目录

[**一、服务器配置** 2](#_Toc381091413)

[**二、组件安装** 3](#_Toc381091414)

[**三、数据库安装配置** 4](#_Toc381091415)

[**四、Keystone配置** 4](#_Toc381091416)

[**五、glance配置** 7](#_Toc381091417)

[**六、nova配置** 8](#_Toc381091418)

[**七、dashboard配置** 11](#_Toc381091419)

[**八、镜像制作** 11](#_Toc381091420)

[**附件一：配置文件** 11](#_Toc381091421)

[**附件二：安装错误处理** 18](#_Toc381091422)

**一、服务器配置**

**1、下载ubuntu 12.04. 服务器版本。**

地址：http://releases.ubuntu.com//precise/ubuntu-12.04-beta2-server-amd64.iso

**2、安装OS**

操作系统最小化安装，只需要安装ssh server，其他组件不需要。

操作系统安装好需要更新源里的包、系统。确保装的是最新版本的包。命令如下：

|  |
| --- |
| apt-get update  apt-get upgrade |

注：更新时网络代理配置如下：

|  |
| --- |
| root@ubuntu:/opt# cat /etc/apt/apt.conf  Acquire::http::Proxy "http://133.64.81.236:8080/"; |

**3、root权限**

|  |
| --- |
| zhang@ubuntu:~$ sudo passwd  [sudo] password for zhang:  Enter new UNIX password:  Retype new UNIX password:  passwd: password updated successfully |

**4、网络配置**

|  |
| --- |
| root@ubuntu:/opt# cat /etc/network/interfaces  # This file describes the network interfaces available on your system  # and how to activate them. For more information, see interfaces(5).  # The loopback network interface  auto lo  iface lo inet loopback  # The primary network interface  auto eth0  iface eth0 inet static  address 133.64.96.26  netmask 255.255.255.128  network 133.64.96.0  broadcast 133.64.96.127  gateway 133.64.96.1  # dns-\* options are implemented by the resolvconf package, if installed  dns-nameservers 133.64.96.1  auto eth1  iface eth1 inet static  address 192.168.3.130  netmask 255.255.255.128  network 192.168.3.128  broadcast 192.168.3.255 |

**二、组件安装**

**1、bridge配置**

使用apt-get安装如下：

|  |
| --- |
| apt-get install bridge-utils  /etc/init.d/networking restart |

**2、NTP配置**

|  |
| --- |
| apt-get install ntp |

编辑 /etc/ntp.conf 在末尾添加下面3行

|  |
| --- |
| server ntp.ubuntu.com iburst  server 127.127.1.0  fudge 127.127.1.0 stratum 10 |

重启服务

|  |
| --- |
| service ntp restart |

**3、iscsi配置**

|  |
| --- |
| apt-get install tgt  /etc/init.d/tgt start  apt-get install open-iscsi open-iscsi-utils |

**4、rabbitmq配置**

|  |
| --- |
| apt-get install rabbitmq-server memcached python-memcache  apt-get install kvm libvirt-bin |

**三、数据库安装配置**

**1、安装数据库**

|  |
| --- |
| apt-get install mysql-server python-mysqldb |

编辑/etc/mysql/my.cnf, 允许网络访问mysql

|  |
| --- |
| #bind-address = 127.0.0.1  bind-address = 0.0.0.0 |

重启mysql服务

|  |
| --- |
| /etc/init.d/mysql restart |

**2、创建相关数据库**

|  |
| --- |
| mysql -uroot -p123456  CREATE DATABASE nova;  GRANT ALL PRIVILEGES ON nova.\* TO 'nova'@'%' IDENTIFIED BY '123456';  CREATE DATABASE glance;  GRANT ALL PRIVILEGES ON glance.\* TO 'glance'@'%' IDENTIFIED BY '123456';  CREATE DATABASE keystone;  GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'%'IDENTIFIED BY '123456';  quit |

**四、Keystone配置**

**1、keystone的安装**

|  |
| --- |
| apt-get install keystone python-keystone python-keystoneclient |

**2、keystone配置**

编辑/etc/keystone/keystone.conf

|  |
| --- |
| [DEFAULT]  #bind\_host = 0.0.0.0  public\_port = 5000  admin\_port = 35357  #admin\_token = ADMIN  admin\_token = admin  [sql]  #connection = sqlite:////var/lib/keystone/keystone.db  connection = mysql://keystone:123456@133.64.96.26/keystone |

**3、重启服务**

|  |
| --- |
| service keystone restart |

**4、同步数据库**

|  |
| --- |
| keystone-manage db\_sync |

**5、导入数据和endpoint**

为了方便，你可以直接使用下面2个脚本来进行全部的设置

（1）、keystone\_data.sh 导入用户信息

|  |
| --- |
| wget http://www.chenshake.com/wp-content/uploads/2012/07/keystone\_data.sh\_.txt  mv keystone\_data.sh\_.txt keystone\_data.sh  chmod +x keystone\_data.sh |

对于keystone\_data.sh 脚本，默认的登陆dashboard的密码是：chenshake，Token是chenshake。

你可以根据你的情况进行调整。

第一行是登陆dashboard的密码。

第三行是上面设置的Keystone的Token

|  |
| --- |
| ADMIN\_PASSWORD=${ADMIN\_PASSWORD:-123456}  SERVICE\_PASSWORD=${SERVICE\_PASSWORD:-$ADMIN\_PASSWORD}  #export SERVICE\_TOKEN="chenshake"  export SERVICE\_TOKEN="admin"  export SERVICE\_ENDPOINT="http://localhost:35357/v2.0"  SERVICE\_TENANT\_NAME=${SERVICE\_TENANT\_NAME:-service}  ENABLED\_SERVICES="swift" |

验证是否正常

|  |
| --- |
| ./keystone\_data.sh 没任何输出，就表示正确  echo $? 显示0，就表示脚本正确运行 |

（2）、endpoints.sh 设置endpoint

|  |
| --- |
| wget http://www.chenshake.com/wp-content/uploads/2012/07/endpoints.sh\_.txt  mv endpoints.sh\_.txt endpoints.sh  chmod +x endpoints.sh |

这个脚本运行，需要使用不少参数

|  |
| --- |
| ./endpoints.sh -m 133.64.96.26 -u keystone -D keystone -p 123456 -T admin -K 133.64.96.26 -R RegionOne -E "http://localhost:35357/v2.0" -S 133.64.96.26 |

参数说明

|  |
| --- |
| -m mysql\_hostname  -u mysql\_username  -D mysql\_database  -p mysql\_password  -K keystone 服务器IP  -R keystone\_region  -E keystone\_endpoint\_url  -S swift proxy节点IP  -T keystone\_token |

正常运行，会输出一堆内容。

（3）、设置环境变量

|  |
| --- |
| root@server1:~# Vi ~/.bashrc.或/etc/profile 在文件末尾处  export OS\_TENANT\_NAME=admin  export OS\_USERNAME=admin  export OS\_PASSWORD=123456  export OS\_AUTH\_URL=http://localhost:5000/v2.0/ |

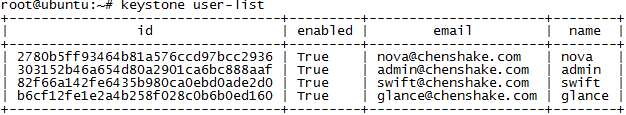
验证：

|  |
| --- |
| root@server1:~#source ~/.bashrc.或/etc/profile  root@server1:~# export | grep OS\_  declare -x OS\_AUTH\_URL="http://localhost:5000/v2.0/"  declare -x OS\_PASSWORD="123456"  declare -x OS\_TENANT\_NAME="admin"  declare -x OS\_USERNAME="admin" |

**6、测试keyston是否正确安装**

|  |
| --- |
| keystone user-list  keystone endpoint-list  keystone tenant-list  keystone user-list  keystone role-list |

如：



若测试成功说明keystone安装正确。

**五、glance配置**

**1、glance安装**

|  |
| --- |
| apt-get install glance glance-api glance-client glance-common glance-registry python-glance |

**2、glance配置**

编辑 /etc/glance/glance-api-paste.ini，/etc/glance/glance-registry-paste.ini,两个文件，都是修改文档最后3行

|  |
| --- |
| #admin\_tenant\_name = %SERVICE\_TENANT\_NAME%  #admin\_user = %SERVICE\_USER%  #admin\_password = %SERVICE\_PASSWORD%  admin\_tenant\_name = service  admin\_user = glance  admin\_password = 123456  #dashboard password |

编辑/etc/glance/glance-registry.conf，改成使用mysql验证

|  |
| --- |
| #sql\_connection = sqlite:////var/lib/glance/glance.sqlite  sql\_connection = mysql://glance:123456@122.204.144.201/glance |

编辑/etc/glance/glance-registry.conf 和 /etc/glance/glance-api.conf ，都在文件末尾添加两行

|  |
| --- |
| [paste\_deploy]  flavor = keystone |

**3、glance 同步数据库**

|  |
| --- |
| glance-manage version\_control 0  glance-manage db\_sync |

**4、重启服务**

|  |
| --- |
| service glance-api restart && service glance-registry restart |

**5、设置永久环境变量**

修改 ~/.bashrc.或/etc/profile , 在末尾添加下面内容

|  |
| --- |
| export OS\_TENANT\_NAME=admin  export OS\_USERNAME=admin  export OS\_PASSWORD=123456  export OS\_AUTH\_URL=http://localhost:5000/v2.0/ |

注：在keystone中设置过环境变量后无需再配置环境变量。

**6、测试glance**

|  |
| --- |
| glance index |

没有输出，表示正常，因为目前还没有镜像。

**六、nova配置**

**1、安装nova**

|  |
| --- |
| apt-get install nova-api nova-cert nova-compute nova-compute-kvm nova-doc nova-network nova-objectstore nova-scheduler nova-volume rabbitmq-server novnc nova-consoleauth |

**2、配置nova**

使用vi /etc/nova/nova.conf编辑该文件修改。

外网地址133.64.96.26。内网地址192.168.4.130。

|  |
| --- |
| --dhcpbridge\_flagfile=/etc/nova/nova.conf  --dhcpbridge=/usr/bin/nova-dhcpbridge  --logdir=/var/log/nova  --state\_path=/var/lib/nova  --lock\_path=/run/lock/nova  --allow\_admin\_api=true  --use\_deprecated\_auth=false  --auth\_strategy=keystone  --scheduler\_driver=nova.scheduler.simple.SimpleScheduler  --s3\_host=133.64.96.26  --ec2\_host=133.64.96.26  --rabbit\_host=133.64.96.26  --cc\_host=133.64.96.26  --nova\_url=http://133.64.96.26:8774/v1.1/  --routing\_source\_ip=133.64.96.26  --glance\_api\_servers=133.64.96.26:9292  --image\_service=nova.image.glance.GlanceImageService  --iscsi\_ip\_prefix=192.168.4  --sql\_connection=mysql://nova:123456@133.64.96.26/nova  --ec2\_url=http://133.64.96.26:8773/services/Cloud  --keystone\_ec2\_url=http://133.64.96.26:5000/v2.0/ec2tokens  --api\_paste\_config=/etc/nova/api-paste.ini  --libvirt\_type=kvm  --libvirt\_use\_virtio\_for\_bridges=true  --start\_guests\_on\_host\_boot=true  --resume\_guests\_state\_on\_host\_boot=true  # vnc specific configuration  --novnc\_enabled=true  --novncproxy\_base\_url=http://133.64.96.26:6080/vnc\_auto.html  --vncserver\_proxyclient\_address=133.64.96.26  --vncserver\_listen=133.64.96.26  # network specific settings  --network\_manager=nova.network.manager.FlatDHCPManager  --public\_interface=eth0  --flat\_interface=eth1  --flat\_network\_bridge=br100  --fixed\_range=192.168.4.130/25  --floating\_range=133.64.96.26/25  --network\_size=32  --flat\_network\_dhcp\_start=192.168.4.162  --flat\_injected=False  --force\_dhcp\_release  --iscsi\_helper=tgtadm  --connection\_type=libvirt  --root\_helper=sudo nova-rootwrap  --verbose |

使用vi /etc/nova/api-paste.ini编辑文件修改！

|  |
| --- |
| #admin\_tenant\_name = %SERVICE\_TENANT\_NAME%  #admin\_user = %SERVICE\_USER%  #admin\_password = %SERVICE\_PASSWORD%  admin\_tenant\_name = service  admin\_user = nova  admin\_password = 123456 |

**3、相关服务重启**

|  |
| --- |
| /etc/init.d/libvirt-bin restart  /etc/init.d/nova-network restart  /etc/init.d/nova-compute restart  /etc/init.d/nova-api restart  /etc/init.d/nova-objectstore restart  /etc/init.d/nova-scheduler restart  /etc/init.d/nova-volume restart  /etc/init.d/nova-consoleauth restart |

**4、同步数据库**

|  |
| --- |
| nova-manage db sync |

**5、设置目录权限**

|  |
| --- |
| chown -R nova:nova /etc/nova  chmod 644 /etc/nova/nova.conf |

**6、创建fix ip(内网ip)**

|  |
| --- |
| nova-manage network create private --fixed\_range\_v4=192.168.4.130/25 --num\_networks=1 --bridge=br100 --bridge\_interface=eth1 --network\_size=32 |

**7、创建floating IP(公网ip)**

|  |
| --- |
| nova-manage floating create --ip\_range=133.64.96.26/25 |

**8、重启服务**

|  |
| --- |
| /etc/init.d/libvirt-bin restart  /etc/init.d/nova-network restart  /etc/init.d/nova-compute restart  /etc/init.d/nova-api restart  /etc/init.d/nova-objectstore restart  /etc/init.d/nova-scheduler restart  /etc/init.d/nova-volume restart  /etc/init.d/nova-consoleauth restart |

**9、验证测试**

|  |
| --- |
| nova-manage service list  nova list  nova image-list  nova floating-ip-create  nova flavor-list  nova secgroup-list  nova secgroup-list-rules default |

**10、开放远程连接端口**

|  |
| --- |
| 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 |

**七、dashboard配置**

**1、安装dashboard**

|  |
| --- |
| apt-get install openstack-dashboard |

**2、重启apache**

|  |
| --- |
| service apache2 restart |

**八、镜像制作**

下载官方做好的ubuntu镜像即可

|  |
| --- |
| wget http://cloud-images.ubuntu.com/precise/current/precise-server-cloudimg-amd64-disk1.img |

上传镜像

|  |
| --- |
| glance add name="Ubuntu 12.04 cloudimg amd64" is\_public=true container\_format=ovf disk\_format=qcow2 < /root/precise-server-cloudimg-amd64-disk1.img |

**附件一：配置文件**

**keystone\_data.sh**

|  |
| --- |
| #!/bin/bash  #  # Initial data for Keystone using python-keystoneclient  #  # Tenant User Roles  # ------------------------------------------------------------------  # admin admin admin  # service glance admin  # service nova admin, [ResellerAdmin (swift only)]  # service quantum admin # if enabled  # service swift admin # if enabled  # demo admin admin  # demo demo Member, anotherrole  # invisible\_to\_admin demo Member  #  # Variables set before calling this script:  # SERVICE\_TOKEN - aka admin\_token in keystone.conf  # SERVICE\_ENDPOINT - local Keystone admin endpoint  # SERVICE\_TENANT\_NAME - name of tenant containing service accounts  # ENABLED\_SERVICES - stack.sh's list of services to start  # DEVSTACK\_DIR - Top-level DevStack directory  #ADMIN\_PASSWORD=${ADMIN\_PASSWORD:-chenshake}  ADMIN\_PASSWORD=${ADMIN\_PASSWORD:-$OS\_PASSWORD}  #SERVICE\_PASSWORD=${SERVICE\_PASSWORD:-$ADMIN\_PASSWORD}  #export SERVICE\_TOKEN="chenshake"  #export SERVICE\_ENDPOINT="http://localhost:35357/v2.0"  SERVICE\_TENANT\_NAME=${SERVICE\_TENANT\_NAME:-service}  ENABLED\_SERVICES="swift"  function get\_id () {  echo `$@ | awk '/ id / { print $4 }'`  }  # Tenants  ADMIN\_TENANT=$(get\_id keystone tenant-create --name=admin)  SERVICE\_TENANT=$(get\_id keystone tenant-create --name=$SERVICE\_TENANT\_NAME)  #DEMO\_TENANT=$(get\_id keystone tenant-create --name=demo)  #INVIS\_TENANT=$(get\_id keystone tenant-create --name=invisible\_to\_admin)  # Users  ADMIN\_USER=$(get\_id keystone user-create --name=admin \  --pass="$ADMIN\_PASSWORD" \  --email=admin@chenshake.com)  #DEMO\_USER=$(get\_id keystone user-create --name=demo \  # --pass="$ADMIN\_PASSWORD" \  # --email=demo@chenshake.com)  # Roles  ADMIN\_ROLE=$(get\_id keystone role-create --name=admin)  KEYSTONEADMIN\_ROLE=$(get\_id keystone role-create --name=KeystoneAdmin)  KEYSTONESERVICE\_ROLE=$(get\_id keystone role-create --name=KeystoneServiceAdmin)  # ANOTHER\_ROLE demonstrates that an arbitrary role may be created and used  # TODO(sleepsonthefloor): show how this can be used for rbac in the future!  ANOTHER\_ROLE=$(get\_id keystone role-create --name=anotherrole)  # Add Roles to Users in Tenants  keystone user-role-add --user $ADMIN\_USER --role $ADMIN\_ROLE --tenant\_id $ADMIN\_TENANT  #keystone user-role-add --user $ADMIN\_USER --role $ADMIN\_ROLE --tenant\_id $DEMO\_TENANT  #keystone user-role-add --user $DEMO\_USER --role $ANOTHER\_ROLE --tenant\_id $DEMO\_TENANT  # TODO(termie): these two might be dubious  keystone user-role-add --user $ADMIN\_USER --role $KEYSTONEADMIN\_ROLE --tenant\_id $ADMIN\_TENANT  keystone user-role-add --user $ADMIN\_USER --role $KEYSTONESERVICE\_ROLE --tenant\_id $ADMIN\_TENANT  # The Member role is used by Horizon and Swift so we need to keep it:  MEMBER\_ROLE=$(get\_id keystone role-create --name=Member)  #keystone user-role-add --user $DEMO\_USER --role $MEMBER\_ROLE --tenant\_id $DEMO\_TENANT  #keystone user-role-add --user $DEMO\_USER --role $MEMBER\_ROLE --tenant\_id $INVIS\_TENANT  # Configure service users/roles  NOVA\_USER=$(get\_id keystone user-create --name=nova \  --pass="$SERVICE\_PASSWORD" \  --tenant\_id $SERVICE\_TENANT \  --email=nova@chenshake.com)  keystone user-role-add --tenant\_id $SERVICE\_TENANT \  --user $NOVA\_USER \  --role $ADMIN\_ROLE  GLANCE\_USER=$(get\_id keystone user-create --name=glance \  --pass="$SERVICE\_PASSWORD" \  --tenant\_id $SERVICE\_TENANT \  --email=glance@chenshake.com)  keystone user-role-add --tenant\_id $SERVICE\_TENANT \  --user $GLANCE\_USER \  --role $ADMIN\_ROLE  if [[ "$ENABLED\_SERVICES" =~ "swift" ]]; then  SWIFT\_USER=$(get\_id keystone user-create --name=swift \  --pass="$SERVICE\_PASSWORD" \  --tenant\_id $SERVICE\_TENANT \  --email=swift@chenshake.com)  keystone user-role-add --tenant\_id $SERVICE\_TENANT \  --user $SWIFT\_USER \  --role $ADMIN\_ROLE  # Nova needs ResellerAdmin role to download images when accessing  # swift through the s3 api. The admin role in swift allows a user  # to act as an admin for their tenant, but ResellerAdmin is needed  # for a user to act as any tenant. The name of this role is also  # configurable in swift-proxy.conf  RESELLER\_ROLE=$(get\_id keystone role-create --name=ResellerAdmin)  keystone user-role-add --tenant\_id $SERVICE\_TENANT \  --user $NOVA\_USER \  --role $RESELLER\_ROLE  fi  if [[ "$ENABLED\_SERVICES" =~ "quantum" ]]; then  QUANTUM\_USER=$(get\_id keystone user-create --name=quantum \  --pass="$SERVICE\_PASSWORD" \  --tenant\_id $SERVICE\_TENANT \  --email=quantum@chenshake.com)  keystone user-role-add --tenant\_id $SERVICE\_TENANT \  --user $QUANTUM\_USER \  --role $ADMIN\_ROLE  fi |

**endpoints.sh配置文件**

|  |
| --- |
| #!/bin/sh  # Author: Martin Gerhard Loschwitz  # (c) 2012 hastexo Professional Services GmbH  # Licensed under the Apache License, Version 2.0 (the "License");  # you may not use this file except in compliance with the License.  # You may obtain a copy of the License at  #  # http://www.apache.org/licenses/LICENSE-2.0  #  # Unless required by applicable law or agreed to in writing, software  # distributed under the License is distributed on an "AS IS" BASIS,  # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  # See the License for the specific language governing permissions and  # limitations under the License.  #  # On Debian-based systems the full text of the Apache version 2.0  # license can be found in `/usr/share/common-licenses/Apache-2.0'.  # MySQL definitions  MYSQL\_USER=keystone  MYSQL\_DATABASE=keystone  MYSQL\_PASSWORD=$MYSQL\_PASS  MYSQL\_HOST=$MASTER  #MYSQL\_HOST=localhost  # Keystone definitions  KEYSTONE\_REGION=RegionOne  #SERVICE\_TOKEN=password  SERVICE\_ENDPOINT="http://localhost:35357/v2.0"  # other definitions  #MASTER=localhost  while getopts "u:D:p:m:K:R:E:S:T:vh" opt; do  case $opt in  u)  MYSQL\_USER=$OPTARG  ;;  D)  MYSQL\_DATABASE=$OPTARG  ;;  p)  MYSQL\_PASSWORD=$OPTARG  ;;  m)  MYSQL\_HOST=$OPTARG  ;;  K)  MASTER=$OPTARG  ;;  R)  KEYSTONE\_REGION=$OPTARG  ;;  E)  export SERVICE\_ENDPOINT=$OPTARG  ;;  S)  SWIFT\_MASTER=$OPTARG  ;;  T)  export SERVICE\_TOKEN=$OPTARG  ;;  v)  set -x  ;;  h)  cat <<EOF  Usage: $0 [-m mysql\_hostname] [-u mysql\_username] [-D mysql\_database] [-p mysql\_password]  [-K keystone\_master ] [ -R keystone\_region ] [ -E keystone\_endpoint\_url ]  [ -S swift\_master ] [ -T keystone\_token ]    Add -v for verbose mode, -h to display this message.  EOF  exit 0  ;;  \?)  echo "Unknown option -$OPTARG" >&2  exit 1  ;;  :)  echo "Option -$OPTARG requires an argument" >&2  exit 1  ;;  esac  done  if [ -z "$KEYSTONE\_REGION" ]; then  echo "Keystone region not set. Please set with -R option or set KEYSTONE\_REGION variable." >&2  missing\_args="true"  fi  if [ -z "$SERVICE\_TOKEN" ]; then  echo "Keystone service token not set. Please set with -T option or set SERVICE\_TOKEN variable." >&2  missing\_args="true"  fi  if [ -z "$SERVICE\_ENDPOINT" ]; then  echo "Keystone service endpoint not set. Please set with -E option or set SERVICE\_ENDPOINT variable." >&2  missing\_args="true"  fi  if [ -z "$MYSQL\_PASSWORD" ]; then  echo "MySQL password not set. Please set with -p option or set MYSQL\_PASSWORD variable." >&2  missing\_args="true"  fi  if [ -n "$missing\_args" ]; then  exit 1  fi  keystone service-create --name nova --type compute --description 'OpenStack Compute Service'  keystone service-create --name volume --type volume --description 'OpenStack Volume Service'  keystone service-create --name glance --type image --description 'OpenStack Image Service'  keystone service-create --name swift --type object-store --description 'OpenStack Storage Service'  keystone service-create --name keystone --type identity --description 'OpenStack Identity'  keystone service-create --name ec2 --type ec2 --description 'OpenStack EC2 service'  create\_endpoint () {  case $1 in  compute)  keystone endpoint-create --region $KEYSTONE\_REGION --service\_id $2 --publicurl 'http://'"$MASTER"':8774/v2/%(tenant\_id)s' --adminurl 'http://'"$MASTER"':8774/v2/%(tenant\_id)s' --internalurl 'http://'"$MASTER"':8774/v2/%(tenant\_id)s'  ;;  volume)  keystone endpoint-create --region $KEYSTONE\_REGION --service\_id $2 --publicurl 'http://'"$MASTER"':8776/v1/%(tenant\_id)s' --adminurl 'http://'"$MASTER"':8776/v1/%(tenant\_id)s' --internalurl 'http://'"$MASTER"':8776/v1/%(tenant\_id)s'  ;;  image)  keystone endpoint-create --region $KEYSTONE\_REGION --service\_id $2 --publicurl 'http://'"$MASTER"':9292/v1' --adminurl 'http://'"$MASTER"':9292/v1' --internalurl 'http://'"$MASTER"':9292/v1'  ;;  object-store)  if [ $SWIFT\_MASTER ]; then  keystone endpoint-create --region $KEYSTONE\_REGION --service\_id $2 --publicurl 'http://'"$SWIFT\_MASTER"':8080/v1/AUTH\_%(tenant\_id)s' --adminurl 'http://'"$SWIFT\_MASTER"':8080/v1' --internalurl 'http://'"$SWIFT\_MASTER"':8080/v1/AUTH\_%(tenant\_id)s'  else  keystone endpoint-create --region $KEYSTONE\_REGION --service\_id $2 --publicurl 'http://'"$MASTER"':8080/v1/AUTH\_%(tenant\_id)s' --adminurl 'http://'"$MASTER"':8080/v1' --internalurl 'http://'"$MASTER"':8080/v1/AUTH\_%(tenant\_id)s'  fi  ;;  identity)  keystone endpoint-create --region $KEYSTONE\_REGION --service\_id $2 --publicurl 'http://'"$MASTER"':5000/v2.0' --adminurl 'http://'"$MASTER"':35357/v2.0' --internalurl 'http://'"$MASTER"':5000/v2.0'  ;;  ec2)  keystone endpoint-create --region $KEYSTONE\_REGION --service\_id $2 --publicurl 'http://'"$MASTER"':8773/services/Cloud' --adminurl 'http://'"$MASTER"':8773/services/Admin' --internalurl 'http://'"$MASTER"':8773/services/Cloud'  ;;  esac  }  for i in compute volume image object-store identity ec2; do  id=`mysql -h "$MYSQL\_HOST" -u "$MYSQL\_USER" -p"$MYSQL\_PASSWORD" "$MYSQL\_DATABASE" -ss -e "SELECT id FROM service WHERE type='"$i"';"` || exit 1  create\_endpoint $i $id  done |

**附件二：安装错误处理**

**报错一**

|  |
| --- |
| root@ubuntu:~# keystone-manage db\_sync  Traceback (most recent call last):  File "/usr/bin/keystone-manage", line 28, in <module>  cli.main(argv=sys.argv, config\_files=config\_files)  File "/usr/lib/python2.7/dist-packages/keystone/cli.py", line 148, in main  return run(cmd, (args[:1] + args[2:]))  File "/usr/lib/python2.7/dist-packages/keystone/cli.py", line 134, in run  return CMDS[cmd](argv=args).run()  File "/usr/lib/python2.7/dist-packages/keystone/cli.py", line 36, in run  return self.main()  File "/usr/lib/python2.7/dist-packages/keystone/cli.py", line 57, in main  driver.db\_sync()  File "/usr/lib/python2.7/dist-packages/keystone/identity/backends/sql.py", line 135, in db\_sync  migration.db\_sync()  File "/usr/lib/python2.7/dist-packages/keystone/common/sql/migration.py", line 50, in db\_sync  current\_version = db\_version()  File "/usr/lib/python2.7/dist-packages/keystone/common/sql/migration.py", line 66, in db\_version  return db\_version\_control(0)  File "/usr/lib/python2.7/dist-packages/keystone/common/sql/migration.py", line 72, in db\_version\_control  CONF.sql.connection, repo\_path, version)  File "<string>", line 2, in version\_control  File "/usr/lib/python2.7/dist-packages/migrate/versioning/util/\_\_init\_\_.py", line 159, in with\_engine  return f(\*a, \*\*kw)  File "/usr/lib/python2.7/dist-packages/migrate/versioning/api.py", line 250, in version\_control  ControlledSchema.create(engine, repository, version)  File "/usr/lib/python2.7/dist-packages/migrate/versioning/schema.py", line 139, in create  table = cls.\_create\_table\_version(engine, repository, version)  File "/usr/lib/python2.7/dist-packages/migrate/versioning/schema.py", line 180, in \_create\_table\_version  if not table.exists():  File "/usr/lib/python2.7/dist-packages/sqlalchemy/schema.py", line 549, in exists  self.name, schema=self.schema)  File "/usr/lib/python2.7/dist-packages/sqlalchemy/engine/base.py", line 2274, in run\_callable  conn = self.contextual\_connect()  File "/usr/lib/python2.7/dist-packages/sqlalchemy/engine/base.py", line 2340, in contextual\_connect  self.pool.connect(),  File "/usr/lib/python2.7/dist-packages/sqlalchemy/pool.py", line 210, in connect  return \_ConnectionFairy(self).checkout()  File "/usr/lib/python2.7/dist-packages/sqlalchemy/pool.py", line 371, in \_\_init\_\_  rec = self.\_connection\_record = pool.\_do\_get()  File "/usr/lib/python2.7/dist-packages/sqlalchemy/pool.py", line 697, in \_do\_get  con = self.\_create\_connection()  File "/usr/lib/python2.7/dist-packages/sqlalchemy/pool.py", line 174, in \_create\_connection  return \_ConnectionRecord(self)  File "/usr/lib/python2.7/dist-packages/sqlalchemy/pool.py", line 256, in \_\_init\_\_  self.connection = self.\_\_connect()  File "/usr/lib/python2.7/dist-packages/sqlalchemy/pool.py", line 316, in \_\_connect  connection = self.\_\_pool.\_creator()  File "/usr/lib/python2.7/dist-packages/sqlalchemy/engine/strategies.py", line 80, in connect  return dialect.connect(\*cargs, \*\*cparams)  File "/usr/lib/python2.7/dist-packages/sqlalchemy/engine/default.py", line 280, in connect  return self.dbapi.connect(\*cargs, \*\*cparams)  File "/usr/lib/python2.7/dist-packages/MySQLdb/\_\_init\_\_.py", line 81, in Connect  return Connection(\*args, \*\*kwargs)  File "/usr/lib/python2.7/dist-packages/MySQLdb/connections.py", line 187, in \_\_init\_\_  super(Connection, self).\_\_init\_\_(\*args, \*\*kwargs2)  sqlalchemy.exc.OperationalError: (OperationalError) (1045, "Access denied for user 'keystone'@'ubuntu' (using password: YES)") None None |

**报错一：处理**

增加普通用户后，执行：

|  |
| --- |
| mysql> use mysql  mysql> delete from user where user='';  mysql> flush privileges; |

意思是删除匿名用户。

**报错二**

|  |
| --- |
| root@ubuntu:/etc/nova# nova boot --flavor 1 --image 2dba2c9c-21c1-4ccd-a1b9-d62a0c5d12a0 --key\_name key1 superfrobnicator  ERROR: Invalid key\_name provided. (HTTP 400 |

**报错二：处理**

|  |
| --- |
| 1: 创建密钥  # ssh-keygen  一路回车，就可以了。  ## 2：上传密钥到数据库  nova keypair-add --pub\_key ~/.ssh/id\_rsa.pub key1 |

**报错三**

RabbitMQ运行失败解决方法

|  |
| --- |
| Error:  Restarting rabbitmq-server: RabbitMQ is not running  Best way is to tell rabbitmq to not to look for hostname. You can do that using rabbitmq config file.  edit config, create it if it does not exist |

**报错三：处理**

|  |
| --- |
| vim /etc/rabbitmq/rabbitmq.conf  vim /etc/rabbitmq/rabbitmq-env.conf (在新版的rabbitMQ中编辑此文件)  Add following entries:  NODENAME=rabbit@localhost  NODE\_IP\_ADDRESS=127.0.0.1 |

Paipai 测试