上海 云数智聚 砥柱笃行

CHINA APACHE HADOOP MEETUP 2022

2022年09月24日 9:00

上海古井假日酒店4楼

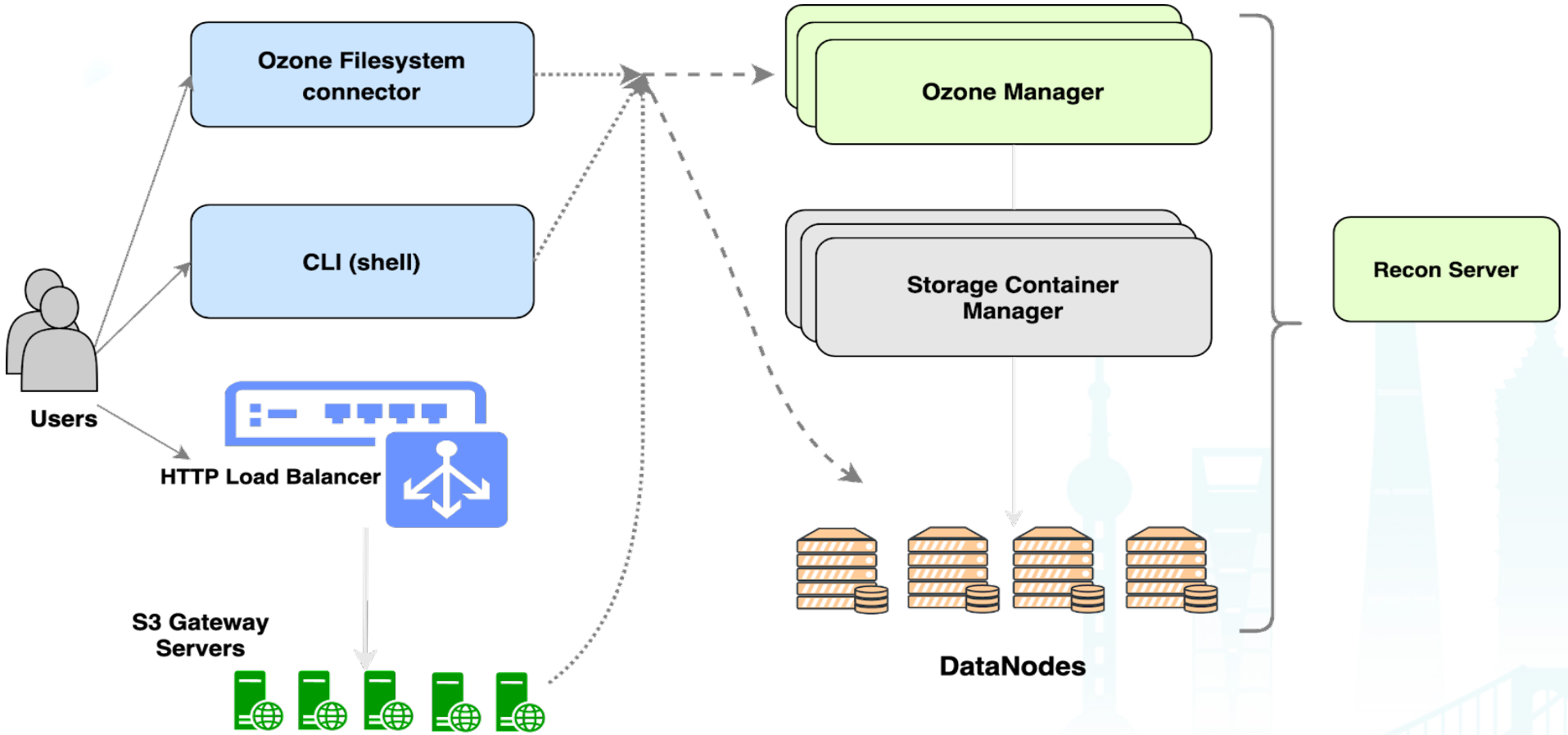
What’s new in Apache Ozone 1.3

陈怡

Apache Ozone PMC 主席

目录

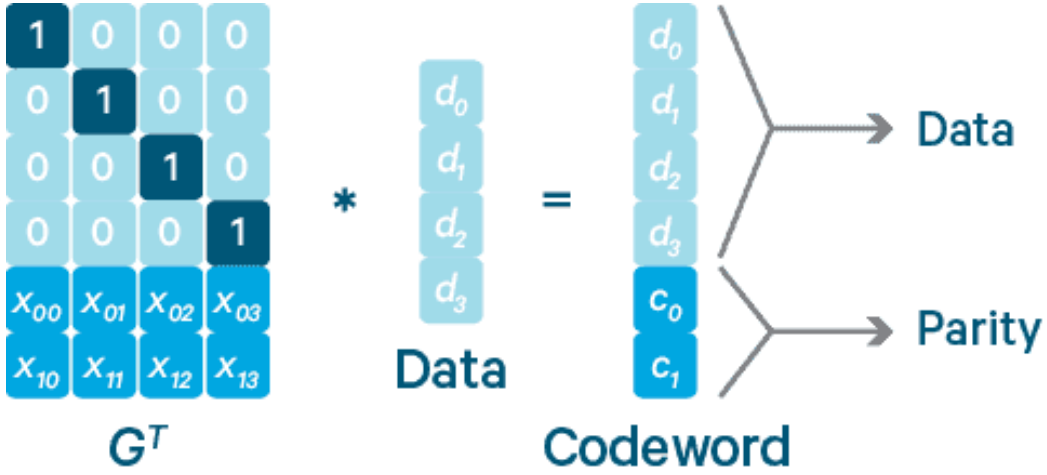
Ozone 构架


Ozone Filesystem connector
CLI (shell)
Users
HTTP Load Balancer
S3 Gateway Servers
Ozone Manager
Storage Container Manager
Recon Server
DataNodes

Ozone 1.3 新功能

I. 纠删码(Erasure coding)
II. 系统均衡器(Container Balancer)
III. 性能优化 - 文件系统优化(File System Optimization)
IV. 性能优化 - 合并Container RocksDB实例
V. 很多其他的性能和稳定性优化

纠删码



	数据可靠性 (越高越好)	存储效率 (越高越好)
1-replica	0	100%
3-replica	2	33%
EC RS(6,3)	3	67%
EC RS(10, 4)	4	71%
EC RS(3,2)	2	60%

以计算为代价，满足数据可靠性的同时，降低数据存储成本

数据可靠性 vs. 存储效率

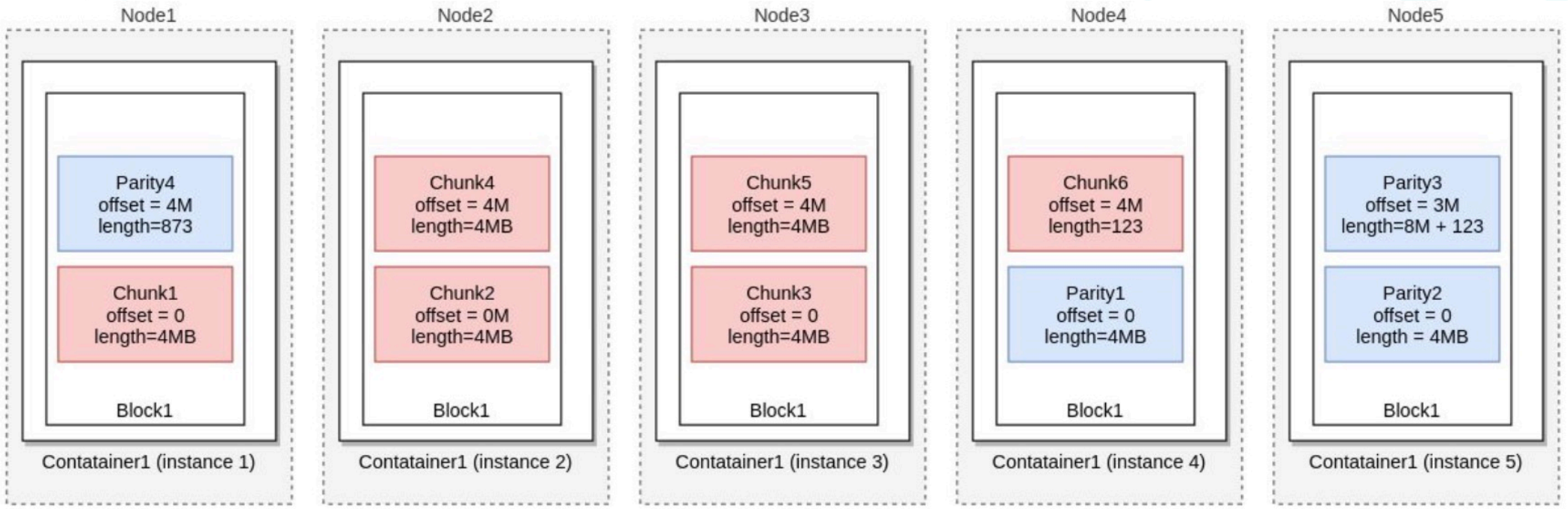
Ozone条带纠删码

I. 物理块：每个DN磁盘上的数据块，默认256MB

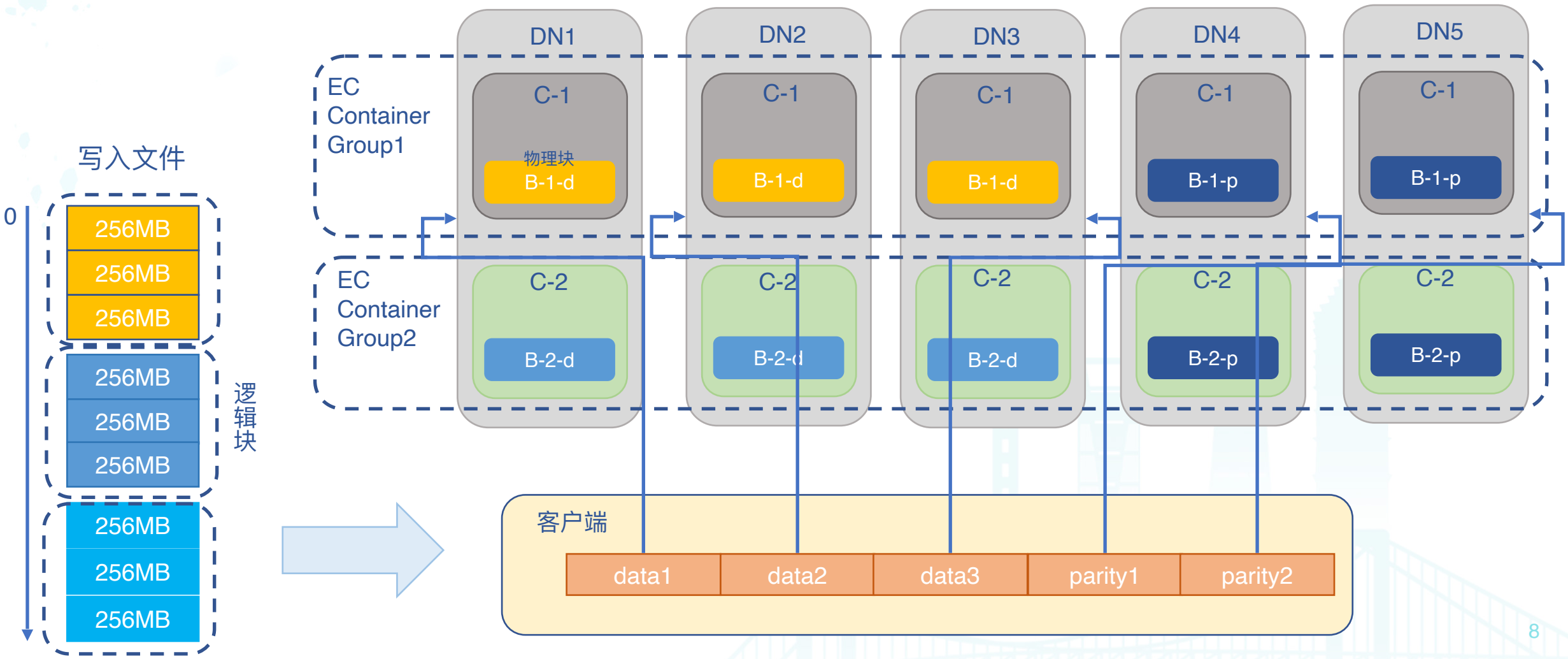
II. 逻辑EC块：满足EC策略的一个用户数据块。例如RS-3-2，一个逻辑块3*256MB大小

III. 条带：条带的默认粒度1MB，可配置

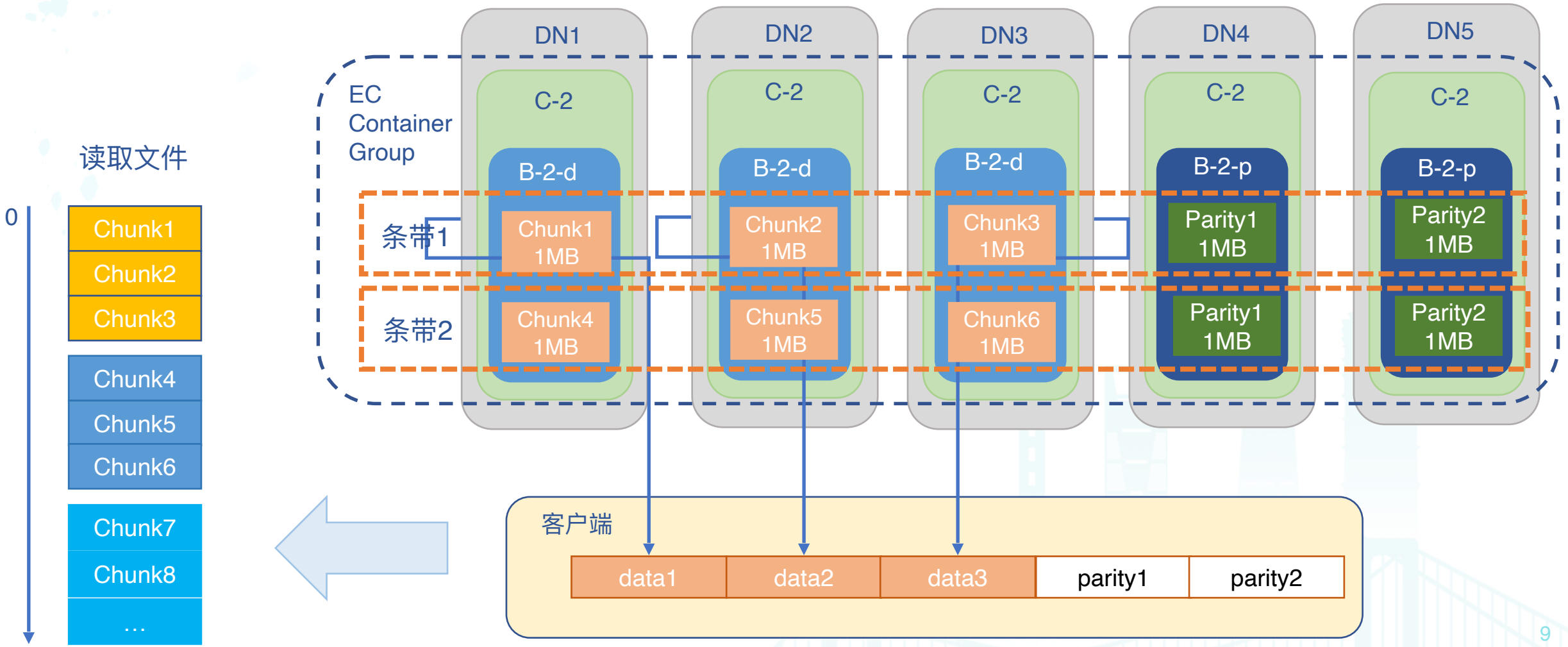
IV. EC Container Group：给定Container的一组满足EC策略的副本实例



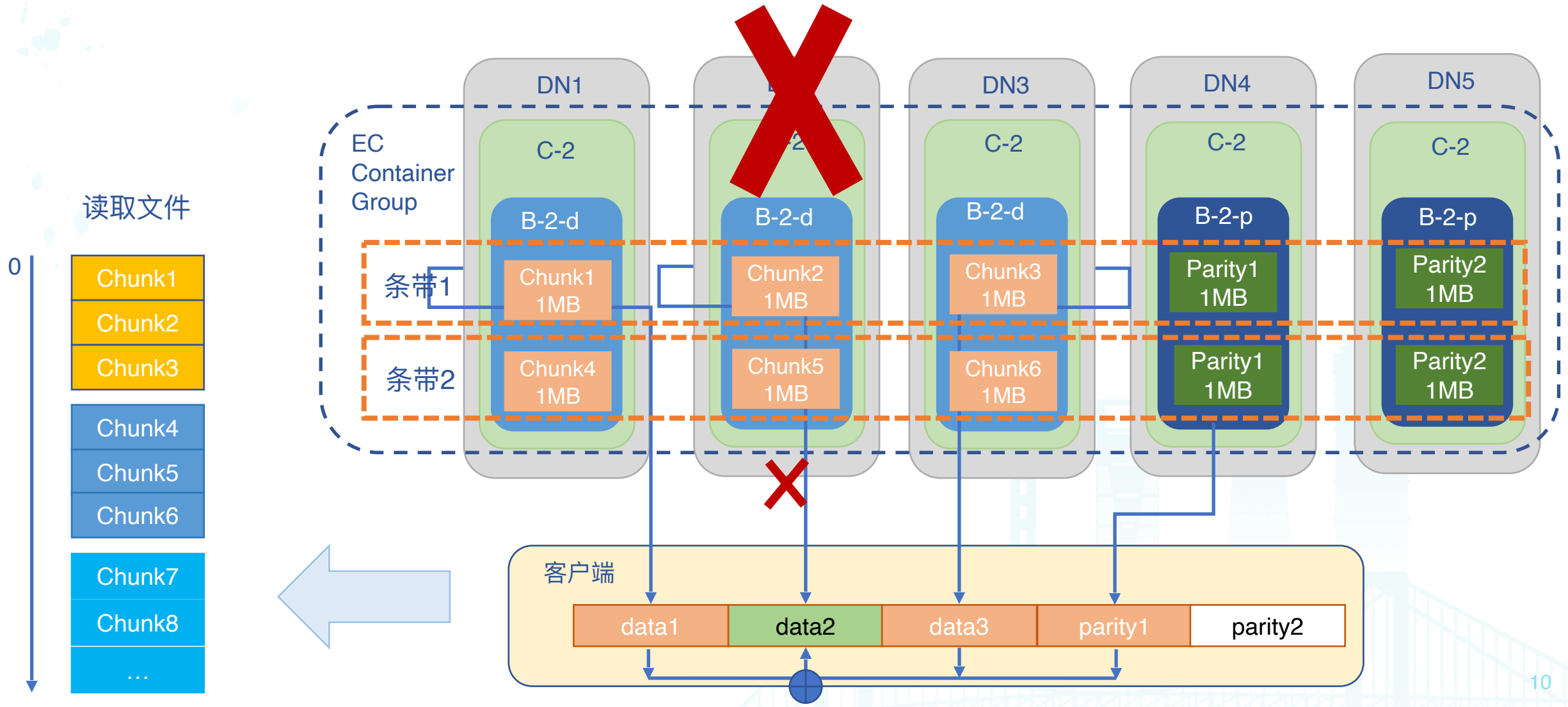
数据写入


写入文件
0
256MB
256MB
256MB
256MB
256MB
256MB
256MB
256MB
256MB
逻辑块
EC Container Group1
EC Container Group2
DN1
DN2
DN3
DN4
DN5
C-1
C-2
物理块
B-1-d
B-2-d
B-1-p
B-2-p
客户端
data1
data2
data3
parity1
parity2

数据读取


读取文件
0
Chunk1
Chunk2
Chunk3
Chunk4
Chunk5
Chunk6
Chunk7
Chunk8
…
DN1
DN2
DN3
DN4
DN5
EC Container Group
C-2
B-2-d
B-2-p
条带1
条带2
Chunk1 1MB
Chunk2 1MB
Chunk3 1MB
Chunk4 1MB
Chunk5 1MB
Chunk6 1MB
Parity1 1MB
Parity2 1MB
客户端
data1
data2
data3
parity1
parity2

数据在线修复


读取文件
0
Chunk1
Chunk2
Chunk3
Chunk4
Chunk5
Chunk6
Chunk7
Chunk8
…
DN1
DN3
DN4
DN5
EC
Container
Group
C-2
B-2-d
B-2-p
条带1
条带2
Chunk1
1MB
Chunk2
1MB
Chunk3
1MB
Chunk4
1MB
Chunk5
1MB
Chunk6
1MB
Parity1
1MB
Parity2
1MB
客户端
data1
data2
data3
parity1
parity2

Ozone支持的纠删码策略

I. 内建支持的策略
    I. RS-3-2-1024K
    II. RS-6-3-1024K
    III.XOR-2-1-1024K
II. 可定制新的策略
III.策略设置支持
    I. 全局策略设置
    II. 桶级别策略设置
    III.对象/文件级别策略设置

Container Balancer

时机

I. 新的节点加入Ozone集群

II. 删除大量数据后

好处

III. 充分利用集群资源

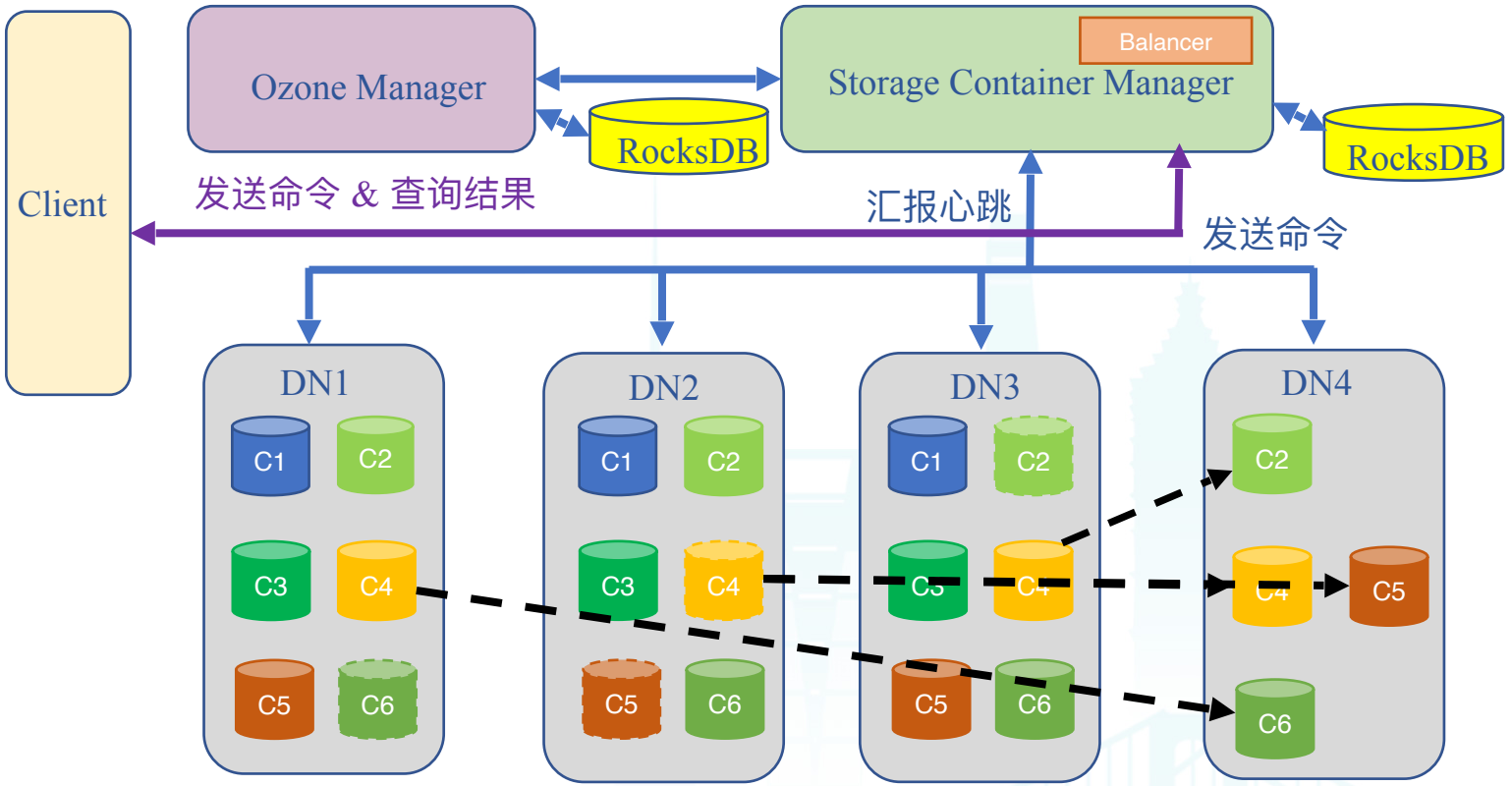
IV. 均衡集群IO访问

实现

V. Balancer实现为SCM的子功能

VI. Container是数据迁移的最小单位，只迁移CLOSE状态的Container

VII. 客户端发送命令给SCM，SCM负责执行和控制整个流程



Container Balancer

主要配置项

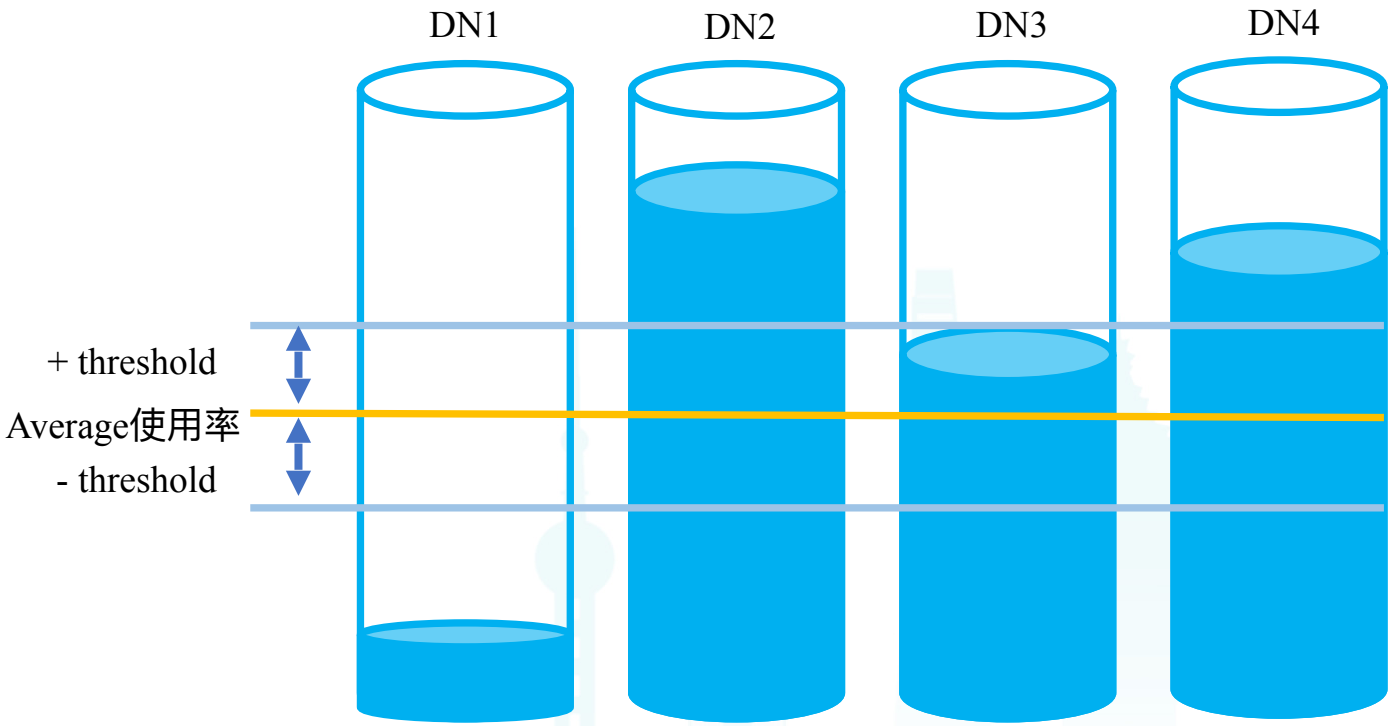
I. 启动服务

II.停止服务

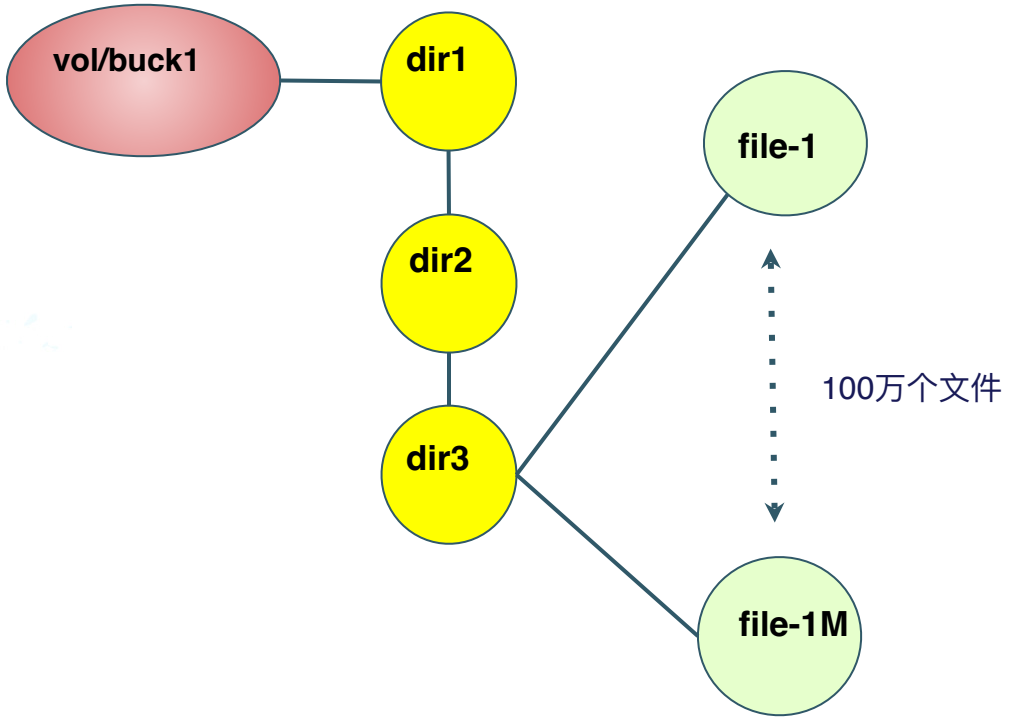
III.Threshold配置

IV.最多连续迭代运行次数

V.每次迭代最大迁移数据量



文件系统优化(File System Optimization)



对象存储：采用 **KV** 方式管理对象元数据，无需管理元数据之间的关系

文件系统：额外地，需要采用**树结构**作为索引，管理元数据之间的关系

Ozone Key的存储

Key entry
/vol/buck1/dir1/
/vol/buck1/dir1/dir2/
/vol/buck1/dir1/dir2/dir3/
/vol/buck1/dir1/dir2/dir3/file-1
/vol/buck1/dir1/dir2/dir3/file-2
/vol/buck1/dir1/dir2/dir3/file-3
……..
/vol/buck1/dir1/dir2/dir3/file-n

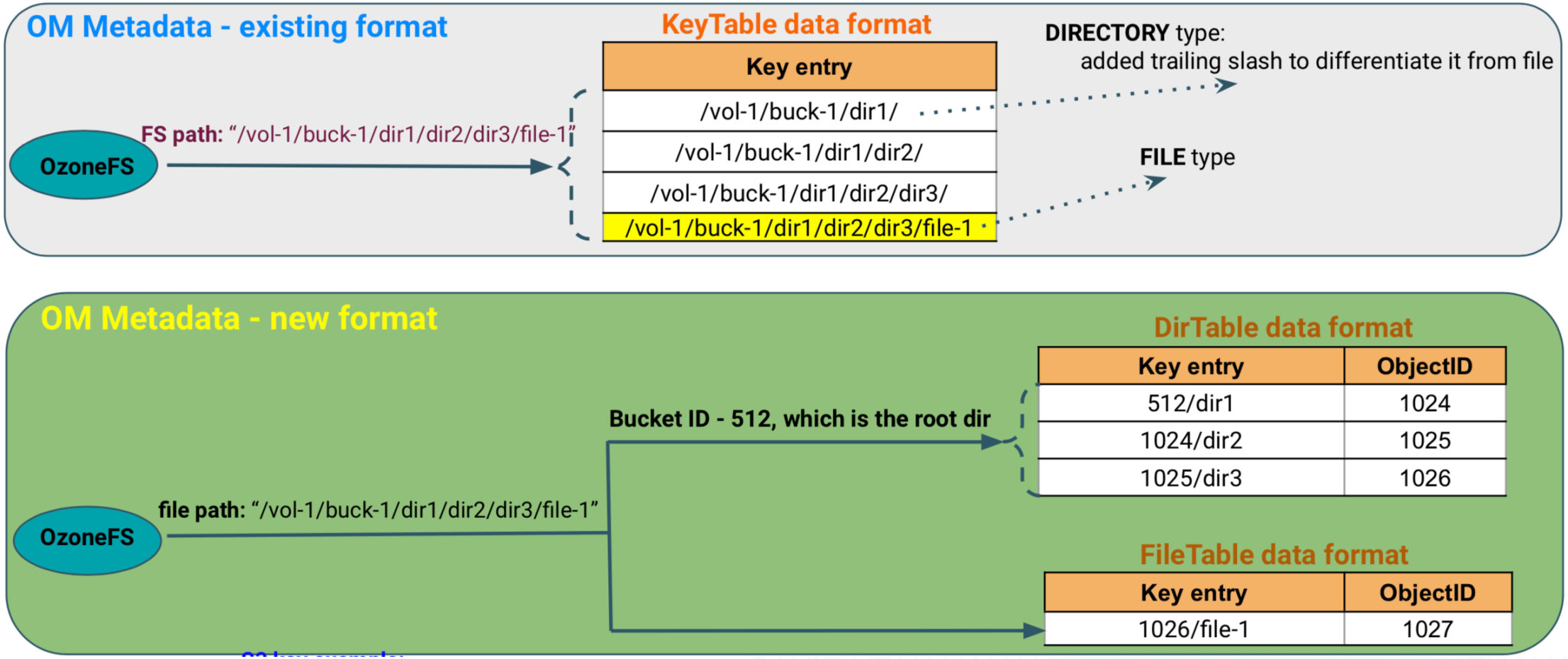
目录

文件

删除/重命名目录 耗时且非原子操作

新存储格式

Proposal : KeyTable ⟶ DirTable & FileTable



引入Bucket Metadata Layout

I. **FILE_SYSTEM_OPTIMIZED (FSO)**：支持纯粹的文件语义，有限的 S3 兼容性

文件的存储Key格式: “<parent unique-id>/<filename>”

例如, “1026/file-1”

II. **OBJECT_STORE (OBS)** : key-value 存储, 纯粹的S3 对象存储语义

对象的存储Key格式 : <keyname>

例如, “/vol-1/buck-1/dir1/dir2/dir3/file-1”

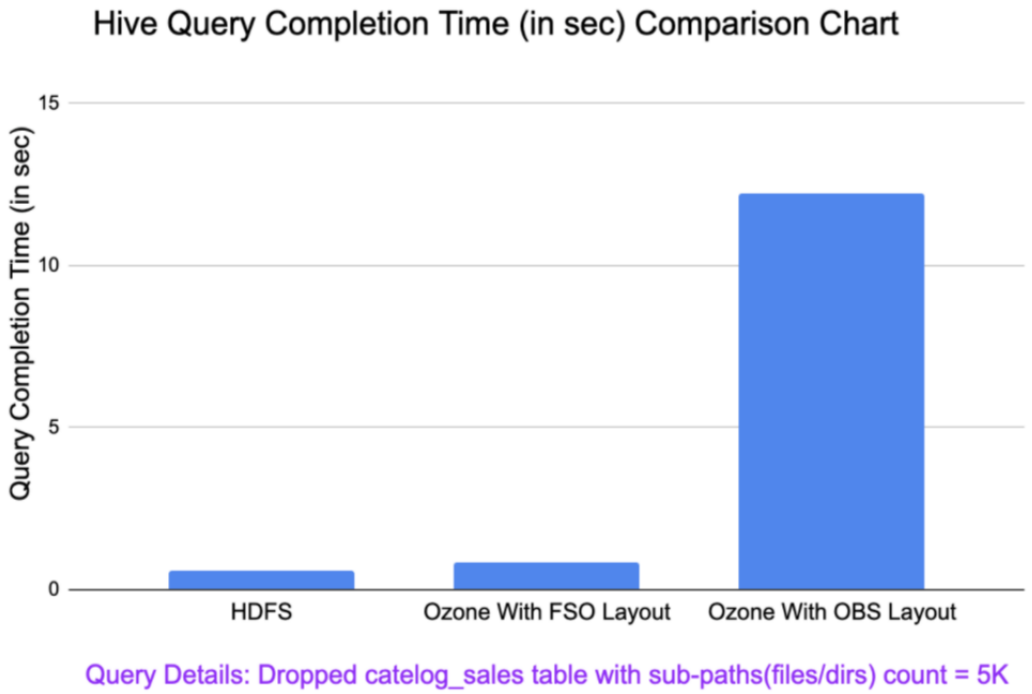
III.**LEGACY**: 所有已存在的桶，升级后变成LEGACY 版本，以支持向后兼容

存储Key格式基本同**OBS**，通过配置项区分偏向文件，还是偏向S3对象的支持

在Bucket 创建时指定Layout，后续不支持更改

Benchmark

Query Details: Dropped "catelog_sales" table with sub-paths(files/dirs) count = 5K	
	Query Completion Time (in sec)
HDFS	0.572
Ozone With FSO Layout	0.854
Ozone With OBS Layout	12.219



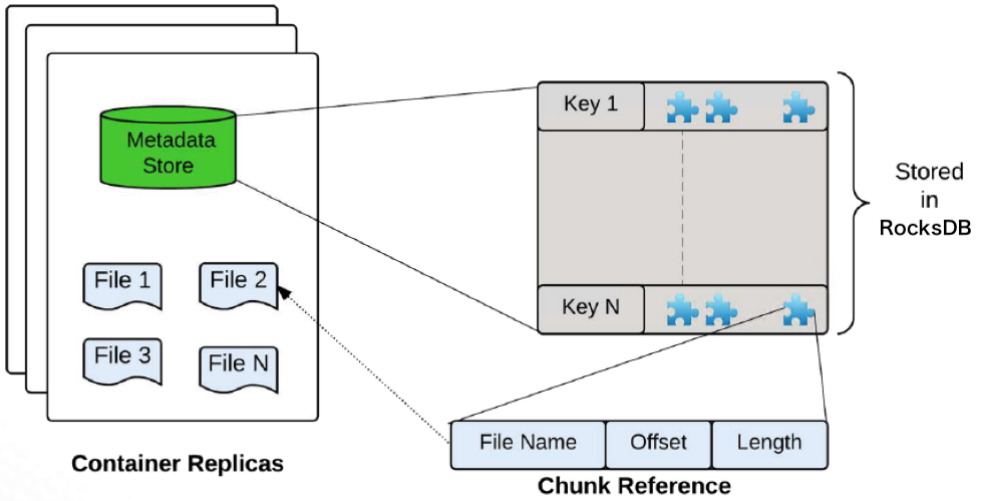
Hive 删除表(Rename操作)

- FileSystem delete on table directory path
- Moves table data to trash

举例: *fs.delete("<prefix_path>/catelog_sales")*

合并Container RocksDB实例 - 现状和问题

每个Container有独立的RocksDB实例保存元数据(V2)



问题

I. 大容量磁盘，系统中有上万个Container和RocksDB实例

II. 内存开销大，需保留众多RocksDB实例

III. 性能影响，频繁create/open/close实例

IV. 磁盘使用量，不可精准预测

V. 稳定性，频繁open/close非RocksDB的推荐用法，容易触发潜在问题

解决办法 - 合并Container RocksDB实例

新方案

每个盘上所有Container共用一个RocksDB实例保存元数据(V3)

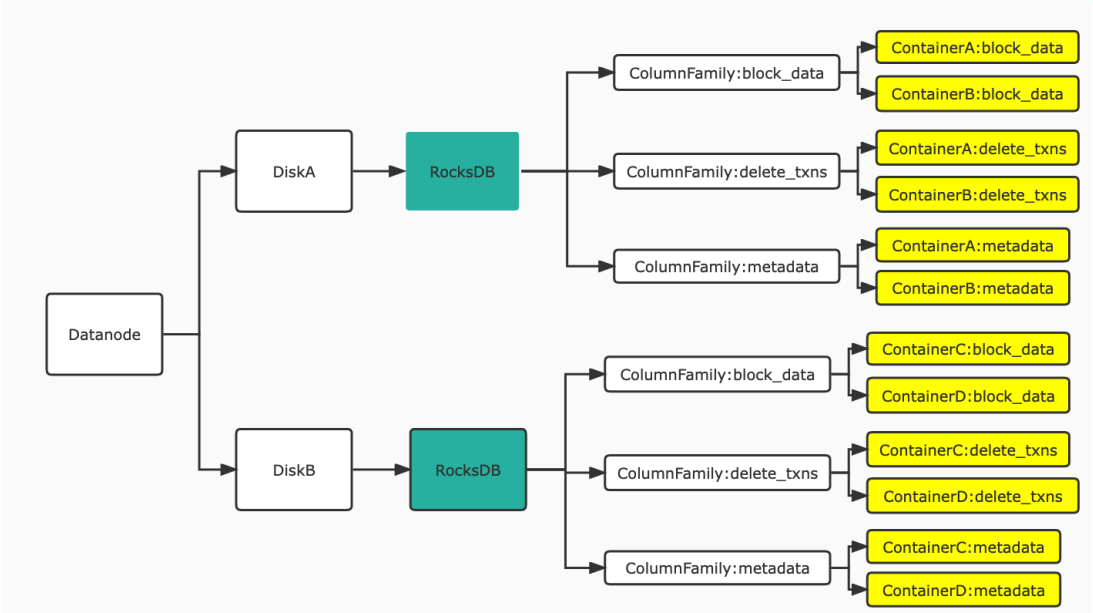
好处

I. 磁盘空间，容易估算RocksDB的磁盘使用量上限

II. 内存开销，所有RocksDB实例可Cache在内存

III. 性能影响，DN运行期间无需create/open/close实例

IV. 监控，监控RocksDB的关键指标，更好的调优参数



TableName/CF	(Key, Value)
block_data	<**ContainerID** | LocalID, BlockData>
metadata	<**ContainerID** | MetadataPrefix, MetadataValue>
delete_txns	<**ContainerID** | TxnID, DeletedBlocksTransaction>

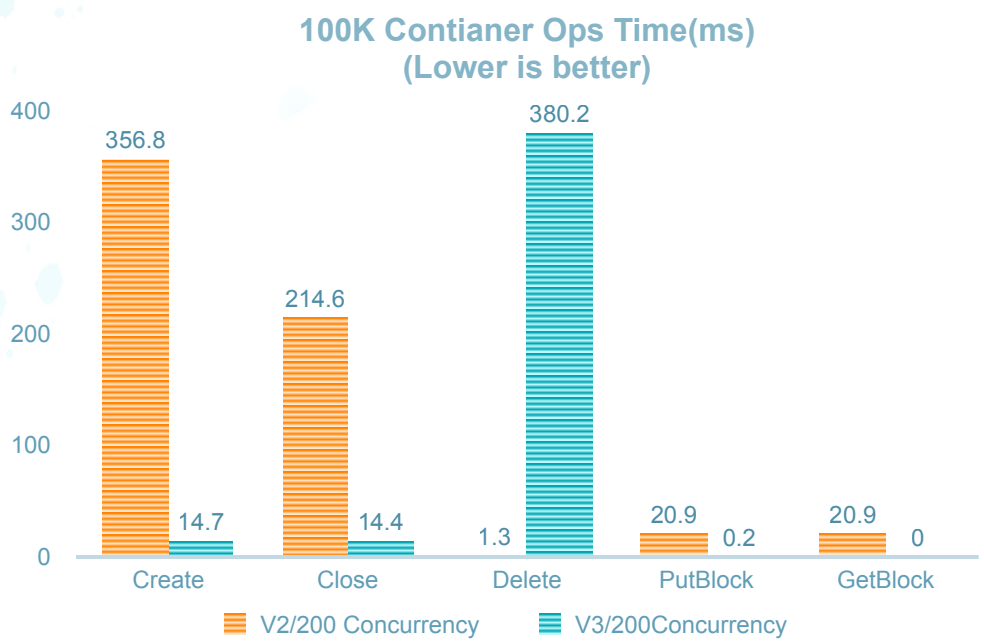
ContainerID 作为前缀

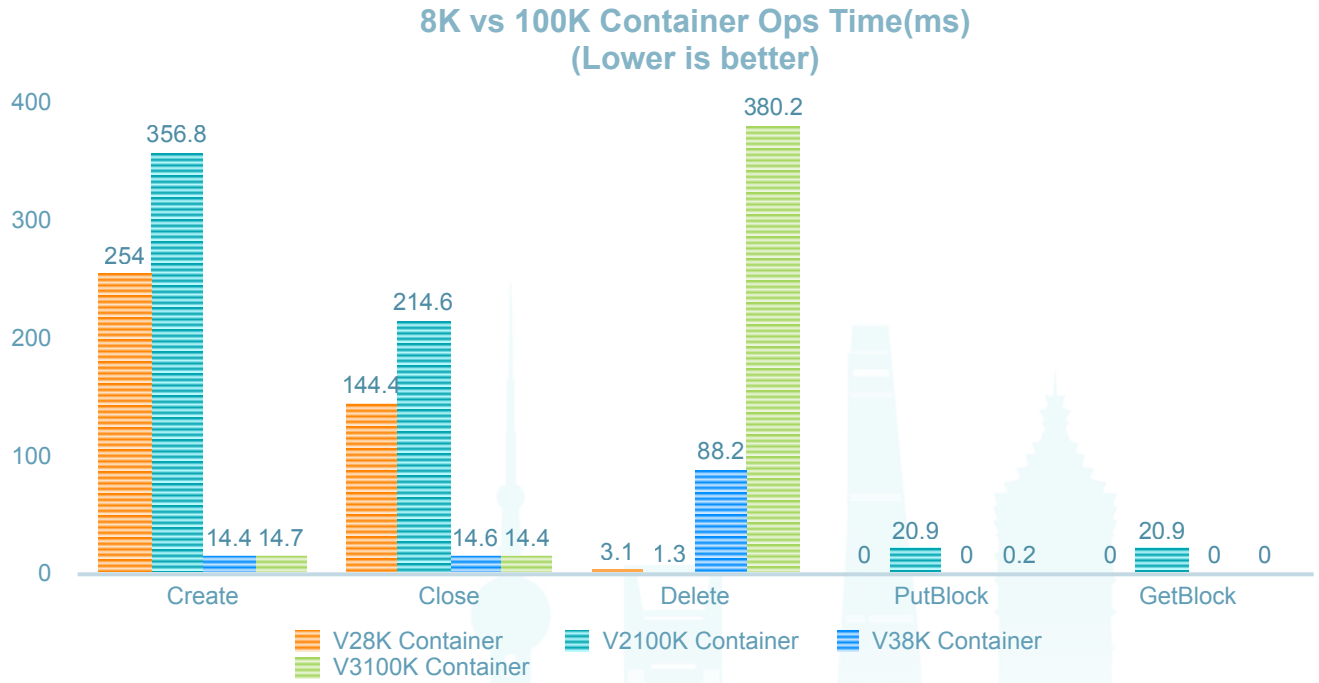
一些考量点

I. 支持两种场景
    I. RocksDB保存在每个数据盘上，适用于大部分场景
    II. RocksDB保存在独立的DB盘上。当系统中有快速SSD盘时，可以充分利用SSD盘
II. 兼容性
    I. 当前运行的Ozone系统可能已存在大量数据，考虑到升级时间，Ozone不支持在升级时将V1/V2转换成V3
    II. 所有V1 & V2 Container 数据在升级后仍然可以访问
    III. 新的数据将以V3 的形式保存(在V3 开启的情况下)
III. 一个RocksDB是否足够保存所有单盘Container的元数据？ 举例，
    - 16TB HDD，大文件场景，数据块平均大小128MB，总共 16 * 1024 * 1024 / 128 ~= 131K 数据块 足够!
    - 16TB HDD，小文件场景(例如照片)，数据块平均1MB，总共 16 * 1024 * 1024 ~= 16 million 数据块 足够!
    - 16TB HDD，极小文件场景，数据块平均1KB， 总共 16 billion 数据块 有挑战!

Micro Benchmark

Single node(96C 250G), 10 disks(3TB)





10 万Container

I. V3 大部分操作有15X ~ 100X性能提升

II. V3 Container 删除操作变慢了

Container 数量上升

I. V2 性能明显下降

II. V3 性能波动不大(除Container删除操作)

配置

属性	说明	默认值
hdds.datanode.container.db.dir	可选项，配置RocksDB的存放路径。 若未配置，则存放在数据盘上(hdds.datanode.dir)	Null(optional)
hdds.datanode.failed.db.volumes.tolerated	允许多少个配置的DB盘失败	-1
hdds.datanode.container.schema.v3.enabled	是否启动Schema V3	false

未来展望 - 开发中

I. 磁盘均衡器(Disk Balancer HDDS-5713)

II. 快照(Snapshot HDDS-6517)

THANK YOU

云 数 智 聚　　砥 柱 笃 行