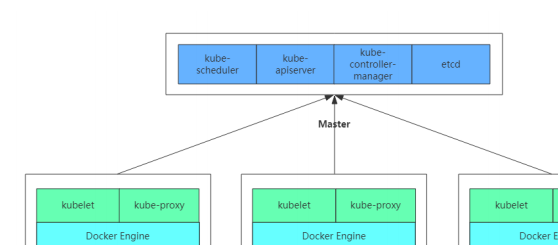
我的运维经理微信公众号：



我的运维技术论坛网站：http://www.jkkcss.cn/

**K8s集群搭建单master和单node集群搭建**

**一、准备实验环境**



1.准备三台centos7虚拟机，用来安装k8s集群，下面是三台虚拟机的配置情况

**master1192.168.1.10）配置：**

操作系统：centos7.4、centos7.5、centos7.6以及更高版本都可以

配置：4核cpu，6G内存，两块40G硬盘

网络：桥接网络

**Node1（192.168.1.11）配置：**

操作系统：centos7.4、centos7.5、centos7.6以及更高版本都可以

配置：4核cpu，6G内存，两块40G硬盘

网络：桥接网络

**二、初始化实验环境**

#### **设置主机名**

hostnamectl set-hostname k8s-master

#### 1、让每台机子可以相互解析 在添加

vi /etc/hosts

192.168.1.10 k8s-master

192.168.1.11 k8s-node

**2.安装基础软件包，各个节点操作**

yum install -y conntrack ntpdate ntp ipvsadm ipset jq iptables curl sysstat libseccomp wget vim net-tools git ntpdate

**3.关闭firewalld防火墙，各个节点操作，centos7系统默认使用的是firewalld防火墙，停止firewalld防火墙，并禁用这个服务**

systemctl stop firewalld && systemctl disable firewalld

swapoff -a && sed -i '/ swap / s/^\(.\*\)$/#\1/g' /etc/fstab

setenforce 0 && sed -i 's/^SELINUX=.\*/SELINUX=disabled/' /etc/selinux/config

**4.时间同步，各个节点操作**

4.1 时间同步

ntpdate cn.pool.ntp.org

4.2 编辑计划任务，每小时做一次同步

crontab -e

\* \*/1 \* \* \* /usr/sbin/ntpdate cn.pool.ntp.org

**5 修改内核参数，各个节点操作**

#### **调整内核参数，对于K8S**

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

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

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

net.ipv4.ip\_forward=1

net.ipv4.ip\_nonlocal\_bind = 1

net.ipv4.tcp\_tw\_recycle=0

vm.swappiness=0 # 禁止使用 swap 空间，只有当系统 OOM 时才允许使用它

vm.overcommit\_memory=1 # 不检查物理内存是否够用

vm.panic\_on\_oom=0 # 开启 OOM

fs.inotify.max\_user\_instances=8192

fs.inotify.max\_user\_watches=1048576

fs.file-max=52706963

fs.nr\_open=52706963

net.ipv6.conf.all.disable\_ipv6=1

EOF

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

#### **升级系统内核为 4.44**

# CentOS 7.x 系统自带的 3.10.x 内核存在一些 Bugs，导致运行的 Docker、Kubernetes 不稳定，查看内核命令uname -r，升级步骤如下：

rpm -Uvh http://www.elrepo.org/elrepo-release-7.0-3.el7.elrepo.noarch.rpm#安装完成后检查 /boot/grub2/grub.cfg 中对应内核 menuentry 中是否包含 initrd16 配置，如果没有，再安装一次!

yum --enablerepo=elrepo-kernel install -y kernel-lt#设置开机从新内核启动

grub2-set-default "CentOS Linux (4.4.182-1.el7.elrepo.x86\_64) 7 (Core)"# 重启

reboot# 查看内核变化啦

1. **安装kubernetes1.18高可用集群**

#### **1、 安装docker**

yum install wget

wget https://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo -O /etc/yum.repos.d/docker-ce.repo

yum -y install docker-ce-18.06.1.ce-3.el7

systemctl enable docker && systemctl start docker #启动并设置开机启动

****配置阿里云镜像docker加速器****

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

{

"registry-mirrors": [ "https://pee6w651.mirror.aliyuncs.com"]

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

}

EOF

systemctl daemon-reload && systemctl restart docker && systemctl enable docker.service

#### **2、 安装Kubeadm、 kubectl和 kubelet**

# 配置K8S的yum源

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

[kubernetes]

name=Kubernetes

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

enabled=1

gpgcheck=0

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

sudo yum install -y kubelet-1.18.0 kubeadm-1.18.0 kubectl-1.18.0

sudo systemctl restart kubelet

sudo systemctl enable kubelet

**部署Kubernetes Master**

kubeadm init \

--apiserver-advertise-address=192.168.1.10 \

--image-repository registry.aliyuncs.com/google\_containers \

--kubernetes-version v1.18.0 \

--service-cidr=10.1.0.0/16 \

--pod-network-cidr=10.244.0.0/16

拷贝一下这里打印出来的1条kubeadm join命令

配置环境变量

mkdir -p $HOME/.kube

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

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

#### **安装calico网络 --master节点**

kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml

kubectl get node -o wide

#### 6、子节点加入

上面组件都安装之后，kubectl get pods -n kube-system -o wide，查看组件安装是否正常，STATUS状态是Running，说明组件正常

发布项目 测试

*kubectl create deployment nginx --image=nginx*

*kubectl expose deployment nginx --port=80 --type=NodePort*

*使用yml文件发布nginx*

*apiVersion: apps/v1*

*kind: Deployment*

*metadata:*

*name: nginx-deployment*

*labels:*

*app: nginx*

*spec:*

*replicas: 4*

*selector:*

*matchLabels:*

*app: nginx*

*template:*

*metadata:*

*labels:*

*app: nginx*

*spec:*

*containers:*

*- name: nginx*

*image: nginx:1.8 #使用镜像nginx:1.8替换原来的nginx:1.7.9*

*ports:*

*- containerPort: 80*

*---*

*apiVersion: v1*

*kind: Service*

*metadata:*

*name: nginx-service #Service 的名称*

*labels: #Service 自己的标签*

*app: nginx #为该 Service 设置 key 为 app，value 为 nginx 的标签*

*spec: #这是关于该 Service 的定义，描述了 Service 如何选择 Pod，如何被访问*

*selector: #标签选择器*

*app: nginx #选择包含标签 app:nginx 的 Pod*

*ports:*

*- name: nginx-port #端口的名字*

*protocol: TCP #协议类型 TCP/UDP*

*port: 80 #集群内的其他容器组可通过 80 端口访问 Service*

*nodePort: 32600 #通过任意节点的 32600 端口访问 Service*

*targetPort: 80 #将请求转发到匹配 Pod 的 80 端口*

*type: NodePort #Serive的类型，ClusterIP/NodePort/LoaderBalancer*