# 集群搭建

## 机器配置

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| IP地址 | 节点角色 | CPU | Memory | Hostname |
| ~~192.168.5.208~~ | master | >= 2c | >= 2G | Master(c208) |
| ~~192.168.5.217~~ | worker | >= 2c | >= 2G | node1(c217) |
| ~~192.168.5.247~~ | worker | >= 2c | >= 2G | node2(c247) |

## 主机名设置

### 主机名

主机名设置主机名hostname，管理节点设置主机名为 c208

hostnamectl set-hostname c208

### 域名

编辑 /etc/hosts 文件，添加域名解析 (每台机器都要执行)

cat <<EOF >>/etc/hosts

~~192.168.5.208~~ c208

~~192.168.5.217~~ c217

~~192.168.5.247~~ c247

EOF

## 免密登录

### Master生成keygen

设置 master(c208) 可以无密码登录所有节点的 root 账户：

ssh-keygen -t rsa #一路按回车到最后

### 分发到每台机器

将 ssh-keygen 生成的密钥，分别复制到其他三台机器

ssh-copy-id root@c208

ssh-copy-id root@c217

ssh-copy-id root@c247

#### 验证

ssh root@c217

ssh root@c247

## 安装依赖包

在每台机器上安装依赖包：

yum install -y epel-release

yum install -y conntrack ntpdate ntp ipvsadm ipset jq iptables curl sysstat libseccomp wget

/usr/sbin/modprobe ip\_vs

* ipvs 依赖 ipset；
* ntp 保证各机器系统时间同步；

## 关闭防火墙、selinux和swap

在每台机器上关闭防火墙，清理防火墙规则，设置默认转发策略：

systemctl stop firewalld

systemctl disable firewalld

iptables -F && iptables -X && iptables -F -t nat && iptables -X -t nat

iptables -P FORWARD ACCEPT

关闭 SELinux

setenforce 0

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

关闭 swap 分区

swapoff -a

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

### 关闭防火墙相关脚本

systemctl stop firewalld

systemctl disable firewalld

iptables -F && iptables -X && iptables -F -t nat && iptables -X -t nat

iptables -P FORWARD ACCEPT

setenforce 0

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

swapoff -a

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

echo ok

## 配置内核参数

将桥接的IPv4流量传递到iptables的链，swappiness参数调整：

cat <<EOF > /etc/sysctl.d/k8s.conf

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

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

net.ipv4.ip\_forward=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

EOF

modprobe ip\_vs\_rr

modprobe br\_netfilter

sysctl --system

* 由于ipvs已经在内核中，所以为kube-proxy开启ipvs的前提需要加载ip\_vs\_rr内核模块
* br\_netfilter在执行此步骤之前，请确保已加载模块

## 配置yum源

### 阿里软件源

cp /etc/yum.repos.d/CentOS-Base.repo /etc/yum.repos.d/CentOS-Base.repo\_bak

wget -O /etc/yum.repos.d/CentOS-Base.repo http://mirrors.aliyun.com/repo/Centos-7.repo

yum clean all && yum makecache && yum update -y

### 配置国内Kubernetes源

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

[kubernetes]

name=Kubernetes

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

enabled=1

gpgcheck=1

repo\_gpgcheck=1

gpgkey=https://mirrors.aliyun.com/kubernetes/yum/doc/yum-key.gpg https://mirrors.aliyun.com/kubernetes/yum/doc/rpm-package-key.gpg

EOF

### 配置docker源

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

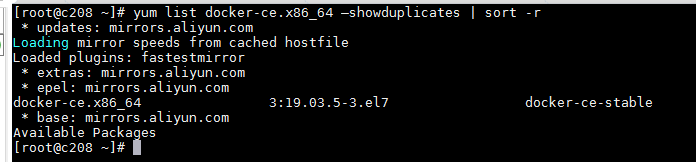
### 执行清理和更新

yum clean all && yum makecache && yum update -y #第一次需要比较久的时间

## 安装Docker

查看最新版Docker

yum list docker-ce.x86\_64 –showduplicates | sort -r



### 指定版本安装

yum install -y docker-ce-18.09.9

systemctl enable docker && systemctl start docker

docker --version

### /etc/docker/daemon.json

不存在，则创建

## Create /etc/docker directory.

mkdir -p /etc/docker

# Setup daemon.

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

{

"registry-mirrors": [

"https://u9r348t0.mirror.aliyuncs.com",

"https://registry.docker-cn.com",

"https://hub-mirror.c.163.com",

"https://docker.mirrors.ustc.edu.cn"

],

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

"log-driver": "json-file",

"log-opts": {

"max-size": "100m",

"max-file": "5"

}

}

EOF

# Restart Docker

systemctl daemon-reload && systemctl restart docker

### daemon.json

{

"registry-mirrors": ["https://uulfe4bd.mirror.aliyuncs.com"],

"insecure-registries":["192.168.1.168:5000","dev227:5000"],

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

"log-driver": "json-file",

"log-opts": {

"max-size": "100m",

"max-file": "5"

}

}

## 使用kubeadm部署k8s集群

### 安装kubeadm、kubelet和kubectl

需要在每台机器上都安装以下的软件包：

* Kubelet：负责与其他节点集群通信，并进行本节点Pod和容器生命周期的管理。
* Kubeadm：是Kubernetes的自动化部署工具，降低了部署难度，提高效率。
* Kubectl：是Kubernetes集群管理工具。

#### 查看可用版本

yum list --showduplicates | grep 'kubeadm\|kubectl\|kubelet'

#### 安装 kubeadm1.15.0, kubelet1.15.0 和 kubectl1.15.0

yum install -y kubelet-1.15.0 kubeadm-1.15.0 kubectl-1.15.0 --disableexcludes=kubernetes

# 此时还不能启动 kubelet，先设置开机启动

systemctl enable --now kubelet

从安装结果可以看出还安装了cri-tools, kubernetes-cni, socat三个依赖：

* socat是kubelet的依赖
* cri-tools是CRI(Container Runtime Interface)容器运行时接口的命令行工具

运行kubelet –help可以看到原来kubelet的绝大多数命令行flag参数都被DEPRECATED了。

而官方推荐我们使用–config指定配置文件，并在配置文件中指定原来这些flag所配置的内容。具体内容可以查看这里[Set Kubelet parameters via a config file](https://kubernetes.io/docs/tasks/administer-cluster/kubelet-config-file/)。这也是Kubernetes为了支持动态Kubelet配置（Dynamic Kubelet Configuration）才这么做的，参考[Reconfigure a Node’s Kubelet in a Live Cluster](https://kubernetes.io/docs/tasks/administer-cluster/reconfigure-kubelet/)。

kubelet的配置文件必须是json或yaml格式，具体可查看[这里](https://github.com/kubernetes/kubernetes/blob/release-1.10/pkg/kubelet/apis/kubeletconfig/v1beta1/types.go)。

#### 配置kubeadm

查看 kubelet 安装了哪些文件？

$ rpm -ql kubelet

/etc/kubernetes/manifests # 清单目录

/etc/sysconfig/kubelet # 配置文件

/etc/systemd/system/kubelet.service # unit file

/usr/bin/kubelet # 主程序

修改/etc/sysconfig/kubelet，加入：

sed -i "s/KUBELET\_EXTRA\_ARGS=/KUBELET\_EXTRA\_ARGS=\"--fail-swap-on=false\"/" /etc/sysconfig/kubelet

# 查看是否修改

cat /etc/sysconfig/kubelet

KUBELET\_EXTRA\_ARGS="--fail-swap-on=false"

安装 kubernetes 主要是安装它的各个镜像，而 kubeadm 已经为我们集成好了运行 kubernetes 所需的基本镜像。但由于国内的网络原因，在搭建环境时，无法拉取到这些镜像。此时我们只需要修改为阿里云提供的镜像服务即可解决该问题。

kubeadm init操作会自动发现master节点的网卡设备，并且将其设置成默认网关。如果想使用另外的网卡设备，那么需要在执行kubeadm init命令之前修改配置文件，修改api.advertiseAddress和api.bindPort两个参数。

api.advertiseAddress=25.10.10.0

api.bindPort=6443

　　这两个参数决定kubeadm join命令中的IP:port，api.advertiseAddress的ip地址必须设置为master节点的内网地址，不然node节点在执行kubeadm join命令的时候，会报错。

#### Kubeadm.yml

apiVersion: kubeadm.k8s.io/v1beta2

bootstrapTokens:

- groups:

- system:bootstrappers:kubeadm:default-node-token

token: abcdef.0123456789abcdef

ttl: 24h0m0s

usages:

- signing

- authentication

kind: InitConfiguration

localAPIEndpoint:

advertiseAddress: 192.168.5.208

bindPort: 6443

nodeRegistration:

criSocket: /var/run/dockershim.sock

name: c208

taints:

- effect: NoSchedule

key: node-role.kubernetes.io/master

---

apiServer:

timeoutForControlPlane: 4m0s

apiVersion: kubeadm.k8s.io/v1beta2

certificatesDir: /etc/kubernetes/pki

clusterName: kubernetes

controllerManager: {}

dns:

type: CoreDNS

etcd:

local:

dataDir: /var/lib/etcd

# imageRepository: k8s.gcr.io

# 国内不能访问 Google 修改为阿里云

imageRepository: registry.aliyuncs.com/google\_containers

kind: ClusterConfiguration

# kubernetesVersion: v1.14.0

kubernetesVersion: v1.15.0

networking:

dnsDomain: cluster.local

podSubnet: 10.244.0.0/16

serviceSubnet: 10.96.0.0/12

scheduler: {}

---

# 开启 IPVS 模式

apiVersion: kubeproxy.config.k8s.io/v1alpha1

kind: KubeProxyConfiguration

featureGates:

SupportIPVSProxyMode: true

mode: ipvs

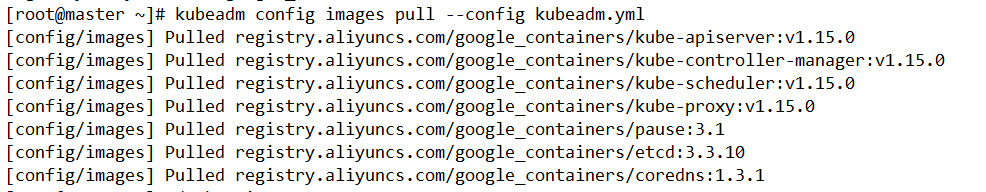
#### 查看和拉取镜像：

# 查看所需镜像列表

kubeadm config images list --config kubeadm.yml

# 拉取镜像

kubeadm config images pull --config kubeadm.yml



#### 使用kubeadm配置master节点

**注：在master节点上进行如下操作**

##### 安装 kubernetes 主节点

执行以下命令初始化主节点，该命令指定了初始化时需要使用的配置文件，其中添加 --experimental-upload-certs 参数可以在后续执行加入节点时自动分发证书文件。追加的 tee kubeadm-init.log 用以输出日志。

kubeadm init --config=kubeadm.yml --upload-certs | tee kubeadm-init.log

# 安装成功则会有如下输出

[init] Using Kubernetes version: v1.15.0

[preflight] Running pre-flight checks

[preflight] Pulling images required for setting up a Kubernetes cluster

[preflight] This might take a minute or two, depending on the speed of your internet connection

[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'

[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"

[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"

…….

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:

https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

# 后面子节点加入需要如下命令

kubeadm join 192.168.100.237:6443 --token abcdef.0123456789abcdef \

--discovery-token-ca-cert-hash sha256:793106d3b4305808d55c3cdb211f89a768bec86ecef64264b131dc8f2548da16

请备份好 kubeadm init 输出中的 kubeadm join 命令，因为您会需要这个命令来给集群添加节点。

如果忘记了 Master 的 Token，可以在 Master 上输入以下命令查看：

kubeadm token list

默认情况下 Token 过期是时间是24小时，如果 Token 过期以后，可以输入以下命令，生成新的 Token

kubeadm token create

--discovery-token-ca-cert-hash 的查看方法，在 Master 运行以下命令查看

openssl x509 -pubkey -in /etc/kubernetes/pki/ca.crt | openssl rsa -pubin -outform der 2>/dev/null | openssl dgst -sha256 -hex | sed 's/^.\* //'

注意：如果安装 kubernetes 版本和下载的镜像版本不统一则会出现 timed out waiting for the condition 错误。中途失败或是想修改配置可以使用 kubeadm reset 命令重置配置，再做初始化操作即可。

###### Token

kubeadm join 192.168.5.208:6443 --token abcdef.0123456789abcdef \

--discovery-token-ca-cert-hash sha256:a9aebdbb109e6477ff29ec8ba92663be9973b77523ca7d221bc880a9b1483493

#### 配置kubectl

执行上面输出的配置：

mkdir -p $HOME/.kube

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

# 非 ROOT 用户执行

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

验证是否成功：

kubectl get node

NAME STATUS ROLES AGE VERSION

master NotReady master 35m v1.15.0

至此主节点配置完成。

#### 配置网络

Kubernetes 中可选的 CNI 插件如下：

* Flannel
* Calico
* Canal
* Weave

接下来安装flannel network add-on：

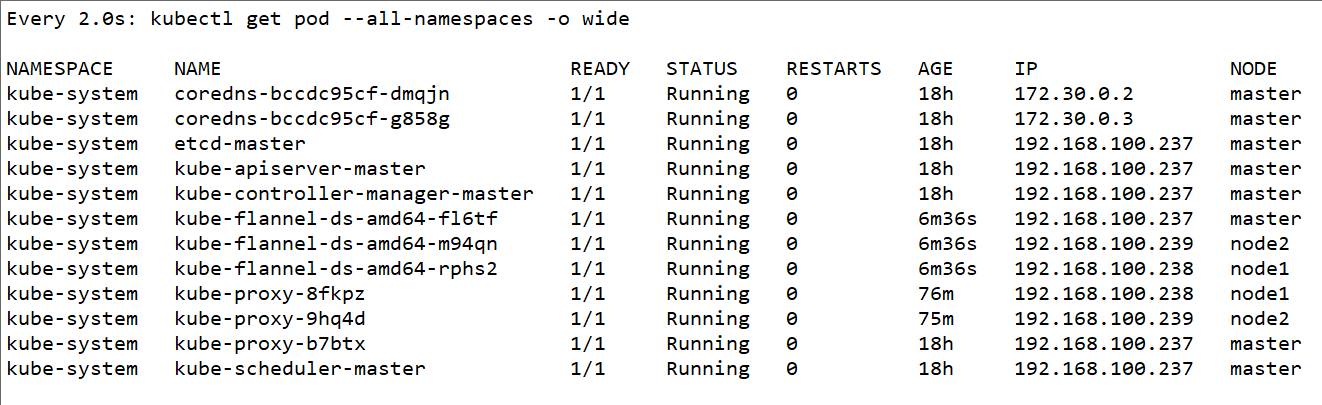
curl -O https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml

kubectl apply -f kube-flannel.yml

使用kubectl get pod –all-namespaces -o wide确保所有的Pod都处于Running状态。

watch kubectl get pod --all-namespaces -o wide

需要等待所有状态为 Running，注意时间可能较久，3 - 5 分钟的样子



至此基本环境已部署完毕。

#### 验证集群DNS是否可用

kubectl run curl --image=radial/busyboxplus:curl -i --tty

# 测试集群dns解析和外网解析

nslookup kubernetes.default

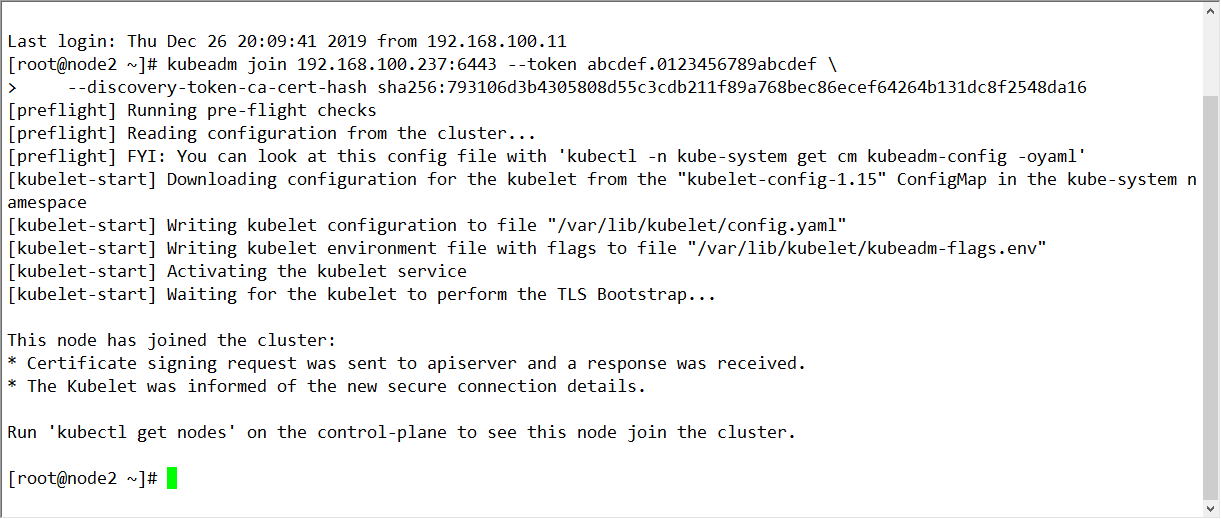
nslookup www.baidu.com.

### 使用kubeadm配置worker节点

kubeadm join 192.168.100.237:6443 --token abcdef.0123456789abcdef \

--discovery-token-ca-cert-hash sha256:793106d3b4305808d55c3cdb211f89a768bec86ecef64264b131dc8f2548da16

安装成功将看到如下信息：



### 部署Dashboard

注：在master节点上进行如下操作

#### 浏览器

需要使用 NodeIP:30001 访问 Dashboard，因为证书原因除火狐浏览器外其它浏览器无法直接打开页面