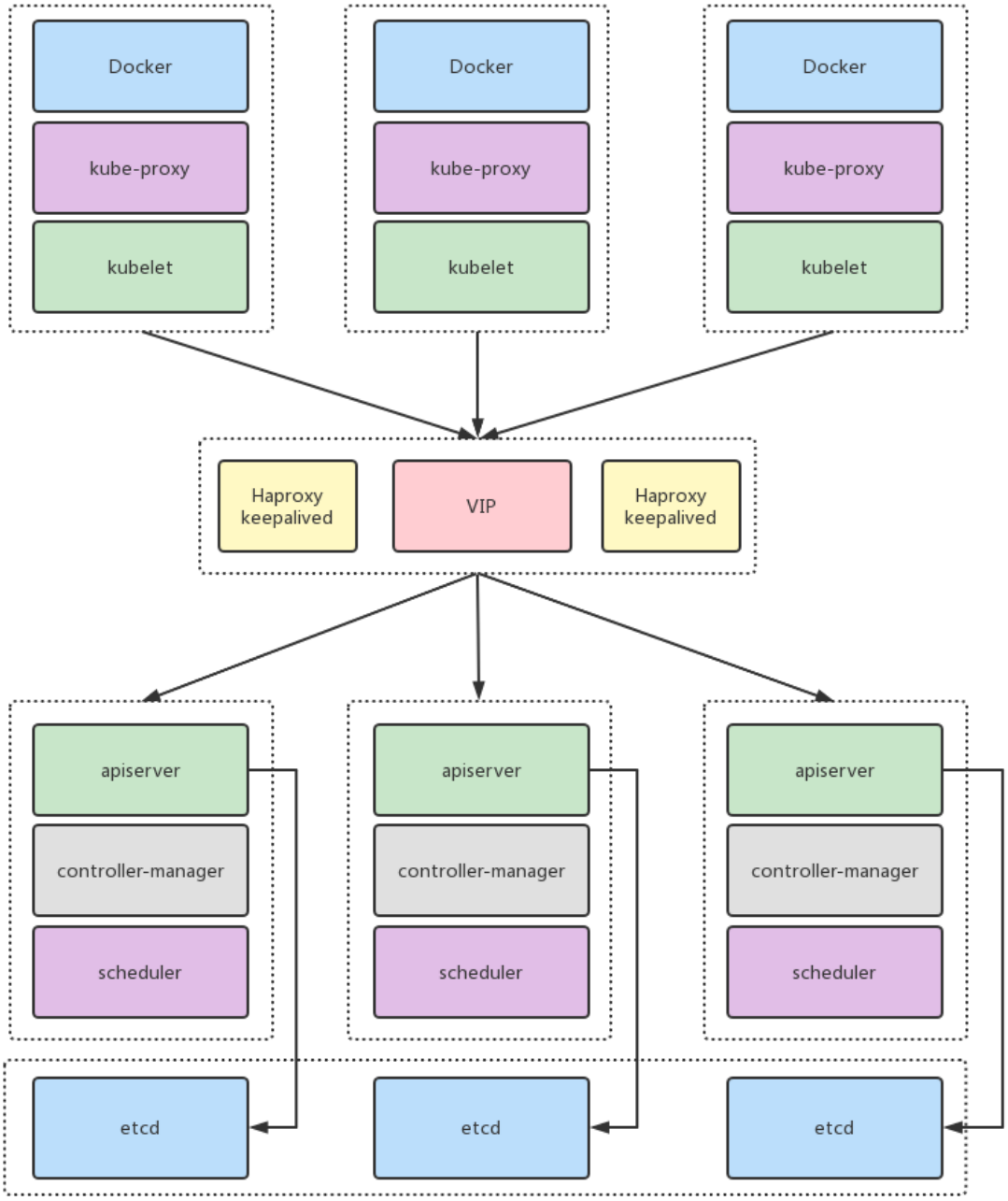
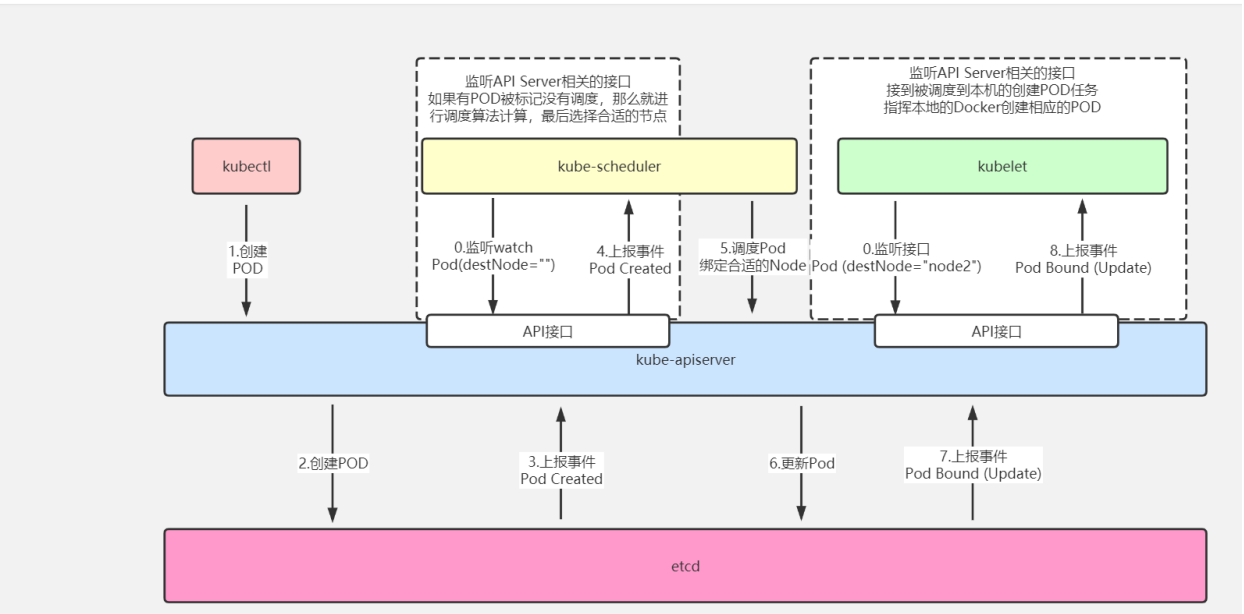

Docker
kube-proxy
kubelet
Docker
kube-proxy
kubelet
Docker
kube-proxy
kubelet
Haproxy
keepalived
VIP
Haproxy
keepalived
apiserver
controller-manager
scheduler
apiserver
controller-manager
scheduler
apiserver
controller-manager
scheduler
etcd
etcd
etcd

创建POD流程图



k8s安装集群步骤

1.配置系统环境

2.安装配置docker环境

3.安装配置kubeadm

4.master节点初始化集群

5.node节点加入集群

6.配置flannel网络插件

7.配置node节点标签

8.检查集群节点状态

第0章 参考网站

0.k8s介绍

https://kubernetes.io/zh/docs/concepts/overview/what-is-kubernetes/

1.docker官方文档

https://docs.docker.com/

2.k8s官方文档

https://kubernetes.io/zh/

3.kubeadm官方文档

https://kubernetes.io/zh/docs/setup/production-environment/tools/

4.prometheus官方文档

https://prometheus.io/docs/introduction/overview/

5.ansible安装k8s项目

https://github.com/easzlab/kubeasz

第1章 k8s系统架构

从系统架构来看，k8s分为2个节点

Master 控制节点 指挥官

Node 工作节点 干活的

1.Master节点组成

API Server：提供k8s API接口

主要处理Rest操作以及更新Etcd中的对象

是所有资源增删改查的唯一入口。

Scheduler：资源调度器

根据etcd里的节点资源状态决定将Pod绑定到哪个Node上

Controller Manager

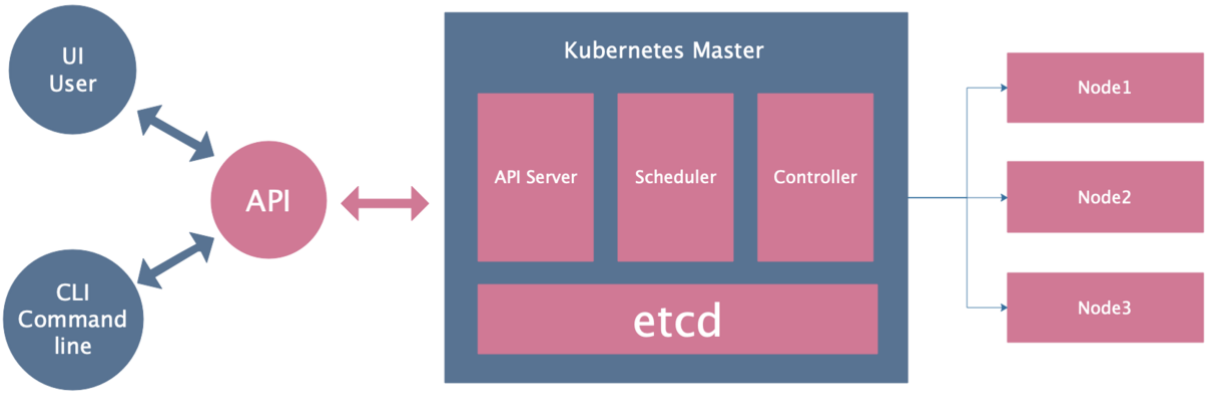
负责保障pod的健康存在

资源对象的自动化控制中心，Kubernetes集群有很多控制器。

Etcd

这个是Kubernetes集群的数据库

所有持久化的状态信息存储在Etcd中



2.Node节点的组成

Docker Engine(container runtime)容器运行时

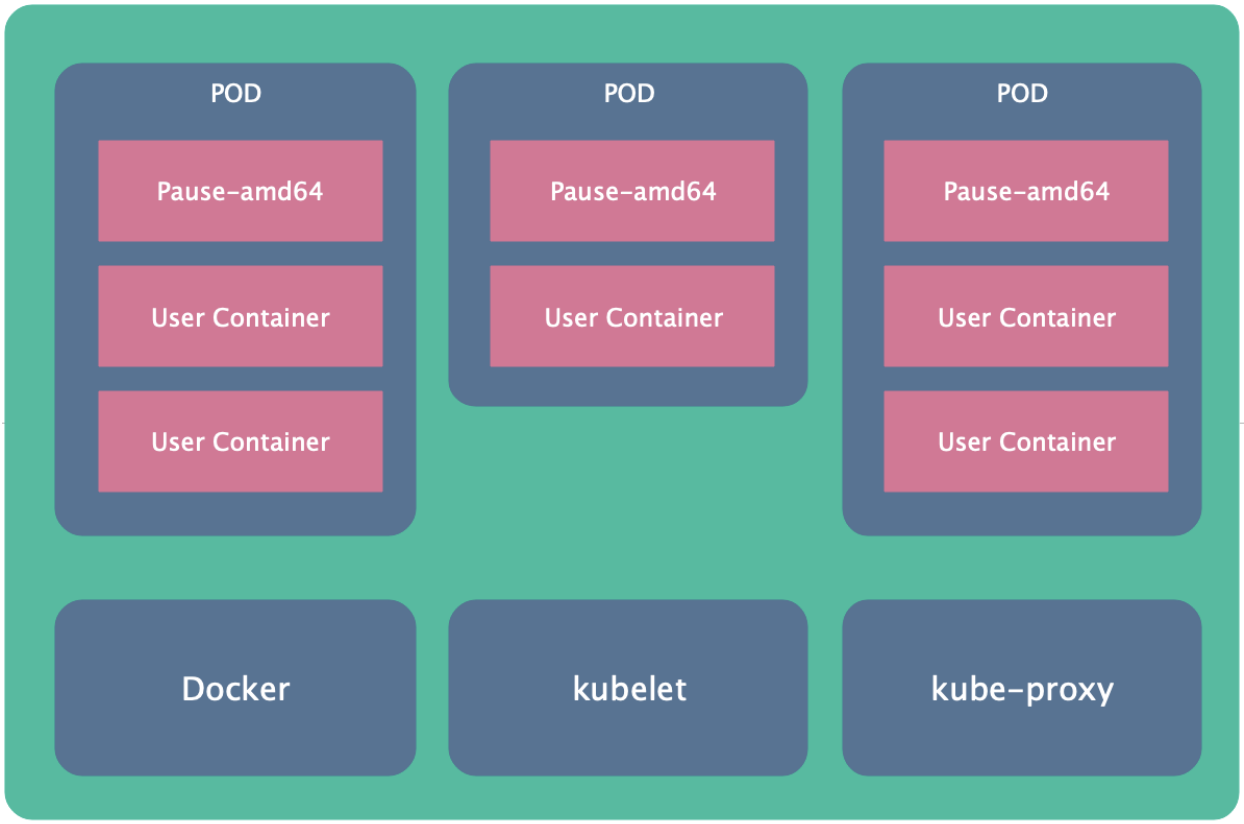
负责节点容器的管理工作，最终创建出来的是一个Docker容器。

kubelet

安装在Node上的代理服务，用来管理Pods以及容器/镜像/Volume等，实现对集群对节点的管理。

kube-proxy

安装在Node上的网络代理服务，提供网络代理以及负载均衡，实现与Service通讯。



第2章 k8s逻辑架构

从逻辑架构上看，k8s分为

Pod

Controller

Service

1.POD

POD是k8s的最小单位

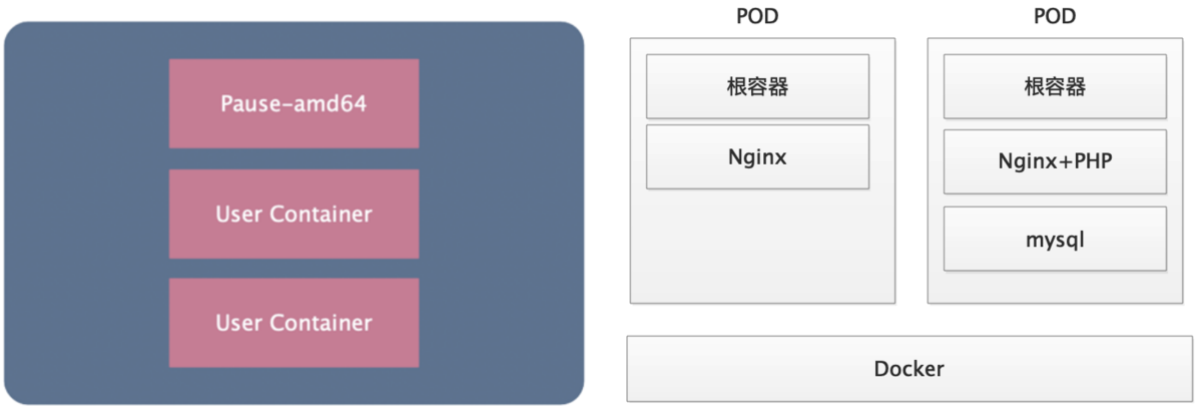
POD的IP地址是随机的，删除POD会改变IP

POD都有一个根容器pause

一个POD内可以由一个或多个容器组成

一个POD内的容器共享根容器的网络命名空间

一个POD的内的网络地址由根容器提供



2.Controller

用来管理POD,控制器的种类有很多

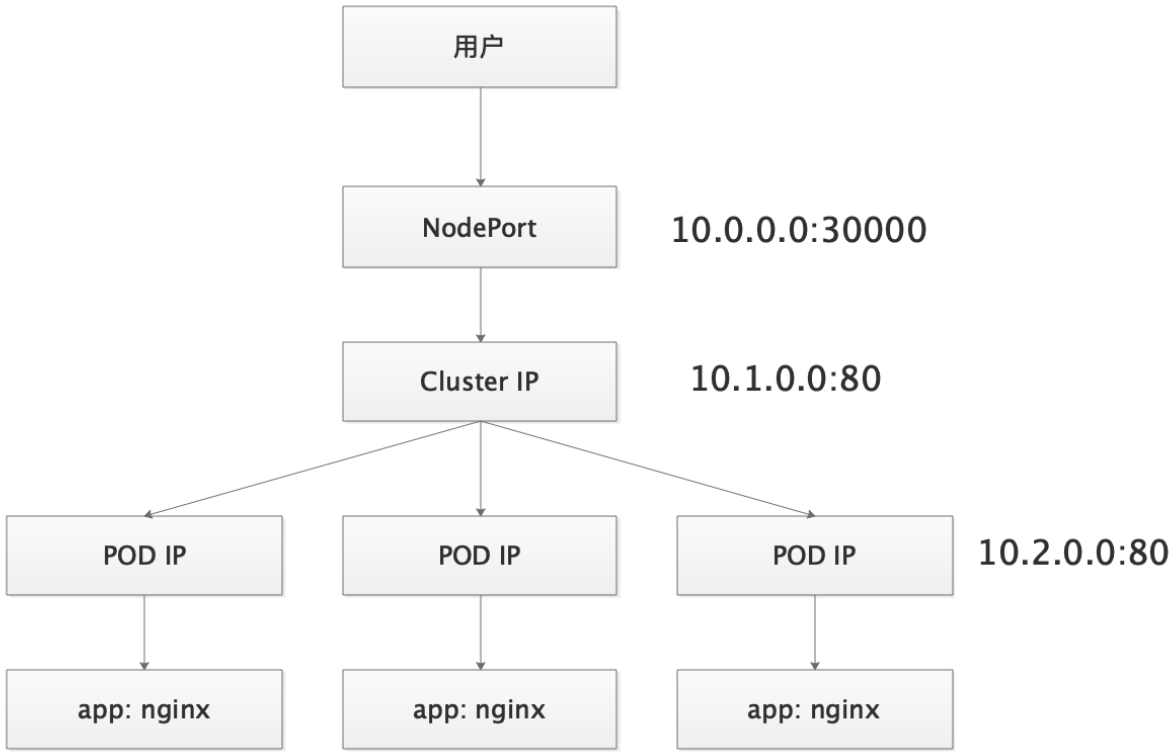
- RC Replication Controller 控制POD有多个副本
- RS ReplicaSet RC控制的升级版
- Deployment 推荐使用，功能更强大，包含了RS控制器
- DaemonSet 保证所有的Node上有且只有一个Pod在运行
- StatefulSet 有状态的应用，为Pod提供唯一的标识，它可以保证部署和scale的顺序

3.Service

NodeIP 对外提供用户访问

ClusterIP 集群内部IP，可以动态感知后面的POD IP

POD IP POD的IP



第3章 k8s实验环境准备

1.配置信息

主机名	IP地址	推荐配置
master	10.0.0.10	2C4G40G
node1	10.0.0.11	1C2G40G
node2	10.0.0.12	1C2G40G

2.初始化操作

干净环境

配置主机名

配置host解析

关闭防火墙

关闭SELinux

配置时间同步

更新好阿里源

确保网络通畅

关闭SWAP分区

```
cat >> /etc/hosts << EOF
10.0.0.10   master
10.0.0.11   node1
10.0.0.12   node2
EOF
```

```
systemctl  stop  firewalld   NetworkManager
systemctl  disable  firewalld   NetworkManager
getenforce
sed -i '/aliyuncs/d' /etc/yum.repos.d/*.repo
```

```
ping -c 1 www.baidu.com
ping -c 1 master
ping -c 1 node1
ping -c 1 node2
```

```
yum install chrony -y
systemctl start chronyd
```

```
systemctl enable chronyd
date
free -h
```

第4章 安装指定版本的docker

所有机器都操作

1.设置k8s禁止使用swap

```
cat > /etc/sysconfig/kubelet<<EOF
KUBELET_CGROUP_ARGS="--cgroup-driver=systemd"
KUBELET_EXTRA_ARGS="--fail-swap-on=false"
EOF
```

2.设置内核参数

```
cat > /etc/sysctl.d/k8s.conf <<EOF
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
EOF
sysctl --system

```

3.加载IPVS模块

```
cat >/etc/sysconfig/modules/ipvs.modules <<EOF
#!/bin/bash
modprobe -- ip_vs
modprobe -- ip_vs_rr
modprobe -- ip_vs_wrr
modprobe -- ip_vs_sh
modprobe -- nf_conntrack_ipv4
EOF
chmod +x /etc/sysconfig/modules/ipvs.modules
```

```
source /etc/sysconfig/modules/ipvs.modules
lsmod | grep -e ip_vs -e nf_conntrack_ipv
```

4.安装配置Docker

```
cd /etc/yum.repos.d/
wget https://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo
yum makecache fast
yum -y install docker-ce-19.03.15 docker-ce-cli-19.03.15
mkdir /etc/docker -p
cat > /etc/docker/daemon.json <<EOF
 {
 "exec-opts": ["native.cgroupdriver=systemd"]
 }
EOF
systemctl enable docker && systemctl start docker
docker -v
Docker version 19.03.15, build 99e3ed8919
```

第5章 部署kubeadm和kubelet

1.设置kubeadm为国内源-所有机器都操作

```
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=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
```

2.安装kubeadm-所有节点都操作

```
yum install -y kubelet-1.19.3 kubeadm-1.19.3 kubectl-1.19.3 ipvsadm
```

3.设置kubelet开机启动-所有节点都操作

```
systemctl enable kubelet
```

第6章 初始化集群部署Master

4.初始化集群命令-只在master节点运行

```
kubeadm init \
--apiserver-advertise-address=10.0.0.10 \
--image-repository registry.aliyuncs.com/google_containers \
--kubernetes-version v1.19.3 \
--service-cidr=10.1.0.0/16 \
--pod-network-cidr=10.2.0.0/16 \
--service-dns-domain=cluster.local \
--ignore-preflight-errors=Swap \
--ignore-preflight-errors=NumCPU
```

5.为kubectl准备kubeconfig-只在master节点运行

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

#支持命令补全

```
yum install bash-completion -y
source /usr/share/bash-completion/bash_completion
source <(kubectl completion bash)
```

```
kubectl completion bash >/etc/bash_completion.d/kubectl
```

6.查看节点状态-只在master节点运行

```
[root@master ~]# kubectl get nodes
NAME STATUS ROLES AGE VERSION
master NotReady master 2m37s v1.19.3
```

7.设置kube-proxy使用ipvs模式-只在master节点运行

执行命令，然后将mode: ""修改为mode: "ipvs"然后保存退出

```
kubectl edit cm kube-proxy -n kube-system
```

重启kube-proxy

```
kubectl -n kube-system get pod|grep kube-proxy|awk '{print "kube
ctl -n kube-system delete pod "$1"'|bash
```

查看pod信息

```
kubectl get -n kube-system pod|grep "kube-proxy"
```

检查IPVS规则

```
[root@master ~]# ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port Forward Weight ActiveConn InActConn
TCP 10.1.0.1:443 rr
-> 10.0.0.11:6443 Masq 1 0 0
TCP 10.1.0.10:53 rr
TCP 10.1.0.10:9153 rr
UDP 10.1.0.10:53 rr
```

第7章 部署网络插件　flannel

注意！只在master节点上安装部署！！！

注意！只在master节点上安装部署！！！

注意！只在master节点上安装部署！！！

1.部署Flannel网络插件-只在master节点上安装部署

```
yum install git -y
git clone --depth 1 https://github.com/coreos/flannel.git
```

2.修改资源配置清单

kubectl apply -f 更新配置

修改第128行和第189行

```
cd flannel/Documentation/
vim kube-flannel.yml
128: "Network": "10.2.0.0/16",
189: - --iface=eth0
```

3.应用配置资源清单

```
kubectl create -f kube-flannel.yml
```

4.检查pod运行状态，等一会应该全是running

```
[root@master ~]# kubectl -n kube-system get pod
NAME READY STATUS RESTARTS AGE
coredns-6d56c8448f-5tq6w 1/1 Running 0 44m
coredns-6d56c8448f-psltj 1/1 Running 0 44m
etcd-master 1/1 Running 0 44m
kube-apiserver-master 1/1 Running 0 44m
kube-controller-manager-master 1/1 Running 0 44m
kube-flannel-ds-8xb54 1/1 Running 0 3m1s
kube-flannel-ds-cklfd 1/1 Running 0 3m1s
 kube-flannel-ds-xvst5 1/1 Running 0 3m1s
 kube-proxy-grx25 1/1 Running 0 39m
 kube-proxy-v4snj 1/1 Running 0 7m33s
 kube-proxy-zvvfk 1/1 Running 0 7m36s
 kube-scheduler-master 1/1 Running 0 44m
```

第8章 部署Node节点

1.master节点输出增加节点的命令

```
kubeadm token create --print-join-command
```

2.node节点执行加入集群命令　node1和node2上

```
kubeadm join 10.0.0.11:6443 --token uqf018.mia8v3i1zcai19sj --di
scovery-token-ca-cert-hash sha256:e7d36e1fb53e59b12f0193f4733edb4
5d924321bcfc055f801cf1ea59d90aae
```

3.在master节点上查看状态

```
[root@master ~]#kubectl get nodes
```

4.在master节点上给其他节点打标签

```
kubectl label nodes node1 node-role.kubernetes.io/node=
kubectl label nodes node2 node-role.kubernetes.io/node=
```

去掉标签　　是把node=　　改为node-

5.再次查看节点状态

```
[root@node1 ~]# kubectl get nodes
NAME STATUS ROLES AGE VERSION
node1 Ready master 8m28s v1.19.3
node2 Ready node 7m56s v1.19.3
node3 Ready node 7m59s v1.19.3
```

创建pod简单体验

常用指令

```
kubectl get nodes #查看节点状态
kubectl get pods #查看默认命名空间的pod运行情况
kubectl get pod -o wide #输出pod的详细信息，包含IP和节点信息
kubectl get nodes -o wide #查看节点状态详细信息，包含ip，系统内核和版本
kubectl describe pod nginx-xxx #查看默认命名空间的pod运行详细情况
kubectl delete pod nginx #删除POD
kubectl explain pods #查看pod资源类型的使用说明
```

创建一个pod：

命令形式创建

```
kubectl create deployment nginx --image=nginx:alpine
kubectl get pods
kubectl describe pod nginx-xxx
```

资源配置清单形式创建：

```
apiVersion: v1 #api版本号
kind: Pod #资源类型为POD
metadata: #原数据
 name: nginx #pod显示名称
 labels: #标签
 app: nginx #具体的标签
spec: #定义pod参数
 containers: #定义容器
 - image: nginx:alpine #镜像名称
  imagePullPolicy: IfNotPresent #拉取镜像策略，如果本地已经存在镜像，就不重新拉取
  name: nginx #容器名称
```