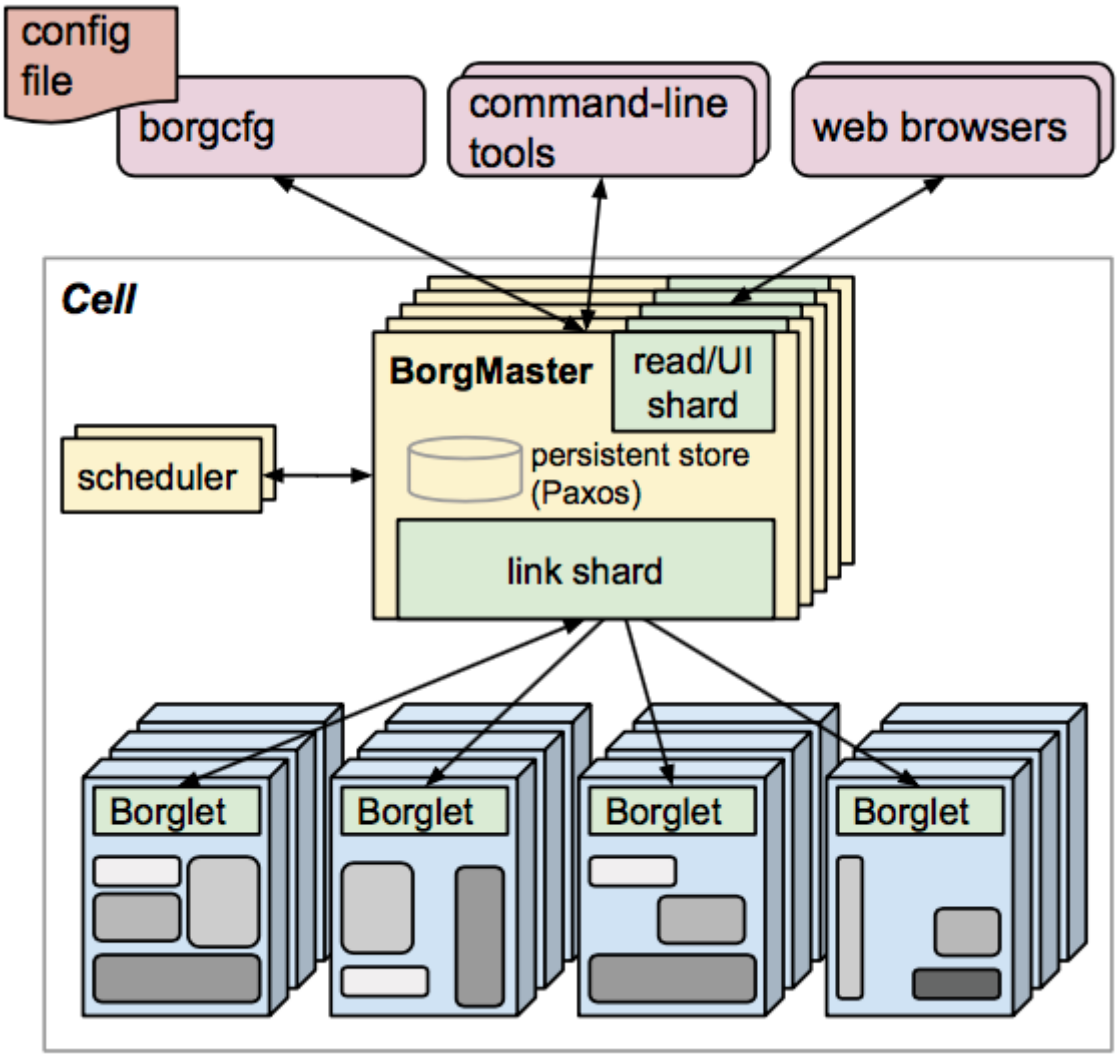
Kubernetes架构

Kubernetes最初源于谷歌内部的Borg，提供了面向应用的容器集群部署和管理系统。Kubernetes的目标旨在消除编排物理/虚拟计算，网络和存储基础设施的负担，并使应用程序运营商和开发人员完全将重点放在以容器为中心的原语上进行自助运营。Kubernetes 也提供稳定、兼容的基础（平台），用于构建定制化的workflows 和更高级的自动化任务。 Kubernetes 具备完善的集群管理能力，包括多层次的安全防护和准入机制、多租户应用支撑能力、透明的服务注册和服务发现机制、内建负载均衡器、故障发现和自我修复能力、服务滚动升级和在线扩容、可扩展的资源自动调度机制、多粒度的资源配额管理能力。
Kubernetes 还提供完善的管理工具，涵盖开发、部署测试、运维监控等各个环节。

Borg简介

Borg是谷歌内部的大规模集群管理系统，负责对谷歌内部很多核心服务的调度和管理。Borg的目的是让用户能够不必操心资源管理的问题，让他们专注于自己的核心业务，并且做到跨多个数据中心的资源利用率最大化。
Borg主要由BorgMaster、Borglet、borgcfg和Scheduler组成，如下图所示

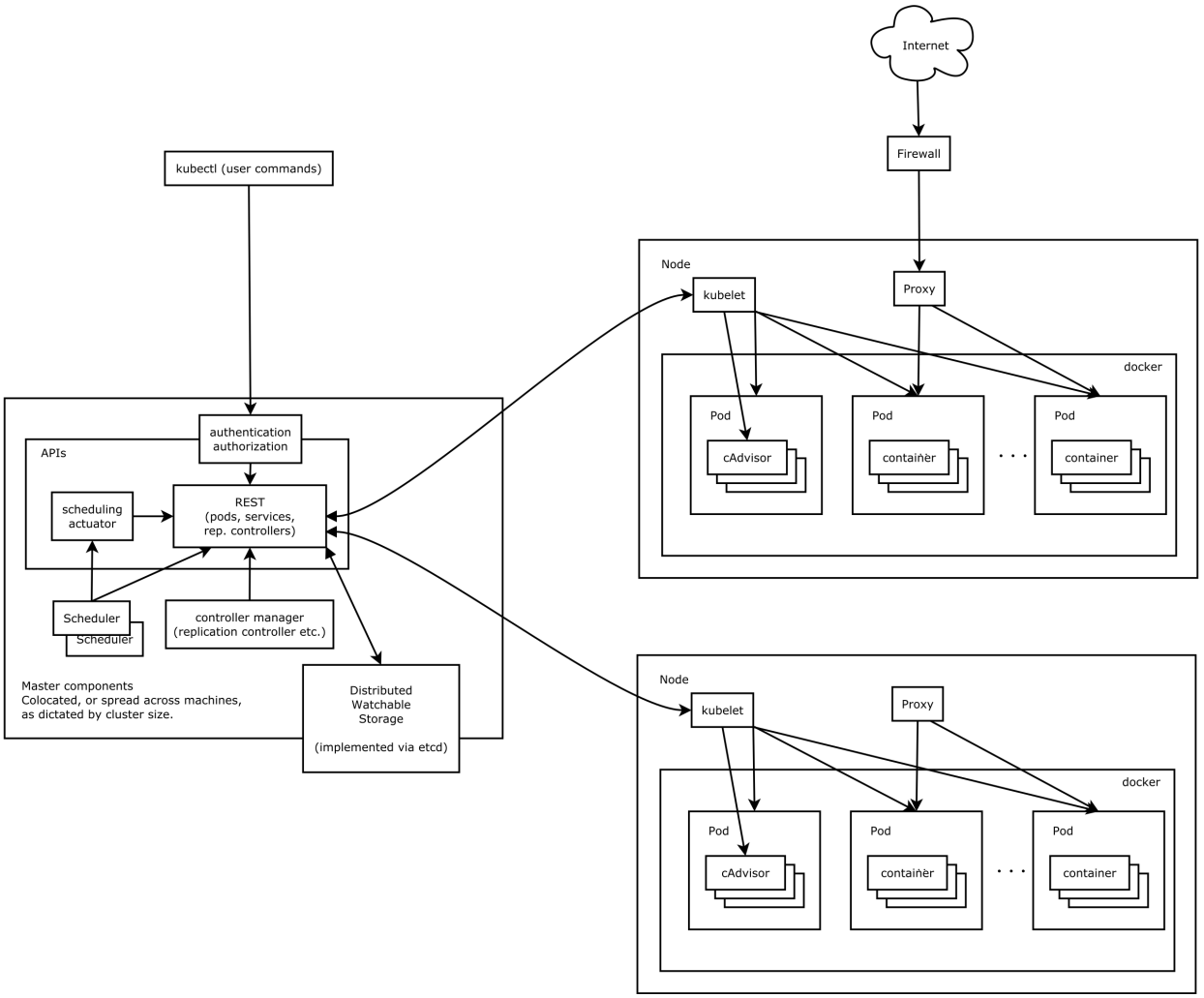


图片 - Borg架构

- BorgMaster是整个集群的大脑，负责维护整个集群的状态，并将数据持久化到Paxos存储中；
- Scheduer负责任务的调度，根据应用的特点将其调度到具体的机器上去；
- Borglet负责真正运行任务（在容器中）；
- borgcfg是Borg的命令行工具，用于跟Borg系统交互，一般通过一个配置文件来提交任务。

Kubernetes架构

Kubernetes借鉴了Borg的设计理念，比如Pod、Service、Labels和单Pod单IP等。Kubernetes的整体架构跟Borg非常像，如下图所示



图片 - Kubernetes架构

Kubernetes主要由以下几个核心组件组成：

- etcd保存了整个集群的状态；
- apiserver提供了资源操作的唯一入口，并提供认证、授权、访问控制、API注册和发现等机制；
- controller manager负责维护集群的状态，比如故障检测、自动扩展、滚动更新等；
- scheduler负责资源的调度，按照预定的调度策略将Pod调度到相应的机器上；

- **kubelet负责维护容器的生命周期，同时也负责Volume（CSI）和网络（CNI）的管理；**
- **Container runtime负责镜像管理以及Pod和容器的真正运行（CRI）；**
- **kube-proxy负责为Service提供cluster内部的服务发现和负载均衡；**

除了核心组件，还有一些推荐的Add-ons：

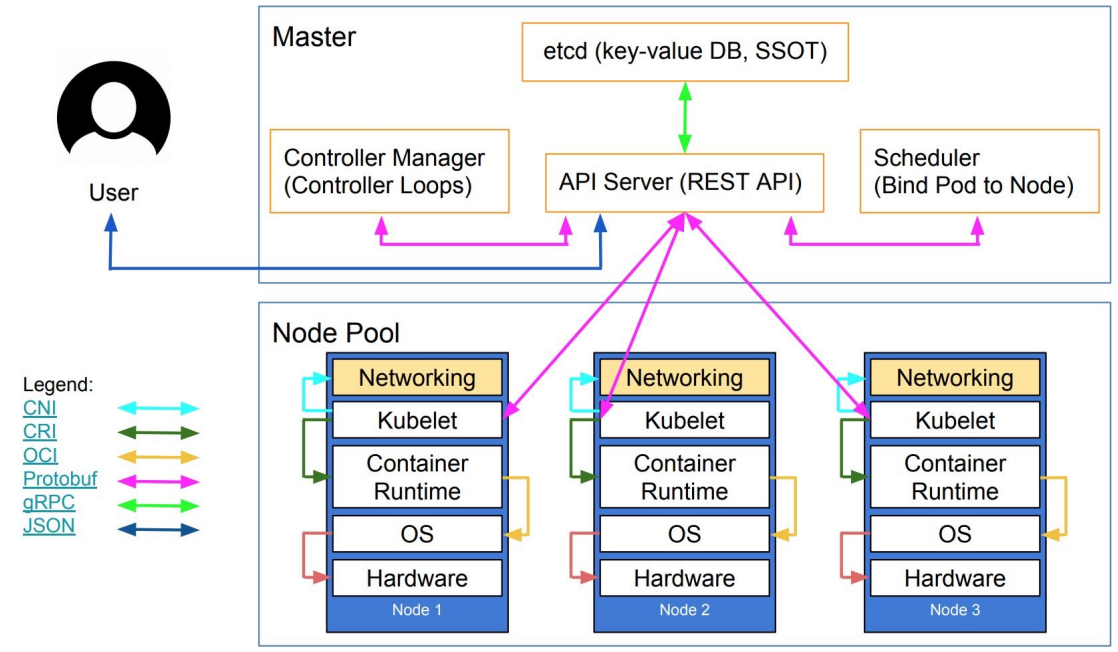
- **kube-dns负责为整个集群提供DNS服务**
- **Ingress Controller为服务提供外网入口**
- **Heapster提供资源监控**
- **Dashboard提供GUI**
- **Federation提供跨可用区的集群**

Kubernetes架构示意图

整体架构

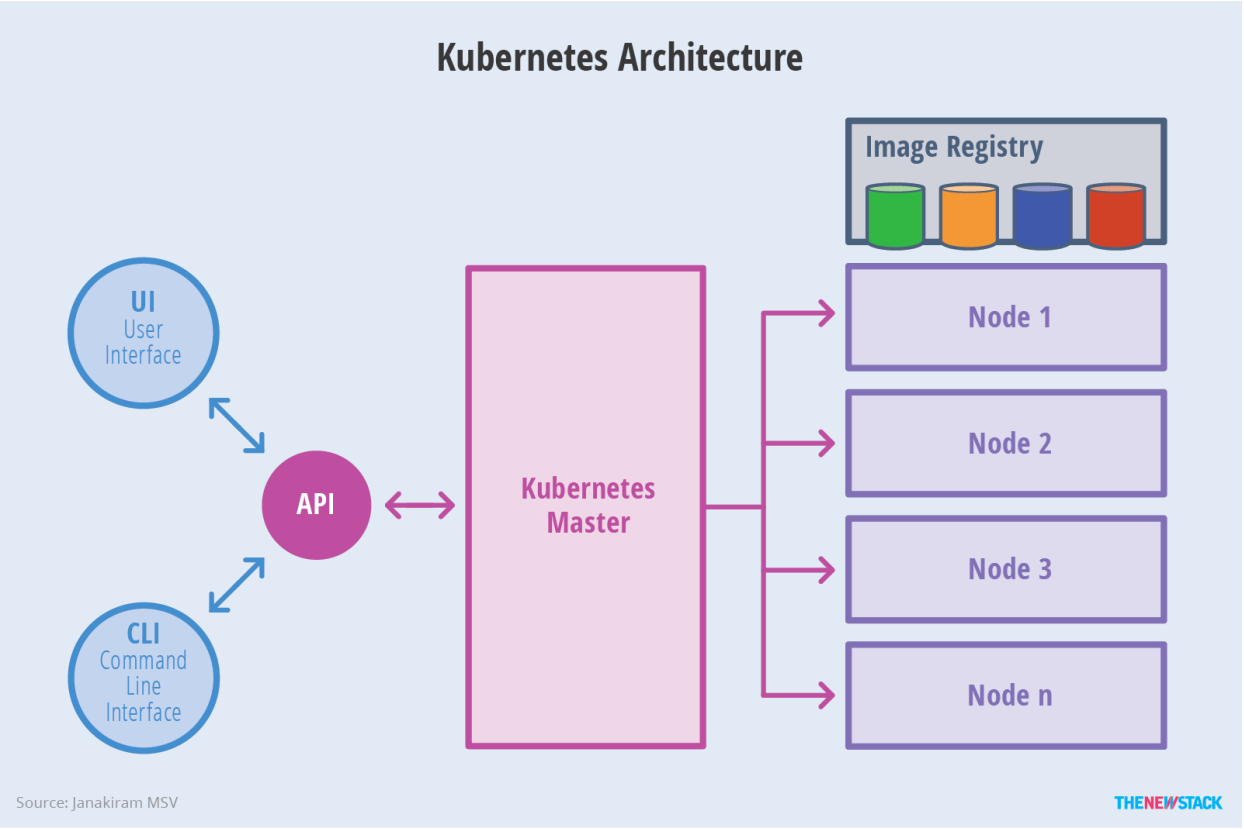
下图清晰表明了Kubernetes的架构设计以及组件之间的通信协议。

Etcd：etcd是CoreOS开源，分布式健值数据库，做为kubernetes集群所有组件的（single source of truth ） SSOT。主节点需要etcd获得节点状态，pods状态和容器状态参数信息。



图片 - Kuberentes架构

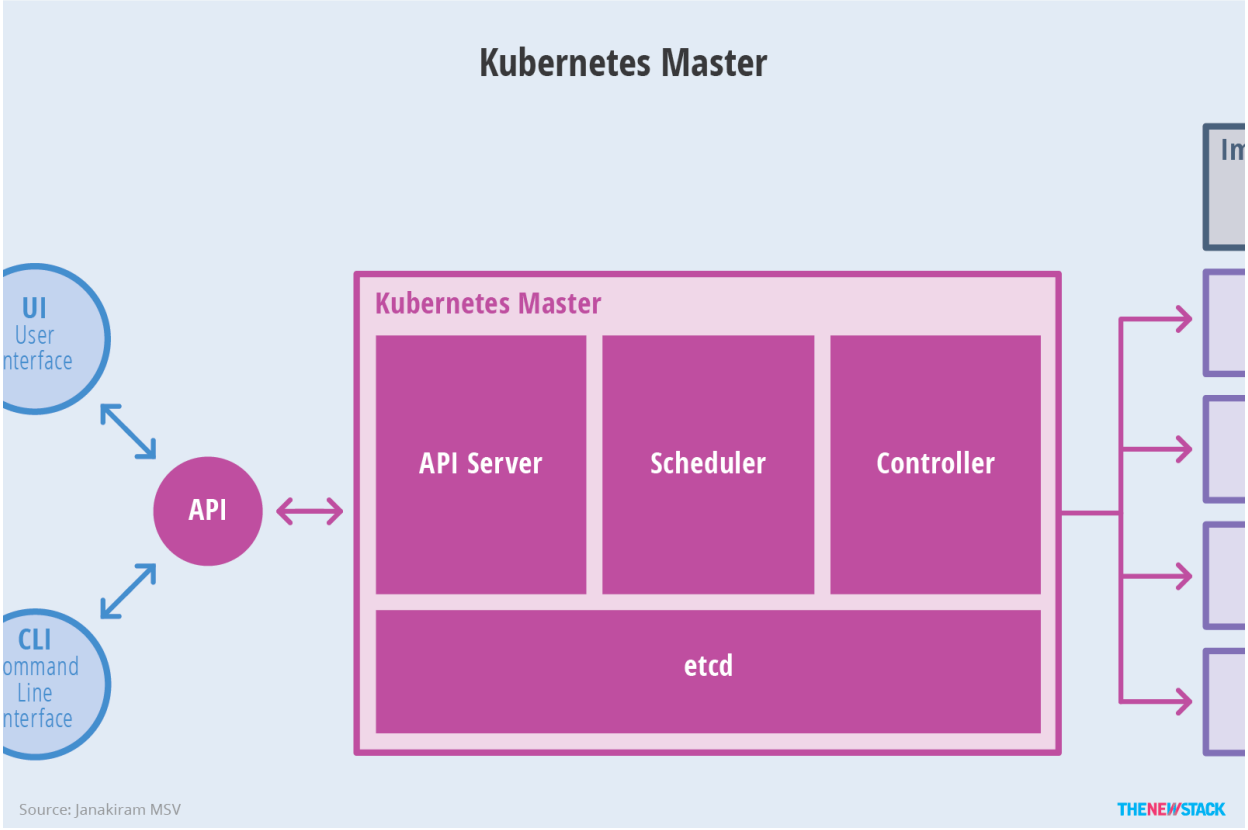
下面是更抽象的一个视图：



图片 - kubernetes整

体架构示意图

Master架构

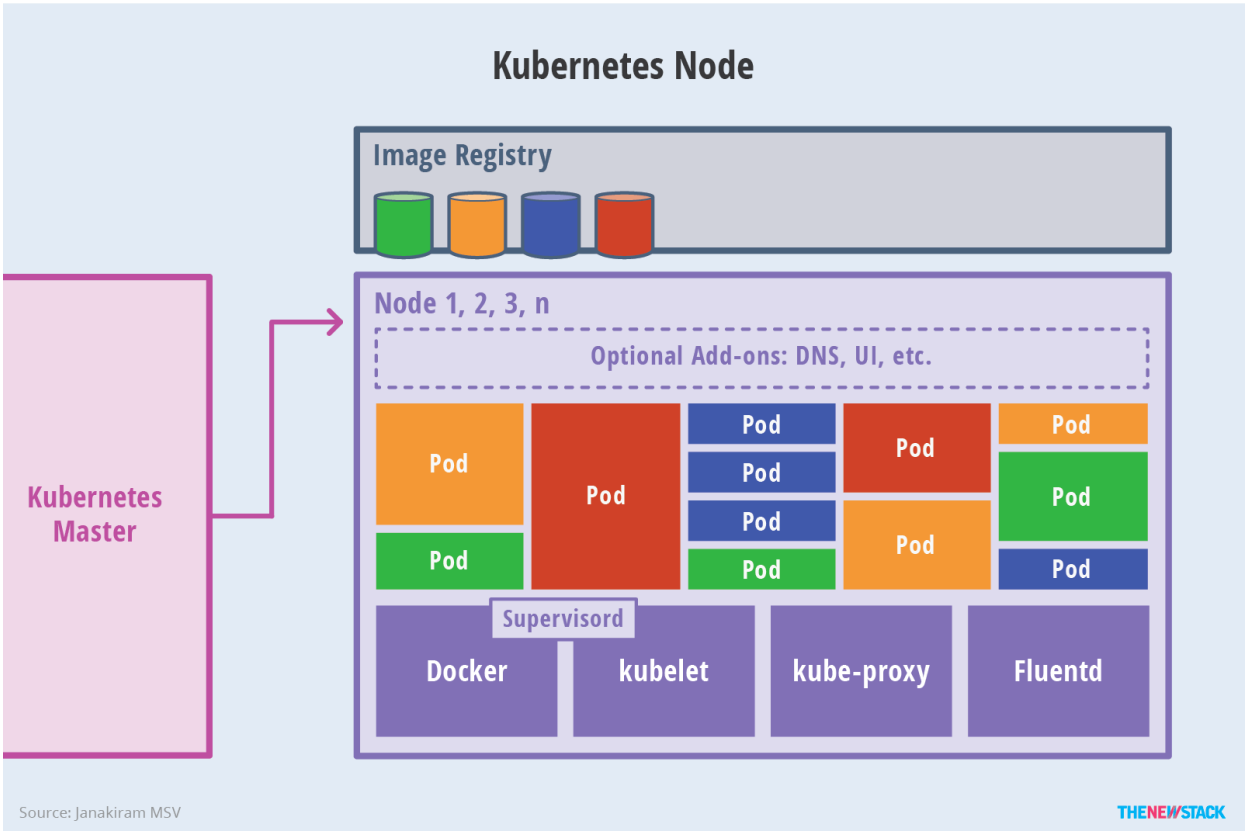


1.

图片 -

Kubernetes master架构示意图

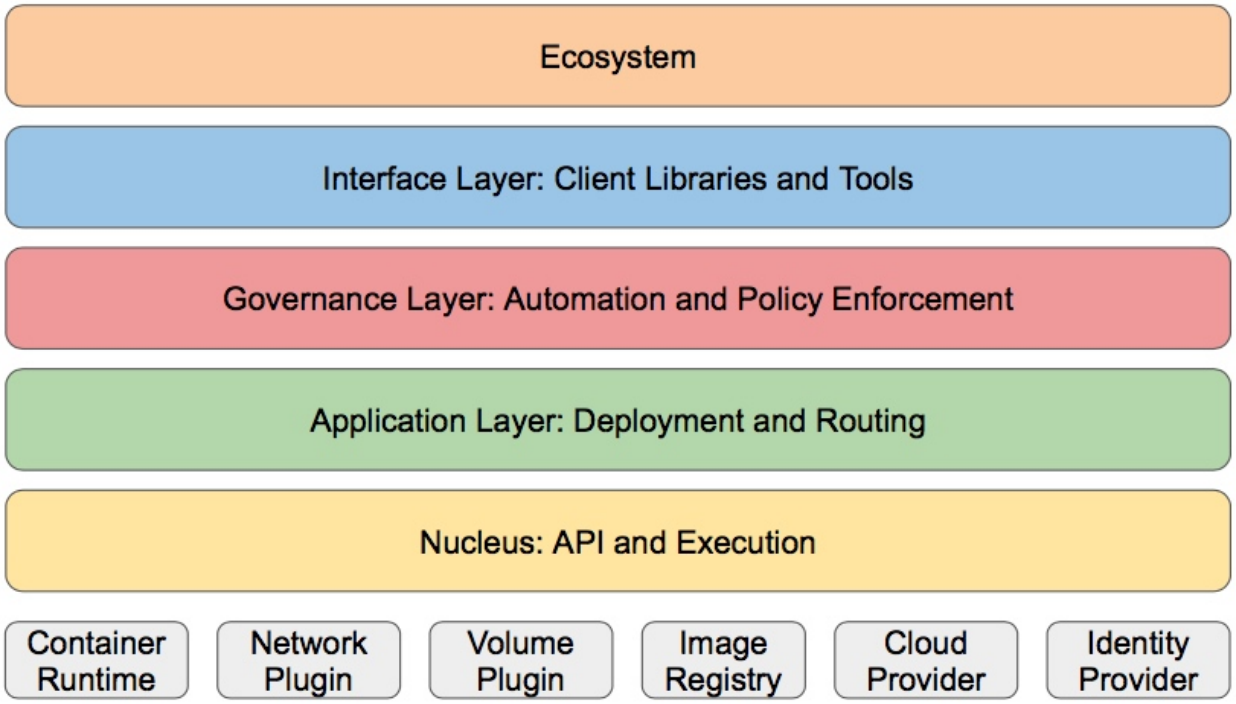
Node架构



图片 - kubernetes node架构示意图

分层架构

Kubernetes设计理念和功能其实就是一个类似Linux的分层架构，如下图所示



图片 - Kubernetes分层架构示意图

- **核心层：Kubernetes最核心的功能，对外提供API构建高层的应用，对内提供插件式应用执行环境**
- **应用层：部署（无状态应用、有状态应用、批处理任务、集群应用等）和路由（服务发现、DNS解析等）**
- **管理层：系统度量（如基础设施、容器和网络的度量），自动化（如自动扩展、动态Provision等）以及策略管理（RBAC、Quota、PSP、NetworkPolicy等）**
- **接口层：kubectl命令行工具、客户端SDK以及集群联邦**
- **生态系统：在接口层之上的庞大容器集群管理调度的生态系统，可以划分为两个范畴**
  - **Kubernetes外部：日志、监控、配置管理、CI、CD、Workflow、FaaS（Functions as a Service）、OTS（Open Table Service）应用、ChatOps等**

- **Kubernetes内部：CRI（容器运行时接口）、CNI（容器网络接口）、CVI、镜像仓库、Cloud Provider、集群自身的配置和管理等**