kolla部署openstack

配置免密登录

安装依赖包

```
yum -y install epel-release
yum -y install python-pip git
```

centos更换阿里云源

```
cp /etc/yum.repos.d/CentOS-Base.repo /etc/yum.repos.d/CentOS-Base.repo.bak
wget -O /etc/yum.repos.d/CentOS.repo http://mirrors.aliyun.com/repo/Centos-
7.repo

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

停止 libvirt 组件，否则后面安装会失败

```
systemctl stop libvirtd.service
systemctl disable libvirtd.service
```

拉取docker-ce.repo，然后才能安装上docker-ce

```
wget -O /etc/yum.repos.d/docker-ce.repo http://mirrors.aliyun.com/repo/Centos-
7.repo

yum -y install docker-ce
```

对docker进行配置，开启 Docker 的 shared mount 共享挂载功能

```
mkdir /etc/systemd/system/docker.service.d
tee /etc/systemd/system/docker.service.d/kolla.conf <<-'EOF'
[Service]
MountFlags=shared
```

如果安装zun组件，需要对docker进行进一步配置

https://docs.openstack.org/kolla-ansible/latest/reference/compute/zun-guide.html （容器计算）

https://docs.openstack.org/kolla-ansible/latest/reference/containers/kuryr-guide.html (容器网络)

```
vim /usr/lib/systemd/system/docker.service
添加如下信息
ExecStart= -H tcp://172.16.1.13:2375 -H unix:///var/run/docker.sock --cluster-
store=etcd://172.16.1.13:2379 --cluster-advertise=172.16.1.13:2375
```

为 Docker 配置访问私有仓库，修改 ExecStart 字段：(如果从docker hub上拉镜像不需要进行此步骤)

```
vim /usr/lib/systemd/system/docker.service

修改如下内容,[controller]字符串替换成对应的控制节点的IP
ExecStart=/usr/bin/dockerd --insecure-registry [controller]:4000
```

重启docker服务

```
systemctl daemon-reload
systemctl restart docker
```

下载python的docker sdk并升级到最新版

```
pip install docker
pip install -U docker
```

配置Docker的私有仓库 (queens版以后kolla不提供，只能自己拉 所有的包上传到本地仓库)

限于国内网络环境的问题，如果直接在线安装很容易失败，所以我们非常有必要建立一个本地仓 库。建立本地仓库所需要的OpenStack的各个组件的镜像，既可以通过 Kolla 进行 build ， 也可 以直接下载官网制作好的镜像。 运行 Registry 服务的容器，由于 5000 端口与 keystone 端口号冲突，所以我们需要修改其映射 到本地 4000 端口上:

```
docker run -d -v /opt/registry:/var/lib/registry -p 4000:5000 --restart=always -
-name registry registry:2

# 查看是否成功启动
docker ps
```

运行以上命令后，会自动在线拉取 registry:2 的镜像，如果因为网络环境不能够正常下载，也可 以在一个网络状况比较好的机器上先下载好，然后导出为 .tar 包，再在相应的机器上导入:

```
# 在网络好的机器上执行
docker pull registry:2
docker images
docker save -o registry_v2.tar registry:2

# 在目标机器上执行如下命令，再执行上面的命令块，将registry服务用容器跑起来
docker load -i ./registry_v2.tar
docker image
```

在浏览器中输入：[SERVICE_IP]:4000/v2/_catalog，如果能够进入，则说明已经成功安装了本地仓 库。

安装Kollaansible部署工具

此过程只需要在部署机上面执行，kollaansible既可以使用 pip 来安装，也可以通过 git 源码安装，本次通过源码安装：　下载源码包，一定要下载对应的版本，部署的是Q版，所以下载Q版的kollaansible

```
cd /opt
git clone http://git.trystack.cn/openstack/kolla-ansible -b stable/rocky
```

安装kolla-ansible：

```
cd  kolla-ansible
pip install ./     (或者python setup.py develop)
```

复制配置文件和inventory文件：

```
cd /opt/kolla-ansible/
cp -r etc/kolla /etc/kolla
cp -r ansible/inventory /home/inventory/
```

如果是在虚拟机里装kolla，希望可以启动再启动虚拟机，那么你需要把virt_type=qemu，默认是kvm：

```
mkdir -p /etc/kolla/config/nova
cat << EOF > /etc/kolla/config/nova-compute.conf
[libvirt]
virt_type=qume
cpu_mode=none
EOF
```

生成密码文件：

```
kolla-genpwd
```

修改登录dashboard的admin用户的登录密码：

```
vim /etc/kolla/passwords.yml
# 修改下面的字段
keystone_admin_password: admin
```

如果安装zun组件，必须安装etcd和kuryr. 安装etcd过程中，kollaansible会出现etcd起不来的现 象，此时需要修改kollaansible文件

```
vim /usr/share/kolla-ansible/ansible/roles/etcd/defaults/main.yml
把
ETCD_LISTEN_CLIENT_URLS: "{{ internal_protocol }}://{{ api_interface_address }}:
{{ etcd_client_port }}"
修改为
ETCD_LISTEN_CLIENT_URLS: "{{ internal_protocol }}://0.0.0.0:{{ etcd_client_port
}}"
```

开始部署
由于我们采用的是多节点物理机部署，在部署之前，我们需要修改 globals.yml 和 multinode 两个配置文件（单节点的话修改的是allinone.yml的配置文件）。

```
vim /etc/kolla/globals.yml
```

编辑好了一个初版的globals.yml文件，请见hithub。

接下来编辑inventory文件，分为all-in-one和multinode文件。An inventory is an Ansible file where we specify hosts and the groups that they belong to. We can use this to define node roles and access credentials.

Kolla-Ansible comes with `all-in-one` and `multinode` example inventory files. The difference between them is that the former is ready for deploying single node OpenStack on localhost. If you need to use separate host or more than one node。

```
# 检查有没有错误，出现failed就要排错
kolla-ansible -i /home/inventory/all-in-one prechecks
或者
kolla-ansible -i /home/inventory/multinode prechecks
```

因为用到octavia，需要一些特别的配置，参考：https://www.lijiawang.org/posts/kolla-octavia.html

prechecks的时候可能会遇到octavia找不到证书的问题，参照上面的链接，进行如下操作：

```
git clone -b stable/rocky https://review.openstack.org/p/openstack/octavia
cd ./octavia
grep octavia_ca /etc/kolla/passwords.yml
# 输出类似下面
octavia_ca_password: mEUyBHLopKk5O1CX3OWRnPuiDmoP3I7eNQIQbC6z

sed -i 's/foobar/mEUyBHLopKk5O1CX3OWRnPuiDmoP3I7eNQIQbC6z/g'
bin/create_certificates.sh
./bin/create_certificates.sh cert $(pwd)/etc/certificates/openssl.cnf
```

会生成一个文件夹cert

```
ls -al ./cert
# 输出如下
total 48
drwxr-xr-x.  4 root root  271 Oct 14 21:59 .
drwxr-xr-x. 19 root root 4096 Oct 14 21:59 ..
-rw-r--r--.  1 root root 1294 Oct 14 21:59 ca_01.pem
-rw-r--r--.  1 root root  989 Oct 14 21:59 client.csr
-rw-r--r--.  1 root root 1704 Oct 14 21:59 client.key
-rw-r--r--.  1 root root 4405 Oct 14 21:59 client-.pem
-rw-r--r--.  1 root root 6109 Oct 14 21:59 client.pem
-rw-r--r--.  1 root root   71 Oct 14 21:59 index.txt
-rw-r--r--.  1 root root   21 Oct 14 21:59 index.txt.attr
-rw-r--r--.  1 root root    0 Oct 14 21:59 index.txt.old
drwxr-xr-x.  2 root root   20 Oct 14 21:59 newcerts
drwx------.  2 root root   31 Oct 14 21:59 private
-rw-r--r--.  1 root root    3 Oct 14 21:59 serial
-rw-r--r--.  1 root root    3 Oct 14 21:59 serial.old
```

将认证放到kolla部署节点上的/etc/kolla/config/octavia目录里，首先要先在/etc/kolla目录下创建一个octavia文件

```
mkdir /etc/kolla/config
mkdir /etc/kolla/config/octavia
# 将生成的cert里的文件复制进去
cp cert/{private/cakey.pem,ca_01.pem,client.pem}  /etc/kolla/config/octavia
```

解决这个问题可以继续执行prechecks的命令，看看还有没有其他错误，直到没有fail出现才开始执行deploy

```
kolla-ansible -i /home/inventory/multinode deploy -v
```

如果中间出现错误则解决错误，最后执行成功以后大概如图：

可以执行`docker images`看看下载了哪些镜像，执行`docker ps -a`查看容器是否正常运行。

如果没什么问题，可以在浏览器中输入你再`globals.yml`制定的`kolla_internal_vip_address`的IP地址来访问 dashboard 。用户名为：admin，登陆密码为你再 passwords.yml 中制定的`keystone_admin_password`字段的值，我们配置的是admin。

如果想要增加组件，可以直接修改 globals.yml 文件中的配置项，然后运行④中的命令。如果想要重新部署，可以使用如下命令清理环境，然后再次运行`kolla-ansible -i /home/inventory/multinode deploy`的命令。如果想清理环境重新来部署，执行命令`kolla-ansible -i /home/inventory/multinode destory`。会有一个提示你确认的过程，只需要按照它的提示，输入相应的命令选项确认即可。

> Note：在执行 destroy 命令之前一定要确保，当前OpenStack平台上没有存储镜像，也没有相应的 instance 和 volume ,也没有swift的存储空间，总而言之，OpenStack上面除了配置选项 外，不能存有任何额外的东西，否则会导致清理环境失败。这点一定要切记，非常重要！！！

验证部署并初始化

部署完成后，我们需要进行验证，会在 /etc/kolla 目录下生成 admin-openrc.sh 文件：

```
kolla-ansible post-deploy
```

安装OpenStack的client端：

```
pip install python-openstackclient
```

编辑初始化脚本文件 /usr/share/kolla-ansible/init-runonce :

```
# 我的配置如下，根据自己的网卡的配置情况，做出相应的修改
# This EXT_NET_CIDR is your public network,that you want to connect to the
internet via.
EXT_NET_CIDR='10.10.87.0/24'
EXT_NET_RANGE='start=10.10.87.100,end=10.0.2.199'
EXT_NET_GATEWAY='10.10.87.1'
```

运行初始化脚本，进行初始化：

```
source /etc/kolla/admin-openrc.sh
cd /usr/share/kolla-ansible/
./init-runonce
```

执行`./init-runonce`可能会出现一些错误，比如`ImportError: cannot import name decorate`，把对应的包安装上`pip install decorator`。这个文件要下载一个img文件，建议进入init-runonce脚本文件看下载地址什么，下载什么版本，下载到那个文件夹，用浏览器下载上传到对应的文件夹，再执行`./init-runonce`。

可以根据脚本最后运行完的提示运行如下命令，创建一个VM：

```
# To deploy a demo instance, run:

openstack server create \
    --image cirros \
    --flavor m1.tiny \
    --key-name mykey \
    --network demo-net \
    demo1
```

登陆dashboard，正常使用OpenStack。