Cloud-Native 与分布式数据库

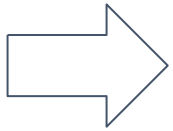
黄东旭 @ PingCAP

关于我

- 黄东旭，PingCAP 联合创始人 & CTO
- MSRA / Netease / Wandoulabs / PingCAP
- Infrastructure engineer / Open-source hacker
- Codis / TiDB / TiKV

数据库怎么了？

- 业务形态多种多样
- 接入终端五花八门
- 存储成本持续降低

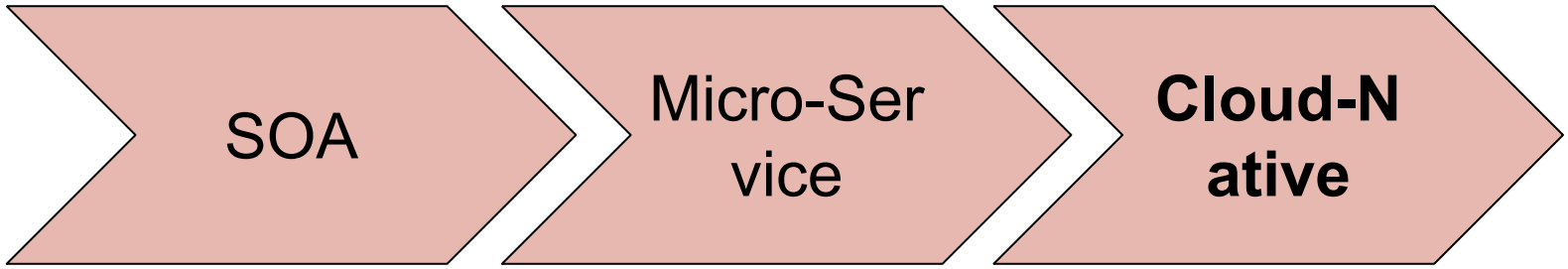


海量数据

数据库怎么了？

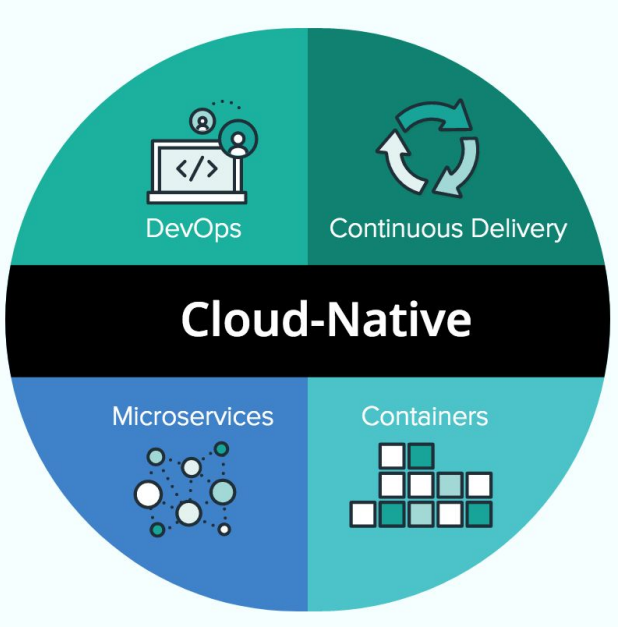
- 对开发效率的无止境渴求
- 关系型数据库仍然是业务的核心
- 扩展性是新时代基础软件的第一要素
  - Everything is WEB-SCALE!

Cloud-Native 是什么？



Cloud-Native 是什么

- **Scale** 作为一等公民
- Micro-service 友好
- 面向容器的部署
- 自管理



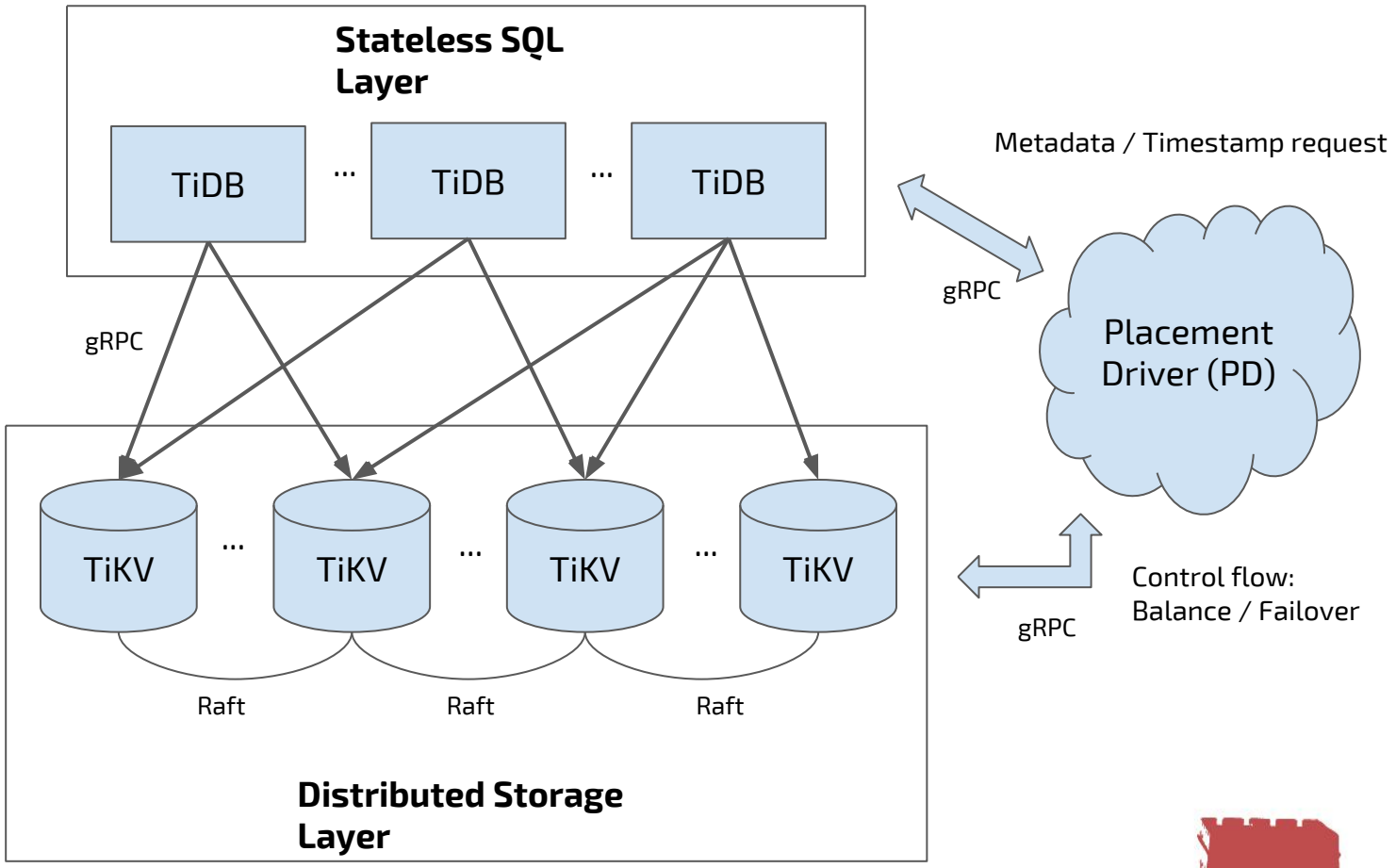
构建 Cloud-Native 的基础设施的两个条件

- 存储本身的云化
- 部署和运维方式本身云化

存储本身的云化

- 使用 Raft / Paxos 这类更先进的一致性协议替换传统的主备高可用
- 使用自动分片策略取代人工预分片
- 接入层去状态化
- 架构中避免一切单点

TiDB


Stateless SQL Layer
TiDB
TiDB
TiDB
gRPC
Metadata / Timestamp request
gRPC
Placement Driver (PD)
TiKV
TiKV
TiKV
TiKV
gRPC
Control flow: Balance / Failover
Raft
Raft
Raft
Distributed Storage Layer

TiDB

TiDB - NewSQL 数据库结合了：

- NoSQL 的弹性伸缩能力
- 传统关系型 SQL 数据库的易用性

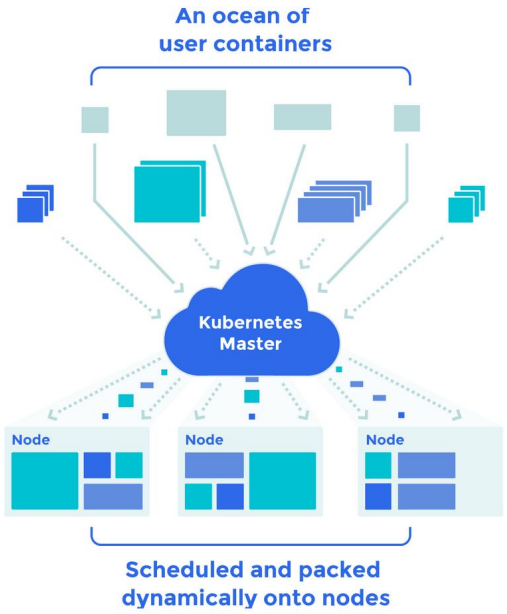
TiDB

- 可扩展性: 完全自动分片 (TiKV)
- 可用性: Raft 保证
- 一致性:强一致性（External consistency, 2PC）

‘Can you have a scalable database without going NoSQL? Yes.’

部署运维方式云化：Kubernetes

- Google 的大规模集群调度系统 Borg 的后继
- 自动集群调度
- 服务编排
- 自动化运维
- DCOS？



最大的问题：状态

- 整个应用层分裂成4个的阵营：
  - Stateless applications
  - Single point stateful applications
  - **Static distributed applications**
  - **Clustered applications ← 老大难**

Single point stateful application

- 单点带状态服务
  - MySQL
  - PostgreSQL
  - Redis
- 使用 Static configuration or StatefulSets

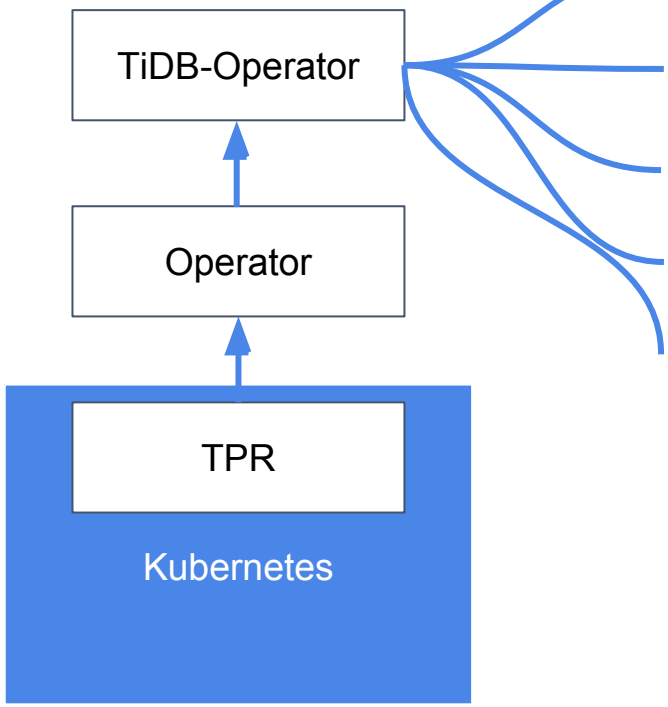
Static distributed application

- 分布式静态带状态服务
  - ZooKeeper
  - Etcd
- StatefulSets

Operator

- 思想很简单
- Putting Operational Knowledge into Software
- CoreOS 出品
- 依赖 K8S ThirdPartyResources API
- *K8S 不知道该怎么调度我，所以我来指导 K8S*

TiDB Operator



- Create
- Rolling update
- Scale out
- Failover
- Backup

Why Operator

- 对 Kubernetes 的侵入性小
- PV / StatefulSet 并没有解决问题或者还不够

没有银弹

- 自增 ID
- Time stamp ordering
- 业务自身存在冲突和热点
  - 秒杀
  - 业务压力集中在小表
- MySQL 容量达到单机瓶颈
- 读写分布相对平均
- 并发事务支持，但是大多数时间冲突不高
- 有实时的复杂查询需求

Thanks