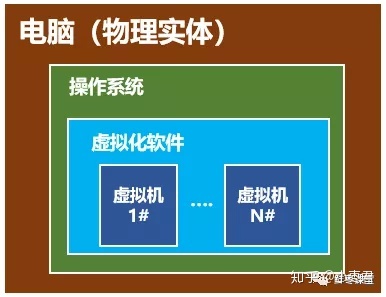
# k8s

## Docker

Docker和容器技术为什么会这么火爆？说白了，就是因为它“轻”。

在容器技术之前，业界的网红是**虚拟机**。虚拟机技术的代表，是**VMWare**和**OpenStack**。

相信很多人都用过虚拟机。虚拟机，就是在你的操作系统里面，装一个软件，然后通过这个软件，再模拟一台甚至多台“子电脑”出来。



虚拟机，类似于“子电脑”

在“子电脑”里，你可以和正常电脑一样运行程序，例如开QQ。如果你愿意，你可以变出好几个“子电脑”，里面都开上QQ。“子电脑”和“子电脑”之间，是**相互隔离**的，互不影响。

虚拟机属于虚拟化技术。而Docker这样的容器技术，也是虚拟化技术，属于**轻量级的虚拟化**。

虚拟机虽然可以隔离出很多“子电脑”，但占用空间更大，启动更慢，虚拟机软件可能还要花钱（例如VMWare）。

而容器技术恰好没有这些缺点。它不需要虚拟出整个操作系统，只需要虚拟一个小规模的环境（类似“沙箱”）。

它启动时间很快，几秒钟就能完成。而且，它对资源的利用率很高（一台主机可以同时运行几千个Docker容器）。此外，它占的空间很小，虚拟机一般要几GB到几十GB的空间，而容器只需要MB级甚至KB级。

容器和虚拟机的对比

正因为如此，容器技术受到了热烈的欢迎和追捧，发展迅速。

大家需要注意，**Docker本身并不是容器**，它是创建容器的工具，是应用容器引擎。

想要搞懂Docker，其实看它的两句口号就行。

第一句，是**“Build, Ship and Run”**。

举个例子：

我来到一片空地，想建个房子，于是我搬石头、砍木头、画图纸，一顿操作，终于把这个房子盖好了。

结果，我住了一段时间，想搬到另一片空地去。这时候，按以往的办法，我只能再次搬石头、砍木头、画图纸、盖房子。

但是，跑来一个老巫婆，教会我一种魔法。

这种魔法，可以把我盖好的房子复制一份，做成“镜像”，放在我的背包里。

等我到了另一片空地，就用这个“镜像”，复制一套房子，摆在那边，拎包入住。

Docker的第二句口号就是：“Build once，Run anywhere（搭建一次，到处能用）”。

Docker技术的三大核心概念，分别是：

* **镜像（Image）**
* **容器（Container）**
* **仓库（Repository）**

我刚才例子里面，那个放在包里的“镜像”，就是**Docker镜像**。而我的背包，就是**Docker仓库**。我在空地上，用魔法造好的房子，就是一个**Docker容器**。

说白了，这个Docker镜像，是一个特殊的文件系统。它除了提供容器运行时所需的程序、库、资源、配置等文件外，还包含了一些为运行时准备的一些配置参数（例如环境变量）。镜像不包含任何动态数据，其内容在构建之后也不会被改变。

也就是说，每次变出房子，房子是一样的，但生活用品之类的，都是不管的。谁住谁负责添置。

每一个镜像可以变出一种房子。那么，我可以有多个镜像呀！

也就是说，我盖了一个欧式别墅，生成了镜像。另一个哥们可能盖了一个中国四合院，也生成了镜像。还有哥们，盖了一个非洲茅草屋，也生成了镜像。。。

于是乎，就变成了一个大的公共仓库。

负责对Docker镜像进行管理的，是**Docker Registry服务**（类似仓库管理员）。

不是任何人建的任何镜像都是合法的。万一有人盖了一个有问题的房子呢？

所以，Docker Registry服务对镜像的管理是非常严格的。

最常使用的Registry公开服务，是官方的**Docker Hub**，这也是默认的 Registry，并拥有大量的高质量的官方镜像。

## Kubernetes

就在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。

**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**。

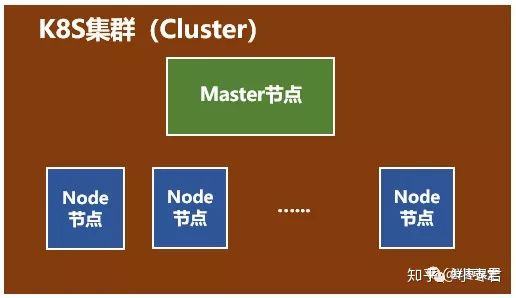
K8S是2014年6月由Google公司正式公布出来并宣布开源的。

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：

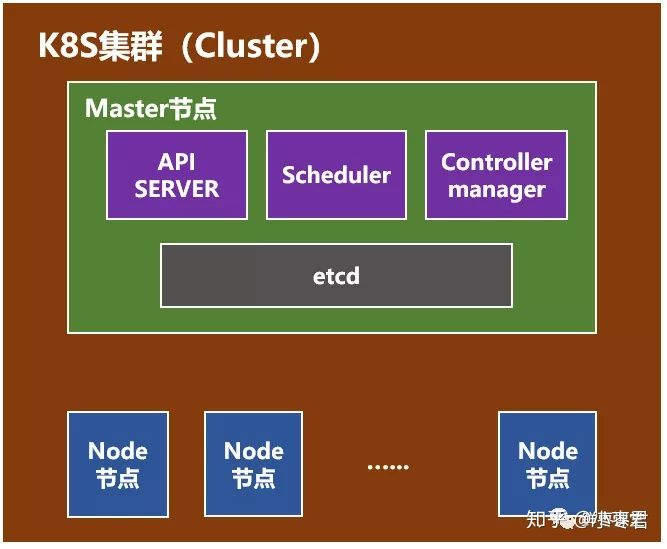
* **一个Master节点（主节点）**
* **一群Node节点（计算节点）**



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。

深入来看这两种节点。

首先是**Master节点**。



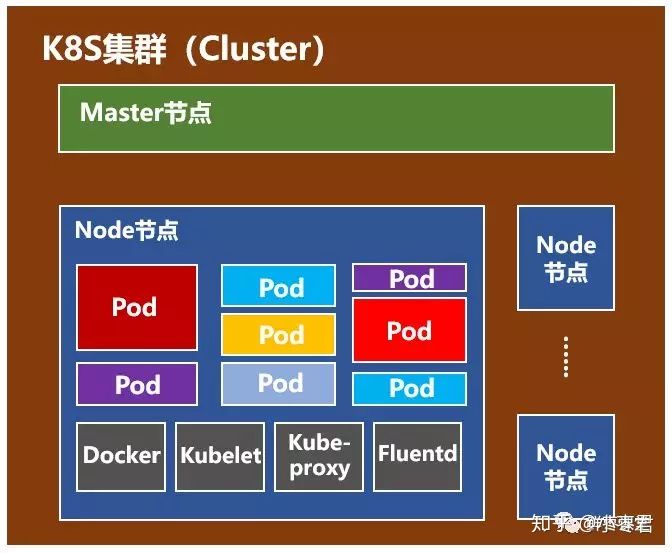
Master节点包括API Server、Scheduler、Controller manager、etcd。

API Server是整个系统的对外接口，供客户端和其它组件调用，相当于“营业厅”。

Scheduler负责对集群内部的资源进行调度，相当于“调度室”。

Controller manager负责管理控制器，相当于“大总管”。

然后是**Node节点**。



Node节点包括Docker、kubelet、kube-proxy、Fluentd、kube-dns（可选），还有就是**Pod**。

Pod是Kubernetes最基本的操作单元。一个Pod代表着集群中运行的一个进程，它内部封装了一个或多个紧密相关的容器。除了Pod之外，K8S还有一个**Service**的概念，一个Service可以看作一组提供相同服务的Pod的对外访问接口。这段不太好理解，跳过吧。

Docker，不用说了，创建容器的。

Kubelet，主要负责监视指派到它所在Node上的Pod，包括创建、修改、监控、删除等。

Kube-proxy，主要负责为Pod对象提供代理。

Fluentd，主要负责日志收集、存储与查询。

## jupyterhub的分布式搭建-基于kubernetes的jupyterhub

<https://blog.csdn.net/qq_42878856/article/details/103185361>

## JupyterHub on Kubernetes部署与应用指南

<https://my.oschina.net/u/2306127/blog/1837196?from=timeline&isappinstalled=0>

## JupyterHub on Kubernetes部署

<https://www.jianshu.com/p/491a562127fa>

安装好了k8s，helm后，终于来到了最后一步，安装jupyterhub。然而可惜的是，官网提供的方法仍然不可用，需要参考[https://my.oschina.net/u/2306127/blog/1836933](https://links.jianshu.com/go?to=https%3A%2F%2Fmy.oschina.net%2Fu%2F2306127%2Fblog%2F1836933)的方法。当然这里面还有坑。。。

基本按照参考文章里的快速安装指南来安装，但是按照他的安装步骤是不成功的，所以这里参考我的来：

* 从github上clone该项目，[https://github.com/openthings/kubernetes-tools/tree/master/jupyter](https://links.jianshu.com/go?to=https%3A%2F%2Fgithub.com%2Fopenthings%2Fkubernetes-tools%2Ftree%2Fmaster%2Fjupyter)
* 执行jupyterhub-images-pull.sh脚本（原来getimages.sh脚本下载的镜像不能用，这里直接用这个脚本）来下载镜像（这里有个镜像gcr.io/google\_containers/pause下载不成功）
* 参考[https://blog.csdn.net/wangtaoking1/article/details/49122977](https://links.jianshu.com/go?to=https%3A%2F%2Fblog.csdn.net%2Fwangtaoking1%2Farticle%2Fdetails%2F49122977)描述的方法下载[gcr.io/google\_containers/pause](https://links.jianshu.com/go?to=http%3A%2F%2Fgcr.io%2Fgoogle_containers%2Fpause)镜像

<https://github.com/openthings/kubernetes-tools/tree/master/jupyter>

## JupyterHub+Docker+K8s 简单总结

安装流程主要包括：

* 安装docker、k8s
* 安装helm
* 安装jupyterhub
* 配置pvc持久化
* 开发并配置auth身份认证
* 更新singleuser镜像

<https://www.jianshu.com/p/90db3803e46d>