -y install httpd

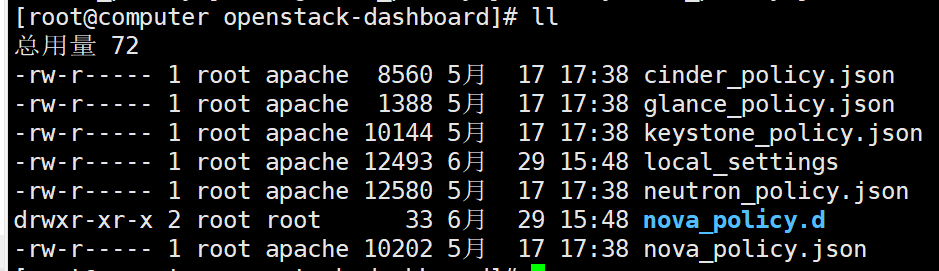
设置httpd服务的自动启动和开始：

systemctl enable httpd.service

systemctl start httpd.service

## 3. 修改local\_setting本地控制台的配置文件

进入 /etc/openstack-dashboard/目录下，查看其中的配置文件：



修改/etc/openstack-dashboard/local\_settings本地控制台的配置文件：

#修改的内容如下：

#修改local\_setting本地控制台的配置文件

#使用Python导入一个模块

import os

from django.utils.translation import ugettext\_lazy as \_

from openstack\_dashboard.settings import HORIZON\_CONFIG

#不开启调式

DEBUG = False

#只允许通过列表中指定的域名访问dashboard；允许通过指定的IP地址及域名#访问dahsboard；['\*']表示允许所有域名

ALLOWED\_HOSTS = ['\*']

LOCAL\_PATH = '/tmp'

#这是一个自定义的key，自动生成的。当多个dashboard实例分布在不同的机器上（通常在#一个负载均衡器）。你必须确保会话获取所有请求路由到相同的dashboard实例或你为它们设置相同的secret\_key。

SECRET\_KEY='9665a3cff2428daec89c'

#指定session引擎

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

#指定memcache地址及端口

CACHES = {

'default': {

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

'LOCATION': '192.168.112.146:11211',

}

}

#以下配置session信息存放到memcache中；session信息不仅可以存放到memcache中，也可以存放到其他地方

EMAIL\_BACKEND = 'django.core.mail.backends.console.EmailBackend'

#Controller节点IP

OPENSTACK\_HOST = "192.168.112.146"

#Keystone的URL

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

#让dashboard支持域

OPENSTACK\_KEYSTONE\_MULTIDOMAIN\_SUPPORT = True

#配置openstack的API版本

OPENSTACK\_API\_VERSIONS = {

"identity": 3,

"image": 2,

"volume": 3,

}

Keystone的默认Domain为Default

OPENSTACK\_KEYSTONE\_DEFAULT\_DOMAIN = "Default"

keystone的默认角色为user

OPENSTACK\_KEYSTONE\_DEFAULT\_ROLE = "user"

neutron网络配置

OPENSTACK\_NEUTRON\_NETWORK = {

'enable\_auto\_allocated\_network': False,

'enable\_distributed\_router': False,

'enable\_fip\_topology\_check': False,

'enable\_ha\_router': False,

'enable\_lb': False,

'enable\_firewall': False,

'enable\_vpn': False,

'enable\_ipv6': True,

'enable\_quotas': True,

'enable\_rbac\_policy': True,

'enable\_router': True,

'default\_dns\_nameservers': [],

'supported\_provider\_types': ['\*'],

'segmentation\_id\_range': {},

'extra\_provider\_types': {},

'supported\_vnic\_types': ['\*'],

'physical\_networks': [],

}

#定义使用的网络类型，[\*]表示

TIME\_ZONE = "Asia/Shanghai" #156行修改

以下两步官方文档中没有，但是需要执行，否则dashboard打不开或显示不对

## 4.重建apache的dashboard配置文件

重新生成openstack-dashboard.conf

cd /usr/share/openstack-dashboard

python manage.py make\_web\_conf --apache > /etc/httpd/conf.d/openstack-dashboard.conf

5.备用机制

若出现不能正常访问，请操作以下步骤： 建立策略文件（policy.json）的软链接，否则登录到dashboard将出现权限错误和显示混乱

ln -s /etc/openstack-dashboard /usr/share/openstack-dashboard/openstack\_dashboard/conf

## 5.重启Apache服务

( 由于dashborad会重新复制代码文件，重启apache会比较慢 ）

[root@compute ~]# systemctl enable httpd.service

[root@compute01 ~]# systemctl restart httpd.service

由于dashboard的运行机制是把网站下的所有文件删除之后再重新复制，所以重启httpd需要等待一段时间。

## 6.重新启动控制节点（controller）上的memcache服务

[root@controller ~]# systemctl restart memcached.service

# （三）验证Horizon服务

连接到Compute节点的IP：192.168.112.145



在登录页面依次填写：“域：default、用户名：admin、密码:ADMIN\_PASS”（在~.bashrc中已定义）；完成后，进行登陆。



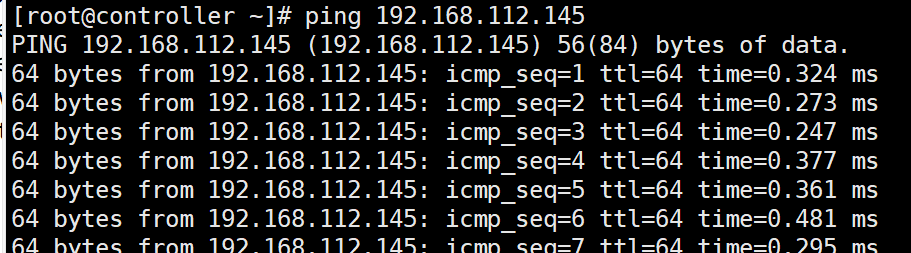
# （四）DashBoard中创建实例测试

## 1、检查各个节点间的网络通讯

在控制节点执行ping

ping compute

ping 192.168.112.145



可以看到，Controller节点和Compute节点通信没有问题，由于Linux中的ping命令会默认持续进行，通过Ctrl + Z 暂停放入后台。然后通过jobs -l查询使用的进程号，再通过kill -9 杀死指定的进程。

## 2、删除NetworkManager软件包 （Controller和Compute）

在控制节点和计算节点都执行

yum remove NetworkManager -y

## 3. controller节点创建网络

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

--provider:network\_type flat WAN

参数说明：

--**share 指明所有项目都可以使用这个网络，否则只有创建者能使用**

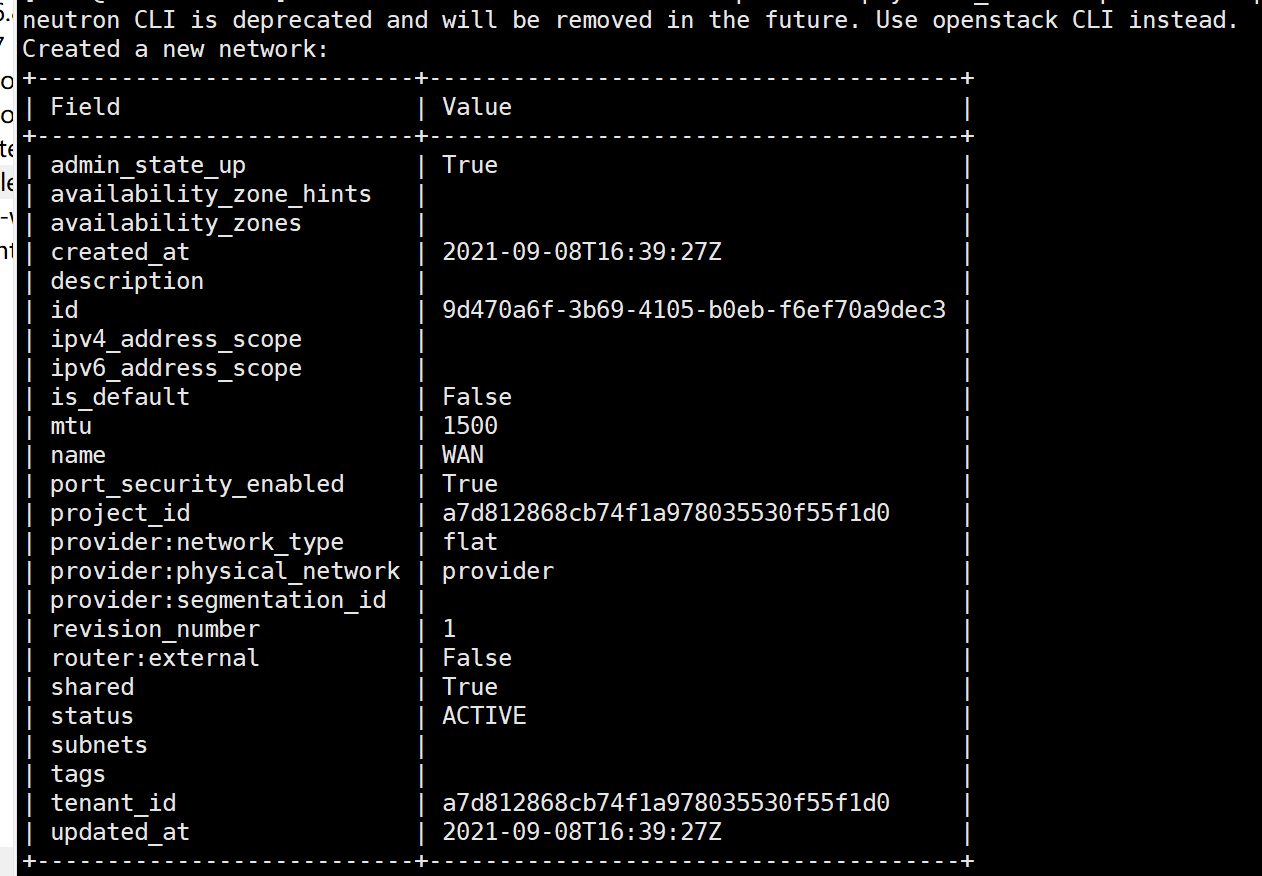
--**external 指明是外部网络**

--**provider:physical-network provider，**指明物理网络的提供者，与下面neutron的配置文件对应，其中provider是标签，可以更改为其他，但是2个地方必须要统一。

[ml2\_type\_flat]

flat\_networks = provider

--provider:network\_type flat，指明这里创建的网络是flat类型，即实例连接到此网络时和物理网络是在同一个网段，无vlan等功能。



## 4. controller节点创建子网

neutron subnet-create --name subnet-wan --allocation-pool

start=192.168.112.200,end=192.168.112.230 --dns-nameserver 223.5.5.5

--gateway 192.168.112.254 WAN 192.168.112.0/24

参数说明：

--network 指明父网络

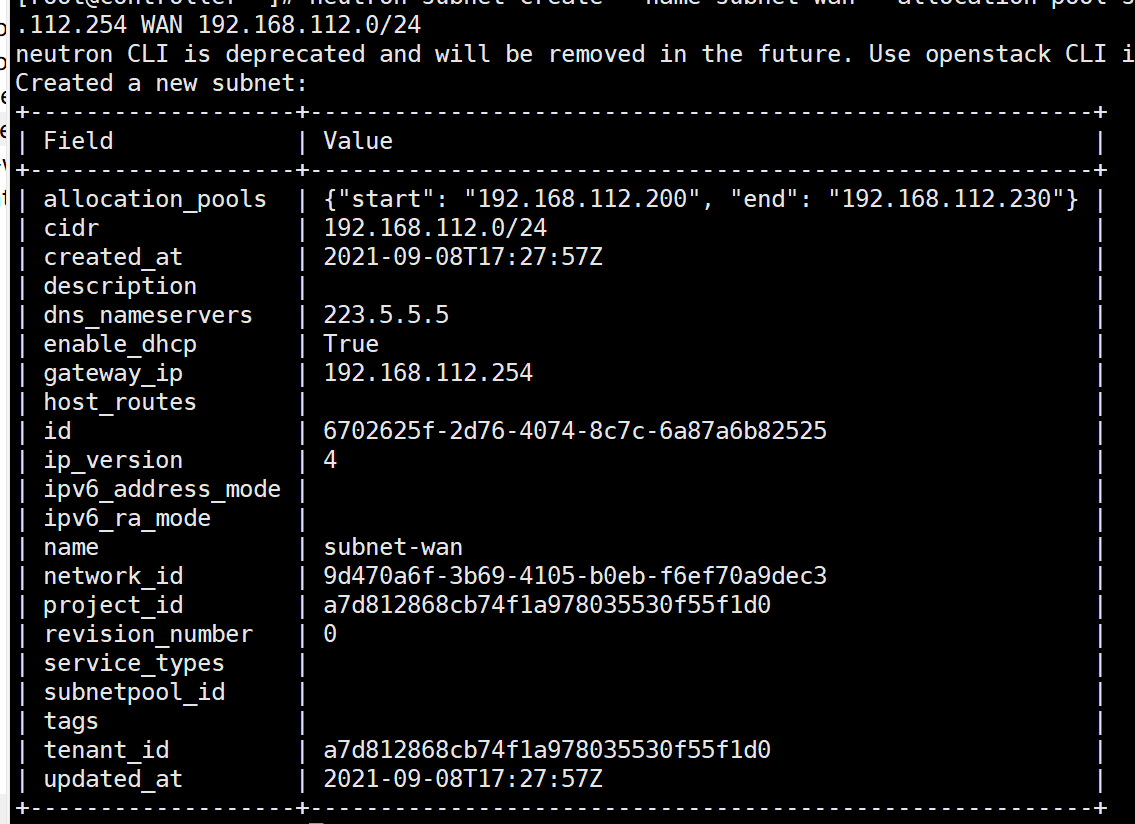
--allocation-pool start=192.168.112.200,end=192.168.112.230 指明子网起始地址和终止地址

--dns-nameserver 指明dns服务器

--gateway 指明网关地址

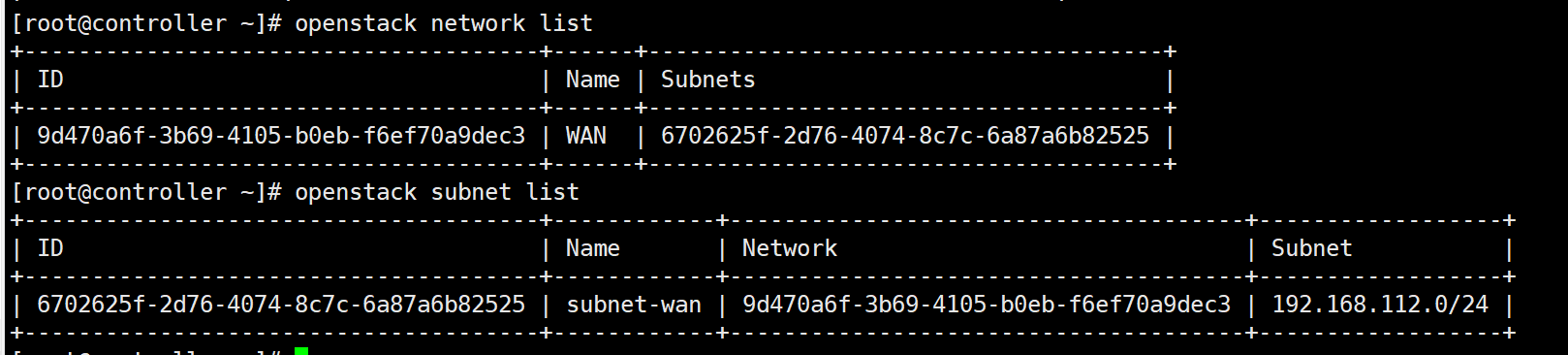
--subnet-range 指明子网网段

vm-subnetwork 子网名称



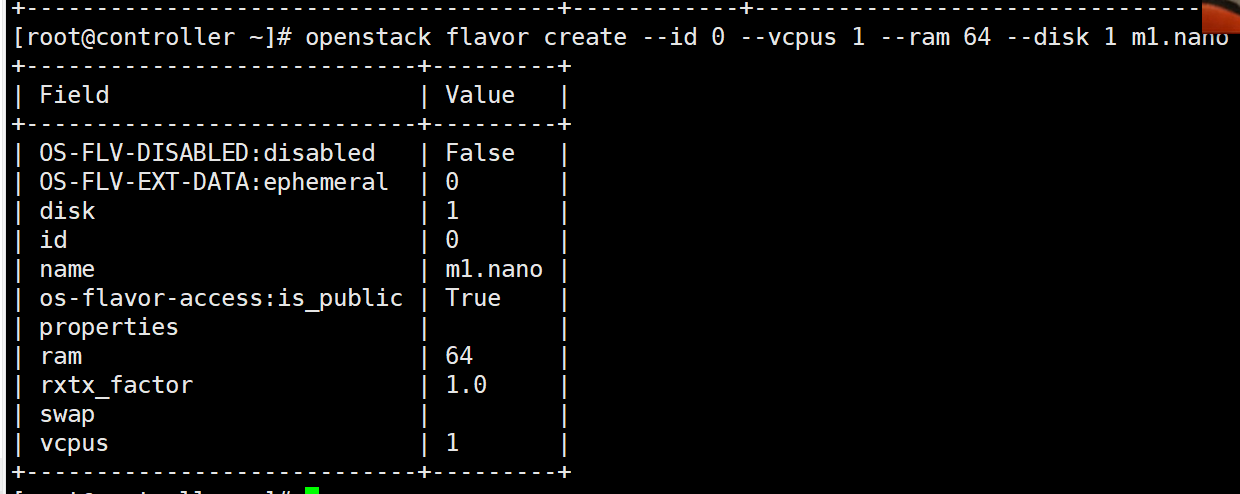
## 5. 检查网络配置

在控制节点执行以下操作



## 6. 控制节点创建硬件配置方案

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



## 7. dashboard创建实例

直接在dashboard界面中直接点击创建实例来创建实例，配置镜像，网络，硬件配置。



## 8.连接实例控制台

启动实例后，进入控制台可以直接进入该实例虚机中。

