# 搭建Kubernetes高可用集群

此文以Kubernetes 1.18.5版本为例！

如未指定，下述命令在所有节点执行！

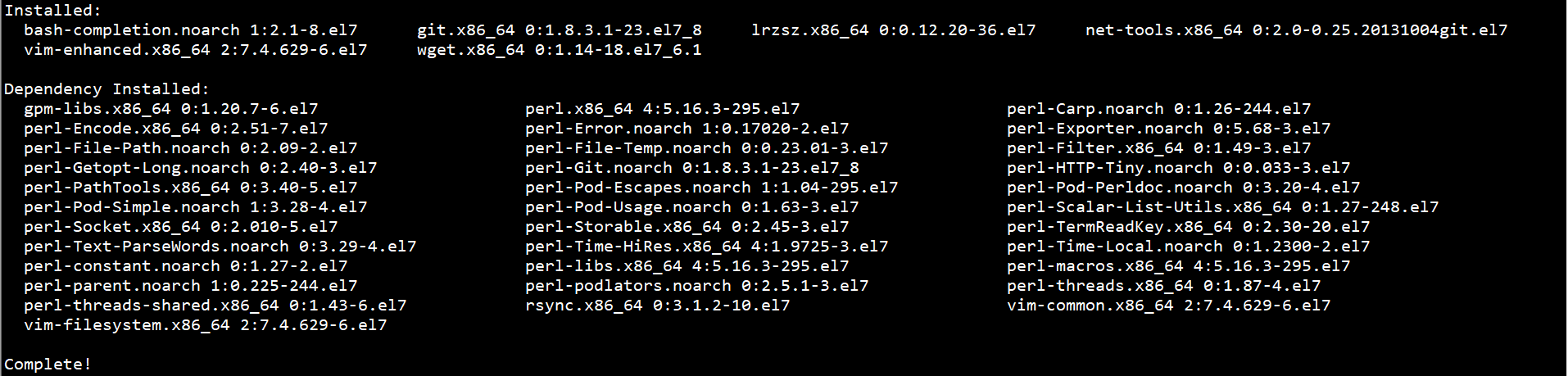
## 一、系统资源规划

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **节点名称** | **系统名称** | **CPU/内存** | **网卡** | **磁盘** | **IP地址** | **OS** |
| Master1 | master1 | 4C/8G | ens33 | 64G | 192.168.0.11 | CentOS7 |
| Master2 | master2 | 4C/8G | ens33 | 64G | 192.168.0.12 | CentOS7 |
| Master3 | master3 | 4C/8G | ens33 | 64G | 192.168.0.13 | CentOS7 |
| Worker1 | worker1 | 4C/8G | ens33 | 64G | 192.168.0.21 | CentOS7 |
| Worker2 | worker2 | 4C/8G | ens33 | 64G | 192.168.0.22 | CentOS7 |
| Worker3 | worker3 | 4C/8G | ens33 | 64G | 192.168.0.23 | CentOS7 |

## 二、系统软件安装与设置

### 1、安装基本软件

yum -y install vim git lrzsz wget net-tools bash-completion



### 2、设置名称解析

echo 192.168.0.11 master1 >> /etc/hosts

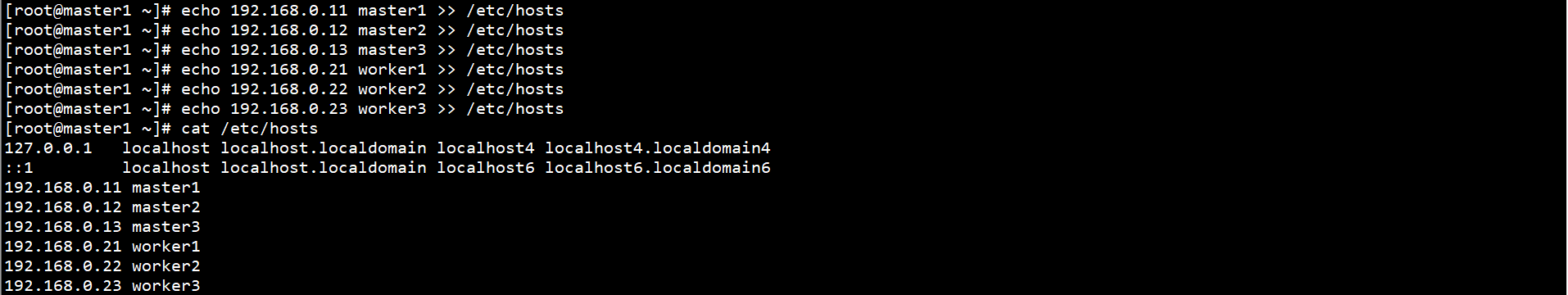
echo 192.168.0.12 master2 >> /etc/hosts

echo 192.168.0.13 master3 >> /etc/hosts

echo 192.168.0.21 worker1 >> /etc/hosts

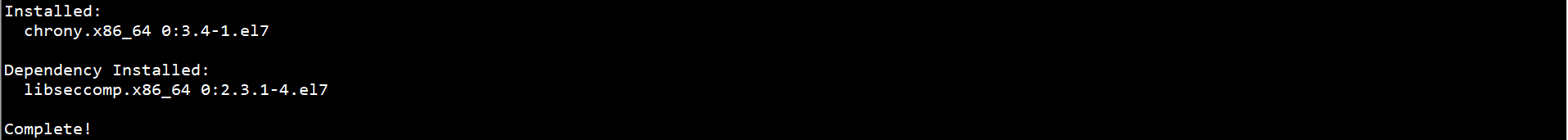
echo 192.168.0.22 worker2 >> /etc/hosts

echo 192.168.0.23 worker3 >> /etc/hosts



### 3、设置NTP

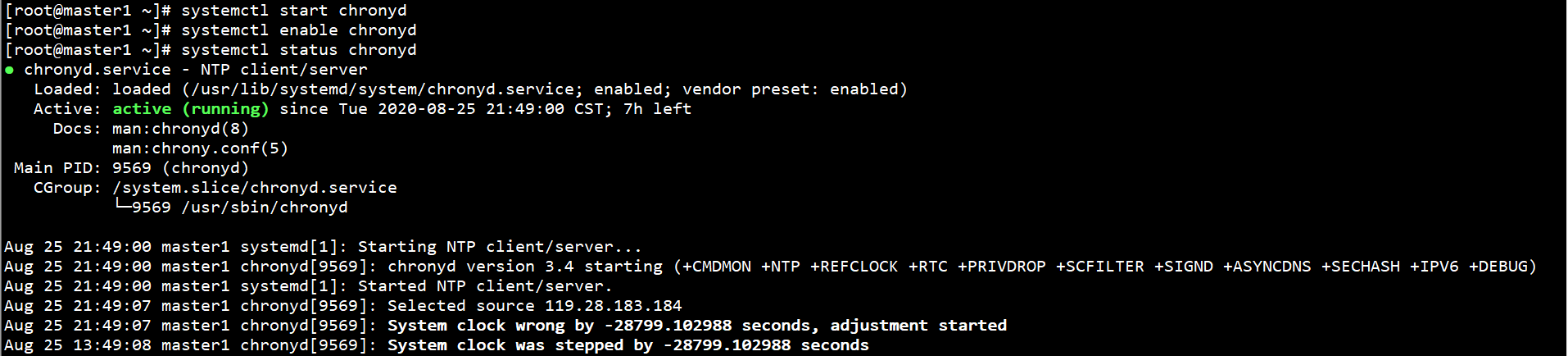
yum -y install chrony



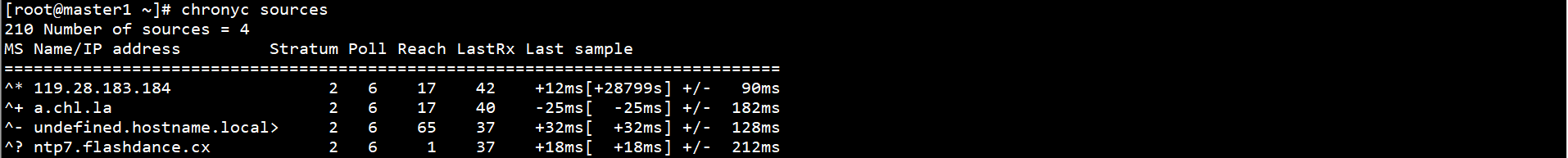
systemctl start chronyd

systemctl enable chronyd

systemctl status chronyd



chronyc sources



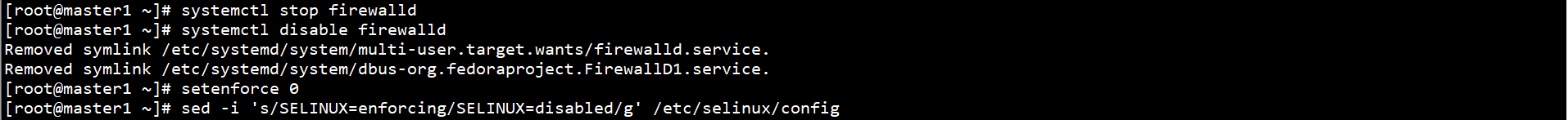
### 4、设置SELinux、防火墙

systemctl stop firewalld

systemctl disable firewalld

setenforce 0

sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config



### 5、设置网桥

配置L2网桥在转发包时会被iptables的FORWARD规则所过滤，CNI插件需要该配置

创建/etc/sysctl.d/k8s.conf文件，添加如下内容：

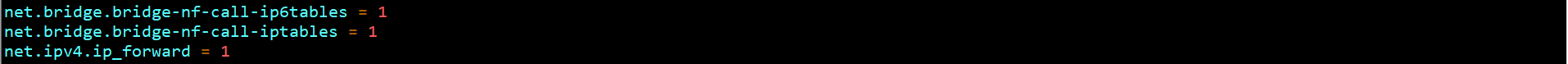
cat > /etc/sysctl.d/k8s.conf << EOF

net.bridge.bridge-nf-call-ip6tables = 1

net.bridge.bridge-nf-call-iptables = 1

net.ipv4.ip\_forward = 1

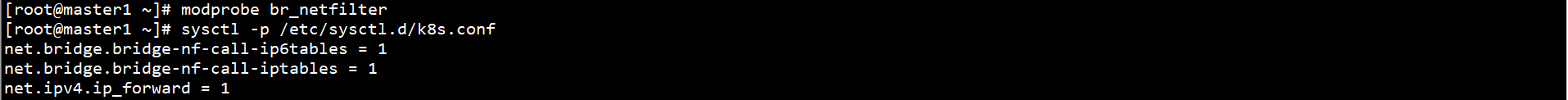
EOF



执行命令，使修改生效：

modprobe br\_netfilter

sysctl -p /etc/sysctl.d/k8s.conf



### 6、设置swap

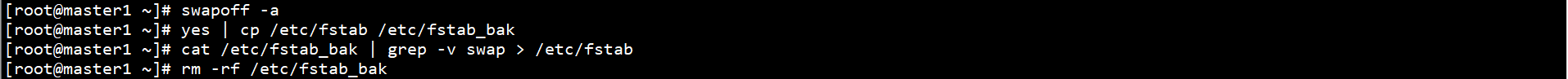
关闭系统swap分区：

swapoff -a

yes | cp /etc/fstab /etc/fstab\_bak

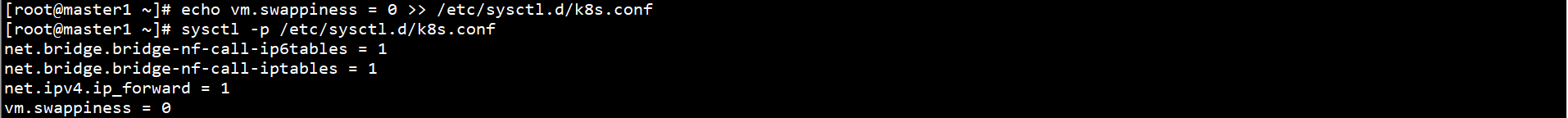
cat /etc/fstab\_bak | grep -v swap > /etc/fstab

rm -rf /etc/fstab\_bak



echo vm.swappiness = 0 >> /etc/sysctl.d/k8s.conf

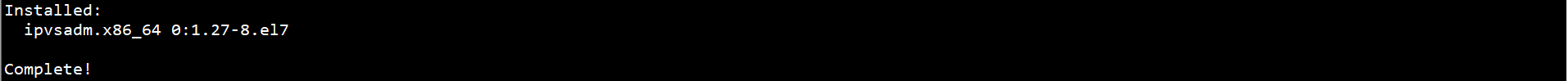
sysctl -p /etc/sysctl.d/k8s.conf



### 7、设置ipvs

安装ipvsadm ipset：

yum -y install ipvsadm ipset



创建ipvs设置脚本：

cat > /etc/sysconfig/modules/ipvs.modules << EOF

#!/bin/bash

modprobe -- ip\_vs

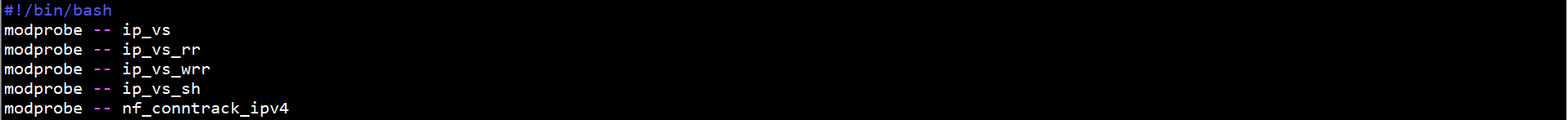
modprobe -- ip\_vs\_rr

modprobe -- ip\_vs\_wrr

modprobe -- ip\_vs\_sh

modprobe -- nf\_conntrack\_ipv4

EOF

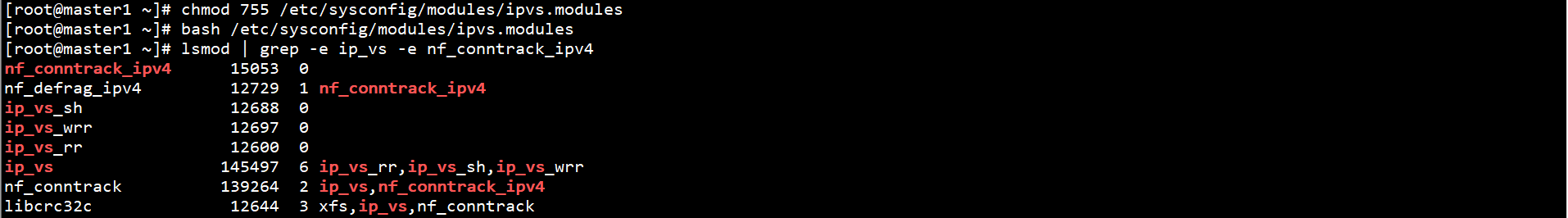


执行脚本，验证修改结果：

chmod 755 /etc/sysconfig/modules/ipvs.modules

bash /etc/sysconfig/modules/ipvs.modules

lsmod | grep -e ip\_vs -e nf\_conntrack\_ipv4

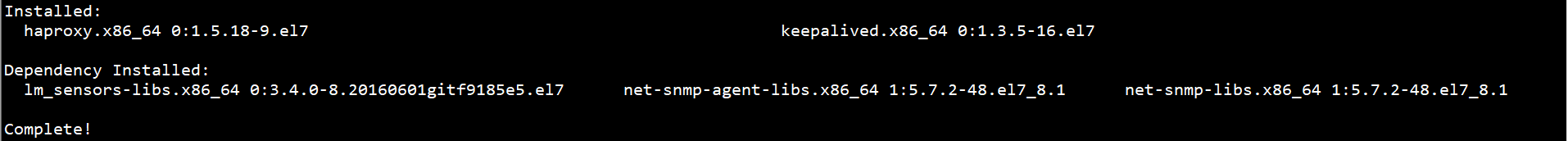


## 三、负载均衡配置

### 1、安装HAProxy和Keepalived

在所有Master节点上安装HAProxy和Keepalived：

yum -y install haproxy keepalived



在所有Master节点上创建HAProxy配置文件：

cat > /etc/haproxy/haproxy.cfg << EOF

global

log 127.0.0.1 local2

chroot /var/lib/haproxy

pidfile /var/run/haproxy.pid

maxconn 4000

user haproxy

group haproxy

daemon

stats socket /var/lib/haproxy/stats

defaults

mode tcp

log global

option tcplog

option dontlognull

option redispatch

retries 3

timeout queue 1m

timeout connect 10s

timeout client 1m

timeout server 1m

timeout check 10s

maxconn 3000

frontend k8s\_https \*:8443

mode tcp

maxconn 2000

default\_backend https\_sri

backend https\_sri

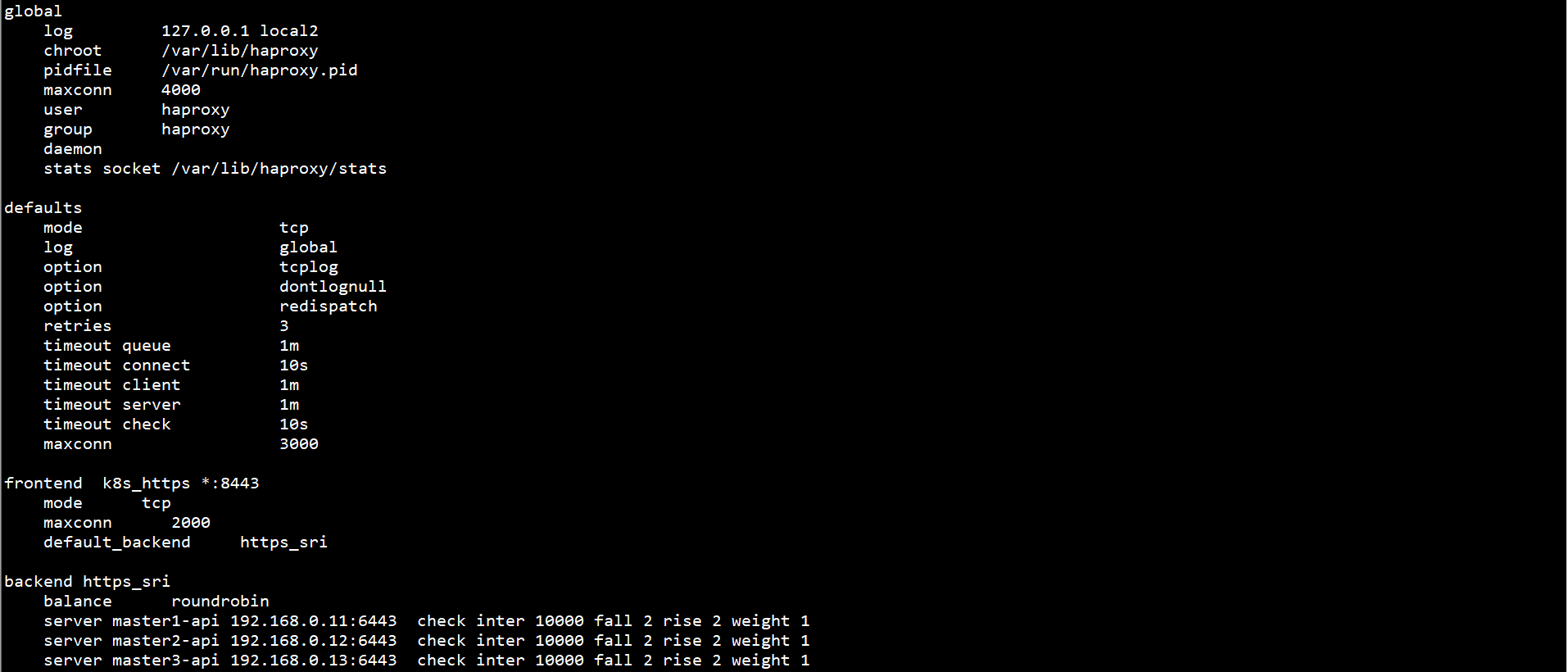
balance roundrobin

server master1-api 192.168.0.11:6443 check inter 10000 fall 2 rise 2 weight 1

server master2-api 192.168.0.12:6443 check inter 10000 fall 2 rise 2 weight 1

server master3-api 192.168.0.13:6443 check inter 10000 fall 2 rise 2 weight 1

EOF



在Master1节点上创建Keepalived配置文件：

cat > /etc/keepalived/keepalived.conf << EOF

global\_defs {

router\_id LVS\_DEVEL

}

vrrp\_script check\_haproxy {

script "/etc/keepalived/check\_haproxy.sh"

interval 3000

}

vrrp\_instance VI\_1 {

state Master

interface ens33

virtual\_router\_id 80

priority 100

advert\_int 1

authentication {

auth\_type PASS

auth\_pass 111111

}

virtual\_ipaddress {

192.168.0.10/24

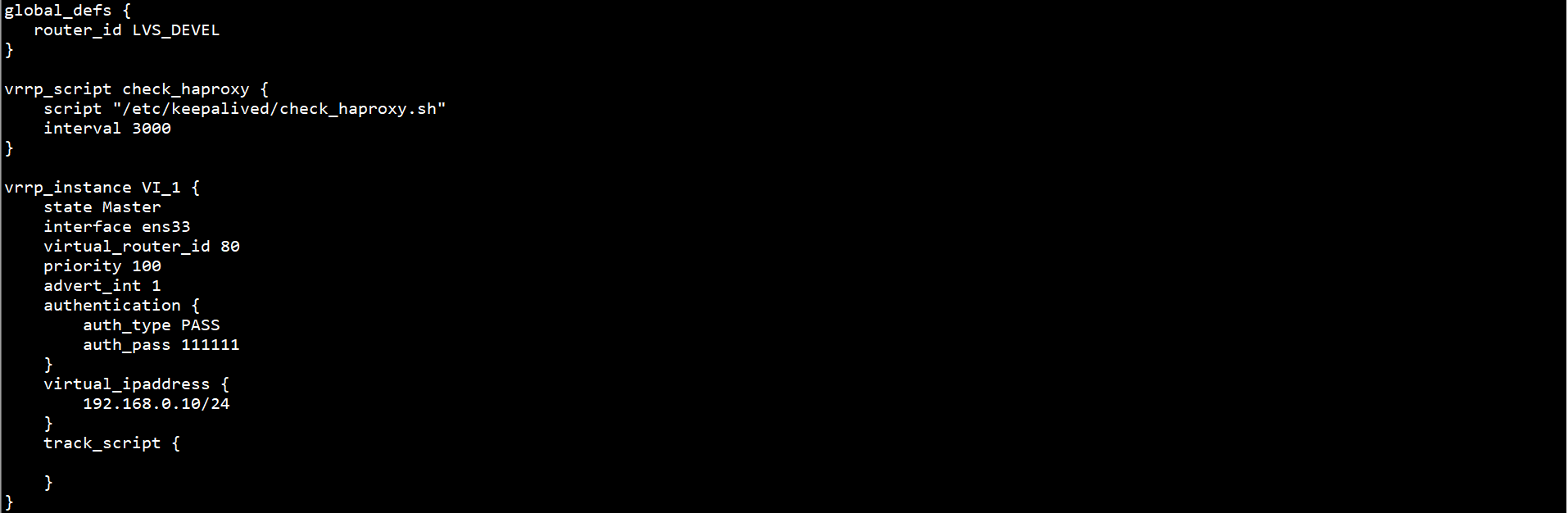
}

track\_script {

}

}

EOF



在Master2节点上创建Keepalived配置文件：

cat > /etc/keepalived/keepalived.conf << EOF

global\_defs {

router\_id LVS\_DEVEL

}

vrrp\_script check\_haproxy {

script "/etc/keepalived/check\_haproxy.sh"

interval 3000

}

vrrp\_instance VI\_1 {

state Slave

interface ens33

virtual\_router\_id 80

priority 50

advert\_int 1

authentication {

auth\_type PASS

auth\_pass 111111

}

virtual\_ipaddress {

192.168.0.10/24

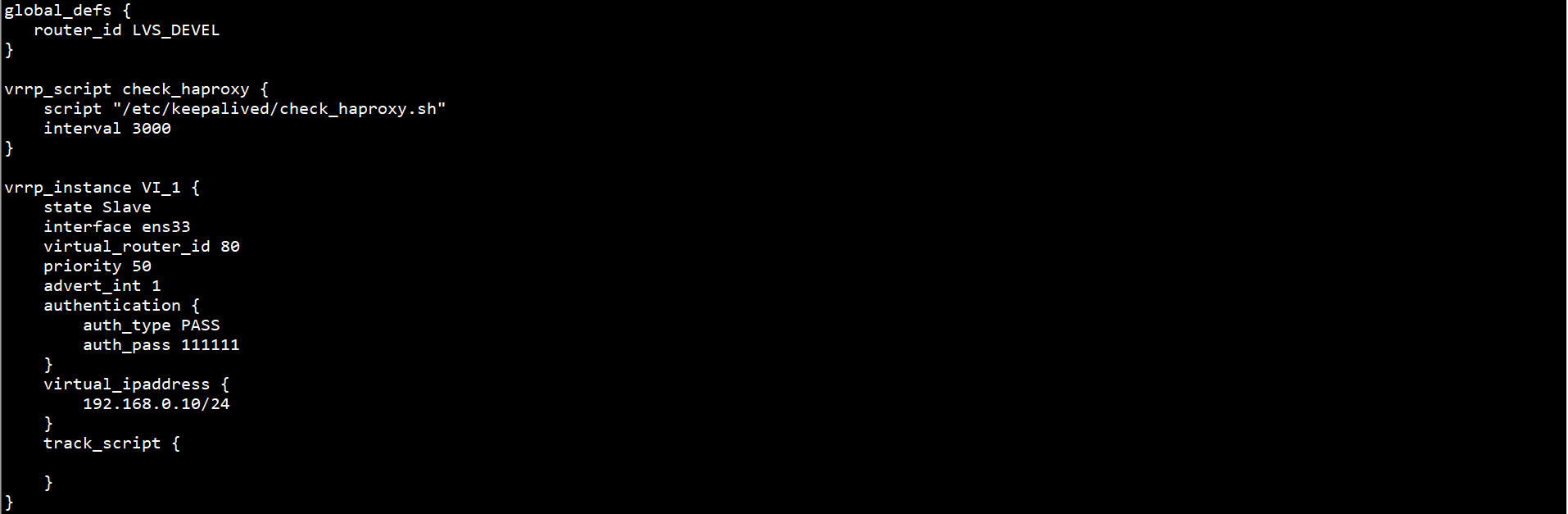
}

track\_script {

}

}

EOF



在Master3节点上创建Keepalived配置文件：

cat > /etc/keepalived/keepalived.conf << EOF

global\_defs {

router\_id LVS\_DEVEL

}

vrrp\_script check\_haproxy {

script "/etc/keepalived/check\_haproxy.sh"

interval 3000

}

vrrp\_instance VI\_1 {

state Slave

interface ens33

virtual\_router\_id 80

priority 30

advert\_int 1

authentication {

auth\_type PASS

auth\_pass 111111

}

virtual\_ipaddress {

192.168.0.10/24

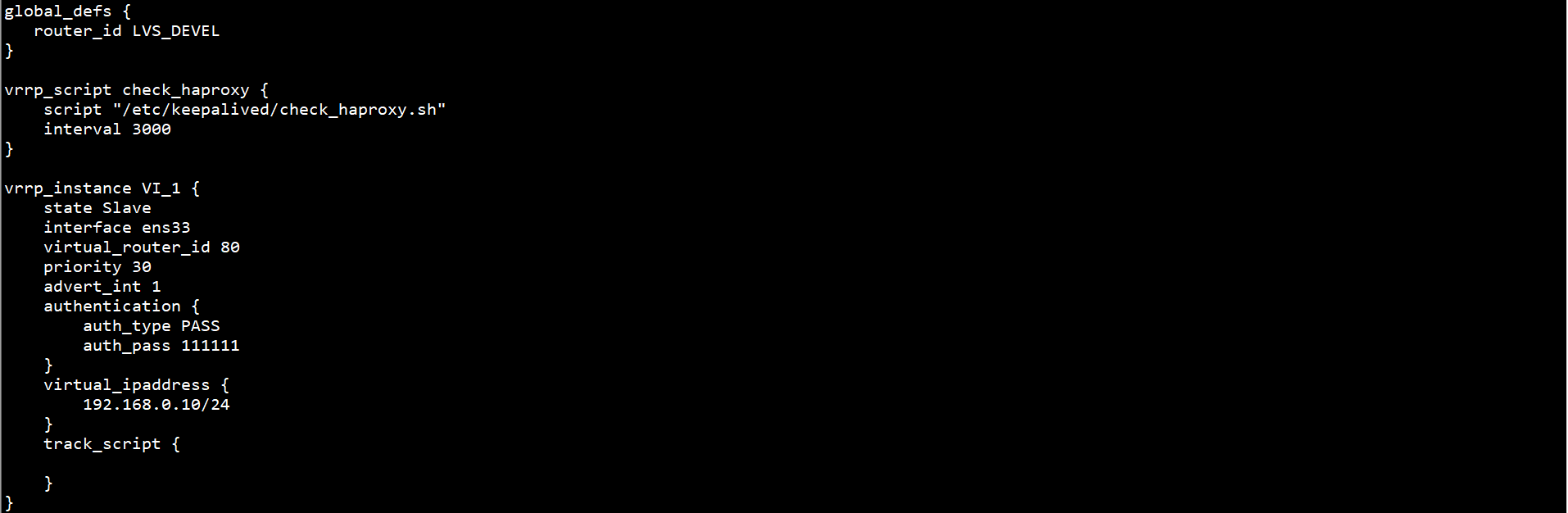
}

track\_script {

}

}

EOF



在所有Master节点上创建HAProxy检查脚本：

vim /etc/keepalived/check\_haproxy.sh

#!/bin/bash

if [ `ps -C haproxy --no-header | wc -l` == 0 ]; then

systemctl start haproxy

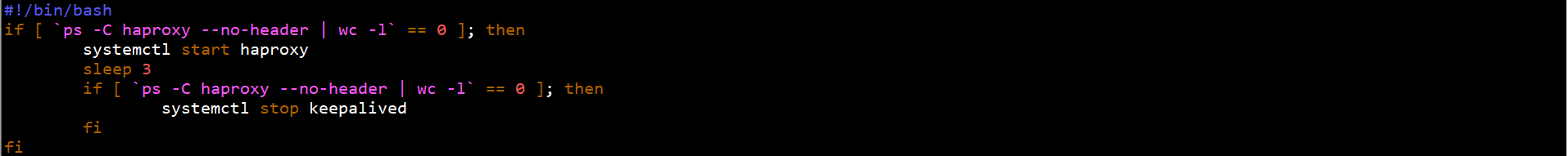
sleep 3

if [ `ps -C haproxy --no-header | wc -l` == 0 ]; then

systemctl stop keepalived

fi

fi



chmod +x /etc/keepalived/check\_haproxy.sh

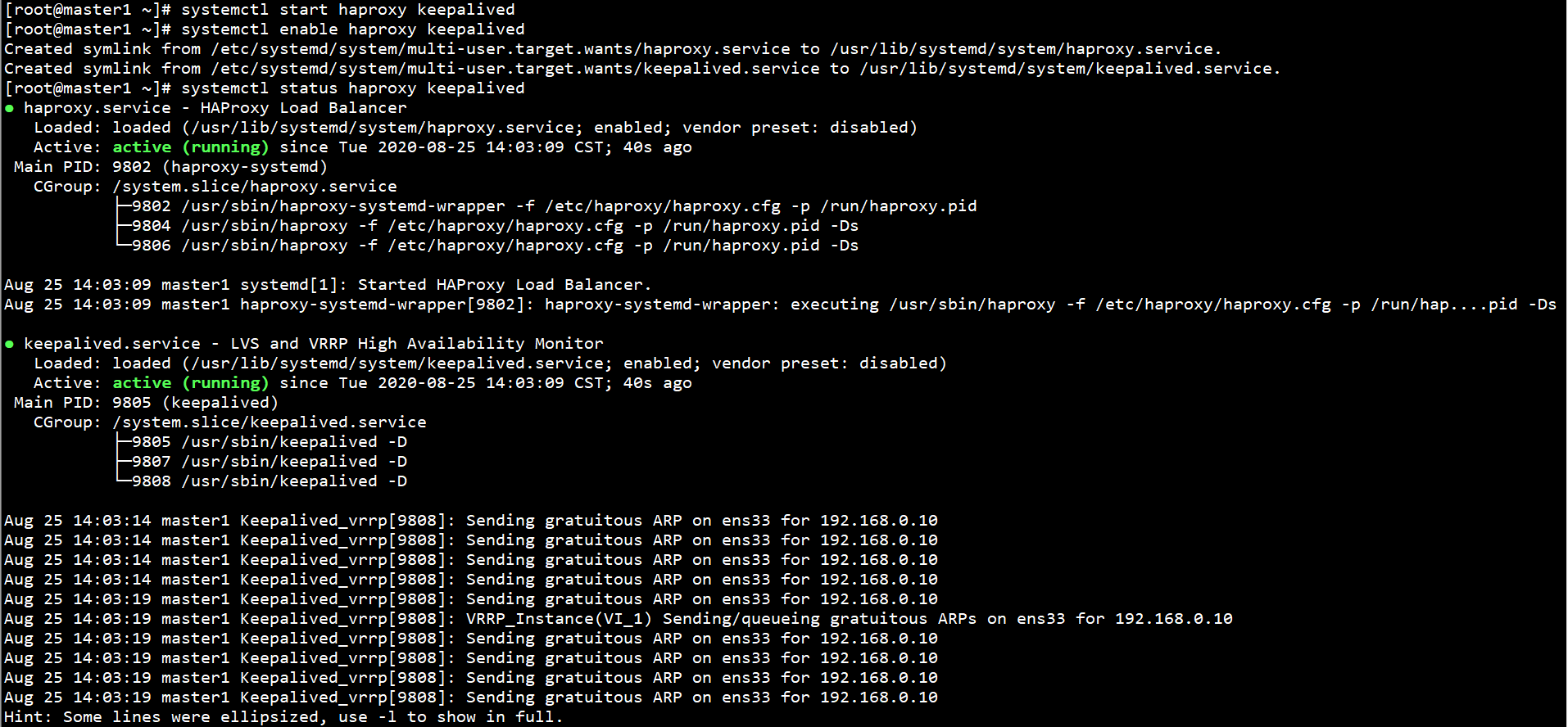


启动HAProxy和Keepalived，并设置自启动：

systemctl start haproxy keepalived

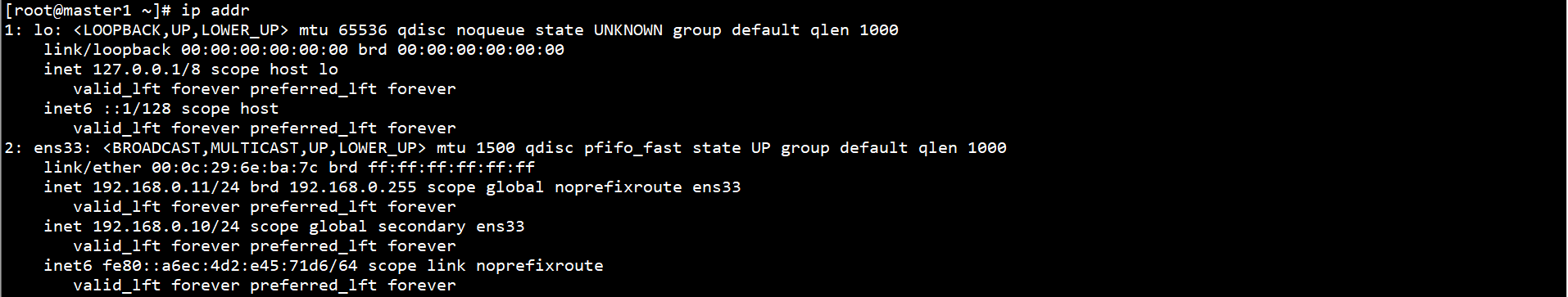
systemctl enable haproxy keepalived

systemctl status haproxy keepalived



查看keepalived工作状态：

ip addr



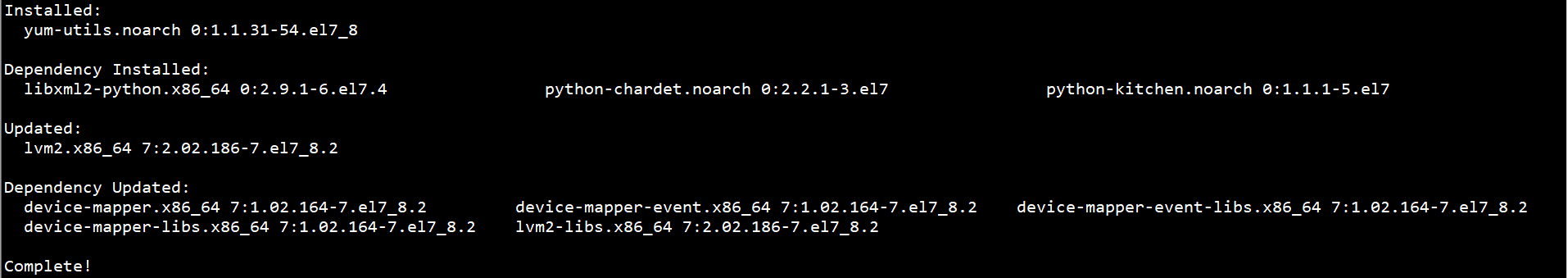
在ens33网卡绑定了192.168.0.10虚拟IP

## 四、Kubernetes集群配置

### 1、安装Docker

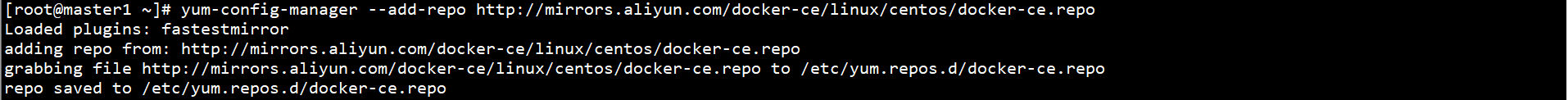
安装所需软件包：

yum -y install yum-utils device-mapper-persistent-data lvm2



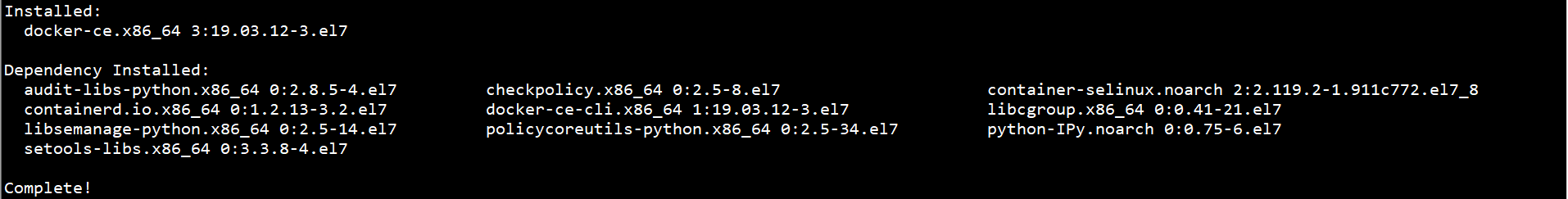
设置稳定存储库：

yum-config-manager --add-repo <http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo>



安装Docker CE：

yum -y install docker-ce

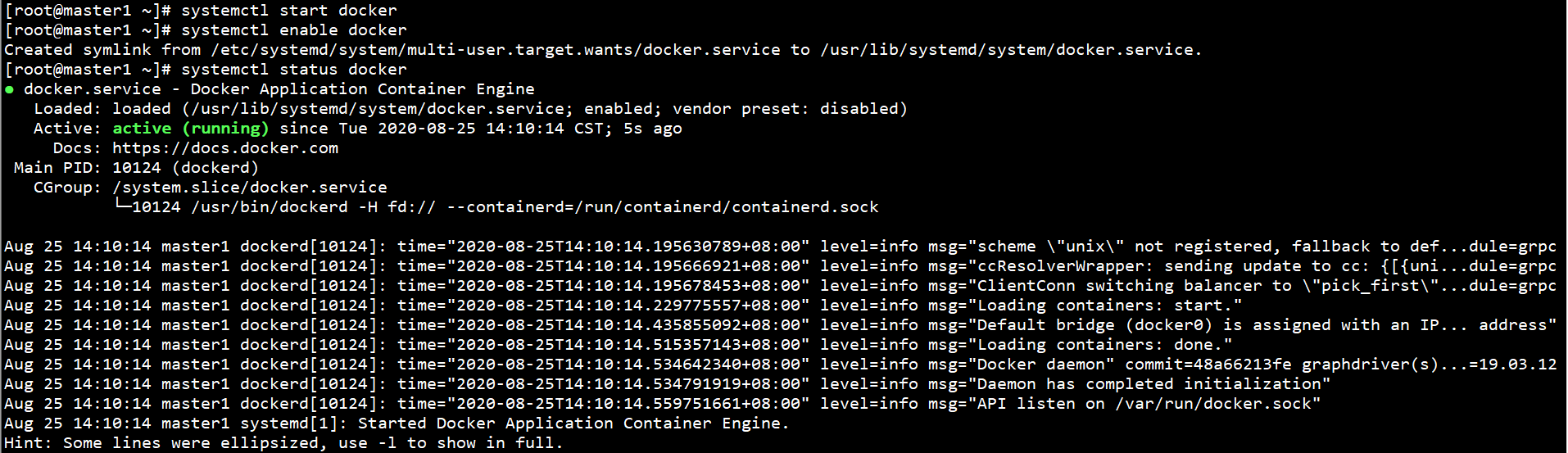


启动Docker，并设置自启动：

systemctl start docker

systemctl enable docker

systemctl status docker



### 2、安装kubelet、kubeadm和kubectl

添加kubernetes存储库：

cat > /etc/yum.repos.d/kubernetes.repo << EOF

[kubernetes]

name=Kubernetes

baseurl=http://mirrors.aliyun.com/kubernetes/yum/repos/kubernetes-el7-x86\_64

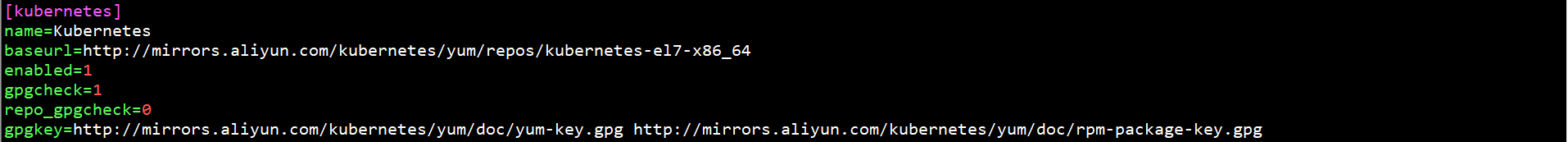
enabled=1

gpgcheck=1

repo\_gpgcheck=0

gpgkey=http://mirrors.aliyun.com/kubernetes/yum/doc/yum-key.gpg http://mirrors.aliyun.com/kubernetes/yum/doc/rpm-package-key.gpg

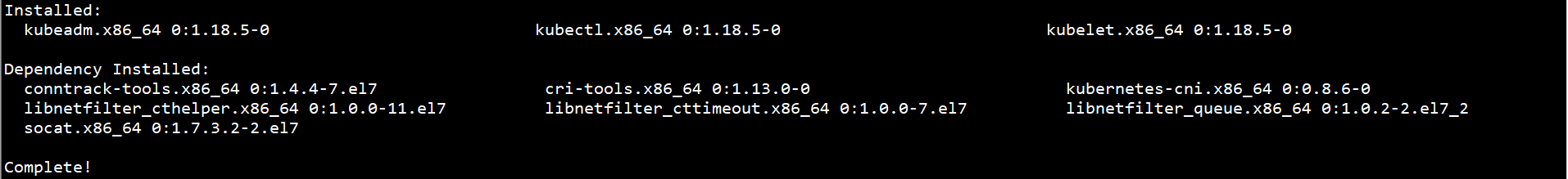
EOF



安装kubelet、kubeadm和kubectl：

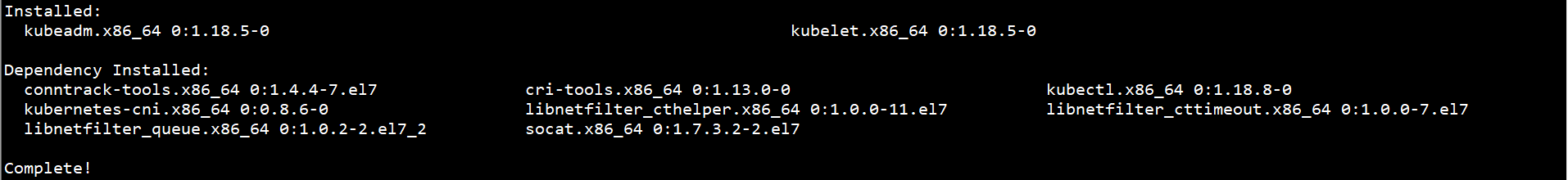
Master节点：

yum -y install kubelet-1.18.5-0 kubeadm-1.18.5-0 kubectl-1.18.5-0



Worker节点：

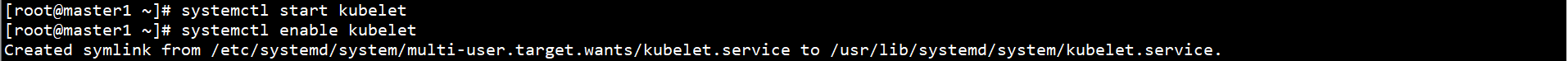
yum -y install kubelet-1.18.5-0 kubeadm-1.18.5-0



启动kubelet，并设置自启动：

systemctl start kubelet

systemctl enable kubelet



此时kubelet缺省配置文件无法启动，可忽略状态

### 3、设置cgroup driver

修改docker cgroup driver为systemd：

cat > /etc/docker/daemon.json << EOF

{

"exec-opts": ["native.cgroupdriver=systemd"],

"log-driver": "json-file",

"log-opts": {

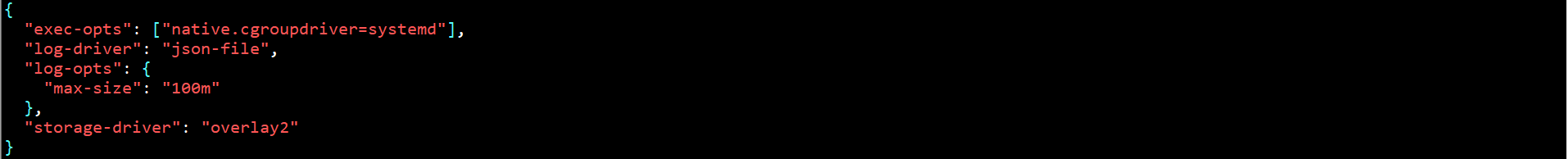
"max-size": "100m"

},

"storage-driver": "overlay2"

}

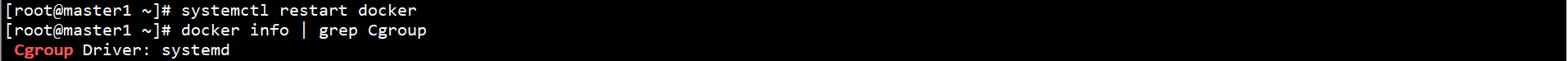
EOF



重启Docker，验证修改结果：

systemctl restart docker

docker info | grep Cgroup

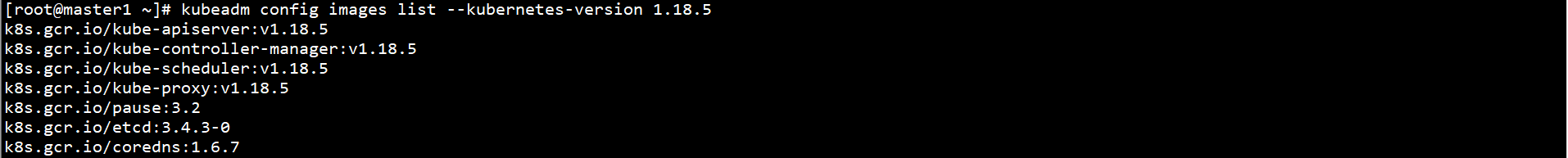


### 4、下载镜像

由于镜像在google在Registry上，国内无法访问，需要手动从阿里云或其他Registry上下载

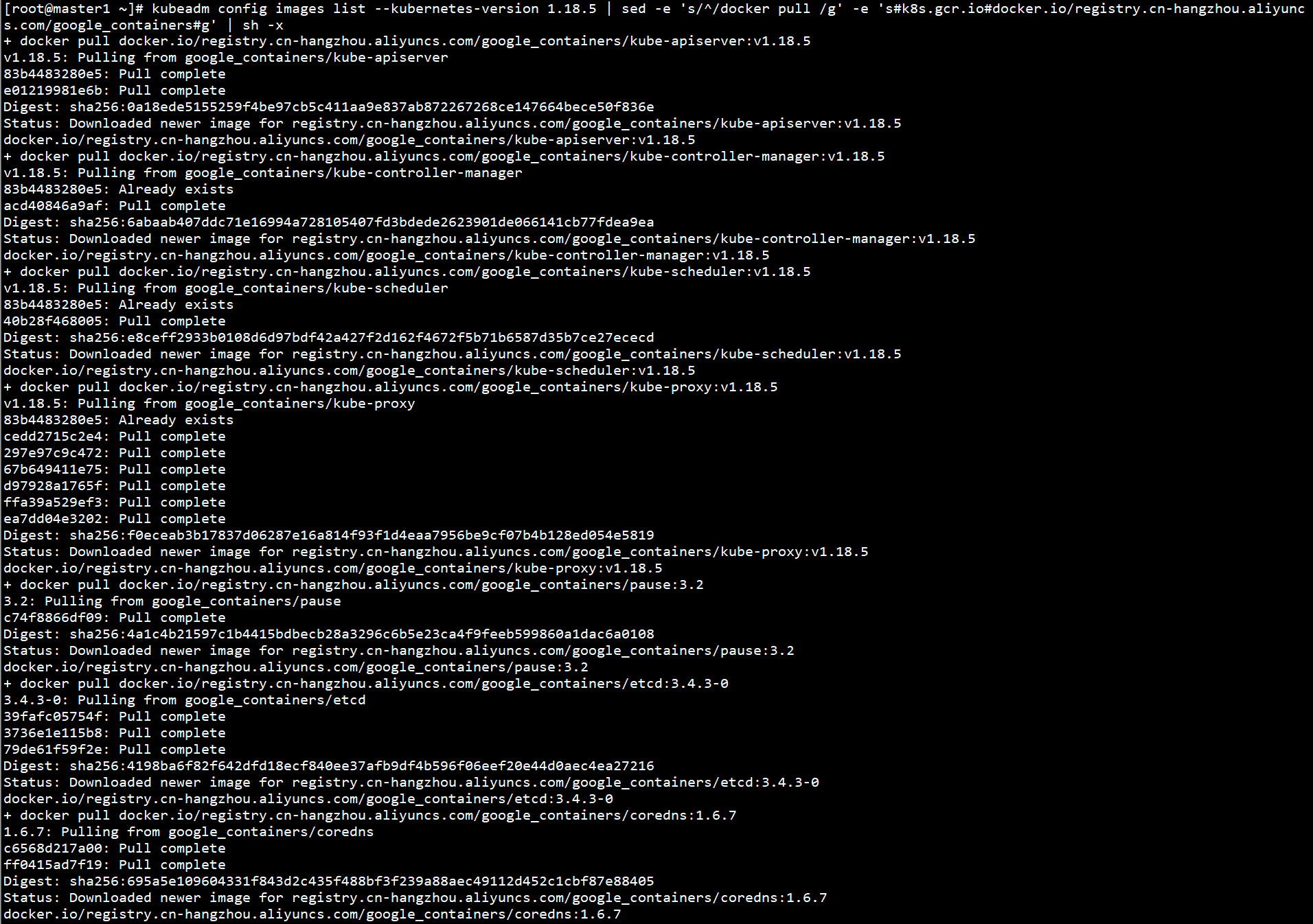
在任意Master节点上查看所需下载镜像：

kubeadm config images list --kubernetes-version 1.18.5



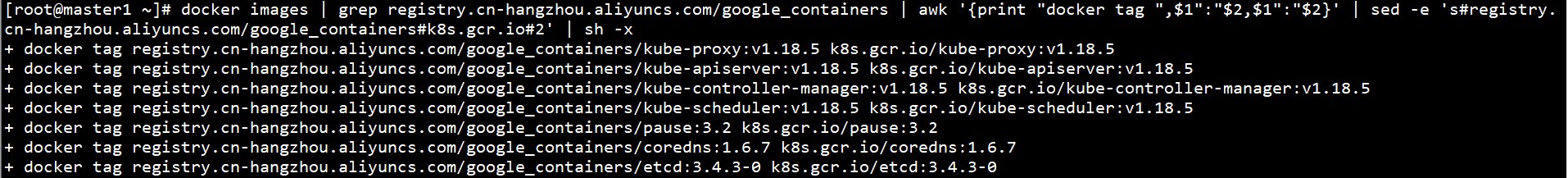
下载镜像：

kubeadm config images list --kubernetes-version 1.18.5 | sed -e 's/^/docker pull /g' -e 's#k8s.gcr.io#docker.io/registry.cn-hangzhou.aliyuncs.com/google\_containers#g' | sh -x



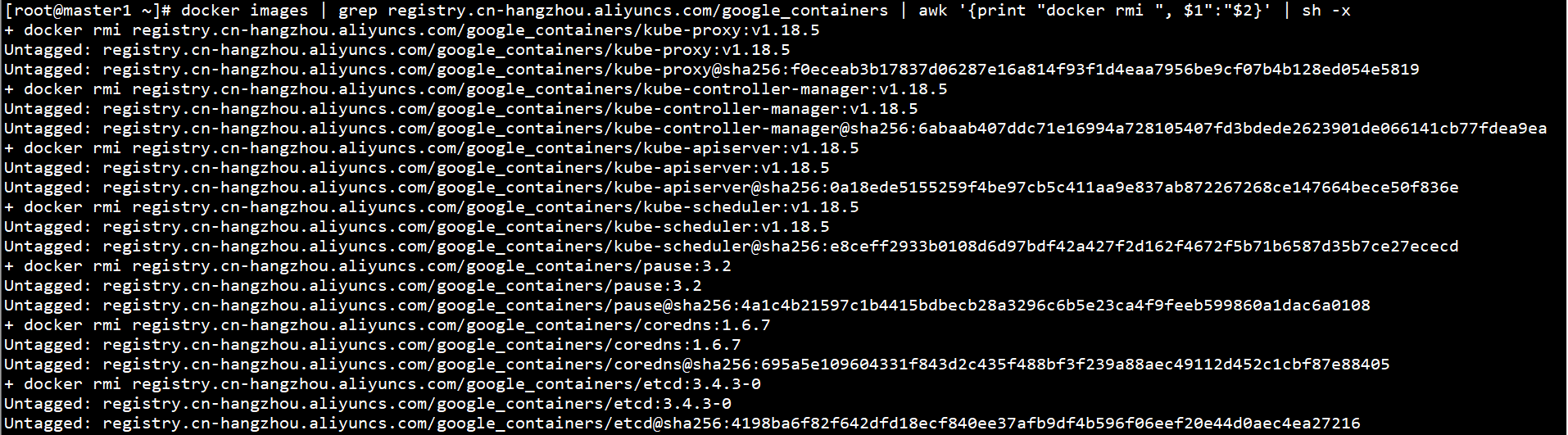
修改镜像：

docker images | grep registry.cn-hangzhou.aliyuncs.com/google\_containers | awk '{print "docker tag ",$1":"$2,$1":"$2}' | sed -e 's#registry.cn-hangzhou.aliyuncs.com/google\_containers#k8s.gcr.io#2' | sh -x



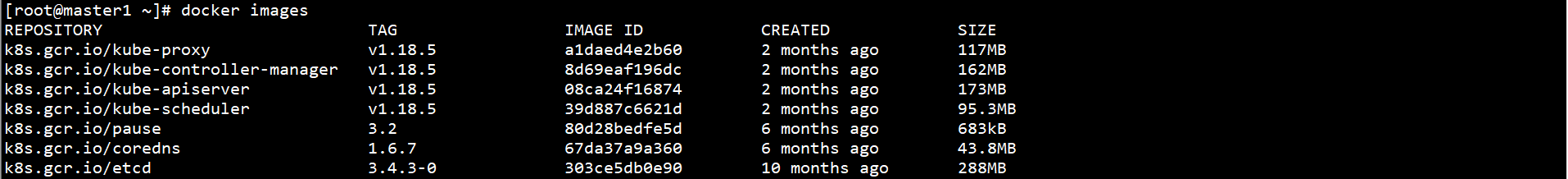
删除无用镜像：

docker images | grep registry.cn-hangzhou.aliyuncs.com/google\_containers | awk '{print "docker rmi ", $1":"$2}' | sh -x



查看镜像：

docker images



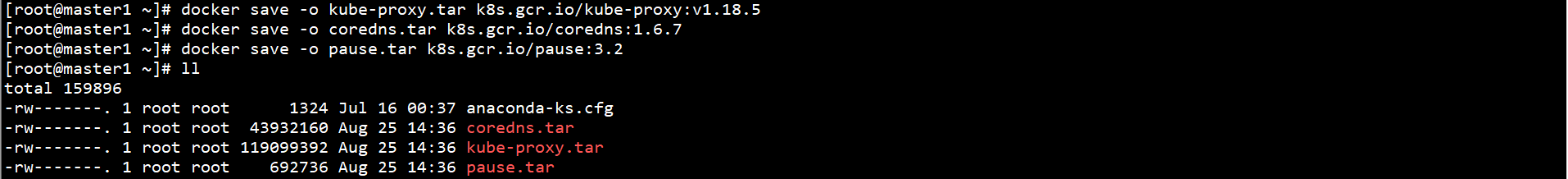
Worker节点也需部分镜像，按上述步骤下载，这里介绍另一种方法，导入镜像文件

在Master节点上保存镜像：

docker save -o kube-proxy.tar k8s.gcr.io/kube-proxy:v1.18.5

docker save -o coredns.tar k8s.gcr.io/coredns:1.6.7

docker save -o pause.tar k8s.gcr.io/pause:3.2

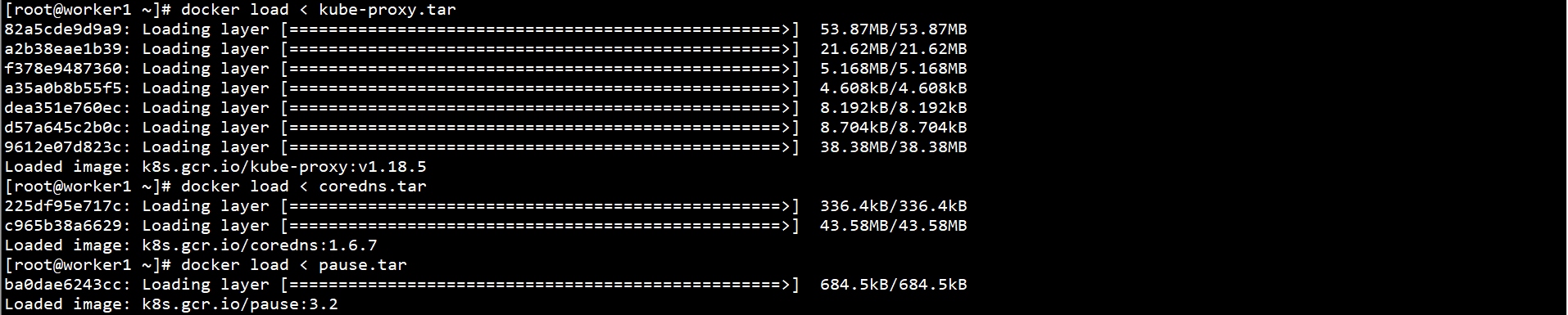


将镜像传到Worker节点上，在Worker节点上导入镜像：

docker load < kube-proxy.tar

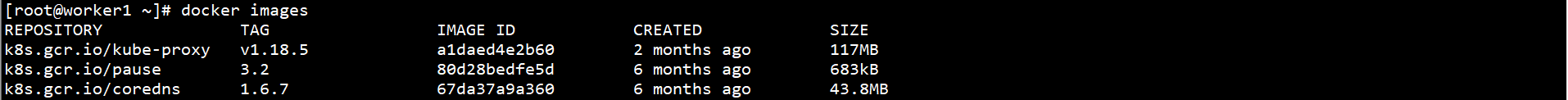
docker load < coredns.tar

docker load < pause.tar



查看镜像：

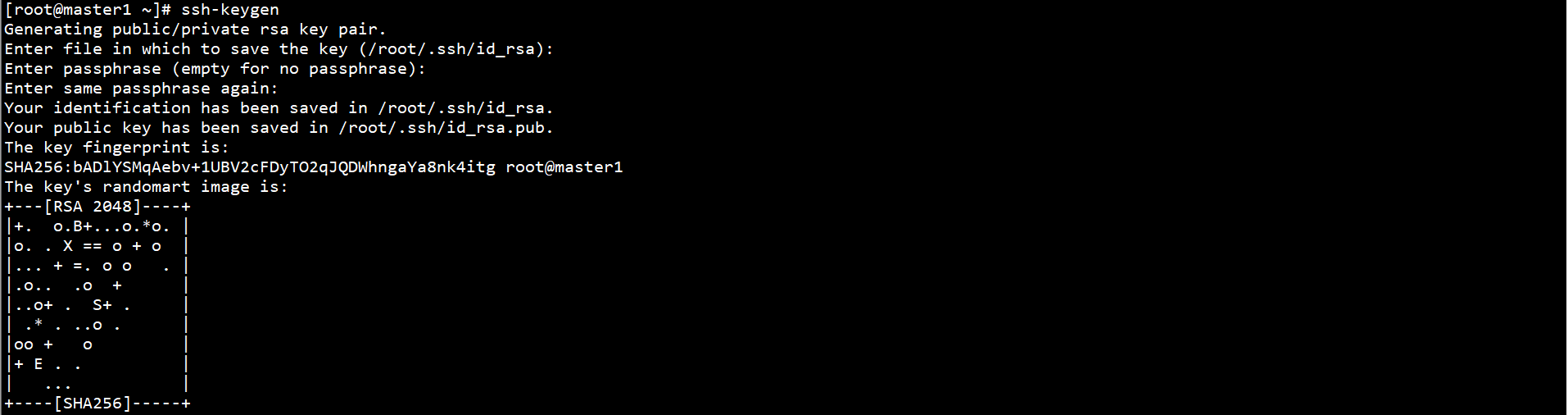
docker images



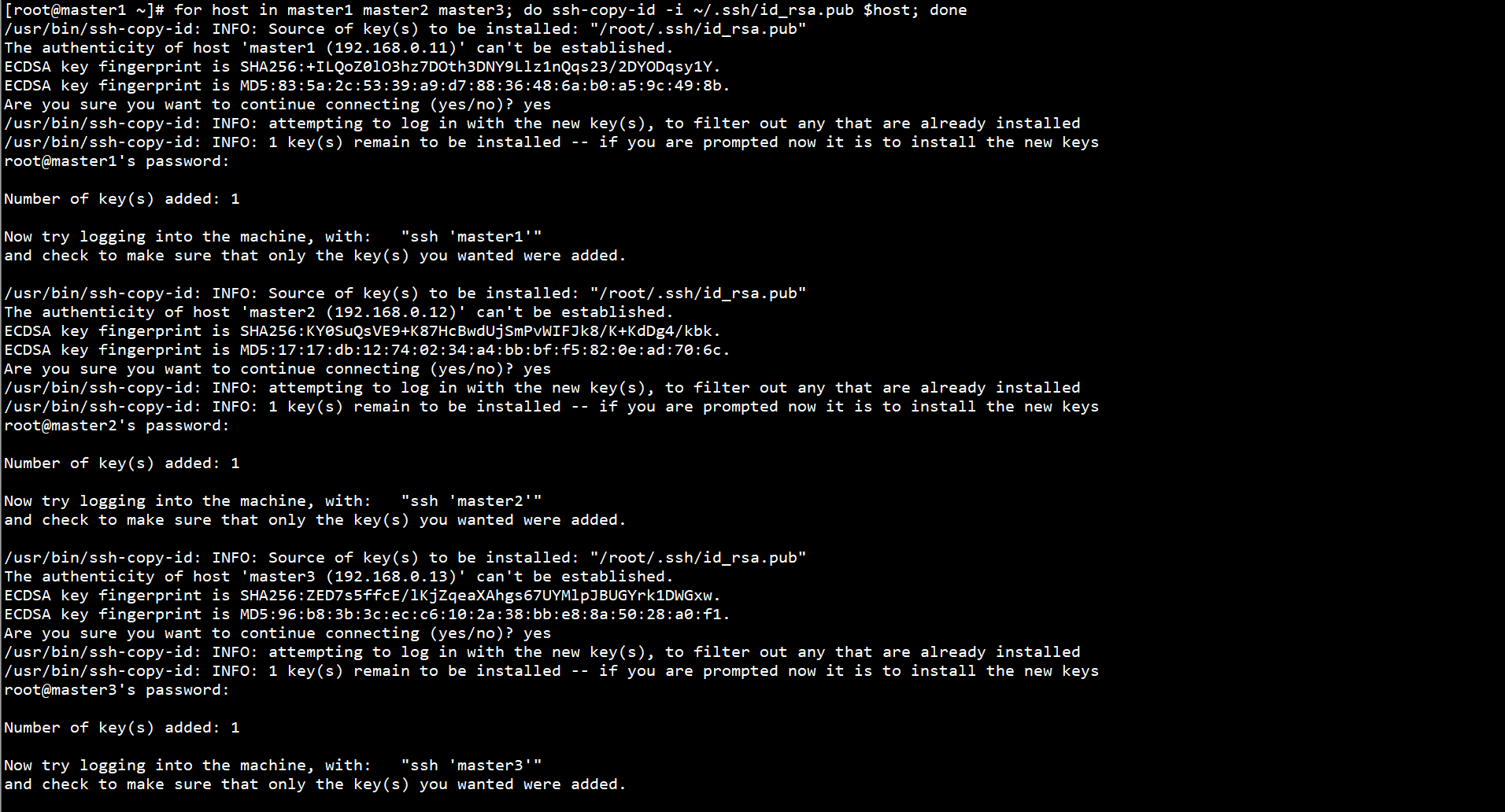
### 5、初始化高可用集群

在Master1节点上生成密钥，可免密登录Master2和Master3

ssh-keygen



for host in master1 master2 master3; do ssh-copy-id -i ~/.ssh/id\_rsa.pub $host; done



在Master1节点上创建集群配置文件：

cat > /etc/kubernetes/kubeadm-config.yaml << EOF

apiVersion: kubeadm.k8s.io/v1beta2

kind: ClusterConfiguration

kubernetesVersion: v1.18.5

controlPlaneEndpoint: "192.168.0.10:8443"

apiServer:

certSANs:

- 192.168.0.11

- 192.168.0.12

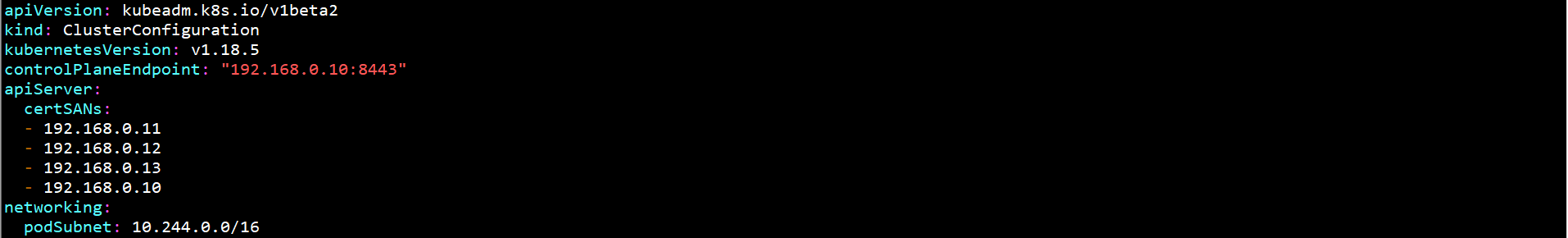
- 192.168.0.13

- 192.168.0.10

networking:

podSubnet: 10.244.0.0/16

EOF



初始化高可用集群：

kubeadm init --config /etc/kubernetes/kubeadm-config.yaml



拷贝证书至其余Master：

for node in master2 master3; do

ssh $node "mkdir -p /etc/kubernetes/pki/etcd; mkdir -p ~/.kube/"

scp /etc/kubernetes/pki/ca.crt $node:/etc/kubernetes/pki/ca.crt

scp /etc/kubernetes/pki/ca.key $node:/etc/kubernetes/pki/ca.key

scp /etc/kubernetes/pki/sa.key $node:/etc/kubernetes/pki/sa.key

scp /etc/kubernetes/pki/sa.pub $node:/etc/kubernetes/pki/sa.pub

scp /etc/kubernetes/pki/front-proxy-ca.crt $node:/etc/kubernetes/pki/front-proxy-ca.crt

scp /etc/kubernetes/pki/front-proxy-ca.key $node:/etc/kubernetes/pki/front-proxy-ca.key

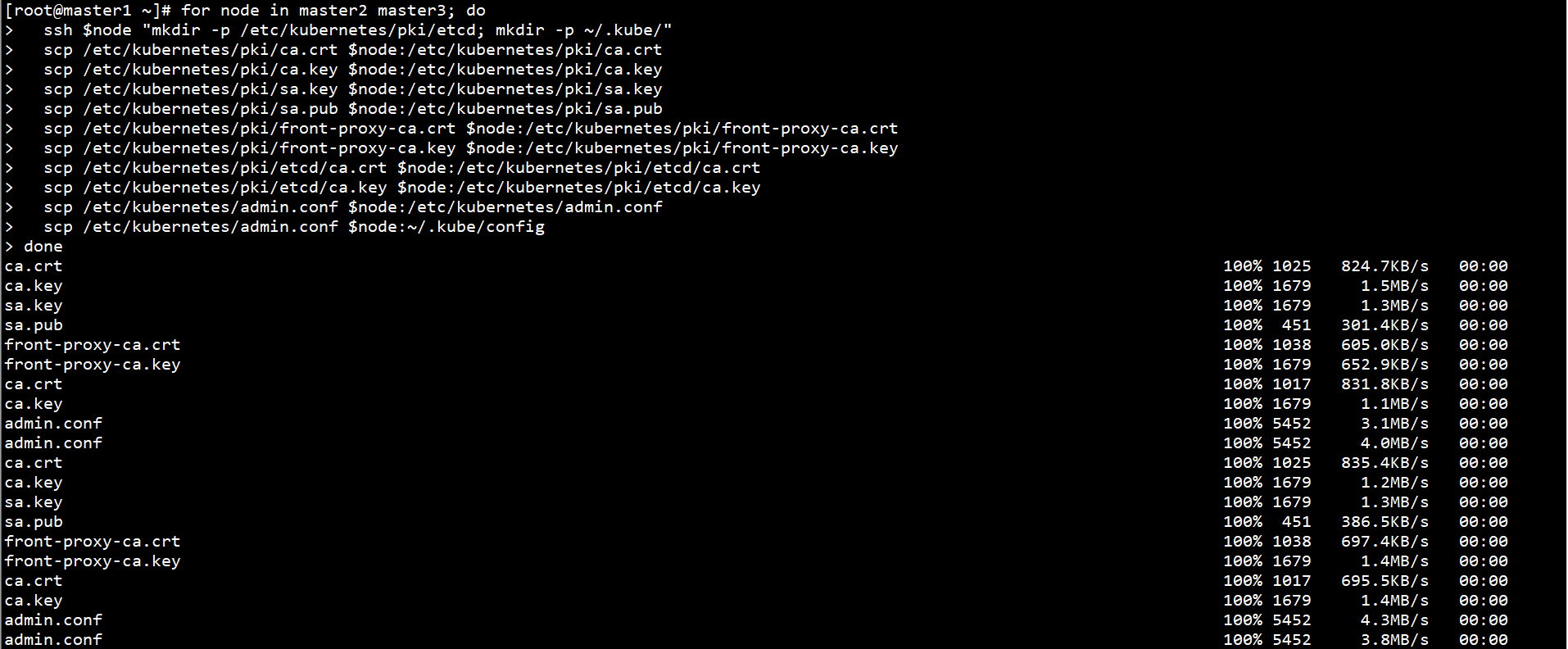
scp /etc/kubernetes/pki/etcd/ca.crt $node:/etc/kubernetes/pki/etcd/ca.crt

scp /etc/kubernetes/pki/etcd/ca.key $node:/etc/kubernetes/pki/etcd/ca.key

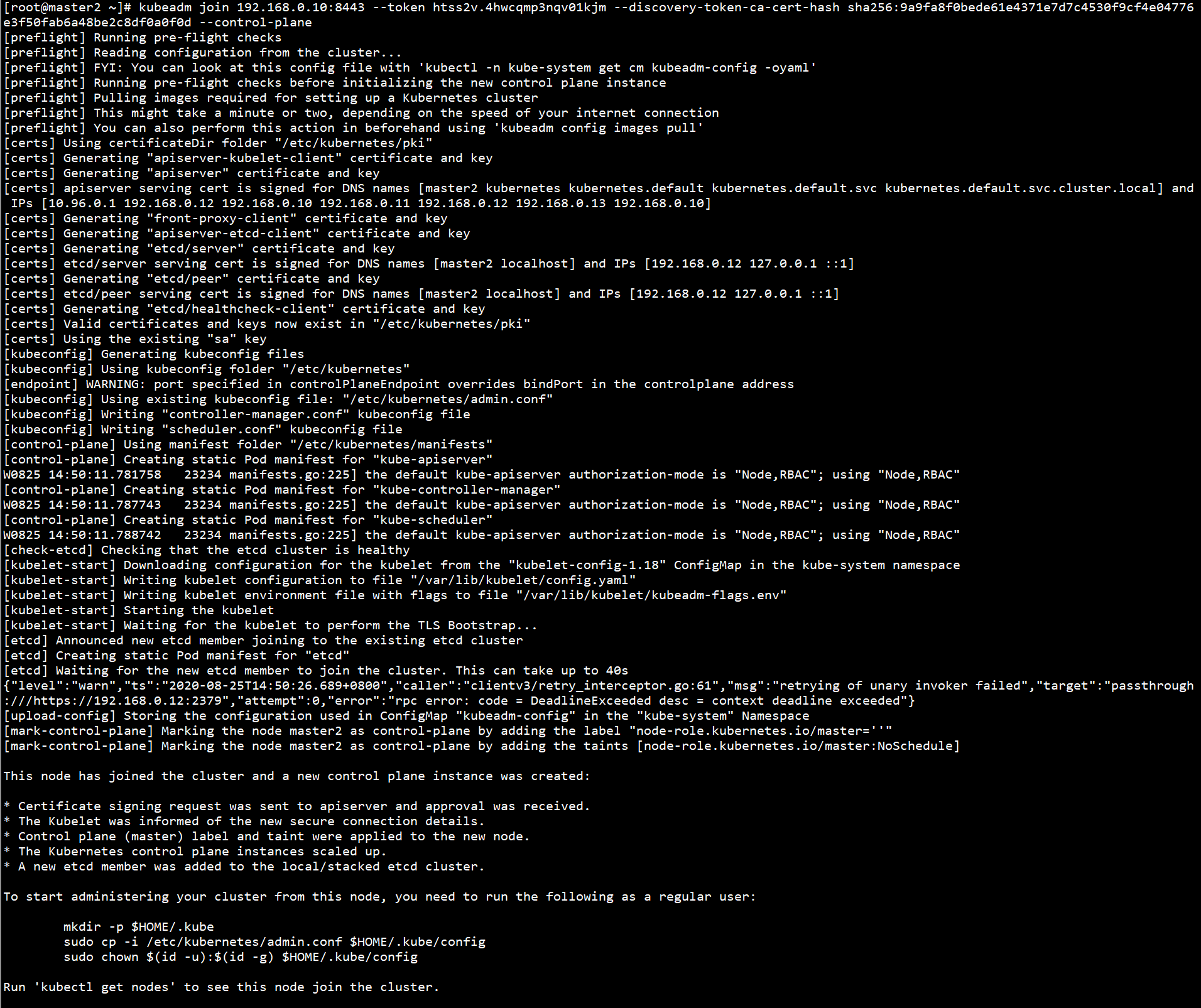
scp /etc/kubernetes/admin.conf $node:/etc/kubernetes/admin.conf

scp /etc/kubernetes/admin.conf $node:~/.kube/config

done



其余Master加入高可用集群：



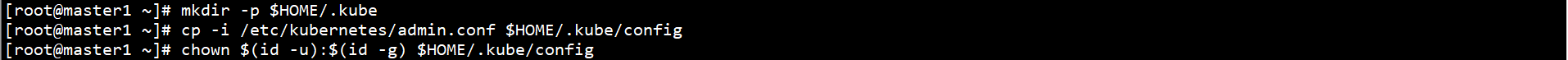
### 6、配置kubectl

在所有Master节点上配置kubectl：

mkdir -p $HOME/.kube

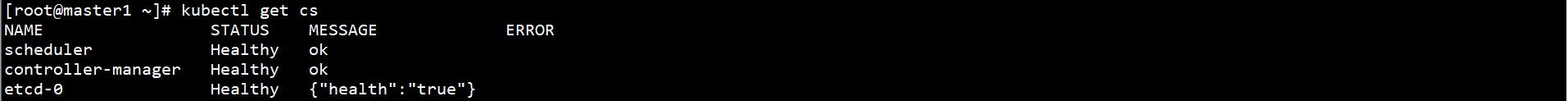
cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

chown $(id -u):$(id -g) $HOME/.kube/config



在任意Master节点上查看集群状态：

kubectl get cs



### 7、安装CNI网络（calico）

在任意Master节点上部署CNI网络：

下载calico部署文件：

下载地址：https://docs.projectcalico.org/manifests/calico.yaml

修改calico.yaml

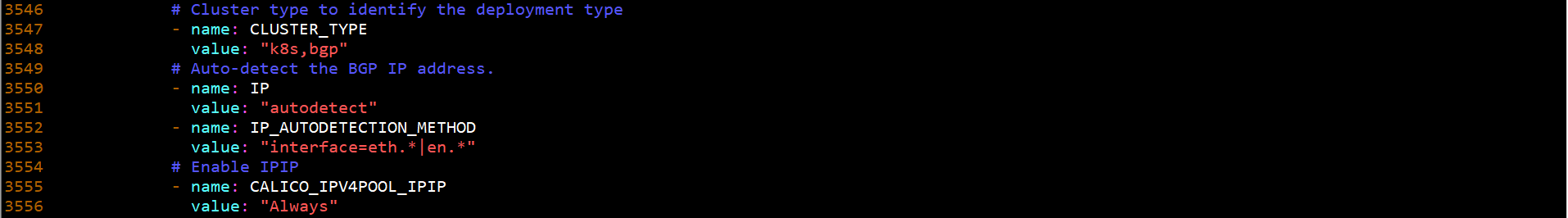
增加

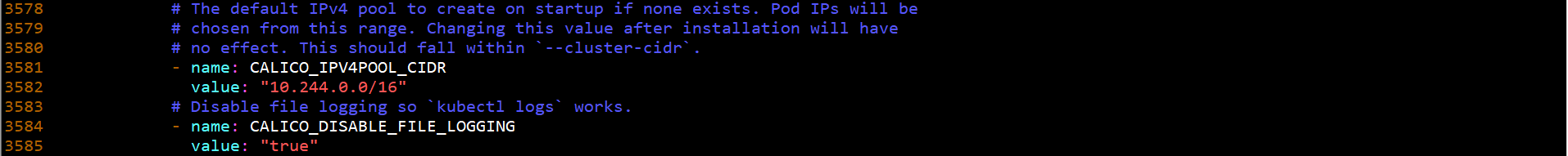
- name: IP\_AUTODETECTION\_METHOD

value: "interface=eth.\*|en.\*"

- name: CALICO\_IPV4POOL\_CIDR

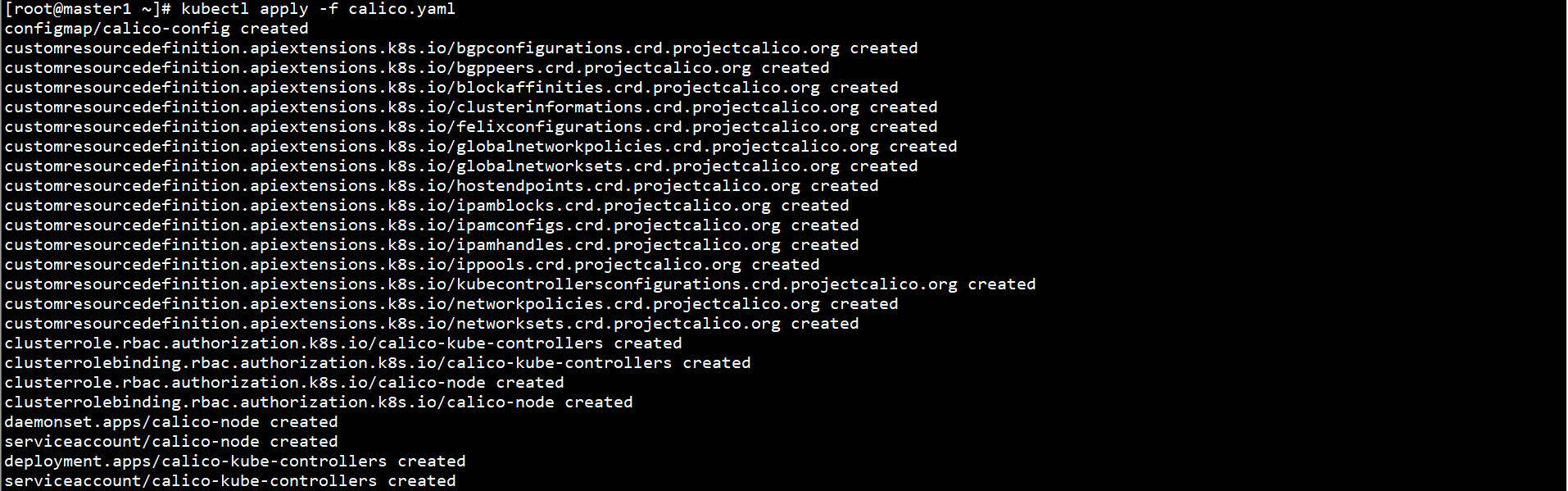
value: "10.244.0.0/16"





部署CNI网络：

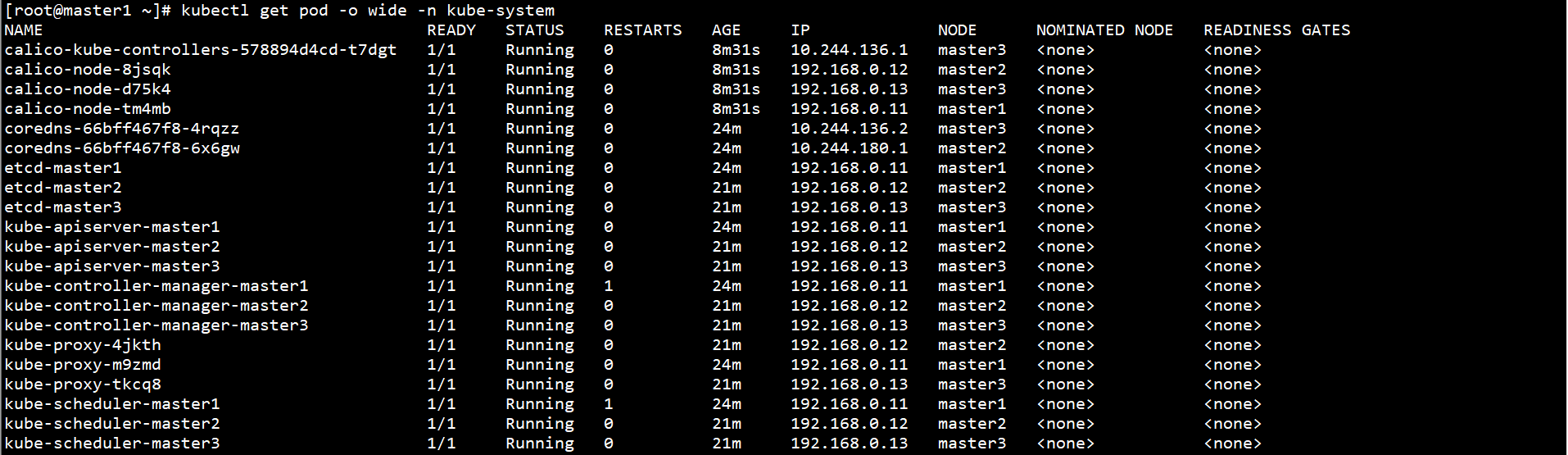
kubectl apply -f calico.yaml



calico.yaml中的CIDR需与初始化集群中的参数一致

查看Pod状态：

kubectl get pod -o wide -n kube-system

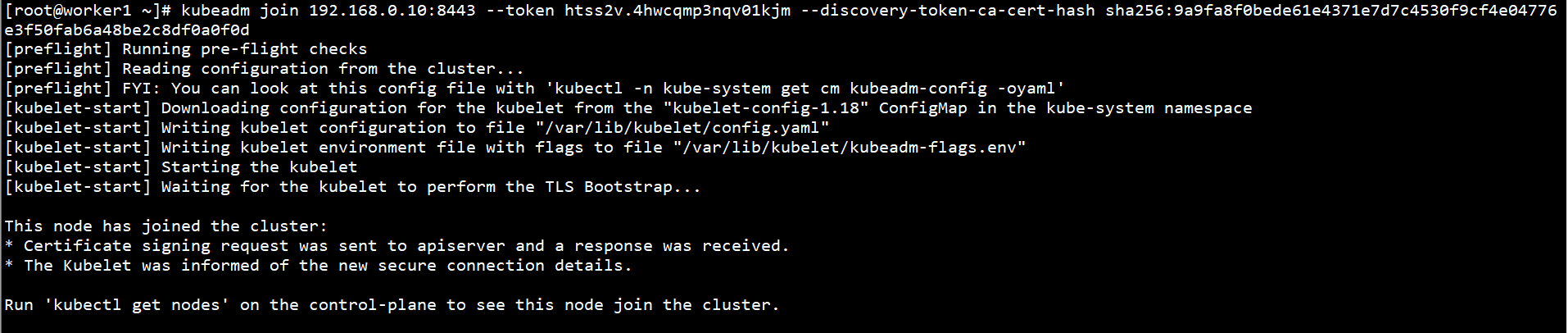


### 8、添加Worker节点

如下操作需在Worker节点执行

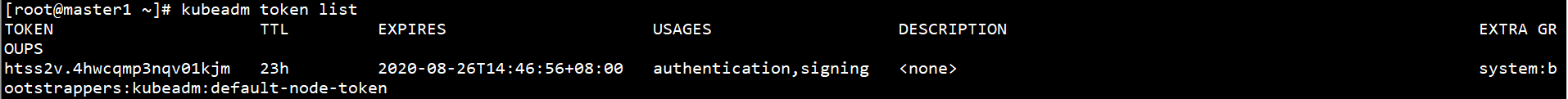
初始化集群时，显示添加节点命令：

kubeadm join 192.168.0.10:8443 --token htss2v.4hwcqmp3nqv01kjm --discovery-token-ca-cert-hash sha256:9a9fa8f0bede61e4371e7d7c4530f9cf4e04776e3f50fab6a48be2c8df0a0f0d

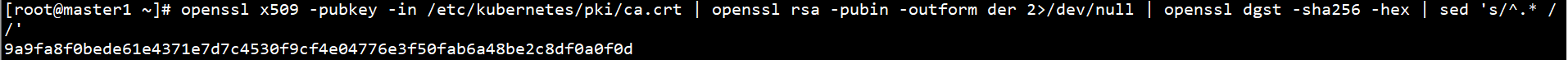


在任意Master节点上通过如下命令token和discovery-token-ca-cert-hash：

kubeadm token list



openssl x509 -pubkey -in /etc/kubernetes/pki/ca.crt | openssl rsa -pubin -outform der 2>/dev/null | openssl dgst -sha256 -hex | sed 's/^.\* //'



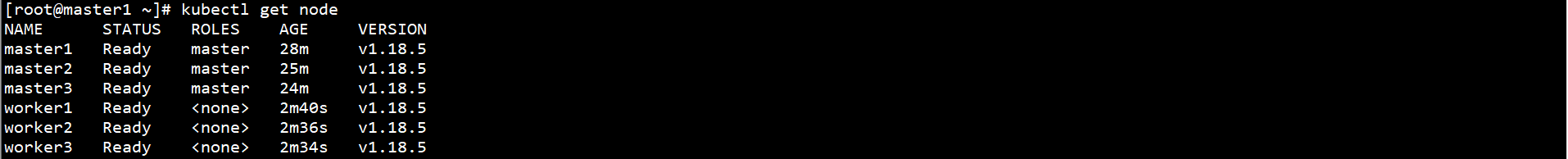
token有效时长为24h，通过如下命令创建：

kubeadm token create



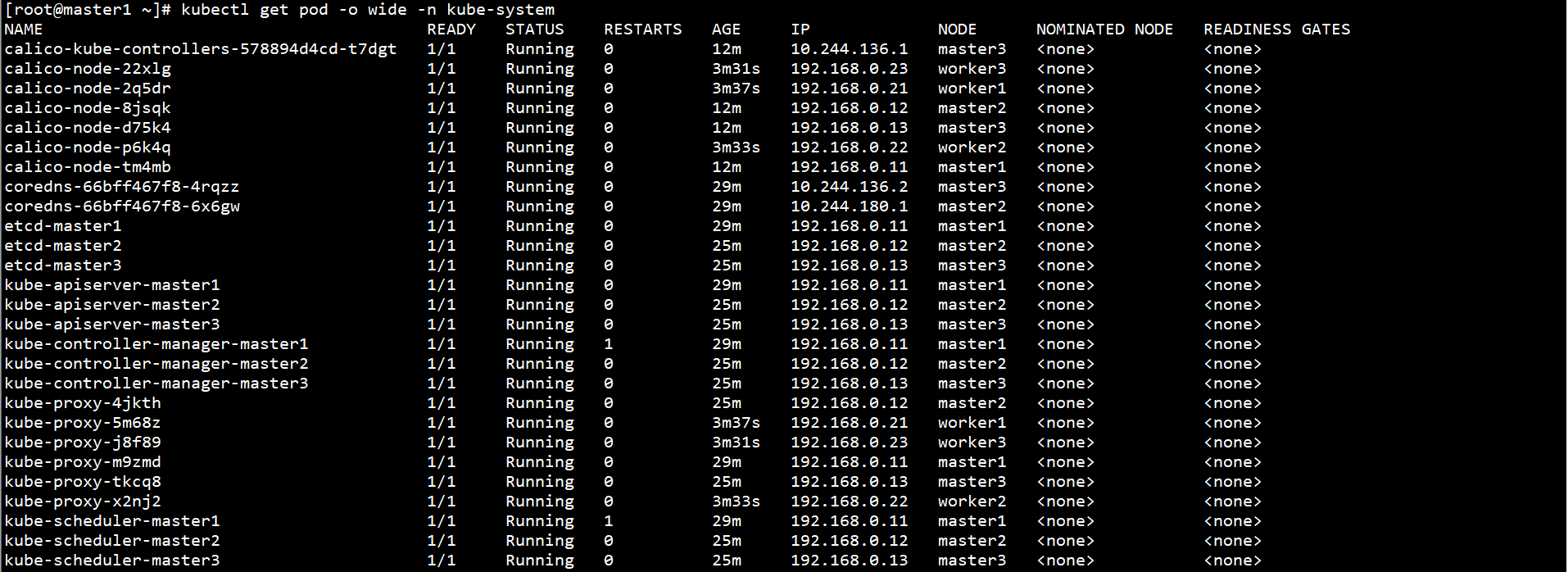
在任意Master节点上查看节点状态：

kubectl get node



在任意Master节点上查看Pod状态：

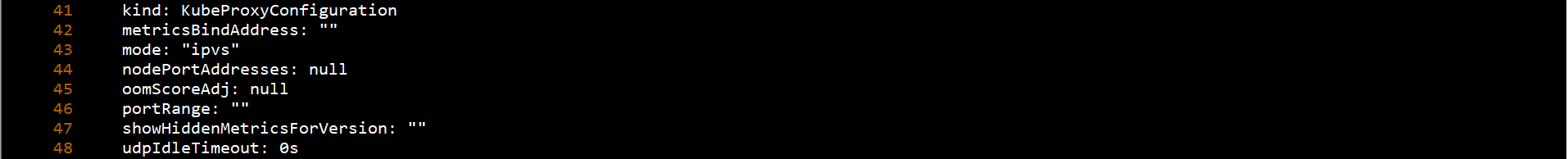
kubectl get pod -o wide -n kube-system



### 9、kube-proxy开启ipvs

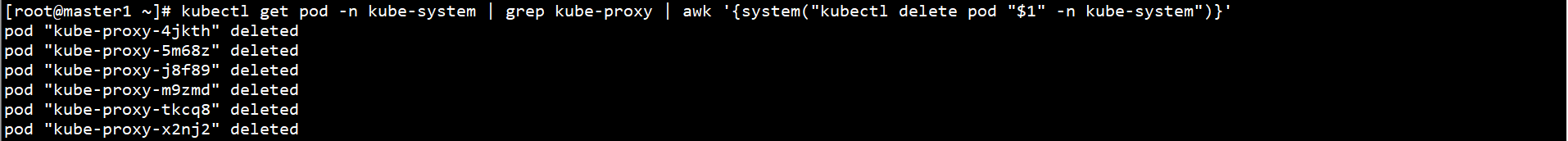
在任意Master节点上修改ConfigMap kube-proxy中的mode: “ipvs”：

kubectl edit configmap kube-proxy -n kube-system



在任意Master节点上重启各个节点上的kube-proxy pod：

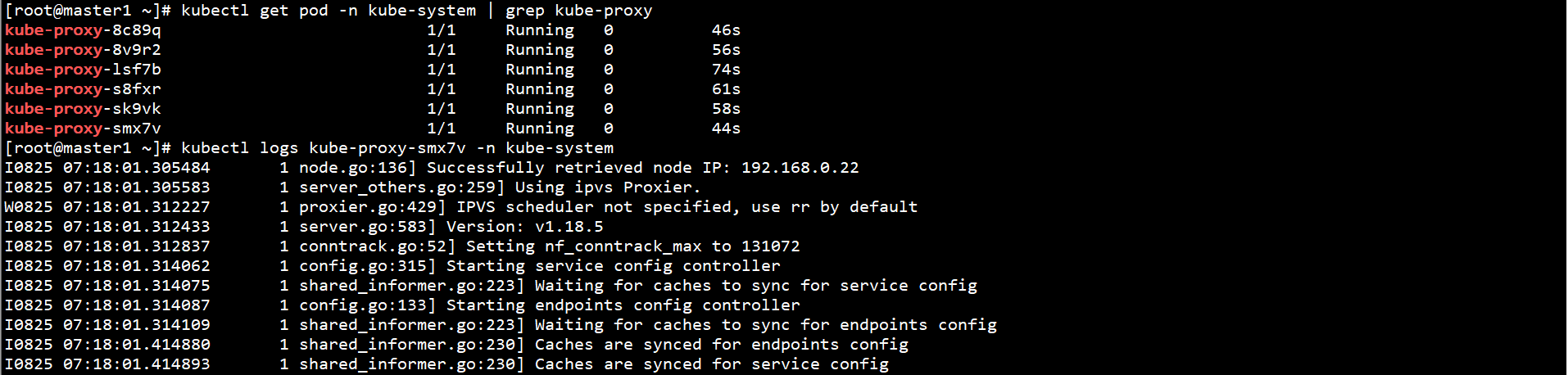
kubectl get pod -n kube-system | grep kube-proxy | awk '{system("kubectl delete pod "$1" -n kube-system")}'



验证修改成功：

kubectl get pod -n kube-system | grep kube-proxy

kubectl logs kube-proxy-qpc8g -n kube-system



日志中打印出了Using ipvs Proxier，说明ipvs模式已经开启