**一准备工作**

**1.1主机名**

[root@XXX ~]# hostnamectl set-hostname XXX

|  |
| --- |
| 主机名列表  192.168.91.100 k8s-master  192.168.91.101 k8s-node1  192.168.91.102 k8s-node2 |

**1.2主机名称解析**

|  |
| --- |
| cat >> /etc/hosts << EOF  192.168.91.100 k8s-master  192.168.91.101 k8s-node1  192.168.91.102 k8s-node2  EOF |

**1.3主机安全配置**

**1.3.1 关闭fifirewalld**

|  |
| --- |
| [root@XXX ~]# systemctl stop firewalld  [root@XXX ~]# systemctl disable firewalld  确认是否运行  [root@XXX ~]# firewall-cmd --state |

**1.3.2 SELINUX配置**

|  |
| --- |
| [root@XXX ~]# sed -ri 's/SELINUX=enforcing/SELINUX=disabled/' /etc/selinux/config # 永久  [root@XXX ~]#setenforce 0 # 临时 |

**1.4主机时间同步**

[root@XXX ~]# yum -y install ntpdate

[root@XXX ~]# crontab -l

0 \*/1 \* \* \* ntpdate time1.aliyun.com

1

**1.5永久关闭swap分区**

使用kubeadm部署必须关闭swap分区，修改配置文件后

需要重启操作系统。

|  |
| --- |
| [root@XXX ~]#swapoff -a # 临时  [root@XXX ~]#sed -ri 's/.\*swap.\*/#&/' /etc/fstab # 永久  在swap文件系统对应的行，行首添加#表示注释。 |

**1.6添加网桥过滤**

**1.6.1添加网桥过滤及地址转发**

|  |
| --- |
| [root@XXX ~]#  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  vm.swappiness = 0  EOF |

#允许 iptables 检查桥接流量

cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf

br\_netfilter

EOF

**1.6.2加载br\_netfilter模块**

|  |
| --- |
| [root@XXX ~]# modprobe br\_netfilter |

**1.6.3查看是否加载**

|  |
| --- |
| [root@XXX ~]# lsmod | grep br\_netfilter |

**1.6.4加载网桥过滤配置文件**

|  |
| --- |
| [root@XXX ~]#sysctl -p /etc/sysctl.d/k8s.conf |

**1.7开启ipvs**

**1.7.1安装ipset及ipvsadm，所有节点。**

|  |
| --- |
| [root@XXX ~]# yum -y install ipset ipvsadm |

**1.7.2 在所有节点执行如下脚本**

|  |
| --- |
| 添加需要加载的模块  （适用于低版本内核3.x的内核）  [root@XXX ~]# 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    （适用于高版本内核5.X的内核，在内核升级情况下）  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  EOF |

**1.7.3授权、运行、检查是否加载**

|  |
| --- |
| [root@XXX ~]#chmod 755 /etc/sysconfig/modules/ipvs.modules && bash /etc/sysconfig/modules/ipvs.modules && lsmod | grep -e ip\_vs -e nf\_conntrack\_ipv4 |

**1.7.4检查是否加载**

|  |
| --- |
| [root@XXX ~]# lsmod | grep -e ipvs -e nf\_conntrack\_ipv4 |

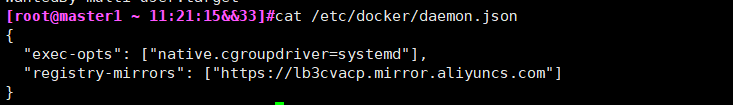
**二、在所有节点安装指定版本的docker-ce**

|  |
| --- |
| # step 1: 安装必要的一些系统工具  [root@XXX ~]#yum install -y yum-utils device-mapper-persistent-data lvm2  # Step 2: 添加软件源信息  [root@XXX ~]#yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo  # Step 3: 更新并安装 Docker-CE  [root@XXX ~]#yum makecache fast  [root@XXX ~]#yum -y install docker-ce  # Step 4: 开启Docker服务  [root@XXX ~]#systemctl start docker  [root@XXX ~]#systemctl enable docker |

**修改docker-ce服务配置文件**

#在/etc/docker/daemon.json添加如下内容

|  |
| --- |
| [root@XXX ~]#mkdir -p /etc/docker  [root@XXX ~]#tee /etc/docker/daemon.json <<-'EOF'  {  "exec-opts": ["native.cgroupdriver=systemd"],  "registry-mirrors": ["https://lb3cvacp.mirror.aliyuncs.com"]  }  EOF  [root@XXX ~]#systemctl daemon-reload  [root@XXX ~]#systemctl restart docker |

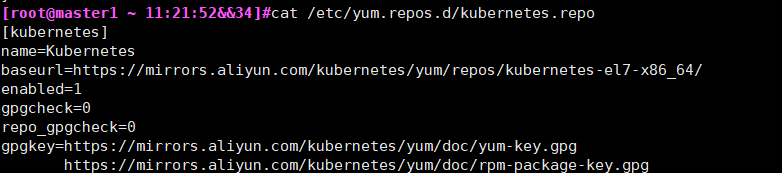


1. **部署软件及配置**

**3.1阿里云YUM源**

编辑/etc/yum.repos.d/kubernetes.repo

|  |
| --- |
| cat > /etc/yum.repos.d/kubernetes.repo << EOF  [kubernetes]  name=Kubernetes  baseurl=https://mirrors.aliyun.com/kubernetes/yum/repos/kubernetes-el7-x86\_64/  enabled=1  gpgcheck=0  repo\_gpgcheck=0  gpgkey=https://mirrors.aliyun.com/kubernetes/yum/doc/yum-key.gpg  https://mirrors.aliyun.com/kubernetes/yum/doc/rpm-package-key.gpg  EOF |



**3.2主要配置kubelet，如果不配置可能会导致k8s集群无法启动。**

|  |
| --- |
| 为了实现docker使用的cgroupdriver与kubelet使用的  cgroup的一致性，建议修改如下文件内容。  [root@XXX ~]# vim /etc/sysconfig/kubelet  KUBELET\_EXTRA\_ARGS="--cgroup-driver=systemd" |

**3.3安装 kubeadm，kubelet 和 kubectl （所有节点）**

|  |
| --- |
| [root@XXX ~]#yum install -y kubelet kubeadm kubectl  指定版本  [root@XXX ~]yum install -y kubelet-1.23.4 kubeadm-1.23.4 kubectl-1.23.4  设置为开机自启动即可，由于没有生成配置文件，集群初始化  后自动启动  [root@XXX ~]# systemctl enable kubelet |

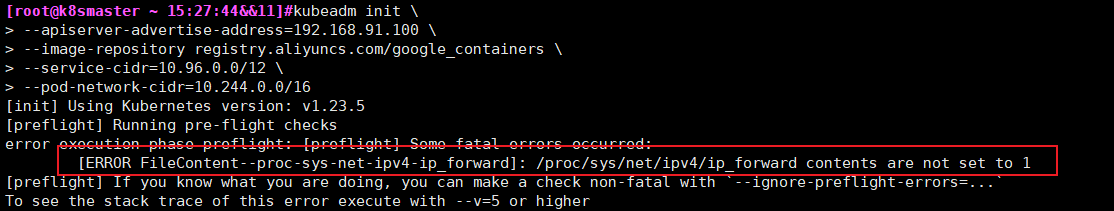
**3.4 部署Kubernetes Master**

**3.4.1在 192.168.91.100（Master）执行**

**所有机器添加master域名映射，以下需要修改为自己的**

echo "172.31.0.4 cluster-endpoint" >> /etc/hosts

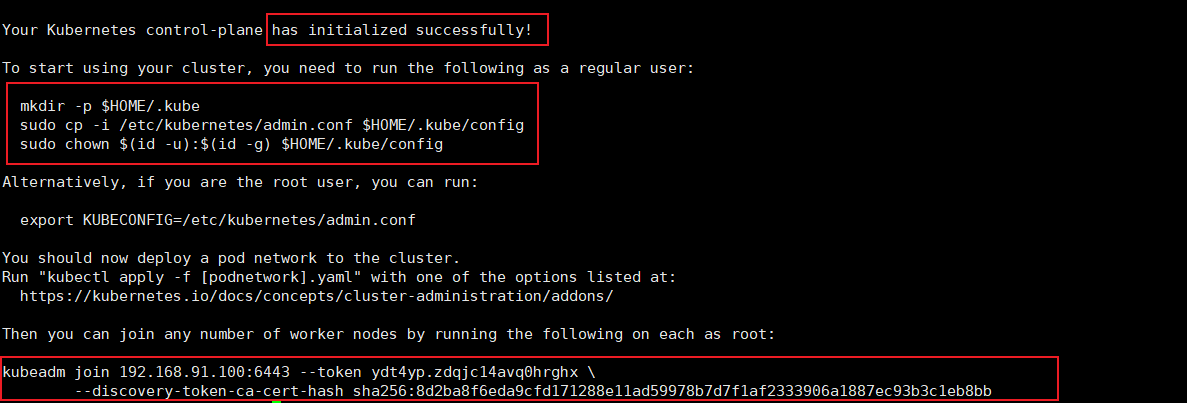
|  |
| --- |
| 不指定版本  kubeadm init \  --apiserver-advertise-address=192.168.91.100 \  --control-plane-endpoint=cluster-endpoint \  --image-repository registry.aliyuncs.com/google\_containers \  --service-cidr=10.96.0.0/12 \  --pod-network-cidr=10.244.0.0/16  指定版本  kubeadm init \  --apiserver-advertise-address=172.28.10.15 \  --control-plane-endpoint=cluster-endpoint \  --image-repository registry.aliyuncs.com/google\_containers \  --kubernetes-version v1.23.4 \  --service-cidr=10.96.0.0/12 \  --pod-network-cidr=10.244.0.0/16 |



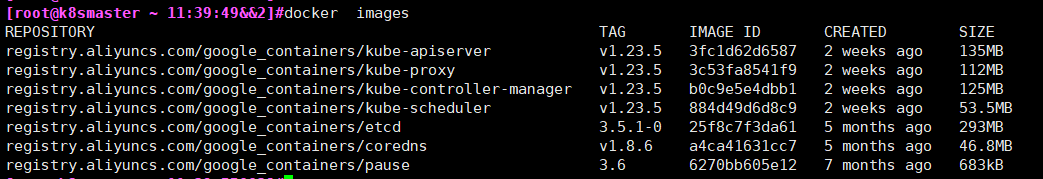
**以上报错重新执行sysctl -p /etc/sysctl.d/k8s.conf**

**由于默认拉取镜像地址 k8s.gcr.io 国内无法访问，这里指定阿里云镜像仓库地址。**

**执行完后会有提示信息以及接下来的操作步骤**



**拉取的相关镜像如下**



**3.4.2准备集群管理文件**

|  |
| --- |
| [root@k8smaster]# mkdir -p $HOME/.kube  [root@k8smaster]# sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config  [root@k8smaster]# sudo chown $(id -u):$(id -g) $HOME/.kube/config |

导出环境变量

export KUBECONFIG=/etc/kubernetes/admin.conf

[root@k8smaster]#kubectl get nodes



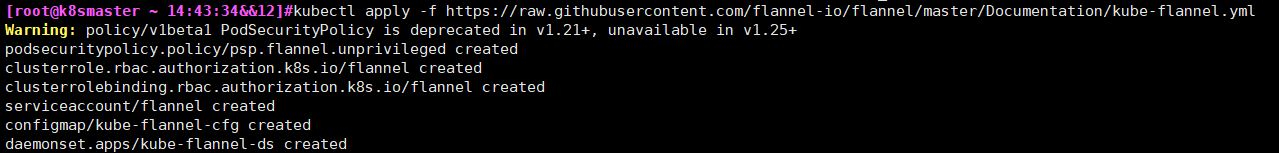
**3.4.3 网络插件CNI**

**3.4.3.1 flannel**

<https://github.com/flannel-io/flannel>

For Kubernetes v1.17+

|  |
| --- |
| [root@k8smaster]#kubectl apply -f https://raw.githubusercontent.com/flannel-io/flannel/master/Documentation/kube-flannel.yml |



**3.4.3.2 CALICO网络插件**

在做三层解决方案时，calico是行业界的大哥，flannel只负责网络通信，而不提供网络策略，而calico即支持网络通信，也支持网络策略。

<https://projectcalico.docs.tigera.io/getting-started/kubernetes/quickstart>

<https://github.com/projectcalico/calico>

[root@k8smaster]#wget <https://github.com/projectcalico/calico/releases/download/v3.22.1/release-v3.22.1.tgz>

#kubectl apply -f <https://docs.projectcalico.org/manifests/calico.yaml> (未验证过)

|  |
| --- |
| sed -i 's@\(.\*\)# \(- name: CALICO\_IPV4POOL\_CIDR\)@\1\2@' ./calico.yaml  sed -i 's@\(.\*\)# value: "192.168.0.0/16"@\1 value: "10.244.0.0/12"@' ./calico.yaml  sed -i 's@docker.io/calico/cni@registry.aliyuncs.com/kubernetesz/calico-cni@' ./calico.yaml  sed -i 's@docker.io/calico/pod2daemon-flexvol@registry.aliyuncs.com/kubernetesz/calico-pod2daemon-flexvol@' ./calico.yaml  sed -i 's@docker.io/calico/node@registry.aliyuncs.com/kubernetesz/calico-node@' ./calico.yaml  sed -i 's@docker.io/calico/kube-controllers@registry.aliyuncs.com/kubernetesz/calico-kube-controllers@' ./calico.yaml |

10.244.0.0/12 是集群 Pod IP 网段

**3.4.3.2.1加载镜像**

|  |
| --- |
| [root@k8smaster]#tar xf release-v3.22.1.tgz  [root@k8smaster]#docker load -i calico-cni.tar  [root@k8smaster]#docker load -i calico-node.tar  [root@k8smaster]#docker load -i calico-pod2daemon.tar  [root@k8smaster]#docker load -i calico-kube-controllers.tar |

**3.4.3.2.2 应用calico资源清单文件（master1上）**

|  |
| --- |
| 在应用caclico资源清单文件之前，需要把calico所有的镜  像导入到node节点中，  最新版不需要修改calico.yaml文件  [root@k8smaster]#kubectl apply -f calico.yaml |

**3.5 加入 Kubernetes Node**

**如果令牌过期**

**kubeadm token create --print-join-command**

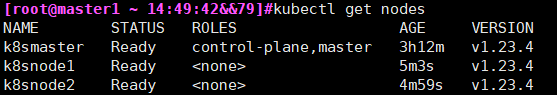


（1）在 192.168.91.101/102（Node）执行

|  |
| --- |
| [root@worker1 ~ 11:38:13&&41]#kubeadm join 192.168.91.100:6443 --token ydt4yp.zdqjc14avq0hrghx \  > --discovery-token-ca-cert-hash sha256:8d2ba8f6eda9cfd171288e11ad59978b7d7f1af2333906a1887ec93b3c1eb8bb |

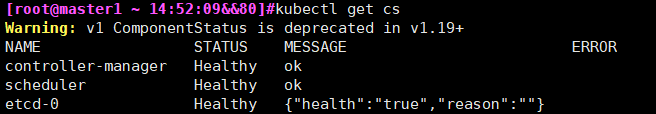
**3.6 测试 kubernetes 集群**

**3.6.1查看节点状态**

[root@k8smaster ~]#kubectl get nodes 

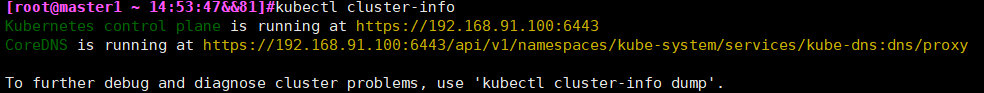
**3.6.2查看集群健康状态**

[root@k8smaster ~]# kubectl get cs



或

[root@k8smaster ~]# kubectl cluster-info

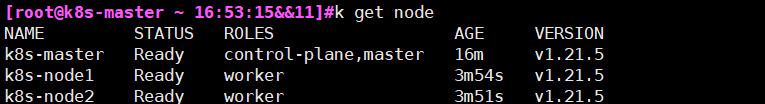


**3.6.3加标签**

kubectl label node k8s-02 node-role.kubernetes.io/worker=''

**3.6.4去标签**

kubectl label node k8s-02 node-role.kubernetes.io/worker-



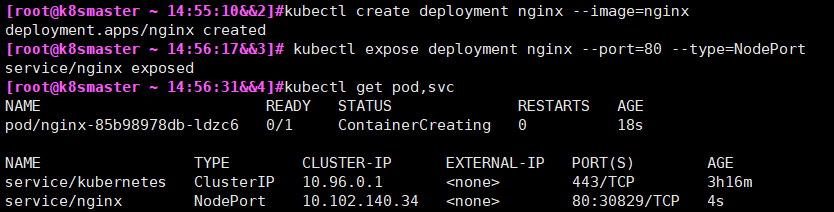
在 Kubernetes 集群中创建一个 pod，验证是否正常运行：

[root@k8smaster ~]# kubectl create deployment nginx --image=nginx

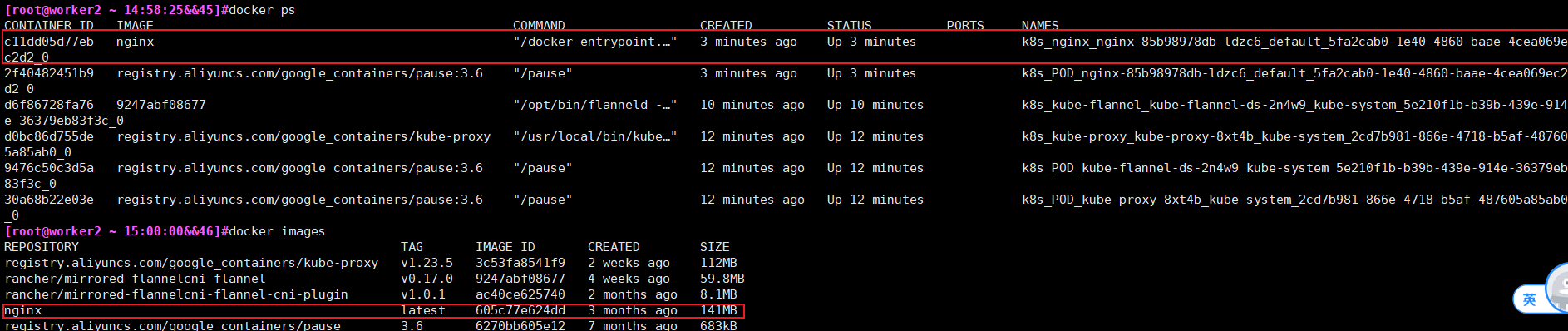
[root@k8smaster ~]#kubectl expose deployment nginx --port=80 --type=NodePort

[root@k8smaster ~]#kubectl get pod,svc

访问地址：<http://NodeIP:Port>



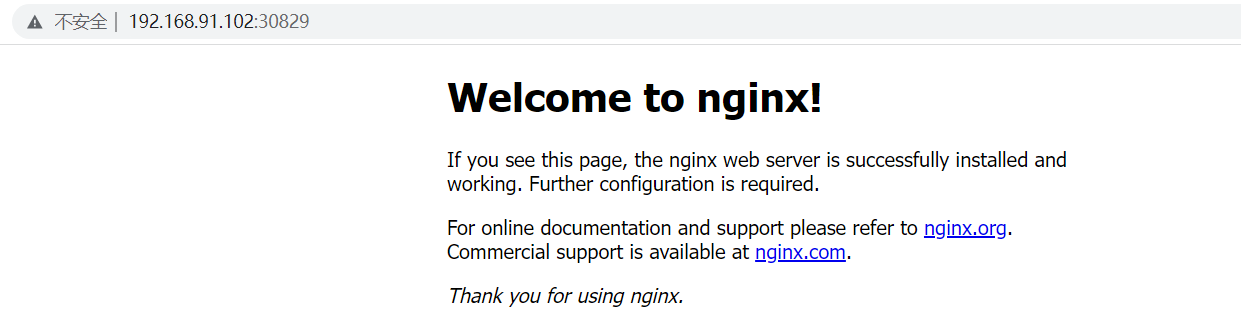




[http://192.168.91.100:30829/](http://192.168.91.102:30829/)

[http://192.168.91.101:30829/](http://192.168.91.102:30829/)

<http://192.168.91.102:30829/>



#### **3.7设置ipvs模式**

k8s整个集群为了访问通；默认是用iptables,性能下（kube-proxy在集群之间同步iptables的内容）

**#1、查看默认kube-proxy 使用的模式**

kubectl logs -n kube-system kube-proxy-28xv4

**#2、需要修改 kube-proxy 的配置文件,修改mode 为ipvs。默认iptables，但是集群大了以后就很慢**

kubectl edit cm kube-proxy -n kube-system

修改如下

ipvs:

excludeCIDRs: null

minSyncPeriod: 0s

scheduler: ""

strictARP: false

syncPeriod: 30s

kind: KubeProxyConfiguration

metricsBindAddress: 127.0.0.1:10249

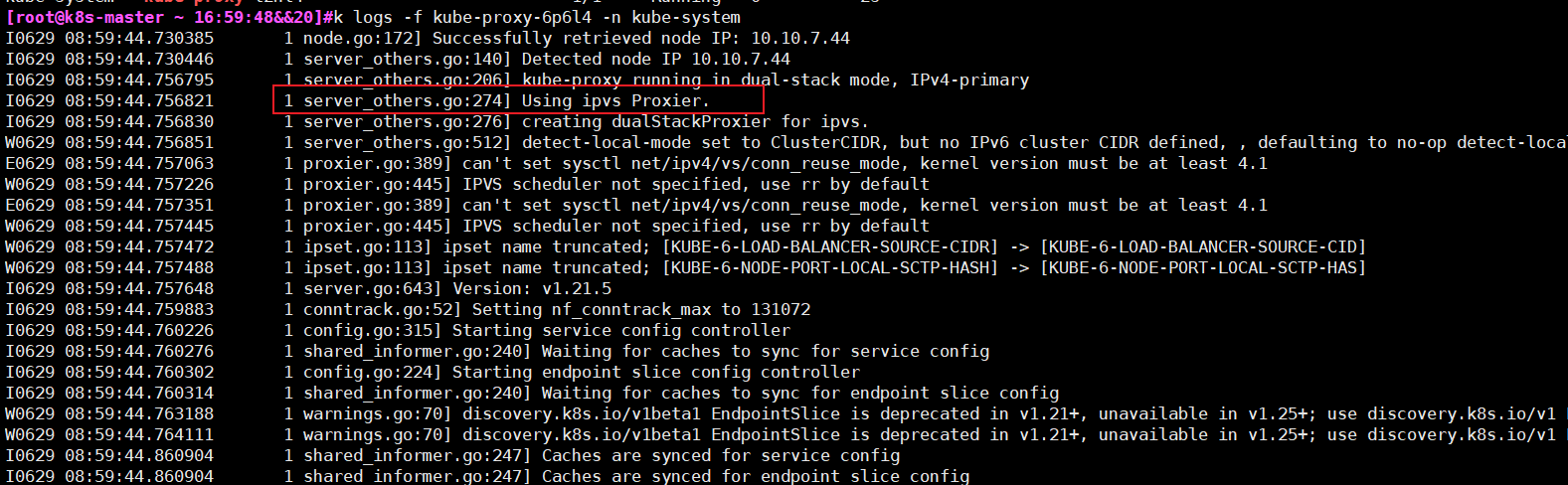
mode: "ipvs"

###修改了kube-proxy的配置，为了让重新生效，需要杀掉以前的Kube-proxy

kubectl get pod -A|grep kube-proxy

kubectl delete pod kube-proxy-pqgnt -n kube-system

### 修改完成后可以重启kube-proxy以生效



**四、在node节点如何使用kubectl命令管理k8s集群？**

|  |
| --- |
| 准备集群管理配置文件  [root@node1 ~]# mkdir .kube  [root@node1 ~]# scp  master1:/root/.kube/config .kube/  使用命令验证  [root@node1 ~]# kubectl get nodes |

**五 重新reset k8s集群**

1. 移除所有工作节点 （master操作）  
   kubectl delete node izwz9ac58lkokssyf8owagz  
   2、所有工作节点删除工作目录，并重置kubeadm （node节点操作）

|  |
| --- |
| rm -rf /etc/kubernetes/\*  rm -rf /var/lib/cni/  rm -fr /etc/cni/net.d kubeadm reset |

3、Master节点删除工作目录，并重置kubeadm （master 节点操作）

|  |
| --- |
| rm -rf /etc/kubernetes/\* rm -rf ~/.kube/\* rm -rf /var/lib/etcd/\*  rm -rf /var/lib/cni/  rm -fr /etc/cni/net.d  kubeadm reset -f |

4、重新init kubernetes

|  |
| --- |
| kubeadm init \  --apiserver-advertise-address=172.28.10.18 \  --control-plane-endpoint=k8s-master \  --image-repository registry.aliyuncs.com/google\_containers \  --kubernetes-version v1.21.5 \  --service-cidr=10.96.0.0/16 \  --pod-network-cidr=192.168.0.0/16 |

**重新生成令牌**

**kubeadm token create --print-join-command**