完整部署 CentOS7.2+OpenStack+kvm 云平台环境（1）--基础环境搭建

公司在 IDC 机房有两台很高配置的服务器，计划在上面部署 openstack 云平台虚拟化环境，用于承载后期开发测试和其他的一些对内业务。
以下对 openstack 的部署过程及其使用做一详细介绍，仅仅依据本人实际经验而述，如有不当，敬请指出~

****
****

1 OpenStack 介绍

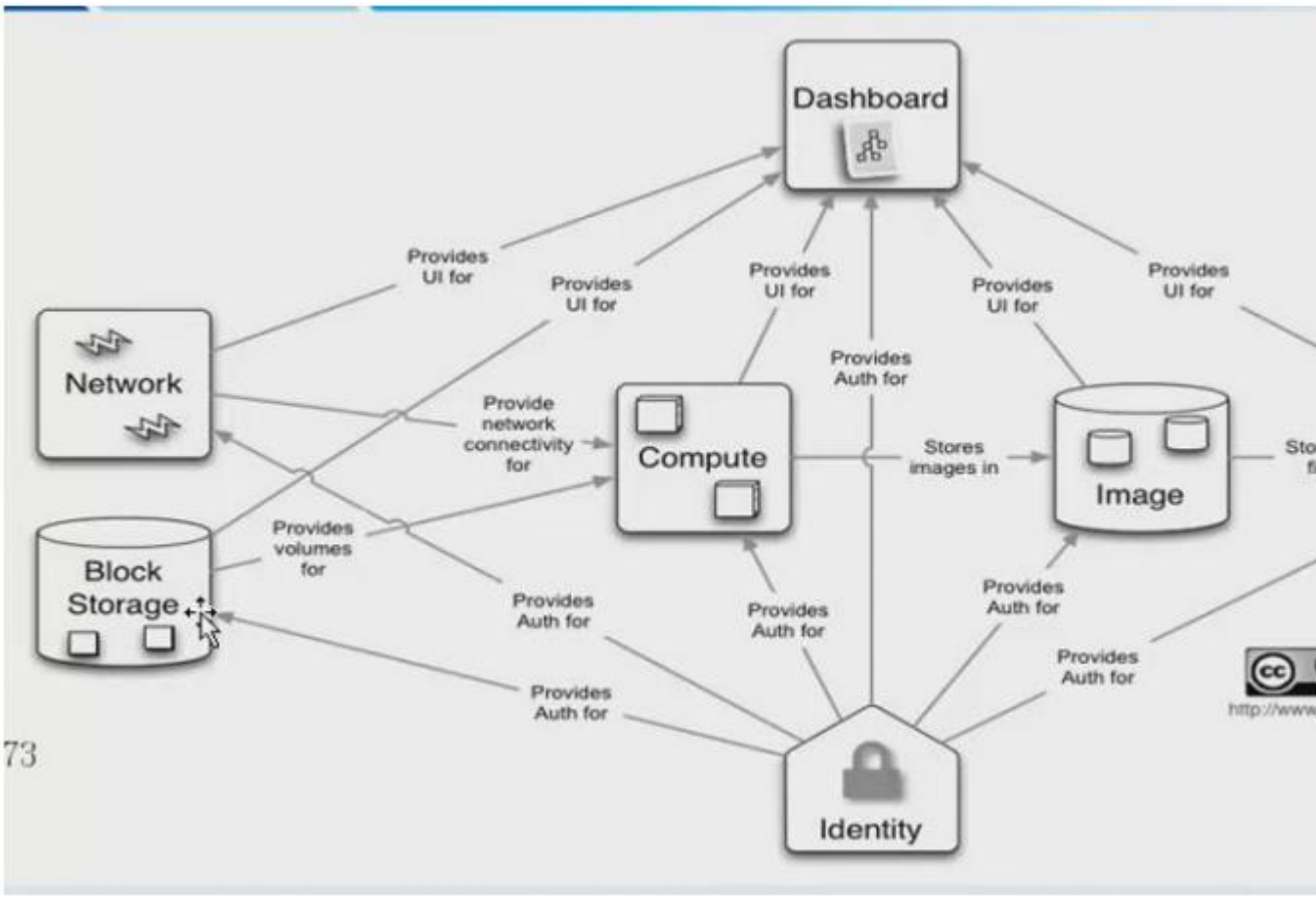
1.1 百度百科

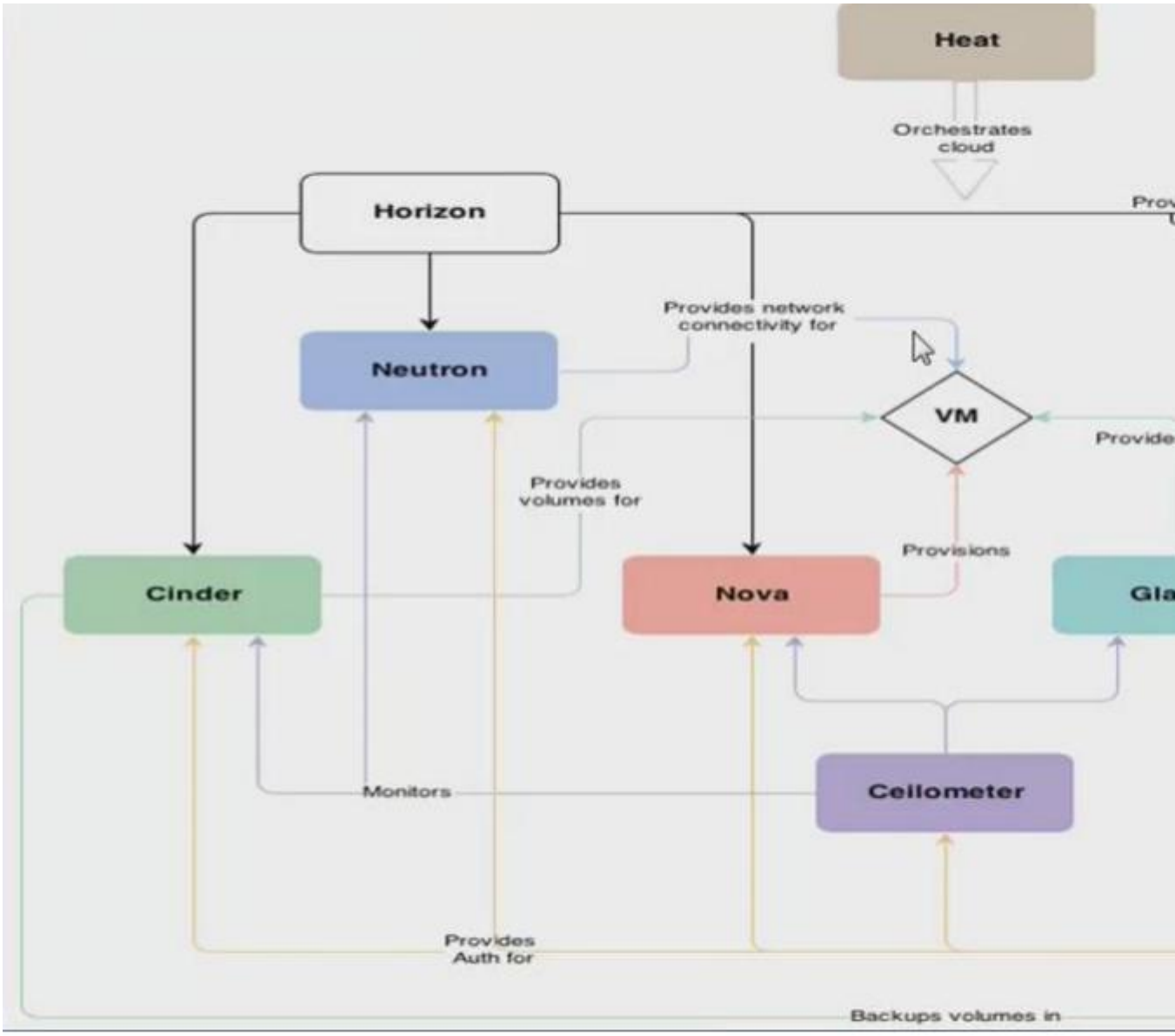
OpenStack 是一个由 NASA （美国国家航空航天局）和 Rackspace 合作研发并发起的，以 Apache
许可证授权的自由软件和开放源代码项目。

1.2 版本历史

Release Name	Release Date	Included Components
Austin	21 October 2010	Nova, Swift
Bexar	3 February 2011	Nova, Glance, Swift
Cactus	15 April 2011	Nova, Glance, Swift
Diablo	22 September 2011	Nova, Glance, Swift
Essex	5 April 2012	Nova, Glance, Swift, Horizon, Keystone
Folsom	27 September 2012	Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder
Grizzly	4 April 2013	Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder
Havana	17 October 2013	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder
Icehouse	April 2014	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, (More to be adde

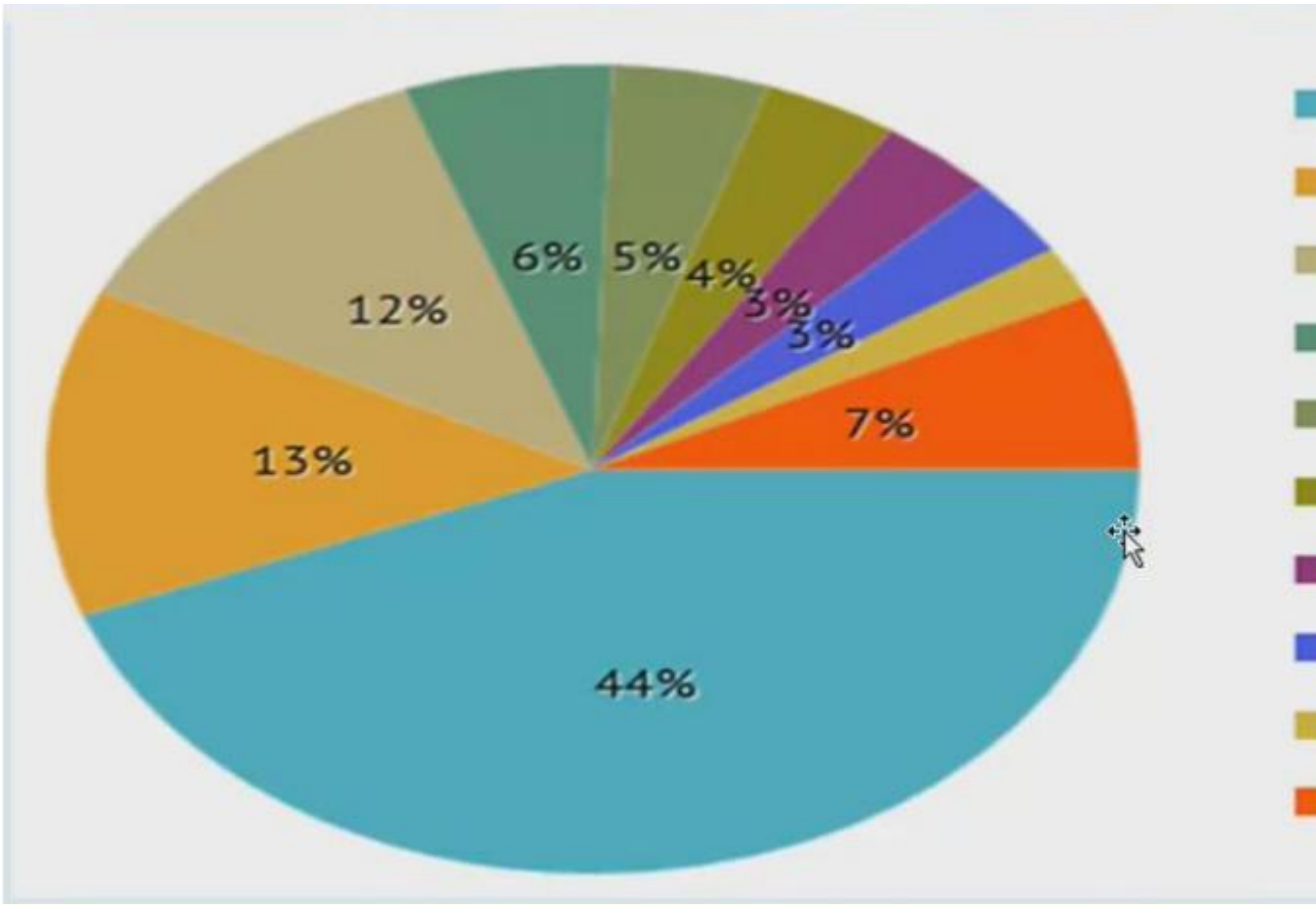
1.3 openstack 架构概念





1.4 openstack 各个服务名称对应

服务名称	项目名称	描述
Dashboard	Horizon	基于OpenStack API接口使用django开发
Compute	Nova	通过虚拟化技术提供计算资源
Networking	Neutron	实现了虚拟机的网络资源管
Storage（存储）		
Object Storage	Swift	对象存储，适用于“一次写入、多
Block Storage	Cinder	块存储，提供存储资源池
Shared Services（共享服务）		
Identity Service	Keystone	认证管理
Image Service	Glance	提供虚拟镜像的注册和存储
Telemetry	Ceilometer	提供监控和数据采集、计量
Higher-level services（高层服务）		
Orchestration	Heat	自动化部署的组件
Database Service	Trove	提供数据库应用服务



****

以下安装部署已经过测试，完全通过！

建议在物理机上部署 openstack，并且是 centos7 或 ubuntu 系统下，centos6x 的源里已不支持 openstack 部分组件下载了。

2 环境准备

openstack 主机名不能改，装的时候是什么就是什么， 运维标准化。

1、 CentOS 7.2 系统 2 台

node1 即作为控制节点，也作为计算节点；(即可以单机部署，单机部署时则下面记录的控制节点和计算节点的操作步骤都要在本机执行下)

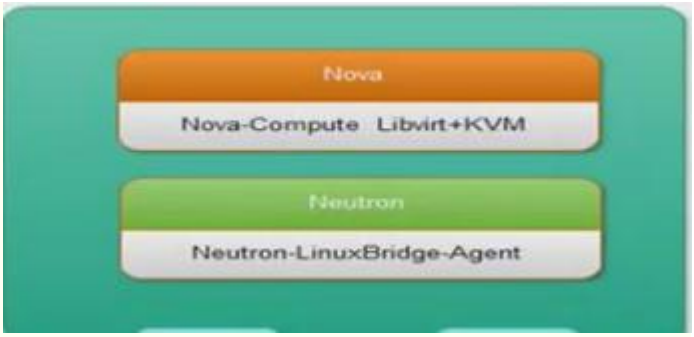
node2 就只是计算节点

控制节点去操控计算节点，计算节点上可以创建虚拟机

linux-node1.openstack　192.168.1.17 网卡 NAT em2 （外网 ip 假设是 58.68.250.17）
(em2 是内网网卡，下面 neutron 配置文件里会设置到)
linux-node2.openstack　192.168.1.8　网卡 NAT em2

控制节点：
linux-node1.openstack　192.168.1.17

计算节点：linux-node2.openstack 192.168.1.8



2.域名解析和关闭防火墙 （控制节点和计算节点都做）

/etc/hosts　#主机名一开始设置好，后面就不能更改了，否则就会出问题！这里设置好 ip 与主机名的对应关系

192.168.1.17 linux-node1.openstack
192.168.1.8 linux-node2.openstack

关闭 selinux
sed -i 's#SELINUX=enforcing#SELINUX=disabled#g' /etc/sysconfig/selinux
setenforce 0
关闭 iptables
systemctl start firewalld.service
systemctl stop firewalld.service
systemctl disable firewalld.service

3 安装配置 OpenStack

官方文档 http://docs.openstack.org/

3.1 安装软件包

linux-node1.openstack 安装

****

#Base
yum install -y
http://dl.fedoraproject.org/pub/epel/7/x86_64/e/epel-release-7-8.noarch.rpm
yum install -y centos-release-openstack-liberty
yum install -y python-openstackclient

##MySQL
yum install -y mariadb mariadb-server MySQL-python

##RabbitMQ
yum install -y rabbitmq-server

##Keystone
yum install -y openstack-keystone httpd mod_wsgi memcached python-memcached

##Glance
yum install -y openstack-glance python-glance python-glanceclient

##Nova
yum install -y openstack-nova-api openstack-nova-cert openstack-nova-conductor openstack-nova-console openstack-nova-novncproxy openstack-nova-scheduler python-novaclient

##Neutron linux-node1.example.com
yum install -y openstack-neutron openstack-neutron-ml2 openstack-neutron-linuxbridge python-neutronclient ebtables ipset

##Dashboard
yum install -y openstack-dashboard

##Cinder
yum install -y openstack-cinder python-cinderclient

****

linux-node2.openstack 安装

##Base
yum install -y http://dl.fedoraproject.org/pub/epel/7/x86_64/e/epel-release-7-8.noarch.rpm
yum install centos-release-openstack-liberty
yum install python-openstackclient

##Nova linux-node2.openstack
yum install -y openstack-nova-compute sysfsutils

##Neutron linux-node2.openstack
yum install -y openstack-neutron openstack-neutron-linuxbridge ebtables ipset

##Cinder
yum install -y openstack-cinder python-cinderclient targetcli python-oslo-policy

****

3.2 设置时间同步、 关闭 selinux 和 iptables

在 linux-node1 上配置（ 只有 centos7 能用， 6 还用 ntp）

```
[root@linux-node1 ~]# yum install -y chrony
[root@linux-node1 ~]# vim /etc/chrony.conf
allow 192.168/16 #允许那些服务器和自己同步时间
[root@linux-node1 ~]# systemctl enable chronyd.service     #开机启动
[root@linux-node1 ~]# systemctl start chronyd.service
[root@linux-node1 ~]# timedatectl set-timezone Asia/Shanghai      #设置时区
[root@linux-node1 ~]# timedatectl status
Local time: Fri 2016-08-26 11:14:19 CST
Universal time: Fri 2016-08-26 03:14:19 UTC
RTC time: Fri 2016-08-26 03:14:19
Time zone: Asia/Shanghai (CST, +0800)
NTP enabled: yes
NTP synchronized: yes
RTC in local TZ: no
DST active: n/a
```

在 linux-node2 上配置

```
[root@linux-node2 ~]# yum install -y chrony
[root@linux-node2 ~]# vim /etc/chrony.conf
server 192.168.1.17 iburst #只留一行
[root@linux-node2 ~]# systemctl enable chronyd.service
[root@linux-node2 ~]# systemctl start chronyd.service
[root@linux-node2 ~]# timedatectl set-timezone Asia/Shanghai
[root@linux-node2 ~]# chronyc sources
```

3.3 安装及配置 mysql

```
[root@linux-node1 ~]# cp /usr/share/mysql/my-medium.cnf
/etc/my.cnf                    #或者是/usr/share/mariadb/my-medium.cnf
[mysqld]
default-storage-engine = innodb
innodb_file_per_table
collation-server = utf8_general_ci
init-connect = 'SET NAMES utf8'
character-set-server = utf8
[root@linux-node1 ~]# systemctl enable
mariadb.service                                    #Centos7 里面 mysql 叫
maridb
[root@linux-node1 ~]# ln -s '/usr/lib/systemd/system/mariadb.service'
'/etc/systemd/system/multi-user.target.wants/mariadb.service'
[root@linux-node1 ~]# mysql_install_db --datadir="/var/lib/mysql"
```

--user="mysql"　　#初始化数据库
[root@linux-node1 ~]# systemctl start mariadb.service
[root@linux-node1 ~]#
mysql_secure_installation　　#设置密码及初始化
密码 123456，一路 y 回车

创建数据库
[root@openstack-server ~]# mysql -p123456
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 5579
Server version: 5.5.50-MariaDB MariaDB Server

Copyright (c) 2000, 2016, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
MariaDB [(none)]> CREATE DATABASE keystone;
MariaDB [(none)]> GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' IDENTIFIED BY 'keystone';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' IDENTIFIED BY 'keystone';
MariaDB [(none)]> CREATE DATABASE glance;
MariaDB [(none)]> GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' IDENTIFIED BY 'glance';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' IDENTIFIED BY 'glance';
MariaDB [(none)]> CREATE DATABASE nova;
MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' IDENTIFIED BY 'nova';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' IDENTIFIED BY 'nova';
MariaDB [(none)]> CREATE DATABASE neutron;
MariaDB [(none)]> GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost' IDENTIFIED BY 'neutron';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%' IDENTIFIED BY 'neutron';
MariaDB [(none)]> CREATE DATABASE cinder;
MariaDB [(none)]> GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' IDENTIFIED BY 'cinder';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'%' IDENTIFIED BY 'cinder';
MariaDB [(none)]> flush privileges;
MariaDB [(none)]> show databases;

```
+--------------------+
| Database |
+--------------------+
| information_schema |
| cinder |
| glance |
| keystone |
| mysql |
| neutron |
| nova |
| performance_schema |
+--------------------+
8 rows in set (0.00 sec)

MariaDB [(none)]>
```

参考另一篇博客：http://www.cnblogs.com/kevingrace/p/5811167.html
修改下 mysql 的连接数，否则 openstack 后面的操作会报错:"ERROR 1040 (08004): Too many connections "

3.4 配置 rabbitmq

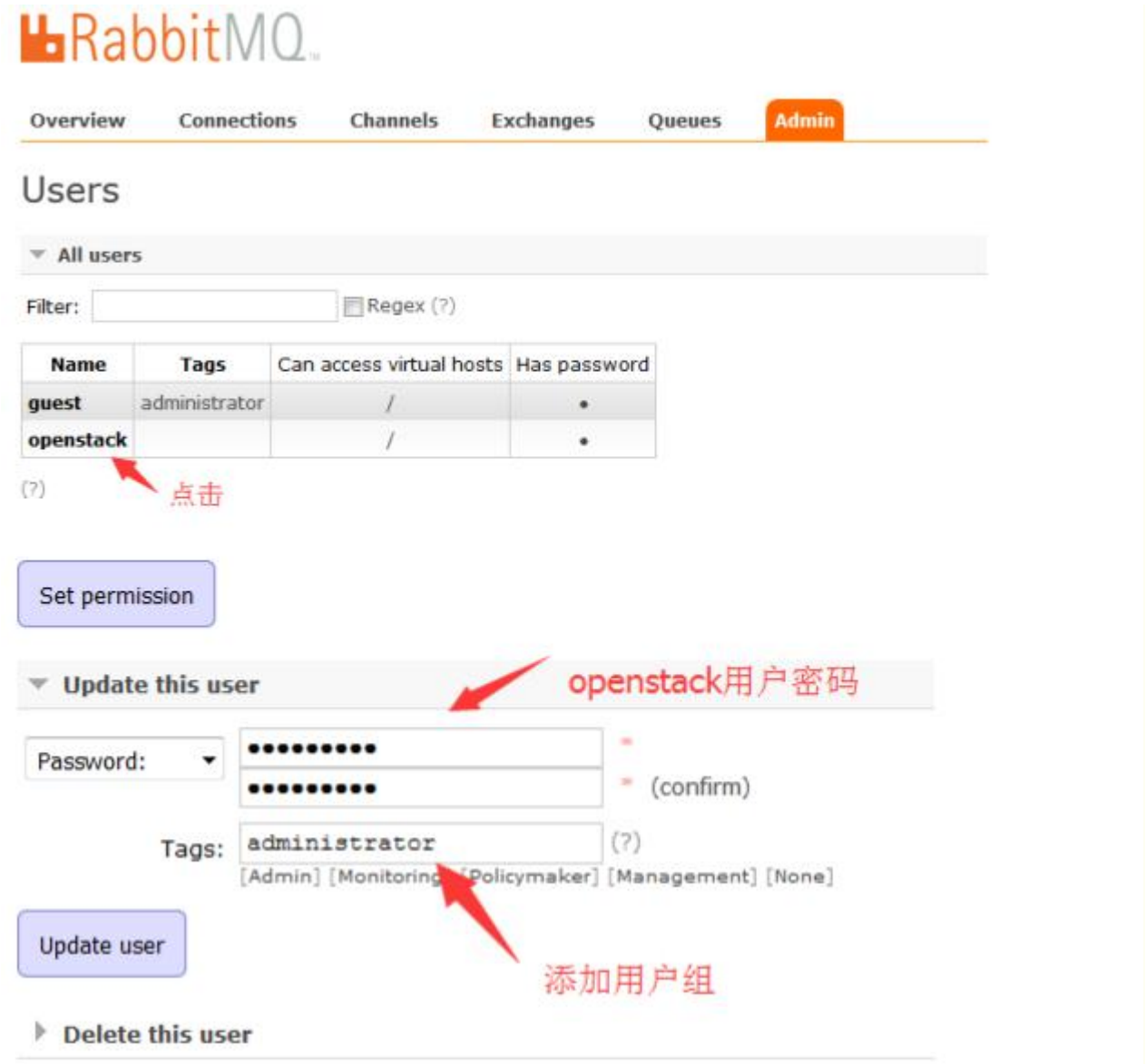
MQ 全称为 Message Queue，消息队列（ MQ）是一种应用程序对应用程序的通信方法。应用程序通过读写出入队列的消息（针对应用程序的数据）来通信，而无需专用连接来链接它们。
消 息传递指的是程序之间通过在消息中发送数据进行通信，而不是通过直接调用彼此来通信，直接调用通常是用于诸如远程过程调用的技术。排队指的是应用程序通过 队列来通信。
队列的使用除去了接收和发送应用程序同时执行的要求。
RabbitMQ 是一个在 AMQP 基础上完整的，可复用的企业消息系统。他遵循 Mozilla Public License 开源协议。
启动 rabbitmq， 端口 5672，添加 openstack 用户

```
[root@linux-node1 ~]# systemctl enable rabbitmq-server.service
[root@linux-node1 ~]# ln -s '/usr/lib/systemd/system/rabbitmq-server.service'
'/etc/systemd/system/multi-user.target.wants/rabbitmq-server.service'
[root@linux-node1 ~]# systemctl start rabbitmq-server.service
[root@linux-node1 ~]# rabbitmqctl add_user openstack
openstack                                  #添加用户及密码
[root@linux-node1 ~]# rabbitmqctl set_permissions openstack ".*" ".*"
".*"                                       #允许配置、写、读访问 openstack
[root@linux-node1 ~]# rabbitmq-plugins list                                  #查看支持的插
```

件
.....
[ ] rabbitmq_management 3.6.2 #使用此插件实现 web 管理
.....
[root@linux-node1 ~]# rabbitmq-plugins enable rabbitmq_management #启动插件
The following plugins have been enabled:
mochiweb
webmachine
rabbitmq_web_dispatch
amqp_client
rabbitmq_management_agent
rabbitmq_management
Plugin configuration has changed. Restart RabbitMQ for changes to take effect.
[root@linux-node1 ~]# systemctl restart rabbitmq-server.service

[root@linux-node1 ~]#lsof -i:15672

访问 RabbitMQ,访问地址是 http://58.68.250.17:15672
默认用户名密码都是 guest，浏览器添加 openstack 用户到组并登陆测试，连不上情况一般是防火墙没有关闭所致！



之后退出使用 openstack 登录

如何使用 zabbix 监控？

左下角有 HTTP API 的介绍，可以实现 zabbix 的监控

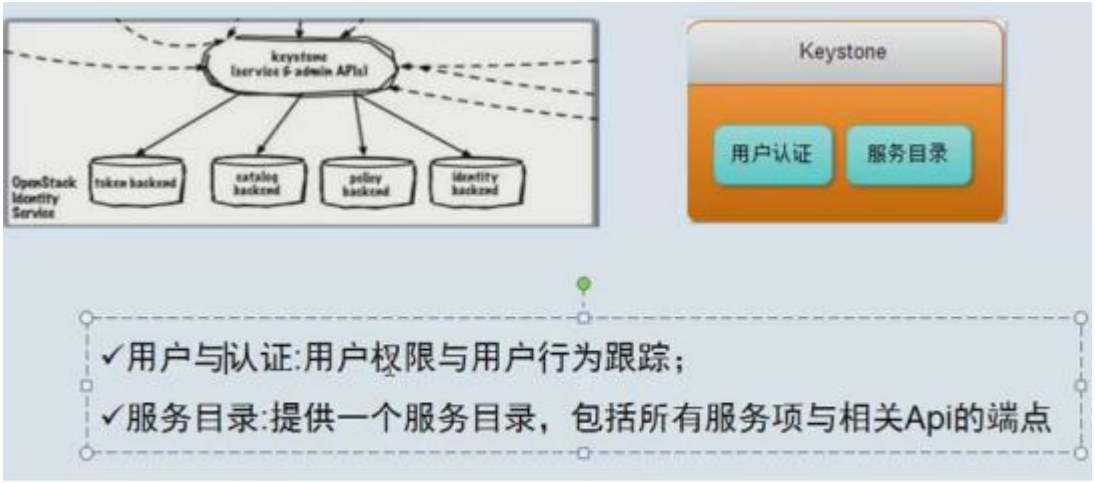
****

以上完成基础环境的配置，下面开始安装 openstack 的组件

3.5 配置 Keystone 验证服务

所有的服务，都需要在 keystone 上注册

3.5.1 Keystone 介绍



OpenStack中的概念	跟宾馆的类比
User	住宾馆的人
Credentials	开启房间的钥匙
Authentication	宾馆为了拒绝不必要的人进入宾馆，专门设置的机制，只有拥有要是的人才能进入
Token	也是一种钥匙，不过有点特别
Tenant	宾馆
Service	宾馆可以提供的服务类别，比如饮食类，娱乐类
Endpoint	具体的一种服务，比如游泳，棋牌
Role	VIP等级，VIP级别越高，享有越高的权限

3.5.2 配置 Keystone

端口 5000 和 35357

1、修改/etc/keystone/keystone.conf

取一个随机数

```
[root@linux-node1 ~]# openssl rand -hex 10
35d6e6f377a889571bcf
[root@linux-node1 ~]# cat /etc/keystone/keystone.conf|grep -v "^#"|grep -v "^$"
[DEFAULT]
admin_token = 35d6e6f377a889571bcf                    #设置 token，和上面产生的随机数值一致
verbose = true
[assignment]
[auth]
[cache]
[catalog]
[cors]
[cors.subdomain]
[credential]
[database]
connection = mysql://keystone:keystone@192.168.1.17/keystone    #设置数据库连接 写到 database 下
[domain_config]
[endpoint_filter]
[endpoint_policy]
[eventlet_server]
[eventlet_server_ssl]
[federation]
[fernet_tokens]
[identity]
[identity_mapping]
[kvs]
[ldap]
[matchmaker_redis]
[matchmaker_ring]
[memcache]
servers = 192.168.1.17:11211
[oauth1]
[os_inherit]
[oslo_messaging_amqp]
[oslo_messaging_qpid]
[oslo_messaging_rabbit]
[oslo_middleware]
```

[oslo_policy]
[paste_deploy]
[policy]
[resource]
[revoke]
driver = sql
[role]
[saml]
[signing]
[ssl]
[token]
provider = uuid
driver = memcache
[tokenless_auth]
[trust]

2、 创建数据库表， 使用命令同步
[root@linux-node1 ~]# su -s /bin/sh -c "keystone-manage db_sync" keystone
No handlers could be found for logger
"oslo_config.cfg" #出现这个信息，不影响后续操作！忽略
~

[root@linux-node1 ~]# ll /var/log/keystone/keystone.log
-rw-r--r--. 1 keystone keystone 298370 Aug 26 11:36
/var/log/keystone/keystone.log #之所以上面 su 切换是因为这个日志文件属主
[root@linux-node1 config]# mysql -h 192.168.1.17 -u keystone -p #数据库检查表，
生产环境密码不要用 keystone，改成复杂点的密码

3、 启动 memcached 和 apache
启动 memcached
[root@linux-node1 ~]# systemctl enable memcached
[root@linux-node1 ~]#ln -s '/usr/lib/systemd/system/memcached.service'
'/etc/systemd/system/multi-user.target.wants/memcached.service'
[root@linux-node1 ~]# systemctl start memcached
配置 httpd
[root@linux-node1 ~]# vim /etc/httpd/conf/httpd.conf
ServerName 192.168.1.17:80
[root@linux-node1 ~]# cat /etc/httpd/conf.d/wsgi-keystone.conf
Listen 5000
Listen 35357

<VirtualHost *:5000>
WSGIDaemonProcess keystone-public processes=5 threads=1 user=keystone

```
group=keystone display-name=%{GROUP}
WSGIProcessGroup keystone-public
WSGIScriptAlias / /usr/bin/keystone-wsgi-public
WSGIApplicationGroup %{GLOBAL}
WSGIPassAuthorization On
<IfVersion >= 2.4>
ErrorLogFormat "%{cu}t %M"
</IfVersion>
ErrorLog /var/log/httpd/keystone-error.log
CustomLog /var/log/httpd/keystone-access.log combined
<Directory /usr/bin>
<IfVersion >= 2.4>
Require all granted
</IfVersion>
<IfVersion < 2.4>
Order allow,deny
Allow from all
</IfVersion>
</Directory>
</VirtualHost>

<VirtualHost *:35357>
WSGIDaemonProcess keystone-admin processes=5 threads=1 user=keystone
group=keystone display-name=%{GROUP}
WSGIProcessGroup keystone-admin
WSGIScriptAlias / /usr/bin/keystone-wsgi-admin
WSGIApplicationGroup %{GLOBAL}
WSGIPassAuthorization On
<IfVersion >= 2.4>
ErrorLogFormat "%{cu}t %M"
</IfVersion>
ErrorLog /var/log/httpd/keystone-error.log
CustomLog /var/log/httpd/keystone-access.log combined
<Directory /usr/bin>
<IfVersion >= 2.4>
Require all granted
</IfVersion>
<IfVersion < 2.4>
Order allow,deny
Allow from all
</IfVersion>
</Directory>
</VirtualHost>
```

启动 httpd

```
[root@linux-node1 config]# systemctl enable httpd
[root@linux-node1 config]#ln -s '/usr/lib/systemd/system/httpd.service'
'/etc/systemd/system/multi-user.target.wants/httpd.service'
[root@linux-node1 config]# systemctl start httpd
[root@linux-node1 ~]# netstat -lntup|grep httpd
tcp6 0 0 :::5000 :::* LISTEN 23632/httpd
tcp6 0 0 :::80 :::* LISTEN 23632/httpd
tcp6 0 0 :::35357 :::* LISTEN 23632/httpd
```

如果 http 起不来关闭 selinux 或者安装 yum install openstack-selinux

4、 创建 keystone 用户

临时设置 admin_token 用户的环境变量，用来创建用户

```
[root@linux-node1 ~]# export
OS_TOKEN=35d6e6f377a889571bcf                    #上面产生的随机数值
[root@linux-node1 ~]# export OS_URL=http://192.168.1.17:35357/v3
[root@linux-node1 ~]# export OS_IDENTITY_API_VERSION=3
```

创建 admin 项目---创建 admin 用户（密码 admin，生产不要这么玩） ---创建 admin 角色---把 admin 用户加入到 admin 项目赋予 admin 的角色（三个 admin 的位置：项目，用户，角色）

```
[root@linux-node1 ~]#openstack project create --domain default --description "Admin
Project" admin
[root@linux-node1 ~]#openstack user create --domain default --password-prompt
admin
[root@linux-node1 ~]#openstack role create admin
[root@linux-node1 ~]#openstack role add --project admin --user admin admin
```

创建一个普通用户 demo

```
[root@linux-node1 ~]#openstack project create --domain default --description "Demo
Project" demo
[root@linux-node1 ~]#openstack user create --domain default --password=demo
demo
[root@linux-node1 ~]#openstack role create user
[root@linux-node1 ~]#openstack role add --project demo --user demo user
```

创建 service 项目，用来管理其他服务用

```
[root@linux-node1 ~]#openstack project create --domain default --description "Service
Project" service
```

以上的名字都是固定的，不能改

查看创建的而用户和项目

```
[root@linux-node1 ~]# openstack user list
+----------------------------------+-------+
| ID | Name |
+----------------------------------+-------+
| b1f164577a2d43b9a6393527f38e3f75 | demo |
| b694d8f0b70b41d883665f9524c77766 | admin |
+----------------------------------+-------+
[root@linux-node1 ~]# openstack project list
+----------------------------------+---------+
| ID | Name |
+----------------------------------+---------+
| 604f9f78853847ac9ea3c31f2c7f677d | demo |
| 777f4f0108b1476eabc11e00dccaea9f | admin |
| aa087f62f1d44676834d43d0d902d473 | service |
+----------------------------------+---------+
```

5、注册 keystone 服务，以下**三种类型分别为公共的、内部的、管理的。**

```
[root@linux-node1 ~]#openstack service create --name keystone --description
"OpenStack Identity" identity
[root@linux-node1 ~]#openstack endpoint create --region RegionOne identity public
http://192.168.1.17:5000/v2.0
[root@linux-node1 ~]#openstack endpoint create --region RegionOne identity internal
http://192.168.1.17:5000/v2.0
[root@linux-node1 ~]#openstack endpoint create --region RegionOne identity admin
http://192.168.1.17:35357/v2.0
[root@linux-node1 ~]# openstack endpoint list #查看
+----------------------------------+-----------+--------------+--------------+---------+---
-
-------+----------------------------------+
| ID | Region | Service Name | Service Type | Enabled |
Interface | URL |
+----------------------------------+-----------+--------------+--------------+---------+---
-
-------+----------------------------------+
| 011a24def8664506985815e0ed2f8fa5 | RegionOne | keystone | identity | True |
internal | http://192.168.1.17:5000/v2.0 |
| b0981cae6a8c4b3186edef818733fec6 | RegionOne | keystone | identity | True | public
| http://192.168.1.17:5000/v2.0 |
| c4e0c79c0a8142eda4d9653064563991 | RegionOne | keystone | identity | True |
admin
| http://192.168.1.17:35357/v2.0 |
+----------------------------------+-----------+--------------+--------------+---------+---
-
-------+----------------------------------+
```

[root@linux-node1 ~]# openstack endpoint delete ID **#使用这个命令删除**

6、 验证，获取 token，只有获取到才能说明 keystone 配置成功

```
[root@linux-node1 ~]# unset OS_TOKEN
[root@linux-node1 ~]# unset OS_URL
[root@linux-node1 ~]# openstack --os-auth-url http://192.168.1.17:35357/v3
--os-project-domain-id default --os-user-domain-id default --os-project-name admin
--os-username admin --os-auth-type password token issue                    #回车
Password: admin
+------------+----------------------------------+
| Field | Value |
+------------+----------------------------------+
| expires | 2015-12-17T04:22:00.600668Z |
| id | 1b530a078b874438aadb77af11ce297e |
| project_id | 777f4f0108b1476eabc11e00dccaea9f |
| user_id | b694d8f0b70b41d883665f9524c77766 |
+------------+----------------------------------+
```

使用环境变量来获取 token，环境变量在后面创建虚拟机时也需要用。

创建两个环境变量文件，使用时直接 source！！！（注意，下面两个 sh 文件所在的路径，在查看命令前都要 source 下，不然会报错！！）

```
[root@linux-node1 ~]# cat admin-openrc.sh
export OS_PROJECT_DOMAIN_ID=default
export OS_USER_DOMAIN_ID=default
export OS_PROJECT_NAME=admin
export OS_TENANT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=admin
export OS_AUTH_URL=http://192.168.1.17:35357/v3
export OS_IDENTITY_API_VERSION=3

[root@linux-node1 ~]# cat demo-openrc.sh
export OS_PROJECT_DOMAIN_ID=default
export OS_USER_DOMAIN_ID=default
export OS_PROJECT_NAME=demo
export OS_TENANT_NAME=demo
export OS_USERNAME=demo
export OS_PASSWORD=demo
export OS_AUTH_URL=http://192.168.1.17:5000/v3
export OS_IDENTITY_API_VERSION=3
[root@linux-node1 ~]# source admin-openrc.sh
[root@linux-node1 ~]# openstack token issue
```

```
+------------+----------------------------------+
| Field | Value |
+------------+----------------------------------+
| expires | 2015-12-17T04:26:08.625399Z |
| id | 58370ae3b9bb4c07a67700dd184ad3b1 |
16
| project_id | 777f4f0108b1476eabc11e00dccaea9f |
| user_id | b694d8f0b70b41d883665f9524c77766 |
+------------+----------------------------------+
```

3.6 配置 glance 镜像服务

3.6.1 glance 介绍

3.6.2 glance 配置

端口：

api 9191

registry 9292

1、修改/etc/glance/glance-api.conf 和/etc/glance/glance-registry.conf

```
[root@linux-node1 ~]# cat /etc/glance/glance-api.conf|grep -v "^#"|grep -v "^$"
[DEFAULT]
verbose=True
notification_driver = noop                                    #galnce 不需要消息队列
```

```
[database]
connection=mysql://glance:glance@192.168.1.17/glance
[glance_store]
default_store=file
filesystem_store_datadir=/var/lib/glance/images/
[image_format]
[keystone_authtoken]
auth_uri = http://192.168.1.17:5000
auth_url = http://192.168.1.17:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = glance
password = glance
[matchmaker_redis]
[matchmaker_ring]
[oslo_concurrency]
[oslo_messaging_amqp]
[oslo_messaging_qpid]
[oslo_messaging_rabbit]
[oslo_policy]
[paste_deploy]
flavor=keystone
[store_type_location_strategy]
[task]
[taskflow_executor]


[root@linux-node1 ~]# cat /etc/glance/glance-registry.conf|grep -v "^#"|grep -v "^$"
[DEFAULT]
verbose=True
notification_driver = noop
[database]
connection=mysql://glance:glance@192.168.1.17/glance
[glance_store]
[keystone_authtoken]
auth_uri = http://192.168.1.17:5000
auth_url = http://192.168.1.17:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = glance
password = glance
```

[matchmaker_redis]
[matchmaker_ring]
[oslo_messaging_amqp]
[oslo_messaging_qpid]
[oslo_messaging_rabbit]
[oslo_policy]
[paste_deploy]
flavor=keystone

2、创建数据库表，同步数据库
[root@linux-node1 ~]# su -s /bin/sh -c "glance-manage db_sync" glance
[root@linux-node1 ~]# mysql -h 192.168.1.17 -uglance -p

3、 创建关于 glance 的 keystone 用户
[root@linux-node1 ~]# source admin-openrc.sh
[root@linux-node1 ~]# openstack user create --domain default --password=glance glance
[root@linux-node1 ~]# openstack role add --project service --user glance admin

4、启动 glance
[root@linux-node1 ~]#systemctl enable openstack-glance-api
[root@linux-node1 ~]#systemctl enable openstack-glance-registry
[root@linux-node1 ~]#systemctl start openstack-glance-api
[root@linux-node1 ~]#systemctl start openstack-glance-registry
[root@linux-node1 ~]# netstat -lnutp |grep 9191 #registry
tcp 0 0 0.0.0.0:9191 0.0.0.0:* LISTEN
24890/python2
[root@linux-node1 ~]# netstat -lnutp |grep 9292 #api
tcp 0 0 0.0.0.0:9292 0.0.0.0:* LISTEN
24877/python2

5、在 keystone 上注册
[root@linux-node1 ~]# source admin-openrc.sh
[root@linux-node1 ~]#openstack service create --name glance --description "OpenStack Image service" image
[root@linux-node1 ~]#openstack endpoint create --region RegionOne image public http://192.168.1.17:9292
[root@linux-node1 ~]#openstack endpoint create --region RegionOne image internal http://192.168.1.17:9292
[root@linux-node1 ~]#openstack endpoint create --region RegionOne image admin http://192.168.1.17:9292

6、添加 glance 环境变量并测试

```
[root@linux-node1 src]# echo "export OS_IMAGE_API_VERSION=2" | tee -a admin-openrc.sh demo-openrc.sh
[root@linux-node1 src]# glance image-list
+----+------+
| ID | Name |
+----+------+
+----+------+
```

7、 下载镜像并上传到 glance【此处下载的 qcow2 格式镜像比较小，可以直接下载 ios 格式镜像，然后用 oz 工具制造】

```
[root@linux-node1 ~]# wget -q http://download.cirros-cloud.net/0.3.4/cirros-0.3.4-x86_64-disk.img
                    #也可以提前下载下来
[root@linux-node1 ~]# glance image-create --name "cirros" --file cirros-0.3.4-x86_64-disk.img --disk-format qcow2 --container-format bare --visibility public --progress
[==============================>] 100%
+------------------+--------------------------------------+
| Property | Value |
+------------------+--------------------------------------+
| checksum | ee1eca47dc88f4879d8a229cc70a07c6 |
| container_format | bare |
| created_at | 2015-12-17T04:11:02Z |
| disk_format | qcow2 |
| id | 2707a30b-853f-4d04-861d-e05b0f1855c8 |
| min_disk | 0 |
| min_ram | 0 |
| name | cirros |
| owner | 777f4f0108b1476eabc11e00dccaea9f |
| protected | False |
| size | 13287936 |
| status | active |
| tags | [] |
| updated_at | 2015-12-17T04:11:03Z |
| virtual_size | None |
| visibility | public |
+------------------+--------------------------------------+
```

下载 ios 格式镜像，需要用 OZ 工具制造 openstack 镜像，具体操作请见另一篇博客：

实际生产环境下，肯定要使用 ios 镜像进行制作了

http://www.cnblogs.com/kevingrace/p/5821823.html

或者直接下载 centos 的 qcow2 格式镜像进行上传，qcow2 格式镜像直接就可以在 openstack 里使用，不需要进行格式转换！
下载地址：http://cloud.centos.org/centos，可以到里面下载 centos5/6/7 的 qcow2 格式的镜像

[root@linux-node1 ~]#wget http://cloud.centos.org/centos/7/images/CentOS-7-x86_64-GenericCloud.qcow2
[root@linux-node1 ~]#glance image-create --name "CentOS-7-x86_64" --file CentOS-7-x86_64-GenericCloud.qcow2 --disk-format qcow2 --container-format bare --visibility public --progress

---

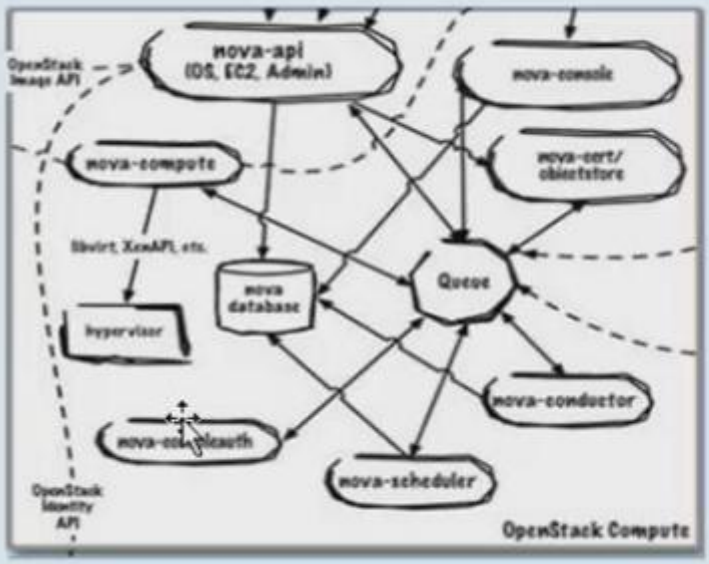
```
[root@linux-node1 ~]# glance image-list
+--------------------------------------+--------+
| ID | Name |
+--------------------------------------+--------+
| 2707a30b-853f-4d04-861d-e05b0f1855c8 | cirros |
+--------------------------------------+--------+

[root@linux-node1 ~]# ll /var/lib/glance/images/
总用量 12980
-rw-r-----. 1 glance glance 1569390592 Aug 26 12:50 35b36f08-eeb9-4a91-9366-561f0a308a1b
```

3.7 配置 nova 计算服务

3.7.1 nova 介绍

nova 必备的组件

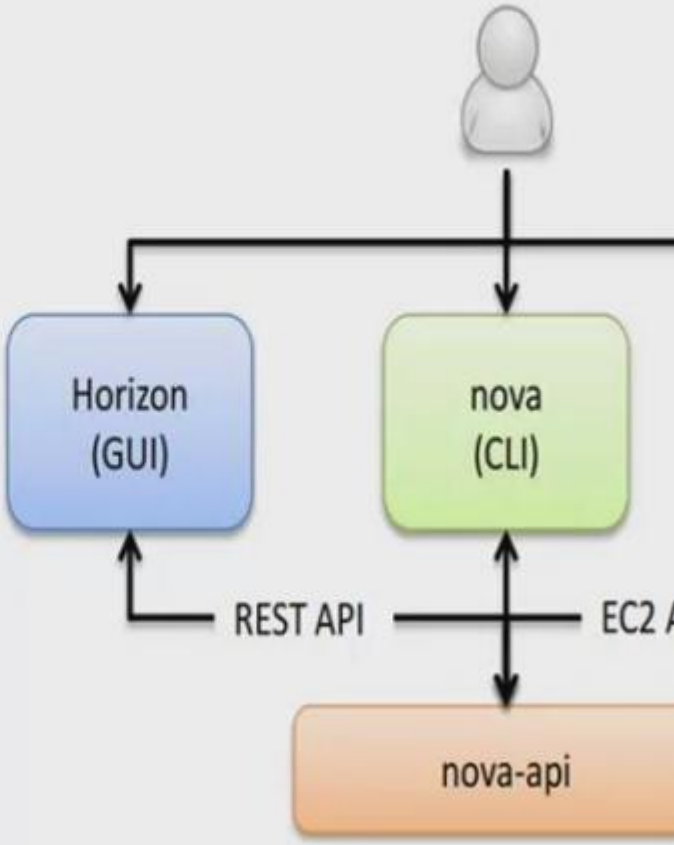


➢API：负责接收和响应外部
OpenStack API，EC2API。
➢Cert：负责身份认证。
➢Scheduler：用于云主机调
➢Conductor：计算节点访问
➢Consoleauth：用于控制台
➢Novncproxy：VNC代理。

nova API

Nova API

- nova-api组件实现了RESTful API功能，是外部访问Nova的唯一途径。
- 接收外部的请求并通过Message Queue将请求发送给其他的服务组件，同时也兼容EC2 API，所以也可以用EC2的管理工具对nova进行日常管理。



nova scheduler

Nova scheduler

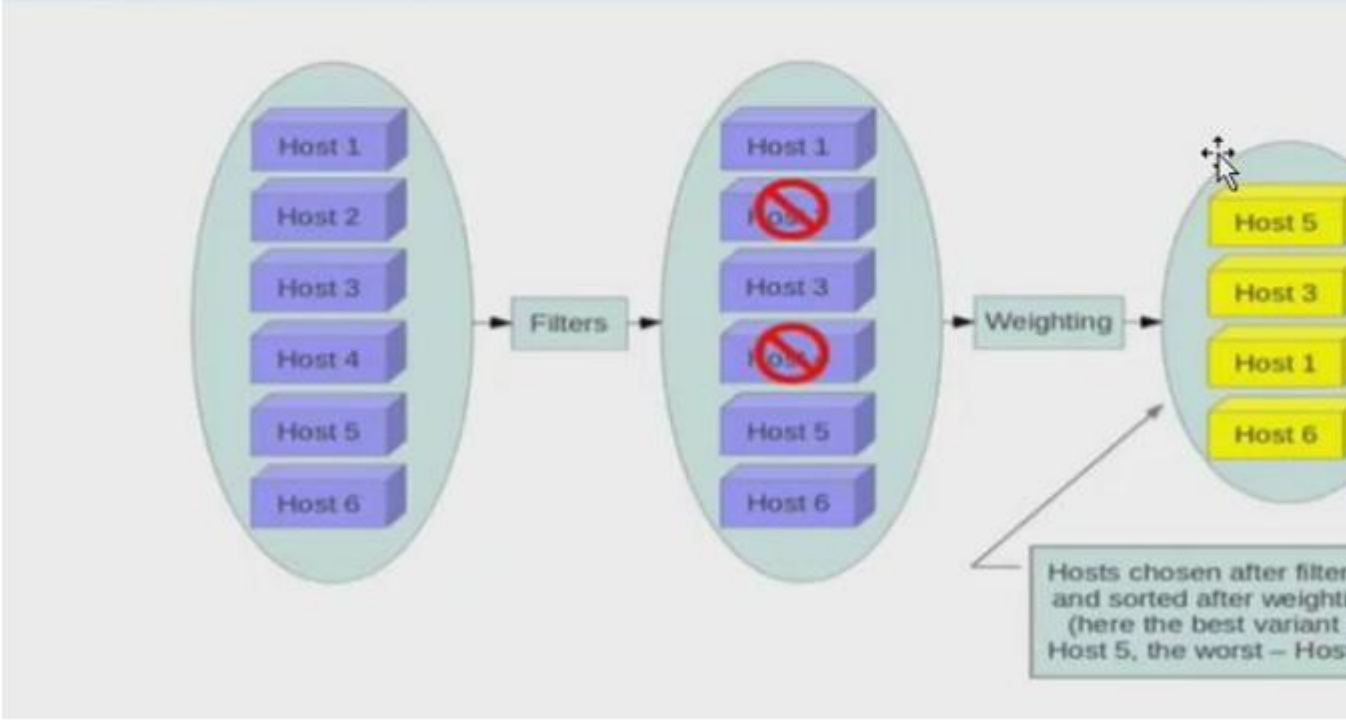
Nova Scheduler模块在openstack中的作用就是决策
建在哪个主机（计算节点）上。

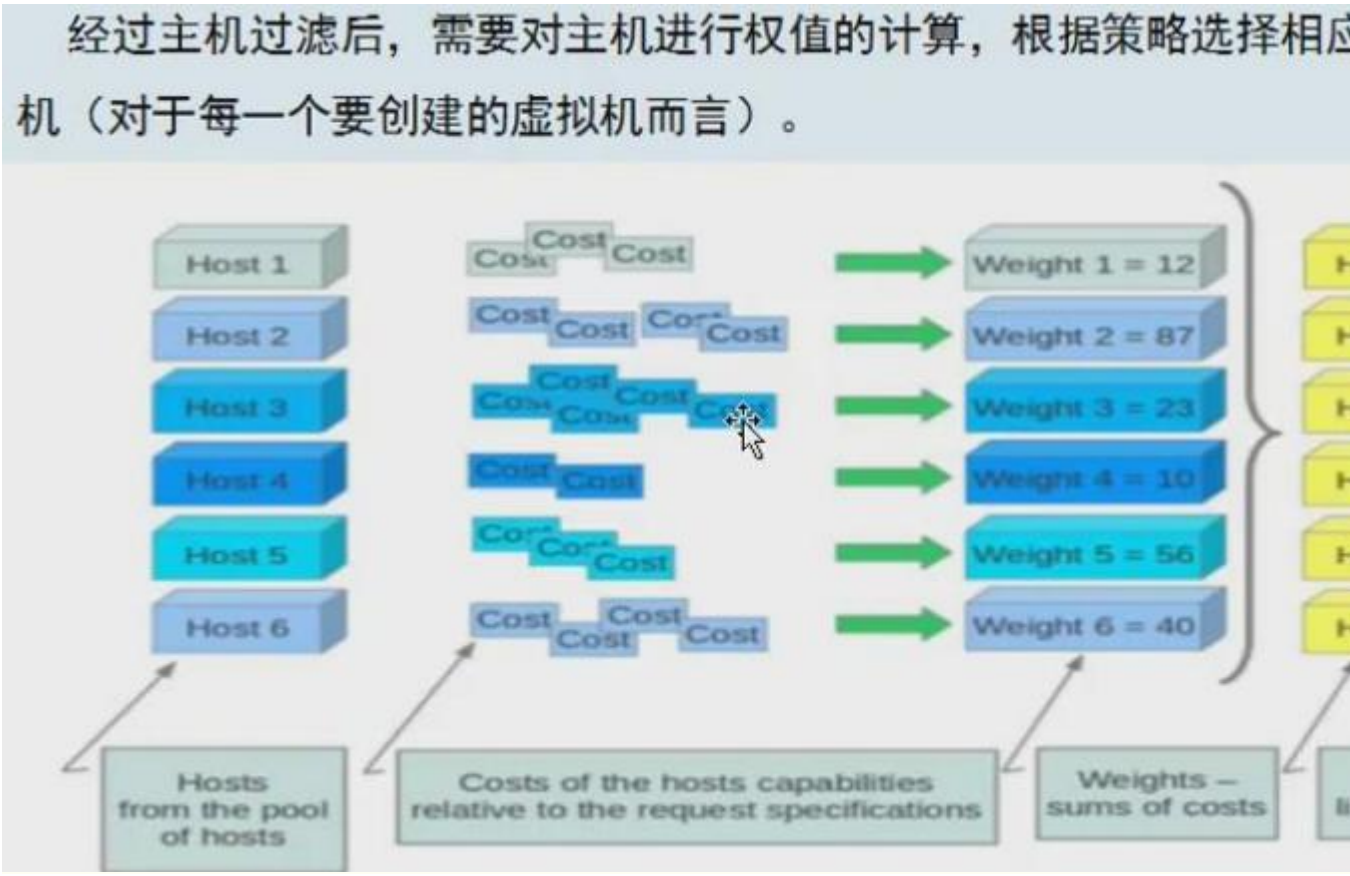
决策一个虚机应该调度到某物理节点，需要分两个步骤：

- 过滤（Fliter）
- 计算权值（Weight）

Nova Dashboard

Filter Scheduler首先得到未经过滤的主机列表，然后根据过滤属性的计算节点主机。





3.7.2 Nova 控制节点配置

1、修改/etc/nova/nova.conf

```
[root@linux-node1 ~]# cat /etc/nova/nova.conf|grep -v "^#"|grep -v "^$"
[DEFAULT]
my_ip=192.168.1.17
enabled_apis=osapi_compute,metadata
auth_strategy=keystone
network_api_class=nova.network.neutronv2.api.API
linuxnet_interface_driver=nova.network.linux_net.NeutronLinuxBridgeInterfaceDriver
security_group_api=neutron
firewall_driver = nova.virt.firewall.NoopFirewallDriver
debug=true
verbose=true
rpc_backend=rabbit
allow_resize_to_same_host=True
scheduler_default_filters=RetryFilter,AvailabilityZoneFilter,RamFilter,ComputeFilter,Co
mputeCapabilitiesFilter,ImagePropertiesFilter,ServerGroupAntiAffinityFilter,ServerGrou
pAffinityFilter
[api_database]
[barbican]
```

```
[cells]
[cinder]
[conductor]
[cors]
[cors.subdomain]
[database]
connection=mysql://nova:nova@192.168.1.17/nova
[ephemeral_storage_encryption]
[glance]
host=$my_ip
[guestfs]
[hyperv]
[image_file_url]
[ironic]
[keymgr]
[keystone_authtoken]
auth_uri = http://192.168.1.17:5000
auth_url = http://192.168.1.17:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = nova
password = nova
[libvirt]
virt_type=kvm                          #如果控制节点也作为计算节点（单机部署的话），
这一行也添加上（这行是计算节点配置的）
[matchmaker_redis]
[matchmaker_ring]
[metrics]
[neutron]
url = http://192.168.1.17:9696
auth_url = http://192.168.1.17:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
region_name = RegionOne
project_name = service
username = neutron
password = neutron
service_metadata_proxy = True
metadata_proxy_shared_secret = neutron
lock_path=/var/lib/nova/tmp
[osapi_v21]
```

[oslo_concurrency]
[oslo_messaging_amqp]
[oslo_messaging_qpid]
[oslo_messaging_rabbit]
rabbit_host=192.168.1.17
rabbit_port=5672
rabbit_userid=openstack
rabbit_password=openstack
[oslo_middleware]
[rdp]
[serial_console]
[spice]
[ssl]
[trusted_computing]
[upgrade_levels]
[vmware]
[vnc]
novncproxy_base_url=http://58.68.250.17:6080/vnc_auto.html #如果控制节点也作为计算节点（单机部署的话），这一行也添加上（这行是计算节点配置的），配置控制节点的公网ip
vncserver_listen= $my_ip
vncserver_proxyclient_address= $my_ip
keymap=en-us #如果控制节点也作为计算节点（单机部署的话），这一行也添加上（这行是计算节点配置的）
[workarounds]
[xenserver]
[zookeeper]

****
{网络部分为啥这么写：network_api_class=nova.network.neutronv2.api.API}
[root@linux-node1 ~]# ls
/usr/lib/python2.7/site-packages/nova/network/neutronv2/api.py
/usr/lib/python2.7/site-packages/nova/network/neutronv2/api.py
这里面有一个 API 方法，其他配置类似
****

2、同步数据库
[root@linux-node1 ~]# su -s /bin/sh -c "nova-manage db sync" nova
[root@linux-node1 ~]# mysql -h 192.168.1.17 -unova -p 检查

3、创建 nova 的 keystone 用户

```
[root@linux-node1 ~]# openstack user create --domain default --password=nova nova
[root@linux-node1 ~]# openstack role add --project service --user nova admin
```

4、启动 nova 相关服务

```
[root@linux-node1 ~]#systemctl enable openstack-nova-api.service
openstack-nova-cert.service openstack-nova-consoleauth.service
openstack-nova-scheduler.service openstack-nova-conductor.service
openstack-nova-novncproxy.service
[root@linux-node1 ~]#systemctl start openstack-nova-api.service
openstack-nova-cert.service openstack-nova-consoleauth.service
openstack-nova-scheduler.service openstack-nova-conductor.service
openstack-nova-novncproxy.service
```

5、在 keystone 上注册

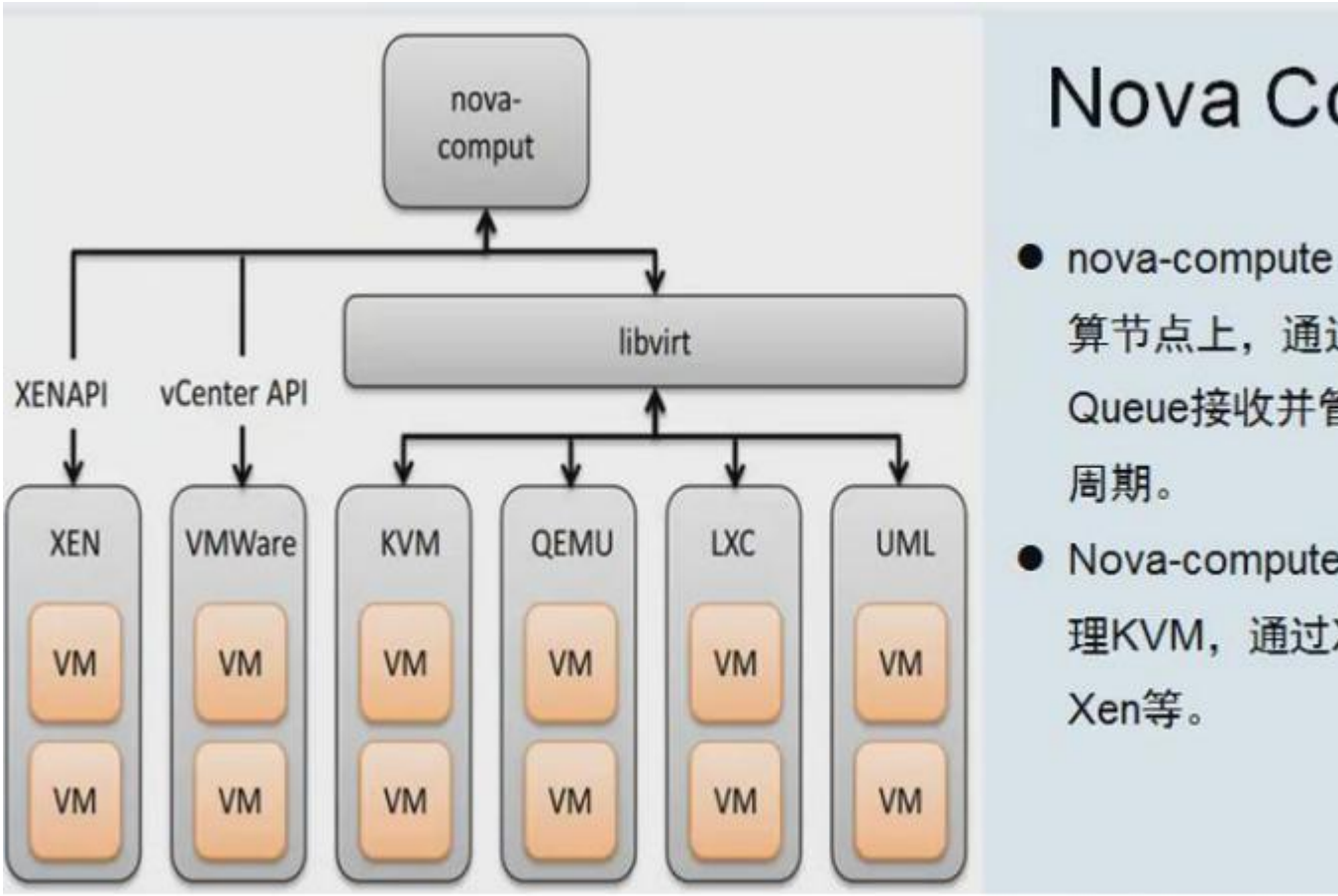
```
[root@linux-node1 ~]# source admin-openrc.sh
[root@linux-node1 ~]# openstack service create --name nova --description "OpenStack
Compute" compute
[root@linux-node1 ~]# openstack endpoint create --region RegionOne compute public
http://192.168.1.17:8774/v2/%\(tenant_id\)s
[root@linux-node1 ~]# openstack endpoint create --region RegionOne compute
internal http://192.168.1.17:8774/v2/%\(tenant_id\)s
[root@linux-node1 ~]# openstack endpoint create --region RegionOne compute admin
http://192.168.1.17:8774/v2/%\(tenant_id\)s
```

检查

```
[root@linux-node1 ~]# openstack host list
+--------------------------+-------------+----------+
| Host Name | Service | Zone |
+--------------------------+-------------+----------+
| linux-node1.oldboyedu.com | conductor | internal |
| linux-node1.oldboyedu.com | scheduler | internal |
| linux-node1.oldboyedu.com | consoleauth | internal |
| linux-node1.oldboyedu.com | cert | internal |
+--------------------------+-------------+----------+
```

3.7.3 nova 计算节点配置

1、 nova compute 介绍



2、修改配置文件/etc/nova/nova.conf

可以直接从 **node1** 拷贝到 **node2** 上

[root@linux-node1 ~]# scp /etc/nova/nova.conf 192.168.1.8:/etc/nova/

手动更改如下配置

[root@**linux-node2** ~]# vim /etc/nova/nova.conf

```
my_ip=192.168.1.8
novncproxy_base_url=http://192.168.1.17:6080/vnc_auto.html
vncserver_listen=0.0.0.0
vncserver_proxyclient_address= $my_ip
keymap=en-us
[glance]
host=192.168.56.17
[libvirt]
virt_type=kvm                    #虚拟机类型，默认是 kvm
```

3、启动服务

[root@**linux-node2** ~]# systemctl enable libvirtd openstack-nova-compute

[root@**linux-node2** ~]# systemctl start libvirtd openstack-nova-compute

4、在控制节点测试（计算节点上也行，需要环境变量）

```
[root@linux-node1 ~]# openstack host list
+--------------------------+-------------+----------+
| Host Name | Service | Zone |
+--------------------------+-------------+----------+
| linux-node1.oldboyedu.com | conductor | internal |
| linux-node1.oldboyedu.com | consoleauth | internal |
| linux-node1.oldboyedu.com | scheduler | internal |
| linux-node1.oldboyedu.com | cert | internal |
| linux-node2.oldboyedu.com | compute | nova |
+--------------------------+-------------+----------+


[root@linux-node1 ~]# nova image-list                    #测试 glance 是否正常
+--------------------------------------+--------+--------+--------+
| ID | Name | Status | Server |
+--------------------------------------+--------+--------+--------+
| 2707a30b-853f-4d04-861d-e05b0f1855c8 | cirros | ACTIVE | |
+--------------------------------------+--------+--------+--------+
[root@linux-node1 ~]# nova endpoints                     #测试 keystone
WARNING: keystone has no endpoint in ! Available endpoints for this service:          #
这一行告警不影响后面的操作
+-----------+----------------------------------+
| keystone | Value |
+-----------+----------------------------------+
| id | 02fed35802734518922d0ca2d672f469 |
| interface | internal |
| region | RegionOne |
| region_id | RegionOne |
| url | http://192.168.1.17:5000/v2.0 |
+-----------+----------------------------------+
+-----------+----------------------------------+
| keystone | Value |
+-----------+----------------------------------+
| id | 52b0a1a700f04773a220ff0e365dea45 |
| interface | public |
| region | RegionOne |
| region_id | RegionOne |
| url | http://192.168.1.17:5000/v2.0 |
+-----------+----------------------------------+
+-----------+----------------------------------+
| keystone | Value |
+-----------+----------------------------------+
| id | 88df7df6427d45619df192979219e65c |
| interface | admin |
```

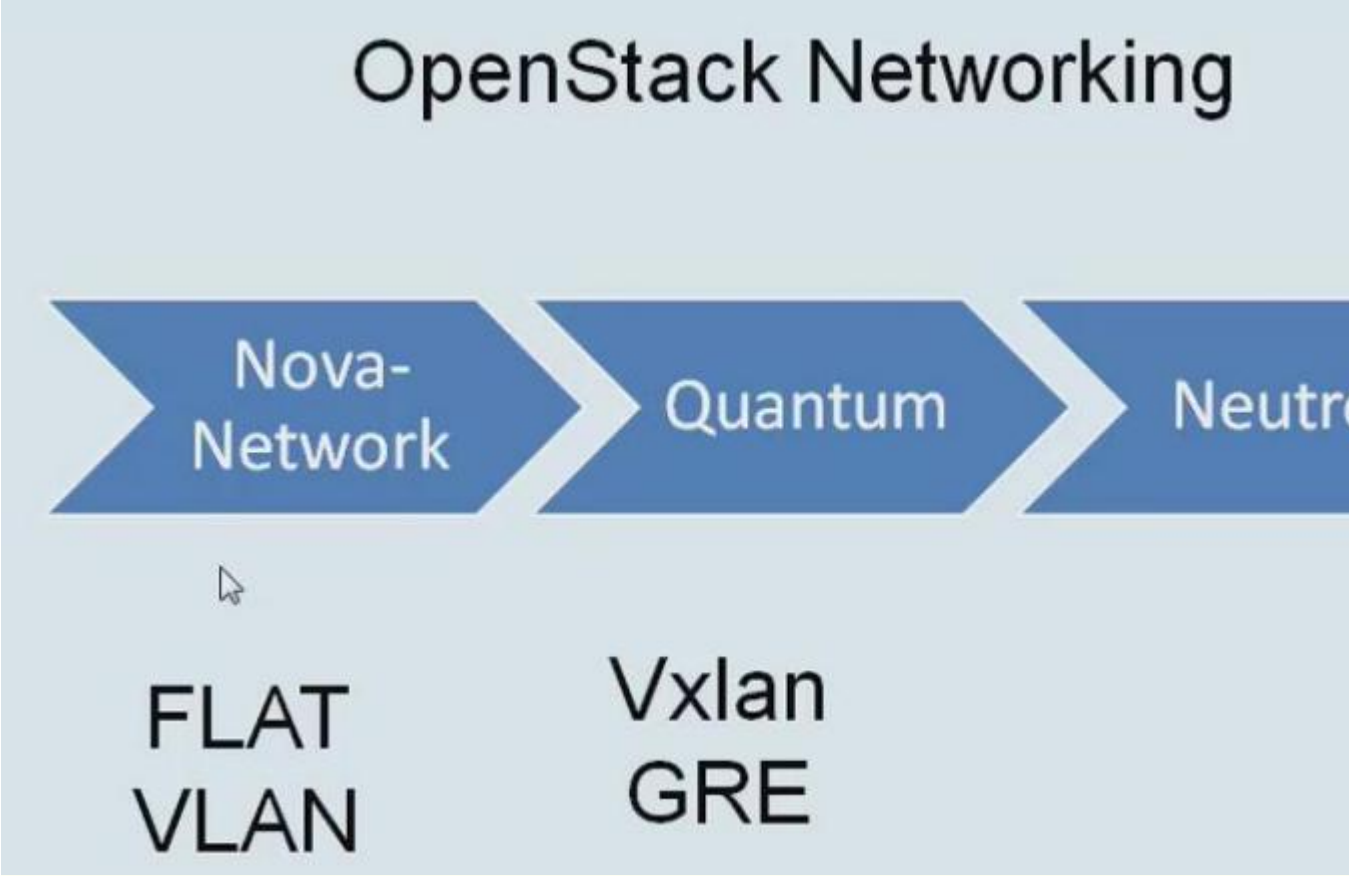
region	RegionOne
region_id	RegionOne
url	http://192.168.1.17:35357/v2.0
+-----------+----------------------------------+
WARNING: nova has no endpoint in ! Available endpoints for this service:
+-----------+--+
| nova | Value |
+-----------+--+
id	1a3115941ff54b7499a800c7c43ee92a
interface	internal
region	RegionOne
region_id	RegionOne
url	http://192.168.1.17:8774/v2/65a0c00638c247a0a274837aa6eb165f
+-----------+--+
+-----------+--+
| nova | Value |
+-----------+--+
id	5278f33a42754c9a8d90937932b8c0b3
interface	admin
region	RegionOne
region_id	RegionOne
url	http://192.168.1.17:8774/v2/65a0c00638c247a0a274837aa6eb165f
+-----------+--+
+-----------+--+
| nova | Value |
+-----------+--+
id	8c4fa7b9a24949c5882949d13d161d36
interface	public
region	RegionOne
region_id	RegionOne
url	http://192.168.1.17:8774/v2/65a0c00638c247a0a274837aa6eb165f
+-----------+--+
WARNING: glance has no endpoint in ! Available endpoints for this service:
+-----------+----------------------------------+
| glance | Value |
+-----------+----------------------------------+
id	31fbf72537a14ba7927fe9c7b7d06a65
interface	admin
region	RegionOne
region_id	RegionOne
url	http://192.168.1.17:9292
+-----------+----------------------------------+
+-----------+----------------------------------+
| glance | Value |

```
+-----------+----------------------------------+
| id | be788b4aa2ce4251b424a3182d0eea11 |
| interface | public |
| region | RegionOne |
| region_id | RegionOne |
| url | http://192.168.1.17:9292 |
+-----------+----------------------------------+
+-----------+----------------------------------+
| glance | Value |
+-----------+----------------------------------+
| id | d0052712051a4f04bb59c06e2d5b2a0b |
| interface | internal |
| region | RegionOne |
| region_id | RegionOne |
| url | http://192.168.1.17:9292 |
+-----------+----------------------------------+
```

3.8 Neutron 网络服务

3.8.1 Neutron 介绍

neutron 由来

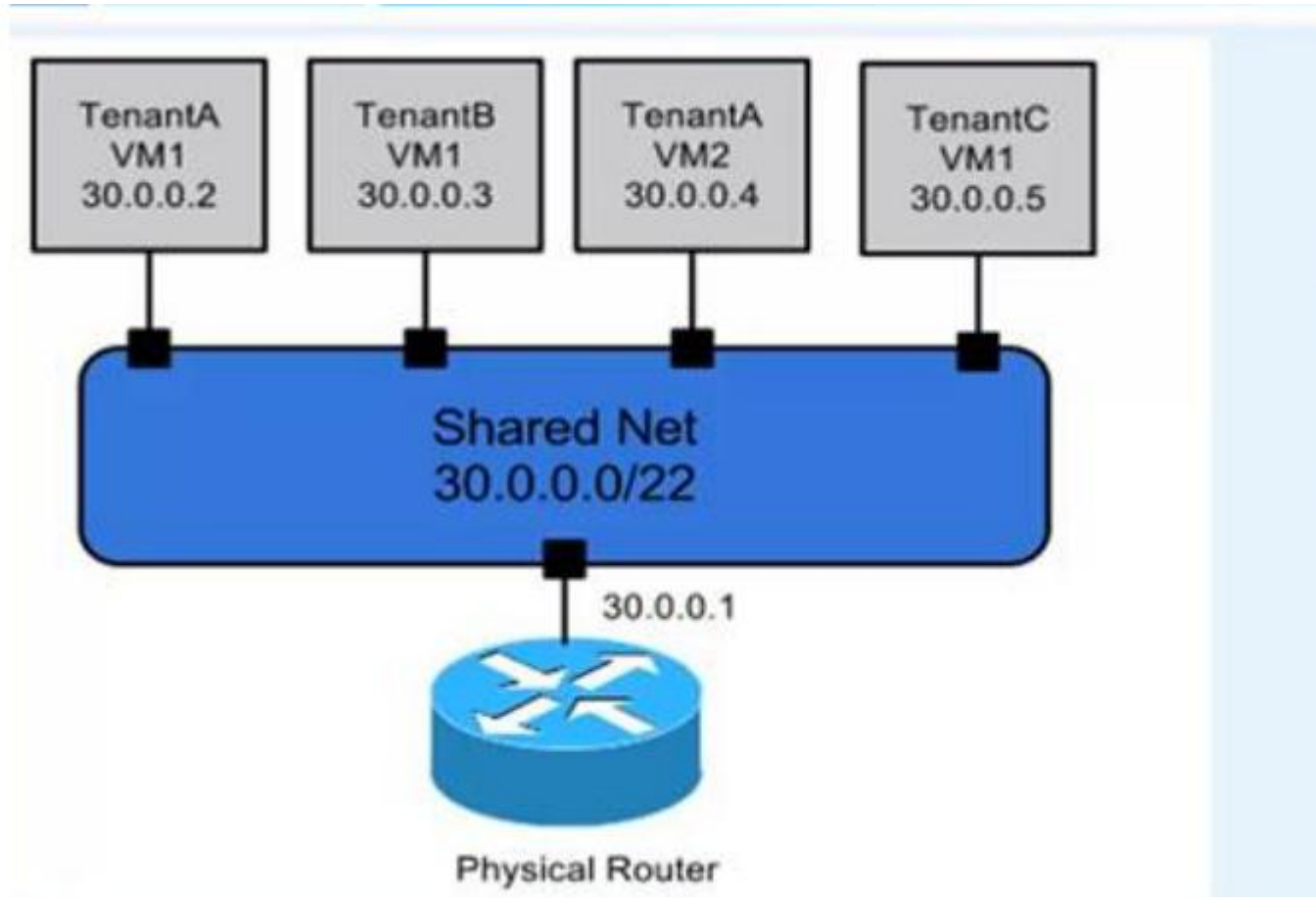


openstack 网络分类：

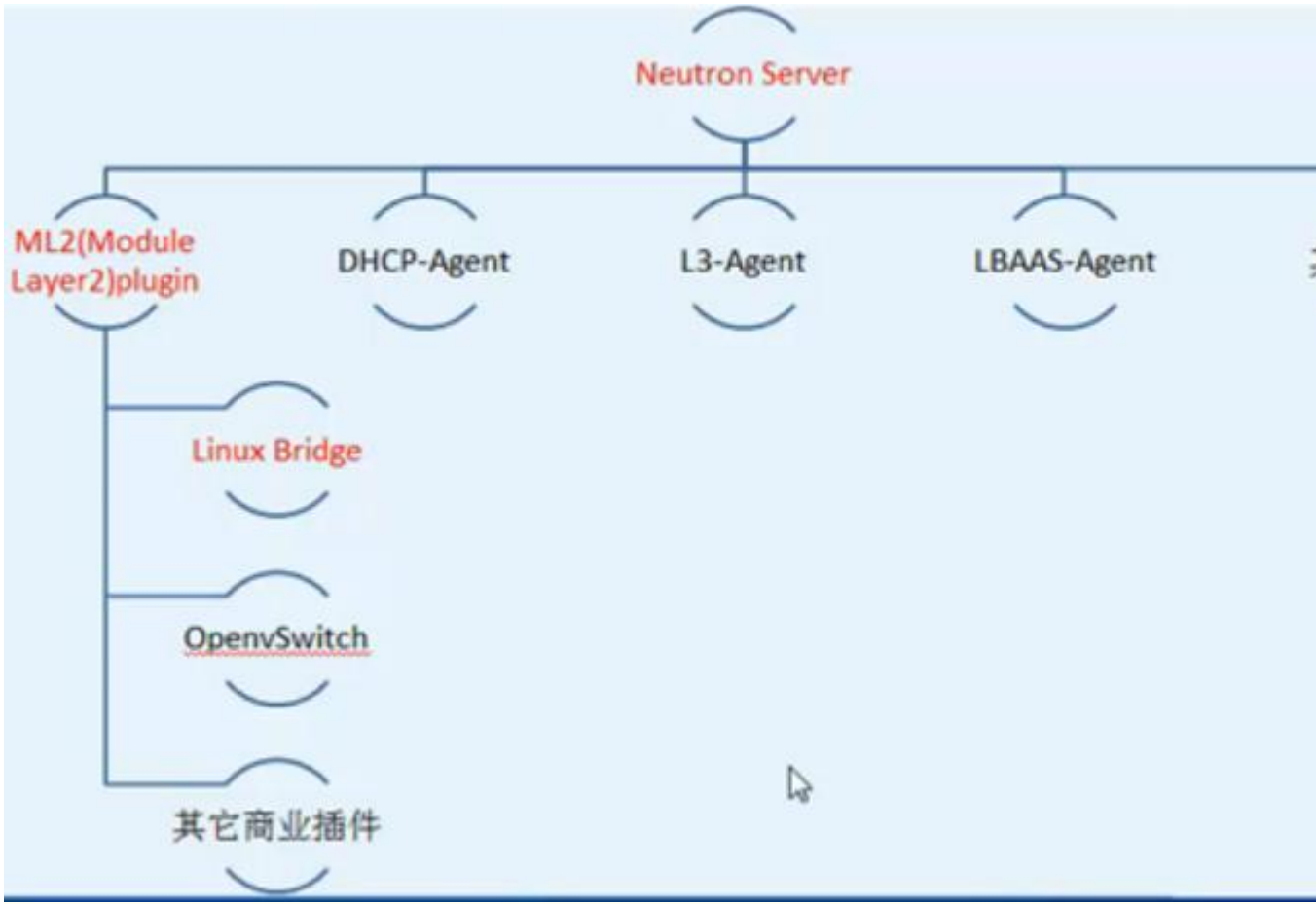
公共网络　向租户提供访问或者 API 调用
管理网络　云中物理机之间的通信
存储网络　云中存储的网络，如 ISCSI 或 GlusterFS 使用
服务网络　虚拟机内部使用的网络

OpenStack Networking

- **网络**：在实际的物理环境下，我们使用交换机或者集线器把多个计算机连
  网络。在Neutron的世界里，网络也是将多个不同的云主机连接起来。
- **子网**：在实际的物理环境下，在一个网络中。我们可以将网络划分成多为
  Neutron的世界里，子网也是隶属于网络下的。
- **端口**：是实际的物理环境下，每个子网或者每个网络，都有很多的端口，
  来供计算机连接。在Neutron的世界里端口也是隶属于子网下，云主机的
  个端口上。
- **路由器**：在实际的网络环境下，不同网络或者不同逻辑子网之间如果需要
  通过路由器进行路由。在Neutron的实际里路由也是这个作用。用来连接
  子网。


TenantA
VM1
30.0.0.2
TenantB
VM1
30.0.0.3
TenantA
VM2
30.0.0.4
TenantC
VM1
30.0.0.5
Shared Net
30.0.0.0/22
30.0.0.1
Physical Router

Neutron 组件



ML2 可以实现下面多个网络插件的共存。
DHCP-Agent 分配 IP 地址。
L3-Agent 用来路由
LBASS-Agent 负载均衡

3.8.2 Neutron 控制节点配置（ 5 个配置文件）

1、修改**/etc/neutron/neutron.conf** 文件

```
[root@linux-node1 ~]# cat /etc/neutron/neutron.conf|grep -v "^#"|grep -v "^$"
[DEFAULT]
state_path = /var/lib/neutron
core_plugin = ml2
service_plugins = router
auth_strategy = keystone
notify_nova_on_port_status_changes = True
notify_nova_on_port_data_changes = True
```

```
nova_url = http://192.168.1.17:8774/v2
rpc_backend=rabbit
[matchmaker_redis]
[matchmaker_ring]
[quotas]
[agent]
[keystone_authtoken]
auth_uri = http://192.168.1.17:5000
auth_url = http://192.168.1.17:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = neutron
password = neutron
admin_tenant_name = %SERVICE_TENANT_NAME%
admin_user = %SERVICE_USER%
admin_password = %SERVICE_PASSWORD%
[database]
connection = mysql://neutron:neutron@192.168.1.17:3306/neutron
[nova]
auth_url = http://192.168.1.17:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
region_name = RegionOne
project_name = service
username = nova
password = nova
[oslo_concurrency]
lock_path = $state_path/lock
[oslo_policy]
[oslo_messaging_amqp]
[oslo_messaging_qpid]
[oslo_messaging_rabbit]
rabbit_host = 192.168.1.17
rabbit_port = 5672
rabbit_userid = openstack
rabbit_password = openstack
[qos]
```

2、 配置/etc/neutron/plugins/ml2/ml2_conf.ini

```
[root@linux-node1 ~]# cat /etc/neutron/plugins/ml2/ml2_conf.ini|grep -v "^#"|grep
-v "^$"
```

```
[ml2]
type_drivers = flat,vlan,gre,vxlan,geneve
tenant_network_types = vlan,gre,vxlan,geneve
mechanism_drivers = openvswitch,linuxbridge
extension_drivers = port_security
[ml2_type_flat]
flat_networks = physnet1
[ml2_type_vlan]
[ml2_type_gre]
[ml2_type_vxlan]
[ml2_type_geneve]
[securitygroup]
enable_ipset = True
```

3、配置/etc/neutron/plugins/ml2/ linuxbridge_agent.ini

```
[root@linux-node1 ~]# cat /etc/neutron/plugins/ml2/linuxbridge_agent.ini|grep -v "^#"|grep -v "^$"
[linux_bridge]
physical_interface_mappings = physnet1:em2
[vxlan]
enable_vxlan = false
[agent]
prevent_arp_spoofing = True
[securitygroup]
firewall_driver = neutron.agent.linux.iptables_firewall.IptablesFirewallDriver
enable_security_group = True
```

4、修改/etc/neutron/dhcp_agent.ini

```
[root@linux-node1 ~]# cat /etc/neutron/dhcp_agent.ini|grep -v "^#"|grep -v "^$"
[DEFAULT]
interface_driver = neutron.agent.linux.interface.BridgeInterfaceDriver
dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
enable_isolated_metadata = true
[AGENT]
```

5、修改/etc/neutron/metadata_agent.ini

```
[root@linux-node1 ~]# cat /etc/neutron/metadata_agent.ini|grep -v "^#"|grep -v "^$"
[DEFAULT]
auth_uri = http://192.168.1.17:5000
auth_url = http://192.168.1.17:35357
auth_region = RegionOne
auth_plugin = password
```

project_domain_id = default
user_domain_id = default
project_name = service
username = neutron
password = neutron
nova_metadata_ip = 192.168.1.17
metadata_proxy_shared_secret = neutron
admin_tenant_name = %SERVICE_TENANT_NAME%
admin_user = %SERVICE_USER%
admin_password = %SERVICE_PASSWORD%
[AGENT]

6、创建连接并创建 keystone 的用户

```
[root@linux-node1 ~]# ln -s /etc/neutron/plugins/ml2/ml2_conf.ini
/etc/neutron/plugin.ini
[root@linux-node1 ~]# openstack user create --domain default --password=neutron
neutron
[root@linux-node1 ~]# openstack role add --project service --user neutron admin
```

7、更新数据库

```
[root@linux-node1 ~]# 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
```

8、注册 keystone

```
[root@linux-node1 ~]# source admin-openrc.sh
[root@linux-node1 ~]# openstack service create --name neutron --description
"OpenStack Networking" network
[root@linux-node1 ~]# openstack endpoint create --region RegionOne network public
http://192.168.1.17:9696
[root@linux-node1 ~]# openstack endpoint create --region RegionOne network internal
http://192.168.1.17:9696
[root@linux-node1 ~]# openstack endpoint create --region RegionOne network admin
http://192.168.1.17:9696
```

9、 启动服务并检查

因为 neutron 和 nova 有联系，做 neutron 时修改 nova 的配置文件，上面 nova.conf 已经做了 neutron 的关联配置，所以要重启 openstack-nova-api 服务。

这里将 nova 的关联服务都一并重启了：

```
[root@linux-node1 ~]# systemctl restart openstack-nova-api.service
openstack-nova-cert.service openstack-nova-consoleauth.service
openstack-nova-scheduler.service openstack-nova-conductor.service
openstack-nova-novncproxy.service
```

启动 neutron 相关服务

```
[root@linux-node1 ~]# systemctl enable neutron-server.service
neutron-linuxbridge-agent.service neutron-dhcp-agent.service
neutron-metadata-agent.service
[root@linux-node1 ~]# systemctl start neutron-server.service
neutron-linuxbridge-agent.service neutron-dhcp-agent.service
neutron-metadata-agent.service
```

检查

```
[root@linux-node1 ~]# neutron agent-list
+--------------------------------------+--------------------+-----------------+-------+----
----------+---------------------------+
| id | agent_type | host | alive | admin_state_up | binary |
+--------------------------------------+--------------------+-----------------+-------+----
----------+---------------------------+
| 385cebf9-9b34-4eca-b780-c515dbc7eec0 | Linux bridge agent | openstack-server | :-)
| True | neutron-linuxbridge-agent |
| b3ff8ffe-1ff2-4659-b823-331def4e6a93 | DHCP agent | openstack-server | :-) | True
| neutron-dhcp-agent |
| b5bed625-47fd-4e79-aa55-01cf8a8cc577 | Metadata agent | openstack-server | :-) |
True | neutron-metadata-agent |
+--------------------------------------+--------------------+-----------------+-------+----
----------+---------------------------+
```

查看注册信息

```
[root@openstack-server src]# openstack endpoint list
+----------------------------------+-----------+--------------+--------------+---------+---
--------+-----------------------------------------+
| ID | Region | Service Name | Service Type | Enabled | Interface | URL |
+----------------------------------+-----------+--------------+--------------+---------+---
--------+-----------------------------------------+
| 02fed35802734518922d0ca2d672f469 | RegionOne | keystone | identity | True |
internal | http://192.168.1.17:5000/v2.0 |
| 1a3115941ff54b7499a800c7c43ee92a | RegionOne | nova | compute | True | internal
| http://192.168.1.17:8774/v2/%(tenant_id)s |
| 31fbf72537a14ba7927fe9c7b7d06a65 | RegionOne | glance | image | True | admin |
http://192.168.1.17:9292 |
| 5278f33a42754c9a8d90937932b8c0b3 | RegionOne | nova | compute | True | admin
| http://192.168.1.17:8774/v2/%(tenant_id)s |
| 52b0a1a700f04773a220ff0e365dea45 | RegionOne | keystone | identity | True | public
| http://192.168.1.17:5000/v2.0 |
| 88df7df6427d45619df192979219e65c | RegionOne | keystone | identity | True |
admin | http://192.168.1.17:35357/v2.0 |
```

8c4fa7b9a24949c5882949d13d161d36	RegionOne	nova	compute	True	public	http://192.168.1.17:8774/v2/%(tenant_id)s
be788b4aa2ce4251b424a3182d0eea11	RegionOne	glance	image	True	public	http://192.168.1.17:9292
c059a07fa3e141a0a0b7fc2f46ca922c	RegionOne	neutron	network	True	public	http://192.168.1.17:9696
d0052712051a4f04bb59c06e2d5b2a0b	RegionOne	glance	image	True	internal	http://192.168.1.17:9292
ea325a8a2e6e4165997b2e24a8948469	RegionOne	neutron	network	True	internal	http://192.168.1.17:9696
ffdec11ccf024240931e8ca548876ef0	RegionOne	neutron	network	True	admin	http://192.168.1.17:9696
+----------------------------------+-----------+--------------+--------------+---------+----------+---+

3.8.3 Neutron 计算节点配置

1、修改相关配置文件

从 node1 上直接拷贝

[root@**linux-node1** ~]# scp /etc/neutron/neutron.conf 192.168.1.8:/etc/neutron/
[root@linux-node1 ~]# scp /etc/neutron/plugins/ml2/linuxbridge_agent.ini 192.168.1.8:/etc/neutron/plugins/ml2/
[root@linux-node1 ~]# scp /etc/neutron/plugins/ml2/ml2_conf.ini 192.168.1.8:/etc/neutron/plugins/ml2/
修改计算节点的 nova 配置文件中 neutron 部分， 并重启 openstack-nova-compute 服务，因为
上面 nova 计算节点也是从控制节点拷贝的，此处无需操作

2、 创建软连接并启动服务

[root@**linux-node2** ~]# ln -s /etc/neutron/plugins/ml2/ml2_conf.ini /etc/neutron/plugin.ini
[root@**linux-node2** ~]# systemctl enable neutron-linuxbridge-agent.service
[root@**linux-node2** ~]# systemctl start neutron-linuxbridge-agent.service

检查

[root@**linux-node1** ~]# neutron agent-list
+--------------------------------------+--------------------+-----------------+-------+----------------+---------------------------+
| id | agent_type | host | alive | admin_state_up | binary |
+--------------------------------------+--------------------+-----------------+-------+----------------+---------------------------+
| 385cebf9-9b34-4eca-b780-c515dbc7eec0 | Linux bridge agent | openstack-server | :-) | True | neutron-linuxbridge-agent |

| b3ff8ffe-1ff2-4659-b823-331def4e6a93 | DHCP agent | openstack-server | :-) | True | neutron-dhcp-agent |
| b5bed625-47fd-4e79-aa55-01cf8a8cc577 | Metadata agent | openstack-server | :-) | True | neutron-metadata-agent |
+--------------------------------------+--------------------+-----------------+-------+----------+---------------------------+

3.9 创建虚拟机

3.9.1 创建桥接网络

1、 创建网络

```
[root@linux-node1 ~]# source admin-openrc.sh                    #在哪个项目下创建虚拟机，这里选择在 demo 下创建；也可以在 admin 下
[root@linux-node1 ~]# neutron net-create flat --shared --provider:physical_network physnet1 --provider:network_type flat
```

2、 创建子网（填写宿主机的内网网关，下面 DNS 和内网网关可以设置成宿主机的内网 ip，下面 192.168.1.100-200 是分配给虚拟机的 ip 范围）

```
[root@linux-node1 ~]# neutron subnet-create flat 192.168.1.0/24 --name flat-subnet --allocation-pool start=192.168.1.100,end=192.168.1.200--dns-nameserver 192.168.1.1 --gateway 192.168.1.1
```

3、 查看子网

```
[root@linux-node1 ~]# neutron net-list
+--------------------------------------+------+-------------------------------------------------------+
| id | name | subnets |
+--------------------------------------+------+-------------------------------------------------------+
| 1d9657f6-de9e-488f-911f-020c8622fe78 | flat | c53da14a-01fe-4f6c-8485-232489deaa6e 192.168.1.0/24 |
+--------------------------------------+------+-------------------------------------------------------+
```

```
[root@linux-node1 ~]# neutron subnet-list
+--------------------------------------+-------------+---------------+--------------------------------------------------------+
| id | name | cidr | allocation_pools |
+--------------------------------------+-------------+---------------+--------------------------------------------------------+
| c53da14a-01fe-4f6c-8485-232489deaa6e | flat-subnet | 192.168.1.0/24 | {"start": "192.168.1.100", "end": "192.168.1.200"} |
```

```
+--------------------------------------+------------+---------------+----------------------
------------------------------+
```

需要关闭 VMware 的 dhcp

3.9.2 创建虚拟机（为 vm 分配内网 ip，后续利用 squid 代理或宿主机 NAT 端口转发进行对外或对内访问）

1、创建 key

```
[root@linux-node1 ~]# source demo-openrc.sh
```

（这是在 demo 账号下创建虚拟机；要是在 admin 账号下创建虚拟机，就用 source admin-openrc.sh）

```
[root@linux-node1 ~]# ssh-keygen -q -N ""
```

2、将公钥添加到虚拟机

```
[root@linux-node1 ~]# nova keypair-add --pub-key /root/.ssh/id_rsa.pub mykey
[root@linux-node1 ~]# nova keypair-list
+-------+-------------------------------------------------+
| Name | Fingerprint |
+-------+-------------------------------------------------+
| mykey | cd:7a:1e:cd:c0:43:9b:b1:f4:3b:cf:cd:5e:95:f8:00 |
+-------+-------------------------------------------------+
```

3、创建安全组

```
[root@linux-node1 ~]# nova secgroup-add-rule default icmp -1 -1 0.0.0.0/0
[root@linux-node1 ~]# nova secgroup-add-rule default tcp 22 22 0.0.0.0/0
```

4、 创建虚拟机

查看支持的虚拟机类型

```
[root@linux-node1 ~]# nova flavor-list
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------
+
| ID | Name | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor | Is_Public
|
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------
+
| 1 | m1.tiny | 512 | 1 | 0 | | 1 | 1.0 | True |
| 2 | m1.small | 2048 | 20 | 0 | | 1 | 1.0 | True |
| 3 | m1.medium | 4096 | 40 | 0 | | 2 | 1.0 | True |
| 4 | m1.large | 8192 | 80 | 0 | | 4 | 1.0 | True |
| 5 | m1.xlarge | 16384 | 160 | 0 | | 8 | 1.0 | True |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------
+
```

查看镜像

```
[root@linux-node1 ~]# nova image-list
+--------------------------------------+--------+--------+--------+
```

| ID | Name | Status | Server |
+--------------------------------------+--------+--------+--------+
| 2707a30b-853f-4d04-861d-e05b0f1855c8 | cirros | ACTIVE | |
+--------------------------------------+--------+--------+--------+
查看网络
[root@linux-node1 ~]# neutron net-list
+--------------------------------------+------+---+
| id | name | subnets |
+--------------------------------------+------+---+
| **1d9657f6-de9e-488f-911f-020c8622fe78** | flat | c53da14a-01fe-4f6c-8485-232489deaa6e 192.168.1.0/24 |
+--------------------------------------+------+---+

创建虚拟机 【这一步容易报错，一般都是由于上面的 nova.conf 配置填写有误所致】
[root@linux-node1 ~]# nova boot --flavor **m1.tiny** --image **cirros** --nic net-id=**1d9657f6-de9e-488f-911f-020c8622fe78** --security-group default --key-name **mykey** hello-instance

5、查看虚拟机

[root@linux-node1 ~]# nova list
+--------------------------------------+--------------+--------+------------+-------------+--------------------+
| ID | Name | Status | Task State | Power State | Networks |
+--------------------------------------+--------------+--------+------------+-------------+--------------------+
| 7a6215ac-aea7-4e87-99a3-b62c06d4610e | hello-instance| ACTIVE | - | Running | flat=192.168.1.102 |
+--------------------------------------+--------------+--------+------------+-------------+--------------------+

****
如果要删除虚拟机（利用虚拟机 ID 进行删除）
[root@linux-node1 ~]# nova delete 7a6215ac-aea7-4e87-99a3-b62c06d4610e
****

```
[root@linux-node1 src]# nova list
+--------------------------------------+---------------+--------+------------+-------------
+--------------------+
| ID | Name | Status | Task State | Power State | Networks |
+--------------------------------------+---------------+--------+------------+-------------
+--------------------+
| 007db18f-ae3b-463a-b86d-9a8455a21e2d | hello-instance | ACTIVE | - | Running |
flat=192.168.1.101 |
+--------------------------------------+---------------+--------+------------+-------------
+--------------------+
```

[root@linux-node1 ~]# ssh cirros@192.168.1.101 登录查看

****

上面创建虚拟机的时候，openstack 在 neutron 组网内是采用 dhcp-agent 自动分配 ip 的！

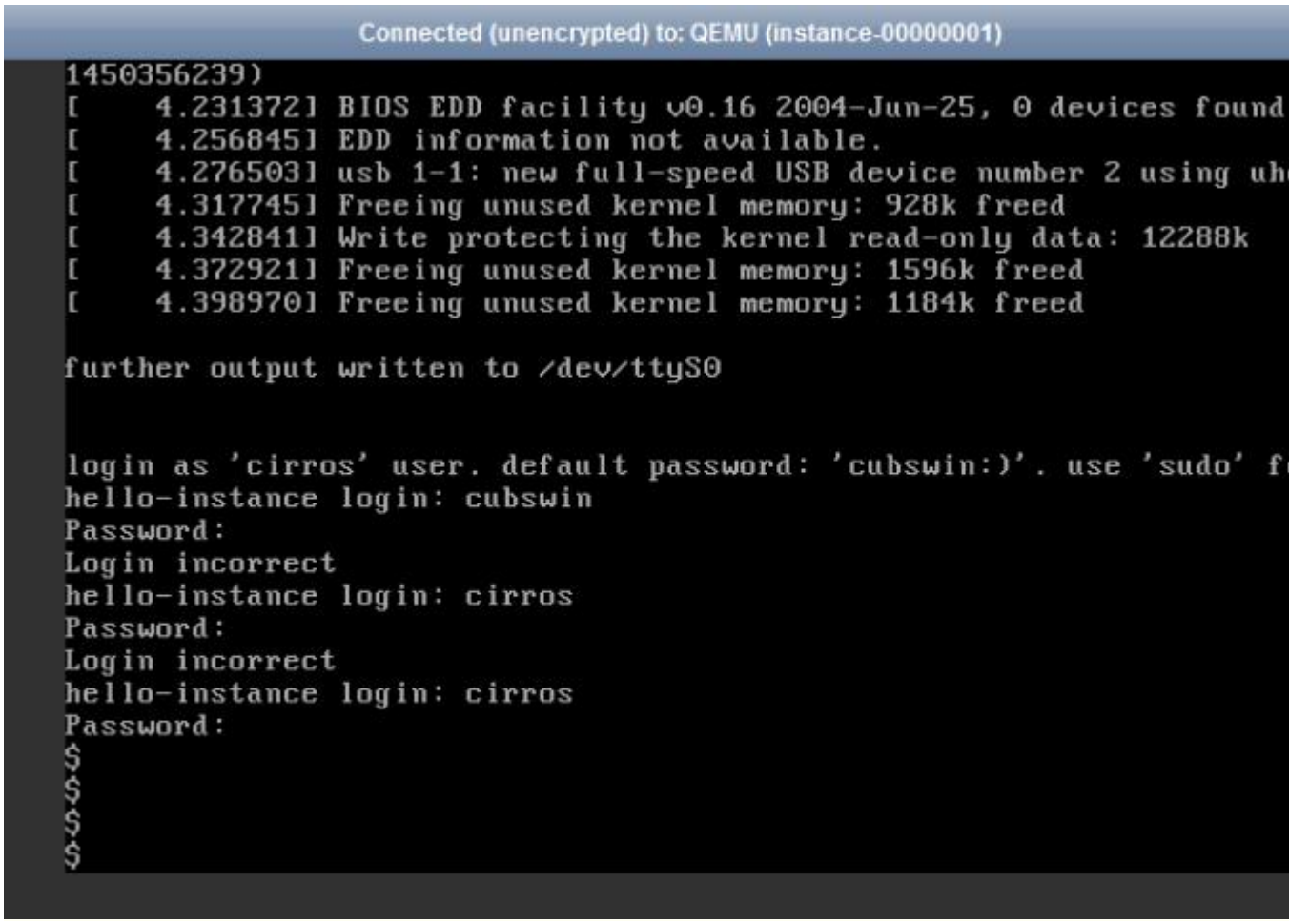
可以在创建虚拟机的时候，指定固定 ip，方法详见于另一篇博客：

http://www.cnblogs.com/kevingrace/p/5822660.html

****

6、 web 界面打开虚拟机

```
[root@linux-node1 ~]# nova get-vnc-console hello-instance novnc
+-------+-----------------------------------------------------------------------------------
-+
| Type | Url
| +
-------+-----------------------------------------------------------------------------------
-+
| novnc
| http://58.68.250.17:6080/vnc_auto.html?token=303d5a78-c85f-4ed9-93b6-be9d5d
28fba6 |        #访问这个链接即可打开 vnc 界面
+-------+-----------------------------------------------------------------------------------
-+
```



4.0 安装 dashboard，登陆 web 管理界面

[root@linux-node1 ~]# yum install openstack-dashboard -y

[root@linux-node1 ~]# vim /etc/openstack-dashboard/local_settings #按照下面几行进行配置修改

OPENSTACK_HOST = "192.168.1.17" #更改为 keystone 机器地址

OPENSTACK_KEYSTONE_DEFAULT_ROLE = "user" #默认的角色

ALLOWED_HOSTS = ['*'] #允许所有主机访问

CACHES = {

'default': {

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

'LOCATION': '192.168.1.17:11211', #连接 memcached

}

}

#CACHES = {

'default': {

'BACKEND': 'django.core.cache.backends.locmem.LocMemCache',

}

#}
TIME_ZONE = "Asia/Shanghai"　　#设置时区

重启 httpd 服务
[root@linux-node1 ~]# systemctl restart httpd

web 界面登录访问 dashboard
http://58.68.250.17/dashboard/
用户密码 demo 或者 admin（管理员）

如果要修改 dashboard 的访问端口（比如将 80 端口改为 8080 端口），则需要修改下面两个配置文件：
1）vim /etc/httpd/conf/httpd.conf
将 80 端口修改为 8080 端口

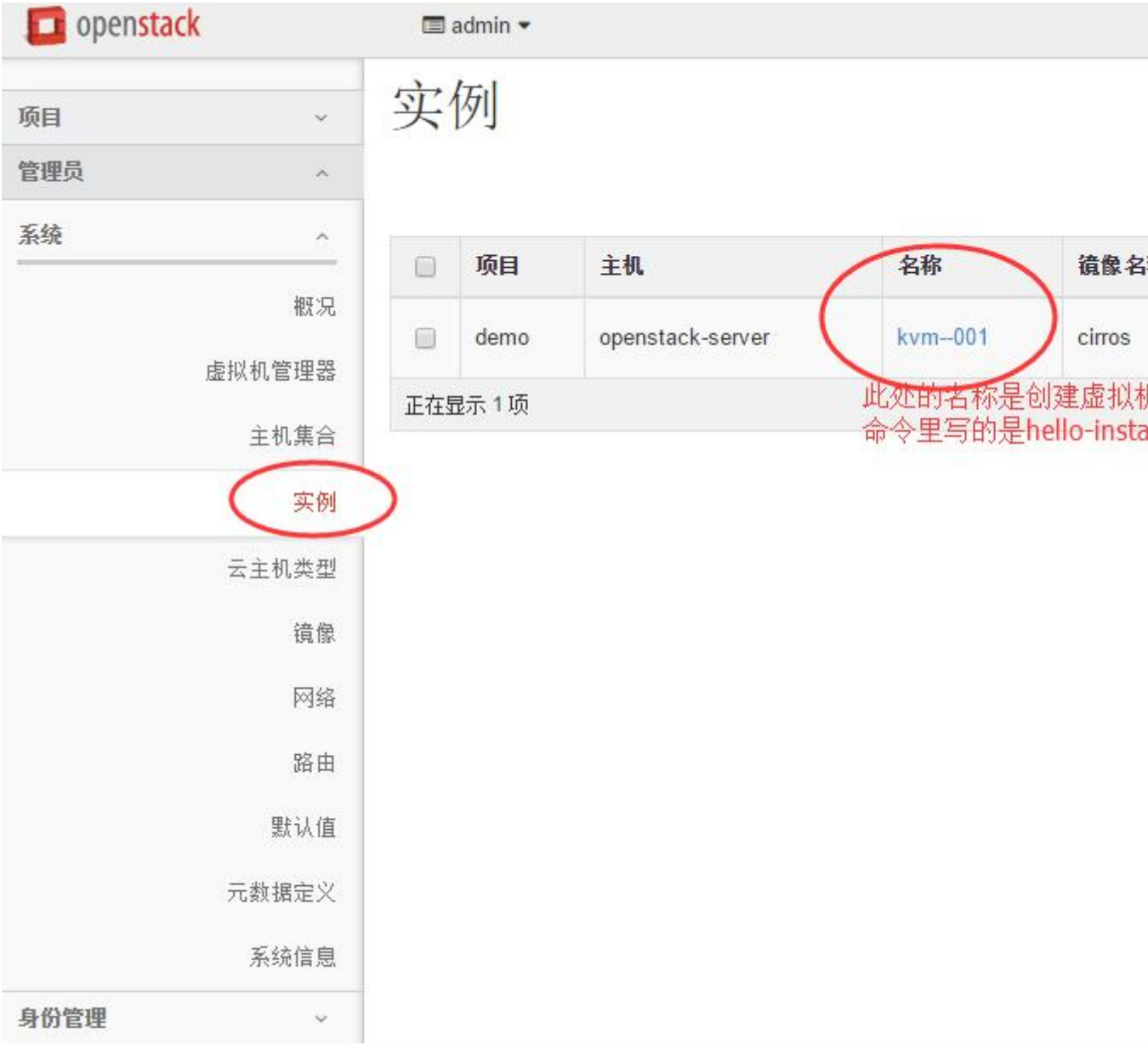
Listen 8080
ServerName 192.168.1.17:8080

2）vim /etc/openstack-dashboard/local_settings #将下面两处的端口由 80 改为 8080
'from_port': '8080',
'to_port': '8080',

然后重启 http 服务：
systemctl restart httpd

如果开启了防火墙，还需要开通 8080 端口访问规则

这样，dashboard 访问 url：
http://58.68.250.17:8080/dashboard



前面建立了两个账号：admin 和 demo，两个账号都可以登陆 web！只不过， admin 是管理员账号， admin 登陆后可以看到其他账号下的状态

demo 等普通账号登陆后只能看到自己的状态

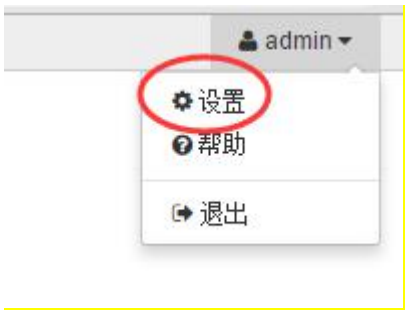
注意：

上面的 Rabbit 账号 admin 和 openstack 是消息队列的 web 登陆账号。

比如一下子要建立 10 个虚拟机的指令，但是当前资源处理不过来，就通过 Rabbit 进行排队！！

修改 OpenStack 中 dashboard 用户登陆密码的方法：

登陆 dashboard：



创建虚拟机的时候，我们可以自己定义虚拟机的类型（即配置）。

登陆 openstack 的 web 管理界面里进行自定义，也可以将之前的删除。

查看上传到 glance 的镜像


openstack
admin
项目
管理员
系统
概况
虚拟机管理器
主机集合
实例
云主机类型
镜像
镜像
项目
镜像名称
类型
admin
CentOS-6.5
镜像
正在显示 1 项

查看创建的虚拟机实例

自定义虚拟主机类型，设置如下：

（如果想让虚拟机有空闲磁盘空间，用于新建分区之用，则可以在这里分配临时磁盘）

创建云主机类型

主机类型信息 * | 云主机类型访问

名称 *

kvm002

ID

auto

虚拟内核 *

2

内存 (MB) *

6144

根磁盘(GB) *

10

临时磁盘(GB)

0

Swap磁盘(MB)

1000

云主机类型定义RAM和磁盘的大小、核数，以及其他
源，在用户部署实例的时候使用。

取消 | 创建云主机

我创建了四个虚拟机实例，采用的是同一个虚拟主机类型（即上面的 kvm002），四个实例总共占用宿主机 40G 的空间。

登陆到 openstack，可以看到，左侧一共有四个标签栏：


ashboard/admin/
openstack
admin
项目
管理员
系统
身份管理
概况
使用情况摘要
选择一段时间来查询其用量：
从: 2016-08-01
到: 2016-08-31
运行中的实例: 4 活跃的内存: 24GB 这一时期的VCPU
用量


openstack
admin
项目
计算
概况
实例
云硬盘
镜像
访问 & 安全
网络
管理员
身份管理
概况
使用情况摘要
选择一段时间来查询其
从: 2016-08-01
运行中的实例: 4 活跃的内存:
用量
项目名称
admin
demo
正在显示 2 项

可以登陆 dashboard 界面，在“计算”->“实例”里选择“启动云主机"或者“计算->网络->网络拓扑”里选择“启动虚拟机”就可以再创建一个虚拟机

也可以按照快照再启动（创建）一个虚拟机，不过这样启动起来的虚拟机是一个新的 ip（快照前的源虚拟机就要关机了）

查看实例，发现 kvm-server005 虚拟机已经创建成功了。默认创建后的 ip 是 dhcp 自动分配的，可以登陆虚拟机改成 static 静态 ip

在 openstack 中重启实例有两种，分别被称为"软重启"和"硬重启"。所谓的软重启会尝试正常关机并重启实例，硬重启会直接将实例"断电"并重启。也就是说硬重启会"关闭"电源。其具体命令如下：
默认情况下，如果您通过 nova 重启，执行的是软重启。

```
$ nova reboot SERVER
```

如果您需要执行硬重启，添加--hard 参数即可：

```
$ nova reboot --hard SERVER
```

nova 命令管理虚拟机：

```
$ nova list #查看虚拟机
$ nova stop [vm-name]或[vm-id] #关闭虚拟机
$ nova start [vm-name]或[vm-id] #启动虚拟机
$ nova suspend [vm-name]或[vm-id] #暂停虚拟机
$ nova resume [vm-name]或[vm-id] #启动暂停的虚拟机
$ nova delete [vm-name]或[vm-id] #删除虚拟机

$nova-manage service list     #检查服务是否正常

[root@openstack-server ~]# source /usr/local/src/admin-openrc.sh
[root@openstack-server ~]# nova list
```

```
+--------------------------------------+---------------+--------+------------+-------------
+--------------------+
| ID | Name | Status | Task State | Power State | Networks |
+--------------------------------------+---------------+--------+------------+-------------
+--------------------+
| 11e7ad7f-c0a8-482b-abca-3a4b7cfdd55d | hello-instance | ACTIVE | - | Running |
flat=192.168.1.107 |
| 67f71703-c32c-4bf1-8778-b2a6600ad34a | kvm-server0 | ACTIVE | - | Running |
flat=192.168.1.120 |
+--------------------------------------+---------------+--------+------------+-------------
+--------------------+
[root@openstack-server ~]# ll /var/lib/nova/instances/          #下面是虚拟机的存放路
径
total 8
drwxr-xr-x. 2 nova nova 85 Aug 29 15:22 11e7ad7f-c0a8-482b-abca-3a4b7cfdd55d
drwxr-xr-x. 2 nova nova 85 Aug 29 15:48 67f71703-c32c-4bf1-8778-b2a6600ad34a
drwxr-xr-x. 2 nova nova 80 Aug 29 15:40 _base
-rw-r--r--. 1 nova nova 39 Aug 29 16:44 compute_nodes
drwxr-xr-x. 2 nova nova 4096 Aug 29 13:58 locks


-------------------------------------------------------------------------------------------
----------------------------------
virsh 命令行管理虚拟机：

[root@openstack-server ~]# virsh list #查看虚拟机


Id Name State
----------------------------------------------------
9 instance-00000008 running
41 instance-00000015 running
[root@openstack-server ~]# ll /etc/libvirt/qemu/ #虚拟机文件
total 16
-rw-------. 1 root root 4457 Aug 26 17:46 instance-00000008.xml
-rw-------. 1 root root 4599 Aug 29 15:40 instance-00000015.xml
drwx------. 3 root root 22 Aug 24 12:06 networks


其中：
virsh list #显示本地活动虚拟机
virsh list --all #显示本地所有的虚拟机（活动的+不活动的）
virsh define instance-00000015.xml #通过配置文件定义一个虚拟机（这个虚拟机还不是活动
的）
virsh edit instance-00000015 # 编辑配置文件（一般是在刚定义完虚拟机之后）
virsh start instance-00000015 #启动名字为 ubuntu 的非活动虚拟机
```

virsh reboot instance-00000015 #重启虚拟机
virsh create instance-00000015.xml #创建虚拟机（创建后，虚拟机立即执行，成为活动主机）
virsh suspend instance-00000015 #暂停虚拟机
virsh resume instance-00000015 #启动暂停的虚拟机
virsh shutdown instance-00000015 #正常关闭虚拟机
virsh destroy instance-00000015 #强制关闭虚拟机
virsh dominfo instance-00000015 #显示虚拟机的基本信息
virsh domname 2 #显示 id 号为 2 的虚拟机名
virsh domid instance-00000015 #显示虚拟机 id 号
virsh domuuid instance-00000015 #显示虚拟机的 uuid
virsh domstate instance-00000015 #显示虚拟机的当前状态
virsh dumpxml instance-00000015 #显示虚拟机的当前配置文件（可能和定义虚拟机时的配置不同，因为当虚拟机启动时，需要给虚拟机分配 id 号、uuid、vnc 端口号等等）
virsh setmem instance-00000015 512000 #给不活动虚拟机设置内存大小
virsh setvcpus instance-00000015 4 # 给不活动虚拟机设置 cpu 个数
virsh save instance-00000015 a #将该 instance-00000015 虚拟机的运行状态存储到文件 a 中
virsh restore a #恢复被存储状态的虚拟机的状态，即便虚拟机被删除也可以恢复（如果虚拟机已经被 undefine 移除，那么恢复的虚拟机也只是一个临时的状态，关闭后自动消失）
virsh undefine instance-00000015 #移除虚拟机，虚拟机处于关闭状态后还可以启动，但是被该指令删除后不能启动。在虚拟机处于 Running 状态时，调用该指令，该指令暂时不生效，但是当虚拟机被关闭后，该指令生效移除该虚拟机，也可以在该指令生效之前调用 define+TestKVM.xml 取消该指令

注意：
virsh destroy instance-00000015 这条命令并不是真正的删除这个虚拟机，只是将这个虚拟机强制关闭了。可以通过该虚拟机的 xml 文件恢复。如下：
[root@kvm-server ~]# virsh list
Id Name State
----
1 dev-new-test2 running
2 beta-new2 running
5 test-server running
8 ubuntu-test03 running
9 elk-node1 running
10 elk-node2 running
11 ubuntu-test01 running
12 ubuntu-test02 running

强制关闭虚拟机
[root@kvm-server ~]# virsh destroy ubuntu-test02
Domain ubuntu-test02 destroyed

发现 ubuntu-test02 虚拟机已经关闭了

```
[root@kvm-server ~]# virsh list
Id Name State
----------------------------------------------------
1 dev-new-test2 running
2 beta-new2 running
5 test-server running
8 ubuntu-test03 running
9 elk-node1 running
10 elk-node2 running
11 ubuntu-test01 running
```

但是该虚拟机的 xml 文件还在，可以通过这个文件恢复

```
[root@kvm-server ~]# ll /etc/libvirt/qemu/ubuntu-test02.xml
-rw------- 1 root root 2600 Dec 26 13:55 /etc/libvirt/qemu/ubuntu-test02.xml
[root@kvm-server ~]# virsh define /etc/libvirt/qemu/ubuntu-test02.xml #这只是重新添
```

加了这个虚拟机，目前还不是活动的虚拟机，需要启动下

```
[root@kvm-server ~]# virsh start ubuntu-test02
Domain ubuntu-test02 started
[root@kvm-server ~]# virsh list
Id Name State
----------------------------------------------------
1 dev-new-test2 running
2 beta-new2 running
5 test-server running
8 ubuntu-test03 running
9 elk-node1 running
10 elk-node2 running
11 ubuntu-test01 running
12 ubuntu-test02 running
```