## Kubernetes使用手册（Ubuntu）

环境为从华为云租用的虚机（N1、N2主机），安装有ubuntu 18.04操作系统。

1. 安装Docker

N1、N2主机

sudo apt-get install apt-transport-https

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

sudo apt update

sudo apt-get install docker-ce docker-ce-cli containerd.io

docker --version

可以看到，所安装docker版本为20.10.14

sudo gpasswd -a $USER docker

groups $USER

newgrp - docker

docker ps

实现免sudo使用docker

**调整docker驱动**

sudo vi /etc/docker/daemon.json

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

sudo systemctl daemon-reload

sudo systemctl restart docker

docker info | grep group

看到Cgroups驱动由cgroups改为systemd

1. 主节点准备镜像

N1主机为主节点

curl https://mirrors.aliyun.com/kubernetes/apt/doc/apt-key.gpg | sudo apt-key add -

在/etc/apt/source.list文件添加：

deb https://mirrors.aliyun.com/kubernetes/apt/ kubernetes-xenial main

sudo apt update

sudo apt install kubelet kubeadm kubectl

kubelet --version

可以看到，待安装k8s版本为1.23.5

kubeadm config images list

看到待安装镜像清单，根据列表更新ksetup.sh脚本

./ksetup.sh

国内镜像拉取到本地，再改tag，如有报错，则手工拉取、手工改tag

1. 工作节点准备镜像

N1主机

a=(k8s.gcr.io/kube-apiserver k8s.gcr.io/kube-proxy k8s.gcr.io/kube-controller-manager k8s.gcr.io/kube-scheduler k8s.gcr.io/etcd k8s.gcr.io/coredns/coredns k8s.gcr.io/pause)

for x in ${a[\*]}; do docker save -o $x.bak $x; done

将镜像导出为文件

局域网拷贝

scp -r hifi@192.168.0.47:kwork/k8s.gcr.io .

N2主机为工作节点

a=(k8s.gcr.io/kube-apiserver k8s.gcr.io/kube-proxy k8s.gcr.io/kube-controller-manager k8s.gcr.io/kube-scheduler k8s.gcr.io/etcd k8s.gcr.io/coredns/coredns k8s.gcr.io/pause)

for x in ${a[\*]}; do docker load --input $x.bak; done

将文件导入为镜像

N2主机

curl https://mirrors.aliyun.com/kubernetes/apt/doc/apt-key.gpg | sudo apt-key add -

在/etc/apt/source.list文件添加：

deb https://mirrors.aliyun.com/kubernetes/apt/ kubernetes-xenial main

sudo apt update

sudo apt install kubelet kubeadm kubectl

kubelet --version

可以看到，待安装k8s版本为1.23.5

1. 其它准备工作

关闭swap内存

sudo vi /etc/fstab

注释掉第二行

sudo reboot

top

看到“KiB Swap: 0”字样

1. 安装Kubernetes

N1主机

初始化

sudo kubeadm init --pod-network-cidr 10.244.0.0/16

记录命令行输出内容，主要是token

注意：kubeadm init/join命令一旦执行失败，请执行下述命令作清理

sudo kubeadm reset

**配置文件**

mkdir -p $HOME/.kube

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

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

kubectl get nodes

可以看到当前节点

创建Flannel网络

kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/a70459be0084506e4ec919aa1c114638878db11b/Documentation/kube-flannel.yml

kubectl apply -f https://raw.githubusercontent.com/flannel-io/flannel/master/Documentation/kube-flannel.yml

N2主机

**配置文件**

mkdir -p $HOME/.kube

scp hifi@n1.hifiax.com:.kube/config $HOME/.kube/

获取join命令，该命令来自N1主机初始化命令后的输出，或者N1主机初始化超过24小时，在N1主机上重新获取

kubeadm token create --print-join-command --ttl=0

加入集群

sudo kubeadm join 192.168.0.47:6443 --token gc2cwe.nlbu4qxs9x8q71x0 \

--discovery-token-ca-cert-hash sha256:d6d7ece9a13c717e40cc72e8732b992d7dda7f7dbffb9c2ebb628ef7a22e06fa

在N1或N2主机执行

kubectl get nodes

可以看到两个节点，且状态均为Ready

1. Registry镜像仓库

N1主机：

docker pull docker.io/registry

/etc/docker/daemon.json文件中添加：

"insecure-registries": ["localhost:5000","192.168.0.47:5000","n1.hifiax.com:5000","124.71.162.55:5000"]

sudo systemctl restart docker

docker run -d -p 5000:5000 -v /home/hifi/rdata:/tmp/registry registry

docker tag jhello localhost:5000/jhello

docker push localhost:5000/jhello

访问http://n1.hifiax.com:5000/v2/\_catalog，看到jhello镜像

docker pull 192.168.0.47:5000/jhello

可以从私有镜像库拉取成功

1. Harbor镜像仓库

安装docker-compose

sudo curl -L https://mirrors.aliyun.com/docker-toolbox/linux/compose/1.21.2/docker-compose-Linux-x86\_64 -o /usr/local/bin/docker-compose

sudo chmod a+x /usr/local/bin/docker-compose

docker-compose –version

可以看到，所安装docker compose版本是1.21.2

1. 部署hello服务

N1主机，~/kwork/java

docker build -t jhello .

docker images

看到jhello镜像

docker run jhello

看到输出，成功运行

kubectl create -f hello.yml

kubectl get pods

1. 部署web服务
2. 部署mysql服务