单机环境安装 k8s

author: Uncle Dragon
date: 2021-06-28

初始化 master 节点
安装网络插件
部署一个应用测试是否可用

前面步骤与 之前 k8s 搭建无差别。
只是这次是容器运行时是使用的 **CRI-O**

默认已经安装容器运行时

```

# 腾讯云 docker hub 镜像
# export REGISTRY_MIRROR="https://mirror.ccs.tencentyun.com"
# DaoCloud 镜像
# export REGISTRY_MIRROR="http://f1361db2.m.daocloud.io"
# 阿里云 docker hub 镜像
export REGISTRY_MIRROR=https://registry.cn-hangzhou.aliyuncs.com

export MASTER_IP=10.8.30.7
# 替换 apiserver.demo 为 您想要的 dnsName
export APISERVER_NAME=k8s-master
# Kubernetes 容器组所在的网段，该网段安装完成后，由 kubernetes 创建，事先并不存在于您的物理网络中
export POD_SUBNET=10.244.0.0/16
echo "${MASTER_IP}    ${APISERVER_NAME}" >> /etc/hosts

```

初始化 master 节点

```
apt-get update
apt-get install -y apt-transport-https ca-certificates curl


curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg http://mirrors.aliyun.com/kubernetes/apt/doc/apt-key.gpg
echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] http://mirrors.aliyun.com/kubernetes/apt/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list


apt-get update
apt-get install -y kubelet kubeadm kubectl

apt-mark hold kubelet kubeadm kubectl


# 安装 cri-o 工具 crictl

VERSION="v1.21.0"
wget https://github.com/kubernetes-sigs/cri-tools/releases/download/$VERSION/crictl-$VERSION-linux-amd64.tar.gz
```

```
tar zxvf crictl-$VERSION-linux-amd64.tar.gz -C /usr/local/bin
rm -f crictl-$VERSION-linux-amd64.tar.gz


```

另外我这里还有一步替换 kubeadm 的操作, kubeadm 是 我修改过 证书有效期后，重新编译的。

```
tar zxvf kubernetes-1.21.2.tar.gz
cp kubernetes-1.21.2/kubeadm /usr/bin/kubeadm
```

输出kubeadm 初始化的配置文件

```
kubeadm config print init-defaults --kubeconfig ClusterConfiguration > kubeadm.yml
```

kubeadm-config.yaml

```
---
apiVersion: kubeadm.k8s.io/v1beta2
bootstrapTokens:
- groups:
  - system:bootstrappers:kubeadm:default-node-token
  token: abcdef.0123456789abcdef
  ttl: 24h0m0s
  usages:
  - signing
  - authentication
kind: InitConfiguration
localAPIEndpoint:
  # 改为当前节点ip或者hostname
  advertiseAddress: 10.8.30.7
  bindPort: 6443
nodeRegistration:
  # 改为当前 cri 运行时
  criSocket: /var/run/crio/crio.sock
  name: test
  taints: null
---
apiServer:
  timeoutForControlPlane: 4m0s
apiVersion: kubeadm.k8s.io/v1beta2
certificatesDir: /etc/kubernetes/pki
clusterName: kubernetes
controllerManager: {}
dns:
  type: CoreDNS
  # 改为华为云的镜像地址

```

```
  imageRepository: swr.cn-east-2.myhuaweicloud.com/coredns
  imageTag: 1.8.0
etcd:
  local:
    dataDir: /var/lib/etcd
# 改为 阿里云 k8s 仓库
imageRepository: registry.aliyuncs.com/google_containers
kind: ClusterConfiguration
kubernetesVersion: 1.21.2
networking:
  dnsDomain: cluster.local
  # 服务子网
  serviceSubnet: 10.96.0.0/12
  # pod 子网
  podSubnet: 10.244.0.0/16
scheduler: {}
---
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
# 设置cgroup 驱动
cgroupDriver: systemd
```

```
# 拉取 所需的镜像
kubeadm config images pull --config=kubeadm-config.yaml --v=5

# 初始化 master
kubeadm init --config=kubeadm-config.yaml --upload-certs --v=5

# 或者使用命令初始化
kubeadm init --apiserver-advertise-address=10.8.30.7 --pod-network-cidr=10.244.0.0/16 --service-cidr=10.96.0.0/12 --image-repository=registry.aliyuncs.com/google_containers --kubernetes-version=v1.12.2

```

初始化完成：

```
Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

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

You should now deploy a Pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
```

```
  /docs/concepts/cluster-administration/addons/

You can now join any number of machines by running the following
on each node
as root:

  kubeadm join <control-plane-host>:<control-plane-port> --token
<token> --discovery-token-ca-cert-hash sha256:<hash>
```

创建kubectl使用的kubeconfig文件：

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

查看节点状态

```
$ kubectl get node

NAME   STATUS   ROLES    AGE     VERSION
test   Ready    master   2d21h   v1.21.2
```

可能 master 会 Not Ready

```
修改配置文件
vim /etc/kubernetes/manifests/kube-controller-manager.yaml
vim /etc/kubernetes/manifests/kube-scheduler.yaml

# 把下面这一行内容注释，等待集群自动加载配置，需要时间

#    - --port=0                    ## 注释掉这行

```

设置master参与工作负载

```
kubectl taint nodes --all node-role.kubernetes.io/master-
node/test untainted
```

安装网络插件

```
kubectl create -f https://docs.projectcalico.org/manifests/tigera-operator.yaml

wget https://docs.projectcalico.org/manifests/custom-resources.yaml

sed -i "s#192.168.0.0/16#${POD_SUBNET}#" custom-resources.yaml

kubectl create -f custom-resources.yaml
```

部署一个应用测试是否可用

test-nginx.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myapp
  namespace: default
  labels:
    app: myapp
spec:
  replicas: 1
  revisionHistoryLimit: 3
  selector:
    matchLabels:
      app: myapp
  template:
    metadata:
      labels:
        app: myapp
    spec:
      containers:
      - name: myapp
        image: dr6tjot4.mirror.aliyuncs.com/library/nginx
        imagePullPolicy: IfNotPresent
        ports:
        - containerPort: 80
        resources:
          requests:
            memory: "1000Mi"
            cpu: "500m"
          limits:
            memory: "1000Mi"
            cpu: "500m"
---
apiVersion: v1
kind: Service
metadata:
```

```
  name: myapp
  labels:
    app: myapp
spec:
  ports:
    - port: 80
      targetPort: 80
      nodePort: 30001
  type: NodePort
  selector:
    app: myapp

```

部署：

```
kubectl apply -f test-nginx.yaml
```

查看 pod

```
$ kubectl get po
NAME                     READY   STATUS    RESTARTS   AGE
myapp-79b7f6dd77-4gzkp   1/1     Running   0          47m
```

然后访问：http://10.8.30.7:30001 看是否可以看到 nginx 的欢迎页面。