kubernetes K8S，就是基于容器的集群管理平台

一. K8S 起源

在Docker容器技术被炒得热火朝天之时，大家发现，如果想要将Docker应用于具体的业务实现，是存在困难的——编排、管理和调度等各个方面，都不容易。于是，人们迫切需要一套管理系统，对Docker及容器进行更高级更灵活的管理。就在这个时候，K8S出现了。

K8S，就是基于容器的集群管理平台，它的全称，是kubernetes。

Kubernetes这个单词来自于希腊语，含义是舵手或领航员。

K8S是它的缩写，用“8”字替代了“ubernete”这8个字符。

和Docker不同，K8S的创造者，是众人皆知的行业巨头——**Google**。

然而，K8S并不是一件全新的发明。它的前身，是Google自己捣鼓了十多年的**Borg系统**。

K8S是2014年6月由Google公司正式公布出来并宣布开源的。

同年7月，微软、Red Hat、IBM、Docker、CoreOS、Mesosphere和Saltstack等公司，相继加入K8S。

之后的一年内，VMware、HP、Intel等公司，也陆续加入。

2015年7月，Google正式加入OpenStack基金会。与此同时，Kuberentes v1.0正式发布。

目前，kubernetes的版本已经发展到V1.13。

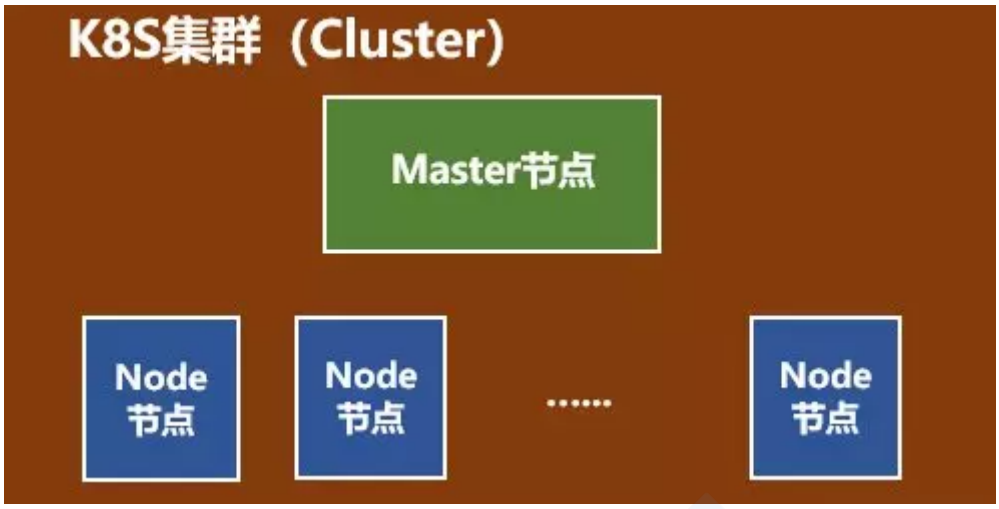
二. K8S 集群

K8S的架构，略微有一点复杂，我们简单来看一下。

一个K8S系统，通常称为一个**K8S集群（Cluster）**。

这个集群主要包括两个部分：

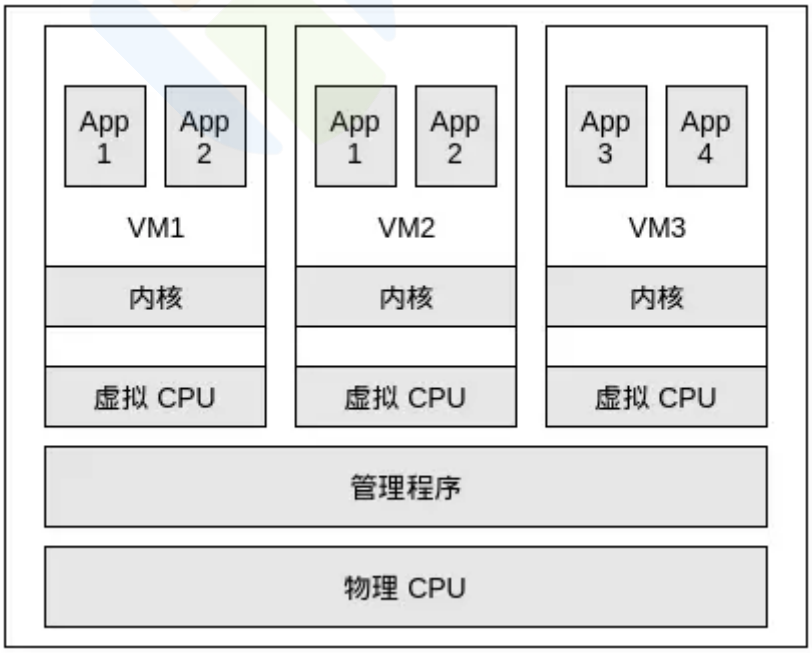
- **一个Master节点（主节点）**
- **一群Node节点（计算节点）**



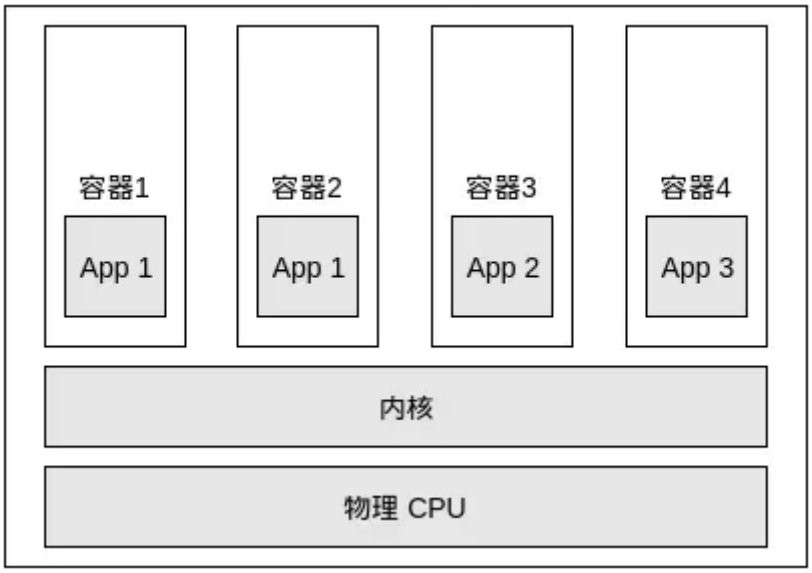
一看就明白：Master节点主要还是负责管理和控制。Node节点是工作负载节点，里面是具体的容器。

三. 为什么需要 k8s?

1. 应用部署模式的演进



虚拟化模式



容器化模式

相比虚拟机和容器

- 容器更加轻量级，启动更快（秒级）
- 容器可移植性更好

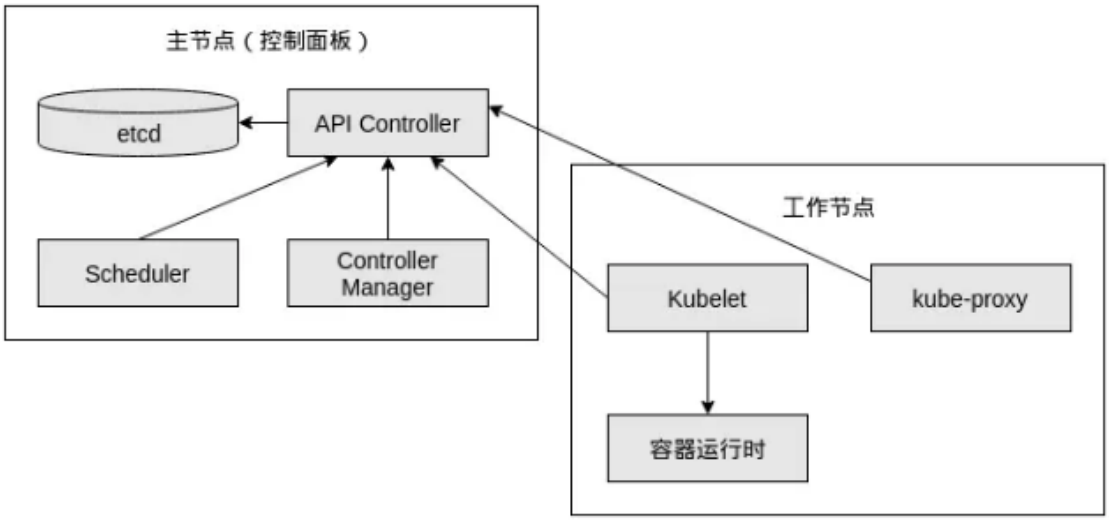
2. 管理大量的容器带来了新的挑战

容器编排调度引擎 —— k8s 的好处

- 简化应用部署
- 提高硬件资源利用率
- 健康检查和自修复
- 自动扩容缩容
- 服务发现和负载均衡

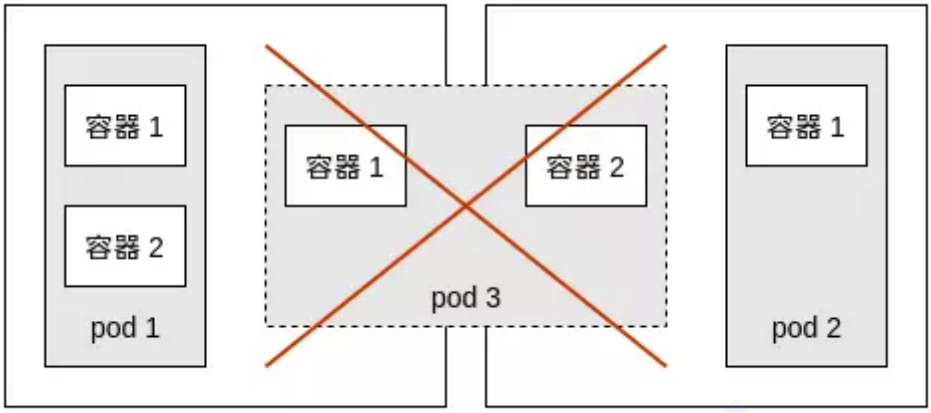
四. k8s 的集群架构

主节点，承载 k8s 的控制和管理整个集群系统的控制面板
工作节点，运行用户实际的应用



五. pod —— k8s 调度的最小单元

1. 一个 pod 包含一组容器，一个 pod 不会跨越多个工作节点



pod 不会跨越工作节点

2. 了解 pod

- pod 相当与逻辑主机，每个 pod 都有自己的 IP 地址
- pod 内的容器共享相同的 IP 和端口空间
- 默认情况下，每个容器的文件系统与其他容器完全隔离

六. 快速部署kubernetes集群

kubeadm是官方社区推出的一个用于快速部署kubernetes集群的工具。

这个工具能通过两条指令完成一个kubernetes集群的部署：

```
# 创建一个 Master 节点
$ kubeadm init
# 将一个 Node 节点加入到当前集群中
$ kubeadm join <Master节点的IP和端口 >
```

1. 安装要求

在开始之前，部署Kubernetes集群机器需要满足以下几个条件：

```
· 一台或多台机器，操作系统 CentOS7.x-86_x64
· 硬件配置：2GB或更多RAM，2个CPU或更多CPU，硬盘30GB或更多
· 集群中所有机器之间网络互通
· 可以访问外网，需要拉取镜像
· 禁止swap分区
```

2. 学习目标

```
1.在所有节点上安装Docker和kubeadm
2.部署Kubernetes Master
3.部署容器网络插件
4.部署 Kubernetes Node，将节点加入Kubernetes集群中
5.部署Dashboard Web页面，可视化查看Kubernetes资源
```

3. 准备环境

```
关闭防火墙：
$ systemctl stop firewalld
$ systemctl disable firewalld

关闭selinux：
$ sed -i 's/enforcing/disabled/' /etc/selinux/config
$ setenforce 0

关闭swap：
$ swapoff -a $ 临时
$ vim /etc/fstab $ 永久

添加主机名与IP对应关系（记得设置主机名）：
$ cat /etc/hosts
192.168.23.35 k8s-master
192.168.23.36 k8s-node1
192.168.23.37 k8s-node2

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
$ sysctl --system
```

4. 所有节点安装Docker/kubeadm/kubelet

Kubernetes默认CRI（容器运行时）为Docker，因此先安装Docker。

5. 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-18.06.1.ce-3.el7
$ systemctl enable docker && systemctl start docker
$ docker --version
Docker version 18.06.1-ce, build e68fc7a
```

6. 添加阿里云YUM软件源

```
$ 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
```

7. 安装kubeadm，kubelet和kubectl

由于版本更新频繁，这里指定版本号部署：

```
$ yum install -y kubelet-1.15.0 kubeadm-1.15.0 kubectl-1.15.0
$ systemctl enable kubelet
```

8. 部署Kubernetes Master

在192.168.31.61（Master）执行

```
$ kubeadm init \
--apiserver-advertise-address=192.168.31.61 \
--image-repository registry.aliyuncs.com/google_containers \
--kubernetes-version v1.15.0 \
--service-cidr=10.1.0.0/16 \
--pod-network-cidr=10.244.0.0/16
```

由于默认拉取镜像地址k8s.gcr.io国内无法访问，这里指定阿里云镜像仓库地址。

使用kubectl工具：

```
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
```

9. 安装Pod网络插件（CNI）

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

确保能够访问到quay.io这个registery。

如果下载失败，可以改成这个镜像地址：lizhenliang/flannel:v0.11.0-amd64

10 .加入Kubernetes Node

在192.168.31.62/63（Node）执行。

向集群添加新节点，执行在kubeadm init输出的kubeadm join命令：

```
$ kubeadm join 192.168.23.35:6443 --token esce21.q6hetwm8si29qxwn \
  --discovery-token-ca-cert-hash
sha256:00603a05805807501d7181c3d60b478788408cfe6cedefedb1f97569708be9c5
```

11. 测试kubernetes集群

在Kubernetes集群中创建一个pod，验证是否正常运行：

```
$ kubectl create deployment nginx --image=nginx
$ kubectl expose deployment nginx --port=80 --type=NodePort
$ kubectl get pod,svc
```

访问地址：http://NodeIP:Port

12. 部署 Dashboard

```
$ kubectl apply -f
https://raw.githubusercontent.com/kubernetes/dashboard/v1.10.1/src/deploy/recommended/kubernetes-dashboard.yaml
```

默认镜像国内无法访问，修改镜像地址为： lizhenliang/kubernetes-dashboard-amd64:v1.10.1

默认Dashboard只能集群内部访问，修改Service为NodePort类型，暴露到外部：

```
kind: Service
apiVersion: v1
metadata:
labels:
  k8s-app: kubernetes-dashboard
name: kubernetes-dashboard
namespace: kube-system
spec:
type: NodePort
ports:
  - port: 443
    targetPort: 8443
```

```
    nodePort: 30001
selector:
  k8s-app: kubernetes-dashboard
$ kubectl apply -f kubernetes-dashboard.yaml
```

访问地址：http://NodeIP:30001

创建service account并绑定默认cluster-admin管理员集群角色：

```
$ kubectl create serviceaccount dashboard-admin -n kube-system
$ kubectl create clusterrolebinding dashboard-admin --clusterrole=cluster-admin
--serviceaccount=kube-system:dashboard-admin
$ kubectl describe secrets -n kube-system $(kubectl -n kube-system get secret |
awk '/dashboard-admin/{print $1}')
```

使用输出的token登录Dashboard。