

组网及说明

- 1、CloudOS2.0版本: 1139H06  
CloudOS2.0版本对应的CAS版本: 0526H11
- 2、CloudOS3.0 版本: 3106H01  
CloudOS3.0版本对应的CAS版本: 0530H11

配置步骤

一、CloudOS 2.0版本容器固化步骤

这里以修改cloudos-openstack镜像为例。

1、备份镜像

使用“docker ps | grep cloudos-openstack”查看关键字为cloudos-openstack容器。

```
[root@cloudosone ~]# docker ps | grep cloudos-openstack
fe908115e8fd        cloudos-openstack-compute:E1139H06          "/root/docker-opensta
About an hour ago   Up About an hour                               k8s_demon.40e78b1d_demonrc-b4
5kl_default_bca14297-f7c0-11e9-959d-0cda411d182f_ec88778f
4124f0bbf718        cloudos-openstack:E1139H06                  "/root/docker-opensta
5 hours ago        Up 5 hours                                       k8s_openstack.2e840874_ope
ackrc-9d5s7_default_f70389d8-f7a1-11e9-959d-0cda411d182f_8eadeea1
[root@cloudosone ~]# docker images | grep cloudos-openstack
```

图1 查找对应容器

查找需要修改的镜像文件，使用tag命令对标签为“E113906”的cloudos-openstack镜像进行备份，备份后的标签为E113906bak。输入“docker images | grep cloudos-openstack”查看到备份的镜像和原镜像大小相同。

```
[root@cloudosone ~]# docker images | grep cloudos-openstack
cloudos-openstack-compute          E1139H06
39235bd0949f        About an hour ago   1.846 GB
192.168.12.226:9999/cloudos-openstack-manila-share E1139H06
34ef868dcc49        9 months ago       730.6 MB
cloudos-openstack-manila-share     E1139H06
34ef868dcc49        9 months ago       730.6 MB
cloudos-openstack-compute          E1139H06_bak201910261515
ca8b3e41b7f0        9 months ago       1.799 GB
192.168.12.226:9999/cloudos-openstack E1139H06
d5545a14541d        9 months ago       2.146 GB
cloudos-openstack                  E1139H06
d5545a14541d        9 months ago       2.146 GB
[root@cloudosone ~]# docker tag cloudos-openstack:E1139H06 cloudos-openstack:E1139H06bak
[root@cloudosone ~]# docker images | grep cloudos-openstack
cloudos-openstack-compute          E1139H06
39235bd0949f        About an hour ago   1.846 GB
192.168.12.226:9999/cloudos-openstack-manila-share E1139H06
34ef868dcc49        9 months ago       730.6 MB
cloudos-openstack-manila-share     E1139H06
34ef868dcc49        9 months ago       730.6 MB
cloudos-openstack-compute          E1139H06_bak201910261515
ca8b3e41b7f0        9 months ago       1.799 GB
192.168.12.226:9999/cloudos-openstack E1139H06
d5545a14541d        9 months ago       2.146 GB
cloudos-openstack                  E1139H06
d5545a14541d        9 months ago       2.146 GB
cloudos-openstack                  E1139H06bak
d5545a14541d        9 months ago       2.146 GB
[root@cloudosone ~]#
```

图2 找到目标镜像，备份镜像

2、去除原镜像标签

在备份完镜像后，使用“docker rmi cloudos-openstack:E1139H06”去除原镜像的标签。命令查看验证原来镜像的标签已经被成功去除。

```
[root@cloudosone ~]# docker rmi cloudos-openstack:E1139H06
Untagged: cloudos-openstack:E1139H06
[root@cloudosone ~]# docker images | grep cloudos-openstack
cloudos-openstack-compute          E1139H06
39235bd0949f        About an hour ago   1.846 GB
192.168.12.226:9999/cloudos-openstack-manila-share E1139H06
34ef868dcc49        9 months ago       730.6 MB
cloudos-openstack-manila-share     E1139H06
34ef868dcc49        9 months ago       730.6 MB
cloudos-openstack-compute          E1139H06_bak201910261515
ca8b3e41b7f0        9 months ago       1.799 GB
192.168.12.226:9999/cloudos-openstack E1139H06
d5545a14541d        9 months ago       2.146 GB
cloudos-openstack                  E1139H06bak
d5545a14541d        9 months ago       2.146 GB
[root@cloudosone ~]#
```

图3 删除原镜像标签

3、制作新镜像

查询对应的容器，根据容器id，使用“docker exec -it [容器id] bash”命令进入容器。将/root/scripts/目录下的pre-install.sh拷贝至本地/opt/openstack-transfer。

```

c99498149470 3 minutes ago 2.196 GB
[root@cloudosone ~]# docker ps | grep openstack
fe908115e8fd cloudos-openstack-compute:E1193H06 "/root/docker-opensta
About an hour ago Up About an hour k8s_demon.40e78b1d_demonrc-b4
5k1_default_bca14297-f7c0-11e9-959d-0cda411d182f_ec68778f "/root/docker-opensta
4124f0bb718 d5845a14541d "/root/docker-opensta
5 hours ago Up 5 hours k8s_openstack.2e840874_openst
ackrc-9d5s7_default_f70389d8-f7a1-11e9-959d-0cda411d182f_8eadeea1
997b00a69b44 scr.io/google_containers/pause-amd64:3.0 "/pause"
5 hours ago Up 5 hours k8s_POD.c0da188_openstackrc-9
d5s7_default_f70389d8-f7a1-11e9-959d-0cda411d182f_bf83f08d
[root@cloudosone ~]# docker exec -it fe908115e8fd bash
[root@demonrc /]# ls
bin boot dev etc home lib lib64 lost+found media mnt opt proc root run sbin selinux srv sys tmp usr var
[root@demonrc /]# cd /root/scripts/
[root@demonrc scripts]# ls
install-cas-plugin.sh post-install.sh pre-install.py region-openstack.sh upgrade upgrade-packages.sh
openstack-install-yum.sh post-startup.sh pre-install.sh tools upgrade-openssl.sh
[root@demonrc scripts]#

```

图4 进入对应容器，找到目标文件

```

bin boot dev etc home lib lib64 lost+found media mnt opt proc root run sbin selinux srv sys tmp usr var
[root@demonrc /]# cd /root/scripts/
[root@demonrc scripts]# ls
install-cas-plugin.sh post-install.sh pre-install.py region-openstack.sh upgrade upgrade-packages.sh
openstack-install-yum.sh post-startup.sh pre-install.sh tools upgrade-openssl.sh
[root@demonrc scripts]# cp pre-install.sh /opt/openstack-transfer
[root@demonrc scripts]#

```

图5 将目标文件拷贝至本地

退出容器，在/root下新建文件夹，命名为guhua，并将刚才/opt/openstack-transfer目录下的pre-install.sh拷贝至root/guhua/下，再进行修改。

```

[root@cloudosone ~]# cd /opt/openstack-transfer/
[root@cloudosone openstack-transfer]# ls
pre-install.sh
[root@cloudosone openstack-transfer]# cd ..
[root@cloudosone opt]# cd ..
[root@cloudosone /]# cd /root
[root@cloudosone /]# ls
anaconda-ks.cfg anaconda-post-before-chroot.log certs keystore.key nginx.conf
anaconda-post-after-chroot.log anaconda-post-partition.log dockerReg keystore-pem.cer patch_clouds2.0
[root@cloudosone /]# mkdir guhua
[root@cloudosone /]# ls
anaconda-ks.cfg anaconda-post-before-chroot.log certs guhua keystore-pem.cer patch_clouds2.0
anaconda-post-after-chroot.log anaconda-post-partition.log dockerReg keystore.key nginx.conf
[root@cloudosone /]#

```

图6 将目标文件拷贝至/root/guhua/下

拷贝后，验证是否拷贝成功。

```

anaconda-post-after-chroot.log anaconda-post-partition.log dockerReg keystore.key nginx.conf
[root@cloudosone ~]# cp /opt/openstack-transfer/pre-install.sh /root/guhua/
[root@cloudosone ~]# cd guhua
[root@cloudosone guhua]# ls
pre-install.sh
[root@cloudosone guhua]#

```

图7 检验拷贝是否成功

通过“vi pre-install.sh”修改文件，使用“/openstack-cinfig”进行关键字搜索。

```

# /bin/bash
# Script for install MariaDB & RabbitMQ & Openstack Controller/Compute node. Including:
# Local repository configuration
# MariaDB
# MariaDB configuration
# RabbitMQ
# RabbitMQ configuration
# Keystone
# Glance
# Nova-Controller
# Horizon
# Cinder-Controller
# Neutron
# Nova-Compute
# Cinder-Volume

set -x

OLD_DIR=`pwd`
CURR_DIR=`dirname $0`
cd $CURR_DIR

##### input enviroment data #####
input inner and outer net IP
LOCAL_ADDR1=$(ifconfig | grep 'inet' | grep -v '127.0.0.1' | awk '(NR==1) { print $2}')
LOCAL_ADDR2=$(ifconfig | grep 'inet' | grep -v '127.0.0.1' | awk '(NR==2) { print $2}')
if [ ! $LOCAL_ADDR1 -a ! $LOCAL_ADDR2 ]; then
echo "Please configure your net ip correctly:"
exit 1
fi

#Public Network IP address of the host
OUTER_NET_IP=$LOCAL_ADDR1

#The Manage Network IP address of the host
/openstack-config

#####

```

图8 使用vi命令关键字搜索

搜索的第一次结果如图。

```
sed -i 's/ob_ip[ ]=[ ]*$/ob_ip=${HOSTNAME}.${SERVICE_SERVICE_HOST}/g' /etc/ftp_server.conf
sed -i 's/"RabbitHost[ ]=[ ]*$/RabbitHost=${RABBITMQ_SERVICE_SERVICE_HOST}/g' /etc/ftp_server.conf
sed -i 's/"pasv_address[ ]=[ ]*$/pasv_address=${VIP}/g' /etc/vsftpd/vsftpd.conf

##### config Nova_controller #####

openstack-config --set /etc/nova/nova.conf DEFAULT my_ip $LOCAL_IP
openstack-config --set /etc/nova/nova.conf DEFAULT scheduler_default_filters RetryFilter,AvailabilityZoneFilter,RamFilter,ComputeFilter,Compu
tiesFilter,ImagePropertiesFilter,ServerGroupAntiAffinityFilter,ServerGroupAffinityFilter
openstack-config --set /etc/nova/nova.conf vnc vncserver_listen $LOCAL_IP
openstack-config --set /etc/nova/nova.conf vnc vncserver_proxyclient_address $LOCAL_IP

openstack-config --set /usr/lib/systemd/system/openstack-nova-novncproxy.service Service StandardOutput null
openstack-config --set /usr/lib/systemd/system/openstack-nova-novncproxy.service Service StandardError null

##### Neutron #####

#create keystone neutron user
#vlan configure
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2_type_vlan network_vlan_ranges physnet1:"$VLAN_RANGE"
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2_type_vxlan vni_ranges "$VLAN_RANGE"
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2_type_flat flat_networks "$FLAT_NETWORKS"

fi
```

图9 第一次搜索结果

多次检索后，找到要修改的位置，将firewall\_type从CGSR修改为GATEWAY，将lb\_type 从CGSR修改为SERVICE\_CHAIN，将resource\_mode从CORE\_GATEWAY修改为NFV。

```
fi
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER firewall_type CGSR
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER lb_type CGSR
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER resource_mode CORE_GATEWAY
if [ "$upgradevalue" = "null" ] || [ -z "$upgradevalue" ];then
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER enable_security_group False
fi

openstack-config --set /etc/neutron/neutron.conf DEFAULT service_plugins h3c_vcfplugin,l3_router,h3c_l3_router_plugin,H3CL3
r,lbaas,vpnaaS

openstack-config --set /etc/neutron/neutron_lbaas.conf service_providers service_provider LOADBALANCER:H3C:h3c_vcfplugin
r,H3CLbaasPluginDriver:default
```

图10 pre-install.sh修改前

```
fi
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER enable_l3_vxlan "$l3vxlanmodel"
fi

openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER firewall_type GATEWAY
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER lb_type SERVICE_CHAIN
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER resource_mode NFV
if [ "$upgradevalue" = "null" ] || [ -z "$upgradevalue" ];then
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER enable_security_group False
fi
```

图11 pre-install.sh修改后

创建dockerfile文件（即图中的dockerfileopenstackbuild.txt）来新建镜像。

```
[root@cloudosone ~]# vi pre-install.sh
[root@cloudosone ~]# vi dockerfileopenstackbuild.txt
```

图12 创建dockerfile文件

这里的dockerfile规定了：FROM是指明当前新镜像是基于哪个镜像的；USER表示用户为root；COPY表示拷贝文件和目录到镜像中，这里是将当前目录下修改后的pre-install.sh拷贝至/root/scripts目录下。

```
FROM cloudos-openstack:E1193H06bak
USER root
COPY pre-install.sh /root/scripts/pre-install.sh

#####
```

图13 dockerfile修改内容

检查刚刚创建的dockerfile文件是否正确。

```
[root@cloudosone ~]# vi pre-install.sh
[root@cloudosone ~]# vi dockerfileopenstackbuild.txt
[root@cloudosone ~]# cat dockerfileopenstackbuild.txt
FROM cloudos-openstack:E1193H06bak
USER root
COPY pre-install.sh /root/scripts/pre-install.sh
[root@cloudosone ~]#
```

图14 检验dockerfile文件

通过“docker build”命令构建新的镜像，成功后显示“Successfully built [新镜像id]”。

```
Build an image from a Dockerfile
[root@cloudosone guhua]# docker build -t cloudos-openstack:E1139H06 -f dockerfileopenstackbuild.txt .
Sending build context to Docker daemon 19.46 kB
Step 1 : FROM cloudos-openstack:E1139H06bak
--> d5545a14541d
Step 2 : USER root
--> Running in 06acbf701ca3
--> f4aeld0a7e12
Removing intermediate container 06acbf701ca3
Step 3 : COPY pre-install.sh /root/scripts/pre-install.sh
--> 707951461a63
Removing intermediate container e6168d206c91
Successfully built 707951461a63
[root@cloudosone guhua]#
```

图15 新建镜像

查看是否新增镜像。

```
Successfully built 707951461a63
[root@cloudosone guhua]# docker images | grep cloudos-openstack
cloudos-openstack E1139H06 707951461a63 52 seconds ago 2.146 GB
cloudos-openstack E1139H06 39235bd0949f 4 hours ago 1.049 GB
192.168.12.226:9999/cloudos-openstack-manila-share E1139H06 34ef888dccc49 9 months ago 730.6 MB
cloudos-openstack-manila-share E1139H06 34ef888dccc49 9 months ago 730.6 MB
cloudos-openstack-compute E1139H06_bak201910261515 ca8b9e41b7f0 9 months ago 1.798 GB
192.168.12.226:9999/cloudos-openstack E1139H06 d5545a14541d 9 months ago 2.146 GB
cloudos-openstack E1139H06bak d5545a14541d 9 months ago 2.146 GB
[root@cloudosone guhua]#
```

图16 查看是否新增镜像

#### 4. 删除当前容器

查找当前openstack容器，获得容器名称“openstackrc”，并进行删除。

```
[root@cloudosone guhua]# /opt/bin/kubectl --server=127.0.0.1:8888 get rc
NAME          DESIRED  CURRENT  AGE
coreapi/rc    1        1        8h
demon/rc     1        1        4h
openstack/rc  1        1        8h
parametcd/rc 1        1        8h
portal/rc    1        1        8h
postgres/rc  1        1        8h
rabbitmq/rc  1        1        8h
rdb/rc       1        1        8h
webapp/rc    1        1        8h
[root@cloudosone guhua]#
```

图17 查找当前容器

检查容器是否已经删除成功，可以看到，已经没有“openstackrc”。

```
[root@cloudosone guhua]# /opt/bin/kubectl --server=127.0.0.1:8888 get pod -o wide
NAME          READY   STATUS    RESTARTS  AGE   IP             NODE
coreapi/rc-hj03x 1/1     Running   0          8h   10.101.58.8   192.168.12.226
demon/rc-b45kl  1/1     Running   0          4h   10.101.58.14  192.168.12.226
parametcd/rc-we0dj 1/1     Running   0          8h   10.101.58.6   192.168.12.226
portal/rc-p4aiq  1/1     Running   0          8h   10.101.58.10  192.168.12.226
postgres/rc-ulc7x 1/1     Running   0          8h   10.101.58.5   192.168.12.226
rabbitmq/rc-0o6e5 1/1     Running   0          8h   10.101.58.4   192.168.12.226
rdb/rc-qe10y     1/1     Running   0          8h   10.101.58.12  192.168.12.226
webapp/rc-8dg29 1/1     Running   0          8h   10.101.58.9   192.168.12.226
[root@cloudosone guhua]#
```

图18 检查容器删除是否成功

#### 5. 使用新镜像创建容器

重新创建容器前需要先检查openstack-rc.yaml文件中的image字段是否与上一步build出来的tag一致。

```
[root@cloudosone guhua]# cd /opt/bin/confFile/
[root@cloudosone confFile]# ls
coreapi-rc.yaml grafana-service.yaml nginx-service.yaml param-service.yaml rabbitmq-service.yaml webapp-service.yaml
coreapi-service.yaml kubedns-rc.yaml openstack-compute-rc.yaml portal-rc.yaml rdb-rc.yaml
db-install-rc.yaml kubedns-service.yaml openstack-rc.yaml portal-service.yaml rdb-service.yaml
db-install-service.yaml manila-share openstack-service.yaml postgresql-rc.yaml skydns-rc.yaml
demon-rc.yaml manila-share-rc.yaml param-rc-templ.yaml postgresql-service.yaml skydns-service.yaml
grafana-rc.yaml nginx-rc.yaml param-rc.yaml rabbitmq-rc.yaml webapp-rc.yaml
[root@cloudosone confFile]#
```

图19 找到openstack-rc.yaml文件

从图中可以看到，openstack-rc.yaml文件中的image字段与上一步build出来的tag一致，镜像信息为cloudos-openstack:E1139H06。

```

apiVersion: v1
kind: ReplicationController
metadata:
  name: openstackrc
spec:
  replicas: 1
  # selector identifies the set of Pods that this
  # replication controller is responsible for managing
  selector:
    app: openstack
  # podTemplate defines the 'cookie cutter' used for creating
  # new pods when necessary
  template:
    metadata:
      labels:
        # Important: these labels need to match the selector above
        # The api server enforces this constraint.
        app: openstack
    annotations:
      "pod.beta.kubernetes.io/hostname": openstack-server
    spec:
      containers:
      - name: openstack
        image: cloudos-openstack:E1139H06
        env:
        - name: ALL_PASS
          value: cloudos
        - name: VIP
          value: 192.168.12.226
        ports:
        - containerPort: 20
        - containerPort: 21
        - containerPort: 8080

```

图20 检验文件中镜像的tag

通过“/opt/bin/kubect| --server=127.0.0.1:8888 create -f /opt/bin/confFile/openstack-rc.yaml”创建新容器。

```

[root@cloudosone confFile]# vi openstack-rc.yaml
[root@cloudosone confFile]# /opt/bin/kubect| --server=127.0.0.1:8888 create -f /opt/bin/confFile/openstack-rc.yaml
replicationcontroller "openstackrc" created
[root@cloudosone confFile]#

```

图21 创建新容器

查看是否拉起openstack的docker， 以及其运行状态是否为正常的Running状态。

```

[root@cloudosone confFile]# vi openstack-rc.yaml
[root@cloudosone confFile]# /opt/bin/kubect| --server=127.0.0.1:8888 create -f /opt/bin/confFile/openstack-rc.yaml
replicationcontroller "openstackrc" created
[root@cloudosone confFile]# /opt/bin/kubect| --server=127.0.0.1:8888 get pod -o wide

```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE
coreapprc-hj03x	1/1	Running	0	8h	10.101.58.8	192.168.12.226
democr-b45kl	1/1	Running	0	4h	10.101.58.14	192.168.12.226
openstackrc-x1rfs	1/1	Running	0	21s	10.101.58.7	192.168.12.226
parametodrc-we9dj	1/1	Running	0	8h	10.101.58.8	192.168.12.226
portalcrc-p4aiq	1/1	Running	0	8h	10.101.58.10	192.168.12.226
postgresqlrc-ulc7x	1/1	Running	0	8h	10.101.58.5	192.168.12.226
rabbitmarc-0e8e5	1/1	Running	0	8h	10.101.58.4	192.168.12.226
rdbrc-qe1ly	1/1	Running	0	8h	10.101.58.12	192.168.12.226
webapprc-8dg29	1/1	Running	0	8h	10.101.58.9	192.168.12.226

图22 看容器是否拉起， 正常运行

## 6. 查看容器化操作是否成功

通过查找对应容器id进入cloudos-openstack容器， 检验对应的参数是否固化。

```

[root@cloudosone confFile]# docker ps | grep openstack
88a1df6f0c24 cloudos-openstack:E1139H06 "/root/docker-opensta" 3 minutes ago Up 3 minutes
k8s_openstack.2e84074_openstackrc-x1rfs_default_b
8a1c2a8-f7e9-11e9-959d-0cda411d192f_90d5e21b
f12fcd2cfab scr.io/soosle_containers/pause-awd84:3.0 "/pause" 3 minutes ago Up 3 minutes
k8s_POD.c0da188_openstackrc-x1rfs_default_b8a1c2a8
-f7e9-11e9-959d-0cda411d192f_b89d5c1c
fe908115e8fd cloudos-openstack-compute:E1139H06 "/root/docker-opensta" 4 hours ago Up 4 hours
k8s_demon.40e78b1d_demonrc-b45kl_default_bca14297-
f7e9-11e9-959d-0cda411d192f_es89770f
[root@cloudosone confFile]# docker exec -it 88a1df6f0c24 bash
[root@openstack-server /]#

```

图23 查找并进入目标容器

找到/root/scripts/下的pre-install.sh找到刚刚修改的位置， 可以看到， 参数和修改时一样， 即固化成功。

```

[root@openstack-server /]#
[root@openstack-server /]# cd /root/scripts/
[root@openstack-server scripts]# ls
create-task.txt openstack-auto-task.sh post-install.sh region-openstack.sh upgrade upgrade-opens1.sh
ldap-openstack.py openstack-install-yum.sh post-startup.sh region-openstack.sh_bak upgrade-db-to-mitaka.sh upgrade-vsftpd.sh
ldap-openstack.sh post-install2.sh pre-install.sh tools upgrade-httpd.sh wsgi-keystone.conf
[root@openstack-server scripts]# vi pre-install.sh

```

图24 找到目标文件

```

fi
# if nmcli added, for evpn solution
if [ "${vxlanmodel}" != "null" ]; then
  openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER enable_l3_vxlan "${l3_vxlanmodel}"
fi

openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER firewall_type GATEWAY
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER lb_type SERVICE_CHAIN
openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER resource_mode NFV
if [ "${upgradevalue}" = "null" ] || [ -z $upgradevalue ]; then
  openstack-config --set /etc/neutron/plugins/ml2/ml2_conf_h3c.ini VCFCONTROLLER enable_security_group False
fi

openstack-config --set /etc/neutron/neutron.conf DEFAULT service_plugins h3c_vcfplugin_l3_router_h3c_l3_router_plugin_H3C

```

图25 查看是否固化成功

## 二、CloudOS 3.0版本容器固化步骤

这里以修改cloudos-openstack-sahara镜像为例。

### 1、修改镜像文件

查找要修改的容器镜像，这里选择cloudos-openstack-sahara，进入该容器。

```
[root@single1 /]#  
[root@single1 /]# docker ps | grep cloudos-openstack-sahara  
e861ff5f5c5b    192.168.10.99:9999/cloudos-openstack-sahara@sha256:9df50d20e3cb464437bad5c881d3140ab6cb5e9f1948124cc49bc3ef832d97  
83          "/docker-entrypoint.s" 4 minutes ago      Up 4 minutes      k8s_cloudos-opensta  
ck-sahara_sahararc-t179d_default_19736af5-f7f5-11e9-9ffe-0cda411d52a1_0  
[root@single1 /]# docker exec -it e861ff5f5c5b bash  
[root@sahara-service /]#
```

图26 找到目标容器并进入

找到该容器的/root/scripts/目录下的health-check.sh，将该文件拷贝至本地/opt/openstack-transfer/sahara/目录下。

```
[root@sahara-service /]#  
[root@sahara-service /]# cd /root/scripts/  
[root@sahara-service scripts]# ls  
health-check.sh  pre-startup.sh  region-openstack.sh  upgrade          upgrade-openssl.sh  
post-startup.sh  pre-stop.sh       tools                upgrade-db-to-pike.sh  
[root@sahara-service scripts]# cd /opt/openstack-transfer/sahara/  
[root@sahara-service sahara]# ls  
[root@sahara-service sahara]# cd /root/scripts/  
[root@sahara-service scripts]# cp health-check.sh /opt/openstack-transfer/sahara/  
[root@sahara-service scripts]# ls  
health-check.sh  pre-startup.sh  region-openstack.sh  upgrade          upgrade-openssl.sh  
post-startup.sh  pre-stop.sh       tools                upgrade-db-to-pike.sh  
[root@sahara-service scripts]# cd /opt/openstack-transfer/sahara/  
[root@sahara-service sahara]# ls  
health-check.sh  
[root@sahara-service sahara]#
```

图27 找到目标文件，拷贝至本地

对该文件进行修改，如图中新增一行注释，写上“#hello”。

```
#!/bin/bash  
#hello  
source /root/admin-openrc.sh  
openstack dataprocessing cluster list  
if [ $? -ne 0 ]; then  
    echo "WARNING: unhealthy"  
    exit 1  
fi  
  
exit 0  
~  
~  
~  
~
```

图28 修改目标文件

检验修改是否成功。

```
health-check.sh 10L, 100C written  
[root@sahara-service sahara]# cat health-check.sh  
#!/bin/bash  
#hello  
source /root/admin-openrc.sh  
openstack dataprocessing cluster list  
if [ $? -ne 0 ]; then  
    echo "WARNING: unhealthy"  
    exit 1  
fi  
  
exit 0  
[root@sahara-service sahara]#
```

图29 检验修改是否成功

退出容器，查找到/opt/openstack-transfer/sahara/目录下的health-check.sh，可以看到是刚才修改的文件。

```

exit 0
[root@sahara-service sahara]# exit
exit
[root@single1 /]# cd /opt/openstack-transfer/sahara/
[root@single1 sahara]# ls
health-check.sh
[root@single1 sahara]# cat health-check.sh
#!/bin/bash
#hello
source /root/admin-openrc.sh
openstack dataprocessing cluster list
if [ $? -ne 0 ]; then
    echo "WARNING: unhealthy"
    exit 1
fi
exit 0
[root@single1 sahara]#

```

图30 本地目录下检验修改的文件

## 2. 利用固化工具替换镜像

将修改后的health-check.sh拷贝到固化工具（固化工具事先拷贝至/root/下，并进行解压）解压出来的目录/root/update\_image/patches/replace\_files/files/目标文件在镜像中的目录，在该示例中为/root/update\_image/patches/replace\_files/files/root/scripts/。

```

[root@single1 sahara]# cp health-check.sh /root/update_image/patches/replace_files/files/root/scripts/
cp: overwrite '/root/update_image/patches/replace_files/files/root/scripts/health-check.sh'? y
[root@single1 sahara]#

```

图31 将目标文件拷贝至特定目录

查找要替换文件的镜像名称和镜像的tag。如图中本地镜像cloudos-openstack-sahara，tag为E3107-V300R001B01D030SP01-RC4。然后进入/root/update\_image目录，使用“sh main.sh [镜像名称]:[镜像tag] patches/replace\_files/”命令执行替换镜像文件脚本。

```

[root@single1 sahara]# cp health-check.sh /root/update_image/patches/replace_files/files/root/scripts/
cp: overwrite '/root/update_image/patches/replace_files/files/root/scripts/health-check.sh'? y
[root@single1 sahara]# docker images | grep cloudos-openstack-sahara
192.168.10.99:9999/cloudos-openstack-sahara      E3107-V300R001B01D030SP01-RC4      0a02de697908
3 minutes ago      845.2 MB      E3107-V300R001B01D030SP01-RC4      0a02de697908
cloudos-openstack-sahara
cloudos-openstack-sahara      845.2 MB
3 minutes ago
cloudos-openstack-sahara      E3107-V300R001B01D030SP01-RC4-20191026213246      ccf9a3459f58
months ago      845.2 MB
192.168.10.99:9999/cloudos-openstack-sahara      E3106H01-V300R001B01D029-RC3      6001b510dbc3
months ago      841.3 MB
cloudos-openstack-sahara      E3106H01-V300R001B01D029-RC3      6001b510dbc3
months ago      841.3 MB
[root@single1 sahara]# cd /root/update_image
[root@single1 update_image]# sh main.sh cloudos-openstack-sahara:E3107-V300R001B01D030SP01-RC4 patches/replace_files/
Log file /var/log/update-image.log

*****
***** H3CloudOS Docker Image Build Tool *****
*****

```

图32 执行替换镜像脚本

执行完成后会进行二次确认，输入“Y”则开始执行脚本。

```

Log file /var/log/update-image.log

*****
***** H3CloudOS Docker Image Build Tool *****
*****

The following k8s pods will be automatically restarted:
sahararc-1179d

Continue to build cloudos-openstack-sahara:E3107-V300R001B01D030SP01-RC4 with patches/replace_files/? (Y/N): Y
+ cat
+ echo -e '\033[32mBegin to build docker image: cloudos-openstack-sahara:E3107-V300R001B01D030SP01-RC4-new.\033[0m'
Begin to build docker image: cloudos-openstack-sahara:E3107-V300R001B01D030SP01-RC4-new.
+ docker build -t cloudos-openstack-sahara:E3107-V300R001B01D030SP01-RC4-new patches/replace_files/
Sending build context to Docker daemon 7.168 kB
Step 1 : FROM cloudos-openstack-sahara:E3107-V300R001B01D030SP01-RC4
--> 0a02de697908
Step 2 : COPY ./cmds.sh /root/upgrade_temp/cmds.sh

```

图33 替换镜像过程

最终显示镜像替换完成。

```

de/ef240838: Layer already exists
7d29602353cc: Layer already exists
5620ca9371b3: Layer already exists
ade1effa8aab: Layer already exists
7feafa657c41: Layer already exists
cle00c6e770a: Layer already exists
551ccdbfff5b: Layer already exists
99841e2ecldb: Layer already exists
129b69770e9: Layer already exists
f0e153206981: Pushed
3f2be103310a: Pushed
d027bc8f53d5: Pushed
E3107-V300R001B01D0305P01-RC4: digest: sha256:4b19350ee573ecbe3f263d7a79472d635c9ec21ca74139d3ddc7f7e4c22bf016 size: 5321
+ for node in 'snodes'
+ echo -e '\033[32mSSH to 192.168.10.99 and pull image: 192.168.10.99:9999/cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4.\033[0m'
SSH to 192.168.10.99 and pull image: 192.168.10.99:9999/cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4
+ ssh -t 192.168.10.99 'docker pull 192.168.10.99:9999/cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4 && docker tag 192.168.10.99:9999/cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4 cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4'
E3107-V300R001B01D0305P01-RC4: Pulling from cloudos-openstack-sahara
Digest: sha256:4b19350ee573ecbe3f263d7a79472d635c9ec21ca74139d3ddc7f7e4c22bf016
Status: Image is up to date for 192.168.10.99:9999/cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4
Connection to 192.168.10.99 closed.
+ for pod in 'spods'
+ echo -e '\033[32mDelete sahararc-t179d for enabled modify.\033[0m'
Delete sahararc-t179d for enabled modify.
+ /opt/bin/kubectl -s 127.0.0.1:8888 delete pod sahararc-t179d
pod "sahararc-t179d" deleted
+ docker rmi cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4-backup
Untagged: cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4-backup
+ echo -e '\033[32mUpgrade docker image cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4 finished.\033[0m'
Upgrade docker image cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4 finished.
[root@single1 update_image]#

```

图34 替换镜像成功

### 3. 查看容器固化操作是否成功

使用“pod | grep [镜像名称]”命令查找由该镜像启动的pod。图中所示“59s”表示该容器是59秒前创建的。

```

Upgrade docker image cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4 finished.
[root@single1 update_image]# pod | grep sahara
default          sahararc-wnrq8      1/1      Running      0          59s         10.101.13.26   192.168.10.99

```

图35 查看新建镜像

进入容器中，找到/root/scripts/目录下目标文件health-check.sh，查看文件是否替换成功。使用cat命令查看，如图，可以看到之前增加的注释“#hello”，说明容器固化操作成功。

```

E3107-V300R001B01D0305P01-RC4: Pulling from cloudos-openstack-sahara
Digest: sha256:4b19350ee573ecbe3f263d7a79472d635c9ec21ca74139d3ddc7f7e4c22bf016
Status: Image is up to date for 192.168.10.99:9999/cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4
Connection to 192.168.10.99 closed.
+ for pod in 'spods'
+ echo -e '\033[32mDelete sahararc-t179d for enabled modify.\033[0m'
Delete sahararc-t179d for enabled modify.
+ /opt/bin/kubectl -s 127.0.0.1:8888 delete pod sahararc-t179d
pod "sahararc-t179d" deleted
+ docker rmi cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4-backup
Untagged: cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4-backup
+ echo -e '\033[32mUpgrade docker image cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4 finished.\033[0m'
Upgrade docker image cloudos-openstack-sahara:E3107-V300R001B01D0305P01-RC4 finished.
[root@single1 update_image]# pod | grep sahara
default          sahararc-wnrq8      1/1      Running      0          59s         10.101.13.26   192.168.10.99
[root@single1 update_image]# kubectl exec -it sahararc-wnrq8 bash
[root@sahara-service /]# cd /root/scripts/
[root@sahara-service scripts]# ls
health-check.sh  pre-stop.sh          upgrade
post-startup.sh  resign-openstack.sh upgrade-db-to-pike.sh
pre-startup.sh   tools                upgrade-opensls.sh
[root@sahara-service scripts]# cat health-check.sh
#!/bin/bash
#hello
source /root/admin-openrc.sh
openstack dataprocessing cluster list
if [ $? -ne 0 ]; then
    echo "WARNING: unhealthy"
    exit 1
fi
exit 0
[root@sahara-service scripts]#

```

图36 查看容器固化是否成功

#### 配置关键点

- 1、在CloudOS2.0版本中使用“docker rmi cloudos-openstack:E1139H06”命令并没有风险，该命令只是去除原镜像的标签，并没有将原镜像删除，因此只要记住原镜像的名称和uuid即可。