**《部署一套单Master的K8s集群》**

**v1.23，kubeadm方式**

|  |  |
| --- | --- |
| 作者信息 | 李振良（阿良），微信：k8init |
| 阿良教育官网 | [http://www.aliangedu.cn](http://www.aliangedu.cn/) |
| 说明 | 该文档有导航窗格，方便阅读，如果左侧没有显示，请检查word是否启用。  内部学员资料，请勿随意转载。 |



## 1、前置知识点

### 1.1 生产环境可部署Kubernetes集群的两种方式

目前生产部署Kubernetes集群主要有两种方式：

* **kubeadm**

Kubeadm是一个K8s部署工具，提供kubeadm init和kubeadm join，用于快速部署Kubernetes集群。

* **二进制包**

从github下载发行版的二进制包，手动部署每个组件，组成Kubernetes集群。

这里采用kubeadm搭建集群。

kubeadm工具功能：

* **kubeadm init：**初始化一个Master节点
* **kubeadm join：**将工作节点加入集群
* **kubeadm upgrade：**升级K8s版本
* **kubeadm token：**管理 kubeadm join 使用的令牌
* **kubeadm reset：**清空 kubeadm init 或者 kubeadm join 对主机所做的任何更改
* **kubeadm version：**打印 kubeadm 版本
* **kubeadm alpha：**预览可用的新功能

### 1.2 准备环境

服务器要求：

* 建议最小硬件配置：2核CPU、2G内存、20G硬盘
* 服务器最好可以访问外网，会有从网上拉取镜像需求，如果服务器不能上网，需要提前下载对应镜像并导入节点

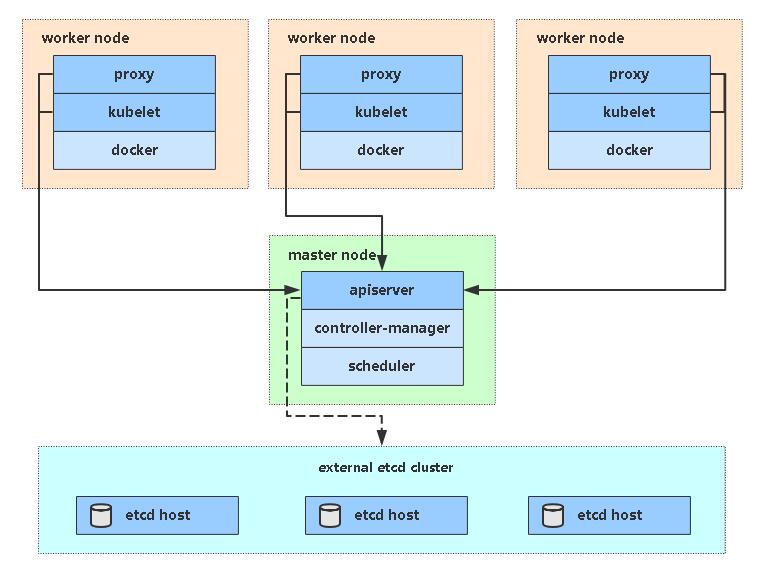
软件环境：

|  |  |
| --- | --- |
| **软件** | **版本** |
| 操作系统 | CentOS7.9\_x64 （mini） |
| Docker | 20-ce |
| Kubernetes | 1.23 |

服务器规划：

|  |  |
| --- | --- |
| 角色 | IP |
| k8s-master | 192.168.31.71 |
| k8s-node1 | 192.168.31.72 |
| k8s-node2 | 192.168.31.73 |

架构图：



### 1.3 操作系统初始化配置【所有节点】

# 关闭防火墙  
systemctl stop firewalld  
systemctl disable firewalld  
  
# 关闭selinux  
sed -i 's/enforcing/disabled/' /etc/selinux/config # 永久  
setenforce 0 # 临时  
  
# 关闭swap  
swapoff -a # 临时  
sed -ri 's/.\*swap.\*/#&/' /etc/fstab # 永久  
  
# 根据规划设置主机名  
hostnamectl set-hostname <hostname>  
  
# 在master添加hosts  
cat >> /etc/hosts << EOF  
192.168.31.71 k8s-master  
192.168.31.72 k8s-node1  
192.168.31.73 k8s-node2  
EOF  
  
# 将桥接的IPv4流量传递到iptables的链  
cat > /etc/sysctl.d/k8s.conf << EOF  
net.bridge.bridge-nf-call-ip6tables = 1  
net.bridge.bridge-nf-call-iptables = 1  
EOF  
sysctl --system # 生效  
  
# 时间同步  
yum install ntpdate -y  
ntpdate time.windows.com

## 2. 安装Docker/kubeadm/kubelet【所有节点】

这里使用Docker作为容器引擎，也可以换成别的，例如containerd

### 2.1 安装Docker

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  
systemctl enable docker && systemctl start docker

配置镜像下载加速器：

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

{

"registry-mirrors": ["https://b9pmyelo.mirror.aliyuncs.com"],

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

}  
EOF

systemctl restart docker  
docker info

### 2.2 添加阿里云YUM软件源

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

### 2.3 安装kubeadm，kubelet和kubectl

由于版本更新频繁，这里指定版本号部署：

yum install -y kubelet-1.23.0 kubeadm-1.23.0 kubectl-1.23.0  
systemctl enable kubelet

## 3. 部署Kubernetes Master

在192.168.31.71（Master）执行。

kubeadm init \  
 --apiserver-advertise-address=192.168.31.71 \  
 --image-repository registry.aliyuncs.com/google\_containers \  
 --kubernetes-version v1.23.0 \  
 --service-cidr=10.96.0.0/12 \  
 --pod-network-cidr=10.244.0.0/16 \  
 --ignore-preflight-errors=all

* --apiserver-advertise-address 集群通告地址
* --image-repository 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
* --kubernetes-version K8s版本，与上面安装的一致
* --service-cidr 集群内部虚拟网络，Pod统一访问入口
* --pod-network-cidr Pod网络，与下面部署的CNI网络组件yaml中保持一致

初始化完成后，最后会输出一个join命令，先记住，下面用。

拷贝kubectl使用的连接k8s认证文件到默认路径：

mkdir -p $HOME/.kube  
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config  
sudo chown $(id -u):$(id -g) $HOME/.kube/config

查看工作节点：

kubectl get nodes  
NAME STATUS ROLES AGE VERSION  
localhost.localdomain NotReady control-plane,master 20s v1.23.0

注：由于网络插件还没有部署，还没有准备就绪 NotReady，先继续

参考资料：

<https://kubernetes.io/zh/docs/reference/setup-tools/kubeadm/kubeadm-init/#config-file>

<https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/create-cluster-kubeadm/#initializing-your-control-plane-node>

## 4. 加入Kubernetes Node

在192.168.31.72/73（Node）执行。

向集群添加新节点，执行在kubeadm init输出的kubeadm join命令：

kubeadm join 192.168.31.71:6443 --token 7gqt13.kncw9hg5085iwclx \  
--discovery-token-ca-cert-hash sha256:66fbfcf18649a5841474c2dc4b9ff90c02fc05de0798ed690e1754437be35a01

默认token有效期为24小时，当过期之后，该token就不可用了。这时就需要重新创建token，可以直接使用命令快捷生成：

kubeadm token create --print-join-command

参考资料：<https://kubernetes.io/docs/reference/setup-tools/kubeadm/kubeadm-join/>

## 5. 部署容器网络（CNI）

Calico是一个纯三层的数据中心网络方案，是目前Kubernetes主流的网络方案。

下载YAML：

wget https://docs.projectcalico.org/manifests/calico.yaml

下载完后还需要修改里面定义Pod网络（CALICO\_IPV4POOL\_CIDR），与前面kubeadm init的 --pod-network-cidr指定的一样。

修改完后文件后，部署：

kubectl apply -f calico.yaml  
kubectl get pods -n kube-system

等Calico Pod都Running，节点也会准备就绪。

注：以后所有yaml文件都只在Master节点执行。

安装目录：/etc/kubernetes/

组件配置文件目录：/etc/kubernetes/manifests/

参考资料：<https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/create-cluster-kubeadm/#pod-network>

## 6. 部署 Dashboard

Dashboard是官方提供的一个UI，可用于基本管理K8s资源。

YAML下载地址：

https://raw.githubusercontent.com/kubernetes/dashboard/v2.4.0/aio/deploy/recommended.yaml

课件中文件名是：kubernetes-dashboard.yaml

默认Dashboard只能集群内部访问，修改Service为NodePort类型，暴露到外部：

vi recommended.yaml  
...  
kind: Service  
apiVersion: v1  
metadata:  
 labels:  
 k8s-app: kubernetes-dashboard  
 name: kubernetes-dashboard  
 namespace: kubernetes-dashboard  
spec:  
 ports:  
 - port: 443  
 targetPort: 8443  
 nodePort: 30001  
 selector:  
 k8s-app: kubernetes-dashboard  
 type: NodePort  
...  
  
kubectl apply -f recommended.yaml  
kubectl get pods -n kubernetes-dashboard

访问地址：<https://NodeIP:30001>

创建service account并绑定默认cluster-admin管理员集群角色：

# 创建用户  
kubectl create serviceaccount dashboard-admin -n kube-system  
# 用户授权  
kubectl create clusterrolebinding dashboard-admin --clusterrole=cluster-admin --serviceaccount=kube-system:dashboard-admin  
# 获取用户Token  
kubectl describe secrets -n kube-system $(kubectl -n kube-system get secret | awk '/dashboard-admin/{print $1}')

使用输出的token登录Dashboard。

## 7. 切换容器引擎为Containerd

参考资料：<https://kubernetes.io/zh/docs/setup/production-environment/container-runtimes/#containerd>

**1、配置先决条件**

cat <<EOF | sudo tee /etc/modules-load.d/containerd.conf  
overlay  
br\_netfilter  
EOF  
  
sudo modprobe overlay  
sudo modprobe br\_netfilter  
  
# 设置必需的 sysctl 参数，这些参数在重新启动后仍然存在。  
cat <<EOF | sudo tee /etc/sysctl.d/99-kubernetes-cri.conf  
net.bridge.bridge-nf-call-iptables = 1  
net.bridge.bridge-nf-call-ip6tables = 1

net.ipv4.ip\_forward = 1  
EOF  
  
sudo sysctl --system

**2、安装containerd**

yum install -y yum-utils device-mapper-persistent-data lvm2  
yum-config-manager \  
 --add-repo \  
 https://download.docker.com/linux/centos/docker-ce.repo  
yum install -y containerd.io  
mkdir -p /etc/containerd  
containerd config default > /etc/containerd/config.toml

**3、修改配置文件**

* pause镜像设置过阿里云镜像仓库地址
* cgroups驱动设置为systemd
* 拉取Docker Hub镜像配置加速地址设置为阿里云镜像仓库地址

vi /etc/containerd/config.toml  
 [plugins."io.containerd.grpc.v1.cri"]  
 sandbox\_image = "registry.aliyuncs.com/google\_containers/pause:3.2"   
 ...  
 [plugins."io.containerd.grpc.v1.cri".containerd.runtimes.runc.options]  
 SystemdCgroup = true  
 ...  
 [plugins."io.containerd.grpc.v1.cri".registry.mirrors."docker.io"]  
 endpoint = ["https://b9pmyelo.mirror.aliyuncs.com"]  
   
systemctl restart containerd

**4、配置kubelet使用containerd**

vi /etc/sysconfig/kubelet   
KUBELET\_EXTRA\_ARGS=--container-runtime=remote --container-runtime-endpoint=unix:///run/containerd/containerd.sock --cgroup-driver=systemd  
  
systemctl restart kubelet

**5、验证**

kubectl get node -o wide  
  
k8s-node1 xxx containerd://1.4.4

**6、管理容器工具**

containerd提供了ctr命令行工具管理容器，但功能比较简单，所以一般会用crictl工具检查和调试容器。

项目地址：<https://github.com/kubernetes-sigs/cri-tools/>

设置crictl连接containerd：

vi /etc/crictl.yaml

runtime-endpoint: unix:///run/containerd/containerd.sock

image-endpoint: unix:///run/containerd/containerd.sock

timeout: 10

debug: false

下面是docker与crictl命令对照表：

|  |  |  |
| --- | --- | --- |
| **镜像相关功能** | **Docker** | **Containerd** |
| 显示本地镜像列表 | docker images | crictl images |
| 下载镜像 | docker pull | crictl pull |
| 上传镜像 | docker push | 无，例如buildk |
| 删除本地镜像 | docker rmi | crictl rmi |
| 查看镜像详情 | docker inspect IMAGE-ID | crictl inspecti IMAGE-ID |

|  |  |  |
| --- | --- | --- |
| **容器相关功能** | **Docker** | **Containerd** |
| 显示容器列表 | docker ps | crictl ps |
| 创建容器 | docker create | crictl create |
| 启动容器 | docker start | crictl start |
| 停止容器 | docker stop | crictl stop |
| 删除容器 | docker rm | crictl rm |
| 查看容器详情 | docker inspect | crictl inspect |
| 附加容器 | docker attach | crictl attach |
| 执行命令 | docker exec | crictl exec |
| 查看日志 | docker logs | crictl logs |
| 查看容器资源 | docker stats | crictl stats |

|  |  |  |
| --- | --- | --- |
| **POD** **相关功能** | **Docker** | **Containerd** |
| 显示 POD 列表 | 无 | crictl pods |
| 查看 POD 详情 | 无 | crictl inspectp |
| 运行 POD | 无 | crictl runp |
| 停止 POD | 无 | crictl stopp |

注：练习完后，建议还切回Docker引擎，就是把kubelet配置参数去掉即可。

**DevOps实战学院：** [http://www.aliangedu.cn](http://www.aliangedu.cn/)