从百度文件系统看大型分布式系统设计

颜世光
百度 搜索基础架构

自我介绍

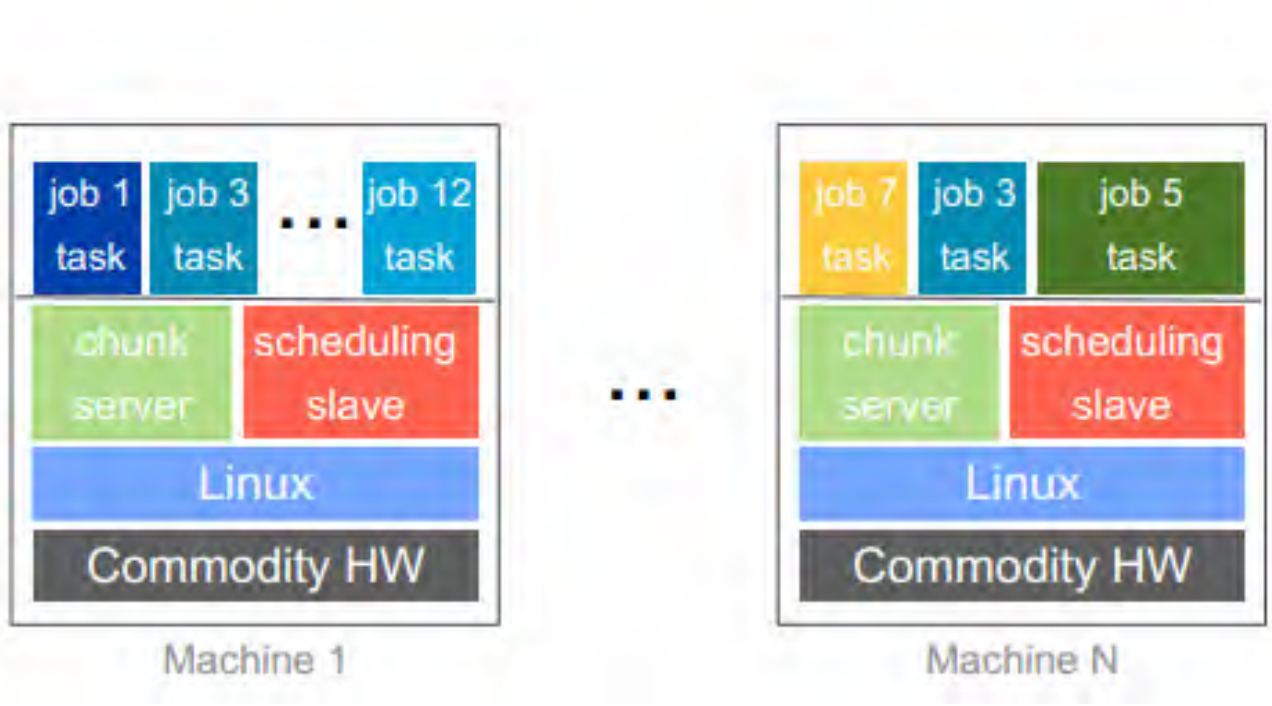
- 颜世光, 专注于大规模分布式系统
- 代表作品
  - 百度第三代Spider系统
  - 百度文件系统BFS
  - 万亿量级实时数据库Tera
  - 集群调度系统Galaxy
- 个人主页&Blog
  - https://github.com/bluebore
  - http://bluebore.cn

提纲

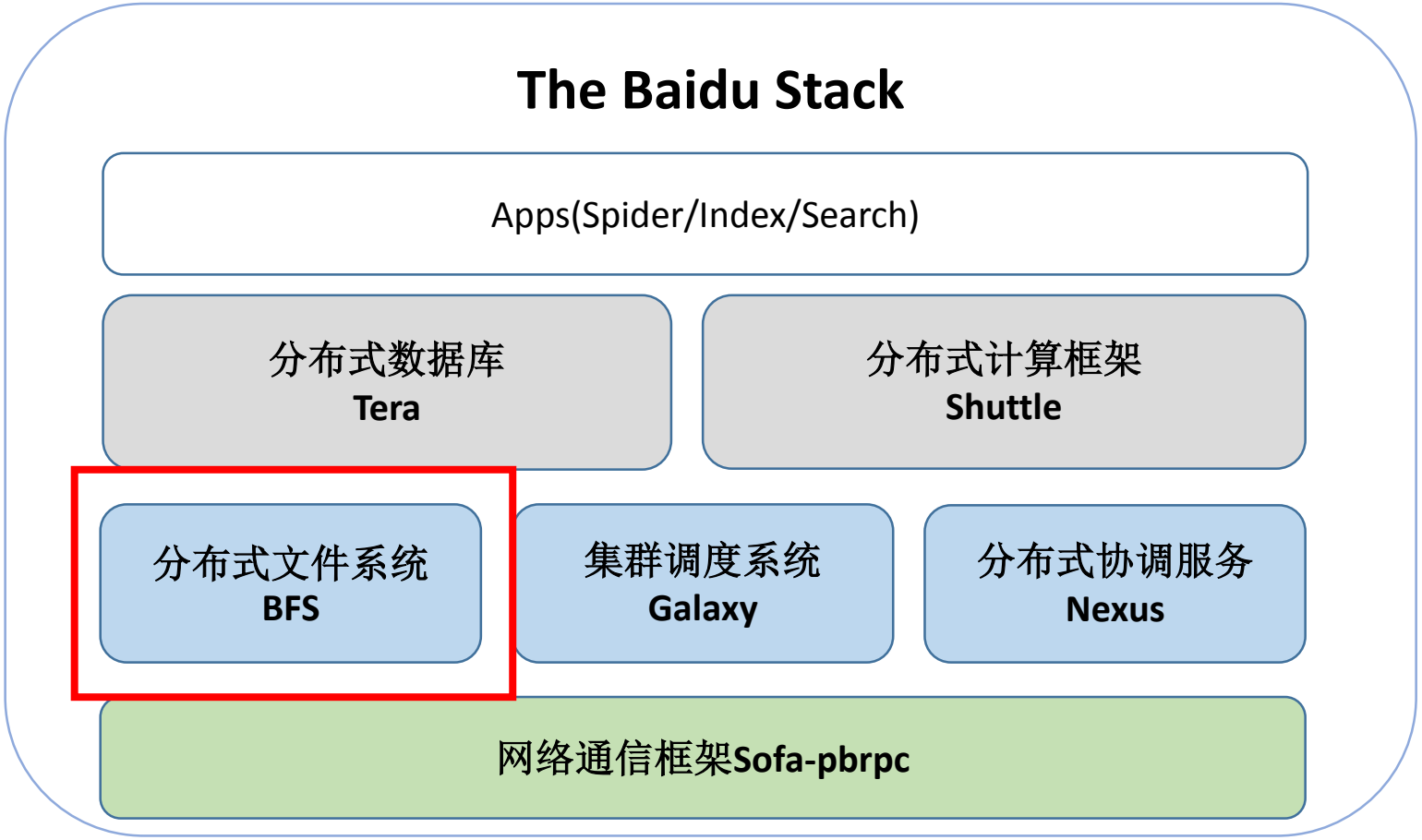
- 百度文件系统简介
- 分布式系统设计实践
- 总结与致谢

百度的集群环境

- 单个集群通常几千台机器
- 百度文件系统(BFS)、集群调度系统（Galaxy）、分布式协调服务（Nexus）是核心服务
- 实时任务与批量任务混合部署



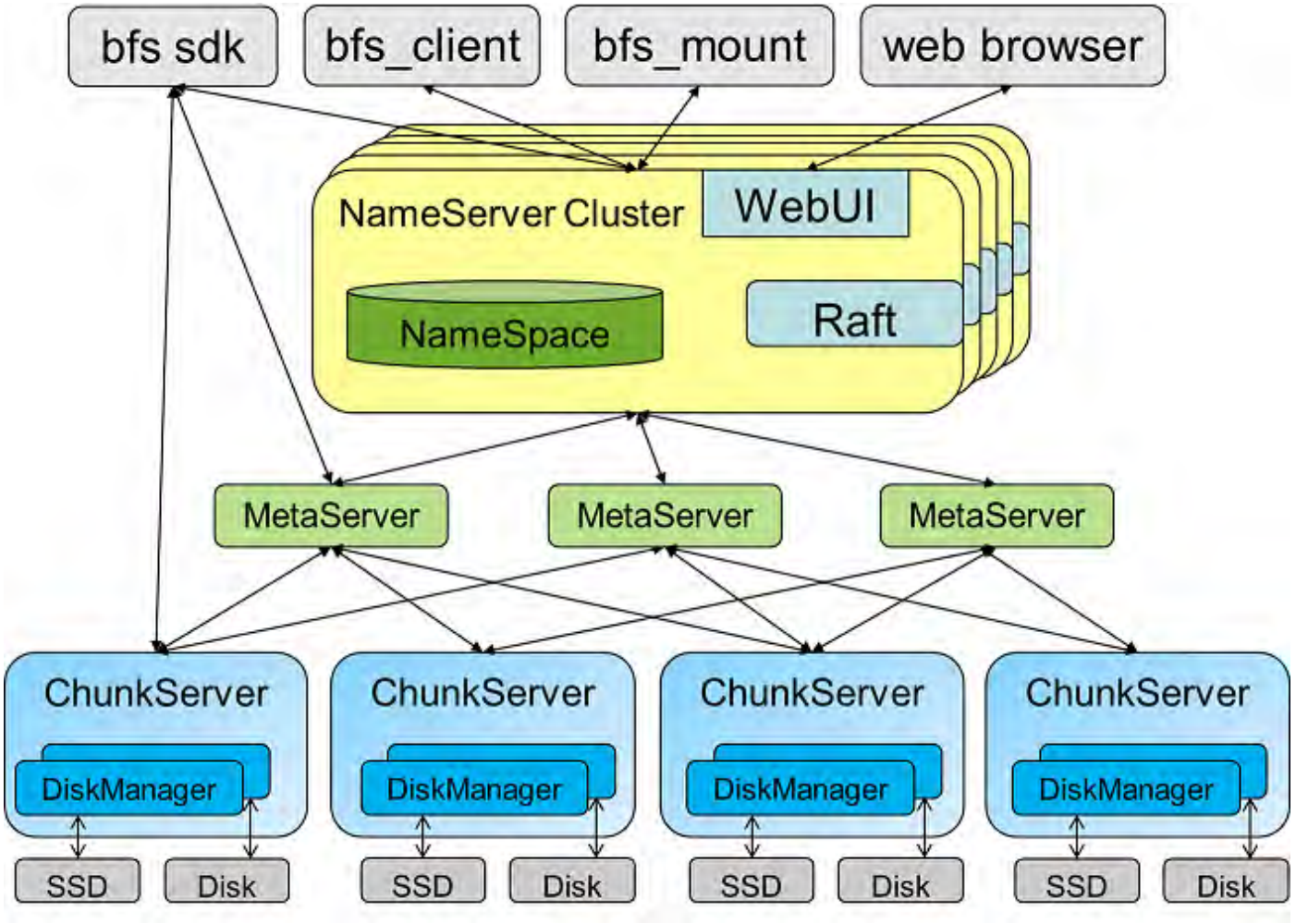
分布式软件栈中的BFS



数据中心操作系统(DCOS)

- 进程调度&内存管理
  - Galaxy
  - 应用部署和任务调度
- 锁和信号量
  - Nexus
  - 分布式锁
  - 分布式通知
- 文件系统
  - The Baidu File System
  - 持久化存储

百度文件系统架构

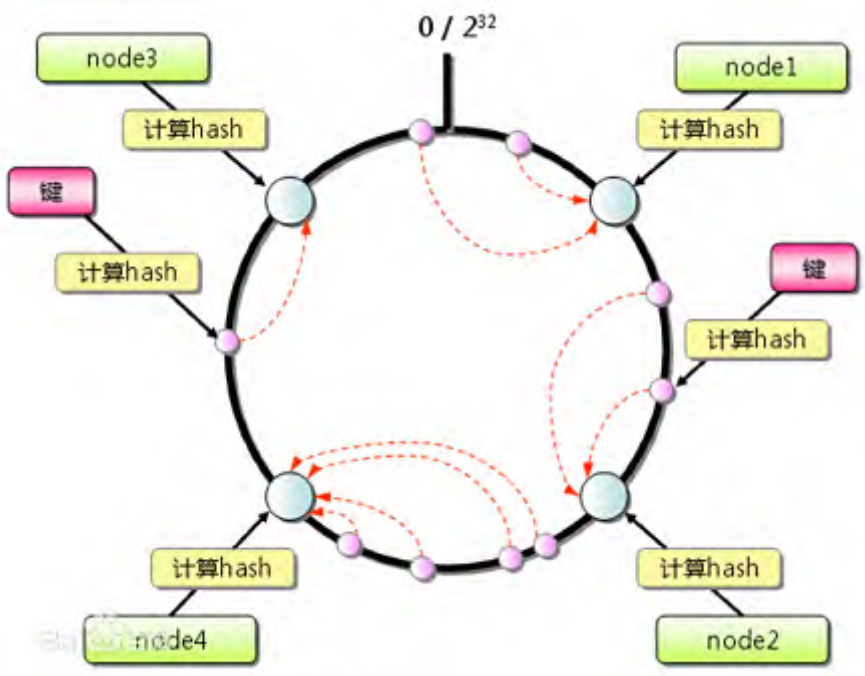


设计一个分布式系统要考虑的

- 数据与计算的分片
- 分区故障容忍
- 数据一致性
- 系统扩展性
- 延迟与吞吐
- 成本与资源利用率
- …

数据与计算的分片

- 哈希分片
  - 简单、均衡
  - 扩容复杂、易用性差
  - 一致性哈希、虚拟节点
- 按范围、数据量分
  - 使用简单
  - 需要管理元数据
  - 中心化与去中心化



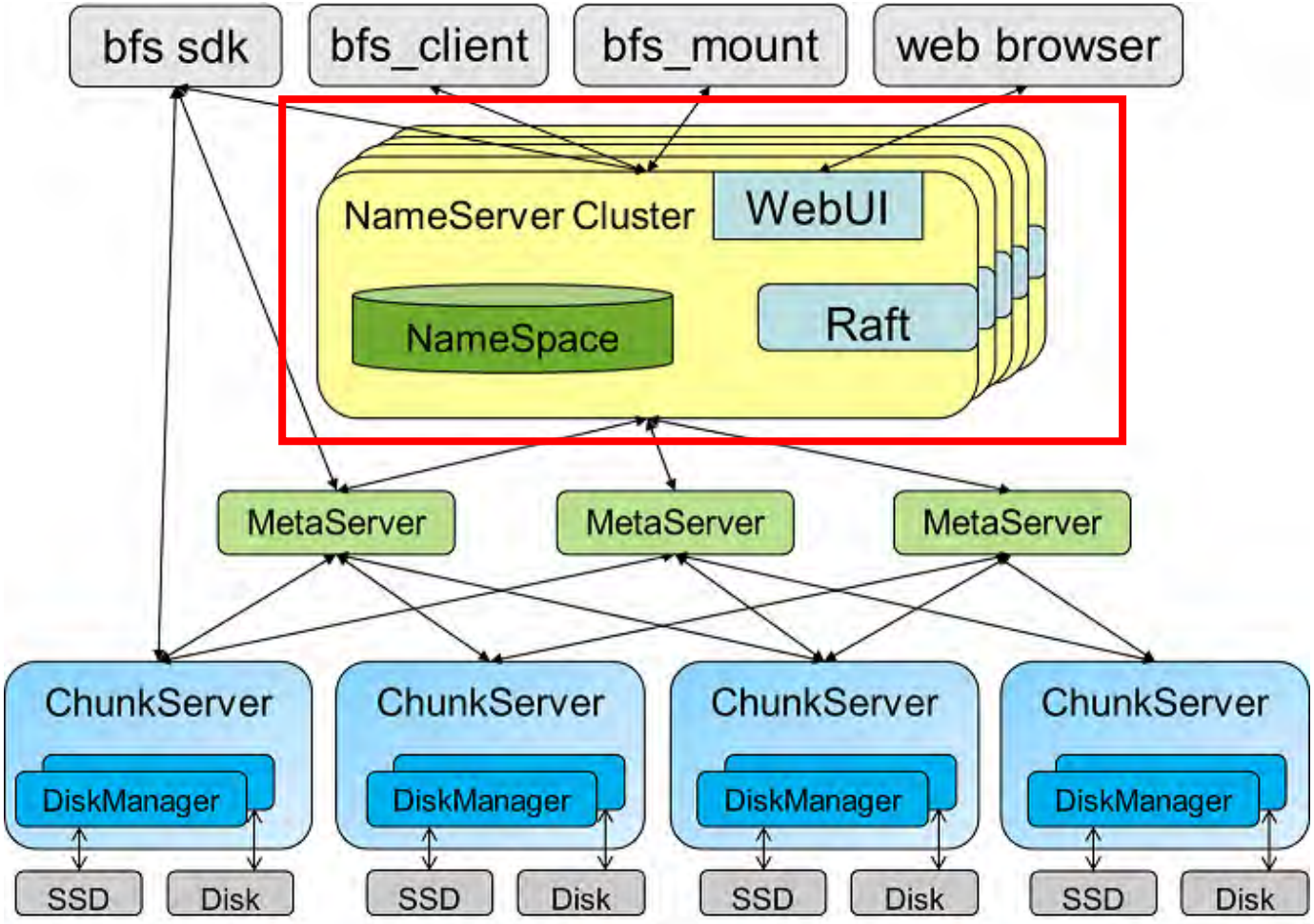
元数据管理

- 去中心化
  - P2P技术
  - 潜在的一致性问题
  - 能管理的元数据有限
- 中心化
  - 设计实现简单
  - Master节点易成为瓶颈

中心化的解决方案

- Master-Slave模型
  - Master是管理者
  - Slave是执行者
- 解决Master节点瓶颈
  - 常规操作不经过Master
    - 一般计算系统
    - Bigtable、Tera等存储系统
  - 使Master无状态
    - 非最底层系统都可以设计无状态Master
  - Master分布化
    - BFS选择的解决方案

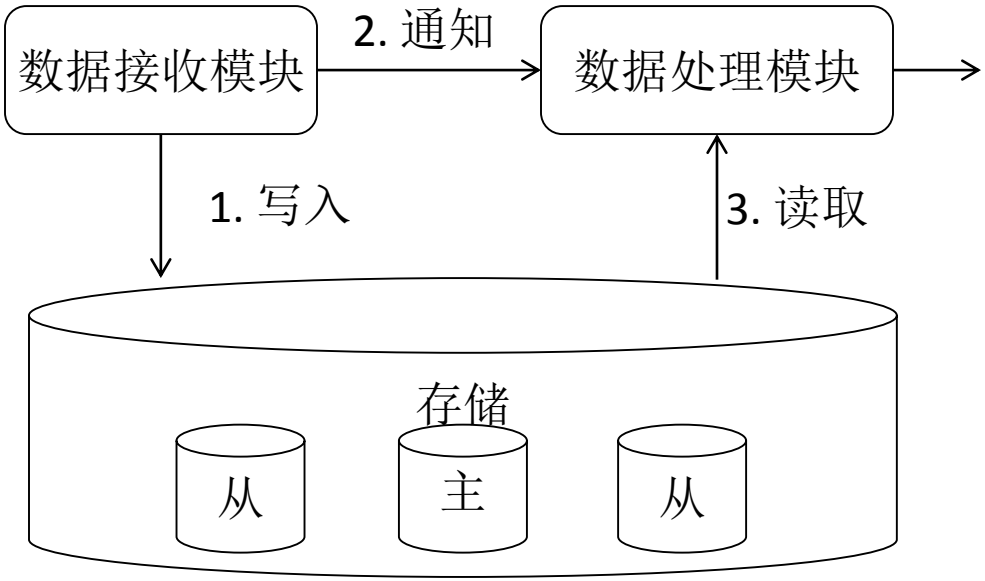
NameServer Cluster



故障容忍

- 设备都是会坏的
  - 假设的服务器MTBF是30年
  - 搭建一个1万台服务器的系统
  - 每一两天就坏一个
- 典型数据中心
  - 过热：5分钟内数千台机器宕机
  - 供电异常： 500~1000台机器突然消失
  - 机架晃动： 几十台机器出现50%丢包
  - 交换机故障： 几十台机器突然消失
  - 磁盘、单机故障
- 通过冗余应对故障

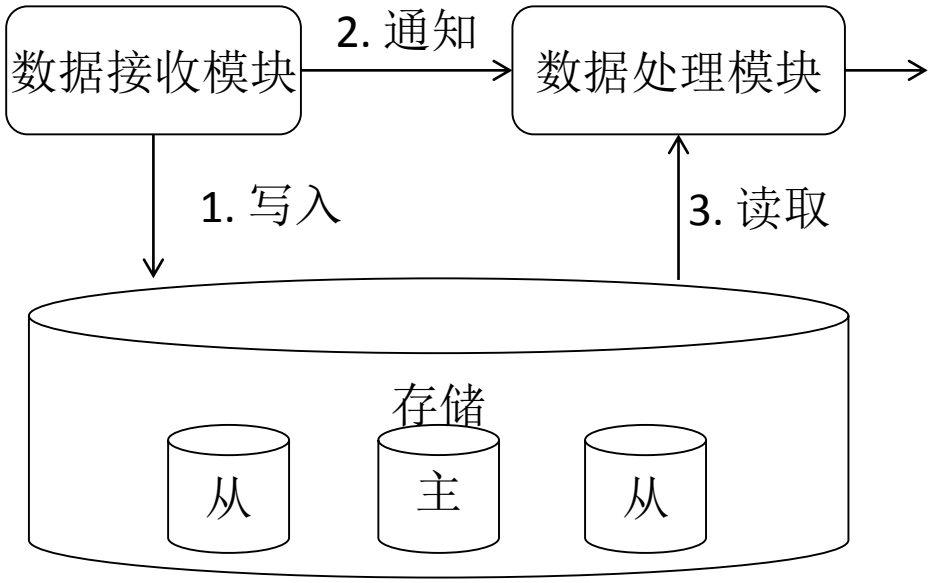
一个典型的数据处理场景



多副本冗余

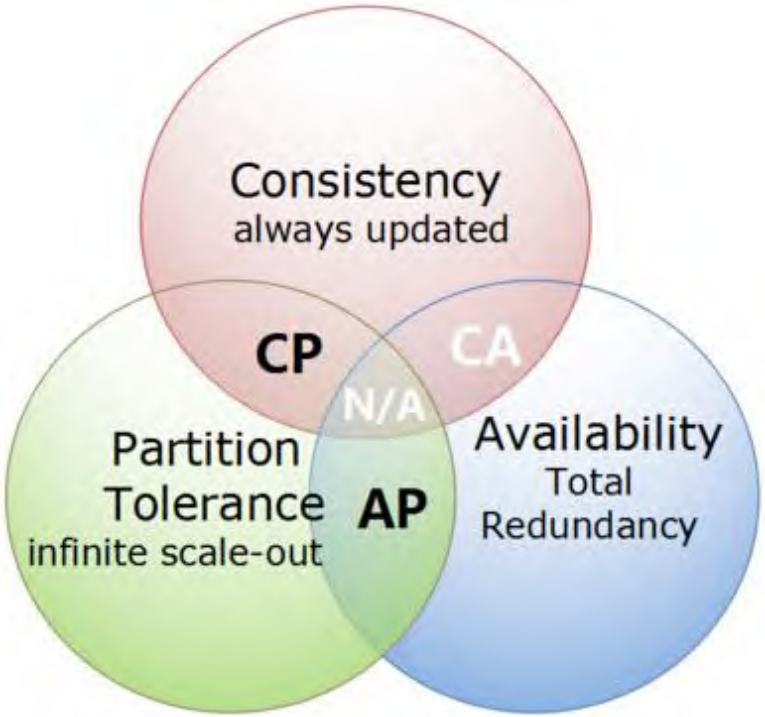
一致性问题

- 怎么定义写成功
  - 3副本成功, 影响可用性
- 可以读从节点
  - 刚写入的读不到
  - 不一致
- 只允许读主节点
  - 扩展性受限



CAP理论

- 简要历史
  - Eric Brewer 1998年提出
  - 2002年证明
- CAP三选二
  - Consistency 一致性
  - Availability 可用性
  - Partition Tolerance 分区容忍性
- 分布式系统
  - 容忍网络隔离是必须的
  - CP、AP



Quorum机制

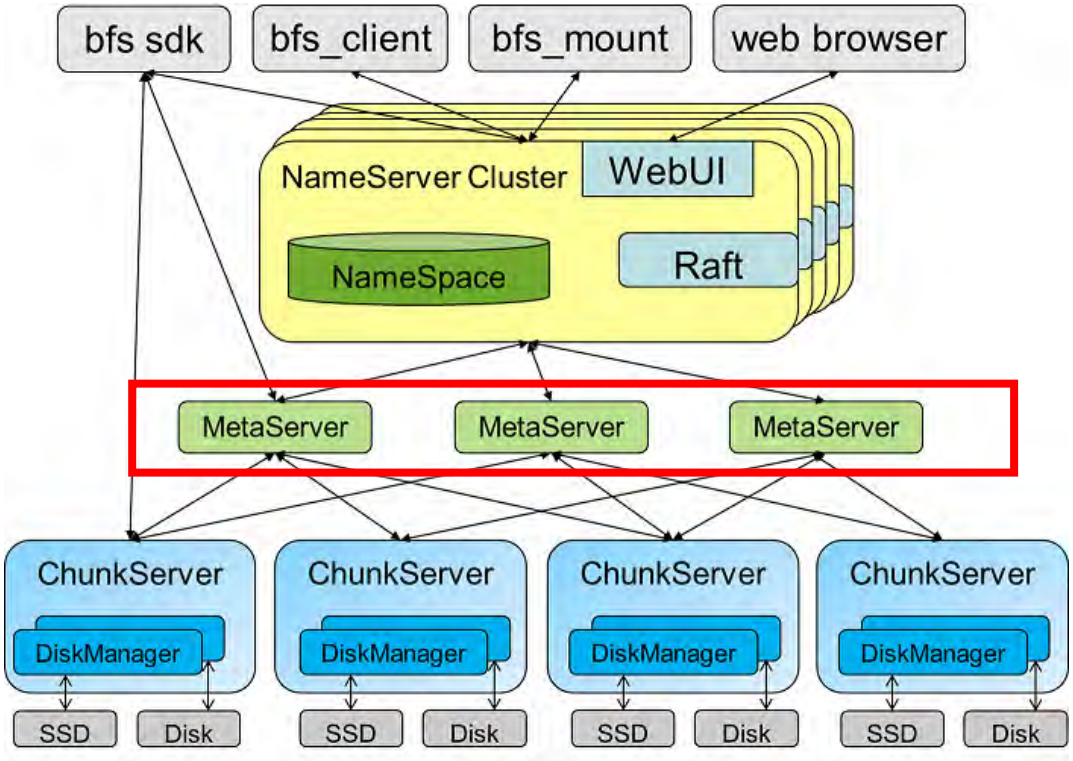
- Quorum写（NWR）
  - Write 写成功W副本
  - Read 读R副本
  - W+R>N，就不会丢失更新
- 一致性协议
  - Paxos
  - Raft

CAP到CAD的演变

- 必须容忍网络隔离
  - CAP->CA
- 跨地域的延迟
  - CA->CAD/CAL
- 多数情况下我们更重视可用性
  - CAD->CD
- 一致性与延迟的折衷
  - 要求强一致的，容忍延迟
  - 要求低延迟的，选择最终一致

提升系统扩展性

- 架构的可扩展性
  - 拆分元数据节点
  - 引入MetaServer



提升系统扩展性

- 设计的可扩展性
  - 保证在规模扩大5倍或10倍是正常工作
- BFS避免了过渡设计
  - 用设计中的不可扩展达到最大的可扩展
    - 最多支持6万台机器
    - 最多支持100亿文件

分布式存储系统设计的特殊性

- 最基础服务的提供者
  - 不可能做成无状态的
- 最底层的仲裁者
  - 不能依赖ZooKeeper等系统选主
  - 分布式的双主问题只从存储系统解决

这些设计给BFS带来哪些优势?

	HDFS	BFS
名字节点 扩展方式	联邦式 分裂的目录树	分布式 统一的目录树
宕机恢复时间	分钟级	秒级
外部依赖	ZooKeeper & QJM	无
开发语言	Java	C++

Q&A

- 欢迎参与BFS的开发
- https://github.com/baidu/bfs
- 谢谢~