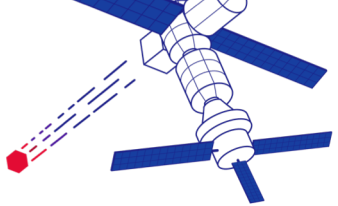
TiDB TechDay 2019

TiKV 在京东云对象存储元数据管理实践

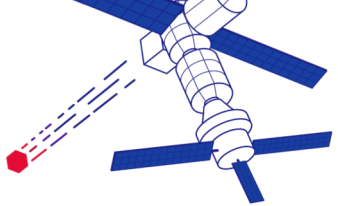
崔灿@京东云

关于我



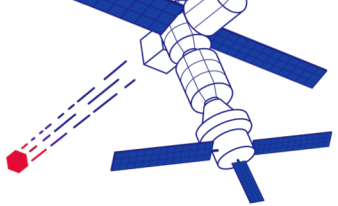
- 崔灿
- 京东云专家架构师，京东云对象存储负责人
- 阿里 -> 腾讯 -> 百度 -> 京东
- cuican3@jd.com

京东云对象存储

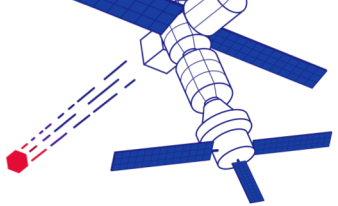


- Since 2016
- 可靠 / 安全 / 海量 / 低成本
- 业务场景
  - 京东商城视频 / 图片
  - 京东公有云外部开发者
  - 政府 / 企业 私有云
  - 混合云

Agenda

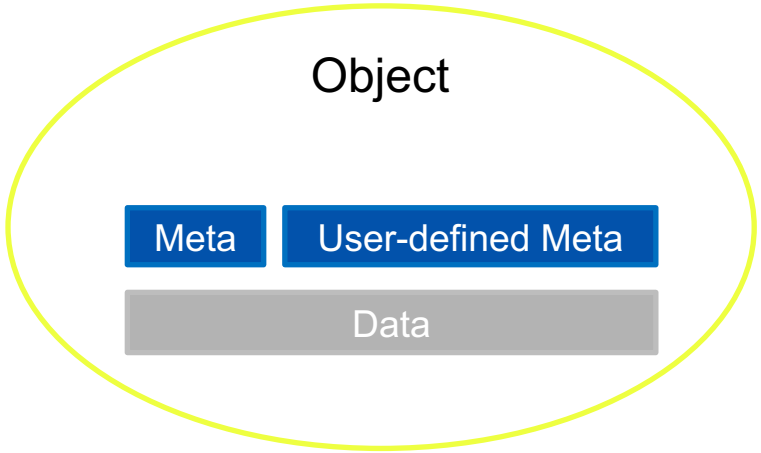


- 对象存储简介
- 对象存储元数据管理演进
- 基于 TiKV 的元数据管理系统
- 业务迁移
- 后续



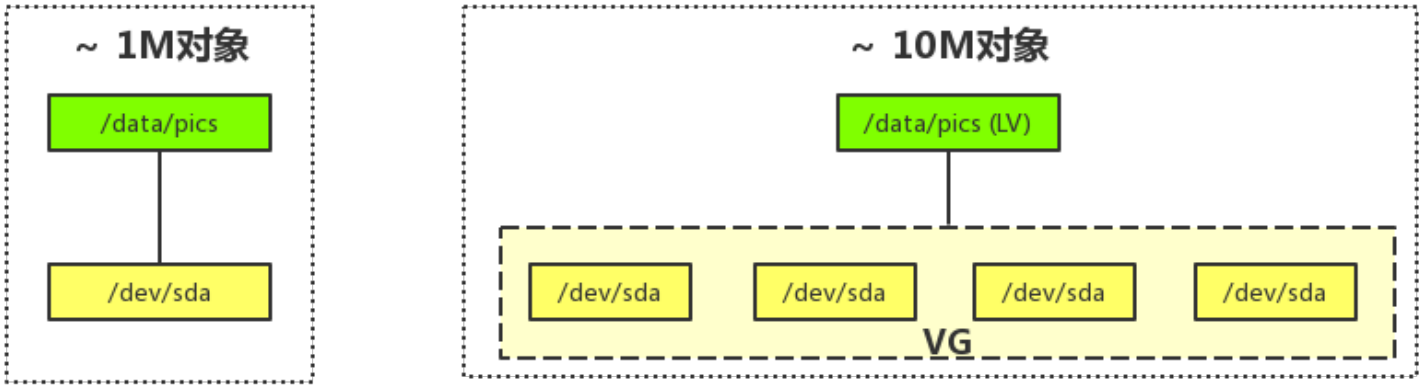
对象存储

什么是对象



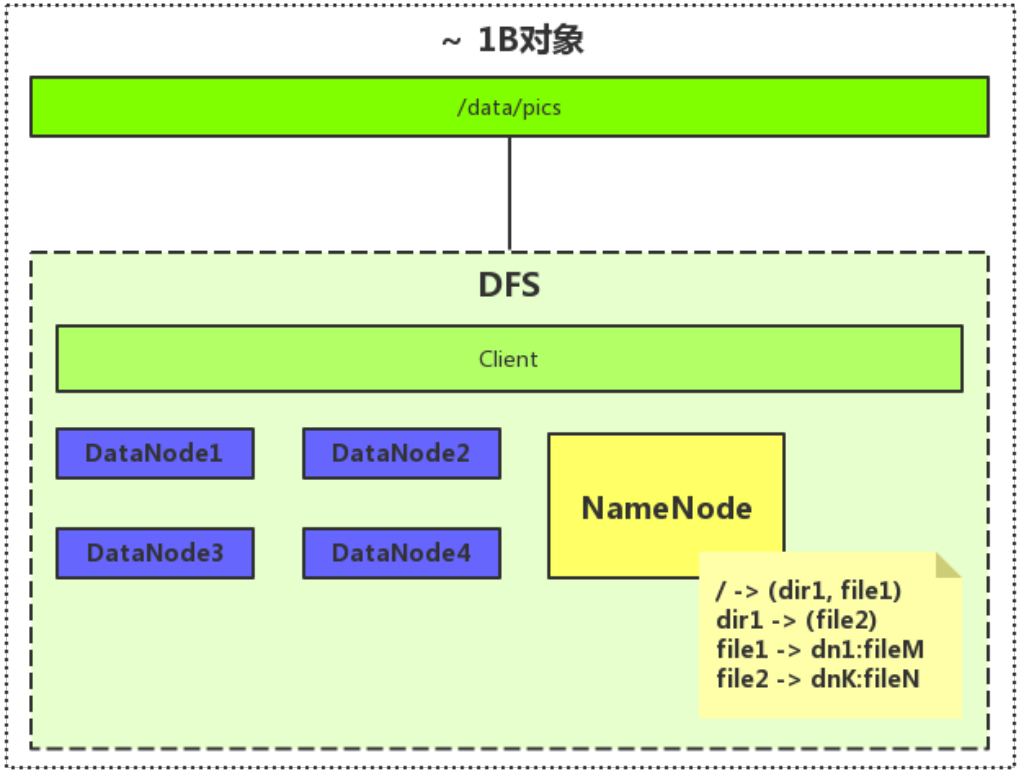
- Object ： 一张照片
- Data：照片的二进制数据
- Meta
  - Length
  - LastModify
  - ……
- User-defined Meta
  - 拍摄者
  - 拍摄设备
  - ……

怎么存储对象



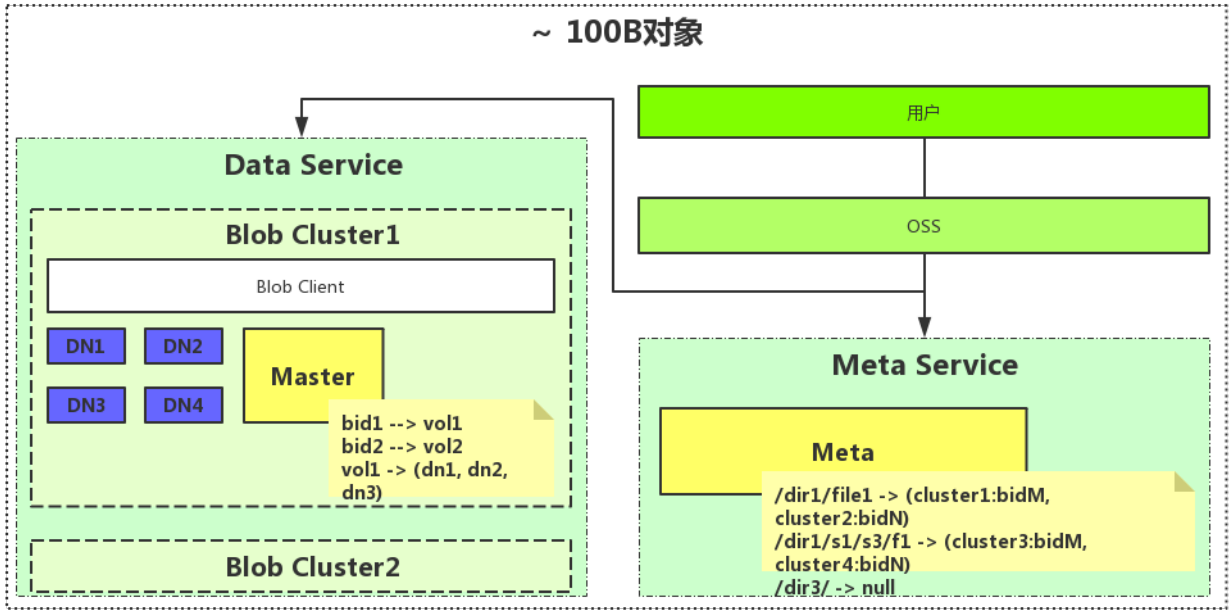
- 数据存储在本地磁盘，文件系统管理元数据
- Object数量 < 1000W

怎么存储对象

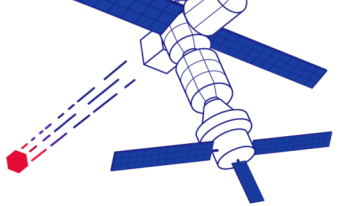


- 数据分布式存储
- 独立NameNode管理元数据
  - 树状目录结构
  - 拆分困难
  - 依赖单机内存
- Object数量 < 10亿

怎么存储对象

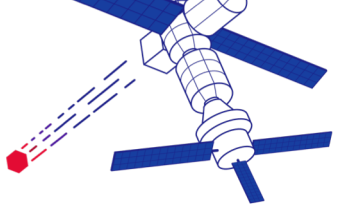


- 元数据平坦存储，分布式存储
- 数据多集群存储



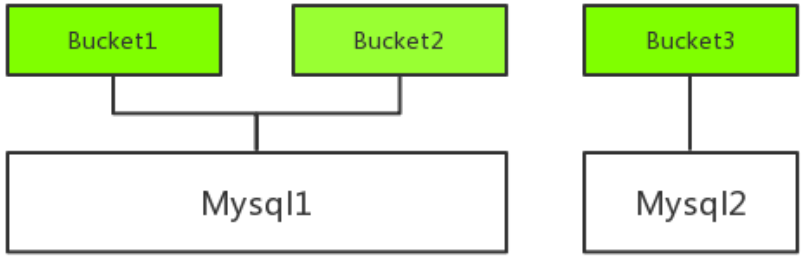
对象存储元数据管理系统

核心需求



- Get(bucketId, objectKey) –> meta
- Put(bucketId, bbjectKey, meta)
- Delete(bucketId, objectKey)
- Scan(bucketId, startObjectKey, endObjectKey, limit)

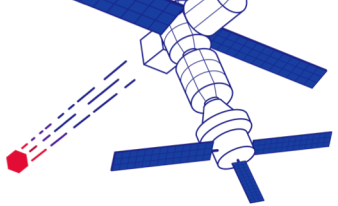
元数据管理系统 — v0.1




```
CREATE TABLE `objects` (
  `bucket` mediumint(9) NOT NULL,
  `object_key` varchar(1023) CHARACTER SET utf8 COLLATE utf8_bin NOT NULL,
  `last_modified` datetime NOT NULL,
  `etag` char(32) NOT NULL,
  `md5` char(32) DEFAULT NULL,
  `length` bigint(20) NOT NULL,
  `path` binary(64) NOT NULL,
  `user_meta` blob,
  PRIMARY KEY (`bucket`,`object_key`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW_FORMAT=DYNAMIC
```

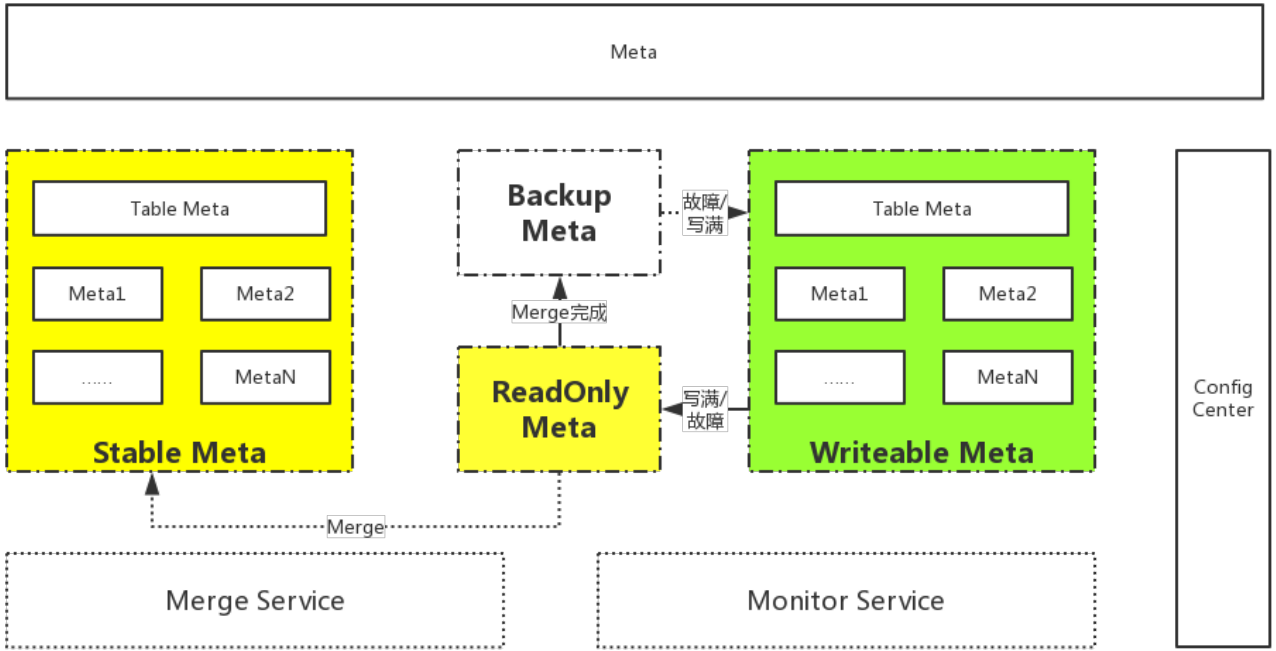
- Put: insert into objects values(1, "1.jpg", ……)
- Get: select * from objects where bucket = 1 and object_key = "1.jpg"
- Scan：select * from objects where bucket = 1 and object_key > "1.jpg" limit 100

挑战

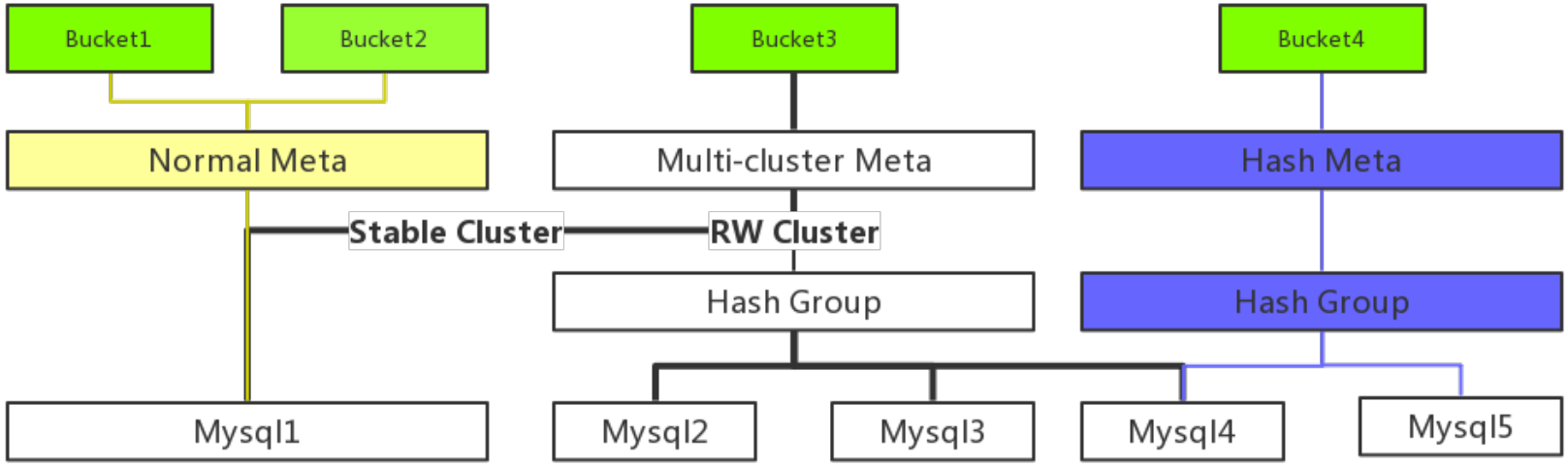


- 单个Bucket预期Object数量巨大
  - 哈希分库分表
  - Scan Merge
- 单个Bucket非预期的变的巨大
  - Rehash
  - Rebalance
- 单个Bucket数据量达到1000亿
  - 全局有序分库分表

元数据管理系统 — v1.0 — 多集群

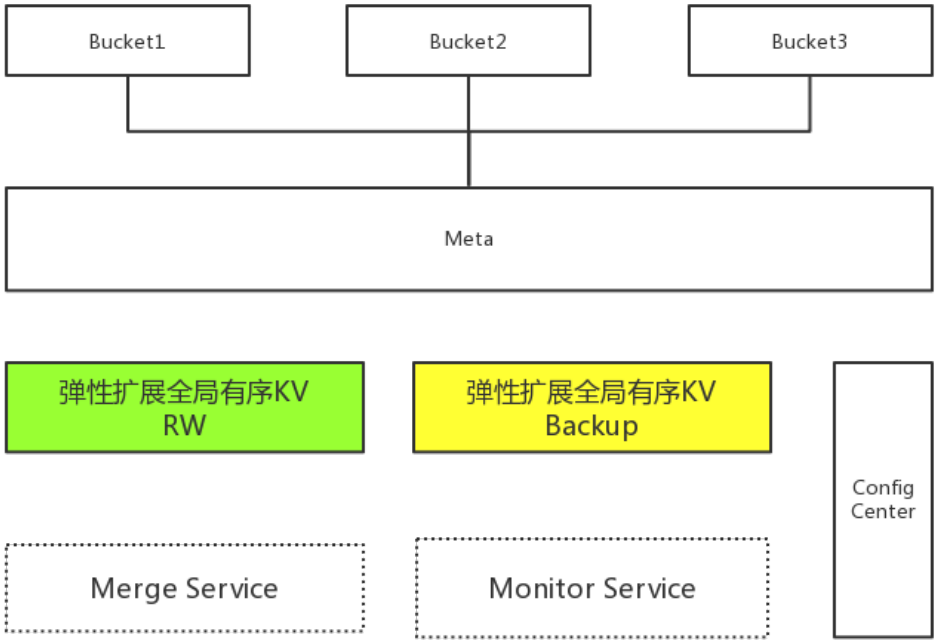


元数据管理系统 — v1.0

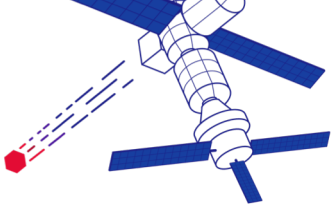


- 数据分布复杂，管理困难
- 调度不灵活

元数据管理系统–目标

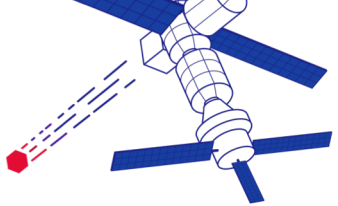


- RW KV
  - 主元数据存储系统
- Backup KV
  - RW KV不可用的时候写入容灾
- Monitor Service
  - 监控主RW KV可用性 & 切换 Backup
- Merge Service
  - Backup KV数据合并回RW KV



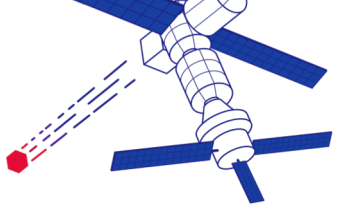
基于 TiKV 的元数据管理系统

Why TiKV



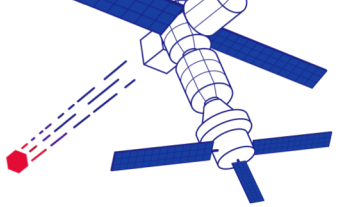
- 功能满足，全局有序KV，可轻松水平扩展
- 性能满足需求
- 社区活跃，文档和工具相对比较完善
- 代码简单

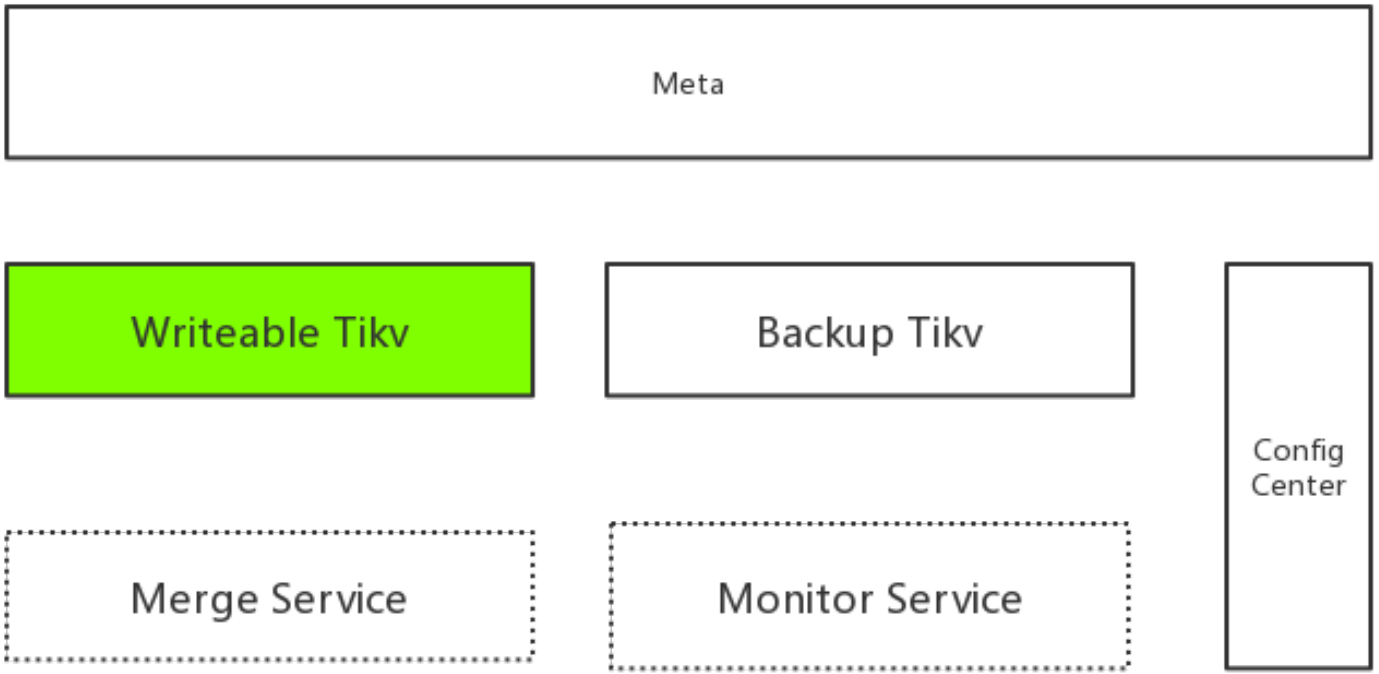
Why TiKV work

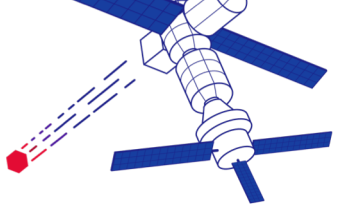


- 功能测试
- 性能测试
  - Qps
  - Latency(Avg, TP90, TP99)
- 异常测试
  - 机器/磁盘/网络 故障
  - 业务异常
- 随机组合异常测试
- 预发布环境验证

元数据管理系统–v2.0

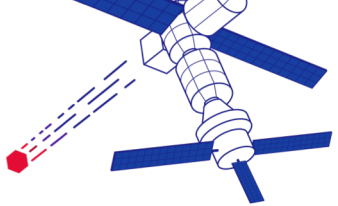



Meta
Writeable Tikv
Backup Tikv
Config Center
Merge Service
Monitor Service



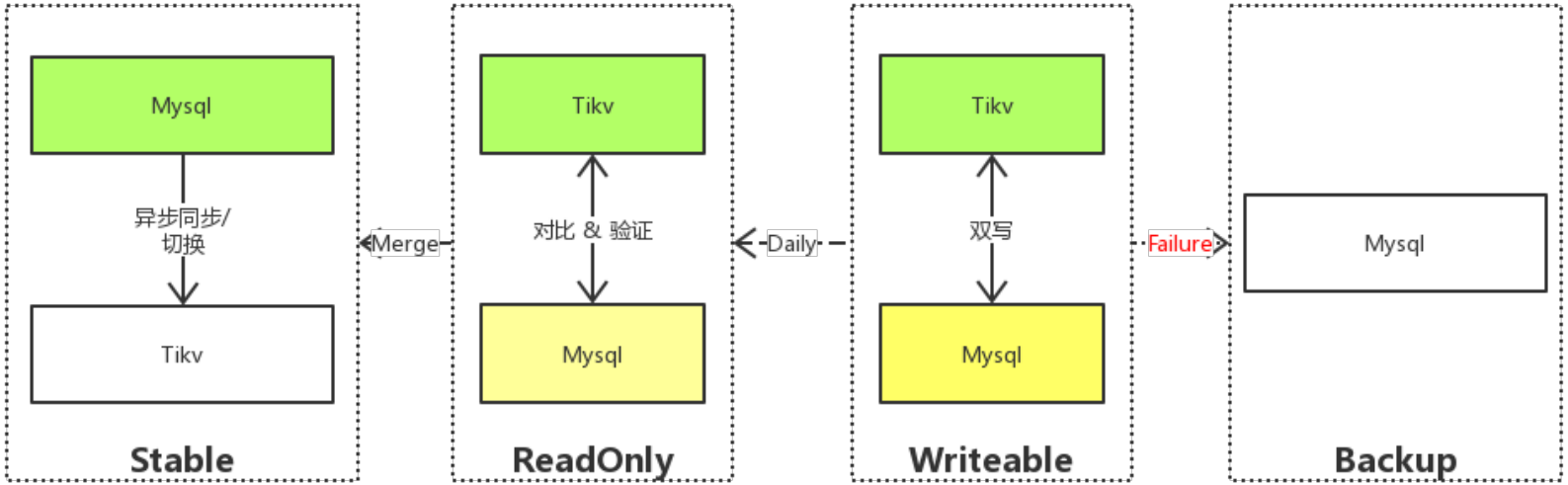
业务迁移

挑战



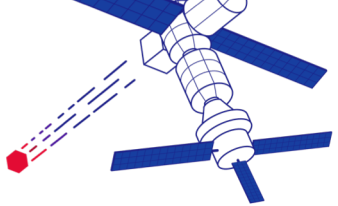
- 切换
- 验证
- 回滚

迁移方案



- 双写保证数据安全
- 数据静态化，方便对比验证
- 存量数据静态化，简化迁移

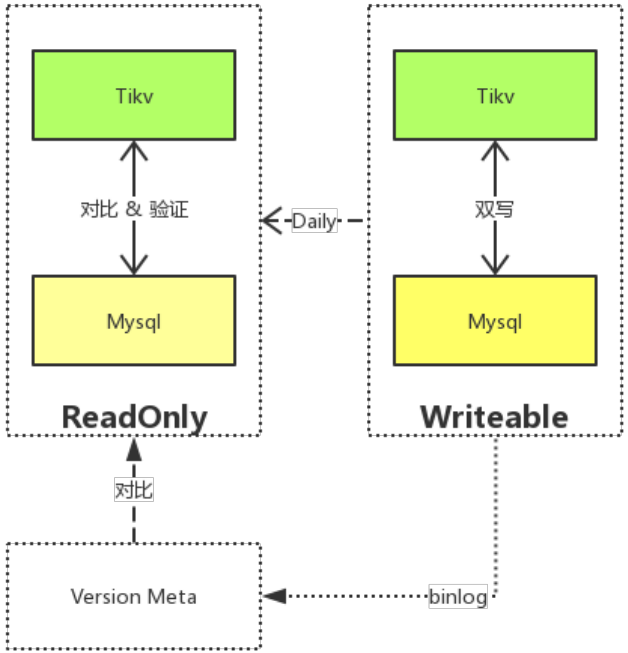
切换



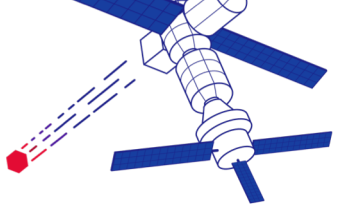
- 存量数据切换
  - 存量数据静态化，简化迁移 / 对比 / 回滚流程
- 增量数据切换
  - 增量数据双写TiKV和Mysql，异常快速回滚
  - TiKV在测试环境验证充分，激进使用TiKV作为双写的Primary
- 切换步骤
  - 存量数据切换到TiKV，验证读
  - 增量数据切换到TiKV，验证读写
  - 直接下线Mysql

验证

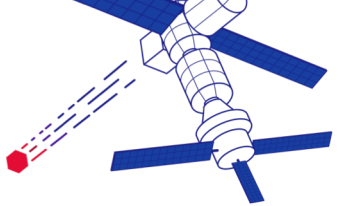
- 存量数据静态化，容易验证
- 增量数据验证
  - 增量数据双写TiKV / Mysql，TiKV写入算成功
  - 操作记录进入Queue，异步进入Version Meta
  - 每天把RW 库静态化
  - 校验静态TiKV和Mysql中数据一致性
  - 不一致的数据，根据Version Meta记录，判断是否合理



现状

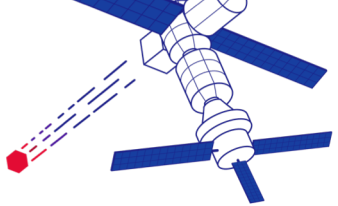


- TiKV作为京东云对象存储元数据的Primary存储
- 10+集群
- 生产环境单集群峰值QPS 4W+(读写1:1)
- 单集群200亿+数据，50W+ Region
- TiKV 3.0验证中，预计Q4上线



后续工作

后续工作



- 灾备
- 集群规模优化
- Region调度策略优化

Thank You !