云原生社区 Cloud Native Community | SOFASTACK™

云原生开放日·上海站

Cloud Native Open Day Shanghai

蚂蚁万级规模K8S集群 etcd 架构优化实践 — ETCD on OceanBase

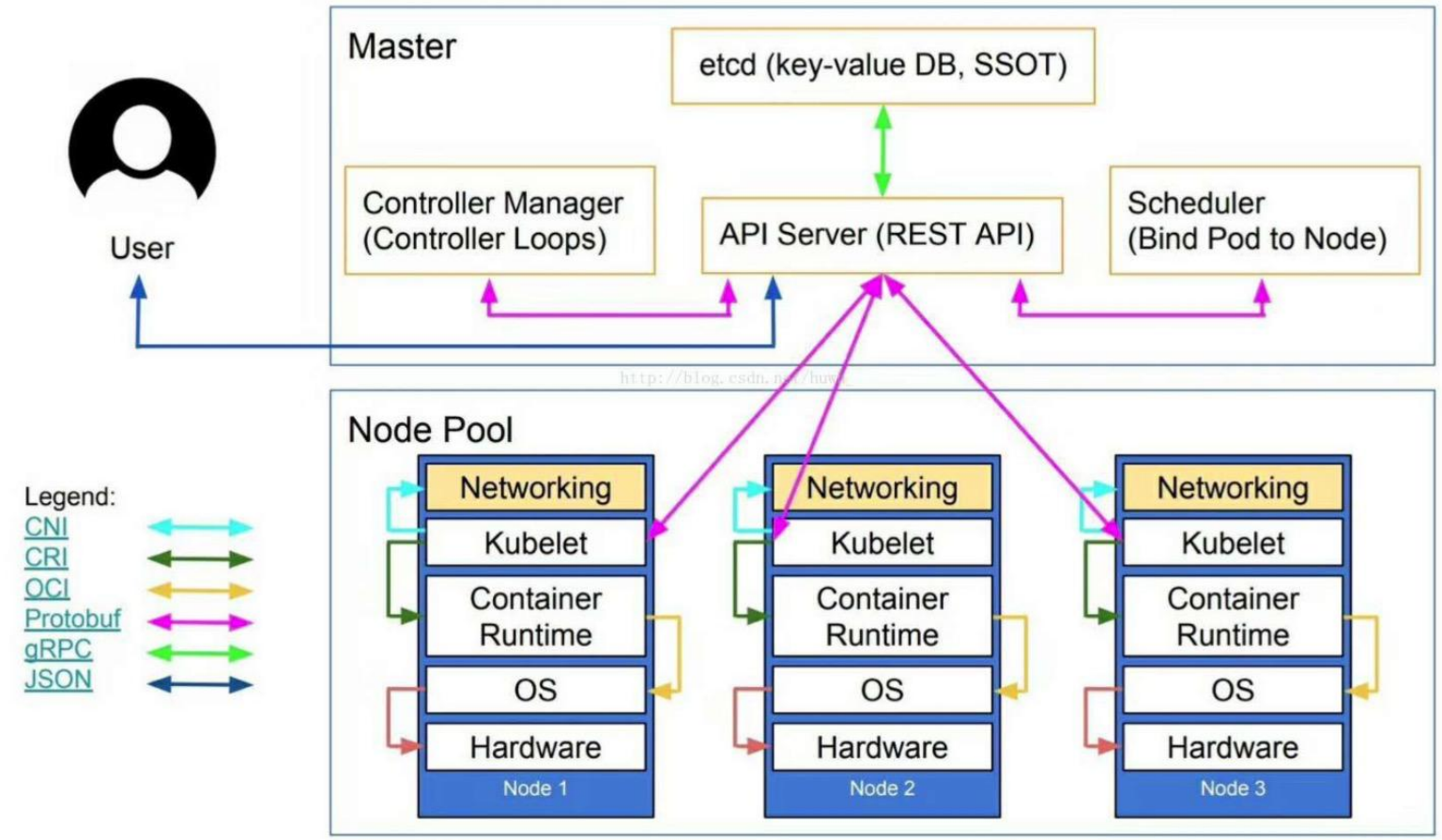
宣　超（锡 林）

蚂蚁数据库技术

目　录

蚂蚁etcd使用现状

- 200+ etcd 集群
- 百万节点规模
- 单集群10K+ 节点
- 数据量超40G
- 机房独立部署
- 小时级备份



蚂蚁etcd使用现状

etcd V3.0 → etcd V3.3 → etcd V3.4

- etcd3 API
- gRPC
- backend
- MVCC

……

- safe read
- N reads, 1 write
- readTx buffer

……

- Raft learner
- FCR
- Client balance

……

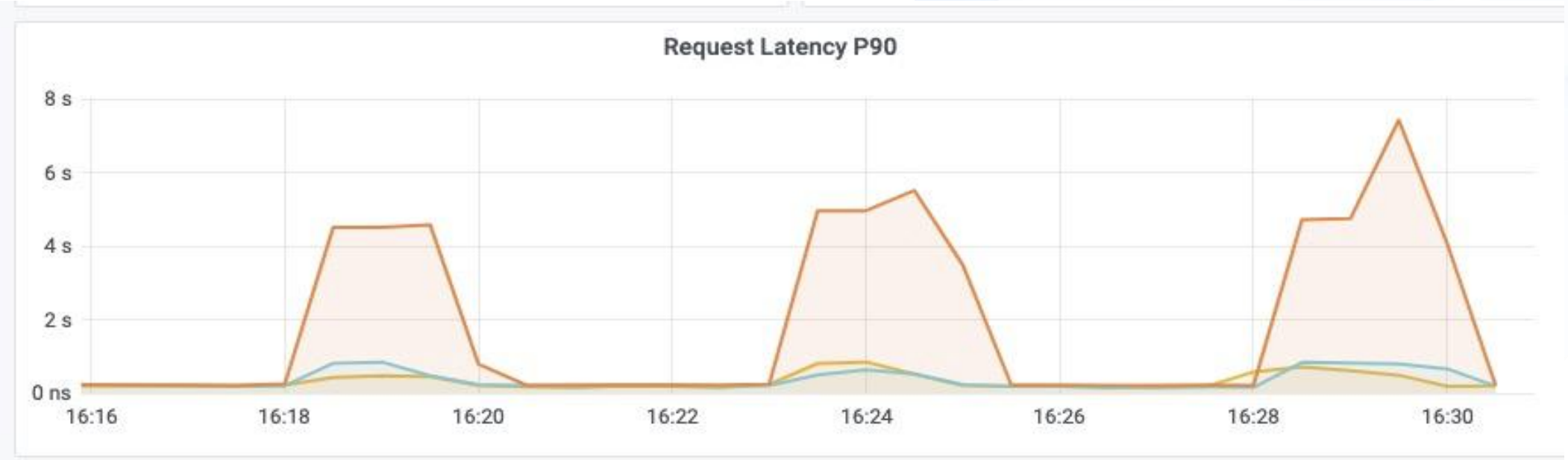
蚂蚁etcd使用现状

痛点：单线程 raft，大规模压力下基本不可服务

	rate	Request
		etcd 3.4
put	20K	13696.7669
range	20K	8957.6919
put * 3	20K*3	error 60%
range * 3	20K*3	7522.2006

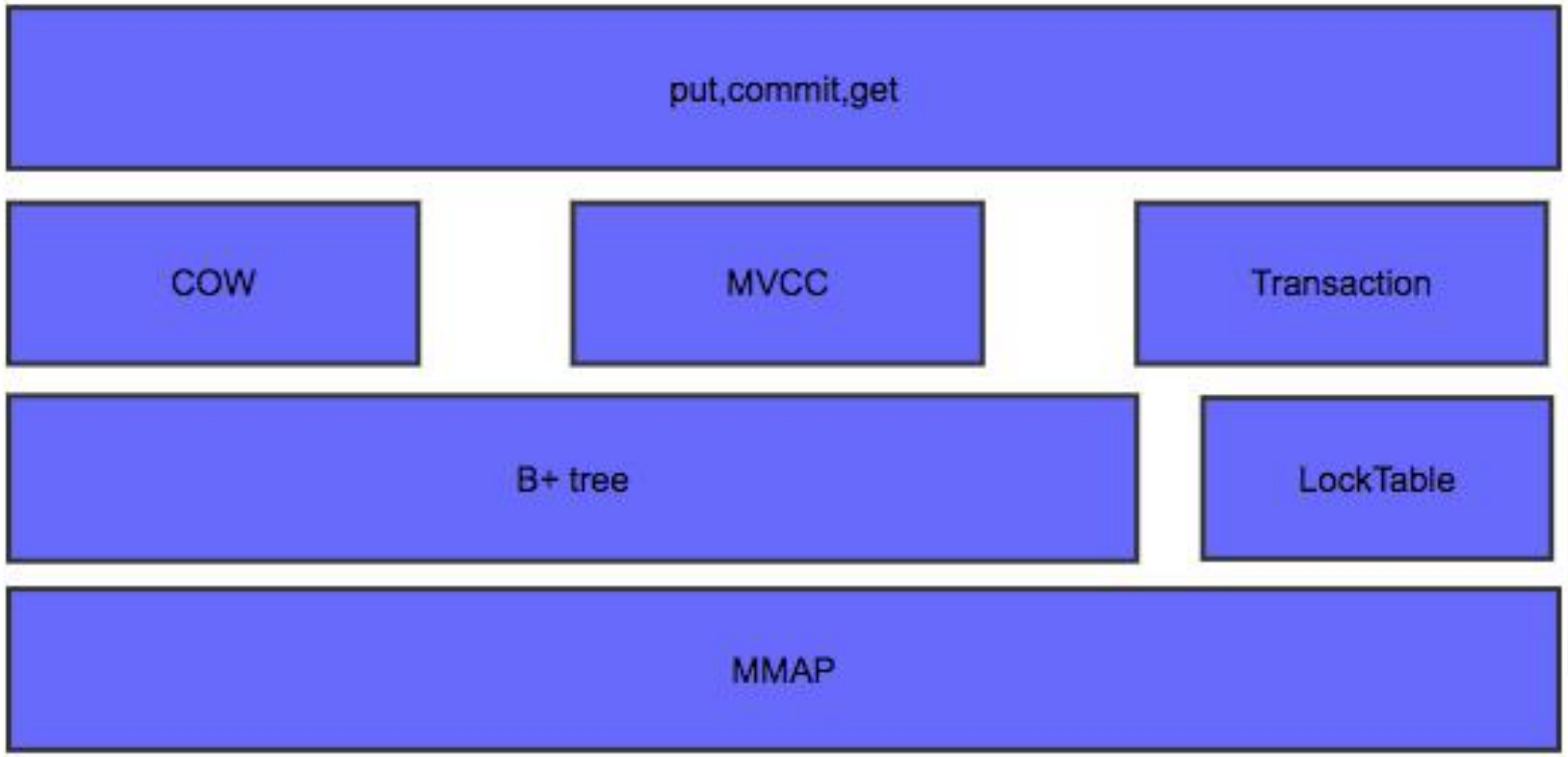
蚂蚁etcd使用现状

痛点： compact 造成线上抖动



蚂蚁etcd使用现状

痛点：boltdb



蚂蚁etcd使用现状

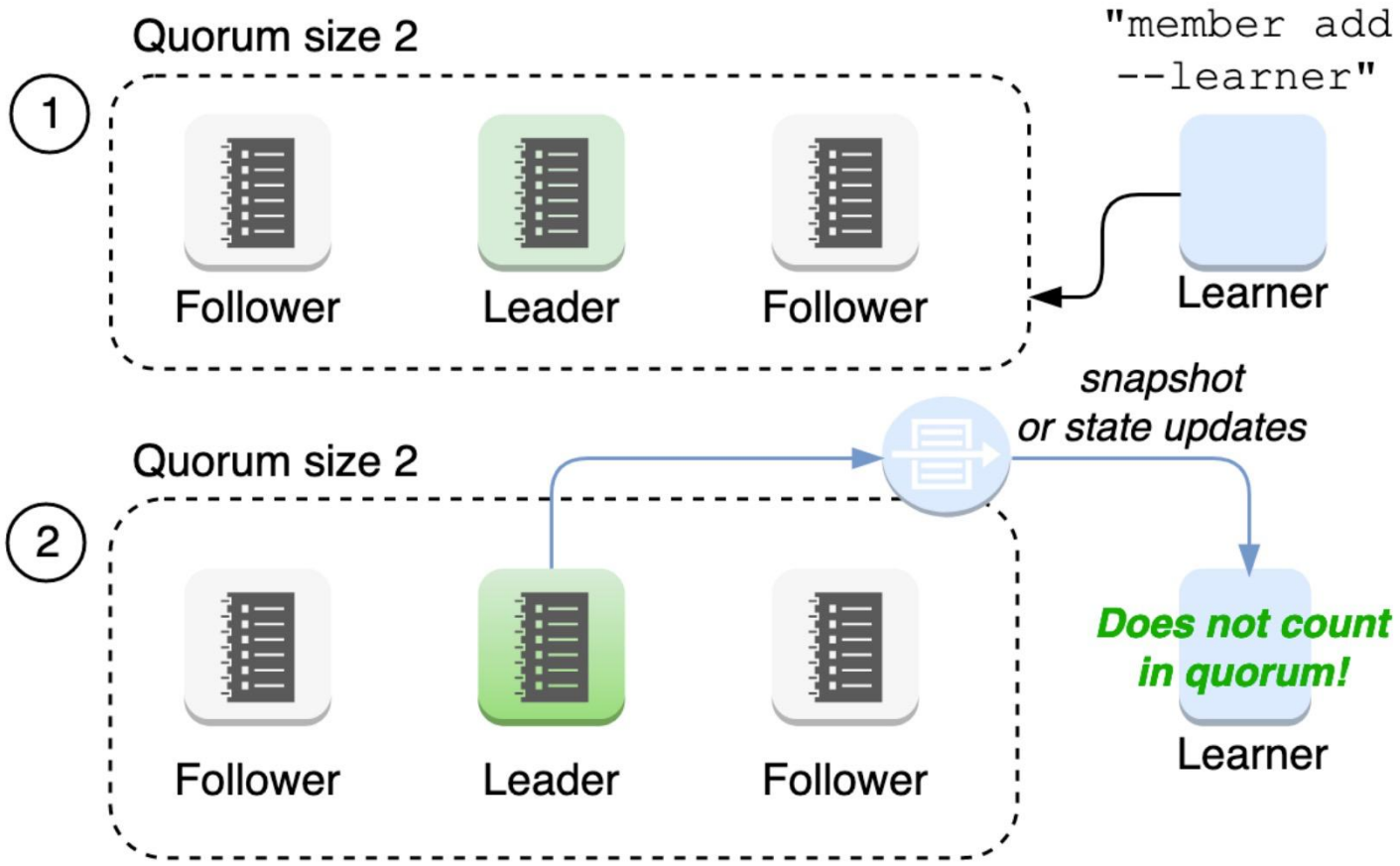
痛点：boltdb

- 一写多读限制写性能

Test step	LeveLDB	RocksDB	HyperLevelDB	LMDB
Write 100M values	36m8.29s	21m18.60s	**10m45.41**	**1h13m21.30s**
DB Size	**2.7G**	3.2G	3.2G	**7.6G**
Query 100M values	2m55.37s	**2m44.99s**	**13m49.01s**	5m24.80s
Delete 50M values	3m47.64s	**1m53.84s**	6m0.38s	**6m15.98s**
Compaction	3m59.87s	**3m20.27s**	6m33.36s	**1.548us**
DB Size	**1.4G**	1.6G	1.6G	**7.6G**
Query 50M values	12.12s	13.59s	**23.98s**	**8.48s**
Write 50M values	3m5.28s	**1m26.9s**	1m54.56s	**3m25.96s**
DB Size	**673M**	993M	928M	**2.5G**

蚂蚁etcd使用现状

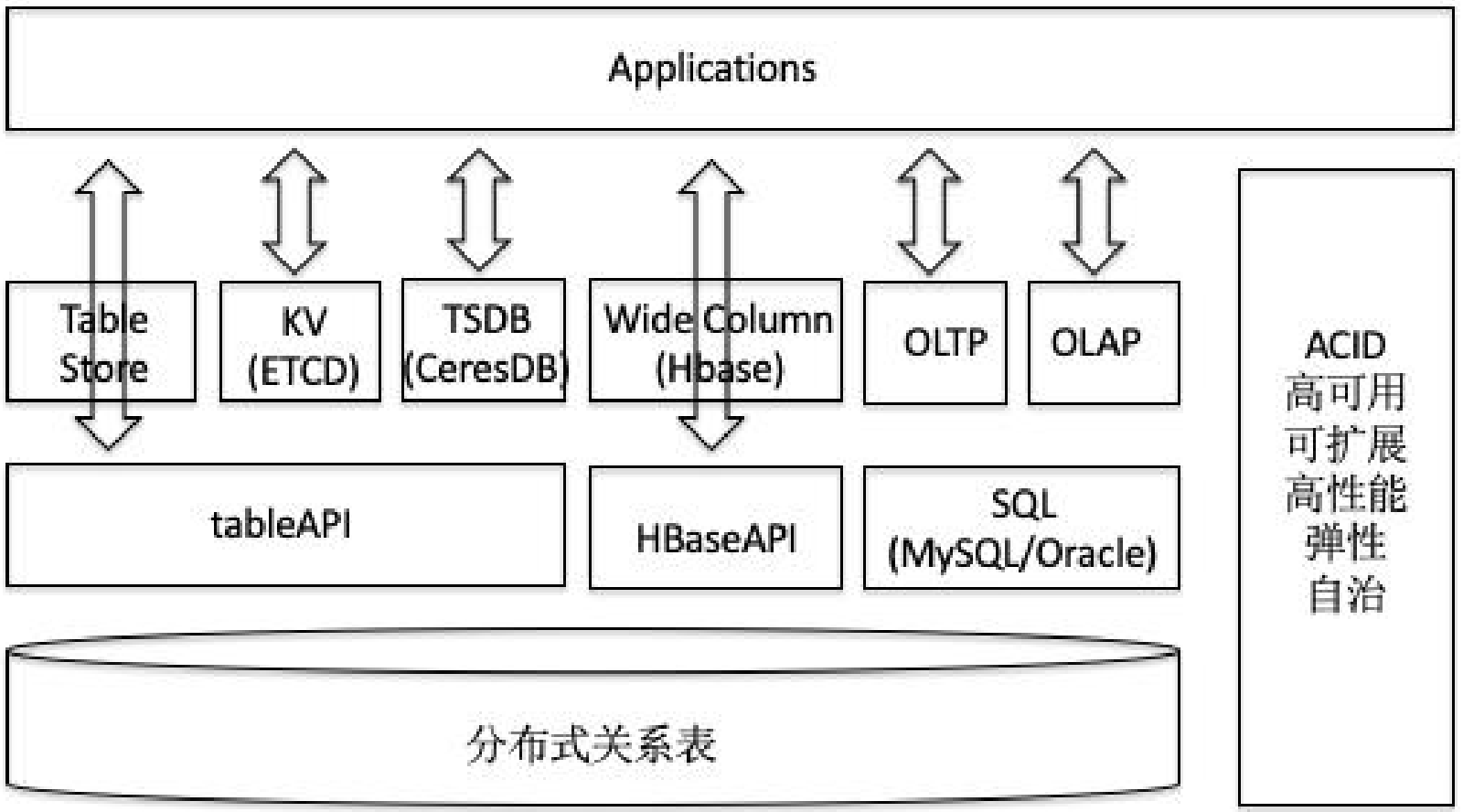
痛点：容灾能力 & 部署模式 有限



目　录

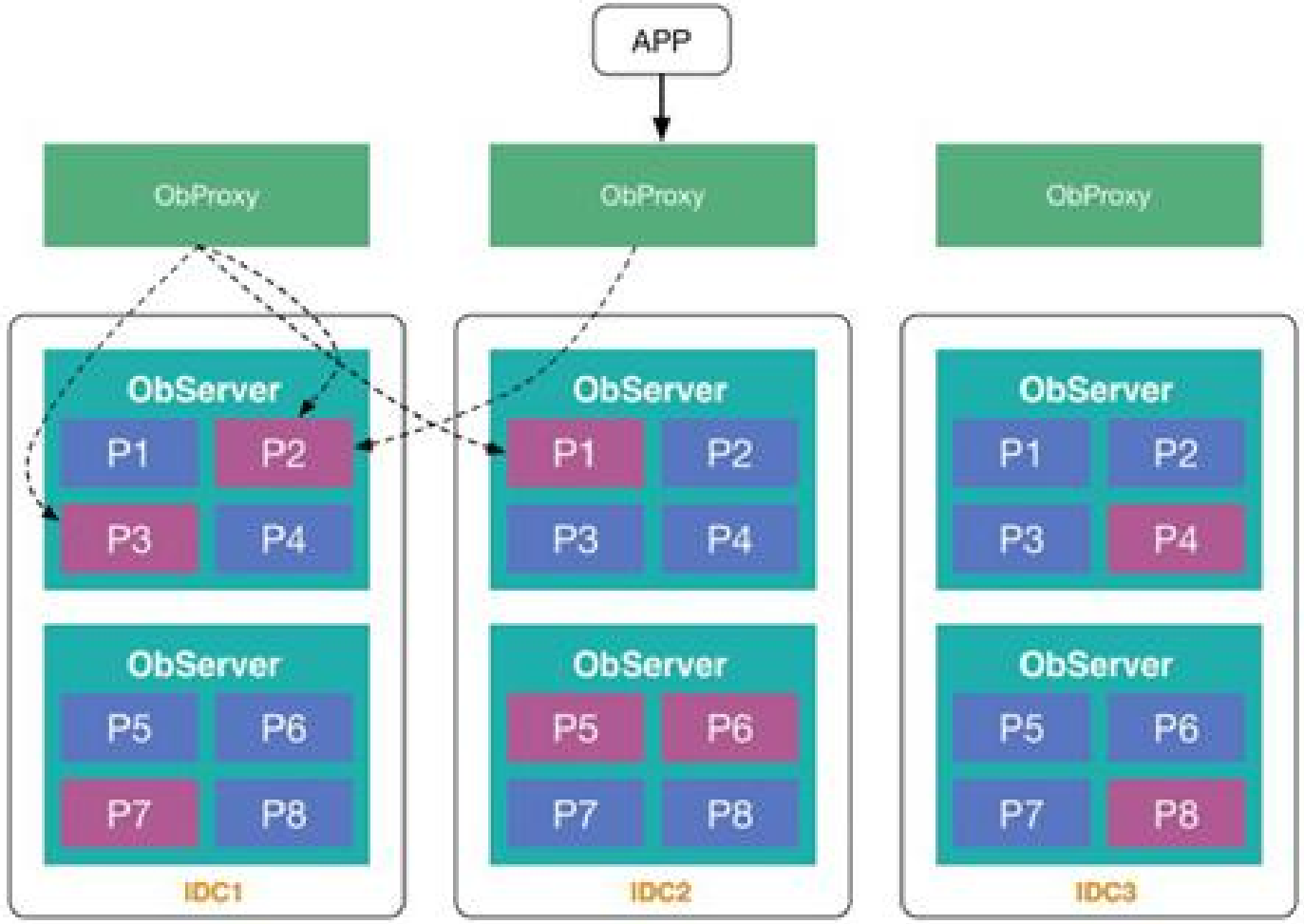
OceanBase 简介

- 蚂蚁自研
- 多模型
- 开　　源
- 高可用 & 高扩展
- HTAP
- 分布式事务
- 只读副本
- Paxos

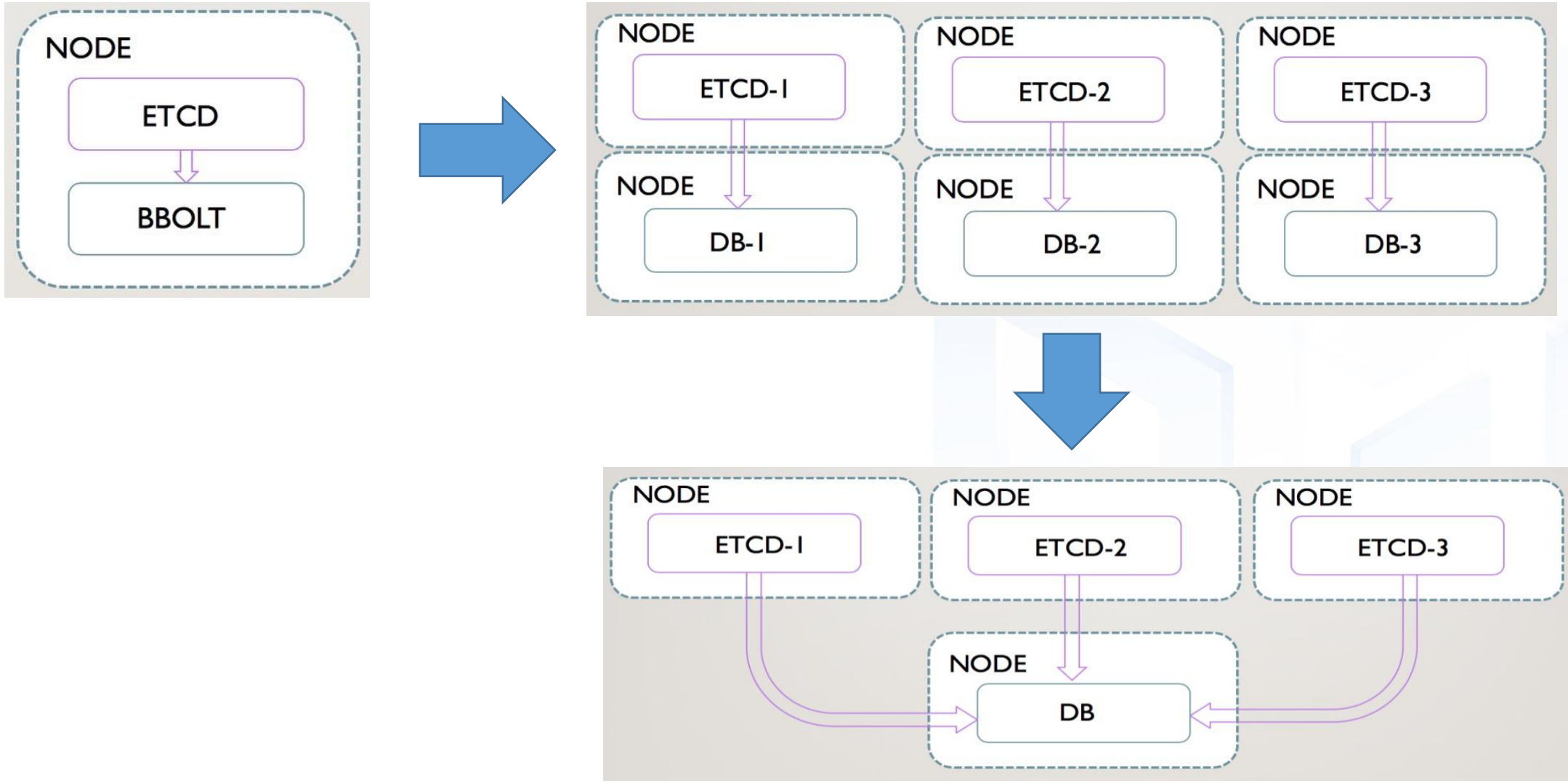


蚂蚁部署实践

- 同城3副本(机房容灾)
- 2地3中心(异地容灾)
- 3地5中心(城市容灾)

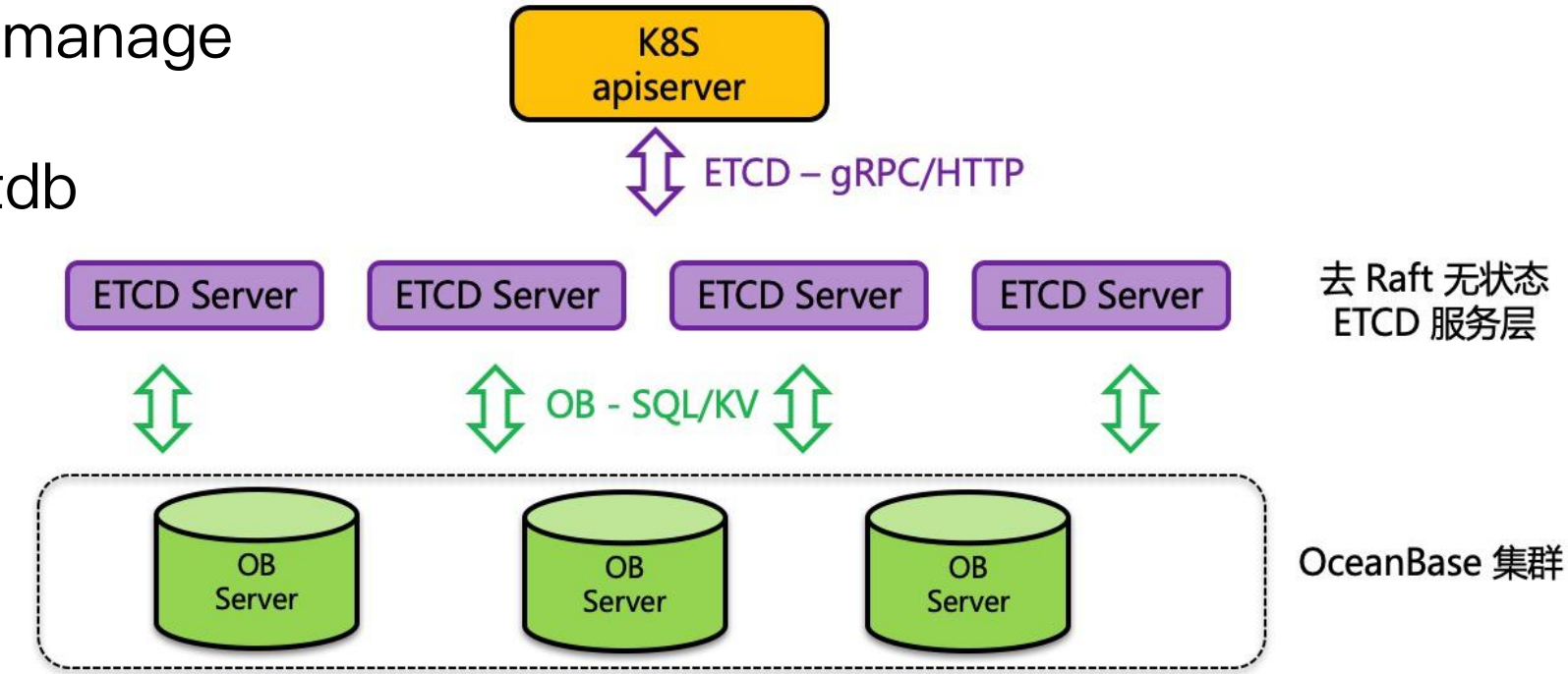


方案论证


NODE
ETCD
BBOLT
NODE
ETCD-1
DB-1
NODE
ETCD-2
DB-2
NODE
ETCD-3
DB-3
NODE
ETCD-1
NODE
ETCD-2
NODE
ETCD-3
NODE
DB

方案介绍

- 梳理 rpc 协议: kv、auth、manage
- 新增 dbbackend 替换 boltdb
- 基于 OB 表设计数据模型



dbbackend

- sql.DB
- ETCD_ENABLE_RDB_STORE
- ETCD_RDB_CONNECT_STR

……

rpc协议

/etcdserverpb.KV/Range

/etcdserverpb.KV/Put

/etcdserverpb.KV/DeleteRange

/etcdserverpb.KV/Txn

/etcdserverpb.KV/Compact

/etcdserverpb.Watch/Watch

/etcdserverpb.Lease/*

OB表存储模型

- etcd_kv
- etcd_event
- etcd_lease
- etcd_meta
- etcd_time

方案介绍

- KV： 当前 KV 视图
- Event: KV 的所有修改事件
- Lease: 租约，KV 的 TTL机制

ETCD_Lease	
ID	
TTL	
EX	

ETCD_KV	
Key	
Value	
Revision	
CreateRev	
Version	
LeaseID	

ETCD_Event	
Key	
Value	
Revision	
CreateRev	
Version	0 ： Deleted
LeaseID	

产品对比

	etcd	etcd On OB
存储规模	100G	PB级
部署模式	机房内部署	同城三节点 2地3中心 3地5中心
容灾能力	机房内单节点容灾	机房容灾、 城市容灾
数据统计能力	只能统计基本的存储大小信息， 缺乏丰富的统计	丰富的统计信息，SQL
数据备份	定时snapshot， 热备依赖learner和make mirror	支持每日全备、增量备份、 实时热备、主备互切，
冷热数据存储	不区分冷热数据	冷热数据区分存储，充分利用机器资源
数据分区	不支持	支持range分区、hash分区、二级分区等
多租户能力	不支持	支持
监控运维	监控基于prometheus和grafana，部署方便；运维能力基于自开发平台	监控基于内部表统计；运维基于自研平台

性能对比

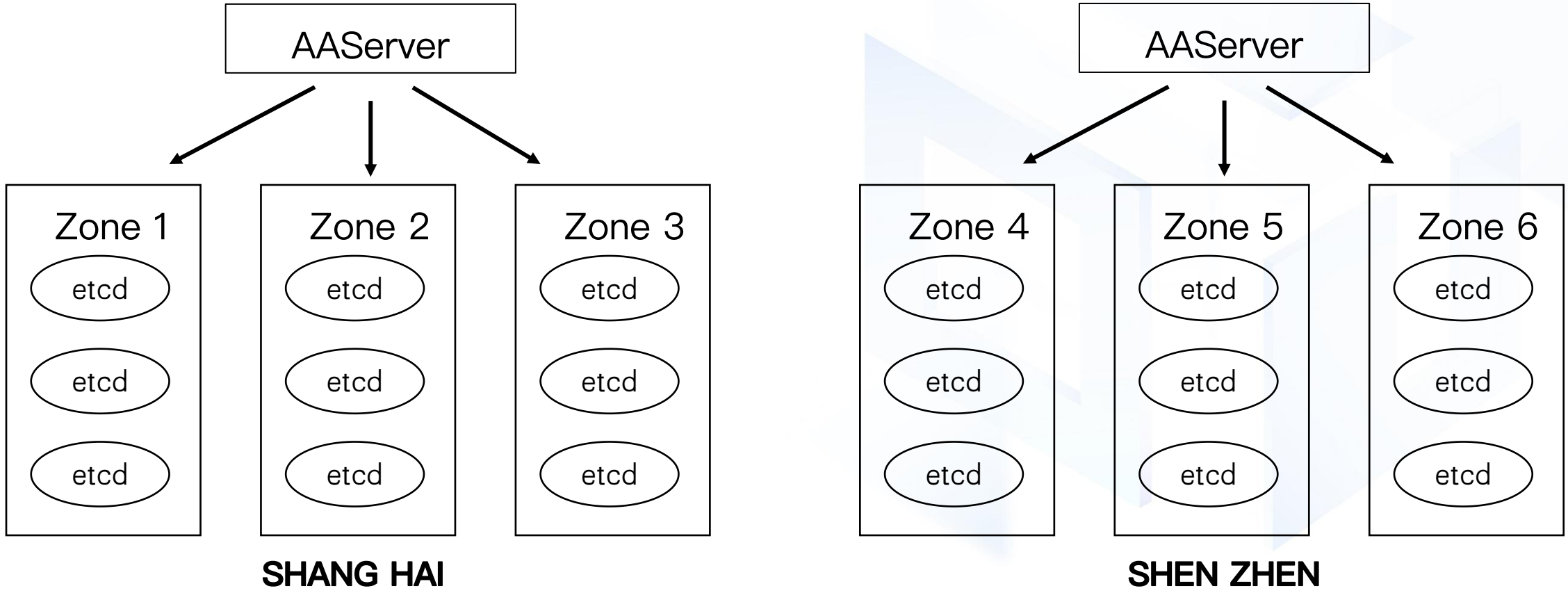
- put、range 性能基本持平
- etcd On OB 高并发的压力下表现优于 etcd 3.4。在高并发情况下（200 * 3 clients），etcd On OB 的 put tps 14773，range qps 14823。而 opensource etcd 3.4 在此并发下，put 错误率飙升到60%，原因为其 apply 的速度跟不上并发请求。
- etcd On OB 的 watch延迟平均在 17.8 ms。由于基于 OB 实现方式，采用轮询的方式读取数据更新，而 etcd 3.4 则是在内存中直接通知 watch，延迟比 etcd 3.4 差在预期之内。 这点仍在做优化

	rate	Requests/Sec		Avg Reponse(s)		Latency P90	
		etcd 3.4	etcd on ob	etcd 3.4	etcd on ob	etcd 3.4	etcd on ob
put	20K	13696.7669	13911.4268	0.0066	0.0121	0.0097	0.0176
range	20K	8957.6919	8672.7164	0.0216	0.0441	0.0421	0.0816
put * 3	20K*3	error 60%	14773.3872	N/A	0.0407	N/A	0.0735
range * 3	20K*3	7522.2006	14823.3094	0.0797	0.0393	0.1486	0.0906
put/range 1:1	20K	11206.1504	9202.7167	0.0126	0.0211	0.0191	0.0400
	20K	6815.6203	6489.5087	0.0289	0.0306	0.0469	0.0593
put/range 3:1	30K	11499.2318	10184.2352	0.0106	0.0187	0.0170	0.0357
	10K	6600.0087	6121.3073	0.0279	0.0318	0.0469	0.0601
put/range 1:3	10K	8926.4975	8584.5248	0.0099	0.0193	0.0155	0.0378
	30K	7404.5701	6623.9851	0.0267	0.0301	0.0436	0.0563
watch-latency	1K	34383.8996	6231.5323	0.0008	0.0178	0.0034	0.0269
			6650.5017(*)		0.0075(*)		0.0089(*)

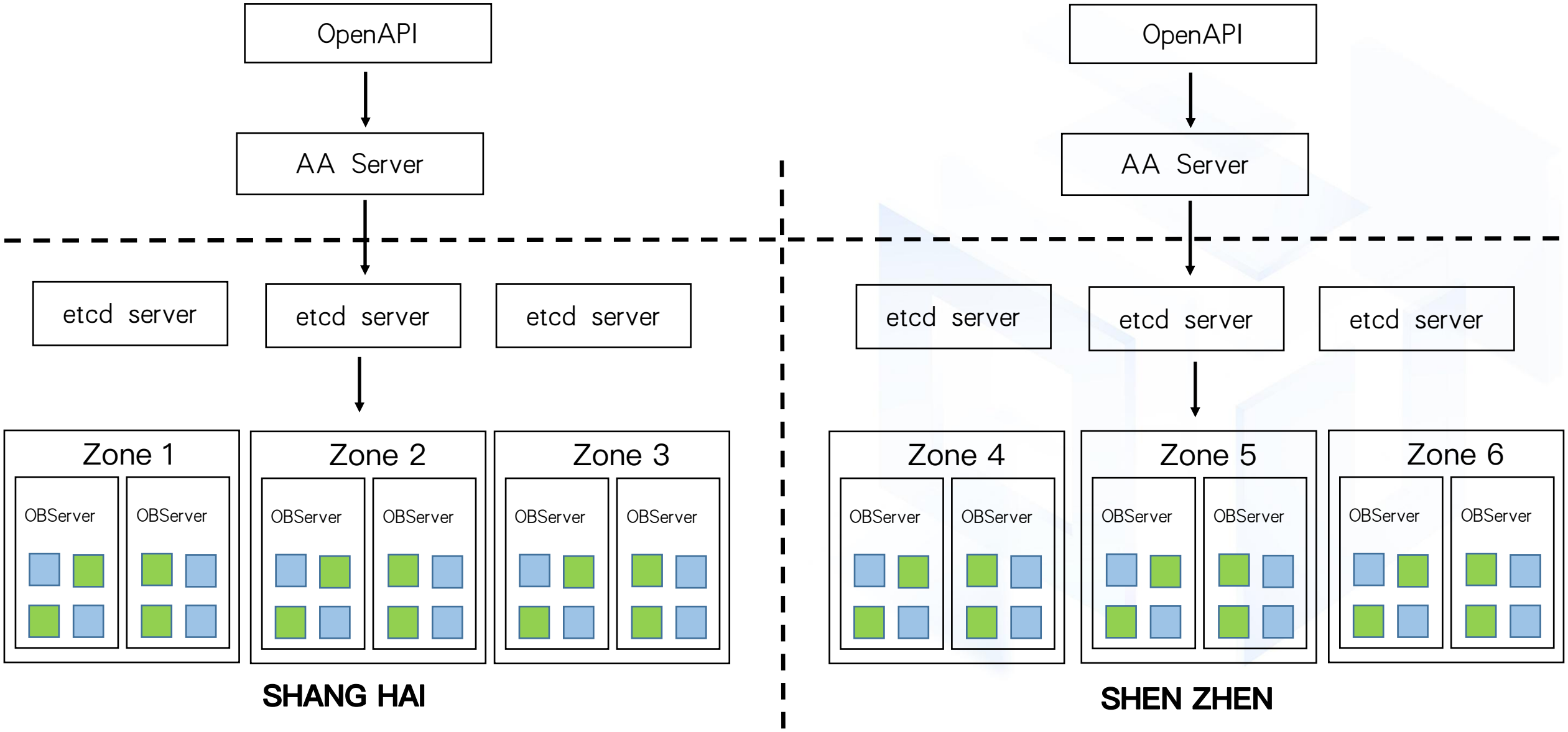
目　录

应用实践 —— DBMesh CR

DBMesh 配置（Custom Resource）数量很多， 对 etcd 造成明显的压力，数据量超过etcd的存储能力。



应用实践 —— DBMesh CR



应用实践 —— Identity

- 单集群写，多集群读
- 对时延没有高要求
- 保障数据最终一致性，例如：set a = 1; set a = 2; set a = 3。最终a必定等于3
- 用到watchPrefix、get、put接口

应用实践 —— Identity

方案一

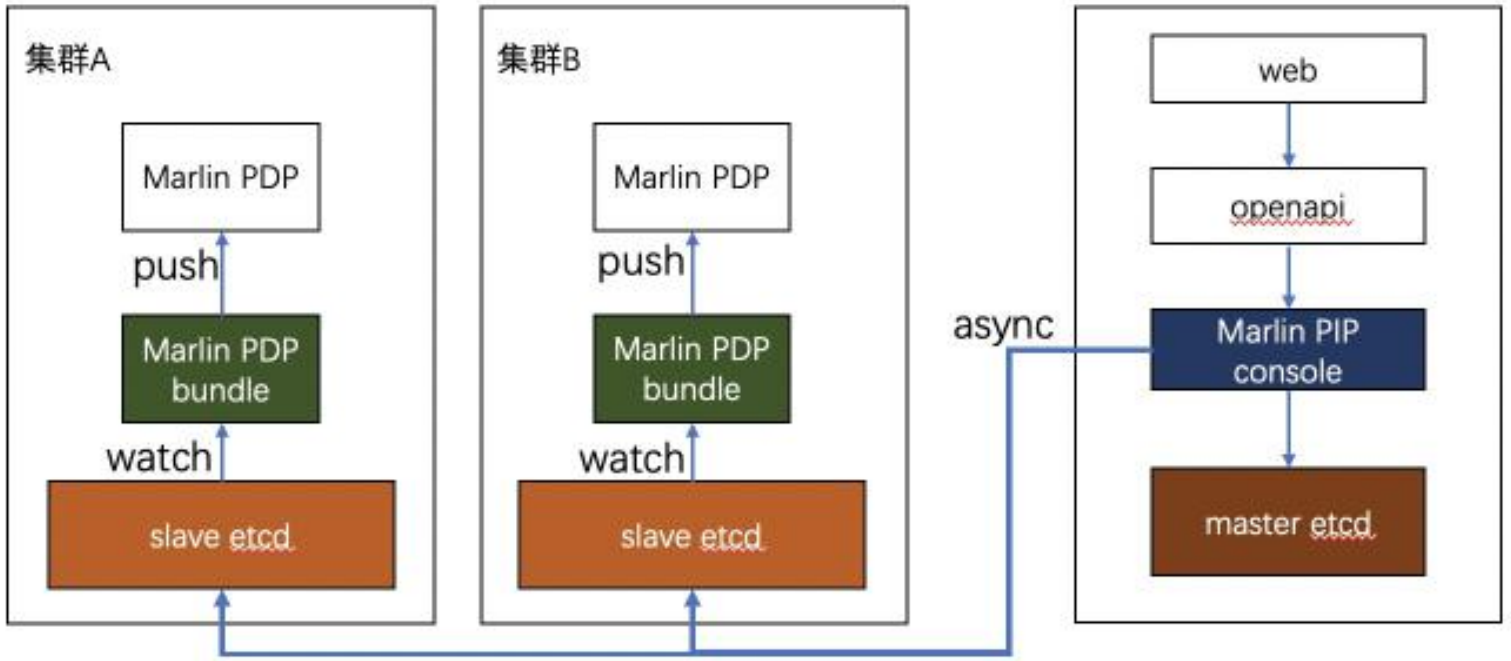
- 多个独立的etcd集群
- 业务多写，写master成功后异步写slave

优点：

- 架构设计简单，易实现

缺点：

- 业务需要大量的开发，保障主从的一致性



应用实践 —— Identity

方案二

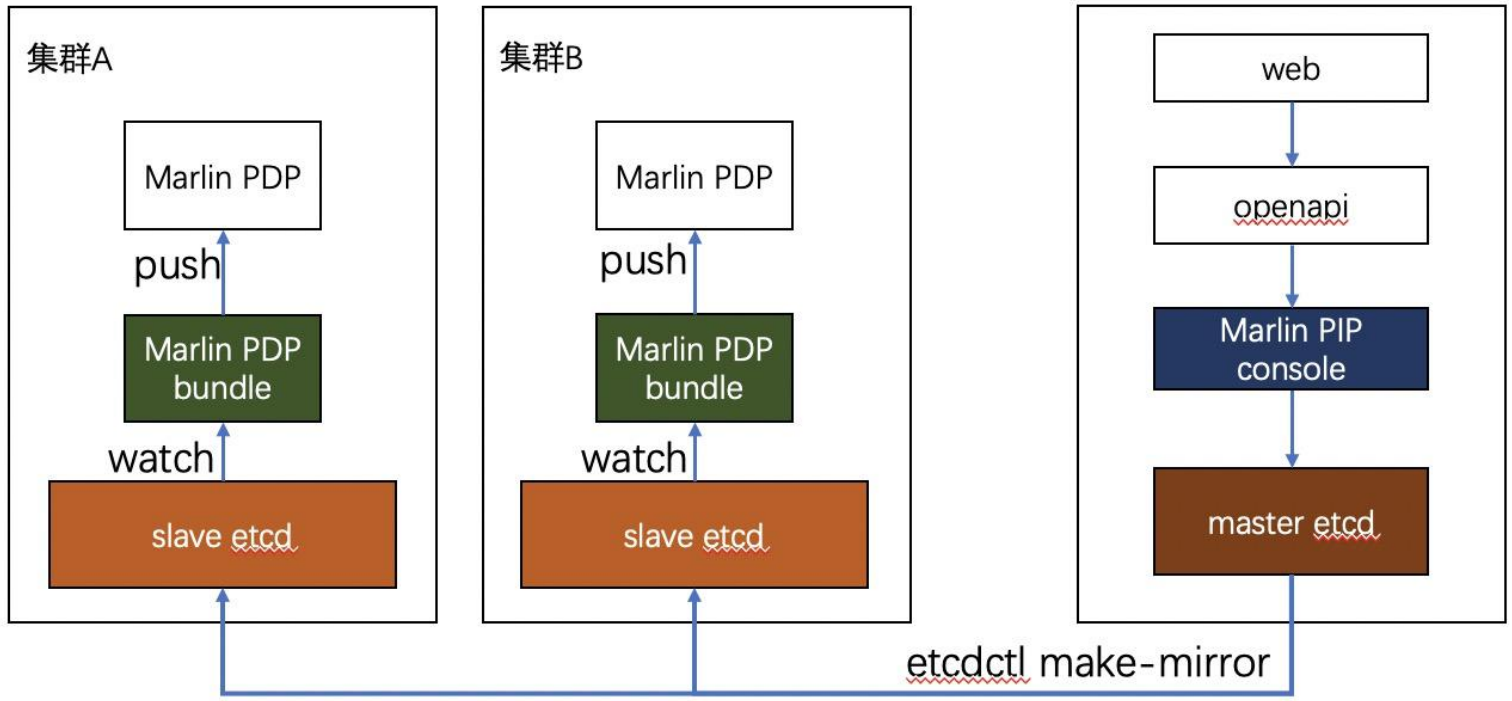
- 利用make-mirror复制数据

优点：

- 架构设计简单，易实现
- 原生支持有保障

缺点：

- 不支持断点续传
- 没有高可用保障



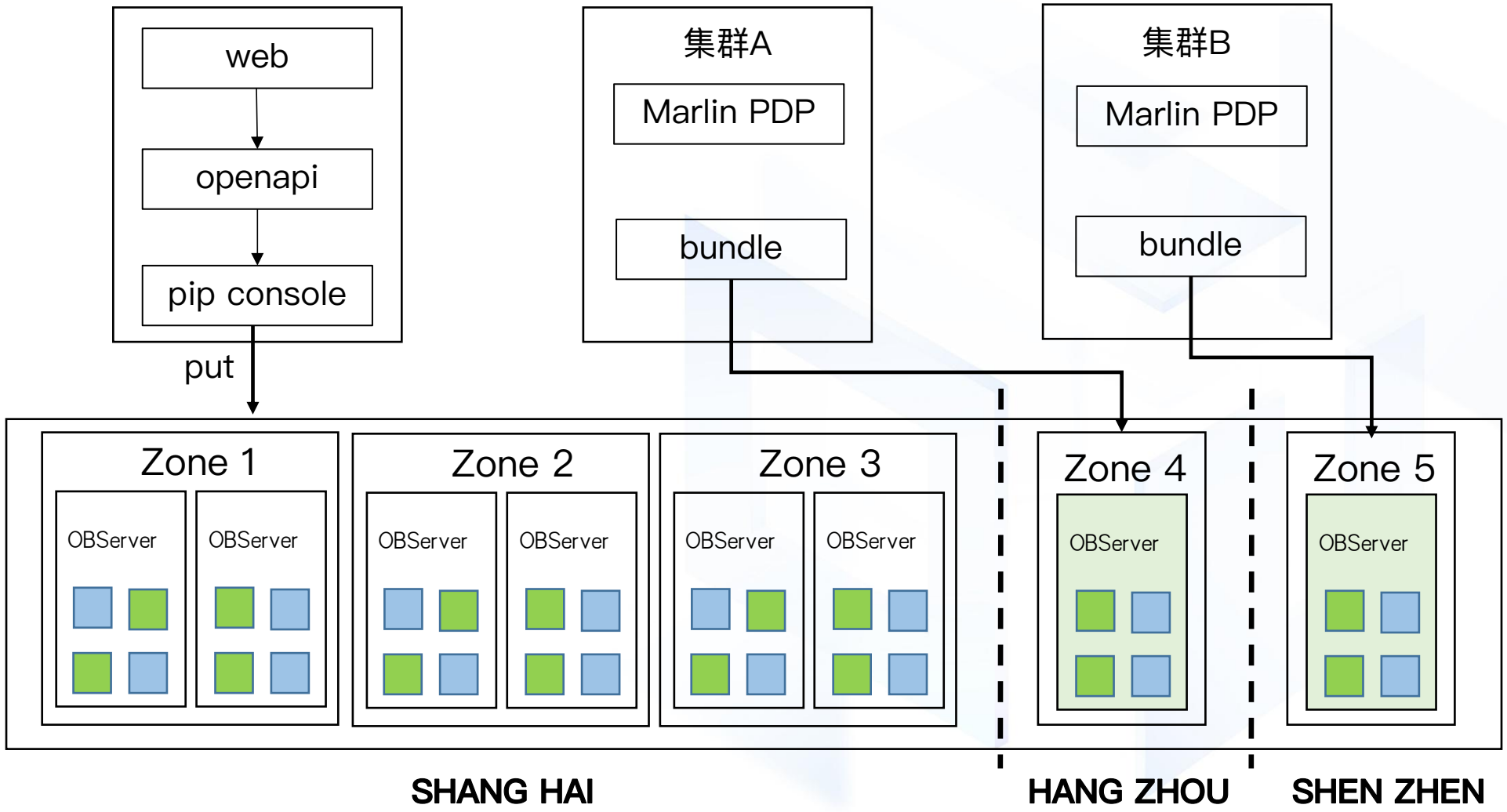
应用实践 —— Identity

方案三

- etcd on OB

优点：

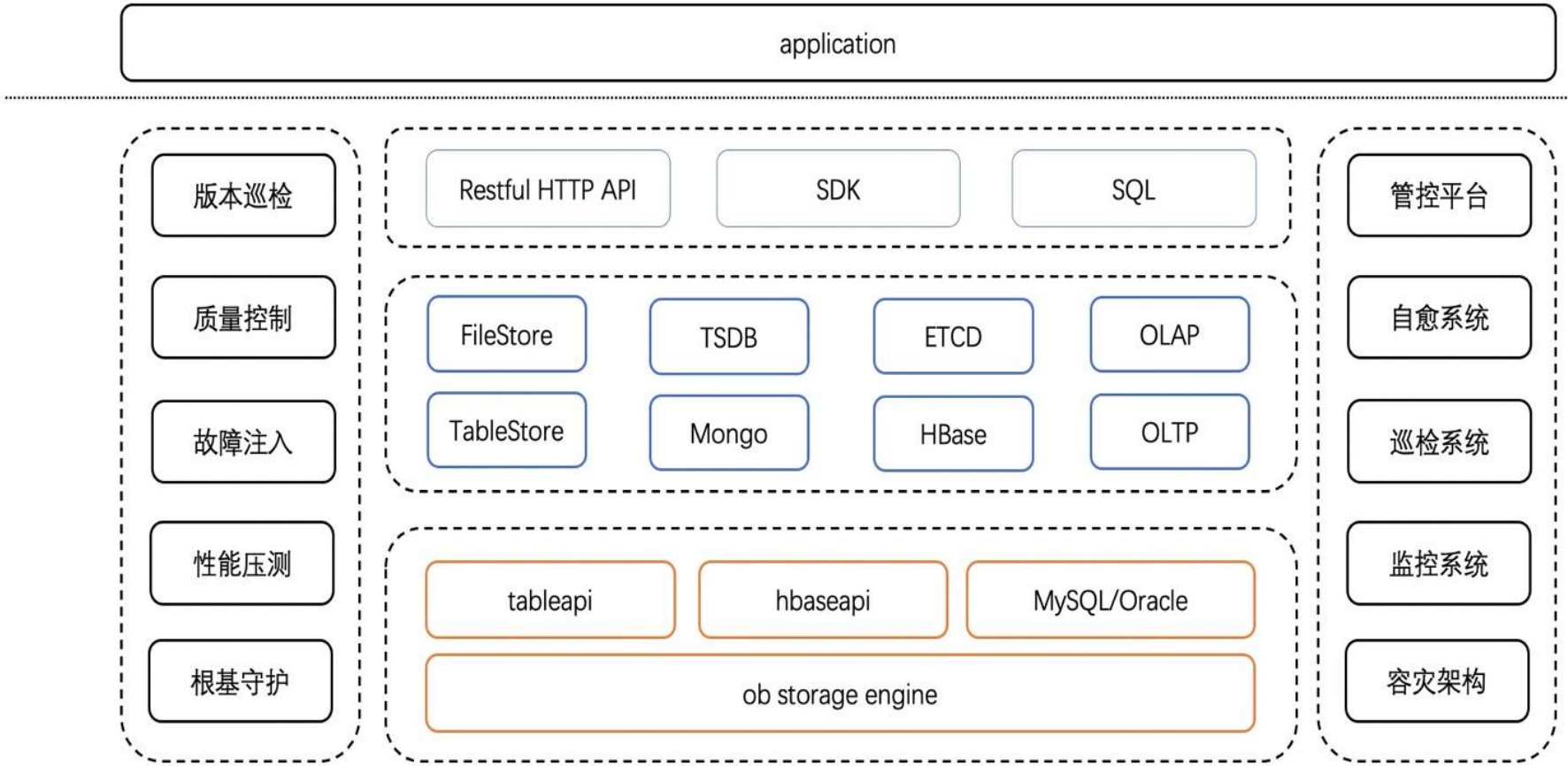
- 架构易实现
- 用户改动少
- 异地只读副本
- 完备的高可用



目　录

未来规划

- 支持OBKV，预计提速 10% – 20%
- 数据迁移方案，支持社区版 etcd 和 etcd on OB 的数据互相迁移



引用参考

[1] https://github.com/etcd-io/bbolt#read-write-transactions

[2] LMDB概述

[3] etcd操作boltdb的优化实现

[4] 蚂蚁集团万级规模 K8s 集群 etcd 高可用建设之路

[5] etcd 在超大规模数据场景下的性能优化

项目地址：

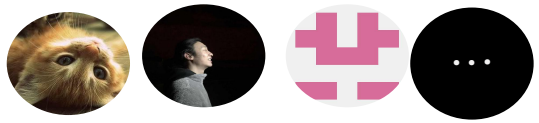
Gitee ： https://gitee.com/oceanbase

Github： https://github.com/oceanbase

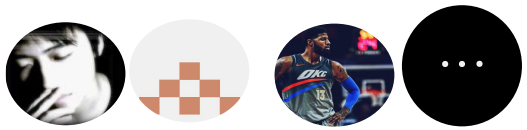
扫码了解
OceanBase

加入OB
开源社群

Supports:



@Alex wang ,@ chen~,@ liying1029,and 3.3K others starred this repository.



@xSky,@ 1008610010,@ lxiuwenL,and 741 others forked this repository.

SOFASTACK™

Thank You

宣 超（锡林）

蚂蚁数据库技术