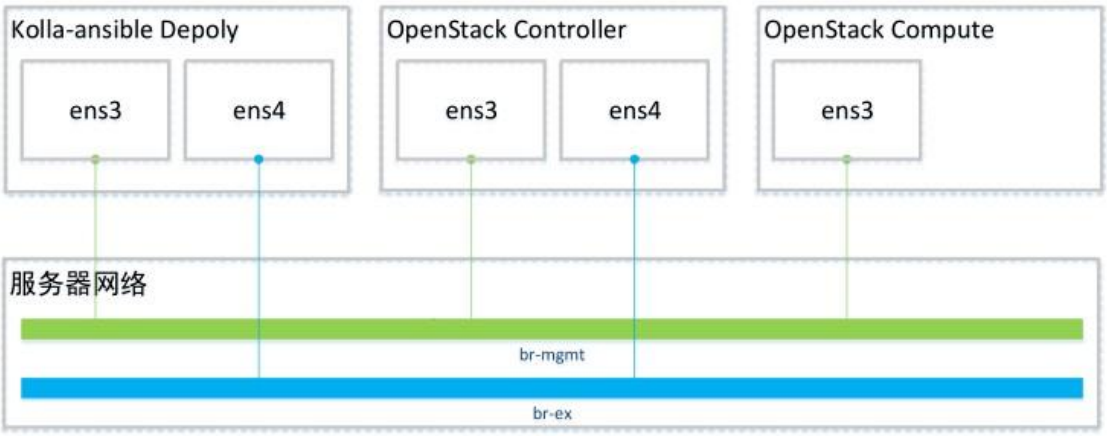
Ubuntu 20.04 kolla-ansible 安装 Openstack　Yoga 版

一、基础环境

角色	操作系统	硬件配置	说明
Control& Deploy	Ubuntu20.04 Mini	CPU:4核 磁盘:40GB 内存:8GB 网卡:ens3(内网) ens4(外网)	控制节点兼做部署节点
Compute	Ubuntu20.04 Mini	CPU:4核 磁盘:40GB 内存:8GB 网卡:ens3(内网) ens4(外网)	计算节点
Storage	Ubuntu20.04 Mini	CPU:4核 磁盘1:40GB，磁盘2：20GB 内存:8GB 网卡:ens3(内网)。	存储节点

注意：你的计算机中网卡的名字未必与文档中相同，请用 ip a 命令查看网卡名字并做相应的替换。

网络配置



主机名	网络地址
controller	192.168.128.131/24(内网) 192.168.1.100/24(外网，安装时可以不配 IP)
computer	192.168.128.132/24(内网) 192.168.1.101/24(外网，安装时可以不配 IP))
storager	192.168.128.133/24(内网) 192.168.1.102/24(外网，安装时可以不配 IP))

二、部署环境准备

1、创建虚拟机

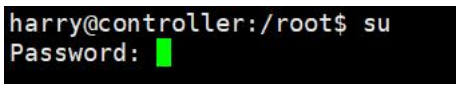
按上述配置要求创建虚拟机，并安装 Ubuntu20.04 操作系统（略）。

2、添加网卡，每个虚拟机两块网卡

配置网卡的 IP 地址，在 yaml 文件中配置，不可以使用网卡的别名，只能使用 ens33 这样的名字，可以使用 ip a 命令查看网卡的名字。

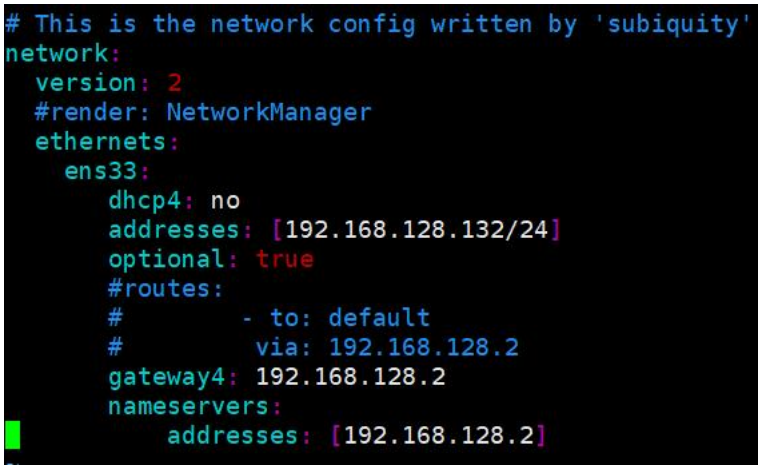
实验时建议以 root 用户操作，进入 ubuntu 以后通过以下命令切换到 root 账户。

```
su
Password:                #此处输入 root 的密码
```



静态 IP 地址配置如下例所示：

root@computer:~# vim /etc/netplan/00-installer-config.yaml



配置完成后要执行以下命令，令配置生效。

```
netplan apply
```

检查配置文件是否正确，并应用配置结果。

注意：配置文件要符合 python 的排版格式。

3、设置计算机名，并修改计算机的/etc/hosts 文件。

设置主机名,，分别在三台计算机上执行以下三条命令

```
hostnamectl set-hostname --static controller # 第一台
hostnamectl set-hostname --static computer # 第二台
hostnamectl set-hostname --static storager # 第三台
```

#在三台计算机上执行下面命令

```
vim /etc/hosts
```

在 hosts 文件最后添加以下内容：

```
192.168.128.131 controller
192.168.128.132 computer
192.168.128.133 storager
```

#测试三台计算机之间是否互通

```
ping computer
ping controller
ping storager
```

4、更新 apt-get 源步骤：

（1）备份源列表

```
sudo cp /etc/apt/sources.list /etc/apt/sources.list.bak
```

（2）命令行打开 sources.list 文件

```
sudo vim /etc/apt/sources.list
```

（3）修改 sources.list 文件

将源文件内容全部注释，并添加阿里源或者清华源（二选一）

阿里源：

```
deb http://mirrors.aliyun.com/ubuntu/ focal main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ focal main restricted universe multiverse

deb http://mirrors.aliyun.com/ubuntu/ focal-security main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ focal-security main restricted universe multiverse

deb http://mirrors.aliyun.com/ubuntu/ focal-updates main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ focal-updates main restricted universe multiverse

deb http://mirrors.aliyun.com/ubuntu/ focal-proposed main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ focal-proposed main restricted universe multiverse

deb http://mirrors.aliyun.com/ubuntu/ focal-backports main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ focal-backports main restricted universe multiverse
```

清华源：

```
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal-updates main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal-security main restricted universe multiverse
multiverse
```

（4）存盘退出

（5）更新源

```
sudo apt-get update
```

（6）更新软件

```
sudo apt-get dist-upgrade
```

```
sudo apt-get upgrade
```

特别说明：根据实验遇到的问题，更新源后安装 python3 虚拟环境时会报错，可以还原回原来的源。如果不行，换回原来的源。

5、安装 ssh 和 Git

apt -y install ssh git

6、安装依赖项

apt -y install python3-dev libffi-dev gcc libssl-dev

7、修改 ssh 配置文件，允许 root 远程登录

vim /etc/ssh/sshd_config

修改 PermitRootLogin，允许 root 用户通过密码登录

Authentication:

LoginGraceTime 2m

PermitRootLogin yes #此行是重点

StrictModes yes

MaxAuthTries 6

MaxSessions 10

8、重启 ssh

systemctl restart ssh

systemctl enable ssh

9、# 关闭防火墙

ufw disable

特别说明：以上操作需要在三台计算机都要执行。建议采用虚机时可以先配置完一台，然后克隆出另外两台虚机，然后修改一下 IP、主机名和网卡的 MAC 地址。

10、在 Storage 计算机上执行以下操作。

storage 节点上创建一个 VG

```
apt -y install lvm2
pvcreate /dev/sdb
vgcreate cinder-volumes /dev/sdb
```

以下操作均在 controller 上执行

三、安装 python-venv 虚拟环境依赖

1、安装 python3-venv

```
apt install python3-venv
```

2、创建虚拟环境并激活它

```
mkdir /path && mkdir /path/to #  也可以自定路径
python3 -m venv /path/to/venv
source /path/to/venv/bin/activate
```

3、pypi 换源

```
pip config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple
```

4、安装 Ansible

```
pip install 'ansible>=4,<6'
```

四、安装 Kolla-Ansible

1、安装 kolla

pip install git+https://opendev.org/openstack/kolla-ansible@stable/yoga

上面是原版命令，建议使用 jihulab.com 克隆一份

```
pip install git+https://jihulab.com/james-curtis/kolla-ansible@stable/yoga
```

2、创建文件夹/etc/kolla

```
sudo mkdir -p /etc/kolla
sudo chown $USER:$USER /etc/kolla
```

3、复制 globals.yml 和 passwords.yml 到 /etc/kolla

```
# 如果你使用了自定义路径，记得替换/path/to
cp -r /path/to/venv/share/kolla-ansible/etc_examples/kolla/* /etc/kolla
```

4、复制清单文件

```
cp /path/to/venv/share/kolla-ansible/ansible/inventory/* .
```

五、安装 Ansible Galaxy 依赖

1、安装 Kolla-Ansible-Collections 依赖

```
kolla-ansible install-deps
```

2、docker 换源 && 指定 docker 版本到 20.10.*

这里不换源是可以的。但是截止 docker 23.0.0 发布时间之后再来使用 kolla-ansible，首次安装没问题，后续如果要添加删除组件的话就会报错。原因是 docker 删除了容器的 `KernelMemory` 属性，但是 kolla 需要读取，导致数组下标不存在报错。请参阅：

https://blog.csdn.net/qq_35485875/article/details/128877591。

```
sed -i.bak 's/https:\/\/download\.docker\.com/https:\/\/mirrors\.tuna\.tsinghua\.edu\.cn\/docker-ce/g' /root/.ansible/collections/ansible_collections/openstack/kolla/roles/baremetal/defaults/main.yml
sed -i.bak 's/_package: "docker-ce"/_package: "docker-ce=5:20.10*"/g' /root/.ansible/collections/ansible_collections/openstack/kolla/roles/baremetal/defaults/main.yml
```

3、配置 Ansible

```
mkdir /etc/ansible
```

```
echo "[defaults]
host_key_checking=False
pipelining=True
forks=100" > /etc/ansible/ansible.cfg
```

六、准备初始化配置

如果单机部署，你也可以跳过此步骤，然后使用 all-in-one 清单，后续如果再增加机器可以切换回到 multinode 清单，只需要注意修改好机器主机名就行。

对于 all-in-one 虚拟环境中的场景，将以下内容添加到 all-in-one 清单的最开头。

```
localhost ansible_python_interpreter=python
```

七、修改配置文件

```
vim ~/multinode
```

文件内容修改如下：（特别注意：下面的参数是本实验环境的参数，如果与你的环境参数不符记住修改）

```
# 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
controller    hostname=controller ansible_user=root ansible_password="换成自己的密码"

# 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]
# network01
# network02
```

```
# when you specify group_name:children, it will use contents of group specified.
[network:children]
control

[compute]
      computer    hostname=computer ansible_user=root ansible_password=”换成自己的密码”
[compute:children]
   compute

[monitoring]
   control

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

  [storage]
      storager    hostname=storager ansible_user=root ansible_password=”换成自己的密码”
# storage01
#[storage:children]
#storager

[deployment]
localhost ansible_connection=local become=true
# 此文件较长，其余内容默认
```

八、测试各个节点的连通性

ansible -i multinode all -m ping

出现以下提示表示正确。

```
-----------------------------------------
localhost | SUCCESS => {
      "ansible_facts": {
            "discovered_interpreter_python": "/usr/bin/python3"
      },
      "changed": false,
      "ping": "pong"
}
manager | SUCCESS => {
      "changed": false,
      "ping": "pong"
}
```

```
compute1 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
storage1 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

九、配置 kolla 文件

```
# 创建 kolla 配置文件
mkdir -p /etc/kolla
chown -R $USER:$USER /etc/kolla
cp -r ~/venv3/share/kolla-ansible/etc_examples/kolla/* /etc/kolla
ls /etc/kolla
# 两个文件 globals.yml  passwords.yml
# 生成密码
kolla-genpwd
# 生成的密码会自动写入/etc/kolla/passwords.yml
# 编写配置模板
cat << EOF > /etc/kolla/globals.yml
---
kolla_base_distro: "ubuntu"
kolla_install_type: "source" #使用基于源代码的 image
openstack_release: "yoga" #该配置项最好与 kolla-ansible 分支版本保持一致
kolla_internal_vip_address: "192.168.128.250" #  找一个网段内没有占用的 ip
# docker_registry: "10.0.0.203:5000" #指定私有 registry
```

network_interface: "ens33" # 内部 openstack 管理网段

neutron_external_interface: "ens4"

enable_haproxy: "yes"

enable_cinder: "yes" # 这一个是启动 cinder 块存储并使用我们的 VG

enable_cinder_backend_lvm: "yes"

nova_compute_virt_type: "qemu" #使用虚拟机部署时，该配置项必须改为 qemu，默认值为 kvm

EOF

注意：（1）以上算作一行命令，从 cat 到 EOF。复制时一起复制。

（2）网卡的名字要与自己网卡的名字一致。

（3）kolla_internal_vip_address 是未来访问 openstack 管理页面的 IP。

十、开始安装

环境安装，这一步会自动安装 docker

kolla-ansible -i ./multinode bootstrap-servers

参数预检查

kolla-ansible -i ./multinode prechecks

下载 openstack 各个组件容器镜像

kolla-ansible -i ./multinode pull

部署

kolla-ansible -i ./multinode deploy

遇到报错，销毁已安装的环境

kolla-ansible -i ./multinode destroy --yes-i-really-really-mean-it

安装时间大约 40 分钟左右。

十二、安装 CLI 命令端组件

```
# CLI 客户端安装
pip install python-openstackclient -c https://releases.openstack.org/constraints/upper/master
# 生成 RC 文件
kolla-ansible post-deploy
. /etc/kolla/admin-openrc.sh

# 查看已经启动的服务
openstack service list
```

十三、登录界面

openstack 的 Dashboard 组件有两个，horizon（已经存在于多个版本中）和 skyline（正在测试，没有官方容器）。kolla 安装过程中会自动帮我们安装一个全功能的 horizon 组件。

登录网址：http://vip/auth/login/?next=/

比如本案例中就是 http://1192.168.128.250/auth/login/?next=/

但是我们还缺 admin 的密码。

执行：

```
cat /etc/kolla/passwords.yml |grep keystone_admin_password
```

查看 admin 密码。

```
(venv) root@controller:~# cat /etc/kolla/passwords.yml |grep keystone_admin_password
keystone_admin_password: mJiyOMnRpzAzjlSKFxLaoTpRY8iCZIIEqGO0vTR0
```

红字后面的字符是 admin 登录密码。


不安全 | 192.168.128.250/auth/login/?next=/
openstack.
登录
用户名
admin
密码
登入