OpenSTACK安装文档

ArryBoom 修订日期:2017-12-19 Ver:2.3

https://github.com/Arryboom/OpenstackDeploymentDoc

## 目录

[目录 1](#_Toc505702013)

[一、 安装前的准备工作 2](#_Toc505702014)

[二、 外部依赖组件安装配置 4](#_Toc505702015)

[1.Mariadb安装 4](#_Toc505702016)

[2.安装Rabbitmq 4](#_Toc505702017)

[3.安装Memcache 5](#_Toc505702018)

[4.安装Etcd 5](#_Toc505702019)

[三、 核心组件安装配置 6](#_Toc505702020)

[1.安装openstack主程序 6](#_Toc505702021)

[2.配置Keystone数据库 6](#_Toc505702022)

[3.安装Keystone 6](#_Toc505702023)

[4.配置Keystone的httpd设置 7](#_Toc505702024)

[5.整合Keystone与openstack 7](#_Toc505702025)

[6.安装Glance镜像服务 8](#_Toc505702026)

[7.安装配置Nova控制节点 10](#_Toc505702027)

[8.控制节点同时部署nova计算节点的配置 16](#_Toc505702028)

[9.安装Neutron网络服务 17](#_Toc505702029)

[10.安装Horizon 25](#_Toc505702030)

[11.最终设置 26](#_Toc505702031)

[四、 按需配置额外组件和多节点部署 27](#_Toc505702032)

[<cinder> 27](#_Toc505702033)

[1. 控制节点安装cinder 27](#_Toc505702034)

[2. 计算节点安装cinder（存储节点） 28](#_Toc505702035)

[3. 安装备份服务（非必要） 31](#_Toc505702036)

[<计算节点部署> 31](#_Toc505702037)

[附录 37](#_Toc505702038)

[故障处理 48](#_Toc505702039)

## 安装前的准备工作

注：以下安装步骤以Openstack最新正式release版本Openstack Pike版本在centos7x64最新版本上的安装为基础进行编写，所有步骤均经过实际操作过程多次验证。暂时不要升级内核，经检查部分服务与el4.x内核不兼容存在假死问题。

物理服务器安装最小化Centos7 x64 后：

1. 更换阿里源（为后续下载速度考虑，非必要）

mv /etc/yum.repos.d/CentOS-Base.repo{,.bak}

切换到/etc/yum.repos.d/目录下

curl -o Centos-Base.repo http://mirrors.aliyun.com/repo/Centos-7.repo

手动新建openstack-pike源Pike.repo:

[Aliyun-pike]

name=Aliyun-pike

baseurl=https://mirrors.aliyun.com/centos/$releasever/cloud/$basearch/openstack-pike/

gpgcheck=0

enabled=1

cost=88

[Aliyun-qemu-ev]

name=Aliyun-CentOS-$releasever - QEMU EV

baseurl=https://mirrors.aliyun.com/centos/$releasever/virt/$basearch/kvm-common/

https://mirrors.aliyun.com/centos

gpgcheck=0

enabled=1

1. yum upgrade,yum update更新所有系统组件至最新版。
2. 安装NTP服务同步时间，为以后扩容节点做准备。

yum -y install wget ntp net-tools tree openssh

1. 配置双网卡（非必要，但考虑安全和性能需要，建议使用双网卡，一块专用于内网与其他计算/存储节点通信，另一块专用为虚拟机连接公网的网桥。）
2. 关闭selinux、防火墙

systemctl stop firewalld.service

systemctl disable firewalld.service

sed -i '/^SELINUX=.\*/c SELINUX=disabled' /etc/selinux/config

sed -i 's/^SELINUXTYPE=.\*/SELINUXTYPE=disabled/g' /etc/selinux/config

grep --color=auto '^SELINUX' /etc/selinux/config

setenforce 0

1. 设置hostname

Host=controller

hostnamectl set-hostname $Host

# hostname $Host

# echo $Host>/etc/hostname

#controller节点需要外网IP（确保使用openstack WEB界面的客户端可以访问到，比如使用其它网段IP地址的客户端需要能够连通controller节点IP）作为VNC代理

#hosts添加 ,添加controller解析关系

echo "10.2.1.20 controller">>/etc/hosts 此处IP地址**需要修改**为本机IP地址

**<非必要>**

设置时区同步时间timedatectl

可以同时配置硬件时间源

## 外部依赖组件安装配置

该部分为控制节点需要进行的基础环境、外部组件的安装与配置。

#### 1.Mariadb安装

1. yum install mariadb mariadb-server python2-PyMySQL

2. 新建/etc/my.cnf.d/openstack.cnf，并添加如下内容：

[mysqld]

bind-address = 0.0.0.0 #为安全考虑可以更改为管理网络的节点地址

default-storage-engine = innodb

innodb\_file\_per\_table = on

max\_connections = 4096

collation-server = utf8\_general\_ci

character-set-server = utf8

3. service mariadb start启动mariadb服务，并使用service mariadb status或mysql客户端连接测试查看服务状态是否正常。

4. systemctl enable mariadb 添加mariadb服务启动项。

5. mysql\_secure\_installation 初始化数据库并设置密码

#### 2.安装Rabbitmq

1. yum install rabbitmq-server

2. service rabbitmq-server start启动rabbitmq服务并使用service rabbitmq-server status检查服务状态是否正常。

3. rabbitmqctl add\_user openstack openstack 为消息队列添加用户。

4. rabbitmqctl set\_permissions openstack ".\*" ".\*" ".\*" 配置消息队列用户远程访问。

5. systemctl enable rabbitmq-server 添加启动项。

注：如果前面配置过主机名后未重启，则此处配置的rabbitmq用户可能会在后续重启后丢失，因为rabbitmq是根据当前hostname作为node节点作为数据名保存，所以建议在此步骤之前先进行一次重启来避免因为主机名进行过修改而没有生效期间手动配置的用户丢失，导致此后出现消息队列无法连接等异常问题。此后使用过程中如果有修改主机名也应注意及时迁移对应的数据文件。

#### 3.安装Memcache

1. yum install memcached python-memcached

2. 编辑/etc/sysconfig/memcached：

修改OPTIONS="-l 127.0.0.1,::1"为OPTIONS="-l 127.0.0.1,::1,controller"

3. service memcached start 并使用service memcached status查看服务状态是否正常

4. systemctl enable memcached 添加开机启动项

**<非必要：性能优化>**

PORT="11211"

USER="memcached"

MAXCONN="1024"

CACHESIZE="64"

OPTIONS="-l 127.0.0.1,::1"

如有需要可以酌情修改maxconn增大连接数来提升性能。

#### 4.安装Etcd

1. yum install etcd

2. 编辑/etc/etcd/etcd.conf文件，设置ETCD\_INITIAL\_CLUSTER, ETCD\_INITIAL\_ADVERTISE\_PEER\_URLS, ETCD\_ADVERTISE\_CLIENT\_URLS, ETCD\_LISTEN\_CLIENT\_URLS为管理网络控制节点的IP地址（如下示例配置为10.0.0.11，需要替换）：

#[Member]

ETCD\_DATA\_DIR="/var/lib/etcd/default.etcd"

ETCD\_LISTEN\_PEER\_URLS="http://10.0.0.11:2380"

ETCD\_LISTEN\_CLIENT\_URLS="http://10.0.0.11:2379"

ETCD\_NAME="controller"

#[Clustering]

ETCD\_INITIAL\_ADVERTISE\_PEER\_URLS="http://10.0.0.11:2380"

ETCD\_ADVERTISE\_CLIENT\_URLS="http://10.0.0.11:2379"

ETCD\_INITIAL\_CLUSTER="controller=http://10.0.0.11:2380"

ETCD\_INITIAL\_CLUSTER\_TOKEN="etcd-cluster-01"

ETCD\_INITIAL\_CLUSTER\_STATE="new"

3. systemctl start etcd 启动etcd服务并使用systemctl status etcd 检查服务状态是否正常

4. systemctl enable etcd 添加服务启动项

## 核心组件安装配置

**注：部分服务配置过程中需要创建openstack用户并手动输入用户密码，下文中如有未明确标明的密码手动配置操作，编写本文档时设置的密码均与用户名相同（生产环境出于安全考虑应自行设置复杂密码，如果openstack user create** xxx**时使用了不同的密码，相应的下文中部分配置文件内的密码参数也应作对应修改，如各服务[keystone\_authtoken]节内password的值应与创建对应openstack用户时所输入密码一致）。**

#### 1.安装openstack主程序

1. yum install centos-release-openstack-pike

2. yum upgrade

3. yum install python-openstackclient

4. yum install openstack-selinux

#### 2.配置Keystone数据库

1. mysql 进入client命令行

2. CREATE DATABASE keystone; 创建keystone数据库

3. GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'localhost' IDENTIFIED BY 'keystone';

GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'%' IDENTIFIED BY 'keystone'; 创建keystone用户并保证可以远程访问.

4. flush privileges; 刷新权限

#### 3.安装Keystone

1. yum install openstack-keystone httpd mod\_wsgi

2. 编辑/etc/keystone/keystone.conf，

在[database]节内添加一行：

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

[token]节内添加一行：

provider = fernet

3. su -s /bin/sh -c "keystone-manage db\_sync" keystone 初始化keystone数据库，正常情况下该步后keystone库内会出现对应的新tables（如没有请检查/var/log/keystone/keystone.log）。

4. 初始化fernet key的repositories:：

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

keystone-manage credential\_setup --keystone-user keystone --keystone-group keystone

5. 初始化识别服务：

keystone-manage bootstrap --bootstrap-password admin --bootstrap-admin-url http://controller:35357/v3/ --bootstrap-internal-url http://controller:5000/v3/ --bootstrap-public-url http://controller:5000/v3/ --bootstrap-region-id RegionOne

#### 4.配置Keystone的httpd设置

1. 编辑/etc/httpd/conf/httpd.conf，添加如下内容：

ServerName controller

2. ln -s /usr/share/keystone/wsgi-keystone.conf /etc/httpd/conf.d/ 衔接keystone配置文件。

3. service httpd start 启动httpd 服务，并通过/var/log/httpd/下日志文件检查状态是否正常。

4. systemctl enable httpd 添加httpd开机启动。

5. cd 回到～目录，并新建授权用admin.sh文件（terminal登录和管理openstack使用，使用时提前source admin.sh），内容如下：

export OS\_USERNAME=admin

export OS\_PASSWORD=admin

export OS\_PROJECT\_NAME=admin

export OS\_USER\_DOMAIN\_NAME=Default

export OS\_PROJECT\_DOMAIN\_NAME=Default

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

export OS\_IDENTITY\_API\_VERSION=3

#### 5.整合Keystone与openstack

1. openstack project create --domain default --description "Service Project" service 创建Openstack自用服务项目。

2. 创建测试项目和用户（**非必要**）：

1）openstack project create --domain default \

--description "Demo Project" demo

2）openstack user create --domain default \

--password-prompt demo

3）openstack role create user

4）openstack role add --project demo --user demo user

3. 基于安全性原则，移除部分配置文件信息：

编辑/etc/keystone/keystone-paste.ini文件并移除[pipeline:public\_api], [pipeline:admin\_api], and [pipeline:api\_v3] 节内的admin\_token\_auth行（如果有）。

测试：

1）管理员获取临时token

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

2）demo获取临时token

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

#需要提前unset OS\_AUTH\_URL OS\_PASSWORD

上述两个测试命令正常情况应该会得到一个类似如下内容的响应：

+------------+-----------------------------------------------------------------+

| Field | Value |

+------------+-----------------------------------------------------------------+

| expires | 2017-11-22T20:14:07.056119Z |

| id | gAAAAABWvi7\_B8kKQD9wdXac8MoZiQldmjEO643d-e\_j-XXq9AmIegIbA7UHGPv |

| | atnN21qtOMjCFWX7BReJEQnVOAj3nclRQgAYRsfSU\_MrsuWb4EDtnjU7HEpoBb4 |

| | o6ozsA\_NmFWEpLeKy0uNn\_WeKbAhYygrsmQGA49dclHVnz-OMVLiyM9ws |

| project\_id | 343d245e850143a096806dfaefa9afdc |

| user\_id | ac3377633149401296f6c0d92d79dc16 |

+------------+-----------------------------------------------------------------+

#### 6.安装Glance镜像服务

1. mysql 进入mariadb命令行界面

2. create database glance; 为glance服务创建数据库

3. 为glance数据库配置用户并设置远程访问：

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

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

4. openstack user create --domain default --password-prompt glance 为glance服务配置openstack用户。

5. openstack role add --project service --user glance admin 配置glance用户为管理员。

6. openstack service create --name glance --description "OpenStack Image" image 创建glance服务。

7. 关联glance API：

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

1. yum install openstack-glance 安装glance
2. 编辑/etc/glance/glance-api.conf：
3. 在[database]节添加如下行：

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

1. 在keystone\_authtoken和paste\_deploy节添加如下内容：

[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 = glance

[paste\_deploy]

# ...

flavor = keystone

1. 在glance\_store节添加如下内容（filesystem\_store\_datadir为镜像路径，根据需要可以配置到外部存储位置，当配置到外部存储位置时，确保glance有足够的权限读写外部存储位置）：

[glance\_store]

# ...

stores = file,http

default\_store = file

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

1. 编辑/etc/glance/glance-registry.conf，并添加如下内容（与上节类似，不过多阐述，具体看节名）：

[database]

# ...

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

[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 = glance

[paste\_deploy]

# ...

flavor = keystone

5) su -s /bin/sh -c "glance-manage db\_sync" glance 初始化glance数据库，正常初始化后glance库内会出现相关tables。

6）service openstack-glance-api start

service openstack-glance-registry.service start

启动glance服务并使用service 服务名status 检查服务状态是否正常。

7）systemctl enable openstack-glance-api.service openstack-glance-registry.service 添加启动项。

#### 7.安装配置Nova控制节点

1. myql 打开mysql命令行窗口配置nova数据库：

CREATE DATABASE nova\_api;

CREATE DATABASE nova;

CREATE DATABASE nova\_cell0;

2. 继续在mysql命令行窗口配置nova用户远程访问和数据库权限:

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

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

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

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

GRANT ALL PRIVILEGES ON nova\_cell0.\* TO 'nova'@'localhost' IDENTIFIED BY 'nova';

GRANT ALL PRIVILEGES ON nova\_cell0.\* TO 'nova'@'%' IDENTIFIED BY 'nova';

3. 创建nova的openstack帐户：

openstack user create --domain default --password-prompt nova

4. openstack role add --project service --user nova admin 添加nova用户到openstack管理员。

5. 创建nova服务：

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

1. 配置nova服务接口：

openstack endpoint create --region RegionOne compute public http://controller:8774/v2.1

openstack endpoint create --region RegionOne compute internal http://controller:8774/v2.1

openstack endpoint create --region RegionOne compute admin http://controller:8774/v2.1

1. 创建placement用户并添加到管理员：

openstack user create --domain default --password-prompt placement

openstack role add --project service --user placement admin

1. 配置placement服务和对应接口：

openstack service create --name placement --description "Placement API" placement

openstack endpoint create --region RegionOne placement public http://controller:8778

openstack endpoint create --region RegionOne placement internal http://controller:8778

openstack endpoint create --region RegionOne placement admin http://controller:8778

1. yum install openstack-nova-api openstack-nova-conductor openstack-nova-console openstack-nova-novncproxy openstack-nova-scheduler openstack-nova-placement-api 安装nova软件包
2. 编辑/etc/nova/nova.conf并添加如下内容（请根据节名自行添加,其中my\_ip需要修改为控制节点IP）：

[DEFAULT]

# ...

enabled\_apis = osapi\_compute,metadata

transport\_url = rabbit://openstack:openstack@controller

my\_ip = 10.0.0.11 **#需要修改为本节点IP**

use\_neutron = True

firewall\_driver = nova.virt.firewall.NoopFirewallDriver

[api\_database]

# ...

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

[database]

# ...

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

[api]

# ...

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 = nova

[vnc]

# ...

enabled = true

vncserver\_listen = $my\_ip

vncserver\_proxyclient\_address = $my\_ip

[glance]

# ...

api\_servers = http://controller:9292

[oslo\_concurrency]

# ...

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

[placement]

# ...

os\_region\_name = RegionOne

project\_domain\_name = Default

project\_name = service

auth\_type = password

user\_domain\_name = Default

auth\_url = http://controller:35357/v3

username = placement

password = placement

1. **<故障排除用>**官方文档中特别说明如果配置完成后或配置placement服务过程中发现错误（目前测试环境安装pike版本发现确实存在此问题），应检查/var/log/nova/nova-compute.log，如果在其中发现类似如下报错：

2017-03-07 15:48:34.956 21930 ERROR nova.scheduler.client.report [req-c2cd8993-1c3e-4443-a262-4fa2de789959 - - - - -] Failed to create resource provider record in placement API for UUID cbbf44b0-fad2-4ad3-a9c5-611890b5ab4c. Got 403: <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">

<html><head>

<title>403 Forbidden</title>

</head><body>

<h1>Forbidden</h1>

<p>You don't have permission to access /resource\_providers

on this server.</p>

</body></html>

则应该修改如下文件

/etc/httpd/conf.d/00-nova-placement-api.conf 添加如下配置内容：

<Directory /usr/bin>

<IfVersion >= 2.4>

Require all granted

</IfVersion>

<IfVersion < 2.4>

Order allow,deny

Allow from all

</IfVersion>

</Directory>

随后 service httpd restart重新启动httpd服务。

1. su -s /bin/sh -c "nova-manage api\_db sync" nova 初始化nova-API数据库
2. su -s /bin/sh -c "nova-manage cell\_v2 map\_cell0" nova 注册cell0数据库
3. su -s /bin/sh -c "nova-manage cell\_v2 create\_cell --name=cell1 --verbose" nova 创建cell1，正常情况下回显一个UUID值。
4. su -s /bin/sh -c "nova-manage db sync" nova 初始化nova管理数据库，此时检查nova\_cell0内已有对应tables。
5. 配置检查确认nova-manage cell\_v2 list\_cells 此时应该得到类似如下回显：

+-------+--------------------------------------+--------------------------------

----+-------------------------------------------------+

| Name | UUID | Transport URL

| Database Connection |

+-------+--------------------------------------+--------------------------------

----+-------------------------------------------------+

| cell0 | 00000000-0000-0000-0000-000000000000 | none:/

| mysql+pymysql://nova:\*\*\*\*@controller/nova\_cell0 |

| cell1 | 3a132683-8392-4f8d-a982-32b75683858a | rabbit://openstack:\*\*\*\*@control

ler | mysql+pymysql://nova:\*\*\*\*@controller/nova |

+-------+--------------------------------------+--------------------------------

----+-------------------------------------------------+

1. systemctl start openstack-nova-api.service openstack-nova-consoleauth.service openstack-nova-scheduler.service openstack-nova-conductor.service openstack-nova-novncproxy.service 启动nova相关服务
2. systemctl status openstack-nova-api.service openstack-nova-consoleauth.service openstack-nova-scheduler.service openstack-nova-conductor.service openstack-nova-novncproxy.service 查看nova各服务状态是否正常

此时如果发现部分服务未启动或为load而非running状态，则应检查/var/log/nova/内日志，如果发现AMQP类错误，请使用rabbitmqctl list\_users查看消息队列用户openstack是否存在，如不存在请参考上方配置节重新添加openstack用户到消息队列服务并重启nova各服务。

1. systemctl enable openstack-nova-api.service openstack-nova-consoleauth.service openstack-nova-scheduler.service openstack-nova-conductor.service openstack-nova-novncproxy.service 上方配置确认正常后，添加nova服务启动项。**<故障排除用>**个别情况安装后无法连接VNC的，检查openstack-nova-novncproxy服务。

**<状态检查-非必要>**

1. 找到此前建立的授权文件admin.sh
2. source admin.sh
3. openstack compute service list 此时应得到类似如下回显：

$ 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 |

+----+--------------------+------------+----------+---------+-------+----------------------------+

1. openstack catalog list 此时应得到类似如下回显：

$ openstack catalog list

+-----------+-----------+-----------------------------------------+

| Name | Type | Endpoints |

+-----------+-----------+-----------------------------------------+

| keystone | identity | RegionOne |

| | | public: http://controller:5000/v3/ |

| | | RegionOne |

| | | internal: http://controller:5000/v3/ |

| | | RegionOne |

| | | admin: http://controller:35357/v3/ |

| | | |

| glance | image | RegionOne |

| | | admin: http://controller:9292 |

| | | RegionOne |

| | | public: http://controller:9292 |

| | | RegionOne |

| | | internal: http://controller:9292 |

| | | |

| nova | compute | RegionOne |

| | | admin: http://controller:8774/v2.1 |

| | | RegionOne |

| | | internal: http://controller:8774/v2.1 |

| | | RegionOne |

| | | public: http://controller:8774/v2.1 |

| | | |

| placement | placement | RegionOne |

| | | public: http://controller:8778 |

| | | RegionOne |

| | | admin: http://controller:8778 |

| | | RegionOne |

| | | internal: http://controller:8778 |

| | | |

+-----------+-----------+-----------------------------------------+

1. openstack image list 此时应得到类似如下回显：

$ openstack image list

+--------------------------------------+-------------+-------------+

| ID | Name | Status |

+--------------------------------------+-------------+-------------+

| 9a76d9f9-9620-4f2e-8c69-6c5691fae163 | cirros | active |

+--------------------------------------+-------------+-------------+

1. nova-status upgrade check 此时应得到类似如下回显：

# nova-status upgrade check

+---------------------------+

| Upgrade Check Results |

+---------------------------+

| Check: Cells v2 |

| Result: Success |

| Details: None |

+---------------------------+

| Check: Placement API |

| Result: Success |

| Details: None |

+---------------------------+

| Check: Resource Providers |

| Result: Success |

| Details: None |

+---------------------------+

**</状态检查>**

#### 8.控制节点同时部署nova计算节点的配置

按照上方说明配置完毕nova控制节点后，如果需要在将该控制节点上同时当作计算节点来运行虚拟机（单机部署），则需要进行如下配置。

1. yum install openstack-nova-compute 安装计算节点nova包。
2. 编辑/etc/nova/nova.conf，在vnc节添加如内容:

[vnc]

# ...

novncproxy\_base\_url = http://$my\_ip:6080/vnc\_auto.html

1. egrep -c '(vmx|svm)' /proc/cpuinfo 执行后检查输入内容来判断该服务器支持的硬件加速类型。
2. 如果该命令返回>=1，则无需额外配置。
3. 如果该命令回显为0，则需要配置/etc/nova/nova.conf文件，在[libvirt]节添加如下内容：

[libvirt]

# ...

virt\_type = qemu

1. systemctl start libvirtd.service openstack-nova-compute.service 启动libvirt和novacompute服务，并使用systemctl status libvirtd.service openstack-nova-compute.service查看服务启动后的状态是否正常。**<故障排查>**此时检查/var/log/nova/nova-compute.log可能会发现上述[安装配置Nova控制节点]内步骤11的错误日志，如果发现类似错误日志请按照该处的方法进行处理并重启这两个服务后再次检查日志。
2. systemctl enable libvirtd.service openstack-nova-compute.service 添加两个服务的开机启动项。
3. openstack compute service list --service nova-compute 查看计算节点是否已经被控制节点发现，正常单节点部署时回显如下：

+----+--------------+------------+------+---------+-------+---------------------

-------+

| ID | Binary | Host | Zone | Status | State | Updated At

|

+----+--------------+------------+------+---------+-------+---------------------

-------+

| 8 | nova-compute | controller | nova | enabled | up | 2017-11-23T04:24:48.

000000 |

+----+--------------+------------+------+---------+-------+---------------------

-------+

1. **非必要<故障排除用>** su -s /bin/sh -c "nova-manage cell\_v2 discover\_hosts --verbose" nova 扫描新计算节点用 nova-manage cell\_v2 discover\_hosts

注：如果安装完成后，创建新虚拟机发现Host 'controller' is not mapped to any cell错误，则应该使用该条命令重新扫描可用节点。

1. **非必要** 配置控制节点自动周期扫描新节点：

编辑/etc/nova/nova.conf，添加如下内容：

[scheduler]

discover\_hosts\_in\_cells\_interval = 300

#### 9.安装Neutron网络服务

**<控制节点>**

1. mysql 进入mysql客户端

2. create database neutron; 创建neutron数据库

3. 配置neutron用户远程访问和相应的数据库权限：

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

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

4. openstack user create --domain default --password-prompt neutron 创建neutron的openstack账号。

5. openstack role add --project service --user neutron admin 配置neutron用户为管理员。

6. 配置neutron服务和对应API接口：

openstack service create --name neutron --description "OpenStack Networking" network

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

1. **<需选择>**此时，网络服务有两种可选，将在下方进行说明，应根据实际需要选择一种来继续安装（只能选择一种）。

**第一种**：部署一个非常简单的网络架构，仅支持虚拟机通过本服务直接连接到外网，不支持私有网络，路由，或者浮动IP地址，防火墙，负载均衡等。只有管理员或其他特别设置的权限用户可以管理网络设置。（跳转至<简单网络架构安装>进行部署）

**第二种**：对第一种简单网络架构的改进增强版，支持layer3层（OSI七层模型中的网络层）网络服务（如路由配置），使用该架构的虚拟机实例可以连接到私有网络，如前面建立的demo帐户或者其他普通权限的用户均可以自行管理自己的私有网络，包括外部网络与私有网络之间的路由。同时，可以通过使用浮动IP来让虚拟机可以使用私有网络来连接到外网。

（跳转至<进阶网络架构安装>进行部署）

**<简单网络架构安装>**

1） yum install openstack-neutron openstack-neutron-ml2 openstack-neutron-linuxbridge ebtables 安装必要neutron包

2） 编辑/etc/neutron/neutron.conf，添加如下内容（根据下文节名查找添加）：

[database]

# ...

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

[DEFAULT]

# ...

core\_plugin = ml2

service\_plugins =

transport\_url = rabbit://openstack:openstack@controller

auth\_strategy = keystone

notify\_nova\_on\_port\_status\_changes = true

notify\_nova\_on\_port\_data\_changes = true

[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 = neutron

[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 = nova

[oslo\_concurrency]

# ...

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

1. 编辑/etc/neutron/plugins/ml2/ml2\_conf.ini并添加如下内容（根据节名自行寻找）：

[ml2]

# ...

type\_drivers = flat,vlan

tenant\_network\_types =

mechanism\_drivers = linuxbridge

extension\_drivers = port\_security

[ml2\_type\_flat]

# ...

flat\_networks = provider

[securitygroup]

# ...

enable\_ipset = true

1. 编辑/etc/neutron/plugins/ml2/linuxbridge\_agent.ini并添加如下内容：

注：外网物理网卡名**需要修改**。

如：服务器实际具有两块网卡，第一块eth0连接内网专用于openstack节点通信通信使用，第二块网卡eth1连接外部互联网，则此处应使用第二块网卡eth1。

[linux\_bridge]

physical\_interface\_mappings = provider:**外网物理网卡名**

[vxlan]

enable\_vxlan = false

[securitygroup]

# ...

enable\_security\_group = true

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

1. 配置/etc/neutron/dhcp\_agent.ini:

[DEFAULT]

# ...

interface\_driver = linuxbridge

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

enable\_isolated\_metadata = true

**<进阶网络架构安装>**

1. yum install openstack-neutron openstack-neutron-ml2 openstack-neutron-linuxbridge ebtables 安装neutron组件
2. 编辑/etc/neutron/neutron.conf，添加如下内容（根据节名填充至对应节）：

[DEFAULT]

# ...

core\_plugin = ml2

service\_plugins = router

allow\_overlapping\_ips = true

transport\_url = rabbit://openstack:openstack@controller

auth\_strategy = keystone

notify\_nova\_on\_port\_status\_changes = true

notify\_nova\_on\_port\_data\_changes = true

[database]

# ...

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

[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 = neutron

[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 = nova

[oslo\_concurrency]

# ...

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

1. 编辑/etc/neutron/plugins/ml2/ml2\_conf.ini文件，加入如下内容：

[ml2]

# ...

type\_drivers = flat,vlan,vxlan

tenant\_network\_types = vxlan

mechanism\_drivers = linuxbridge,l2population

extension\_drivers = port\_security

[ml2\_type\_flat]

# ...

flat\_networks = provider

[ml2\_type\_vxlan]

# ...

vni\_ranges = 1:1000

[securitygroup]

# ...

enable\_ipset = true

1. 编辑/etc/neutron/plugins/ml2/linuxbridge\_agent.ini 添加如下内容：

注：替换物理网卡名为提供外网连接的物理网卡，OVAERLAY\_IP地址为处理overlay流量网卡的静态IP。

如：#physical\_interface\_mappings = provider:eth1

#local\_ip = 10.8.8.8

[linux\_bridge]

physical\_interface\_mappings = provider:**物理网卡名**

[vxlan]

enable\_vxlan = true

local\_ip = **OVERLAY\_IP地址**

l2\_population = true

[securitygroup]

# ...

enable\_security\_group = true

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

1. 编辑/etc/neutron/l3\_agent.ini 添加如下内容：

[DEFAULT]

# ...

interface\_driver = linuxbridge

1. 编辑/etc/neutron/dhcp\_agent.ini 添加如下内容：

[DEFAULT]

# ...

interface\_driver = linuxbridge

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

enable\_isolated\_metadata = true

**<网络架构选择配置完毕>**

1. <上方网络架构选择一种安装完成后>从此处开始继续配置metadata agent。
2. 编辑/etc/neutron/metadata\_agent.ini，添加如下内容：

[DEFAULT]

# ...

nova\_metadata\_host = controller

metadata\_proxy\_shared\_secret = metadata

1. 编辑/etc/nova/nova.conf使neutron与nova计算服务整合：

[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 = neutron

service\_metadata\_proxy = true

metadata\_proxy\_shared\_secret = metadata

1. 为neutron服务插件配置软衔接：

ln -s /etc/neutron/plugins/ml2/ml2\_conf.ini /etc/neutron/plugin.ini

1. 初始化neutron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

此时检查neutron库内应该已经出现了相应的tables。

1. 重启nova-api服务：

systemctl restart openstack-nova-api.service

1. systemctl start neutron-server.service neutron-linuxbridge-agent.service neutron-dhcp-agent.service neutron-metadata-agent.service 启动neutron服务，并使用

systemctl status neutron-server.service neutron-linuxbridge-agent.service neutron-dhcp-agent.service neutron-metadata-agent.service 查看服务状态

1. systemctl enable neutron-server.service neutron-linuxbridge-agent.service neutron-dhcp-agent.service neutron-metadata-agent.service 添加开机启动项

对于选择**进阶网络架构**的用户，还需要：

systemctl enable neutron-l3-agent.service

systemctl start neutron-l3-agent.service

1. 重启各相关服务：

systemctl restart openstack-nova-compute.service

#systemctl enable neutron-linuxbridge-agent.service

#systemctl start neutron-linuxbridge-agent.service

**<计算节点>**

待补充

#### 10.安装Horizon

1. yum install openstack-dashboard 安装horizon包

2. 编辑/etc/openstack-dashboard/local\_settings，注意此处**不是添加**，而是修改相应参数的值：

**1)**

OPENSTACK\_HOST = "controller"

**2)**

ALLOWED\_HOSTS = ['\*']

#注意此处ALLOWED\_HOSTS的值为可以访问horizon的白名单地址，生产环境出于#安全考虑建议做对应限制。多个地址用，分割。

**3)**

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

CACHES = {

'default': {

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

'LOCATION': 'controller:11211',

}

}

**4)**

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

**5)**

OPENSTACK\_KEYSTONE\_MULTIDOMAIN\_SUPPORT = True

#如果不需要多域名支持，此处设置为False

**6)**

OPENSTACK\_API\_VERSIONS = {

"identity": 3,

"image": 2,

"volume": 2,

}

**7)**

OPENSTACK\_KEYSTONE\_DEFAULT\_DOMAIN = "Default"

**8)**

OPENSTACK\_KEYSTONE\_DEFAULT\_ROLE = "user"

**9)**如果选择了**简单网络架构**，还需要禁用掉如下选项：

OPENSTACK\_NEUTRON\_NETWORK = {

...

'enable\_router': False,

'enable\_quotas': False,

'enable\_distributed\_router': False,

'enable\_ha\_router': False,

'enable\_lb': False,

'enable\_firewall': False,

'enable\_vpn': False,

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

}

**10)非必要** 设置时区：

注：时区代码请参考<https://en.wikipedia.org/wiki/List_of_tz_database_time_zones>

如：上海=Asia/Shanghai

TIME\_ZONE = "时区代码"

**11)** systemctl restart httpd.service memcached.service 重启httpd和memcache服务，服务重启完成后，访问http://服务器IP/dashboard ，正常此时应该可以看到openstack WEB界面。

**12)<故障排除用>**如果上述11步骤中服务重启完成后，无法访问web界面，等待超时或者提示500服务器内部错误，则应配置/etc/httpd/conf.d/openstack-dashboard.conf文件，

在WSGISocketPrefix run/wsgi下面加入以下内容：

WSGIApplicationGroup %{GLOBAL}

随后，service httpd restart 重启httpd服务，服务重启完成后检查能否正常访问WEB界面。

#### 11.最终设置

1. **<如果前面已经按照准备工作配置则此处无需再次配置>**配置hosts和主机名为controller，并重启：

1）编辑/etc/hosts,加入如下行（本机IP地址替换为你的控制节点网卡地址）：

本机IP地址 controller

2）hostnamectl set-hostname controller

3）reboot

至此，基础组件已经全部安装完毕，可以使用cirros镜像来进行简单的部署测试各功能是否正常。如需进一步测试，请跳转至附录<最小化安装测试>进行。

## 按需配置额外组件和多节点部署

### <cinder>

#### 控制节点安装cinder

注:如果控制节点也需要运行存储服务(即当作存储节点使用)，请在本步骤完成后跳转至计算节点安装cinder继续进行配置。

1. mysql 进入mysql客户端
2. CREATE DATABASE cinder; 创建cinder数据库
3. 为cinder数据库用户分配远程访问权限

GRANT ALL PRIVILEGES ON cinder.\* TO 'cinder'@'localhost' IDENTIFIED BY 'cinder';

GRANT ALL PRIVILEGES ON cinder.\* TO 'cinder'@'%' IDENTIFIED BY 'cinder';

1. openstack user create --domain default --password-prompt cinder 创建cinder在openstack上的用户
2. openstack role add --project service --user cinder admin 为cinder用户分配管理员权限
3. 创建cinder相关服务：

openstack service create --name cinderv2 --description "OpenStack Block Storage" volumev2

openstack service create --name cinderv3 --description "OpenStack Block Storage" volumev3

1. 创建cinder相关API：

openstack endpoint create --region RegionOne volumev2 public http://controller:8776/v2/%\(project\_id\)s

openstack endpoint create --region RegionOne volumev2 internal http://controller:8776/v2/%\(project\_id\)s

openstack endpoint create --region RegionOne volumev2 admin http://controller:8776/v2/%\(project\_id\)s

openstack endpoint create --region RegionOne volumev3 public http://controller:8776/v3/%\(project\_id\)s

openstack endpoint create --region RegionOne volumev3 internal http://controller:8776/v3/%\(project\_id\)s

openstack endpoint create --region RegionOne volumev3 admin http://controller:8776/v3/%\(project\_id\)s

1. yum install openstack-cinder 安装相关组件包
2. 编辑/etc/cinder/cinder.conf，修改如下配置节：

[database]

# ...

connection = mysql+pymysql://cinder:cinder@controller/cinder

[DEFAULT]

# ...

transport\_url = rabbit://openstack:openstack@controller

auth\_strategy = keystone

my\_ip = **控制节点网卡IP**

[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 = cinder

password = cinder

[oslo\_concurrency]

# ...

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

1. su -s /bin/sh -c "cinder-manage db sync" cinder 初始化cinder数据库
2. 编辑/etc/nova/nova.conf并修改如下配置：

[cinder]

os\_region\_name = RegionOne

1. systemctl restart openstack-nova-api.service 重启nova服务
2. 启动相关服务并查看服务状态是否正常

systemctl start openstack-cinder-api.service openstack-cinder-scheduler.service

1. 添加开机启动项

systemctl enable openstack-cinder-api.service openstack-cinder-scheduler.service

#### 计算节点安装cinder（存储节点）

1. yum install lvm2 device-mapper-persistent-data 安装lvm
2. systemctl start lvm2-lvmetad.service 启动lvm服务并使用

systemctl status lvm2-lvmetad.service查看服务状态是否正常

1. systemctl enable lvm2-lvmetad.service 添加开机启动项
2. 创建物理卷（请修改/dev/sdb为对应的真实硬盘或盘上物理分区）：

# pvcreate **/dev/sdb**

正常回显:Physical volume "/dev/sdb" successfully created

1. 创建卷组（请修改/dev/sdb为上述物理卷位置）：

# vgcreate cinder-volumes /dev/sdb

正常回显:Volume group "cinder-volumes" successfully created

1. 编辑/etc/lvm/lvm.conf，进行如下修改：

devices {

...

filter = [ "a/sdb/", "r/.\*/"]

#在device节配置过滤器为只过滤我们创建了cinder-volumes卷组的设备（或分区）

# 每一个过滤器内的成员都已a或r开头（a表示接受，r表示拒绝）并包含一个设备名的##正则表达式来指向对应设备，整个过滤器的数组最后必须以r/.\*/结尾来排除剩余的其他##设备，你可以使用vgs –vvvv来测试写好的过滤器是否正常工作

#Each item in the filter array begins with a for accept or r for reject and includes a regular #expression for the device name. The array must end with r/.\*/ to reject any remaining devices. #You can use the vgs -vvvv command to test filters.

**<待更新，注意事项>**

Warning

If your storage nodes use LVM on the operating system disk, you must also add the associated device to the filter. For example, if the /dev/sda device contains the operating system:

filter = [ "a/sda/", "a/sdb/", "r/.\*/"]

Similarly, if your compute nodes use LVM on the operating system disk, you must also modify the filter in the /etc/lvm/lvm.conf file on those nodes to include only the operating system disk. For example, if the /dev/sda device contains the operating system:

filter = [ "a/sda/", "r/.\*/"]

1. yum install openstack-cinder targetcli python-keystone 安装相应软件包
2. 编辑/etc/cinder/cinder.conf并进行如下修改：

[database]

# ...

connection = mysql+pymysql://cinder:cinder@controller/cinder

[DEFAULT]

# ...

transport\_url = rabbit://openstack:openstack@controller

auth\_strategy = keystone

my\_ip = **与控制节点通信的网卡IP**

enabled\_backends = lvm #此处可以有多个后端，用，分#割，确保下方有对应此处后端名的节

glance\_api\_servers = <http://controller:9292>

[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 = cinder

password = cinder

[lvm]#如果有多个不同分区

volume\_driver = cinder.volume.drivers.lvm.LVMVolumeDriver

volume\_group = cinder-volumes

iscsi\_protocol = iscsi

iscsi\_helper = lioadm

[oslo\_concurrency]

# ...

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

1. systemctl start openstack-cinder-volume.service target.service 启动服务并使用

systemctl status openstack-cinder-volume.service target.service查看服务状态是否正常

1. systemctl enable openstack-cinder-volume.service target.service 添加开机启动项

#### 安装备份服务（非必要）

注:该服务需要提前安装swift和存储节点

1. yum install openstack-cinder 安装对应包
2. 编辑/etc/cinder/cinder.conf，并进行如下配置：

[DEFAULT]

# ...

backup\_driver = cinder.backup.drivers.swift

backup\_swift\_url = **SWIFT\_URL**

#替换SWIFT\_URL为实际的swift API接口，可以使用openstack catalog show object-store

#查看

1. systemctl start openstack-cinder-backup.service 启动服务并检查状态是否正常
2. systemctl enable openstack-cinder-backup.service 添加开机启动项

### <计算节点部署>

**<计算服务nova-compute>**

物理服务器安装最小化Centos7 x64 后：

1. 更换阿里源（为后续下载速度考虑，非必要）

mv /etc/yum.repos.d/CentOS-Base.repo{,.bak}

切换到/etc/yum.repos.d/目录下

curl -o Centos-Base.repo http://mirrors.aliyun.com/repo/Centos-7.repo

手动加入openstack-pike源Pike.repo:

[Aliyun-pike]

name=Aliyun-pike

baseurl=https://mirrors.aliyun.com/centos/$releasever/cloud/$basearch/openstack-pike/

gpgcheck=0

enabled=1

cost=88

[Aliyun-qemu-ev]

name=Aliyun-CentOS-$releasever - QEMU EV

baseurl=https://mirrors.aliyun.com/centos/$releasever/virt/$basearch/kvm-common/

https://mirrors.aliyun.com/centos

gpgcheck=0

enabled=1

1. yum upgrade,yum update更新所有系统组件至最新版。
2. 安装NTP服务同步时间，为以后扩容节点做准备。

具体时间同步步骤参考附录<时间同步>

yum -y install wget ntp net-tools tree openssh

1. 配置双网卡（非必要，但考虑安全和性能需要，建议使用双网卡，一块专用于内网与其他计算/存储节点通信，另一块专用为虚拟机连接公网的网桥。）
2. 关闭selinux、防火墙

systemctl stop firewalld.service

systemctl disable firewalld.service

sed -i '/^SELINUX=.\*/c SELINUX=disabled' /etc/selinux/config

sed -i 's/^SELINUXTYPE=.\*/SELINUXTYPE=disabled/g' /etc/selinux/config

grep --color=auto '^SELINUX' /etc/selinux/config

setenforce 0

1. 设置hostname

Host=compute1 #此处应根据计算节点名自行分配

hostnamectl set-hostname $Host

# hostname $Host

# echo $Host>/etc/hostname

1. hosts添加 ,添加controller解析关系

echo "10.2.1.20 controller">>/etc/hosts 此处IP地址**需要修改**为控制节点IP地址

echo "10.2.1.21 compute1">>/etc/hosts 此处IP地址**需要修改**为本节点IP地址，解析名修改为本节点解析名

1. yum install openstack-nova-compute 安装必要组件包
2. 编辑/etc/nova/nova.conf，在如下节内添加或修改相应参数：

[DEFAULT]

# ...

enabled\_apis = osapi\_compute,metadata

transport\_url = rabbit://openstack:openstack@controller

my\_ip = **与控制节点通信的网卡IP**

use\_neutron = True

firewall\_driver = nova.virt.firewall.NoopFirewallDriver

[api]

# ...

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 = nova

[vnc]

# ...

enabled = True

vncserver\_listen = 0.0.0.0

vncserver\_proxyclient\_address = $my\_ip

novncproxy\_base\_url = http://**控制节点的外网可以访问的IP**:6080/vnc\_auto.html

[glance]

# ...

api\_servers = http://controller:9292

[oslo\_concurrency]

# ...

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

[placement]

# ...

os\_region\_name = RegionOne

project\_domain\_name = Default

project\_name = service

auth\_type = password

user\_domain\_name = Default

auth\_url = http://controller:35357/v3

username = placement

password = placement

1. egrep -c '(vmx|svm)' /proc/cpuinfo 执行后检查输入内容来判断该服务器支持的硬件加速类型。

1)如果该命令返回>=1，则无需额外配置。

2)如果该命令回显为0，则需要配置/etc/nova/nova.conf文件，在[libvirt]节添加如下内容：

[libvirt]

# ...

virt\_type = qemu

1. systemctl start libvirtd.service openstack-nova-compute.service 启动相关服务并使用systemctl status libvirtd.service openstack-nova-compute.service 查看各服务状态是否正常。
2. systemctl enable libvirtd.service openstack-nova-compute.service添加服务启动项。
3. 回到控制节点，执行su -s /bin/sh -c "nova-manage cell\_v2 discover\_hosts --verbose" nova发现新节点。

**<网络服务nova-neutron>**

1. yum install openstack-neutron-linuxbridge ebtables ipset安装必要组件
2. 编辑/etc/neutron/neutron.conf，参照如下节名进行修改：
3. 在[database]节内，注释掉任何connection参数，计算节点不与数据库直接交互。

随后，参考如下节名进行修改：

[DEFAULT]

# ...

transport\_url = rabbit://openstack:openstack@controller

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 = neutron

[oslo\_concurrency]

# ...

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

1. 此时有两种网络架构可以选择，有关架构之间的差别请参考控制节点该部分的说明，网络架构只能选择一种安装，具体分别的安装步骤见如下：

**<简单网络架构>**

1. 编辑/etc/neutron/plugins/ml2/linuxbridge\_agent.ini，修改如下配置：

[linux\_bridge]

physical\_interface\_mappings = provider:公网网卡名

[vxlan]

enable\_vxlan = false

[securitygroup]

# ...

enable\_security\_group = true

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

**<进阶网络架构>**

1. 编辑/etc/neutron/plugins/ml2/linuxbridge\_agent.ini，修改如下配置：

[linux\_bridge]

physical\_interface\_mappings = provider:公网网卡名

[vxlan]

enable\_vxlan = true

local\_ip = **该网卡分配的与控制/网络节点通信的IP地址**

l2\_population = true

[securitygroup]

# ...

enable\_security\_group = true

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

1. <上方选择一种网络架构安装完成后>
2. 编辑/etc/nova/nova.conf，添加修改如下参数：

[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 = neutron

1. systemctl restart openstack-nova-compute.service 重新启动nova计算服务
2. systemctl start neutron-linuxbridge-agent.service 启动网络服务并使用systemctl status neutron-linuxbridge-agent.service查看服务状态是否正常
3. systemctl enable neutron-linuxbridge-agent.service 添加开机启动项
4. 基础配置完成，此时在控制节点可以正常使用该节点。

# 附录

**<最小化安装测试>**

安装完成后，以管理员用户登录dashboard，下载<http://download.cirros-cloud.net/0.3.5/cirros-0.3.5-x86_64-disk.img>

或

http://download.cirros-cloud.net/0.4.0/cirros-0.4.0-x86\_64-disk.img

，点击左侧工具栏->计算->镜像，上传该镜像，镜像格式为QCOW2。

随后点击该栏内网络->新建网络，类型为flat，不创建子网，共享网络，外部网络，管理员状态，保存。

随后创建新实例，运行，进入vnc界面查看主机是否正常。

**<时间同步>**

echo '时间同步'

/usr/sbin/ntpdate time6.aliyun.com

echo "\*/3 \* \* \* \* /usr/sbin/ntpdate time6.aliyun.com &> /dev/null" > /tmp/crontab

crontab /tmp/crontab

**<内核升级>**

1. rpm --import https://www.elrepo.org/RPM-GPG-KEY-elrepo.org
2. rpm -Uvh http://www.elrepo.org/elrepo-release-7.0-3.el7.elrepo.noarch.rpm
3. yum --enablerepo=elrepo-kernel install kernel-ml

手动部分：

# vi /etc/default/grub

修改成 GRUB\_DEFAULT=0

# grub2-mkconfig -o /boot/grub2/grub.cfg //重新编译内核启动文件，以后升级完内核也要执行一次

注：如果上述grub调整启动顺序失效，参考下列方法重新调整:

cat /boot/grub2/grub.cfg |grep menuentry

可以发现有最新版本的内核，此时可以设置最新版本内核为默认启动：

grub2-set-default 'CentOS Linux (4.8.5-1.el7.elrepo.x86\_64) 7 (Core)'

此时，查看默认启动的系统版本grub2-editenv list：

显示为'CentOS Linux (4.8.5-1.el7.elrepo.x86\_64) 7 (Core)'

**<修改网卡名>**

1. 先编辑网卡的配置文件将里面的NAME DEVICE项修改为eth0

vim /etc/sysconfig/network-scripts/ifcfg-eno16777736

1. 重命名该配置文件。

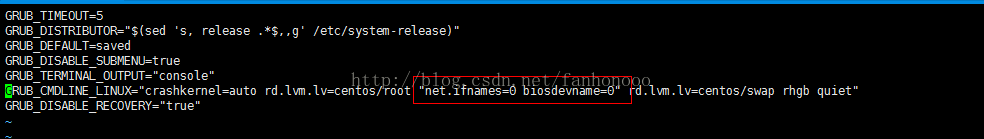
[root@localhost yum.repos.d]# cd /etc/sysconfig/network-scripts/

[root@localhost network-scripts]# mv ifcfg-eno16777736 ifcfg-eth0



1. 禁用该可预测命名规则。对于这一点，你可以在启动时传递“net.ifnames=0 biosdevname=0 ”的内核参数。这是通过编辑/etc/default/grub并加入“net.ifnames=0 biosdevname=0 ”到GRUBCMDLINELINUX变量来实现的。

[root@localhost network-scripts]# vim /etc/default/grub



1. 运行命令grub2-mkconfig -o /boot/grub2/grub.cfg 来重新生成GRUB配置并更新内核参数。

[root@localhost network-scripts]# grub2-mkconfig -o /boot/grub2/grub.cfg

1. reboot

**<禁用NetworkManager>**

**<调整Cinder可用zone>**

注：当多个cinder存储节点，希望在指定的cinder存储节点上面创建卷而不是自动分配时。

<临时方案>

存储节点:

修改Cinder配置文件cinder.conf：

storage\_availability\_zone = BigdataDiskGroup

控制节点：

修改Cinder配置文件cinder.conf：

default\_availability\_zone = TestDiskGroup #此处指定为默认使用的存储zone，

<临时方案>

**<阿里云EPEL源>**

wget -O /etc/yum.repos.d/epel.repo <http://mirrors.aliyun.com/repo/epel-7.repo>

**<eth示例>**

TYPE="Ethernet"

BOOTPROTO="none"

DEFROUTE="yes"

IPV4\_FAILURE\_FATAL="no"

IPV6INIT="yes"

IPV6\_AUTOCONF="yes"

IPV6\_DEFROUTE="yes"

IPV6\_FAILURE\_FATAL="no"

NAME="eth0"

UUID="9b7e0728-7c25-46e7-b4e1-387ffeb5a66c"

DEVICE="eno16777736"

ONBOOT="yes"

IPADDR="192.168.32.31"

PREFIX="24"

GATEWAY="192.169.32.2"

DNS1="8.8.4.4"

IPV6\_PEERDNS="yes"

IPV6\_PEERROUTES="yes"

IPV6\_PRIVACY="no"

NM\_CONTROLLER=no

**<Centos7虚拟网卡>**

https://centos.pkgs.org/7/nux-misc-x86\_64/tunctl-1.5-12.el7.nux.x86\_64.rpm.html

rpm –ivh <http://li.nux.ro/download/nux/misc/el7/x86_64//tunctl-1.5-12.el7.nux.x86_64.rpm>

**<Openswitch模式示例网桥+物理网卡配置>**

**Interface br-ex excerpt:**

TYPE="Ethernet"

BOOTPROTO="none"

DEFROUTE="yes"

IPV4\_FAILURE\_FATAL="no"

IPV6INIT="no"

IPV6\_AUTOCONF="no"

IPV6\_DEFROUTE="no"

IPV6\_FAILURE\_FATAL="no"

NAME="br-ex"

UUID="1d239840-7e15-43d5-a7d8-d1af2740f6ef"

DEVICE="br-ex"

ONBOOT="yes"

IPADDR="192.168.1.41"

PREFIX="24"

GATEWAY="192.168.1.1"

DNS1="127.0.0.1"

DNS2="192.168.1.1"

DNS3="8.8.8.8"

IPV6\_PEERDNS="no"

IPV6\_PEERROUTES="no"

IPV6\_PRIVACY="no"

HWADDR= 33:2d:33:55:66

**Interface eth0:**

TYPE="Ethernet"

BOOTPROTO="none"

DEFROUTE="yes"

IPV4\_FAILURE\_FATAL="no"

IPV6INIT="no"

IPV6\_AUTOCONF="no"

IPV6\_DEFROUTE="no"

IPV6\_FAILURE\_FATAL="no"

NAME="eth0"

DEVICE="eth0"

ONBOOT="yes"

TYPE=”OVSPort”

DEVICETYPE=”ovs”

OVS\_BRIDGE=”br-ex”

正常情况下，horizon创建public网络并创建虚拟路由后，进行测试将得到如下结果：

[root@controller ~]# ip netns

qdhcp-1396b5a8-77e4-410f-b92e-9872613ab910

qdhcp-49ae1aa8-8830-4f56-b02a-b3b008a49719

qrouter-5c001111-faa0-4a2b-b019-ccaf5f582063

qdhcp-ac2e0767-fd30-4ddb-b5f2-ab346ccf8af9

[root@controller ~]# ip netns exec qrouter-5c001111-faa0-4a2b-b019-ccaf5f582063

ping -c 5 www.baidu.com

PING www.a.shifen.com (180.97.33.107) 56(84) bytes of data.

64 bytes from 180.97.33.107 (180.97.33.107): icmp\_seq=1 ttl=49 time=7.79 ms

64 bytes from 180.97.33.107 (180.97.33.107): icmp\_seq=2 ttl=49 time=7.38 ms

64 bytes from 180.97.33.107 (180.97.33.107): icmp\_seq=3 ttl=49 time=7.27 ms

64 bytes from 180.97.33.107 (180.97.33.107): icmp\_seq=4 ttl=49 time=7.23 ms

64 bytes from 180.97.33.107 (180.97.33.107): icmp\_seq=5 ttl=49 time=7.80 ms

--- www.a.shifen.com ping statistics ---

5 packets transmitted, 5 received, 0% packet loss, time 4004ms

rtt min/avg/max/mdev = 7.231/7.498/7.806/0.264 ms

**<RDO一键安装>**

<https://www.rdoproject.org/install/packstack/>

1. If you are using non-English locale make sure your /etc/environment is populated:

LANG=en\_US.utf-8

LC\_ALL=en\_US.utf-8

1. $ sudo systemctl disable firewalld
2. $ sudo systemctl stop firewalld
3. $ sudo systemctl disable NetworkManager
4. $ sudo systemctl stop NetworkManager
5. $ sudo systemctl enable network
6. $ sudo systemctl start network
7. yum install -y <https://rdoproject.org/repos/rdo-release.rpm>
8. $ sudo yum install -y centos-release-openstack-pike
9. $ sudo yum update -y
10. $ sudo yum install -y openstack-packstack
11. $ sudo packstack –allinone

**<添加路由表>**

使用双网卡，同时使用2个网关的时候就需要加一条静态路由了。当然还有很多时候会需要加路由。

一：使用 route 命令添加

使用route 命令添加的路由，机器重启或者网卡重启后路由就失效了，方法：

//添加到主机的路由

# route add –host 192.168.1.11 dev eth0

# route add –host 192.168.1.12 gw 192.168.1.1

//添加到网络的路由

# route add –net 192.168.1.11 netmask 255.255.255.0 eth0

# route add –net 192.168.1.11 netmask 255.255.255.0 gw 192.168.1.1

# route add –net 192.168.1.0/24 eth1

//添加默认网关

# route add default gw 192.168.2.1

//删除路由

# route del –host 192.168.1.11 dev eth0

二：在linux下设置永久路由的方法：

./etc/sysconfig/static-routes :

any net 192.168.3.0/24 gw 192.168.3.254

any net 10.250.228.128 netmask 255.255.255.192 gw 10.250.228.129

使用static-routes的方法是最好的。无论重启系统和service network restart 都会生效

static-routes文件又是什么呢，这个是network脚本执行时调用的一个文件，这个文件的放置在/etc/sysconfig目录下，在network脚本中的位置是：

# Add non interface-specific static-routes.

if [ -f /etc/sysconfig/static-routes ]; then

grep "^any" /etc/sysconfig/static-routes | while read ignore args ; do

/sbin/route add -$args

done

fi

从这段脚本可以看到，这个就是添加静态路由的方法，static-routes的写法是

any net 192.168.0.0/16 gw 网关ip

**<Centos为指定网卡添加路由>**

在/etc/sysconfig/network-scripts/下，如eth0

新建route-eth0文件：

10.88.1.0/24 via 10.88.1.1

**<控制节点使用wget-curl等提示no route to host>**

临时关闭iptables服务，原因暂时不明，生产环境谨慎使用。

**<NTP时间同步>**

内网 NTP 服务器

阿里云为云服务器 ECS 提供了内部 NTP 时间服务器，如下：

ntp1.aliyun.com

ntp2.aliyun.com

ntp3.aliyun.com

ntp4.aliyun.com

ntp5.aliyun.com

ntp6.aliyun.com

ntp1.cloud.aliyuncs.com

ntp2.cloud.aliyuncs.com

ntp3.cloud.aliyuncs.com

ntp4.cloud.aliyuncs.com

ntp5.cloud.aliyuncs.com

ntp6.cloud.aliyuncs.com

ntp7.cloud.aliyuncs.com

ntp8.cloud.aliyuncs.com

ntp9.cloud.aliyuncs.com

ntp10.cloud.aliyuncs.com

ntp11.cloud.aliyuncs.com

ntp12.cloud.aliyuncs.com

公共 NTP 服务器

目前，阿里云以外的设备不能使用以上内网 NTP 服务器。不过，我们同时提供了公共 NTP 服务器，供互联网上的设备使用。主要特性是GPS、北斗授时、原子钟守时的一级时间源多机房、多链路冗余。服务域名为：

Unix 类： time1-7.aliyun.com 此处意为time1.aliyun.com,time2.aliyun.com………

Windows： time.pool.aliyun.com

/usr/sbin/ntpdate time6.aliyun.com

echo "\*/3 \* \* \* \* /usr/sbin/ntpdate time6.aliyun.com &> /dev/null" > /tmp/crontab

crontab /tmp/crontab

<cinder节点存储容量在dashboard上显示错误>

修改计算节点的/etc/nova/nova.conf，进行如下修改：

[libvirt]

images\_type = lvm

images\_volume\_group = cinder-volumes

重启服务service openstack-nova-compute restart

**<keystone内存缓冲设置性能优化>**

注：测试，暂时不要应用于生产环境

编辑/etc/keystone/keystone.conf

[cache]

expiration\_time = 600

backend = dogpile.cache.memcached

enabled = true

memcache\_servers = controller:11211

memcache\_dead\_retry = 300

memcache\_socket\_timeout = 3

memcache\_pool\_connection\_get\_timeout = 10

**<多控制节点VNC>**

nova控制节点，配置memcache缓存，即可解决多节点VNC问题

[cache]

enabled = true

backend = oslo\_cache.memcache\_pool

memcache\_servers = controller1:11211,controller2:11211,controller3:11211

#'>/etc/nova/nova.conf

**<全部安装完成后首次使用需要部署网络>**

**简单网络架构：**

openstack network create --share --external \

--provider-physical-network provider \

--provider-network-type flat provider

2. 创建flat公网地址段

openstack subnet create --network provider \

--allocation-pool start=10.99.1.20,end=10.99.1.253 \

--dns-nameserver 223.5.5.5 --gateway 10.99.1.1 \

--subnet-range 10.99.1.0/24 provider

**进阶网络架构：**

**注：此处部署进阶网络架构需要首先部署上方简单网络架构的内容创建flat网络**

1. openstack network create selfservice
2. 创建虚机内网段，ip段自定义

openstack subnet create --network selfservice \

--dns-nameserver 8.8.4.4 --gateway 172.16.1.1 \

--subnet-range 172.16.1.0/24 selfservice

3. 创建路由

openstack router create router

1. 路由连接到虚机内网段

neutron router-interface-add router selfservice

1. 路由连接到网关

neutron router-gateway-set router provider

<确认检查>

1. $ ip netns

qrouter-89dd2083-a160-4d75-ab3a-14239f01ea0b

qdhcp-7c6f9b37-76b4-463e-98d8-27e5686ed083

qdhcp-0e62efcd-8cee-46c7-b163-d8df05c3c5ad

1. ip netns exec qrouter-89dd2083-a160-4d75-ab3a-14239f01ea0b ping –c 3 [www.baidu.com](http://www.baidu.com)

检查路由是否通外网

1. neutron router-port-list router 列出路由地址，检查各节点主机等能否ping通

neutron router-port-list router

+--------------------------------------+------+-------------------+------------------------------------------+

| id | name | mac\_address | fixed\_ips |

+--------------------------------------+------+-------------------+------------------------------------------+

| bff6605d-824c-41f9-b744-21d128fc86e1 | | fa:16:3e:2f:34:9b | {"subnet\_id": |

| | | | "3482f524-8bff-4871-80d4-5774c2730728", |

| | | | "ip\_address": "172.16.1.1"} |

| d6fe98db-ae01-42b0-a860-37b1661f5950 | | fa:16:3e:e8:c1:41 | {"subnet\_id": |

| | | | "5cc70da8-4ee7-4565-be53-b9c011fca011", |

| | | | "ip\_address": "203.0.113.102"} |

1. ping -c 4 203.0.113.102

PING 203.0.113.102 (203.0.113.102) 56(84) bytes of data.

64 bytes from 203.0.113.102: icmp\_req=1 ttl=64 time=0.619 ms

64 bytes from 203.0.113.102: icmp\_req=2 ttl=64 time=0.189 ms

64 bytes from 203.0.113.102: icmp\_req=3 ttl=64 time=0.165 ms

64 bytes from 203.0.113.102: icmp\_req=4 ttl=64 time=0.216 ms

--- 203.0.113.102 ping statistics ---

rtt min/avg/max/mdev = 0.165/0.297/0.619/0.187 ms

**<节点使用cinder后dashboard磁盘容量显示错误>**

配置该节点的/etc/nova/nova.conf

[libvirt]

images\_type = lvm

images\_volume\_group = cinder-volumes

重启相关服务sudo service nova-compute restart

**<开启cinder镜像缓存>测试**

/etc/cinder/cinder/conf

image\_volume\_cache\_enabled = enable

**<大系统镜像创建超时>**

在对应计算节点上：

/etc/nova/nova.conf

block\_device\_allocate\_retries=1800

#注：1800秒=30分钟

<测试>

/etc/nova/nova.conf：

block\_device\_allocate\_retries\_interval=60

文档说明此处为块设备创建失败时的重试间隔时间，单位为秒

</测试>

<测试>该设置官方未提及，部分技术blog提及

[DEFAULT]节添加block\_device\_creation\_timeout=1800 单位为秒

</测试>

如果镜像下载等导致服务间的交互时间也非常长，为避免token失效，你还需要修改控制节点的keystone的token超时：

# The amount of time that a token should remain valid (in seconds). Drastically

# reducing this value may break "long-running" operations that involve multiple

# services to coordinate together, and will force users to authenticate with

# keystone more frequently. Drastically increasing this value will increase

# load on the `[token] driver`, as more tokens will be simultaneously valid.

# Keystone tokens are also bearer tokens, so a shorter duration will also

# reduce the potential security impact of a compromised token. (integer value)

# Minimum value: 0

# Maximum value: 9223372036854775807

#expiration = 3600

注：可能需要配合上方<keystone内存缓冲设置性能优化>

**<dashboard开启密码接收功能>**

just edit openstack\_dashboard/local/local\_settings.py and add:

OPENSTACK\_ENABLE\_PASSWORD\_RETRIEVE = True

手动接收：

nova get-password <instance> [<ssh\_private\_key\_path>]

**<内核连接数文件数优化>**

报toomany open files错误，尝试：

Params that configure max open connections.

at /etc/sysctl.conf

add:

net.core.somaxconn=50000

fs.file-max=131072

and then:

sudo sysctl -p

at /usr/include/linux/limits.h

change:

NR\_OPEN = 65536

at /etc/security/limits.conf

add:

\* soft nofile 65535

\* hard nofile 65535

**<隐藏界面登录域>**

#隐藏登录界面域#

Setfiles=/etc/openstack-dashboard/local\_settings

sed -i '/MULTIDOMAIN\_SUPPORT/cOPENSTACK\_KEYSTONE\_MULTIDOMAIN\_SUPPORT = False' $Setfiles

**<不同节点使用不同CPU/内存超分>**

<第一种>

we group our different types of servers in host aggregates. some older hardware, some newer. is it possible to tell nova to use different cpu and memory ratios (cpu\_allocation\_ratio and ram\_allocation\_ratio) on different host aggregates?

Yes, its possible to do this.

nova aggregate-set-metadata 1 cpu\_allocation\_ratio=8

nova aggregate-set-metadata 4 cpu\_allocation\_ratio=10

nova aggregate-set-metadata 1 ram\_allocation\_ratio=1

nova aggregate-set-metadata 4 ram\_allocation\_ratio=1.5

Here the 1 and 4 are the ID of your Aggregates.

<第二种>

Hi, I've installed a multi-node Openstack deployment. At install time I configured the cpu\_allocation\_ratio=1.0, which has been applied to all hypervisors.

I would now like to change that ratio on one of my hyper-visors. I updated the settings in the nova.conf on the compute node and restarted the service but it hasn't taken affect. i.e the machine has 24 Cores. When I change cpu\_allocation\_ratio to 4.0 and restart I see the following in the compute.log

14-02-12 11:49:12.756 26381 AUDIT nova.compute.resource\_tracker [-] Free ram (MB): 47742 2014-02-12 11:49:12.756 26381 AUDIT nova.compute.resource\_tracker [-] Free disk (GB): 231 2014-02-12 11:49:12.756 26381 AUDIT nova.compute.resource\_tracker [-] Free VCPUS: 24

Do I need to configure something else, or restart another service ?

Hi ,

First you need add 'CoreFilter' in the following flag in nova.conf of controller :

scheduler\_default\_filters=CoreFilter

and restart the nova-scheduler.

Hopefully this will resolve the problem.

Thanks I had tried that, although I have a number of filters so perhaps there is a ordering issue? scheduler\_default\_filters=AggregateInstanceExtraSpecsFilter,RetryFilter,AvailabilityZoneFilter,RamFilter,ComputeFilter,ComputeCapabilitiesFilter,ImagePropertiesFilter,CoreFilter

DaveJ gravatar imageDaveJ (Feb 14 '14)

I removed all the filters and left only AggregateCoreFilter Then on each Aggregate (through horizon) you can include a new metadata with cpu\_allocation\_ratio and set it to 16.0 (for example). It's the only way it works.

Thanks to Raul at Mirantis for this explanation of what is happening: The Horizon dashboard is presenting the total that is physically available, and now that you have modified the over-subscription ratio the current or in-use count will increase as you deploy more instances and surpass the available number. The reason for this is because the nova scheduler is the component that uses the cpu\_allocation\_ratio to schedule instances and ultimately oversubscribe a host. The figure you are seeing in Horizon is retrieved using the nova API which lists the physically available cores.

注：注意如果进行修改后仍然无法正常开出虚机，在创建虚机界面发现高配置虚机类型不可用（黄色感叹号），需要检查项目CPU，内存配额是否足够。

**<开启密码修改功能>**

第一种：

#补充说明

#参数 --property hw\_qemu\_guest\_agent=yes 是启动qemu-ga功能，重置密码使用

该参数在glance创建镜像时使用

#重置密码方法：

#1、在计算节点

virsh list

virsh set-user-password instance-00000028 --user root --password 123123

#2、在控制节点,按提示输入2次密码

nova set-password vm-ID

<http://www.cnblogs.com/elvi/p/7922421.html>

第二种：

编辑/usr/share/openstack-dashboard/openstack\_dashboard/local/local\_settings.py

# The OPENSTACK\_HYPERVISOR\_FEATURES settings can be used to enable optional

# services provided by hypervisors.

OPENSTACK\_HYPERVISOR\_FEATURES = {

'can\_set\_mount\_point': False,

'can\_set\_password': True,

}

这时候你就可以看到设置管理员密码的地方，不过这个时候，你可能还是不能修改密码。你需要修改计算节点的nova.conf文件

[libvirt]

# Inject the admin password at boot time, without an agent.

# (boolean value)

#inject\_password=false

inject\_password=True

这样重启服务：

openstack-service restart nova

第三种：

可以挂载镜像，更改配置试试

#cloud-int,开启root密码登录#

sed -i 's/disable\_root: 1/disable\_root: 0/g' /etc/cloud/cloud.cfg

sed -i 's/ssh\_pwauth: 0/ssh\_pwauth: 1/g' /etc/cloud/cloud.cfg22

## 故障处理

**<neutron报错1>**

Found error in /var/log/neutron/linuxbridge-agent.log.

INFO neutron.plugins.ml2.drivers.agent.\_common\_agent [req-e511876d-fe0c-4def-83ac-a468255d5521 - - - - -] Linux bridge agent Agent out of sync with plugin!

2016-04-29 14:13:08.264 27614 INFO neutron.agent.securitygroups\_rpc [req-e511876d-fe0c-4def-83ac-a468255d5521 - - - - -] Preparing filters for devices set(['tap5ad39ad4-0d', 'tap026e5c66-46'])

2016-04-29 14:13:08.657 27614 ERROR neutron.plugins.ml2.drivers.agent.\_common\_agent [req-e511876d-fe0c-4def-83ac-a468255d5521 - - - - -] Error in agent loop. Devices info: {'current': set(['tap5ad39ad4-0d', 'tap026e5c66-46']), 'removed': set([]), 'added': set(['tap5ad39ad4-0d', 'tap026e5c66-46']), 'updated': set([])}

... ... ... something like that !

One solution is using neutron-sanity-check to check missed IPSet.

for example, neutron-sanity-check --config-file /etc/neutron/neutron.conf --config-file /etc/neutron/plugins/ml2/ml2\_conf.ini

In this case, for Mitaka openstack version, install ipset on compute node.

解决方法：

Solution:

on compute node, yum install ipset

systemctl restart neutron-linuxbridge-agent

Then error message will disappear, recreate vm, it works well !

**<neutron故障2>**

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.commonagent Traceback (most recent

call last):

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.commonagent File

"/usr/lib/python2.7/site-packages/neutron/plugins/ml2/drivers/agent/commonagent.py",

line 453, in daemonloop

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.commonagent sync =

self.processnetworkdevices(deviceinfo)

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.commonagent File

"/usr/lib/python2.7/site-packages/osprofiler/profiler.py", line 153, in

wrapper

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.commonagent return f(\*args,

\*\*kwargs)

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File

"/usr/lib/python2.7/site-packages/neutron/plugins/ml2/drivers/agent/\_common\_agent.py",

line 213, in process\_network\_devices

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent resync\_b =

self.treat\_devices\_removed(device\_info['removed'])

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File

"/usr/lib/python2.7/site-packages/osprofiler/profiler.py", line 153, in

wrapper

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent return f(\*args,

\*\*kwargs)

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File

"/usr/lib/python2.7/site-packages/neutron/plugins/ml2/drivers/agent/\_common\_agent.py",

line 331, in treat\_devices\_removed

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent

self.sg\_agent.remove\_devices\_filter(devices)

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File

"/usr/lib/python2.7/site-packages/neutron/agent/securitygroups\_rpc.py",

line 238, in remove\_devices\_filter

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent

self.firewall.remove\_port\_filter(device)

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File

"/usr/lib/python2.7/site-packages/neutron/agent/linux/iptables\_firewall.py",

line 222, in remove\_port\_filter

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent

self.\_remove\_conntrack\_entries\_from\_port\_deleted(port)

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File

"/usr/lib/python2.7/site-packages/neutron/agent/linux/iptables\_firewall.py",

line 194, in \_remove\_conntrack\_entries\_from\_port\_deleted

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent [device\_info],

ethertype, set())

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File

"/usr/lib/python2.7/site-packages/neutron/agent/linux/ip\_conntrack.py",

line 121, in delete\_conntrack\_state\_by\_remote\_ips

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent

self.\_delete\_conntrack\_state(device\_info\_list, rule)

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File

"/usr/lib/python2.7/site-packages/neutron/agent/linux/ip\_conntrack.py",

line 103, in \_delete\_conntrack\_state

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent extra\_ok\_codes=[1])

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File

"/usr/lib/python2.7/site-packages/neutron/agent/linux/utils.py", line

128, in execute

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent

execute\_rootwrap\_daemon(cmd, process\_input, addl\_env))

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File

"/usr/lib/python2.7/site-packages/neutron/agent/linux/utils.py", line

115, in execute\_rootwrap\_daemon

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent return

client.execute(cmd, process\_input)

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File

"/usr/lib/python2.7/site-packages/oslo\_rootwrap/client.py", line 128, in

execute

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent res =

proxy.run\_one\_command(cmd, stdin)

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File "", line

2, in run\_one\_command

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent File

"/usr/lib64/python2.7/multiprocessing/managers.py", line 773, in

\_callmethod

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent raise

convert\_to\_error(kind, result)

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent RemoteError:

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent Unserializable message:

('#ERROR', ValueError('I/O operation on closed file',))

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent

2017-04-07 13:37:47.709 1397 ERROR

neutron.plugins.ml2.drivers.agent.\_common\_agent

2017-04-07 13:37:47.710 1397 INFO

neutron.plugins.ml2.drivers.agent.\_common\_agent

[req-93157f5e-9f2d-4547-9002-c473211df719 - - - - -] Linux bridge agent

Agent out of sync with plugin!

解决方法：

Installing the "conntrack-tools" package from the

"centos-openstack-ocata" repository seems to have fixed the problem

**<Neutron 报错3>**

2018-01-23 04:15:49.918 1117 ERROR neutron.agent.linux.utils [req-ecec3ae5-21be-

4047-88ce-76faa52e6695 - - - - -] Rootwrap error running command: ['conntrack',

'-D', '-f', 'ipv4', '-d', '10.99.1.25', '-w', '1']: RemoteError:

2018-01-23 04:15:49.939 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt self.\_remove\_conntrack\_entries\_from\_port\_deleted(port)

2018-01-23 04:15:49.939 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt File "/usr/lib/python2.7/site-packages/neutron/agent/linux/iptables\_firewal

l.py", line 150, in \_remove\_conntrack\_entries\_from\_port\_deleted

2018-01-23 04:15:49.939 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt File "/usr/lib/python2.7/site-packages/neutron/agent/linux/ip\_conntrack.py"

, line 124, in delete\_conntrack\_state\_by\_remote\_ips

2018-01-23 04:15:49.939 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt self.\_delete\_conntrack\_state(device\_info\_list, rule)

2018-01-23 04:15:49.939 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt File "/usr/lib/python2.7/site-packages/neutron/agent/linux/ip\_conntrack.py"

, line 107, in \_delete\_conntrack\_state

2018-01-23 04:15:53.245 1117 ERROR neutron.agent.linux.utils [req-ecec3ae5-21be-

4047-88ce-76faa52e6695 - - - - -] Rootwrap error running command: ['conntrack',

'-D', '-f', 'ipv4', '-d', '10.99.1.25', '-w', '1']: RemoteError:

2018-01-23 04:15:53.267 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt self.\_remove\_conntrack\_entries\_from\_port\_deleted(port)

2018-01-23 04:15:53.267 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt File "/usr/lib/python2.7/site-packages/neutron/agent/linux/iptables\_firewal

l.py", line 150, in \_remove\_conntrack\_entries\_from\_port\_deleted

2018-01-23 04:15:53.267 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt File "/usr/lib/python2.7/site-packages/neutron/agent/linux/ip\_conntrack.py"

, line 124, in delete\_conntrack\_state\_by\_remote\_ips

2018-01-23 04:15:53.267 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt self.\_delete\_conntrack\_state(device\_info\_list, rule)

2018-01-23 04:15:53.267 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt File "/usr/lib/python2.7/site-packages/neutron/agent/linux/ip\_conntrack.py"

, line 107, in \_delete\_conntrack\_state

2018-01-23 04:15:56.432 1117 ERROR neutron.agent.linux.utils [req-ecec3ae5-21be-

4047-88ce-76faa52e6695 - - - - -] Rootwrap error running command: ['conntrack',

'-D', '-f', 'ipv4', '-d', '10.99.1.25', '-w', '1']: RemoteError:

2018-01-23 04:15:56.453 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt self.\_remove\_conntrack\_entries\_from\_port\_deleted(port)

2018-01-23 04:15:56.453 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt File "/usr/lib/python2.7/site-packages/neutron/agent/linux/iptables\_firewal

l.py", line 150, in \_remove\_conntrack\_entries\_from\_port\_deleted

2018-01-23 04:15:56.453 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt File "/usr/lib/python2.7/site-packages/neutron/agent/linux/ip\_conntrack.py"

, line 124, in delete\_conntrack\_state\_by\_remote\_ips

2018-01-23 04:15:56.453 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt self.\_delete\_conntrack\_state(device\_info\_list, rule)

2018-01-23 04:15:56.453 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt File "/usr/lib/python2.7/site-packages/neutron/agent/linux/ip\_conntrack.py"

, line 107, in \_delete\_conntrack\_state

2018-01-23 04:15:59.289 1117 ERROR neutron.agent.linux.utils [req-ecec3ae5-21be-

4047-88ce-76faa52e6695 - - - - -] Rootwrap error running command: ['conntrack',

'-D', '-f', 'ipv4', '-d', '10.99.1.25', '-w', '1']: RemoteError:

2018-01-23 04:15:59.310 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt self.\_remove\_conntrack\_entries\_from\_port\_deleted(port)

2018-01-23 04:15:59.310 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt File "/usr/lib/python2.7/site-packages/neutron/agent/linux/iptables\_firewal

l.py", line 150, in \_remove\_conntrack\_entries\_from\_port\_deleted

2018-01-23 04:15:59.310 1117 ERROR neutron.plugins.ml2.drivers.agent.\_common\_age

nt File "/usr/lib/python2.7/site-packages/neutron/agent/linux/ip\_conntrack.py"

, line 124, in delete\_conntrack\_state\_by\_remote\_ips

修复方法：

Install conntrack-tools on this nodes

**<Cinder示例配置文件>**

[keystone\_authtoken]

memcached\_servers = localhost:11211

signing\_dir = /var/cache/cinder

cafile = /opt/stack/data/ca-bundle.pem

project\_domain\_name = Default

project\_name = service

user\_domain\_name = Default

password = secretservice

username = cinder

auth\_url = http://158.69.89.135/identity

auth\_type = password

[DEFAULT]

cinder\_internal\_tenant\_user\_id = 9f137da6b8cb4373940ce165b7e7bba2

cinder\_internal\_tenant\_project\_id = 46c901db267444579c500c961ccd8584

graceful\_shutdown\_timeout = 5

glance\_api\_version = 2

glance\_api\_servers = http://158.69.89.135/image

osapi\_volume\_workers = 2

logging\_exception\_prefix = ERROR %(name)s [01;35m%(instance)s[00m

logging\_default\_format\_string = %(color)s%(levelname)s %(name)s [[00;36m-%(color)s] [01;35m%(instance)s%(color)s%(message)s[00m

logging\_context\_format\_string = %(color)s%(levelname)s %(name)s [[01;36m%(global\_request\_id)s %(request\_id)s [00;36m%(project\_name)s %(user\_name)s%(color)s] [01;35m%(instance)s%(color)s%(message)s[00m

logging\_debug\_format\_suffix = [00;33m{{(pid=%(process)d) %(funcName)s %(pathname)s:%(lineno)d}}[00m

transport\_url = rabbit://stackrabbit:secretrabbit@158.69.89.135:5672/

backup\_swift\_url = http://158.69.89.135:8080/v1/AUTH\_

default\_volume\_type = lvmdriver-1

enabled\_backends = lvmdriver-1

my\_ip = 158.69.89.135

periodic\_interval = 10

state\_path = /opt/stack/data/cinder

osapi\_volume\_listen = 0.0.0.0

osapi\_volume\_extension = cinder.api.contrib.standard\_extensions

rootwrap\_config = /etc/cinder/rootwrap.conf

api\_paste\_config = /etc/cinder/api-paste.ini

iscsi\_helper = lioadm

debug = True

auth\_strategy = keystone

[database]

connection = mysql+pymysql://root:secretmysql@127.0.0.1/cinder?charset=utf8

[oslo\_concurrency]

lock\_path = /opt/stack/data/cinder

[key\_manager]

fixed\_key = 1ff6e005aad574510416de53eec8c7e8

backend = castellan.key\_manager.barbican\_key\_manager.BarbicanKeyManager

[lvmdriver-1]

image\_volume\_cache\_enabled = True

volume\_clear = none

lvm\_type = thin

iscsi\_helper = lioadm

volume\_group = stack-volumes-lvmdriver-1

volume\_driver = cinder.volume.drivers.lvm.LVMVolumeDriver

volume\_backend\_name = lvmdriver-1

[nova]

region\_name = RegionOne

memcached\_servers = localhost:11211

signing\_dir = /var/cache/cinder

cafile = /opt/stack/data/ca-bundle.pem

project\_domain\_name = Default

project\_name = service

user\_domain\_name = Default

password = secretservice

username = nova

auth\_url = http://158.69.89.135/identity

auth\_type = password

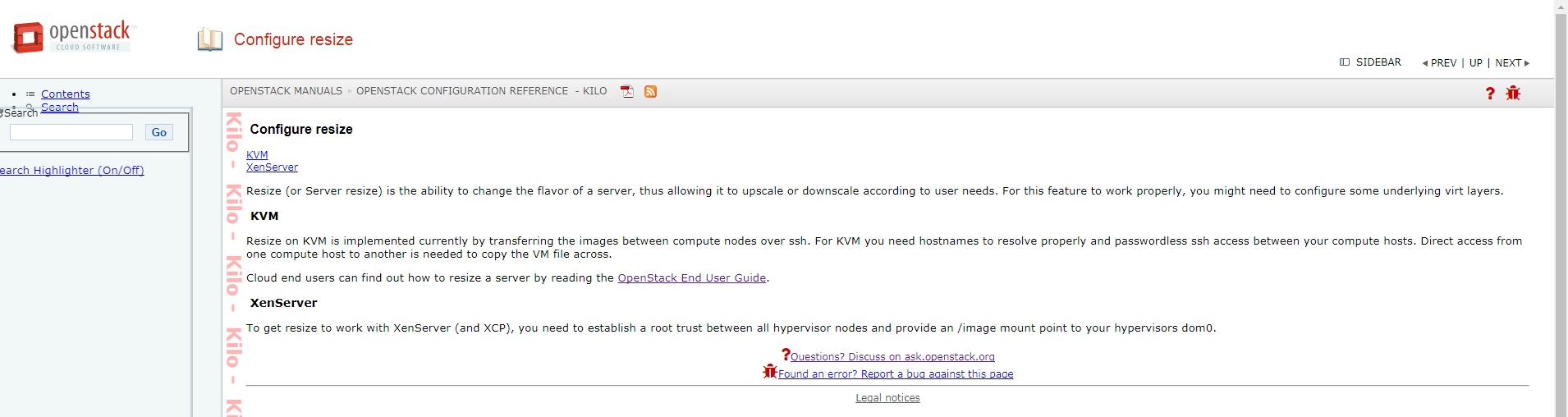
[coordination]

backend\_url = etcd3+http://158.69.89.135:2379

**<节点迁移/resize故障>**

进行节点迁移时报错no valid host was found to resize

<https://docs.openstack.org/kilo/config-reference/content/configuring-resize.html>



**第一种:**

You need to perform the following actions:

Update the “nova.conf” file in both controller and compute with the following lines:

allow\_migrate\_to\_same\_host = True

scheduler\_default\_filters = AllHostsFilter

allow\_resize\_to\_same\_host = True

Restart the following services of the "controller" node

• nova-api

• nova-cert

• nova-consoleauth

• nova-scheduler

• nova-conductor

• nova-novncproxy

Restart the following service of the compute node:

• nova-compute

**第二种：**

Today I had to resize an instance in my OpenStack and noticed that I couldn’t do that because of an error: “No valid host was found. No valid host found for resize (HTTP 400)“. Soon enough I understood OpenStack was trying to resize the instance by using another host, however my setup is a all-in-one node.

The solution

is rather simple: set the property allow\_resize\_to\_same\_host to true in /etc/nova/nova.conf . After this restart nova-compute and nova-api by doing

# systemctl restart openstack-nova-compute

# systemctl restart openstack-nova-api

1

2

# systemctl restart openstack-nova-compute

# systemctl restart openstack-nova-api

Also be careful, if you set up to save the root disk inside of Cinder (boot and create volume). I still haven’t found a way to resize using Nova. Inexplicably the request times out and the instance enters the state VM\_ERROR. The workaround I used is:

Delete the instance while keeping the volume that contains the root.

Create a new instance booting from the root volume of the precedent.

Re-assign floating IP.

EDIT: After a few tweaks and a reboot I can now resize instances with root disk inside Cinder. It was probably due to a malfunction in the cinder-scheduler component.

**第三种：**

If you only have one host, you may need to check out the nova.conf options

"allow\_resize\_to\_same\_host" or "allow\_migrate\_to\_same\_host"

**第四种：**

If you are running a single node OpenStack, you can still resize (and migrate will migrate back to the same host) by adding the following configuration option to your nova.conf:

allow\_resize\_to\_same\_host=True

Then restart your nova scheduler and services:

systemctl restart "openstack-nova-\*"

You can now resize your instances on the same host.

It turns out "resize" is just kind of migration. So extra compute node is needed. After adding a second compute node. Resize works like a charm.

BTW, in order to resize successfully, two compute hosts need to be "ssh" to each other without password.

测试结果:修改allow\_resize\_to\_same\_host=True，只需修改控制节点。

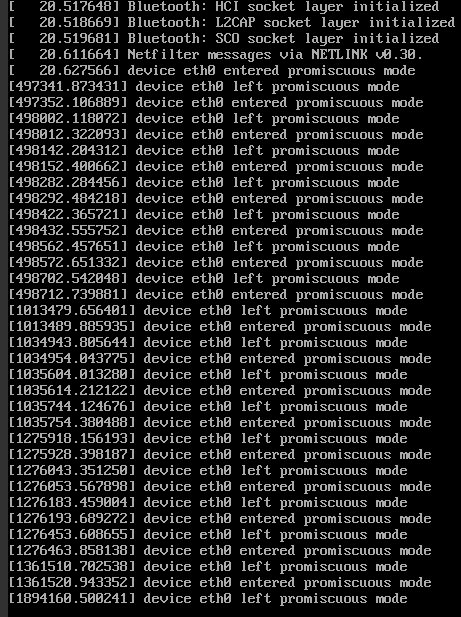
#注：多节点间迁移可能需要配置节点间ssh无密码访问

**<虚机无法连接网络故障>**

虚机正常使用中，开机状态下空置2h到7days之间，虚机屏幕出现如下讯息：

device eth0 left promiscuous mode

device eth0 entered promiscuous mode



<临时方案>待验证

**1）**

<https://bugs.launchpad.net/ubuntu/+source/qemu-kvm/+bug/997978>

This bug affects me as well, is it related to the one discussed here https://bugzilla.kernel.org/show\_bug.cgi?id=42829

I will try using the vhost\_net driver in the host and vhost=on as a guest parameter to see if it bypasses the issue.

Stephane Neveu (stefneveu) wrote on 2012-06-29: #35

Alex,

Thanks for the link, I'll also try to test with :

<driver name='vhost' txmode='iothread' ioeventfd='on' event\_idx='off'/>

Does it work better for you with vhost\_net ?

Stephane Neveu (stefneveu) wrote on 2012-07-03: #36

Ok just in cas it may help...

It does work modprobing vhost\_net and adding :

<driver name='vhost' txmode='iothread' ioeventfd='on' event\_idx='off'/>

on every nics definitions.

Tested on more than 100 VMs.

**2）**

Steve, thank you for finding this bug.

Is there a chance you could test without vhost but with the event\_idx=off ?

This is the xml line in case you use libvirt:

<interface .......

<driver name="qemu" event\_idx="off"/>

</interface>

That would be an effective workaround until we get a fixed qemu.

Thanks for your time.

**3）**

Will there be a fix for 10.04 lucid, too?

I think this bug affects me too: Windows SBS 2008 with virtio network driver on ubuntu 10.04 lucid host loses network connection from time to time. It's not really deterministic when this error occurs. Usually it happens when there's a large amount of traffic sent over the network. After disabling (via VNC console) the network card in Windows and re-enabling it, everything works fine again, no need to reboot the VM.

Any help is appreciated, since it's a quite annoying bug... :-)

**4)**

I faced probably the same problem installing Xen-4.3.2 with Gentoo kernel-3.12.21.

DomU's interface hangs after short time under heavy network load (starting at ~10Mbyte/s). From outside it looks, like the instance would crash, but deactivating and activating the interface e.g. form the "xl console <domU name>" with /etc/init.d/net.eth0 stop/start, restores normal operation.

After 3 days of testing/searching I found a workaround. Setting the following options with ethtool, I could successfully prevent my domU's interfaces from hanging:

**ethtool --offload <network device> gso off tso off sg off gro off**

This http://cloudnull.io/2012/07/xenserver-network-tuning/ leeds me to my solution.

I also posted my other expirience with this bridged network configuration in the Gentoo wiki https://wiki.gentoo.org/wiki/Xen .

**5)**

My problem with missing arp replies was solved( worked around) by setting

bridge\_ageing 0

for the bridge in /etc/network/interfaces, making it a hub forwarding all packets to all hosts.

**6)**

I report the same situation as comment 129.

Spark Cluster installed with Ambari 1.7, installed HDP 2.2 running Zookeeper, Ganglia, HDFS and YARN.

Ubuntu 12.04

uname -r

3.2.0-67-virtual

Instances running fine until the start of a Spark of Hadoop job with YARN. The Job gets accepted but then 2 slaves of the cluster are interested by the bug and lose connectivity. They are still accessible by the openstack web page console but can't reach the network. Reboots brings the VM in a halt state.

Solved with ethtool -K eth0 tx off sg off tso off ufo off gso off gro off lro off

**7)**

修复前网卡offload设置

[root@compute1 ~]# ethtool --show-offload eth2

Features for eth2:

rx-checksumming: on

tx-checksumming: on

tx-checksum-ipv4: off [fixed]

tx-checksum-ip-generic: on

tx-checksum-ipv6: off [fixed]

tx-checksum-fcoe-crc: off [fixed]

tx-checksum-sctp: on

scatter-gather: on

tx-scatter-gather: on

tx-scatter-gather-fraglist: off [fixed]

tcp-segmentation-offload: on

tx-tcp-segmentation: on

tx-tcp-ecn-segmentation: off [fixed]

tx-tcp6-segmentation: on

tx-tcp-mangleid-segmentation: off

udp-fragmentation-offload: off [fixed]

generic-segmentation-offload: on

generic-receive-offload: on

large-receive-offload: off [fixed]

rx-vlan-offload: on

tx-vlan-offload: on

ntuple-filters: off

receive-hashing: on

highdma: on [fixed]

rx-vlan-filter: on [fixed]

vlan-challenged: off [fixed]

tx-lockless: off [fixed]

netns-local: off [fixed]

tx-gso-robust: off [fixed]

tx-fcoe-segmentation: off [fixed]

tx-gre-segmentation: on

tx-ipip-segmentation: on

tx-sit-segmentation: on

tx-udp\_tnl-segmentation: on

tx-mpls-segmentation: off [fixed]

fcoe-mtu: off [fixed]

tx-nocache-copy: off

loopback: off [fixed]

rx-fcs: off [fixed]

rx-all: off

tx-vlan-stag-hw-insert: off [fixed]

rx-vlan-stag-hw-parse: off [fixed]

rx-vlan-stag-filter: off [fixed]

busy-poll: off [fixed]

tx-gre-csum-segmentation: on

tx-udp\_tnl-csum-segmentation: on

tx-gso-partial: on

tx-sctp-segmentation: off [fixed]

l2-fwd-offload: off [fixed]

hw-tc-offload: off [fixed]

[root@compute1 ~]# vi /etc/ne

</临时方案>

**<Neutron消息队列超时过多>**

l3-agent.log:2018-01-24 20:45:20.170 1249 ERROR neutron.common.rpc [req-a56dc217

-e30a-40ef-b445-29bc828d6b0f - - - - -] Timeout in RPC method get\_service\_plugin

\_list. Waiting for 16 seconds before next attempt. If the server is not down, co

nsider increasing the rpc\_response\_timeout option as Neutron server(s) may be ov

erloaded and unable to respond quickly enough.: MessagingTimeout: Timed out wait

ing for a reply to message ID 0c41aeabee3d45b9bee4234fc1255c1f

1）

网络情况较复杂或带宽较少情况下容易出现大量消息队列响应超时

尝试增大neutron配置文件内rpc\_response\_timeout

**<Cinder容量错误>**

使用Cinder时，dashboard上面看到的容量与cinder使用的容量不一致

在计算节点上修改即可

解决方法:

I have the same issue, My openstack is liberty. And the nova report that have no valid host to build VM when the disk size of new vm is more bigger than the remaining disk size. but I build all vm in LVM block storage server, not compute node. And the options [libvirt] images\_type = lvm images\_volume\_group = cinder-volumes are useless.

Hi all, I'm answering myself since I found a solution that worked to me.

I identified this same problem in Openstack Liberty and Juno and IMHO the installation guide for Liberty lacks a configuration of nova-compute service in the compute nodes. I already reported a bug.

I solved the problem by adding the next options in /etc/nova/nova.conf of each compute node:

[libvirt] images\_type = lvm images\_volume\_group = cinder-volumes

Assume that LVM is used and the volume group is named as 'cinder-volumes'.

Then, you need to restart the nova-compute service in each compute node:

sudo service nova-compute restart

Now, the dashboard should report the correct storage size in your volume groups.

Hope it helps! Best, Lluis

**<VNC无法连接>**

VNC无法连接，查看/var/log/nova/nova-novncproxy.log发现pymysql报Toomany connections错误。

解决方法：

1.mariadb数据库最大连接数，默认为151

MariaDB [(none)]> show variables like 'max\_connections';

+-----------------+-------+

| Variable\_name | Value |

+-----------------+-------+

| max\_connections | 151 |

+-----------------+-------+

2.配置/etc/my.cnf

[mysqld]下新添加一行如下参数：

max\_connections=3000

systemctl restart mariadb 重启mariadb服务，再次查看mariadb数据库最大连接数，最大连接数是214，并非我们设置的3000。

MariaDB [(none)]> show variables like 'max\_connections';

+-----------------+-------+

| Variable\_name | Value |

+-----------------+-------+

| max\_connections | 214 |

+-----------------+-------+

由于mariadb有默认打开文件数限制

vi /usr/lib/systemd/system/mariadb.service

取消[Service]前的#号，

[Service]新添加两行如下参数：

LimitNOFILE=10000

LimitNPROC=10000

4.重新加载系统服务，并重启mariadb服务

#systemctl --system daemon-reload 疑似不生效，使用下面一行

systemctl daemon-reload

systemctl restart mariadb.service

再次查看mariadb数据库最大连接数，可以看到最大连接数已经是3000

MariaDB [(none)]> show variables like 'max\_connections';

+-----------------+-------+

| Variable\_name | Value |

+-----------------+-------+

| max\_connections | 3000 |

+-----------------+-------+

**<虚机网速慢，连接丢包量大>**

虚机flat网络疑似丢包严重，ssh偶有连接不上，线上虚机只有10几台，management和flat网络均为千兆网卡。

解决:检查控制节点flat provider网卡流量状况，是否虚机网络flat流量均经控制节点出口流出，由控制节点流出的，说明计算节点flat provider网卡不通，注意检查计算节点provider网卡的网关，路由设置。