**Kubernetes 术语表**

Type

Architecture，Kubernetes内置组件；

Community，与Kubernetes相关的开源开发；

Core Object，Kubernetes默认支持的资源类型；

Extension，Kubernetes的自定义扩展；

Fundamental，Kubernetes新手看的基础定义；

Networking，Kubernetes组件如何交流，以及如何与集群外部交流；

Operation，启动并维护Kubernetes；

Security，Kubernetes安全性；

Storage，Kubernetes应用如何处理持久化数据；

Tool，使Kubernetes更易使用的软件；

User Type，Kubernetes用户的常见类型；

Workload，在Kubernetes上运行的应用。

A

Aggregation Layer：聚合层，允许在集群安装额外的Kubernetes风格的API

——Architecture

Annotation：key/value对，用于附加任意的非标识元数据到对象。

Application：各种容器化应用运行的层。

B

C

Cloud Controller Manager：云控制器管理器，是1.8的alpha特性，用于将Kubernetes与任意cloud集成。

——Architecture

Controller：控制器，一个控制循环，通过apiserver观察集群的共享状态，并在尝试将当前状态向目标状态移动时做出修改。

——Architecture

Cluster：一组机器，称为Nodes，运行容器化应用，被Kubernetes管理。

Container：轻量，可移植，可执行的镜像，包含软件和所需的所有依赖。

Container Runtime：用于运行container的软件。

Container runtime interface（CRI）：Container Runtime的API，与Node上的kubelet集成。

Control Plane：容器编排层，暴露API和接口，用于定义，部署，管理容器的生命周期。

CustomResourceDefinition：自定义资源代码，添加到Kubernetes API服务器而无非绑定一个完整的自定义服务。

Cgroup（control group）：linux进程组，具有可选的资源隔离，记录和限制。

D

DaemonSet：确保Pod的副本在群集中的一组节点上运行。

Data Plane：提供容量层，如CPU，内存，网络和存储，使得容器可以运行和链接网络。

Deployment：API对象，用于管理复制应用。

Device Plugin：设备插件是运行在Kubernetes上的容器，用于提供供应商特定资源的接入方式。

Docker：Docker引擎，用于提供操作系统级的虚拟化软件技术。

E

Etcd：一致性及高可用的key/value存储，用于Kubernetes所有集群数据的本地存储

——Architecture

Extensions：与Kubernetes深度集成的扩展组件，用于支持新的硬件类型。

F

G

H

I

Ingress：入口，API对象，用于管理外部访问，可以提供负载均衡，SSL终端和基于名称的虚拟主机。

——Architecture

Istio：开放平台（非Kubernetes专属），提供统一标准来集成微服务，管理流，执行策略和聚合遥测数据。

——Architecture

Image：镜像，存储容器的实例，包含运行应用所需的软件。

Init Container：一个或多个初始化容器，必须在其他应用容器运行前执行完成。

J

Job：一组有限的任务

K

Kubernetes API：通过RESTful接口服务Kubernetes功能的应用，并存储集群状态。

——Architecture

Kube-apiserver：master组件，暴露Kubernetes API，是Kubernetes控制面板的前端。

——Architecture

Kube-controller-manager：master组件，用于运行多个控制器。

——Architecture

Kube-scheduler：master组件，观察新创建兵没有分配node的pods，并选择一个让pod运行。

——Architecture

Kube-proxy：网络代理，在每个Node上运行，实现了部分Kubernetes Service的概念。

Kubectl：命令行工具，用于与Kubernetes API服务器通信。

Kubelet：代理，保证容器在pod中运行。

L

LimitRange：提供约束来限制命名空间中每个容器或pod的消耗。

——Architecture

Logging：通过集群或应用记录的事件。

——Architecture

Label：标签对象，定义了与用户相关的属性。

M

Minikube：本地运行Kubernetes的工具。

Mirror Pod：pod对象，kubelet用来代表一个静态pod。

N

Network Policy：网络协议，定义Pods群组如何被允许相互沟通以及与其他网络端点沟通。

——Architecture

Name：客户端提供的字符串，在资源地址中代表一个对象。

Namespace：Kubernetes使用的抽象，用于支持单个物理机上多个虚拟集群。

Node：Kubernetes上的工作机器。

O

Operator pattern：操作模式，是系统设计用于连接一个控制器到其他自定义资源。

——Architecture

P

Pod：最小且最简单的Kubernetes对象，代表一组运行中的容器。

Pod Lifecycle：高阶总结，说明Pod在什么阶段。

Pod Security Policy：激活Pod创建和更新的细粒度授权。

Q

QoS Class（Quality of Service Class）：提供验证Pods的方式及对计划和选举的决策。

——Architecture

R

Resource Quotas：提供约束用于限制每个命名空间聚合资源量。

——Architecture

RBAC（Role-Based Access Control）：管理授权决策，允许管理员通过Kubernetes API动态配置可用策略。

ReplicaSet：下一代复制控制器。

S

Selector：允许用户根据标签来过滤资源。

Service：一种抽象方式来暴露应用作为服务。

Service Account：提供进程的识别。

StatefulSet：管理一组pods的部署和扩展，提供这组pods的顺序和唯一性保证。

Static Pod：直接由kubelet守护进程管理的pod

T

Taint：核心对象，由key，value，effect三个必要属性组成。容错性允许在具有匹配的污点的节点或节点组上调度pod。

U

UID：Kubernetes系统生成的唯一标识对象。

V

Volume：包含有数据的目录，容器可以访问。

W

Workloads：用于管理和运行容器的对象。

X

Y

Z