第六章　使用 Kolla 部署 Pike 版本的 OpenStack-多节点云平台

本节所讲内容：

6.1　准备 openstack 多结点实验环境

6.1.1　准备硬件环境

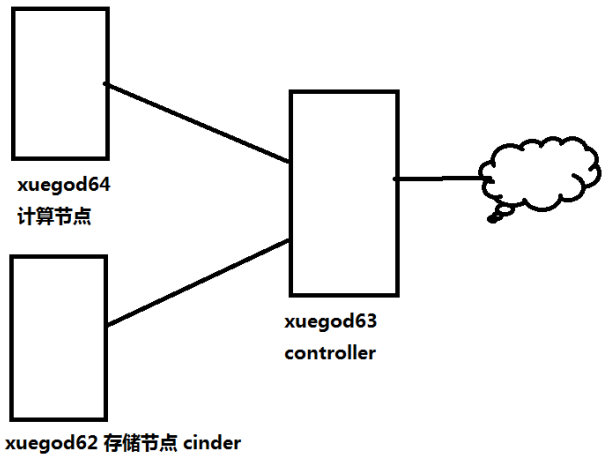
新创建 3 台虚拟机，分别作为 controller 节点，compute 节点，storage 节点。其中 controller 节点 2 张网卡，compute、storage 节点各 1 张网卡。操作系统为 centos7.5

注：controller 节点 也作为安装结点

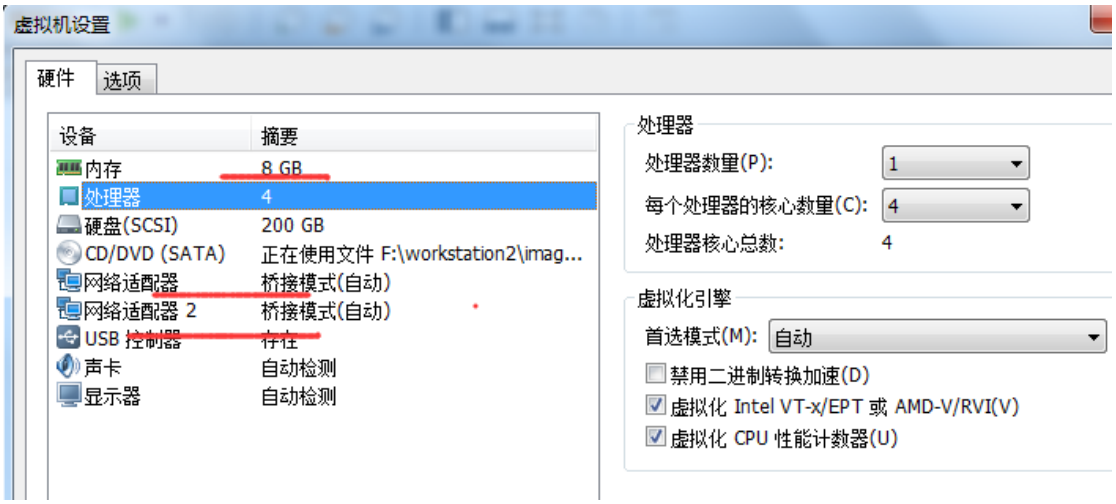
主机名	IP	角色	内存	网卡类型
xuegod63	192.168.1.63	controller 节点	8G	ens33 和 ens38 都桥接
xuegod64	192.168.1.64	compute 节点	4G	ens33 桥接
xuegod62	192.168.1.62	storage 节点	4G	ens33 桥接

注 :每个主机的 ens33 网卡作为内部管理 openstack 的网络和 web 界面的网络接口。 controller 节点的 ens38 网卡作为 对外的 网络。 compute 和 storage 一张即可，因为不需要 tunnel 网络，直接使用 1 个网卡。

实验拓扑图：



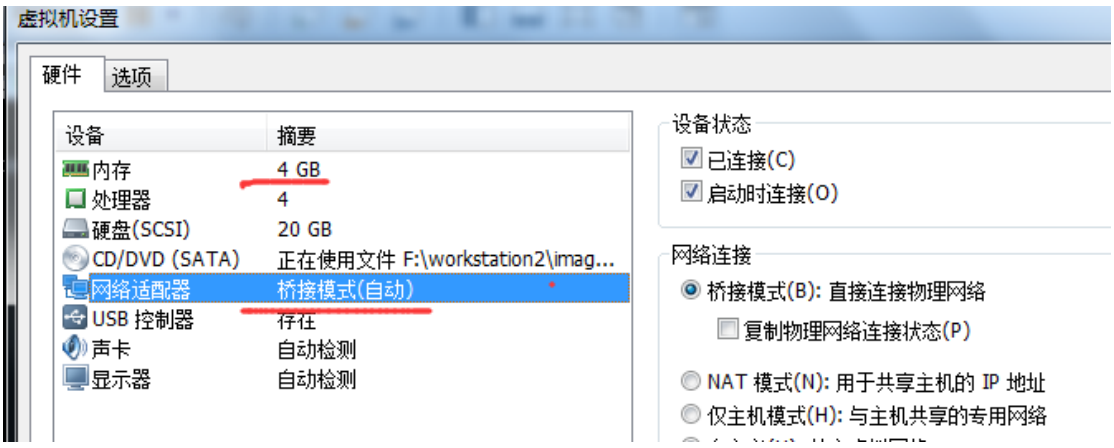
xuegod63 再添加 1 个网卡：两个网卡都是桥接



准备 xuegod62 存储结点环境

只用一个网卡，storage 节点新加一块磁盘，作为 cinder 的 lvm 后端。内存使用 4G

xuegod64 computer 结点，内存改成 4G，只用一个网卡



6.1.2 linux 系统环境配置

3 台机器开始做初始化配置， 以一下步骤需要在每一台机器上操作

1、关闭 Selinux 和防火墙

[root@xuegod63 ~]# vim /etc/selinux/config

SELINUX=disabled

[root@xuegod63 ~]# reboot　　#如果原来的系统开着 selinux ,那么需要重启 ,才能关闭 selinux

2、关闭 Firewalld

[root@xuegod63 ~]# systemctl stop firewalld &&　systemctl disable firewalld　&& systemctl status firewalld

[root@xuegod62 ~]# systemctl stop firewalld &&　systemctl disable firewalld　&& systemctl status firewalld

[root@xuegod64 ~]# systemctl stop firewalld &&　systemctl disable firewalld　&& systemctl status firewalld

3、安装 Epel 源

[root@xuegod63 ~]# mv /opt/CentOS-* /etc/yum.repos.d/　#这个在大家的机器上不需要执行。 因为我的 centos 自带的 yum 源配置文件，被移走了

[root@xuegod63 ~]# yum install epel-release -y

[root@xuegod62 ~]# yum install epel-release -y

[root@xuegod64 ~]# yum install epel-release -y

4、配置 Hostname

[root@xuegod63 ~]# echo xuegod63.cn > /etc/hostname

[root@xuegod62 ~]# echo xuegod62.cn > /etc/hostname

[root@xuegod64 ~]# echo xuegod64.cn > /etc/hostname

5、配置/etc/hosts

[root@xuegod63 ~]# vim /etc/hosts

127.0.0.1　　localhost localhost.localdomain localhost4 localhost4.localdomain4

::1　　localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.1.63 xuegod63.cn xuegod63

192.168.1.64 xuegod64.cn xuegod64

192.168.1.62 xuegod62.cn xuegod62

[root@xuegod63 ~]# scp /etc/hosts 192.168.1.64:/etc/

[root@xuegod63 ~]# scp /etc/hosts 192.168.1.62:/etc/

注：hosts 文件中的短主机名，给 rabbitmq 使用的。 rabbitmq 服务会使用短主机域名

6、同步时间

[root@xuegod63 ~]# yum install ntp -y　&& systemctl enable ntpd.service　&& systemctl start ntpd.service

[root@xuegod62 ~]# yum install ntp -y　&& systemctl enable ntpd.service　&& systemctl start ntpd.service

[root@xuegod64 ~]# yum install ntp -y　&& systemctl enable ntpd.service　&& systemctl start ntpd.service

7、配置 xuegod63 上 pip 软件包源，方便快速下载 python 库（这一步很重要）

[root@xuegod63 ~]# mkdir ~/.pip

[root@xuegod63 ~]# vim /root/.pip/pip.conf　#写入下以内容

[global]

index-url = http://mirrors.aliyun.com/pypi/simple/
[install]
trusted-host=mirrors.aliyun.com

6.1.2 配置 xuegod63 网卡信息

生成 ens38 的网卡配置文件

```
[root@xuegod63 ~]#cd /etc/sysconfig/network-scripts/
[root@xuegod63 network-scripts]# cp ifcfg-ens33 ifcfg-ens38
[root@xuegod63 network-scripts]# vim ifcfg-ens38
TYPE="Ethernet"
BOOTPROTO="none"
NAME="ens38"
DEVICE="ens38"
ONBOOT="yes"
```

检查一下 ens33 网卡配置文件，发现在新添加网卡，重启系统后，启动协议变了，现在改回来。

```
[root@xuegod63 network-scripts]# vim ifcfg-ens33
```

改：BOOTPROTO="dhcp"

为：BOOTPROTO="none"

```
[root@xuegod63 network-scripts]# service  network restart
```

查看网络：

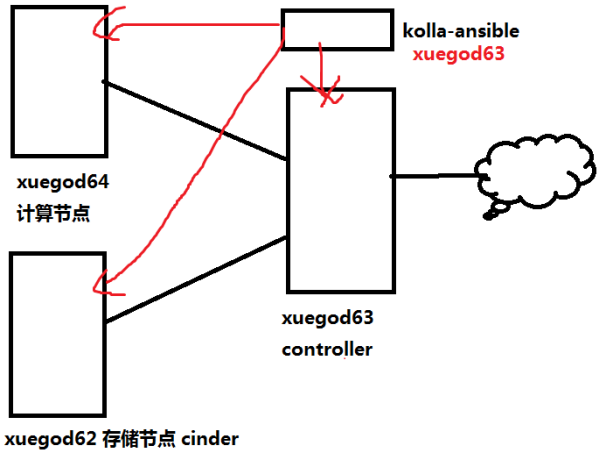
```
[root@xuegod63 network-scripts]# ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500

        inet 192.168.1.63  netmask 256.256.255.0  broadcast 192.168.1.255

            。。。
ens38: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500 #不用配置 IP
            inet6 fe80::20c:29ff:fe49:1fe2  prefixlen 64  scopeid 0x20<link>
```

6.1.3 在 3 台机器上安装基础包和 docker 服务

1、安装基础包和必要的一些系统工具，只在 xuegod63 上操作，因为 xuegod63 是部署节点，这些包是安装 kolla-ansible 必需要的。 单纯的 controller 节点 ,只需要安装一个 docker 服务就可以了，因为 controller 需要的软件包，都在 docker 镜像中包括了，不需要再安装软件的。

注：xuegod63 充当了两个角色： kolla-ansible 部署节点， 和 controller 节点

拓扑图如下：



[root@xuegod63 network-scripts]# yum install python-devel libffi-devel gcc openssl-devel git python-pip -y

[root@xuegod63 ~]# pip install -U pip #升级一下 pip，不然后，后期安装会报警告

[root@xuegod63 ~]# yum install -y yum-utils device-mapper-persistent-data lvm2

[root@xuegod63 ~]# yum install ansible #安装 ansible

2、添加 docker yum 源并安装 docker

关闭 libvirtd 服务：

[root@xuegod63 ~]# systemctl stop libvirtd.service && systemctl disable libvirtd.service && systemctl status libvirtd.service

[root@xuegod62 ~]# systemctl stop libvirtd.service && systemctl disable libvirtd.service && systemctl status libvirtd.service

[root@xuegod64 ~]# systemctl stop libvirtd.service && systemctl disable libvirtd.service && systemctl status libvirtd.service

如果有的话，卸载旧的 Docker，否则可能会不兼容：

[root@xuegod63 ~]# yum remove docker docker-io docker-selinux python-docker-py

[root@xuegod62 ~]# yum remove docker docker-io docker-selinux python-docker-py

[root@xuegod64 ~]# yum remove docker docker-io docker-selinux python-docker-py

因为咱们 centos 默认安装的是 docker 版本，而 kolla 使用的是 docker-ce 版本。

添加 docker-ce yum 源配置文件并安装 docker

[root@xuegod63 ~]# yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo

[root@xuegod62 ~]# yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo

[root@xuegod64 ~]# yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo

更新并安装 Docker-CE 社区版本

[root@xuegod63 ~]# yum -y install docker-ce

[root@xuegod62 ~]# yum -y install docker-ce

[root@xuegod64 ~]# yum -y install docker-ce

开启 Docker 服务

[root@xuegod63 ~]# systemctl start docker && systemctl enable docker && systemctl status docker
[root@xuegod62 ~]# systemctl start docker && systemctl enable docker && systemctl status docker
[root@xuegod64 ~]# systemctl start docker && systemctl enable docker && systemctl status docker

3、设置 3 台服务器的 docker volume 卷挂载方式和 docker 镜像加速器

[root@xuegod63 ~]# mkdir /etc/systemd/system/docker.service.d
[root@xuegod63 ~]# tee /etc/systemd/system/docker.service.d/kolla.conf << 'EOF'
[Service]
MountFlags=shared
EOF

设置 docker 镜像加速器

[root@xuegod63 ~]# vim /etc/docker/daemon.json
{
"registry-mirrors": ["https://e9yneuy4.mirror.aliyuncs.com"]
}

注：如果需要使用自己的本地私有仓库，写成如下：

{
"registry-mirrors": ["https://e9yneuy4.mirror.aliyuncs.com"]
"insecure-registries": ["192.168.1.63:4000"]
}

重启相关服务

[root@xuegod63 ~]# systemctl daemon-reload
[root@xuegod63 ~]# systemctl enable docker && systemctl restart docker && systemctl status docker

xuegod62 设置 volume 挂载方式

[root@xuegod62 ~]# mkdir /etc/systemd/system/docker.service.d
[root@xuegod62 ~]# tee /etc/systemd/system/docker.service.d/kolla.conf << 'EOF'
[Service]
MountFlags=shared
EOF

设置 docker 镜像加速器

[root@xuegod62 ~]# vim /etc/docker/daemon.json
{
"registry-mirrors": ["https://e9yneuy4.mirror.aliyuncs.com"]
}

重启相关服务

[root@xuegod62 ~]# systemctl daemon-reload
[root@xuegod62 ~]# systemctl enable docker && systemctl restart docker && systemctl status docker

xuegod64 设置 volume 挂载方式

[root@xuegod64 ~]# mkdir /etc/systemd/system/docker.service.d

[root@xuegod64 ~]# tee /etc/systemd/system/docker.service.d/kolla.conf << 'EOF'
[Service]
MountFlags=shared
EOF

设置 docker 镜像加速器

[root@xuegod64 ~]# vim /etc/docker/daemon.json
{
  "registry-mirrors": ["https://e9yneuy4.mirror.aliyuncs.com"]
}

重启相关服务

[root@xuegod64 ~]# systemctl daemon-reload
[root@xuegod64 ~]# systemctl enable docker && systemctl restart docker && systemctl status docker

6.1.4 xuegod62 storage 配置 cinder 存储信息

[root@xuegod62 ~]# ls /dev/sdb
/dev/sdb
[root@xuegod62 ~]# pvcreate /dev/sdb
  Physical volume "/dev/sdb" successfully created.
[root@xuegod62 ~]# vgcreate cinder-volumes /dev/sdb #创建一个名字为 cinder-volumes 的卷组，给后期 cinder 使用
  Volume group "cinder-volumes" successfully created
[root@xuegod62 ~]# systemctl enable lvm2-lvmetad.service
[root@xuegod62 ~]# vgs
  VG   #PV #LV #SN Attr   VSize   VFree
  cinder-volumes   1   0   0 wz--n- <20.00g <20.00g

到此 3 台机器的基础软件包环境已经安装好。

6.2 安装 kolla-ansible

1、安装 kolla-ansible

注：kolla-ansible 可以实现使用 ansible 自动安装 docker

[root@xuegod63 ~]# pip install kolla-ansible

注：官方推荐部署环境使用 pip install kolla-ansible 方式来安装 kolla-ansible

https://docs.openstack.org/kolla-ansible/latest/user/quickstart.html #官方安装手册

2、复制 kolla-ansible 的相关配置文件

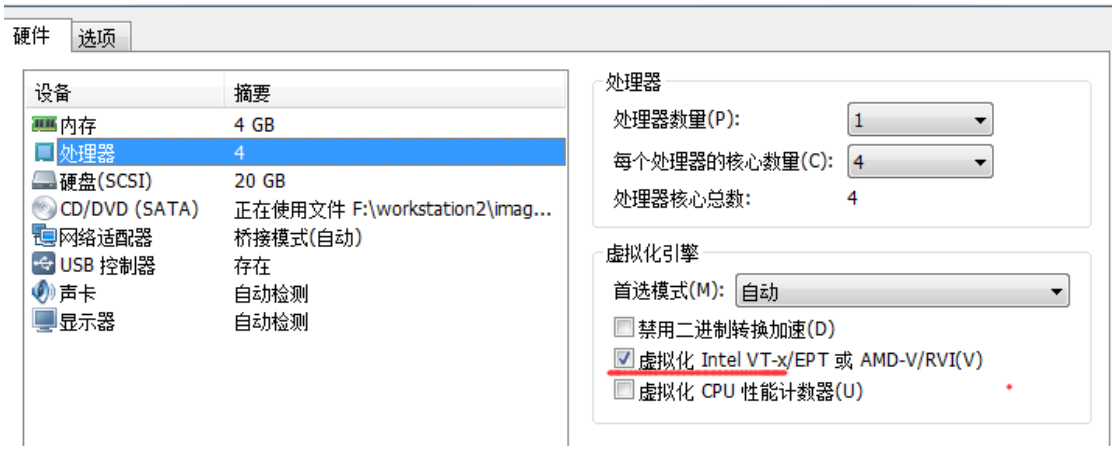
[root@xuegod63 ~]# cp -r /usr/share/kolla-ansible/etc_examples/kolla /etc/
[root@xuegod63 ~]# cp /usr/share/kolla-ansible/ansible/inventory/* /etc/kolla/
[root@xuegod63 kolla-ansible]# ls /etc/kolla/
all-in-one globals.yml multinode passwords.yml

注：all-in-one #安装单节点 openstack 的 ansible 自动安装配置文件，就是 ansible 主机清单文件
  multinode #安装多节点 openstack 的 ansible 自动安装配置文件，就是 ansible 主机清单文件
  globals.yml #部署 openstack 的自定义配置文件

passwords.yml　#openstack 中各个服务的密码

4、修改虚拟机类型为 qemu

注 :如果是在虚拟机里装 kolla ,希望可以在虚拟机中再启动虚拟机 ,那么你需要把 virt_type=qemu，默认是 kvm。如果 vmware 开了“虚拟化 Intel VT”功能，就不用写这个了。



[root@xuegod63 kolla-ansible]# mkdir -p /etc/kolla/config/nova

[root@xuegod63 kolla-ansible]# cat << EOF > /etc/kolla/config/nova/nova-compute.conf

[libvirt]

virt_type=qemu

cpu_mode = none

EOF

6.3　自定义 kolla-ansible 安装 openstack 的相关配置文件

1、自动生成 openstack 各服务的密码文件

[root@xuegod63 kolla-ansible]# kolla-genpwd

[root@xuegod63 ~]# vim /etc/kolla/passwords.yml

改：162 keystone_admin_password: HsPbEQHxTqmewKYNoRPpIOyQNdEYpHy36OX67TG3

为：keystone_admin_password: 123456

注：这是登录 Dashboard，admin 使用的密码，你可以根据自己需要进行修改。

2、编辑 /etc/kolla/globals.yml 自定义 openstack 中部署事项

[root@xuegod63 ~]# vim /etc/kolla/globals.yml #配置 openstack 安装中的参数

改：15 #kolla_base_distro: "centos"　　#选择下载的镜像为基于 centos 版本的镜像

为： kolla_base_distro: "centos"

改 :18 #kolla_install_type: "binary"　#去了前面的#号 ,使用 yum 安装二进制包安装 ,源码安装，指的是使用 git clone 源码安装

为：18 kolla_install_type: "binary"

改：21 #openstack_release: ""

为：openstack_release: "pike"　　#指定安装 pike 版本的 openstack，后期下载的 openstack 相关的 docker 镜像的 tag 标记也都为 pike

改：24 #node_custom_config: "/etc/kolla/config"　　#配置文件的位置

为：24 node_custom_config: "/etc/kolla/config"　#去了前面的#号，因为我对 nova 节点做了自定义。如：我自定义 virt_type 虚拟化类型为 qemu，默认是 kvm。

```
[root@xuegod63 ~]# cat /etc/kolla/config/nova/nova-compute.conf
[libvirt]
virt_type=qemu
cpu_mode = none
```

改：31 kolla_internal_vip_address: "10.10.10.254"

为：31 kolla_internal_vip_address: "192.168.1.63"　# 我们没有启用高可用，所以这里的 IP 可以和 ens33 一样，也可以独立写一个和 ens33 同网段的 IP。

注：如果配置了高可用，这里要使用一个没被占用的 IP。这个 IP 是搭建 HA 高可用的浮动 IP。 此 IP 将由 keepalived 管理以提供高可用性，应设置为和 network_interface ens33 同一个网段的地址。

改：85 #network_interface: "eth0"

为： network_interface: "ens33"　　#设置 OpenStack 所以节点内部通信使用的网络接口。这是 openstack 内部多个管理类型网络的默认接口。这里我们以 ens33 作为内部通信网络

改： 把前面的#号去了

```
 89 #kolla_external_vip_interface: "{{ network_interface }}"
 90 #api_interface: "{{ network_interface }}"
 91 #storage_interface: "{{ network_interface }}"
 92 #cluster_interface: "{{ network_interface }}"
 93 #tunnel_interface: "{{ network_interface }}"
 94 #dns_interface: "{{ network_interface }}"
```

为：

```
 89 kolla_external_vip_interface: "{{ network_interface }}"
 90 api_interface: "{{ network_interface }}"
 91 storage_interface: "{{ network_interface }}"
 92 cluster_interface: "{{ network_interface }}"
 93 tunnel_interface: "{{ network_interface }}"
 94 dns_interface: "{{ network_interface }}"
```

注：这样所有内部通信网络都走 network_interface ，即 ens33

改：100 #neutron_external_interface: "eth1"

为：100 neutron_external_interface: "ens38"　# ens38 作为外部网络，所需的第二个接口专用于 Neutron 外部（或公共）网络，可以是 vlan 或 flat，取决于网络的创建方式。 此接口应在没有 IP 地址的情况下处于活动 如果不是，云主机实例将无法访问外部网络。 所以 ens38 不能有 IP，只要网卡启动着，就可以了。

改：150 #enable_cinder: "no"

为：150 enable_cinder: "yes"　　#删除最前面的#号，并改 no 为 yes。 启用 cinder

改：155 #enable_cinder_backend_lvm: "no"

为：155 enable_cinder_backend_lvm: "yes" #删除最前面的#号，并改 no 为 yes。cinder 后端用 lvm

改：167 #enable_haproxy: "yes"

为：167　enable_haproxy: "no"　#删除最前面的#号，并改 yes 为 no。关闭高可用

改：319 #cinder_volume_group: "cinder-volumes"

为：319 cinder_volume_group: "cinder-volumes"　#取消前面的#号，这个卷组的名字是我们在 xuegod62 上创建的

```
[root@xuegod62 ~]# vgs
  VG             #PV #LV #SN Attr   VSize   VFree
  cinder-volumes   1   0   0 wz--n- <20.00g <20.00g
```

6.4　开始基于 kolla-ansible 安装 openstack 私有云

6.4.1　生成 SSH Key，并授信本节点：

[root@xuegod63 ~]# ssh-keygen

[root@xuegod63 ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@xuegod63

[root@xuegod63 ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@xuegod62

[root@xuegod63 ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@xuegod64

注：ssh-copy-id 复制公钥时，后面要写主机名，不要写 IP。因为后期 ansible 自动安装节点，主机清单中写的是主机名不是 IP。

6.4.2　配置 multinode 多结点清单文件

[root@xuegod63 ~]# vim /etc/kolla/multinode　#修改成以下红色标记内容

```
# These initial groups are the only groups required to be modified. The
# additional groups are for more control of the environment.
[control]
# These hostname must be resolvable from your deployment host
xuegod63
# The above can also be specified as follows:
#control[01:03]     ansible_user=kolla

# The network nodes are where your l3-agent and loadbalancers will run
# This can be the same as a host in the control group
[network]
xuegod63
# inner-compute is the groups of compute nodes which do not have
# external reachability
[inner-compute]

# external-compute is the groups of compute nodes which can reach
# outside
```

[external-compute]
xuegod64

[compute:children]
inner-compute
external-compute

[monitoring]
xuegod63

When compute nodes and control nodes use different interfaces,
you need to comment out "api_interface" and other interfaces from the global
s.yml
and specify like below:
#compute01 neutron_external_interface=eth0 api_interface=em1 storage_interface
=em1 tunnel_interface=em1

[storage]
xuegod62

[deployment]
xuegod63

注：配置下面的内容不需要修改。里面的内容很好理解，表示一个控制节点 controller，网络节点也安装到控制节点，一个计算节点 compute ，一个存储节点 storage，后面的部分不用修改

6.4.3 开始部署 OpenStack

1、对主机进行预部署检查：

[root@xuegod63 kolla]# kolla-ansible -i /etc/kolla/multinode prechecks

报错：

fatal: [xuegod62]: FAILED! => {"changed": false, "cmd": ["/usr/bin/python", "-c", "import docker; print docker.__version__"],

。。。

, "ImportError: No module named docker"], "stdout": "", "stdout_lines": []}

```
fatal: [xuegod62]: FAILED! => {"changed": false, "cmd": ["/usr/bin/python", "-c", "i
mport docker; print docker.__version__"], "delta": "0:00:00.138095", "end": "2018-08
-19 00:51:19.870011", "failed_when_result": true, "msg": "non-zero return code", "rc
": 1, "start": "2018-08-19 00:51:19.731916", "stderr": "Traceback (most recent call
last):\n  File \"<string>\", line 1, in <module>\nImportError: No module named docke
r", "stderr_lines": ["Traceback (most recent call last):", "  File \"<string>\", lin
e 1, in <module>", "ImportError: No module named docker"], "stdout": "", "stdout_lin
es": []}
```

解决：在 xuegod62 和 xuegod64 上执行：

[root@xuegod64 ~]# pip uninstall docker

[root@xuegod64 ~]# pip uninstall docker-py

[root@xuegod64 ~]# pip install -U docker --ignore-installed requests

[root@xuegod63 ~]# kolla-ansible -i /etc/kolla/multinode prechecks #再执行，出现以下界面说明检测没有问题。

```
PLAY RECAP *********************************************************************
xuegod62                   : ok=10   changed=0    unreachable=0    failed=0
xuegod63                   : ok=57   changed=0    unreachable=0    failed=0
xuegod64                   : ok=19   changed=0    unreachable=0    failed=0
```

2、最后进入实际的 OpenStack 部署

[root@xuegod63 ~]# kolla-ansible -i /etc/kolla/multinode deploy #因为此时边下载各种 openstack 相关的镜像并部署 docker 实例 ,会比较慢。等待 30 分钟左右。就可以了。我配置了 docker 镜像加速结点，所以也会比较快

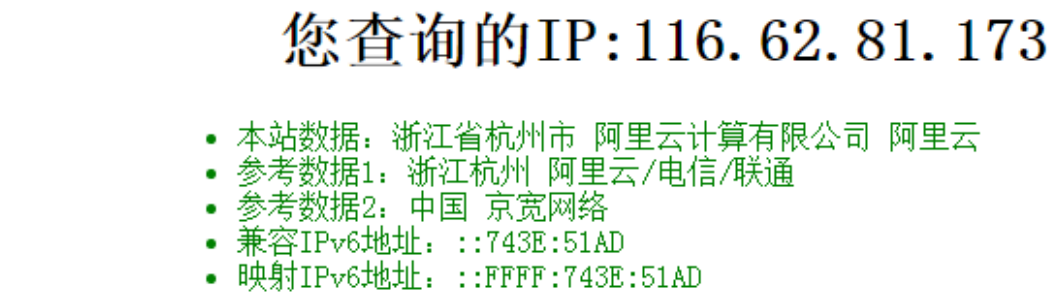
会先安装 xuegod63.

查看 xuegod63 上的网络连接：

[root@xuegod63 kolla-ansible]# netstat -antup

tcp 0 32 192.168.1.63:33052 116.62.81.173:443 LAST_ACK -

tcp 0 32 192.168.1.63:33116 116.62.81.173:443 LAST_ACK -



[root@xuegod63 kolla-ansible]# docker images #查看下载的镜像

[root@xuegod63 kolla-ansible]# docker ps #查看正在运行的 docker 实例

报错总结：

如果安装中运行 docker 实例报错。

```
RUNNING HANDLER [common : Initializing toolbox container using normal user] ********
fatal: [xuegod64]: FAILED! => {"changed": false, "cmd": ["docker", "exec", "-t", "ko
lla_toolbox", "/usr/bin/ansible", "--version"], "delta": "0:00:00.573891", "end": "2
018-08-19 11:04:28.065657", "msg": "non-zero return code", "rc": 126, "start": "2018
-08-19 11:04:27.491766", "stderr": "", "stderr_lines": [], "stdout": "OCI runtime ex
ec failed: exec failed: container_linux.go:348: starting container process caused \"
exec: \\\"/usr/bin/ansible\\\": stat /usr/bin/ansible: no such file or directory\":
unknown", "stdout_lines": ["OCI runtime exec failed: exec failed: container_linux.go
:348: starting container process caused \"exec: \\\"/usr/bin/ansible\\\": stat /usr/
bin/ansible: no such file or directory\": unknown"]}
```

解决：

[root@xuegod63 ~]# kolla-ansible -i /etc/kolla/multinode deploy #重新执行一下就可以了

```
PLAY RECAP *********************************************************************
xuegod62                   : ok=31   changed=11   unreachable=0    failed=0
xuegod63                   : ok=217  changed=114  unreachable=0    failed=0
xuegod64                   : ok=57   changed=25   unreachable=0    failed=0
```

到此安装成功。

3、验证部署

[root@xuegod63 ~]# kolla-ansible post-deploy -i /etc/kolla/multinode

```
PLAY RECAP *********************************************************************
localhost                  : ok=2    changed=0    unreachable=0    failed=0
```

这样就创建 /etc/kolla/admin-openrc.sh 文件

[root@xuegod63 kolla]# ll /etc/kolla/admin-openrc.sh

-rw-r--r-- 1 root root 323 8 月 10 17:58 /etc/kolla/admin-openrc.sh

[root@xuegod63 kolla]# cat /etc/kolla/admin-openrc.sh ##查看 openstack 登录帐号

export OS_PROJECT_DOMAIN_NAME=Default

export OS_USER_DOMAIN_NAME=Default

export OS_PROJECT_NAME=admin

export OS_TENANT_NAME=admin

export OS_USERNAME=admin

export OS_PASSWORD=123456 #帐号和密码

4、测试：http://192.168.1.63 访问成功。

里面没有任何东西

6.5 OpenStack 使用方法

6.6.1 安装 OpenStack client 端，方便后期使用命令行操作 openstack

[root@xuegod63 ~]# pip install python-openstackclient python-glanceclient python-neutronclient

注：软件包的作用：

python-openstackclient #openstack 客户端

python-glanceclient #操作 glance 镜像存储命令

python-neutronclient #操作 openstack 网络相关的命令

报错：已经安装 PyYAML 3.10，但是我们需要 PyYAML

Found existing installation: PyYAML 3.10

Cannot uninstall 'PyYAML'. It is a distutils installed project and thus we cannot accurately determine which files belong to it which would lead to only a partial uninstall.

[root@xuegod63 ~]# pip install PyYAML --ignore-installed PyYAML #忽略安装的 PyYAML，进行安装

[root@xuegod63 ~]# pip install python-openstackclient #再次安装

报错：

Found existing installation: ipaddress 1.0.16

Cannot uninstall 'ipaddress'. It is a distutils installed project and thus we cannot accurately determine which files belong to it which would lead to only a partial uninstall.

[root@xuegod63 ~]# pip install ipaddress --ignore-installed ipaddress

[root@xuegod63 ~]# pip install python-openstackclient

[root@xuegod63 ~]# pip install python-neutronclient #

报错，提示已经安装 pyinotify

[root@xuegod63 ~]# pip install pyinotify --ignore-installed pyinotify

6.6.2 修改 init-runonce

init-runonce 是在 openstack 中快速创建一个云项目例子的脚本

[root@xuegod63 ~]# vim /usr/share/kolla-ansible/init-runonce #网络需要根据实际情况修改

改：

12 EXT_NET_CIDR='10.0.2.0/24'

13 EXT_NET_RANGE='start=10.0.2.150,end=10.0.2.199'

14 EXT_NET_GATEWAY='10.0.2.1'

为：

EXT_NET_CIDR='192.168.1.0/24'

EXT_NET_RANGE='start=192.168.1.230,end=192.168.1.240'

EXT_NET_GATEWAY='192.168.1.1'

注：192.168.1.0 的网络，就是我上面 ens38 接入的局域网中的地址，这个网络是通过局域网络中的路由器访问互联网。配置好这个，装完虚拟机就可以直接 ping 通。

6.6.3 使用 init-runonce 脚本创建一个 openstack 云项目

1、开始创建一个云项目

[root@xuegod63 ~]# source /etc/kolla/admin-openrc.sh #必须先加载这个文件，把文件中的环境变量加入系统中，才有权限执行下面的命令

[root@xuegod63 ~]# cd /usr/share/kolla-ansible

[root@xuegod63 kolla-ansible]# ./init-runonce #最后弹出以下

```
| vcpus                         | 8          |
+-------------------------------+------------+

Done.

To deploy a demo instance, run:

openstack server create \
    --image cirros \
    --flavor m1.tiny \
    --key-name mykey \
    --nic net-id=a60a94b3-d1da-44c3-9c35-7505e1411378 \
    demo1
[root@xuegod63 kolla-ansible]#
```

2、在 openstack 中创建一个虚拟机

[root@xuegod63 kolla-ansible]# openstack server create --image cirros --flavor m1.tiny --key-name mykey --nic net-id=a60a94b3-d1da-44c3-9c35-7505e1411378 demo1

注：a60a94b3-d1da-44c3-9c35-7505e1411378 这个是前面弹出的 ID

6.6 查看创建好的 openstack 项目中的信息和云主机网络连通性

查看网络信息：

[root@xuegod63 ~]# source /etc/kolla/admin-openrc.sh #要读一下这个环境变量配置文件。不然后，后期在执行命令时，会报如下错：

Missing value auth-url required for auth plugin password

[root@xuegod63 ~]# openstack router list #查看路由信息

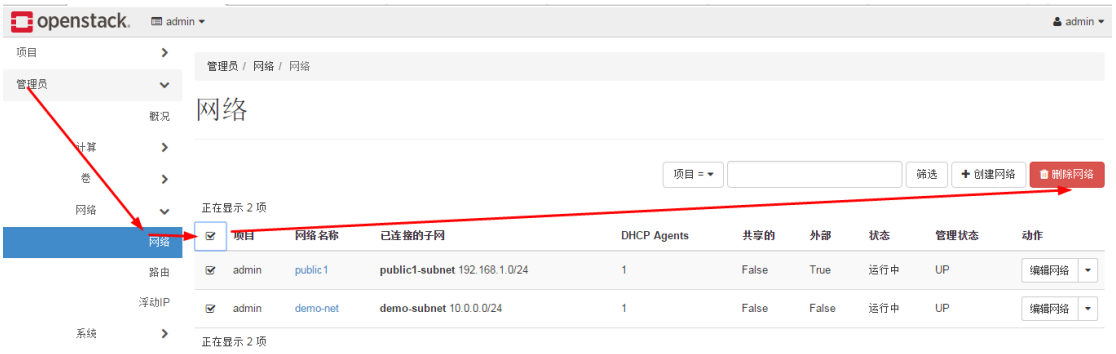
[root@xuegod63 ~]# openstack router show demo-router #查看 demo-router 路由信息

[root@xuegod63 ~]# openstack network list #查看网络信息

[root@xuegod63 ~]# openstack server show demo1 #查看名字为 demo1 的虚拟机信息

把创建的实例云主机，路由，网络都删除，一会我们自己手动创建

删除网络时，要在“管理员”菜单下删除



5.7 实战-通过命令行来创建自己的网络拓扑图

1、首先 source openers.sh 脚本，该脚本中是一些环境变量:

运行该脚本，即可通过命令行来管理云资源了：

[root@xuegod63 ~]# source /etc/kolla/admin-openrc.sh

2、创建对外的公网，名字：public

[root@xuegod63 ~]# openstack network create --external --provider-physical-network physnet1 --provider-network-type flat public

3、给 public 网络添加子网：

[root@xuegod63 ~]# openstack subnet create --no-dhcp --allocation-pool 'start=192.168.1.230,end=192.168.1.240' --network public --subnet-range 192.168.1.0/24 --gateway 192.168.1.1 public-subnet

4、创建私有网络：

[root@xuegod63 ~]# openstack network create --provider-network-type vxlan demo-net

给私有网络添加子网：

[root@xuegod63 ~]# openstack subnet create --subnet-range 10.0.0.0/24 --network demo-net --gateway 10.0.0.1 --dns-nameserver 8.8.8.8 demo-subnet

5、给外网和私网之间添加路由：

[root@xuegod63 ~]# openstack router create demo-router

[root@xuegod63 ~]# openstack router add subnet demo-router demo-subnet

[root@xuegod63 ~]# openstack router set --external-gateway public demo-router

6、通过下面的命令可以查询刚刚所建的网络信息：

[root@xuegod63 ~]# neutron net-list

7、登到 dashboard 上面去看 network topology：

至此，使用命令行创建的网络拓扑结束。

网络拓扑必须在命令行下运行，在 web 界面创建的网络拓扑图，上不了外网。因为在网页上无法设置桥接到物理网络上。命令行下有这一步骤：

```
echo Configuring neutron.
openstack network create --external --provider-physical-network physnet1 \
    --provider-network-type flat public1
    --subnet-range ${EXT_NET_CIDR} --gateway ${EXT_NET_GATEWAY} public1-subnet
```

创建一个台云主机及其他操作，在 web 界面执行就可以了。


创建实例
详情
源 *
实例类型 *
网络 *
网络接口
安全组
密钥对
请提供实例的主机名，欲部署的可用区域和数量。 增大数量以创建多个同样配置的实
实例名称 *
web1
可用域
nova
数量 *
1

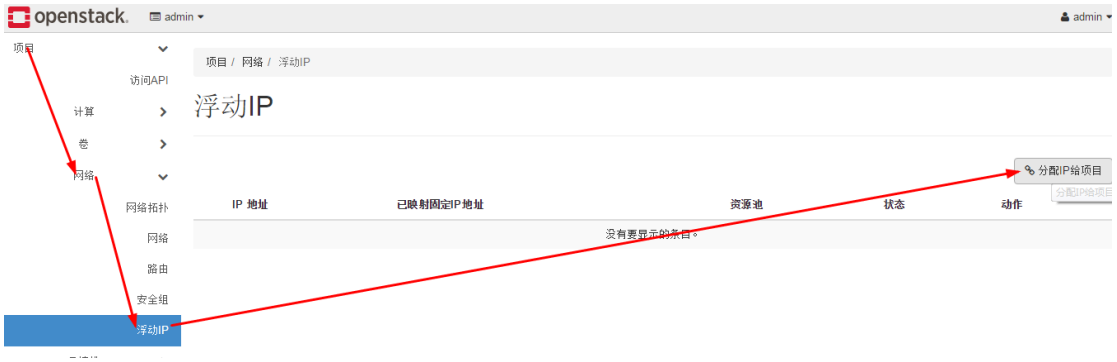

创建实例
详情
源 *
实例类型 *
网络 *
网络接口
安全组
密钥对
配置
服务器组
scheduler hint
元数据
实例的源是用来创建实例的模板。 可以使用一个镜像、一个实例的快照（镜像快照）、一个卷或一个卷快照（如果启用这个功能）。 您也可以通过创建一个新卷来选择使用具有持久性的存储。
选择源
镜像
创建新卷
是
否
卷大小 (GB) *
1
删除实例时删除卷
是
否
已分配
名称
已更新
大小
类型
可见性
从以下可选项中选择一项
可用 1
选择一个
点击这里过滤
名称
已更新
大小
类型
可见性
cirros
8/19/18 3:13 PM
12.67 MB
qcow2
公有

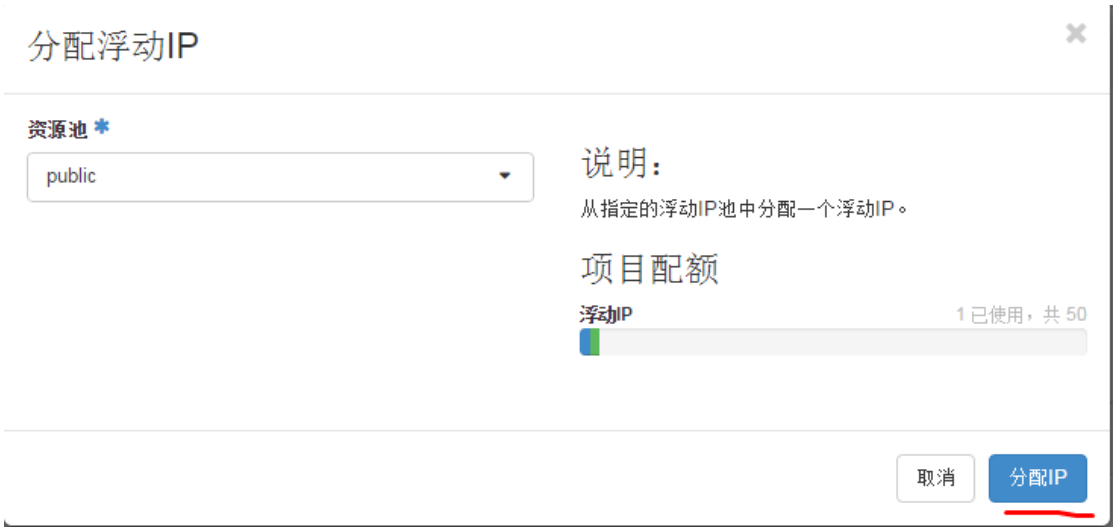

创建实例
详情
源
实例类型 *
网络 *
网络接口
安全组
密钥对
类型管理实例的计算、内存和存储容量的大小。
已分配
名称
虚拟内核
内存
磁盘总计
根磁盘
临时磁盘
公有
从以下可选项中选择一项
可用 5
选择一个
点击这里过滤
名称
虚拟内核
内存
磁盘总计
根磁盘
临时磁盘
公有
m1.tiny
1
512 MB
1 GB
1 GB
0 GB
是

安全组和密钥对，使用默认的就可以了

点创建。

分配浮动 IP 地址：





给云主机绑定浮动 IP

开始测试：

[root@xuegod63 ~]# ping 192.168.1.235 #可以 ping 通

PING 192.168.1.235 (192.168.1.235) 56(84) bytes of data.

64 bytes from 192.168.1.235: icmp_seq=1 ttl=63 time=16.5 ms

直接在物理机上进入云主机

[root@xuegod63 ~]# ssh cirros@192.168.1.235

$

$ ping baidu.com

PING baidu.com (220.181.57.216): 56 data bytes

64 bytes from 220.181.57.216: seq=0 ttl=50 time=18.559 ms

64 bytes from 220.181.57.216: seq=1 ttl=50 time=13.400 ms

总结：

- **6.1 Kolla 概述和 openstack 所有结点 linux 系统初始配置**
- **6.2 openstack 所有结点 linux 系统初始配置**
- **6.2 安装 kolla-ansible**
- **6.3 自定义 kolla-ansible 安装 openstack 的相关配置文件**
- **6.4 开始基于 kolla-ansible 安装 openstack 私有云**
- **6.5 OpenStack 使用方法**
- **6.6. 查看创建好的 openstack 项目中的信息和云主机网络连通性**
- **5.7 实战-通过命令行来创建自己的网络拓扑图**