[TECHO] | TVP Tencent Cloud Valuable Professional

/serverless/DAYS

Techo × TVP 开发者峰会

/serverless/ DAYS China 2021

无服务器，大有未来

Serverless, Empower More

腾讯云弹性容器服务EKS Serverless与Serverful的最佳平衡

腾讯云容器团队 于广游

于广游

腾讯云专家工程师

腾讯云容器团队技术负责人

2014年加入腾讯，主要经历:

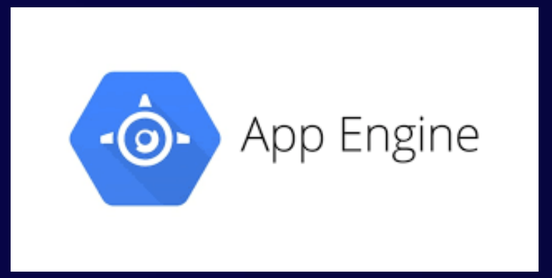
- 从0至今设计、开发腾讯云容器产品，运营了万级数量的k8s集群
- 腾讯自研业务全面云原生上云的主要参与者，深度参与了云原生在腾讯内部的落地

目录

2010年前，云计算的蛮荒时代，两大科技巨头
发布了两款非常不一样的产品，这是他们心中的云计算

云计算之IaaS路线

通过虚拟机替代物理机
按虚拟机配置计费
基于资源指标扩缩容
现在来看，这叫Serverful



云计算之PaaS路线

直接部署代码
应用实例计费
基于业务指标自动扩缩容
现在来看，这也属于Serverless

GAE限制过多、学习、迁移、开发成本太大，市场接受度不如EC2

2014年11月，两家厂商又推出两款迥异的云产品/技术

正式开创云原生概念

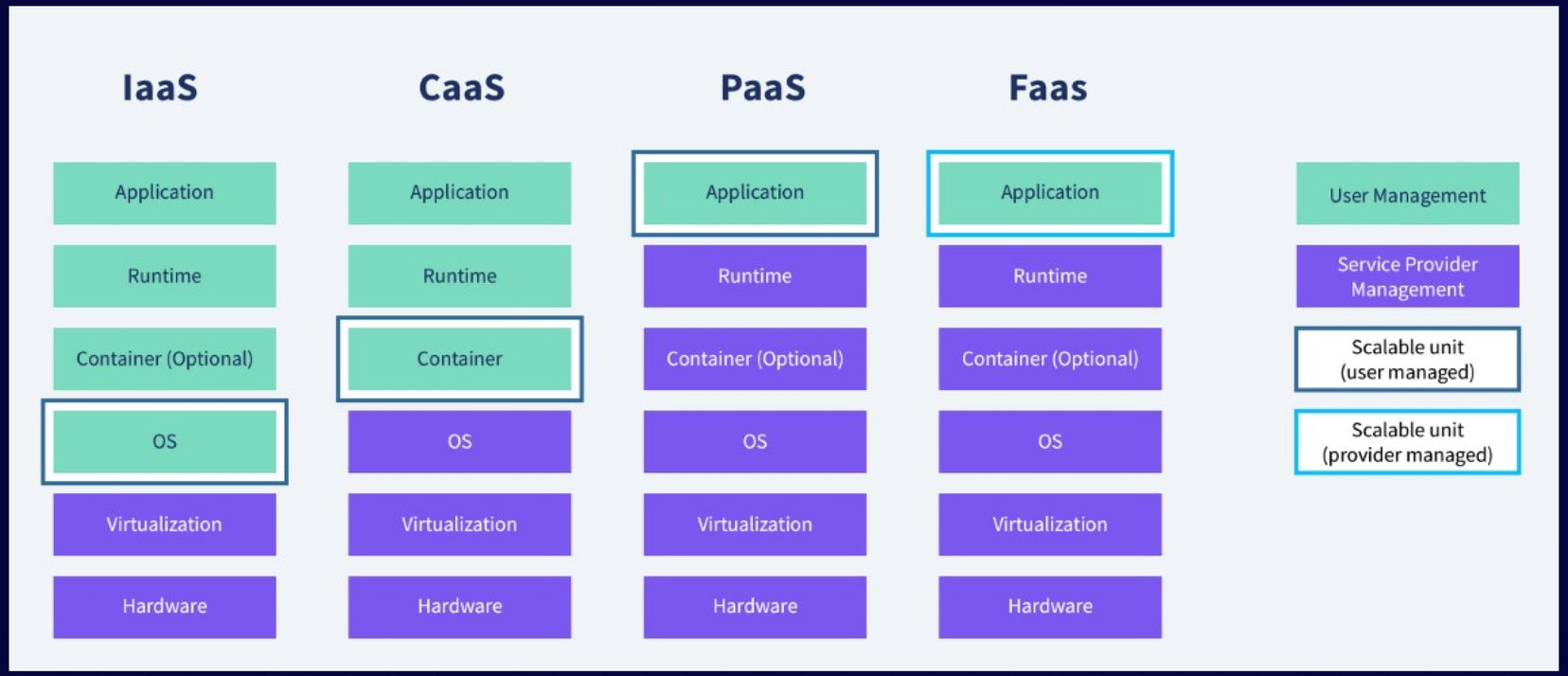
Serverless开始走向成熟

基于容器/kubernetes
简化基础设施管理
我们称之为CaaS

基于函数
完全免服务器管理
现在我们称之为FaaS

Serverless开始走向成熟；k8s由于通用性更强在行业形成了更广泛的影响

Serverful → Serverless



从Serverful到Serverless

抽象层级越高对用户意味着：

- 运维负担越少
- 弹性伸缩能力更强
- 计费模式更加按需

但也意味着：

- 灵活度下降
- 迁移成本更高

Serverless与通用性/迁移成本似乎是不可调和的矛盾？

K8S

优点

通用性：支持微服务、AI、大数据业务等全任务
可移植性：可运行在任意云环境中
可扩展性：可任意扩展调度平台本身的能力
迁移成本低：微服务业务容器化较为简单

不足

节点运维等工作极其复杂甚至超过传统虚拟机
按照预留节点收费，无法按照实际使用收费

FaaS

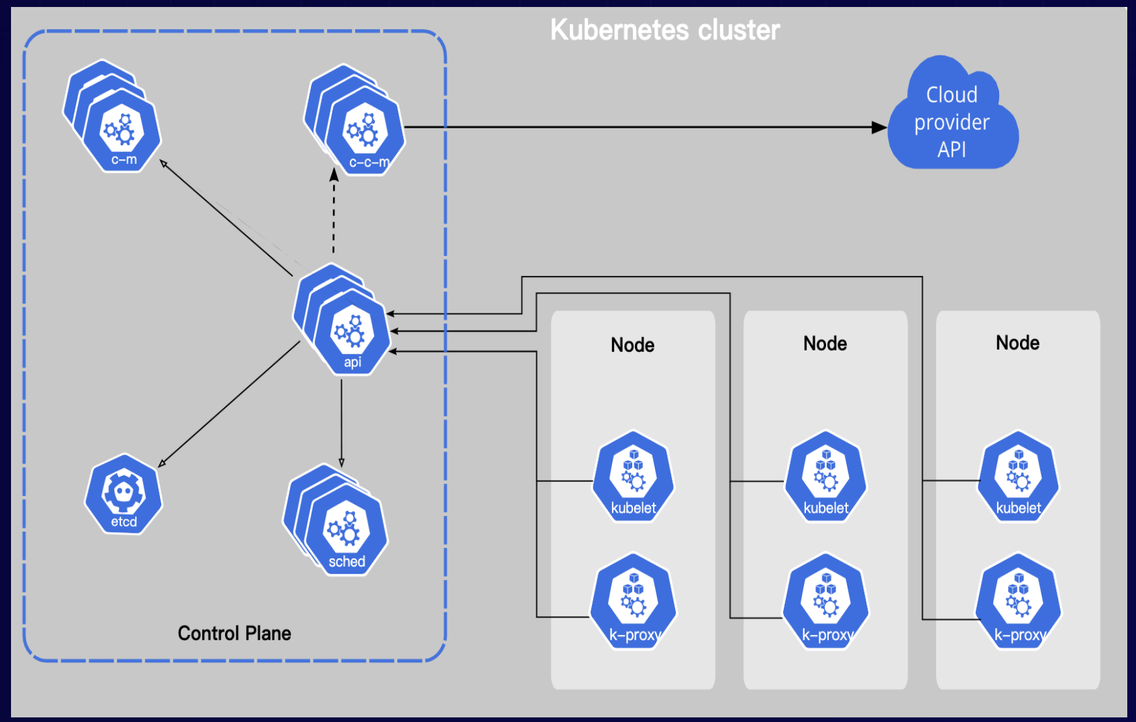
优点

NoOps：只用关心开发，而无需任何运维
极致弹性伸缩：无需关心任何弹性伸缩自动完成
按需付费：完全的按实际请求付费，无需为预留资源付费

不足

需要较大的代码迁移成本
难以支持存量框架

K8s与运维工作



K8S架构

- 由控制平面Master和运行实际计算任务的Node组成
- Node通常为物理机或虚拟机
- 通过k8s api创建出来的Pod(容器)，会调度到Node上运行

巨量的节点维护工作

- 众多测试集群、生产集群海量节点的日常维护；
- 节点os、内核、容器运行时的维护；
- 节点k8s组件的维护和升级；
- 为提升集群资源利用率进行复杂的节点扩缩容工作；

难以避免的容器隔离性问题

- Pod共享宿节点内核；
- Pod共享宿节点硬盘；
- Pod共享宿节点网卡；
- 隔离、干扰、安全；

专业、复杂的资源规划和调度工作

- K8s集群高资源利用率=内核隔离+GPU虚拟化+弹性+混布+成本分析+资源规划
- 集群资源规划须深入了解业务容器资源需求，资源不足则容器无法调度，对波峰业务需预留 buffer 资源。波谷时，预留资源浪费；

如何保留价值的同时降低门槛？

思考

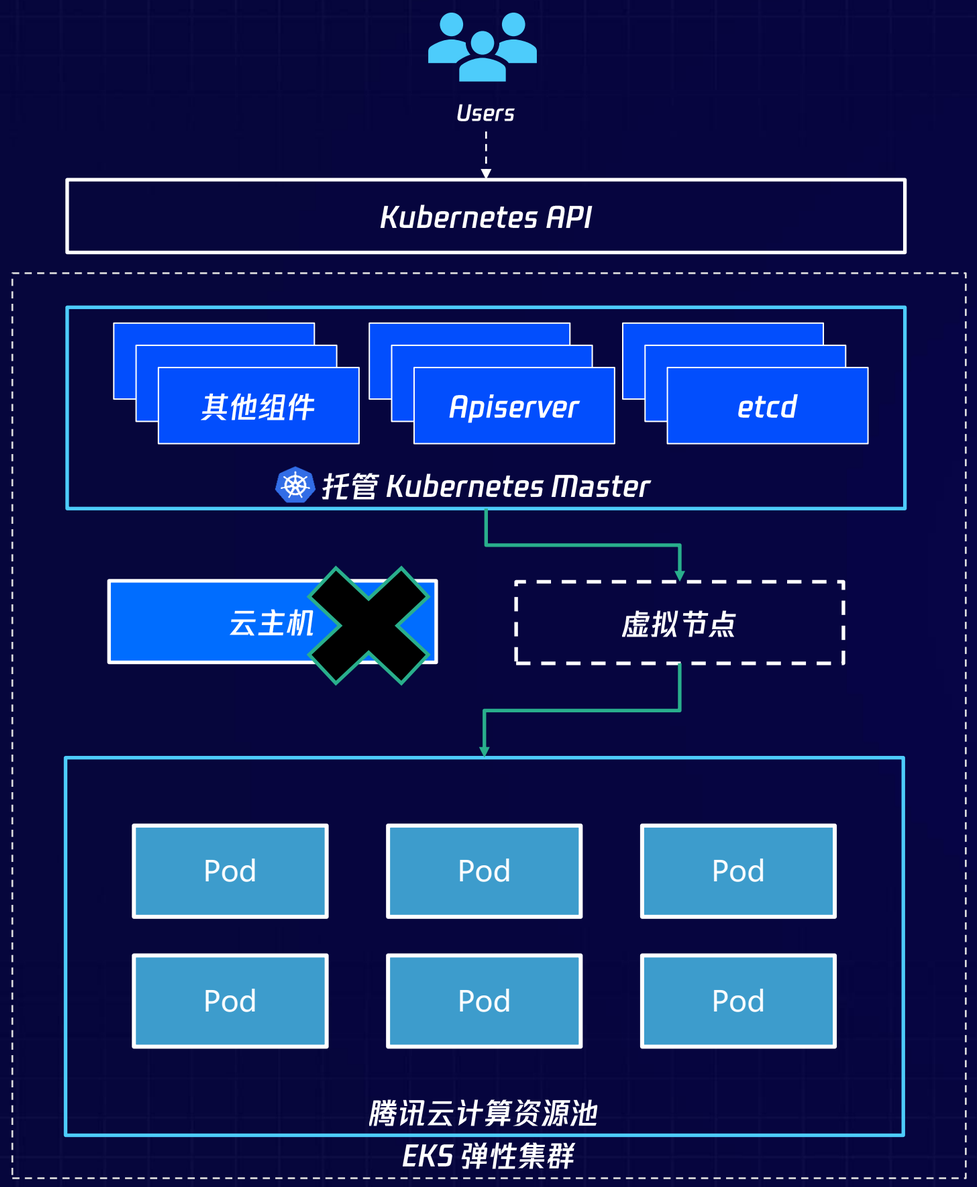
是否有一种产品/技术，兼顾kubernetes的通用性、低迁移成本，又有Serverless的高收益？
比如serverless kubernetes？

分析

K8s的主要价值在于k8s api带来通用性、可扩展性、可移植性
K8s的主要维护复杂度在于其node，也就是server的维护工作

保留k8s api能力，消除node的维护工作，是不是就能够兼顾通用性和Serverless？

腾讯云弹性容器EKS



弹性容器服务（Elastic Kubernetes Service，EKS）是一种全托管形式（Master托管、集群资源托管）的 Serverless Kubernetes 服务

Pod不是运行在真实Node（云主机）中，而是让k8s以为有一个无穷大的虚拟Node，调度过去的Pod实际运行在腾讯云整体大盘池子中。

- 完全标准的k8s Master，原生k8s兼容
- 采用 Nodeless 架构，以容器为交付资源，消除节点运维与资源规划工作
- 通过容器而非节点计费
- 极高的弹性伸缩能力

Serverless 架构

简化 K8s 集群运维复杂度，减少资源浪费，提高资源使用率

降低运维门槛的需求

资源规划

问题定位

升级维护

大量的运维工作和 K8s 使用门槛

使用 Kubernetes:

创建集群
需要基础 K8s 知识

连接集群
5分钟掌握

添加/移出节点
需要选择、购买服务器

部署应用
需要熟悉容器镜像，k8s 处理大部分工作

管理节点
需要专业的运维知识和 K8s 知识

管理应用
K8s 自动处理大部分工作

使用 EKS:

创建集群
需要基础 K8s 知识

连接集群
5分钟掌握

部署应用
需要熟悉容器镜像，K8s 处理大部分工作

管理应用
K8s 自动处理大部分工作

腾讯云弹性容器EKS-弹性

虚拟机弹性

步骤1：触发虚拟机扩容

触发方式：
- 虚拟机指标阈值
- 定时

时耗：秒级

步骤2：虚拟机扩容

扩容方式：
- 横向扩容（Autoscaling）

时耗：分钟级

步骤3：初始化和部署应用

- 基础环境
- 配置
- 应用包

时耗：分钟级

步骤4：服务注册

- 怎么服务注册？
- 怎么健康检查？
- 怎么流量接入？

时耗：分钟级

由于难以自动部署和服务注册
虚拟机用户一般不进行弹性扩缩容

Kubernetes 弹性

步骤1：触发 Pod 扩容

触发方式：
- 容器指标阈值
- 定时
- 事件
- 容器资源智能推荐

时耗：分钟级

步骤2：Pod pending触发虚拟机扩容

触发虚拟机扩容：
- 机型？配置？
- 子网？Zone？
- 调度策略、亲和性？
- 未来怎么缩回去？

时耗：分钟级

步骤3：Pod扩容成本并自动注册服务发现

- Pod自动调度
- 自动注册服务发现
- 自动健康检查
- 自动切流量

时耗：秒钟级

传统k8s服务需要pod和node两层弹性伸缩联动较为复杂，用户一般只进行pod伸缩，不进行Node伸缩

EKS 弹性

步骤1：触发 Pod 扩容

触发方式：
- 容器指标阈值
- 定时
- 事件
- 容器资源智能推荐

时耗：秒级

步骤2：Pod 扩容并自动注册服务发现

扩容方式：
- 横向扩容（HPA）
- 纵向扩容（VPA）

服务注册：
- 自动注册服务发现
- 自动健康检查
- 自动切流量

时耗：秒级

分钟内扩容上千实例，上万核心
多种触发方式
支持Pod横向、纵向扩缩容
自动进行服务注册并切流
自动跨zone打散、容灾
高效易用的弹性伸缩

腾讯云弹性容器EKS-资源利用率对比

物理/虚拟机模式

没有编排和调度能力

↓

资源必须以整机的形式交付
业务间无法有效隔离

↓

不同业务独占资源池，资源碎片严重
应用独占设备，资源利用率低且无法腾挪、复用

↓

CPU利用率通常在10%左右

传统Kubernetes利用率

有编排和调度能力

↓

节点内可能有干扰
节点易伸难缩

↓

不敢密集混布
不敢大量使用弹性伸缩

↓

比虚拟机略高，配置弹性后 CPU 利用率可超过 15%，深入混部后可达30%～40%

EKS 利用率

有编排和调度能力

↓

以Pod交付，无混布无干扰
无需资源规划和浪费

↓

轻松落地弹性伸缩

↓

配置弹性伸缩后，资源利用率轻松超过50%

标准的 Kubernetes 服务和使用方式

满足使用者对容器交付标准的需求
提供了标准的 K8s 编排、运维、扩缩容、日志、监控等使用方式

大部分容器场景对交付标准的需求

- 基于 Kubernetes 的应用编排
- 基于容器的交付模式
- 支持丰富的服务接入、存储、配置、权限管控能力
- 具备故障检查与自恢复机制
- 具备完善的可观测性

标准K8s社区生态

- 保留 K8s 扩展性（插件、CRD）
- 兼容 K8s 生态（基础能力、可观测性、云原生解决方案）

密钥与配置管理

- 支持存储和管理敏感信息，例如密码、OAuth 令牌和 ssh 密钥。支持在不重建容器镜像的情况下更新密钥和应用程序配置，也无需在堆栈配置中暴露密钥。

K8s 对象编排

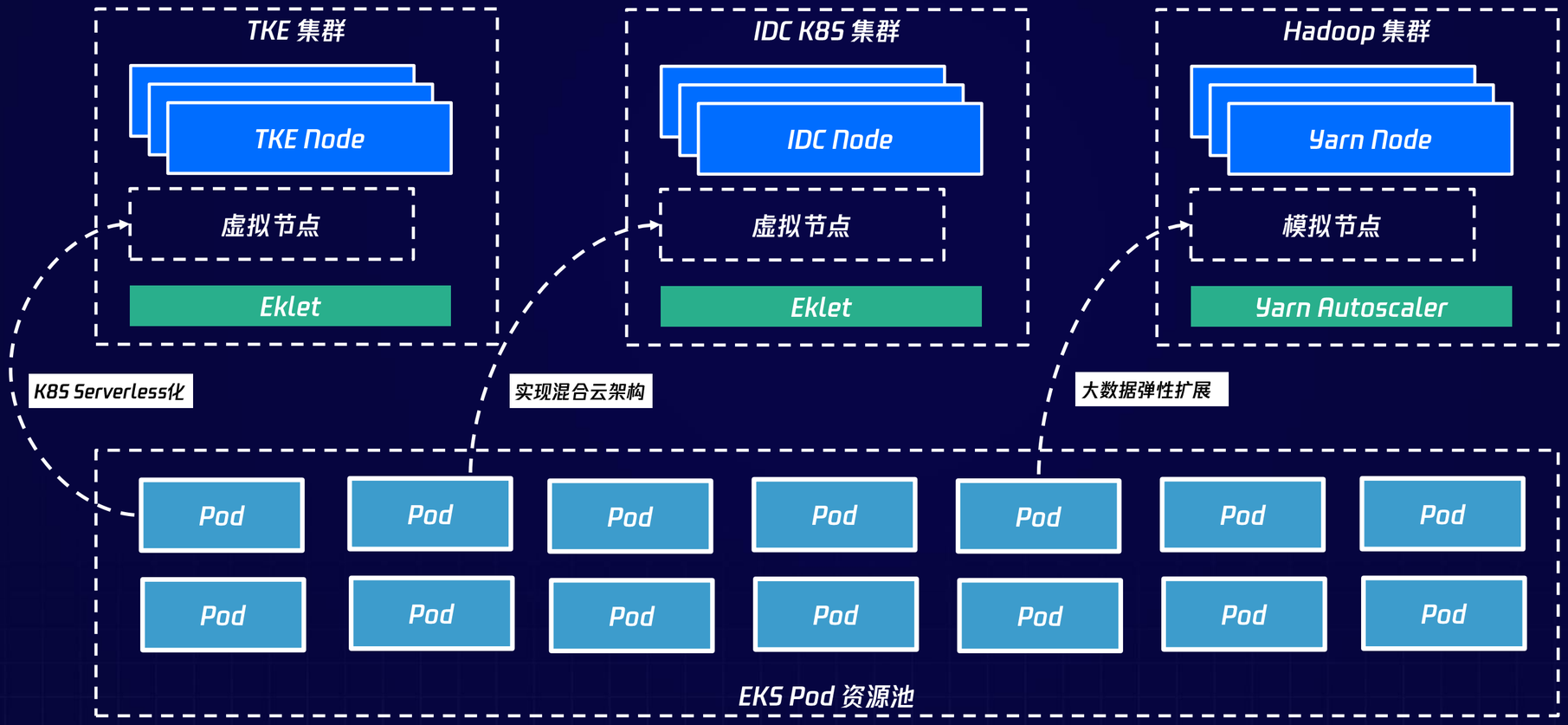
- 支持原生 k8s 工作负载部署模式;
- 支持自动创建服务接入和服务发现;
- 支持自动挂载存储系统，例如云盘、云文件系统等。

自我修复

- 自动重启失败的容器、自动替换容器、杀死健康检查失败的容器，并支持自动对客户端屏蔽不健康容器副本。

腾讯云弹性容器EKS-作为虚拟节点加入其他集群

EKS 支持在公有云或 IDC 的标准 Kubernetes 集群、Hadoop 集群部署 插件，将扩容算力弹性调度到 EKS Pod 资源池，调度模式完全兼容原集群调度方式，可让原集群平滑进行 Serverless化或混合云化升级。



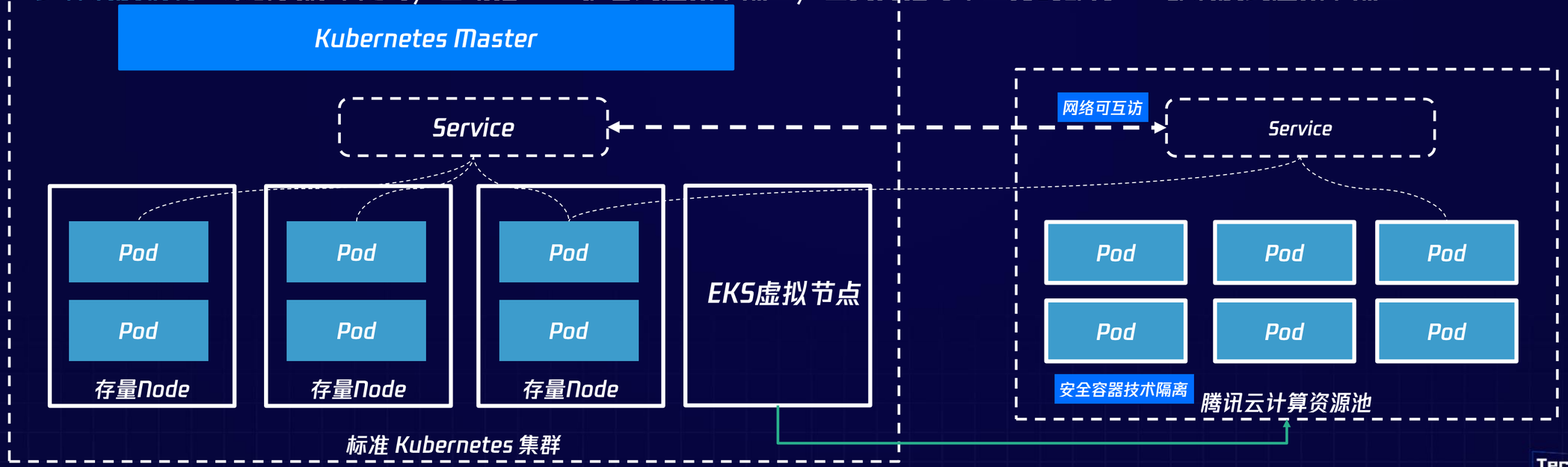
腾讯云弹性容器EKS-微服务渐进式迁移

业务serverless化时，通常需要大量的迁移、改造成本

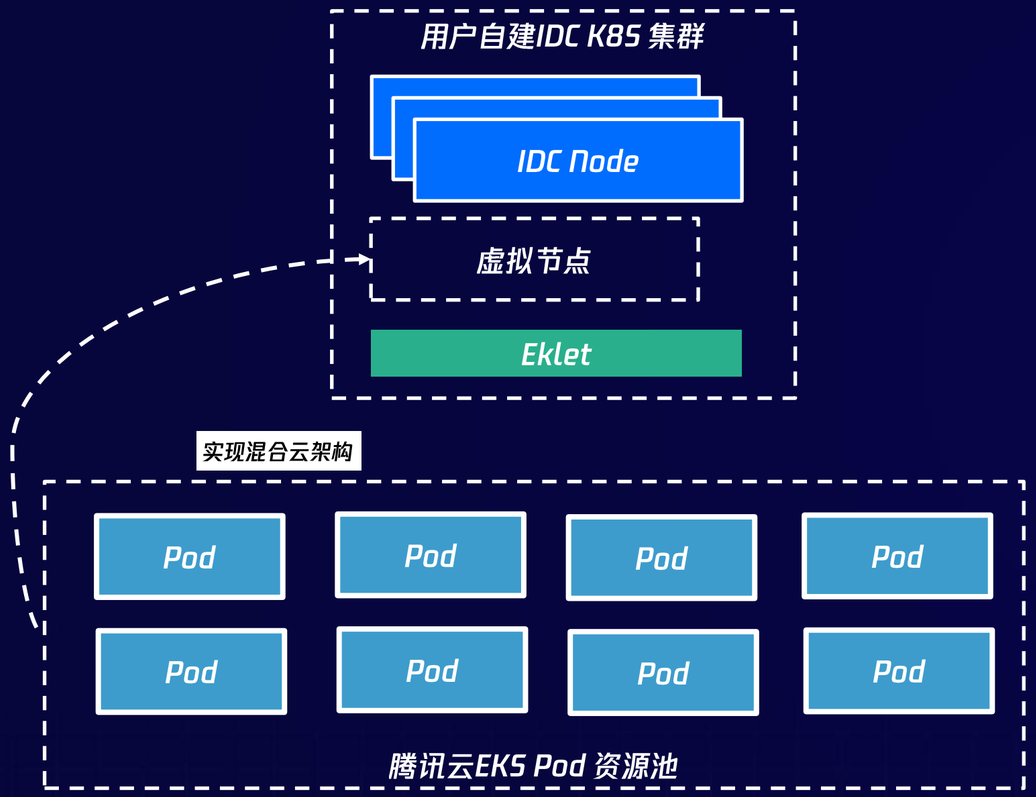
EKS支持作为虚拟节点加入存量k8s集群中并与存量节点并存。通过修改业务调度策略，将Pod从真实节点调度到虚拟节点上，即可完成serverless改造

通过此方法可按需的、渐进的、可回退的、0成本的对业务进行serverless化改造。

- 插件化完全兼容　支持扩展任何标准 k8s 集群（TKE、TKE 混合云、云上/IDC 自建 k8s、第三方云 k8s 等），与存量节点并存，网络互通不受影响
- 多种调度机制　集群资源不足时，自动把Pod扩容到虚拟节点上；也支持把每个业务的部分Pod都调度到虚拟节点上



腾讯云弹性容器EKS-0成本组建混合云



企业通常有通过混合云进行容灾或提升弹性能力应对突发的需求，通常情况下企业需要实施极其复杂的混合云架构

EKS可作为计算节点导入到用户IDC存量k8s集群中并与k8s集群中Pod网络、服务发现全部互通，0成本组件混合云架构

腾讯云弹性容器EKS-大数据场景

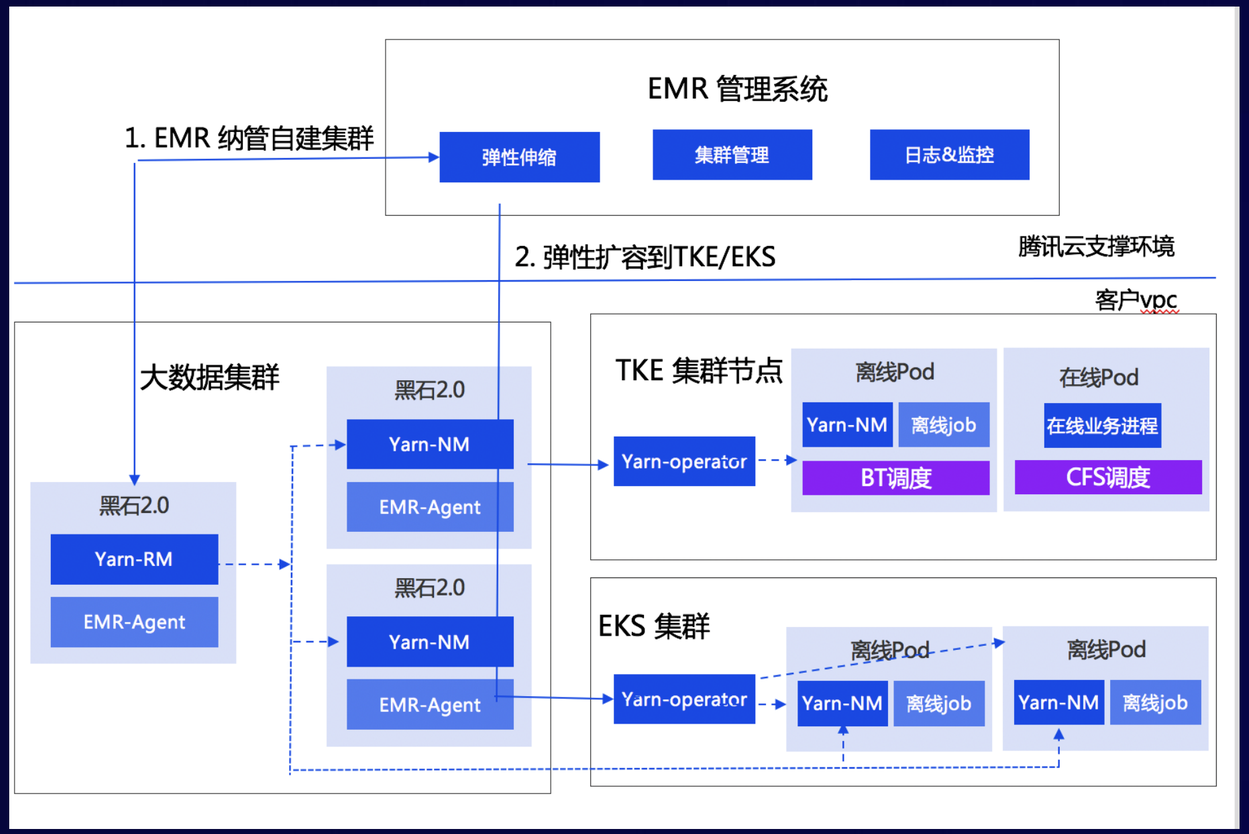
大数据集群的计算任务通常运行在物理机和虚拟机中，面对潮汐较为明显的任务，会产生大量的资源浪费

无缝接入：EKS可作为计算节点导入到任意基于yarn的大护具集群中，而无需大数据平台任何改造

极致利用率：平台可灵活选择将任务运行在存量节点还是EKS集群中，EKS可随任务数量进行弹性扩缩容，0资源浪费

动态将计算任务运行在EKS中

腾讯云弹性容器EKS-客户案例



腾讯云某客户既有在线业务，又有离线业务，两种业务独立部署，资源利用率不高，希望能够提升资源利用率

弹性与混布

- 腾讯云大数据容器化方案对大数据业务进行渐进式容器化
- 在线业务和大数据业务混合部署在k8s 节点上
- 资源不足时动态扩容EKS Pod，无需进行资源规划和节点运维

成果

- 通过优化资源碎片，在离线混合部署，自动扩缩容，整体计算成本下降43%
- 同时有效的支持业务快速迭代，秒级急速扩缩容

进一步Serverless -- 更弹、更省钱

无需设置资源配置

无负载时缩容到 0

以业务指标（如QPS）而非资源进行弹性

根据实际使用CPU/MEM而非预留资源计费

总结与思考

THANK YOU!

感谢聆听！