Openstack 安装文档

（H 版）508 实验室

电子科技大学 2014/3/4

文档修订记录

版本号	内容	时间	作者
V0.0	理清官方文档的安装思路，基本安装文体格式确定。	2014/3/3	邓玉林
V1.0	面向电子科技大学 508 实验室的具体实验环境，详细介绍整个环境搭建的规划以及具体实现步骤。	2014/3/10	邓玉林
V1.1	组织文档整体结构，完善其层次，添加常见文体解决章节。	2014/3/13	邓玉林

目录

第一章：openstack 安装和设置简介

1.1 文档介绍

本文档将以 508 实验室实验环境为模板，介绍 openstack（H 版）整个云平台的初步安装步骤与常见问题解决办法，本文档涉及到的 openstack 服务有：管理服务、计算服务、网络服务、对象存储服务、块存储服务、身份认证服务、镜像服务与计费服务。每个服务都是一个单独的模块，通过消息传递实现各模块的协同工作。

文档旨在帮助读者快速了解与掌握 openstack 具体安装过程，通过安装与解决实际问题加深对 openstack 整个云平台的了解。由于实验环境会有所不同，遇到的问题有可能不一样，如有出入请参照 openstack 的官方安装文档。

1.2 openstack 基本安装部署过程概览

Openstack 是一个复杂的产品，具有多个要安装和设置的模块。为了确保成功安装部署 openstack，需了解所需的任务序列。

安装 openstack 任务如下：

1. 验证系统是否满足 openstack 的硬件与软件要求。详见第二章。
2. 确定要使用的安装需求，openstack 除了个别（如：身份认证服务）服务必须安装外，其他服务可以不安装或者单独安装。
3. 正确规划好 openstack 平台，详见第二章：平台规划。
4. 完成 openstack 安装前，操作系统的准备工作，如数据库，通信服务，时钟同步等服务的安装，详见第三章：基本操作系统配置。
5. 开始 openstack 的安装工作，详见第四章：openstack 详细安装步骤。
6. 期间可能会遇到各种问题，有些问题可能已经在本文档中列出，详见第五章：常见问题解决办法。

第二章：平台要求与规划

2.1 硬件要求

处理器：　双核或者 4 核
内存：　8-12GB
硬盘要求：　20GB 以上，特定节点需要 2 个硬盘
网卡：　特定节点需要双网卡

2.2 操作系统要求

操作系统使用的是 ubuntu12.04 版本，操作系统为 64 位 server 版，使用的安装方式是 #apt-get 在线安装。

2.3 实验节点与 IP 规划

节点	Eth0(外部通信)	Eth1（内部通信）	硬盘要求（单位：块）
控制节点（controller）	192.168.5.120	10.0.0.10	1
计算节点（compute1）	192.168.5.118	不设置	1
块存储节点（block1）	192.168.5.125	不设置	2
对象存储节点 storage1,storage2,storage3	192.168.5.191 192.168.5.192 192.168.5.193	不必须	2
对象存储代理节点（proxy）	192.168.5.6	不必须	1

注意：此处因物理节点足够，故设置了 7 个节点，如果物理节点不够时，可将代理节点安装至控制节点，块存储节点可以和对象存储节点放在一个节点上，但是硬盘必须多挂载一块。计算节点也可作为存储节点，只要不冲突，可视情况安排。

2.4 各模块逻辑关系图

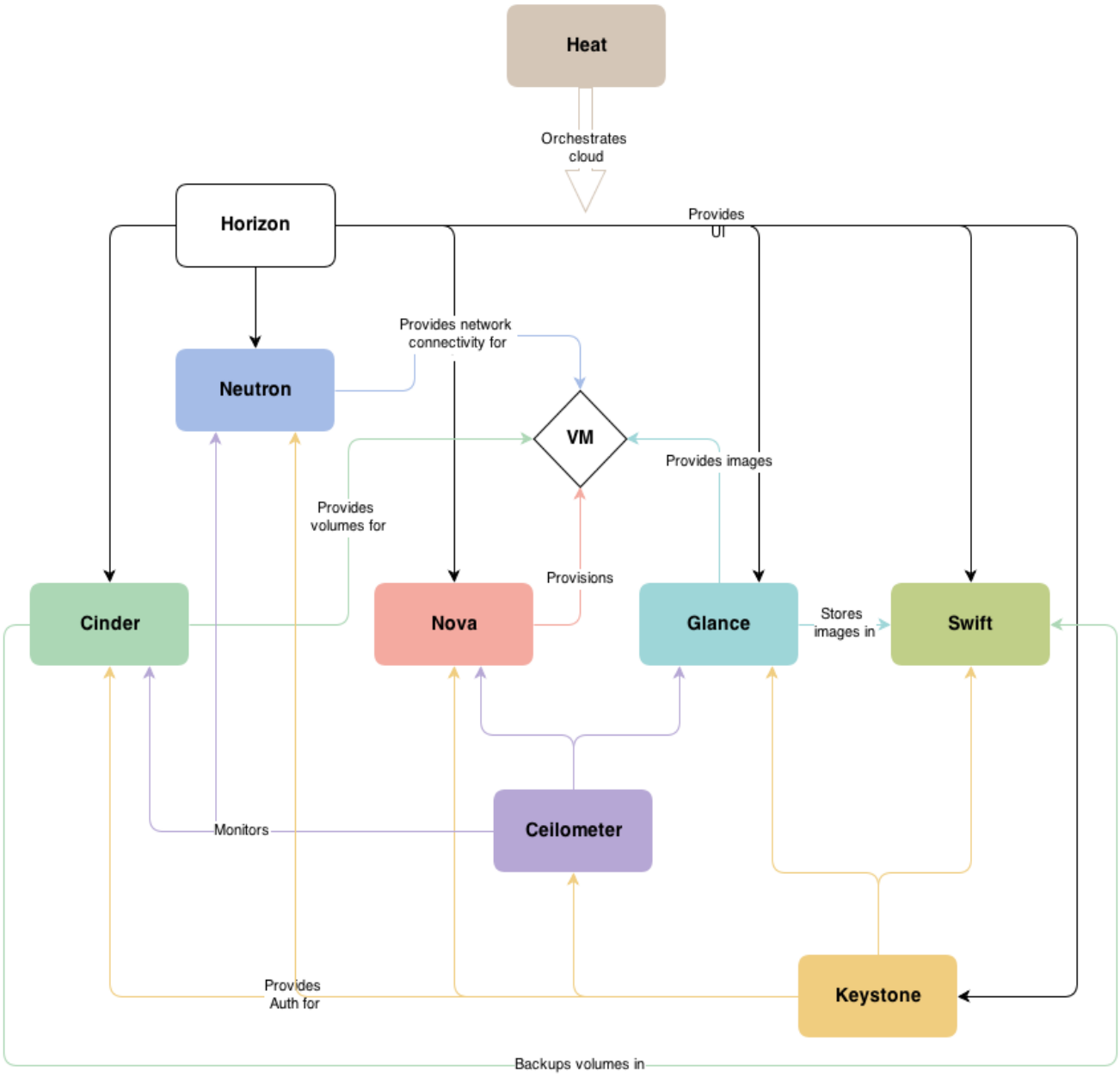

图 1

第三章：基本操作系统配置

3.1 网络配置

网络配置每个实验环境都不相同，总的来说，平台需要有两个网络：

1） 第一个网络为内部网络，负责各节点之间的通信以及 internet 的访问。

2） 第二个网络为外部网络，提供给实例（即虚拟机）使用，各节点可以使用此网络与虚拟机进行通信，并且通过网络共享的方式，可以使虚拟机访问 internet。

3.1.1 控制节点网络配置

修改网络配置文件 /etc/network/interfaces

```
# Internal Network
auto eth0
iface eth0 inet static
address 192.168.5.120
netmask 255.255.255.0
gateway 192.168.5.1
dns-nameservers 202.112.14.151
# External Network
auto eth1
iface eth1 inet static
address 10.0.0.10
netmask 255.255.255.0
```

3.1.2 其他节点网络配置（以计算节点为例）

修改网络配置文件 /etc/network/interfaces

```
# Internal Network
auto eth0
iface eth0 inet static
address 192.168.5.118
netmask 255.255.255.0
gateway 192.168.5.1
dns-nameservers 202.112.14.151
```

修改完成后重启网络，使更改生效：

```
# service networking restart
```

3.1.3 修改节点名与 hosts 文件（此处以控制节点为例）

设置节点名并修改/etc/hostname 文件：

```
# hostname controller
```

为了确保各个节点能与其他节点正常通信，需修改每个节点的/etc/hosts 文件. 格式如下：

```
127.0.0.1       localhost
192.168.5.120   controller
192.168.5.118   compute1
```

3.2 网络定时协议（NTP）

为了确保各个节点的时间相同，必须为每个节点安装 NTP 网络定时协议，避免时间不同步问题。在每个安装了 openstack 服务的节点上，安装 ntp 包：

```
# apt-get install ntp
```

修改/etc/ntp.conf 文件，设置控制节点的时间为主要依据时间：

```
server 192.168.5.120 prefer
```

3.3 MySQL 数据库

3.3.1 控制节点安装

在控制节点上，需要安装 MySQL 服务端与客户端，Python 库：

```
# apt-get install python-mysqldb mysql-server
```

修改/etc/mysql/my.cnf 文件，将 bind-address 改成控制节点的 IP，使得其他节点也可以访问数据库：

```
[mysqld]
...
bind-address = 192.168.5.120
```

重启 MySQL 服务，使修改生效：

```
# service mysql restart
```

修改数据库的权限，删除匿名用户权限，使用如下命令：

```
# mysql_install_db
# mysql_secure_installation
```

注意：这条命令将会显示几个选项，除了修改密码，其他全部选择yes.

3.3.2 其他节点安装

除了控制节点，其他节点一定不要安装和启动数据库的服务端，否则会有冲突，只需安装 MySQL 数据库客户端与 python 库。

```
//不要安装MySQL服务端

# apt-get install python-mysqldb
```

3.4 安装 openstack 包

因为是在线安装，所以需确定安装的版本为havana版本。

1. Install the Ubuntu Cloud Archive for Havana:

```
# apt-get install python-software-properties
# add-apt-repository cloud-archive:havana
```

2. Update the package database, upgrade your system, and reboot for all changes to take effect:

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

3.5 通信服务

在控制节点，安装消息队列服务，此处使用的是RabbitMQ：

```
# apt-get install rabbitmq-server
```

修改RabbitMQ中guest的默认密码：

```
# rabbitmqctl change_password guest RABBIT_PASS
```

至此，openstack 所需的基本环境已经搭建完成，下面开始 openstack 平台的搭建

第四章：openstack 详细安装与配置步骤

4.1 配置身份认证服务

4.1.1 安装身份认证服务

1. 在控制节点安装openstack的身份认证服务：

```
# apt-get install keystone
```

2. 身份认证服务需要数据库存储数据，在配置文件中指定数据库的位置。在本文档中，我们使用控制节点 MySQL 数据库,用户名为:keystone 密码为:*KEYSTONE_DBPASS*，此处可以设置其他密码，但是注意在之后的配置中也修改为相应密码。

   编辑 /etc/keystone/keystone.conf 文件，修改[sql]部分.

```
...
[sql]
# The SQLAlchemy connection string used to connect to the database
connection = mysql://keystone:KEYSTONE_DBPASS@controller/keystone
...
```

3. 在默认情况下，ubuntu会创建一个SQLite数据库，为了避免错误，须将文件 /var/lib/keystone/ keystone.db删除。

4. 使用你之前设置的数据库密码作为root登录数据库，创建一个数据库用户和数据库：

```
# mysql -u root -p
mysql> CREATE DATABASE keystone;
mysql> GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' \
IDENTIFIED BY 'KEYSTONE_DBPASS';
mysql> GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' \
IDENTIFIED BY 'KEYSTONE_DBPASS';
```

5. 为身份认证服务数据库创建表：

```
# keystone-manage db_sync
```

6. 定义一个授权的token当作身份认证服务与其他openstack服务的公共密钥。使用**openssl**产生一个随机的token，并且将这个token存储在配置文件中：

```
# openssl rand -hex 10
```

编辑 /etc/keystone/keystone.conf 文件并且修改 [DEFAULT]部分，将ADMIN_TOKEN修改为刚才产生的token。

```
[DEFAULT]
# A "shared secret" between keystone and other openstack services
admin_token = ADMIN_TOKEN
...
```

7. 重启身份认证服务：

```
# service keystone restart
```

4.1.2 定义用户，租户和角色（users, tenants, roles）

在你安装完身份认证服务，设置完用户、租户、角色之后，就可以使用服务以及端口登录。一般情况下，我们会使用一个授权了的账户与密码登录，但是，到目前为止我们没有创建任何用户，所以，我们只能使用之前生成的 token 登录。运行一下代码，将 ADMIN_TOKEN 替代为之前产生的 token：

```
# export OS_SERVICE_TOKEN= ADMIN_TOKEN
# export OS_SERVICE_ENDPOINT=http://controller:35357/v2.0
```

首先，我们创建一个管理员权限的租户和一个普通 openstack 服务的租户：

```
# keystone tenant-create --name=admin --description="Admin Tenant"
# keystone tenant-create --name=service --description="Service Tenant"
```

然后，创建一个管理员用户 admin 密码为 ADMIN_PASS 并为它定义一个邮箱：

```
# keystone user-create --name=admin --pass=ADMIN_PASS \
--email=admin@example.com
```

创建一个管理员角色 admin，任何的角色创建必须对应 policy.json 文件中的设置。默认的策略中，角色 admin 有大部分的权限。

```
# keystone role-create --name=admin
```

最后，你必须为每个用户定义角色，将 admin 角色赋予给 admin 用户。

```
# keystone user-role-add --user=admin --tenant=admin --role=admin
```

4.1.3 定义服务以及端口

为了使身份认证服务可以追踪那些已经安装的openstack服务，并且在网络中定位它们，你必须在安装完成之后注册它们，注册一个服务，可以运行以下命令：

keystone service-create. Describes the service.
keystone endpoint-create. Associates API endpoints with the service.

你也必须注册身份认证服务它本身。使用之前的 OS_SERVICE_TOKEN 环境变量获取权限：

1. 为身份认证服务创建一个入口：

```
# keystone service-create --name=keystone --type=identity \
  --description="Keystone Identity Service"
+---------------+----------------------------------+
| Property      | Value                            |
+---------------+----------------------------------+
| description   | Keystone Identity Service        |
| id            | 15c11a23667e427e91bc31335b45f4bd |
| name          | keystone                         |
| type          | identity                         |
+---------------+----------------------------------+
```

服务的ID是随机生成的，会和上面显示的不一样。

2. 使用上面返回的服务ID为身份认证服务定义一个API端口。当你定义一个端口，你要提供公共API端口，内部API端口已经管理员API端口。在本文档中，使用了controller的主机名，注意此处为admin定义了不一样的端口。

```
# keystone endpoint-create \
--service-id=the_service_id_above \
--publicurl=http://controller:5000/v2.0 \
--internalurl=http://controller:5000/v2.0 \
--adminurl=http://controller:35357/v2.0
+---------------+----------------------------------+
| Property      | Value                            |
+---------------+----------------------------------+
| adminurl      | http://controller:35357/v2.0     |
| id            | 11f9c625a3b94a3f8e66bf4e5de2679f |
| internalurl   | http://controller:5000/v2.0      |
| publicurl     | http://controller:5000/v2.0      |
| region        | regionOne                        |
```

```
| service_id     | 15c11a23667e427e91bc31335b45f4bd  |
+---------------+----------------------------------+
```

3. 当你添加其他已安装的服务时，也须通过身份认证服务使用以上命令注册该服务。

4.1.4 验证身份认证服务安装

为了验证身份认证服务是否安装配置正确，首先取消环境变量OS_SERVICE_TOKEN 与 OS_SERVICE_ENDPOINT。 它们仅仅是为了用来创建管理员权限的用户。

```
# unset OS_SERVICE_TOKEN OS_SERVICE_ENDPOINT
```

使用基于用户名方式进行授权登录，通过使用用户名与密码请求一个授权的token：

```
# keystone --os-username=admin --os-password=ADMIN_PASS \
--os-auth-url=http://controller:35357/v2.0 token-get
```

你将会收到一个与你的用户名相匹配的token。这样就表示keystone运行在了所期望的端口上。并且你的账户取得了预期的证书。

下面验证这个授权是不是在一个所期望的租户上。

```
# keystone --os-username=admin --os-password=ADMIN_PASS \
--os-tenant-name=admin --os-auth-url=http://controller:35357/v2.0 token-get
```

你会收到一个新的包含租户ID的token。这样就表示你的帐号在一个限定的租户上正确的定义了角色。即租户如所预期的那样设置完成了。

你可以通过为一个文件（如：keystonerc）设置用户授权以及端口，实现方便的登录。

```
export OS_USERNAME=admin
export OS_PASSWORD=ADMIN_PASS
export OS_TENANT_NAME=admin
export OS_AUTH_URL=http://controller:35357/v2.0
```

将这个文件添加至环境变量中：

```
# source keystonerc
```

验证keystonerc文件如之前的命令一样可行：

```
$ keystone token-get
```

最后，验证你的 admin 账户具有管理员权限：

```
# keystone user-list

+----------------------------------+---------+--------------------+--------+
| id                               | enabled | email              |name    |
+----------------------------------+---------+--------------------+--------+
| a4c2d43f80a549a19864c89d759bb3fe | True    | admin@example.com  | admin  |
```

这样就可以确信，你的账户具有管理员权限，并且符合policy.json 文件。

4.2 配置镜像服务

4.2.1 镜像服务安装

Openstack的镜像服务相当于一个登记了的虚拟磁盘镜像，用户可以添加新的镜像或者通过给已存在的虚拟机添加快照的方式创建新的镜像。你可以将镜像存储在对象存储节点中，也可以存储在其他的位置。例如，你可以存储镜像在一个简单的文件系统中或者外部web服务器上。

1. 在控制节点上安装镜像服务：

   ```
   # apt-get install glance python-glanceclient
   ```

2. 镜像服务将镜像信息存放在数据库中，在本安装文档中，使用MySQL服务器。

   配置数据库的位置，镜像服务提供了glance-api与glance-registry这两个服务，这两个服务都要各自的配置文件。你必须修改这两个文件的特定的部分。此处设置的密码为：*GLANCE_DBPASS*

   编辑/etc/glance/glance-api.conf 和/etc/glance/glance-registry.conf 并修改[DEFAULT]部分。

   ```
   ...
   [DEFAULT]
   ...
   # SQLAlchemy connection string for the reference implementation
   # registry server. Any valid SQLAlchemy connection string is fine.
   #See:http://www.sqlalchemy.org/docs/05/reference/sqlalchemy/connectio
   ns.html#sqlalchemy.create_engine
   sql_connection = mysql://glance:GLANCE_DBPASS@controller/glance
   ...
   ```

3. 在默认情况下，ubuntu会创建SQLite数据库，删除/var/lib/glance/glance.sqlite文件。

4. 使用数据库root账户的密码登录数据库，并创建glance数据库用户：

   ```
   # mysql -u root -p
   ```

```
mysql> CREATE DATABASE glance;
mysql> GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' \
IDENTIFIED BY 'GLANCE_DBPASS';
mysql> GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' \
IDENTIFIED BY 'GLANCE_DBPASS';
```

5. 为镜像服务创建数据库表：

```
# glance-manage db_sync
```

6. 创建一个glance用户，使得镜像服务可以获得身份认证服务的授权。glance用户的租户为service，具有admin权限。

```
# keystone user-create --name=glance --pass=GLANCE_PASS \
--email=glance@example.com
# keystone user-role-add --user=glance --tenant=service --role=admin
```

7. 配置镜像服务，使得它通过身份认证服务来授权。

编辑/etc/glance/glance-api.conf和 /etc/glance/glance-registry.conf文件。 如果前面修改了密码也要将密码做相应修改。

a. 添加以下内容至[keystone_authtoken]部分：

```
[keystone_authtoken]
...
auth_uri = http://controller:5000
auth_host = controller
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = glance
admin_password = GLANCE_PASS
```

b. 添加以下内容至 [paste_deploy]部分:

```
[paste_deploy]
...
flavor = keystone
```

8. 增加以下授权信息至/etc/glance/glance-api-paste.ini 文件
与 /etc/glance/glance-registry-paste.ini 文件中。

编辑以上两个文件，将以下项目添加至[filter:authtoken]部分，保留该部分下的其他项目。

```
[filter:authtoken]
paste.filter_factory=keystoneclient.middleware.auth_token:filter_fact
ory
auth_host=controller
admin_user=glance
admin_tenant_name=service
admin_password=GLANCE_PASS
```

9. 为了其他openstack服务可以定位镜像服务，我们必须注册镜像服务，并且创建服务端口:

```
# keystone service-create --name=glance --type=image \
--description="Glance Image Service"
```

10. 使用上面返回的ID创建端口:

```
# keystone endpoint-create \
--service-id=the_service_id_above \
--publicurl=http://controller:9292 \
--internalurl=http://controller:9292 \
--adminurl=http://controller:9292
```

11. 重启glance服务:

```
# service glance-registry restart
# service glance-api restart
```

4.2.2 验证镜像服务的安装

为了验证镜像服务是否安装正确，我们需下载一个镜像文件，此处以CirrOS镜像为例。

1. 下载镜像至指定的位置：

```
$ mkdir images
$ cd images/
$ wget
http://cdn.download.cirros-cloud.net/0.3.1/cirros-0.3.1-x86_64-disk.img
```

2. 上传这个镜像至镜像服务中：

```
# glance image-create --name="CirrOS 0.3.1" --disk-format=qcow2 \
--container-format=bare --is-public=true < cirros-0.3.1-x86_64-disk.img
```

3. 确认这个镜像上传成功，并显示它的属性：

```
 # glance image-list
```

4.3 配置计算服务

//此章节应特别注意各个命令的安装节点，如果安装节点选择错误，有可能会导致安装不下去，而必须重装系统才能修复!!

4.3.1 安装计算控制服务

计算服务是通过收集你所配置好的资源，运行虚拟的计算机实例（虚拟机）的服务。你可以将它配置在一个节点上也可以配置在不同的节点上。本文档中，计算服务中大部分服务运行在控制节点(controller)上，运行的虚拟机则在专用的计算节点（compute）上。这个部分将告诉你如何安装与配置在**控制节点**上的服务。

1. 安装Compute包：

```
# apt-get install nova-novncproxy novnc nova-api \
nova-ajax-console-proxy nova-cert nova-conductor \
nova-consoleauth nova-doc nova-scheduler \
python-novaclient
```

2. 计算服务也要存储数据在数据库中，本文档依旧使用 MySQL 数据库。

   配置数据库位置信息：

   编辑/etc/nova/nova.conf 文件，添加一下内容至[database]部分与[keystone_authtoken] 部分:

```
...
[database]
# The SQLAlchemy connection string used to connect to the database
connection = mysql://nova:NOVA_DBPASS@controller/nova
[keystone_authtoken]
auth_host = controller
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = nova
admin_password = NOVA_PASS
```

3. 配置计算服务使用 RabbitMQ 进行消息传递，将以下键值对添加至/etc/nova/nova.conf 文件的[DEFAULT]部分:

```
rpc_backend = nova.rpc.impl_kombu
rabbit_host = controller
```

```
rabbit_password = RABBIT_PASS
```

4. 删除ubuntu自动创建的数据库：

```
rm /var/lib/nova/nova.sqlite
```

5. 使用root登录数据库，创建nova用户：

```
# mysql -u root -p
mysql> CREATE DATABASE nova;
mysql> GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' \
IDENTIFIED BY 'NOVA_DBPASS';
mysql> GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' \
IDENTIFIED BY 'NOVA_DBPASS';
```

6. 创建数据库中的表：

```
# nova-manage db sync
```

7. 将控制节点（controller）的内部IP地址（192.168.5.120）赋给my_ip, vncserver_listen和vncserver_proxyclient_address ：

   编辑/etc/nova/nova.conf文件，添加一下内容至[DEFAULT]部分：

```
...
[DEFAULT]
...
my_ip=192.168.5.120
vncserver_listen=192.168.5.120
vncserver_proxyclient_address=192.168.5.120
```

8. 创建一个nova用户，使得计算服务可以获得身份认证的授权：

```
# keystone user-create --name=nova --pass=NOVA_PASS
--email=nova@example.com
# keystone user-role-add --user=nova --tenant=service --role=admin
```

9. 配置计算服务使用身份认证服务授权。

   在/etc/nova/nova.conf文件中的[DEFAULT]部分添加一下内容：

```
[DEFAULT]
```

```
...
auth_strategy=keystone
```

10. 将授权信息加入到/etc/nova/api-paste.ini文件中。加入以下选项至[filter:authtoken]部分：

```
[filter:authtoken]
paste.filter_factory =
keystoneclient.middleware.auth_token:filter_factory
auth_host = controller
auth_port = 35357
auth_protocol = http
auth_uri = http://controller:5000/v2.0
admin_tenant_name = service
admin_user = nova
admin_password = NOVA_PASS
```

注意

确保 api_paste_config=/etc/nova/api-paste.ini 文件中的选项也设置在/etc/nova/nova.conf 文件中。

11. 你必须在身份认证服务中注册计算服务：

```
# keystone service-create --name=nova --type=compute \
--description="Nova Compute service"
```

12. 使用返回的ID创建端口：

```
# keystone endpoint-create \
--service-id=the_service_id_above \
--publicurl=http://controller:8774/v2/%\(tenant_id\)s \
--internalurl=http://controller:8774/v2/%\(tenant_id\)s \
--adminurl=http://controller:8774/v2/%\(tenant_id\)s
```

13. 重启计算服务：

```
# service nova-api restart
# service nova-cert restart
# service nova-consoleauth restart
# service nova-scheduler restart
# service nova-conductor restart
# service nova-novncproxy restart
```

14. 验证配置，列出可使用的镜像：

```
# nova image-list
```

4.3.2 配置计算节点

在控制节点配置完计算服务后，我们必须配置另外一个操作系统当作计算节点。计算节点接受从控制节点发送过来的请求，并且运行虚拟机。你可以运行这些服务在单个节点上，但是在本文档中，我们使用不同的节点。以下命令可以通过添加计算节点而轻易水平的扩展计算资源。

计算服务依靠一个超级管理程序来运行虚拟机。Openstack可以使用很多种超级管理程序，但是本文档使用KVM。

1. 配置compute节点的操作系统，使用第三章：基本操作系统配置，但是注意与控制节点有以下不同：

   A. 为eth0配置一个不同的IP地址，本文档使用192.168.5.118作为internal网络IP地址。不要为eth1配置静态IP地址，

   B. 将主机名设置为：compute1，并且在/etc/hosts中配置各个节点的IP地址。

   C. 将时间与控制节点同步，见第三章第二节：ntp。

   D. 安装MySQL客户端与库，不要为控制节点安装服务端。

   E. 确定你使用的openstack包是可以用来搭建环境的，见第三章：安装openstack包。

2. 在配置完操作系统之后，为计算服务安装合适的包，运行一下命令：

   ```
   # apt-get install nova-compute-kvm python-guestfs
   ```

   当弹出是否创建supermin appliance时，选择yes。

3. 因为此处有一个bug，将当前的内核修改为可读的，运行：

   ```
   # dpkg-statoverride --update --add root root 0644 /boot/vmlinuz-$(uname -r)
   ```

   确保不被将来升级的内核所覆盖，创建/etc/kernel/postinst.d/statoverride文件，内容为：

   ```
   #!/bin/sh
   ```

```
version="$1"
# passing the kernel version is required
[ -z "${version}" ] && exit 0

dpkg-statoverride --update --add root root 0644 /boot/vmlinuz-${version}
```

将此文件设置为可执行文件：

```
# chmod +x /etc/kernel/postinst.d/statoverride
```

4. 编辑/etc/nova/nova.conf文件，添加以下内容至合适的位置：

```
...
[DEFAULT]
...
auth_strategy=keystone
...
[database]
# The SQLAlchemy connection string used to connect to the database
connection = mysql://nova:NOVA_DBPASS@controller/nova
```

5. 配置计算服务使用RabbitMQ进行消息传递，编辑/etc/nova/nova.conf文件，在[DEFAULT]部分中添加以下语句：

```
rpc_backend = nova.rpc.impl_kombu
rabbit_host = controller
rabbit_password = RABBIT_PASS
```

6. 配置计算服务提供远程控制台访问实例。

编辑/etc/nova/nova.conf，并且在[DEFAULT]部分添加以下键值对：

```
[DEFAULT]
...
my_ip=192.168.5.118
vnc_enabled=True
vncserver_listen=0.0.0.0
vncserver_proxyclient_address=192.168.5.118
novncproxy_base_url=http://controller:6080/vnc_auto.html
```

7. 指定运行镜像服务的主机。编辑/etc/nova/nova.conf文件，在[DEFAULT]部分添加以下以下内容：

```
[DEFAULT]
...
glance_host=controller
```

8. 编辑/etc/nova/api-paste.ini文件，添加授权信息至[filter:authtoken]部分：

```
[filter:authtoken]
paste.filter_factory = keystoneclient.middleware.auth_token:filter_factory
auth_host = controller
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = nova
admin_password = NOVA_PASS
```

9. 重启compute服务：

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

10. 删除SQLite数据库：

```
# rm /var/lib/nova/nova.sqlite
```

4.3.3 配置网络

配置网络估计会是你一个困惑的经历，本小节将会介绍一个最简单的网络配置方法。即通过openstack compute遗留网络，使用flat network配置一个基本的DHCP网络。如果想更深入的了解网络服务，查看安装网络服务小节。

1. 只在计算（compute）节点上为计算网络安装以下合适的包，**不要在控制节点安装。**

   为了nova-network服务可以在每个计算节点运送元数据请求，每个计算节点必须安装nova-api-metadata服务，如下：

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

2. 编辑nova.conf文件，定义网络模型：

   编辑/etc/nova/nova.conf文件，为[DEFAULT]部分添加以下语句，注意此处public_interface=eth0，代表将eth0网卡作为虚拟机连接外网的接口。

```
[DEFAULT]
...
network_manager=nova.network.manager.FlatDHCPManager
firewall_driver=nova.virt.libvirt.firewall.IptablesFirewallDriver
network_size=254
allow_same_net_traffic=False
multi_host=True
send_arp_for_ha=True
share_dhcp_address=True
force_dhcp_release=True
flat_network_bridge=br100
flat_interface=eth1
public_interface=eth0
```

3. 重启网络服务：

```
# service nova-network restart
```

为虚拟机创建一个网络，此命令旨在安装是创建一遍，并且不要在任何计算节点运行。在控制节点运行以下命令：

```
# source keystonerc

# nova network-create vmnet --fixed-range-v4=10.0.0.0/24 \
    --bridge-interface=br100 --multi-host=T
```

4.3.4 验证计算服务安装，运行实例

配置完计算服务后，你就可以启动实例（即虚拟机）了。本节将告诉你如何通过上节下载的镜像启动一个低资源的实例。

注意：此流程假定你已经成功安装之前的所有服务。

1. 生成一个密钥对，openstack只有使用私钥，公钥才能运行实例。这些密钥将注入至虚拟机中，从而实现无密码SSH登录虚拟机。

```
$ ssh-keygen
$ cd .ssh
$ nova keypair-add --pub_key id_rsa.pub mykey
```

查看可以使用的密钥对：

```
$ nova keypair-list
```

2. 启动实例需要指定实例的ID与大小。使用以下命令查看实例大小套餐：

```
$ nova flavor-list
```

3. 获取将要使用的镜像ID：

```
$ nova image-list
```

4. 要使用SSH与ping，你必须配置安全组规则：

```
# nova secgroup-add-rule default tcp 22 22 0.0.0.0/0
# nova secgroup-add-rule default icmp -1 -1 0.0.0.0/0
```

5. 运行一个实例的语法规则：

```
$ nova boot --flavor flavorType --key_name keypairName --image ID
newInstanceName
```

例如运行一个最小套餐的实例：

```
$ nova boot --flavor 1 --key_name mykey –image
9e5c2bee-0373-414c-b4afb91b0246ad3b --security_group default cirrOS
```

6. 启动实例后，查看实例状态，状态由build转换成active。

```
$ nova list
```

7. 注意：此处如果实例状态为ERROR，请查看日志文件，修改配置的错误，重新启动服务并创建实例，直至实例状态为active。否则表示安装不成功。

4.4 添加 dashboard

Dashboard是基于web对openstack的资源，用户等操作的管理平台。添加方法如下：

1. 使用root用户在可以连接身份认证服务的节点上安装dashboard:

```
# apt-get install memcached libapache2-mod-wsgi openstack-dashboard
```

Ubuntu 用户需要注意：删除 openstack-dashboard-ubuntu-theme 包。这个主题会阻止一些控件与主题的正确显示：

```
# apt-get remove --purge openstack-dashboard-ubuntu-theme
```

2. 修改/etc/openstack-dashboard/local_settings.py 文件中 CACHES 项的['default']与['LOCATION']的值，确保这些值与/etc/memcached.conf 文件中设置的一样。
打开/etc/openstack-dashboard/local_settings.py 文件，找到以下选项：

```
CACHES = {
'default': {
'BACKEND' : 'django.core.cache.backends.memcached.MemcachedCache',
'LOCATION' : '127.0.0.1:11211'
}
```

3. 更新允许访问dashboard的主机，编辑/etc/openstack-dashboard/local_settings.py修改ALLOWED_HOSTS选项：

```
ALLOWED_HOSTS = ['localhost', 'my-desktop']
```

注意：此处设置为仅本机可访问，一般测试需要任何主机都能访问，此时ALLOWED_HOSTS不设置或者设置为：'*'

4. 本文档假定你是在控制节点运行dashboard的，你也可以按需要运行在其他节点。
编辑/etc/openstack-dashboard/local_setting.py并修改openstack_host的值为身份认证服务所在的节点：

```
OPENSTACK_HOST = "controller"
```

5. 启动Apache web服务与memcached:

```
# service apache2 restart
# service memcached restart
```

6. 现在你可以使用http://controller/horizon登录管理平台了，**windows下需添加controller节点IP至hosts文件。**

4.5 添加块存储服务

4.5.1 配置块存储服务的控制节点

本小节介绍在控制节点上需配置的对象存储服务，并且对象存储节点通过cinder-volume服务向控制节点提供存储服务。本文档使用lvm存储系统。

1. 为块存储服务安装合适的包：

```
# apt-get install cinder-api cinder-scheduler
```

2. 配置块存储使用的数据库为MySQL数据库，编辑/etc/cinder/cinder.conf文件，将以下键值对添加至[database]部分，如果[database]部分不存在，自行添加：

```
[database]
...
# The SQLAlchemy connection string used to connect to the
# database (string value)
sql_connection = mysql://cinder:CINDER_DBPASS@localhost/cinder
...
```

3. 使用root登录数据库，创建cinder用户：

```
# mysql -u root -p
mysql> CREATE DATABASE cinder;
mysql> GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' \
IDENTIFIED BY 'CINDER_DBPASS';
mysql> GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'%' \
IDENTIFIED BY 'CINDER_DBPASS';
```

4. 为块存储创建数据库表：

```
# cinder-manage db sync
```

5. 创建cinder用户，使用身份认证服务授权：

```
# keystone user-create --name=cinder --pass=CINDER_PASS
--email=cinder@example.com
# keystone user-role-add --user=cinder --tenant=service --role=admin
```

6. 添加授权信息至/etc/cinder/api-paste.ini文件，定位[filter:authtoken]部分，添加以下

选项：

```
[filter:authtoken]
paste.filter_factory=keystoneclient.middleware.auth_token:filter_factory
auth_host=controller
auth_port = 35357
auth_protocol = http
auth_uri = http://controller:5000
admin_tenant_name=service
admin_user=cinder
admin_password=CINDER_PASS
```

7. 配置RabbitMQ，编辑/etc/cinder/cinder.conf文件，修改[default]部分：

```
[DEFAULT]
...
rpc_backend = cinder.openstack.common.rpc.impl_kombu
rabbit_host = controller
rabbit_port = 5672
rabbit_userid = guest
rabbit_password = RABBIT_PASS
```

8. 注册块存储服务：

```
# keystone service-create --name=cinder --type=volume \
--description="Cinder Volume Service"
```

创建端口，注意修改ID：

```
# keystone endpoint-create \
--service-id=the_service_id_above \
--publicurl=http://controller:8776/v1/%\(tenant_id\)s \
--internalurl=http://controller:8776/v1/%\(tenant_id\)s \
--adminurl=http://controller:8776/v1/%\(tenant_id\)s
```

9. 同样注册块存储第二个版本的服务：

```
# keystone service-create --name=cinderv2 --type=volumev2 \
--description="Cinder Volume Service V2"
```

创建端口，注意修改ID：

```
# keystone endpoint-create \
--service-id=the_service_id_above \
```

```
--publicurl=http://controller:8776/v2/%\(tenant_id\)s \
--internalurl=http://controller:8776/v2/%\(tenant_id\)s \
--adminurl=http://controller:8776/v2/%\(tenant_id\)s
```

10. 重启cinder服务：

```
# service cinder-scheduler restart
# service cinder-api restart
```

4.6 添加对象存储服务

4.7 安装网络服务

4.8 添加 Orchestration 服务

4.9 添加计费模块

第五章：遇到的安装问题与解决办法