使用kubeadm安装K8s集群

Kubeadm是Kubernetes官方推出的集群管理工具，在K8s 1.13版本后已经可以在生产环境中使用，但是需要注意证书的过期问题。Kubeadm提供kubeadm init 和kubeadm join，用于快速部署Kubernetes集群，极大地简化了集群的搭建过程。

1.安装要求

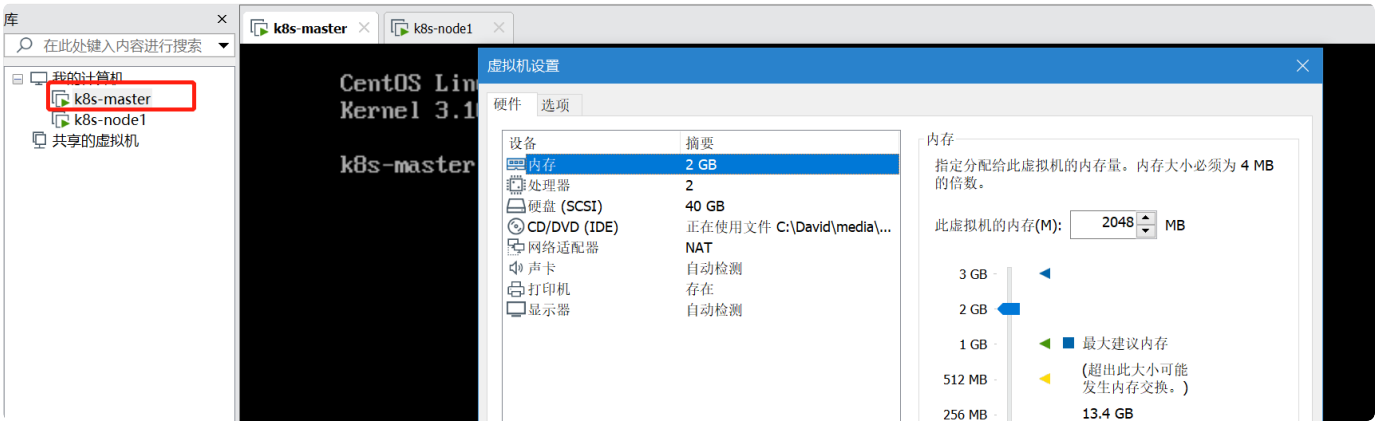
- 节点操作系统需要满足：
  - Ubuntu 16.04+
  - Debian 9+
  - CentOS 7
  - Red Hat Enterprise Linux (RHEL) 7
  - Fedora 25+
  - HypriotOS v1.0.1+
  - Flatcar Container Linux (tested with 2512.3.0)
- 每台机器有2G或更多的内存(少于2G就没有多少空间留给应用了)
- 至少2个CPU
- 集群中所有机器之间的全网络连接(公共网络或私有网络都可以)。
- 每个节点拥有唯一的主机名、MAC地址以及product_uuid
- 机器上需要打开一些端口
- kubelet要想正常工作，必须禁用Swap分区

2.环境准备

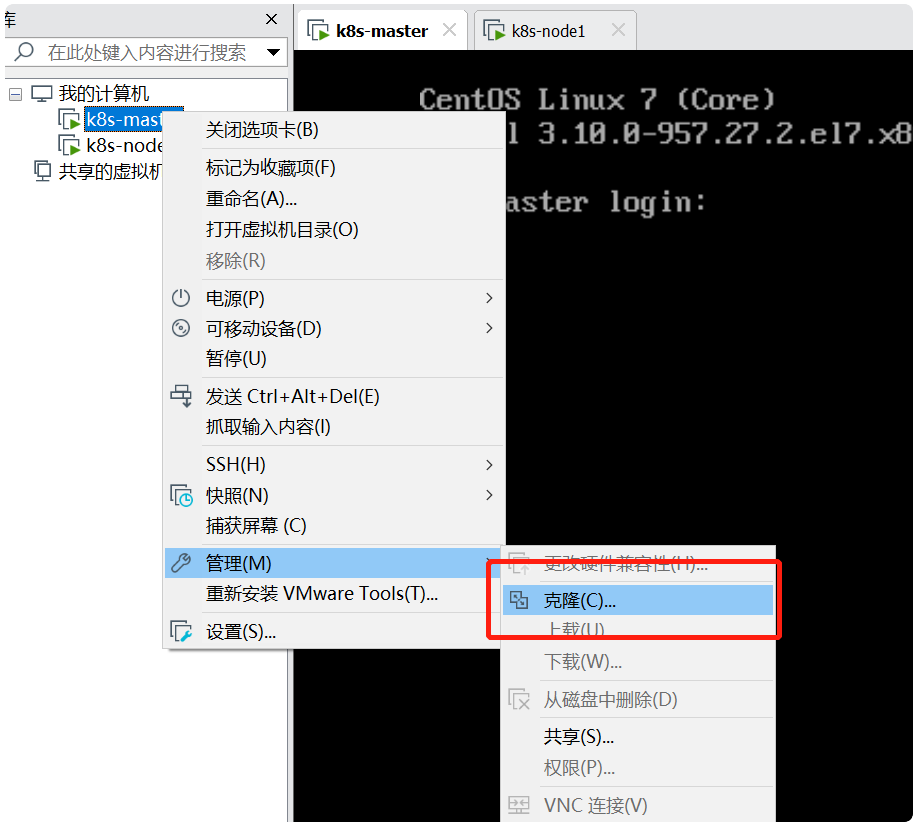
为了演示目的，我搭建一个只包含一个master和一个node的集群：

机器	IP	内存	CPU	硬盘	操作系统
master节点	192.168.188.131	2G	2c	40G	centos-release-7-6
node1节点	192.168.188.132	2G	2c	40G	centos-release-7-6

准备两台虚拟机，一台用作master，一台用作node：



如果希望再添加一个节点，可以通过克隆的方式创建并修改IP：



vim /etc/sysconfig/network-scripts/ifcfg-ens33

```
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="static"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="ens33"
UUID="f6d8a308-2e45-4f84-8693-b5d5a5afa146"
DEVICE="ens33"
ONBOOT="yes"
ZONE=public

NM_CONTROLLED=no
IPADDR=192.168.188.131
NETMASK=255.255.255.0
GATEWAY=192.168.188.2
```

service network restart 重启网络服务

3.环境初始化

所有以下操作,，除非特别说明，都需要在master和node上执行：

关闭防火墙

systemctl stop firewalld

systemctl disable firewalld

关闭selinux

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

关闭swap分区

swapoff –a # 临时

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

设置主机名并添加映射

分别在master和node上设置各自的主机名:

hostnamectl set–hostname k8s–master

hostnamectl set–hostname k8s–node1

在master上添加hosts映射:

cat >> /etc/hosts << EOF

192.168.188.131 k8s–master

192.168.188.132 k8s–node1

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

sudo sysctl ––system

设置系统时区并同步时间服务器

yum install ntpdate –y

ntpdate time.windows.com

4.安装容器运行时

在所有机器上安装容器运行时。

此处以Docker为例子:

下载并安装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–19.03.1.ce

systemctl enable docker && systemctl start docker

配置镜像加速器

sudo mkdir –p /etc/docker

sudo tee /etc/docker/daemon.json <<–'EOF'

{

  "registry–mirrors": ["https://3q4hxrrx.mirror.aliyuncs.com"]

```
}
EOF
sudo systemctl daemon-reload
sudo systemctl restart docker
```

5.添加阿里云Kubernetes YUM源

在所有机器上执行：

```
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
```

6.安装kubelet、kubeadm和kubectl

- kubeadm: 该命令主要用于启动集群。
- kubelet: 该组件运行在集群中所有机器上，负责启动pods和容器等事务。
- kubectl: 该命令行工具负责与集群进行交互。

在所有机器上执行：

```
yum install -y kubelet-1.18.0 kubeadm-1.18.0 kubectl-1.18.0
systemctl enable kubelet
```

7.部署Kubernetes Master节点

在master上执行：

```
kubeadm init \
--apiserver-advertise-address=192.168.188.131 \
--image-repository registry.aliyuncs.com/google_containers \
--kubernetes-version v1.18.0 \
--service-cidr=10.96.0.0/12 \
--pod-network-cidr=10.244.0.0/16
```

此处将镜像仓库地址指定为阿里云，--service-cidr和--pod-network-cidr在保证地址不冲突的情况下可以按自己需求指定。

```
[root@k8s-master yum.repos.d]# kubeadm init \
> --apiserver-advertise-address=192.168.188.131 \
> --image-repository registry.aliyuncs.com/google_containers \
> --kubernetes-version v1.18.0 \
> --service-cidr=10.96.0.0/12 \
> --pod-network-cidr=10.244.0.0/16
W0102 16:42:22.613539   13777 configset.go:202] WARNING: kubeadm cannot validate component configs for API groups [kubelet.config.k8s.io kubeproxy.
[init] Using Kubernetes version: v1.18.0
[preflight] Running pre-flight checks
        [WARNING IsDockerSystemdCheck]: detected "cgroupfs" as the Docker cgroup driver. The recommended driver is "systemd". Please follow the gui
rnetes.io/docs/setup/cri/
[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"
[kubelet-start] Starting the kubelet
```

```
[bootstrap-token] configured RBAC rules to allow Node Bootstrap tokens to get nodes
[bootstrap-token] configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certif
[bootstrap-token] configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap T
[bootstrap-token] configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

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:

kubeadm join 192.168.188.131:6443 --token n2ntd0.afns75z92axnx6wc \
    --discovery-token-ca-cert-hash sha256:a88482632397fbe978d2104eec511609c0228b16f54199713d0586df907faf10
[root@k8s-master yum.repos.d]#
```

根据输出的提示执行：

mkdir –p $HOME/.kube

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

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

执行到这一步后，可以通过kubectl get nodes查看节点状态：

```
[root@k8s-master yum.repos.d]# kubectl get nodes
NAME         STATUS     ROLES    AGE     VERSION
k8s-master   NotReady   master   5m20s   v1.18.0
[root@k8s-master yum.repos.d]#
```

8.加入Kubernetes Node

在Node1节点上执行：

kubeadm join 192.168.188.131:6443 ––token n2ntd0.afns75z92axnx6wc \

––discovery–token–ca–cert–hash

sha256:a88482632397fbe978d2104eec511609c0228b16f54199713d0586df907faf10

默认token的有效期为24小时，当过期之后，该token就不可用了，

如果后续有nodes节点加入，解决方法如下：

重新生成新的token

kubeadm token create

获取ca证书sha256编码hash值

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

```
[root@k8s-node1 yum.repos.d]# kubeadm join 192.168.188.131:6443 --token n2ntd0.afns75z92axnx6wc \
>     --discovery-token-ca-cert-hash sha256:a88482632397fbe978d2104eec511609c0228b16f54199713d0586df907faf10
W0102 16:53:03.371249   15896 join.go:346] [preflight] WARNING: JoinControlPane.controlPlane settings will be ignored when control-
[preflight] Running pre-flight checks
        [WARNING IsDockerSystemdCheck]: detected "cgroupfs" as the Docker cgroup driver. The recommended driver is "systemd". Pleas
rnetes.io/docs/setup/cri/
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -oyaml'
[kubelet-start] Downloading configuration for the kubelet from the "kubelet-config-1.18" ConfigMap in the kube-system namespace
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

[root@k8s-node1 yum.repos.d]#
```

```
[root@k8s-master yum.repos.d]# kubectl get nodes
NAME         STATUS     ROLES    AGE    VERSION
k8s-master   NotReady   master   8m9s   v1.18.0
k8s-node1    NotReady   <none>   51s    v1.18.0
[root@k8s-master yum.repos.d]#
```

9.安装CNI网络插件

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

显示如下错误信息：

```
[root@k8s-master ~]# kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
The connection to the server raw.githubusercontent.com was refused - did you specify the right host or port?
```

通过https://githubusercontent.com.ipaddress.com/raw.githubusercontent.com地址查询到raw.githubusercontent.com的IP。

vim /etc/hosts

加入：

199.232.96.133 raw.githubusercontent.com

```
[root@k8s-master ~]# kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
podsecuritypolicy.policy/psp.flannel.unprivileged created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
```

再次查看集群节点状态：

```
[root@k8s-master ~]# kubectl get nodes
NAME         STATUS   ROLES    AGE   VERSION
k8s-master   Ready    master   32m   v1.18.0
k8s-node1    Ready    <none>   24m   v1.18.0
[root@k8s-master ~]#
```

10.测试Kubernetes集群

kubectl create deployment nginx --image=nginx

```
[root@k8s-master ~]# kubectl create deployment nginx --image=nginx
deployment.apps/nginx created
```

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

```
[root@k8s-master ~]# kubectl expose deployment nginx --port=80 --type=NodePort
service/nginx exposed
```

kubectl get pods,svc

```
[root@k8s-master ~]# kubectl get pods,svc
NAME                         READY   STATUS              RESTARTS   AGE
pod/nginx-f89759699-wxvvq    0/1     ContainerCreating   0          8s

NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   33m
```

```
[root@k8s-master ~]# kubectl get pods,svc
NAME                         READY   STATUS    RESTARTS   AGE
pod/nginx-f89759699-wxvvq    1/1     Running   0          50s

NAME                 TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
service/kubernetes   ClusterIP   10.96.0.1        <none>        443/TCP        34m
service/nginx        NodePort    10.109.254.224   <none>        80:30443/TCP   29s
[root@k8s-master ~]#
```

访问服务：