COMMUNITY
THE ASF CONFERENCE
CODE

Apache Ozone: What’s new in next release

Sammi Chen (sammichen@apache.org)
Cloudera Principal Storage Engineer

CONTENTS

1. Ozone Snapshot
2. HBase on Ozone
3. Recon New Functions
4. Data tiering

Ozone Snapshot

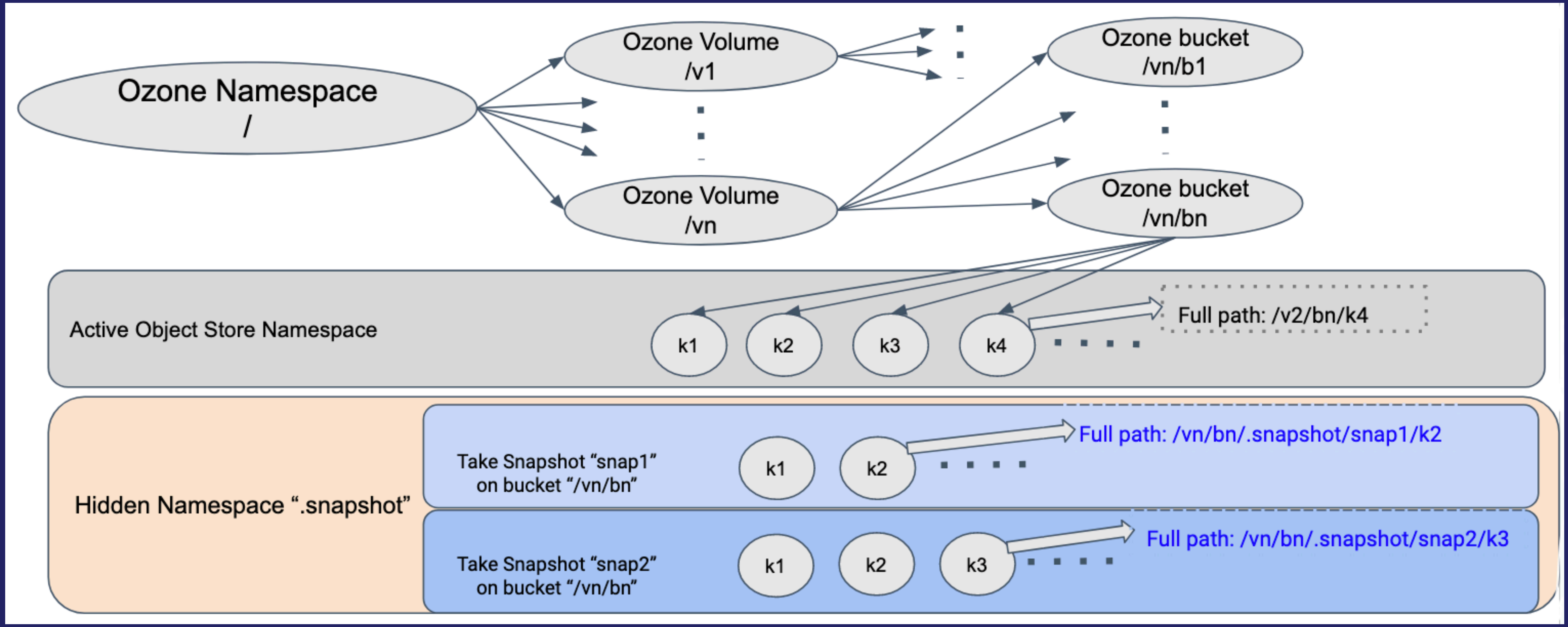
✓ Recovery from user/application errors

✓ Auditing and/or reporting on views of data at specific time

✓ Application testing

Snapshot Overview

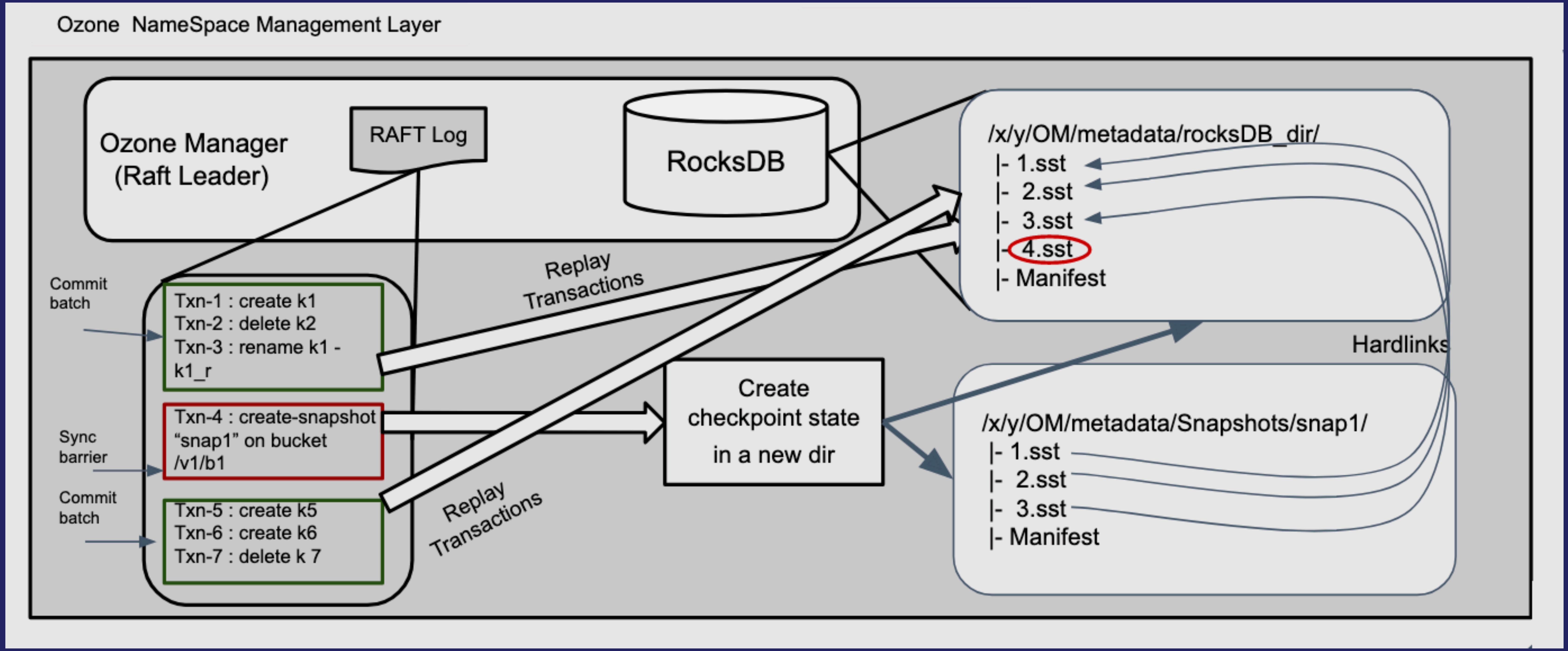
- ✓ Bucket level snapshot
- ✓ Operations
  - Create/List/Delete/Diff
- ✓ Disallow snapshot nest



COMMUNITY
THE ASF CONFERENCE
CODE

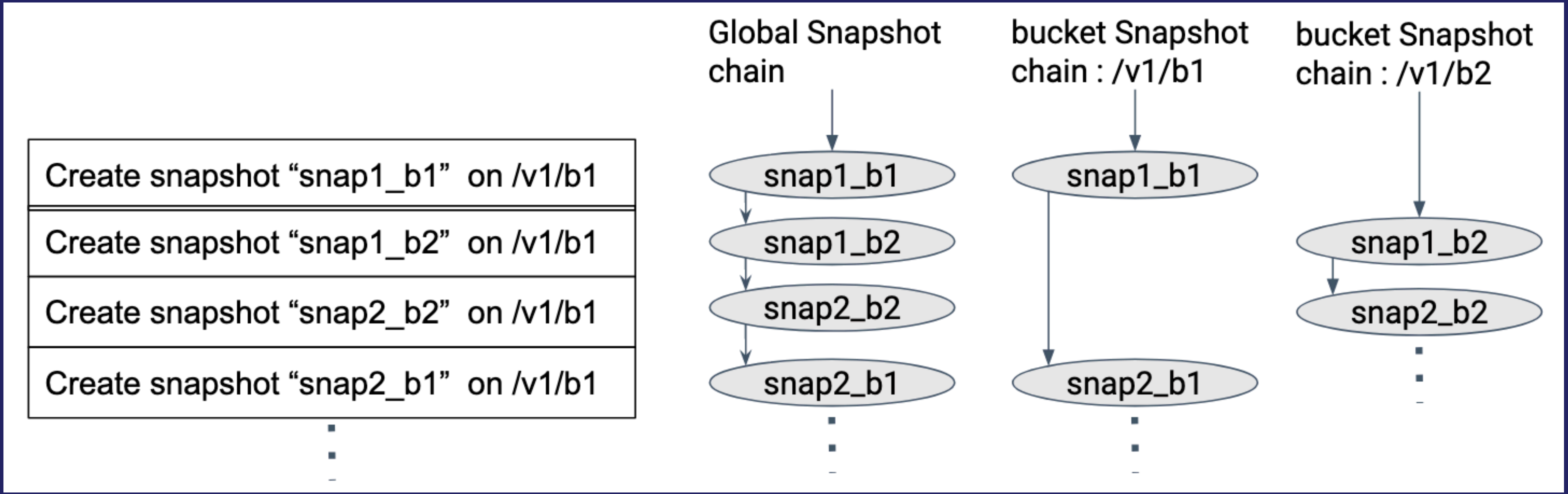
Snapshot Overview

- ✓ Based on RocksDB checkpoint mechanism
- ✓ Hardlink is used to avoid file copy
- ✓ Every snapshot will have a individual directory to hold all its files



Snapshot Creation

A global snapshot chain, and a snapshot chain for every bucket, based on creation time



Create snapshot “snap1_b1” on /v1/b1

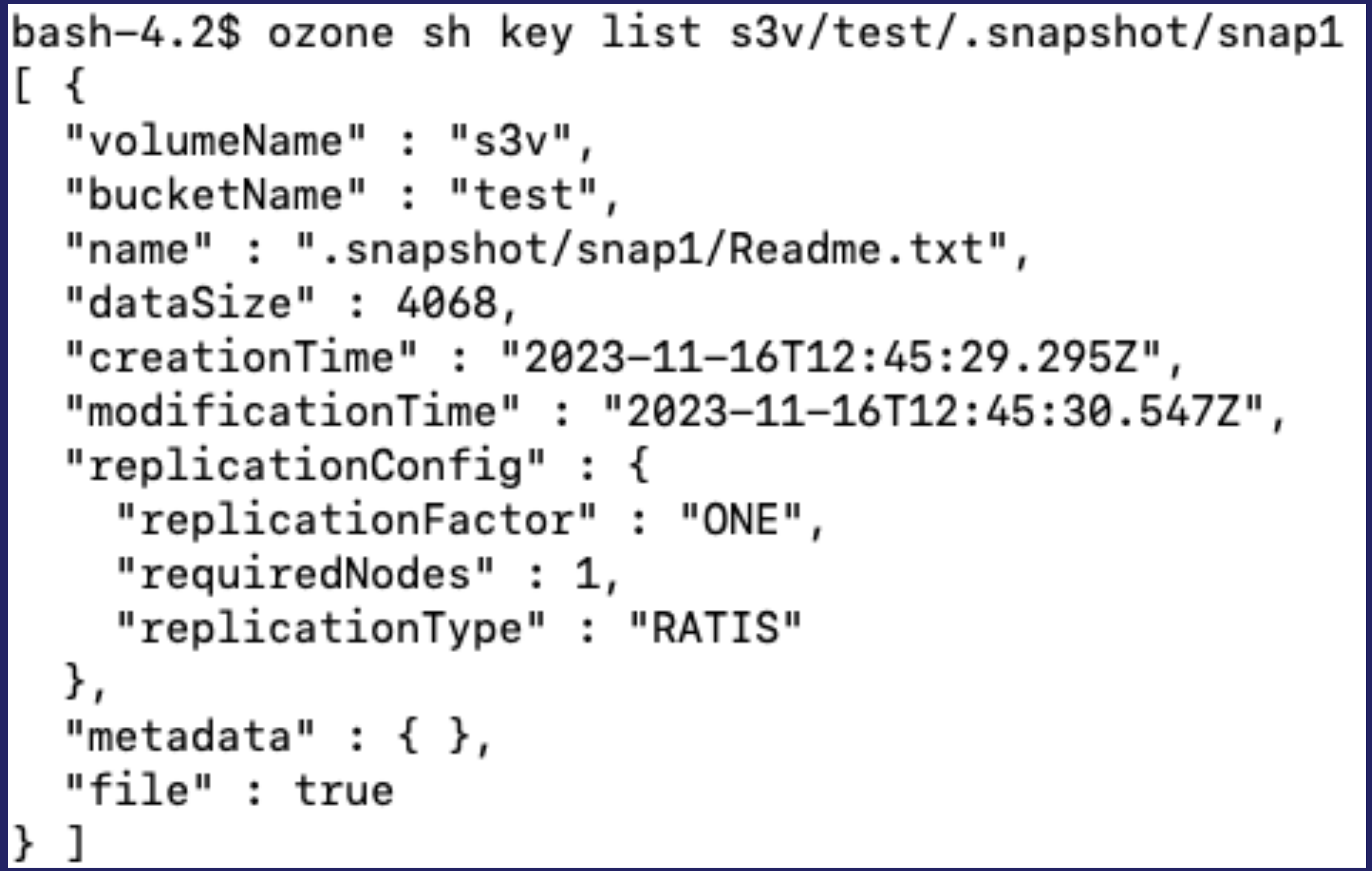
Create snapshot “snap1_b2” on /v1/b1

Create snapshot “snap2_b2” on /v1/b1

Create snapshot “snap2_b1” on /v1/b1

Snapshot Data Access

Access snapshot with “volume/bucket/.snapshot/snapshot_name”



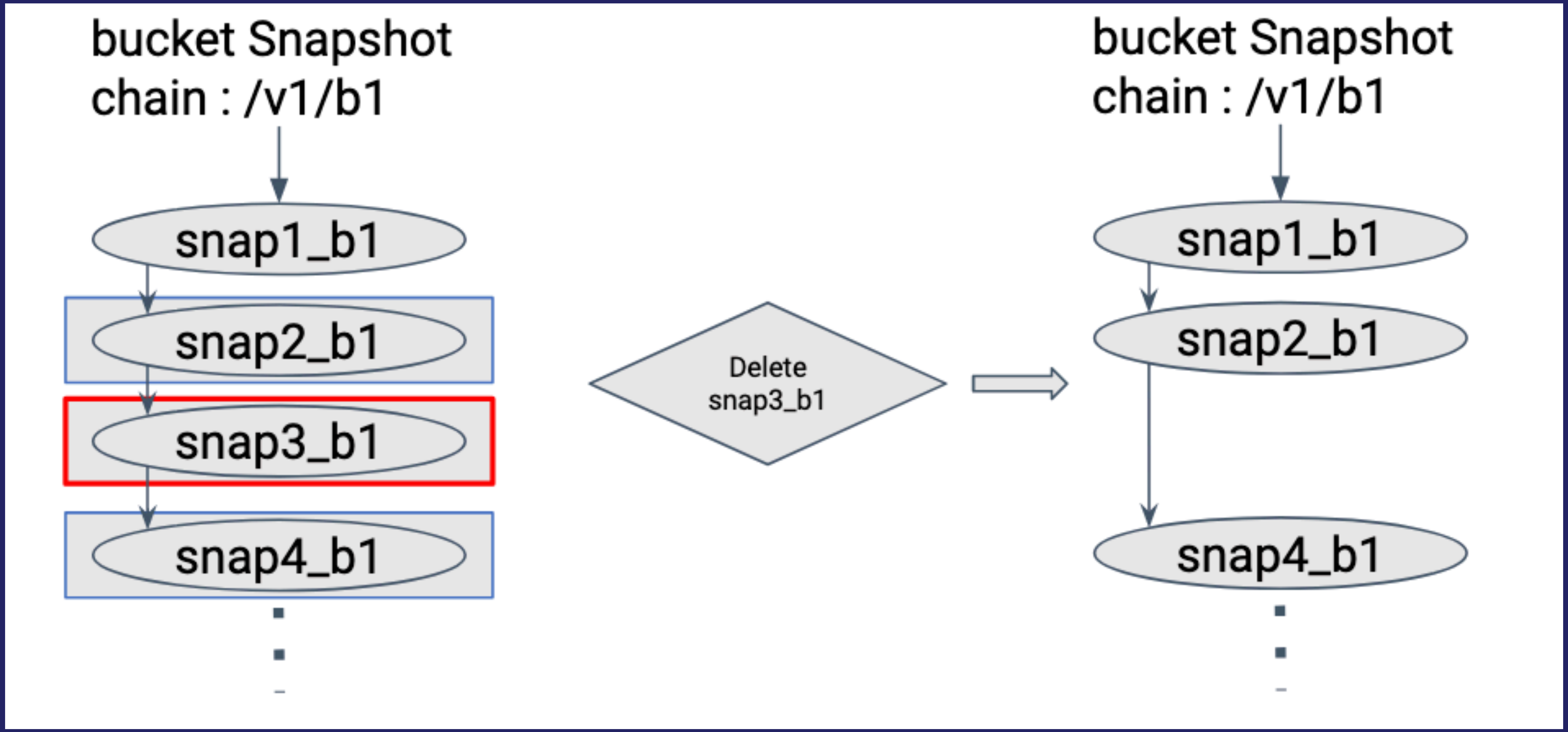
```
bash-4.2$ ozone sh key list s3v/test/.snapshot/snap1
[ {
  "volumeName" : "s3v",
  "bucketName" : "test",
  "name" : ".snapshot/snap1/Readme.txt",
  "dataSize" : 4068,
  "creationTime" : "2023-11-16T12:45:29.295Z",
  "modificationTime" : "2023-11-16T12:45:30.547Z",
  "replicationConfig" : {
    "replicationFactor" : "ONE",
    "requiredNodes" : 1,
    "replicationType" : "RATIS"
  },
  "metadata" : { },
  "file" : true
} ]
```

COMMUNITY
THE ASF CONFERENCE
CODE

Snapshot Deletion

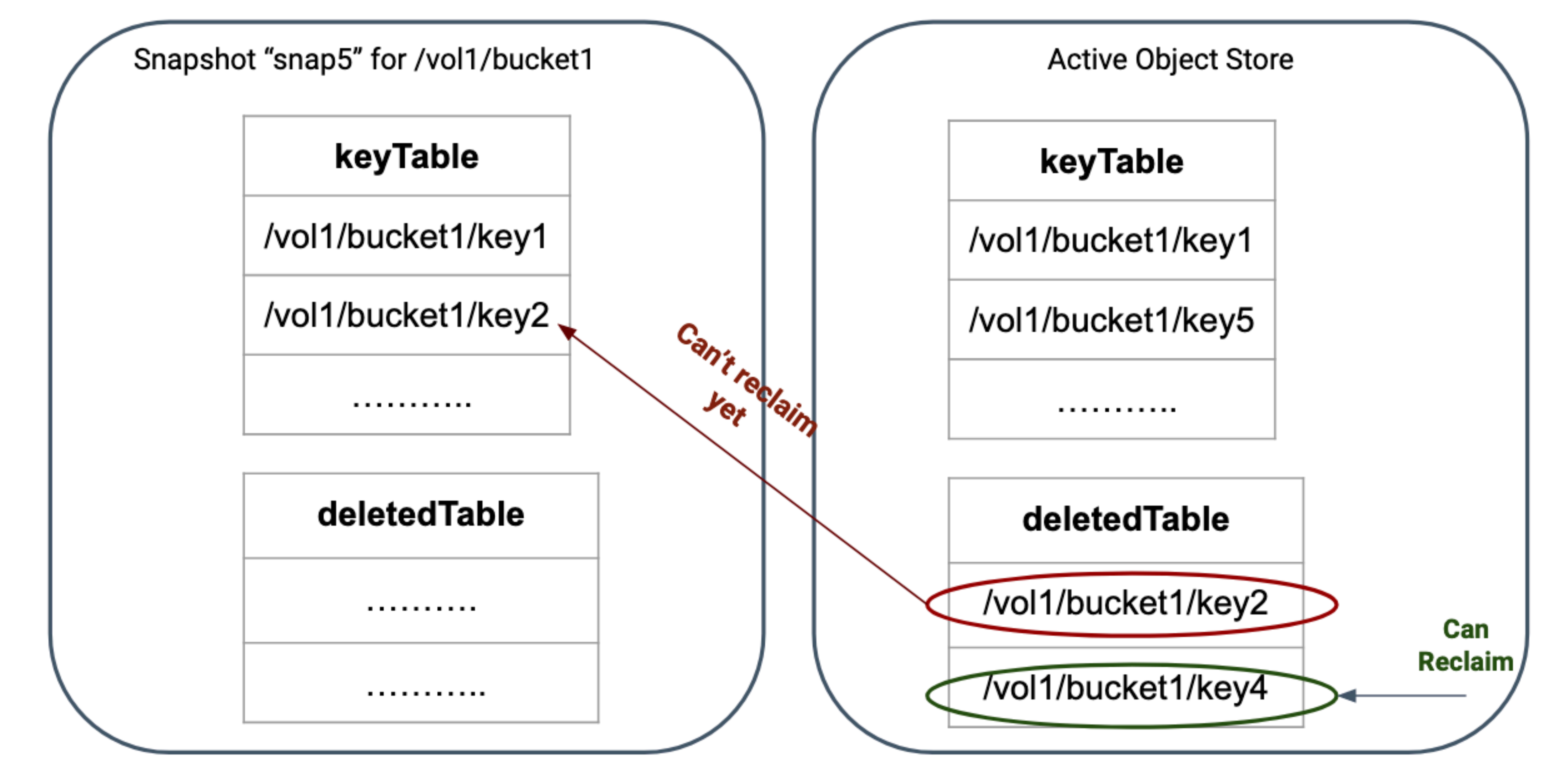
✓ Asynchronously reclaim the space the deleted snapshot holds in backend

✓ No restriction on snapshot deletion order



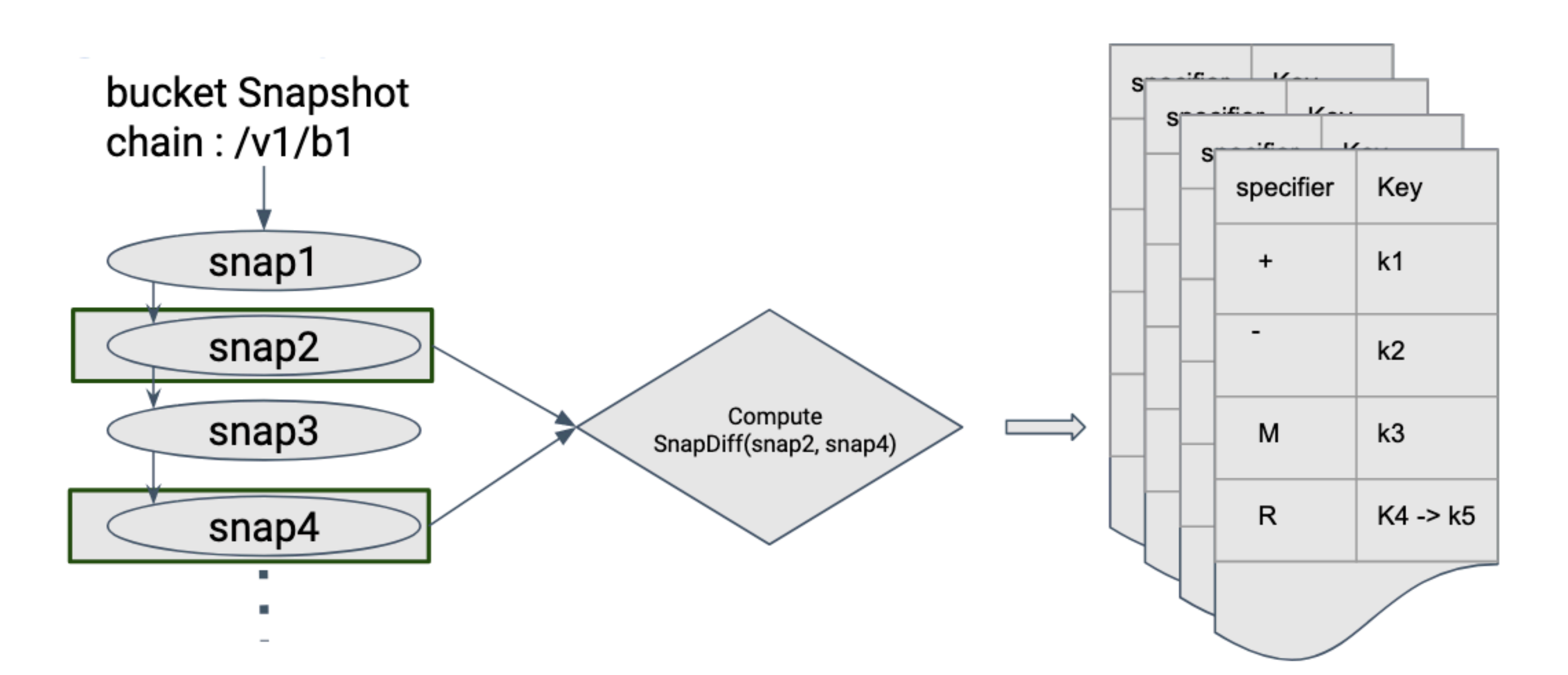
Snapshot Impact to file/key deletion

If file/key is included in any snapshot, it’s space won’t be reclaimed



Snapshot Diff

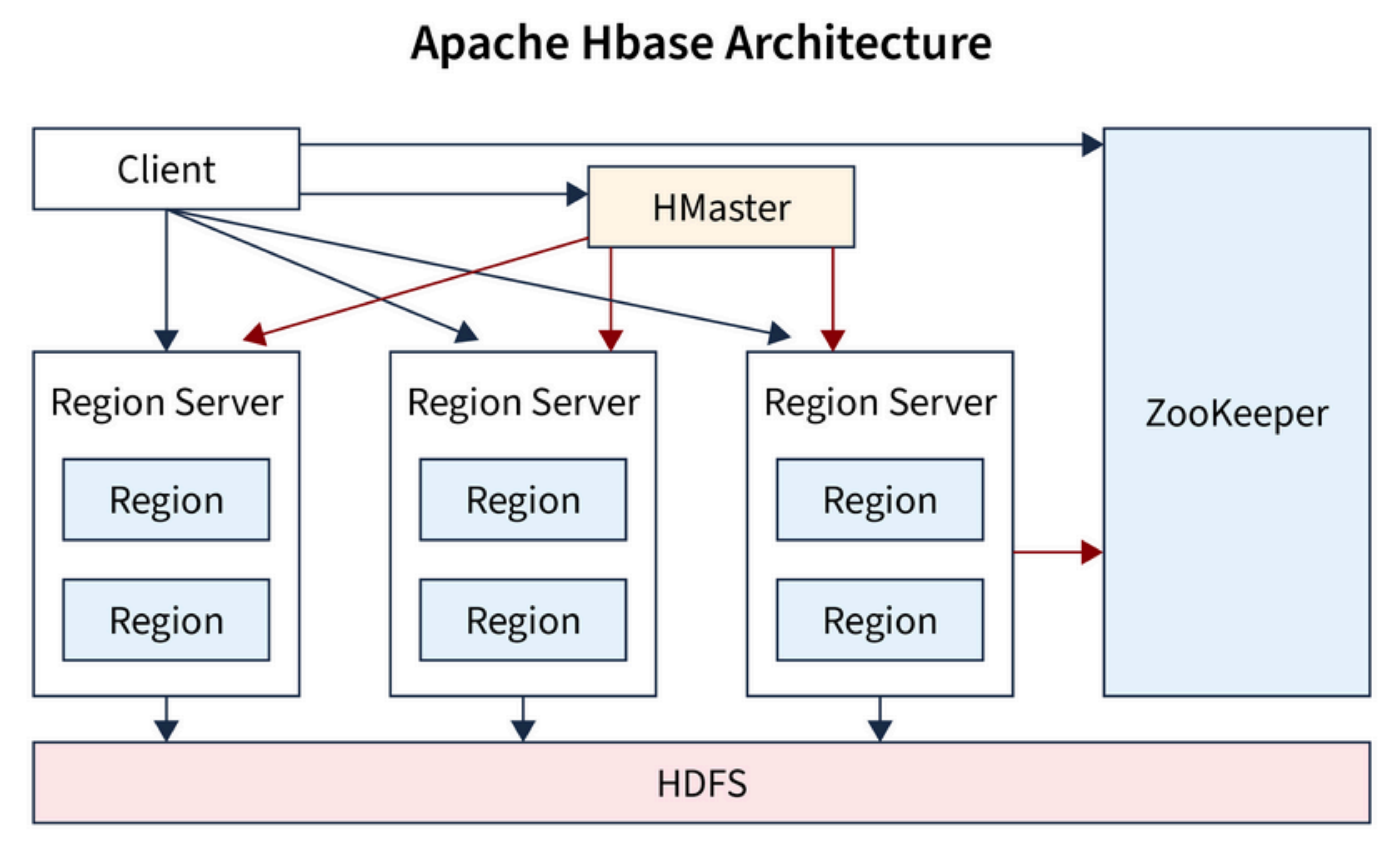
- ✓ ozone sh snapshot diff $bucket $snap1 $snap2
- ✓ Diff command will run in background


bucket Snapshot chain : /v1/b1
snap1
snap2
snap3
snap4
Compute SnapDiff(snap2, snap4)

specifier	Key
+	k1
-	k2
M	k3
R	K4 -> k5

HBase Support

Fully support all big data usages currently provided by HDFS



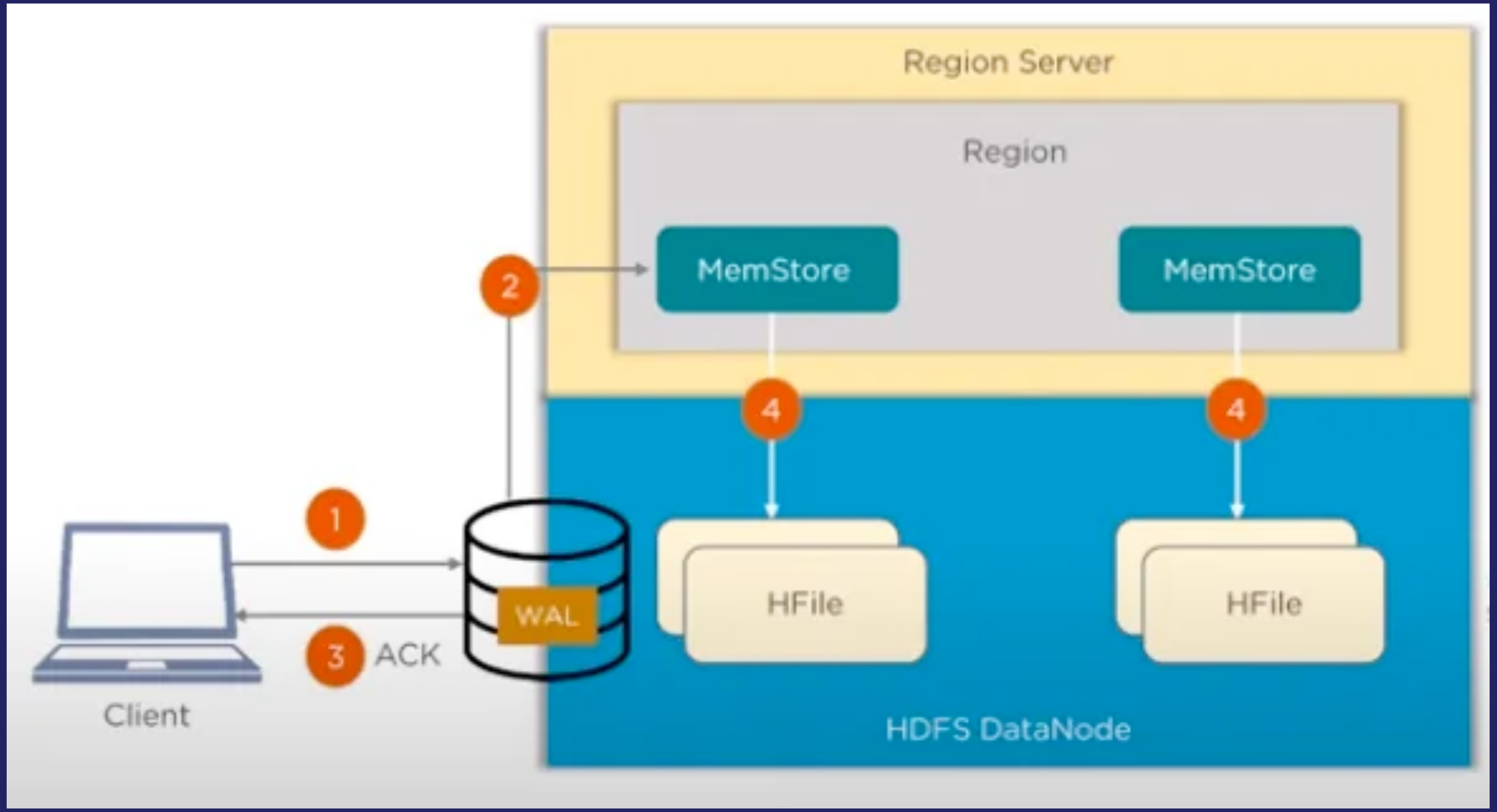
HBase File System Usage

✓ File Types

- Write Ahead Logs(WAL)
- Data file (HFile)
- References/Links (0 length file)

✓ IO Patterns

- Large Files
- Many random seeks
- Latency sensitive
- Frequent sync (WAL) to guarantee data durability
- Large number of open files

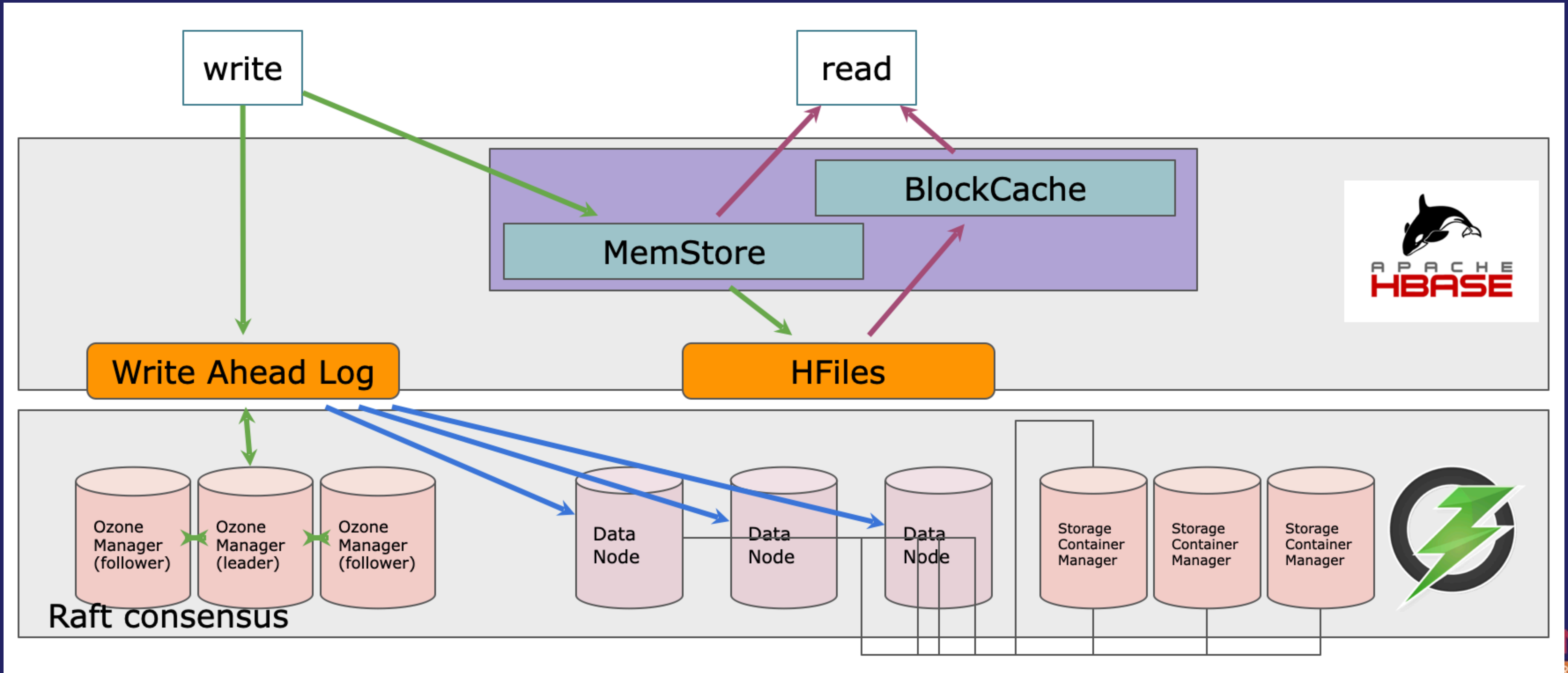


COMMUNITY OVER CODE
THE ASF CONFERENCE

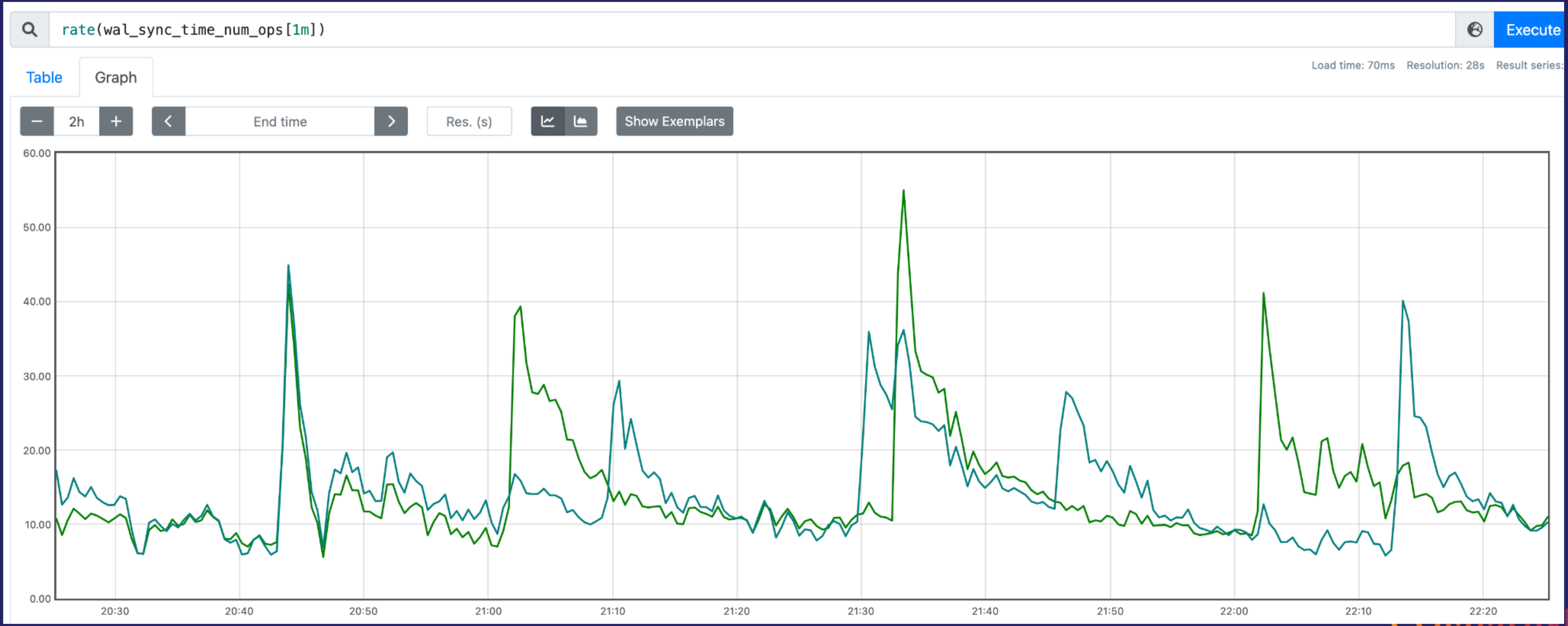
What HDFS does to support HBase

- ✓ Support hsync in HDFS (HDFS-744)
- ✓ HDFS Short Circuit Local Read (HDFS-347, HDFS-2246)
- ✓ Data Locality - A favored nodes hint to enable clients to have control over block placement (HDFS-2576)
- ✓ HDFS needs to support a very large number of open files (HDFS-374)
- ✓ Create symbolic links in HDFS (HDFS-245)

HBase on Ozone

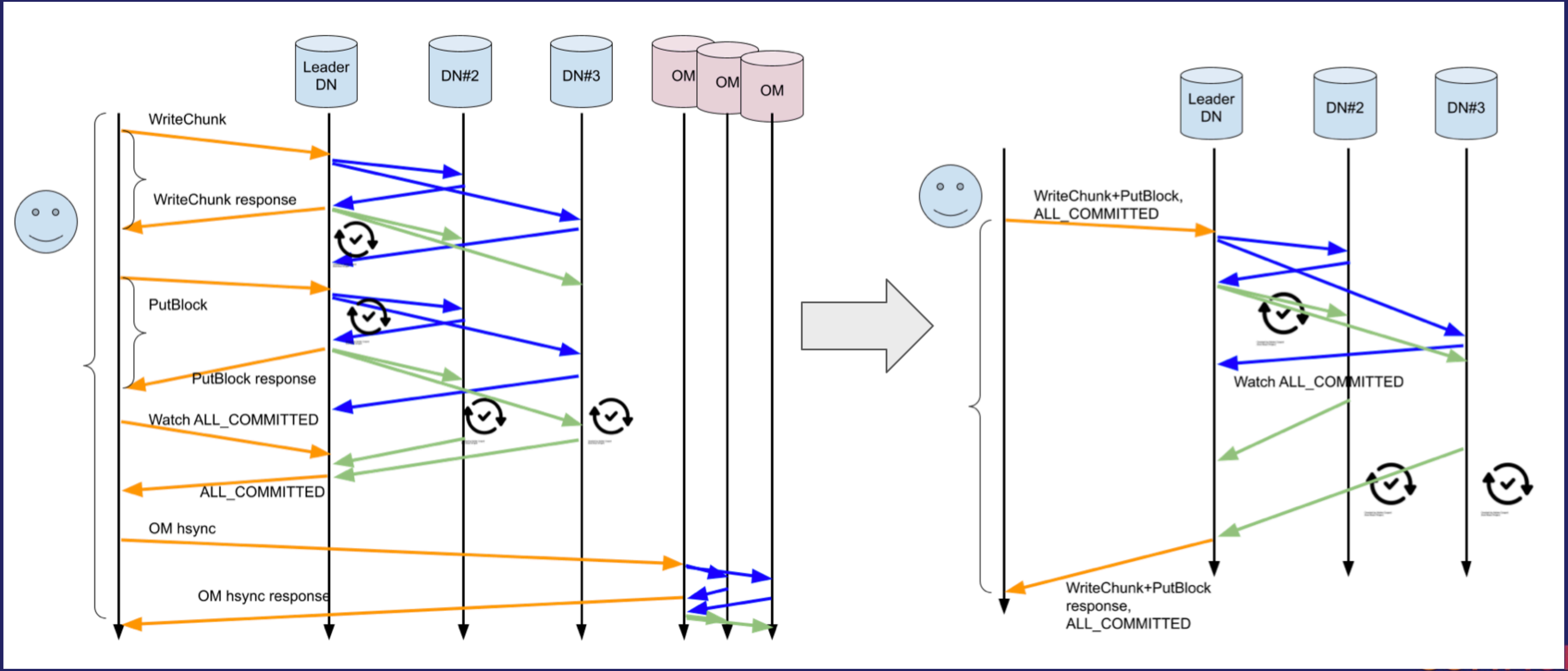


Challenge - HBase requires > 4K/s hsync per RS



```
rate(wal_sync_time_num_ops[1m])
```

Hsync optimization - Reduce RPC calls to DN and OM



Reduced from 4x RPCs down to 1x RPC

Support data majority commit option (HDDS-2887)

✓ Data all commit (better read performance)

- write succeeds after all 3 datanode's confirmation

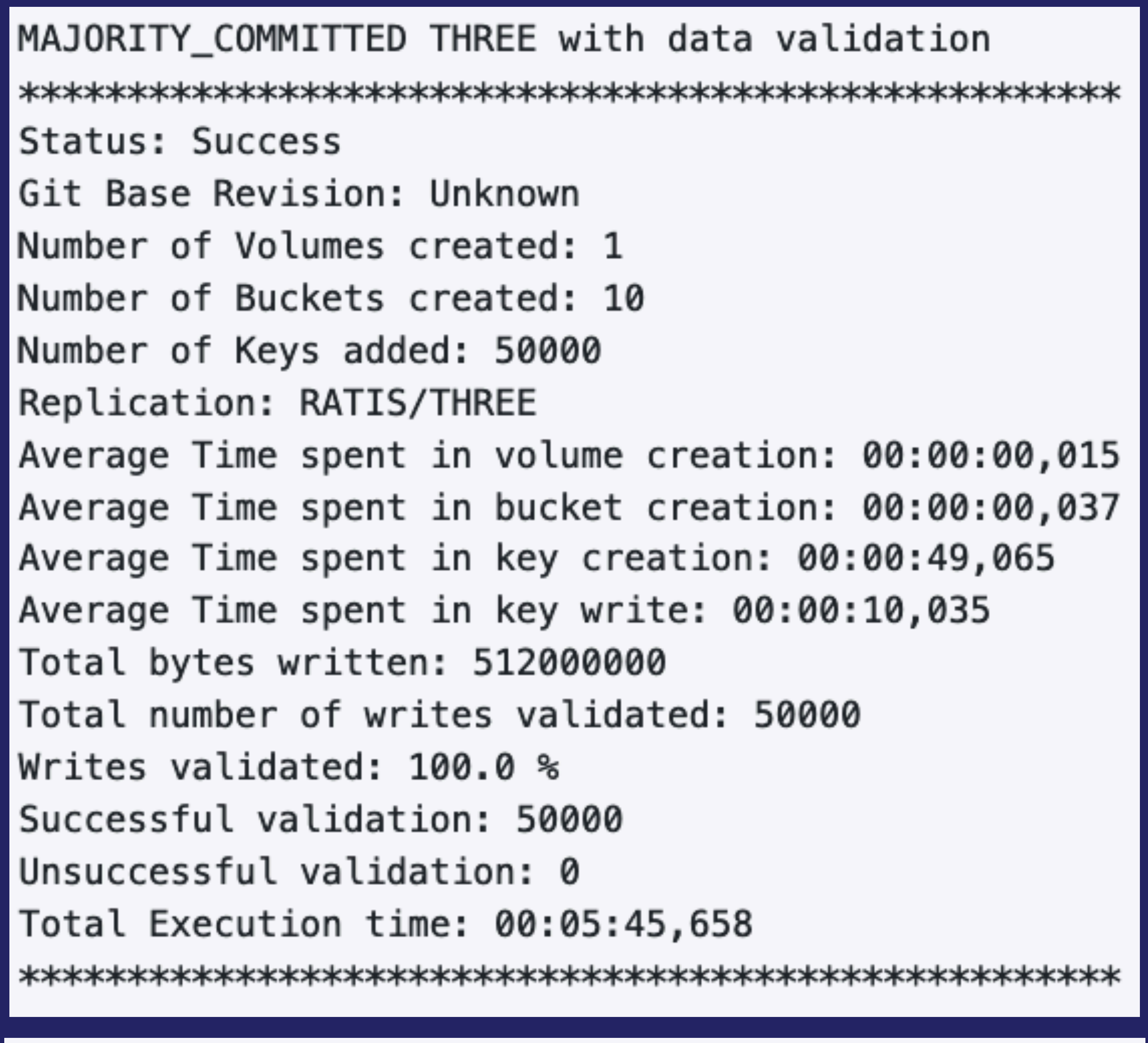
- data of 3 replica are the same

✓ Data majority commit (better write performance)

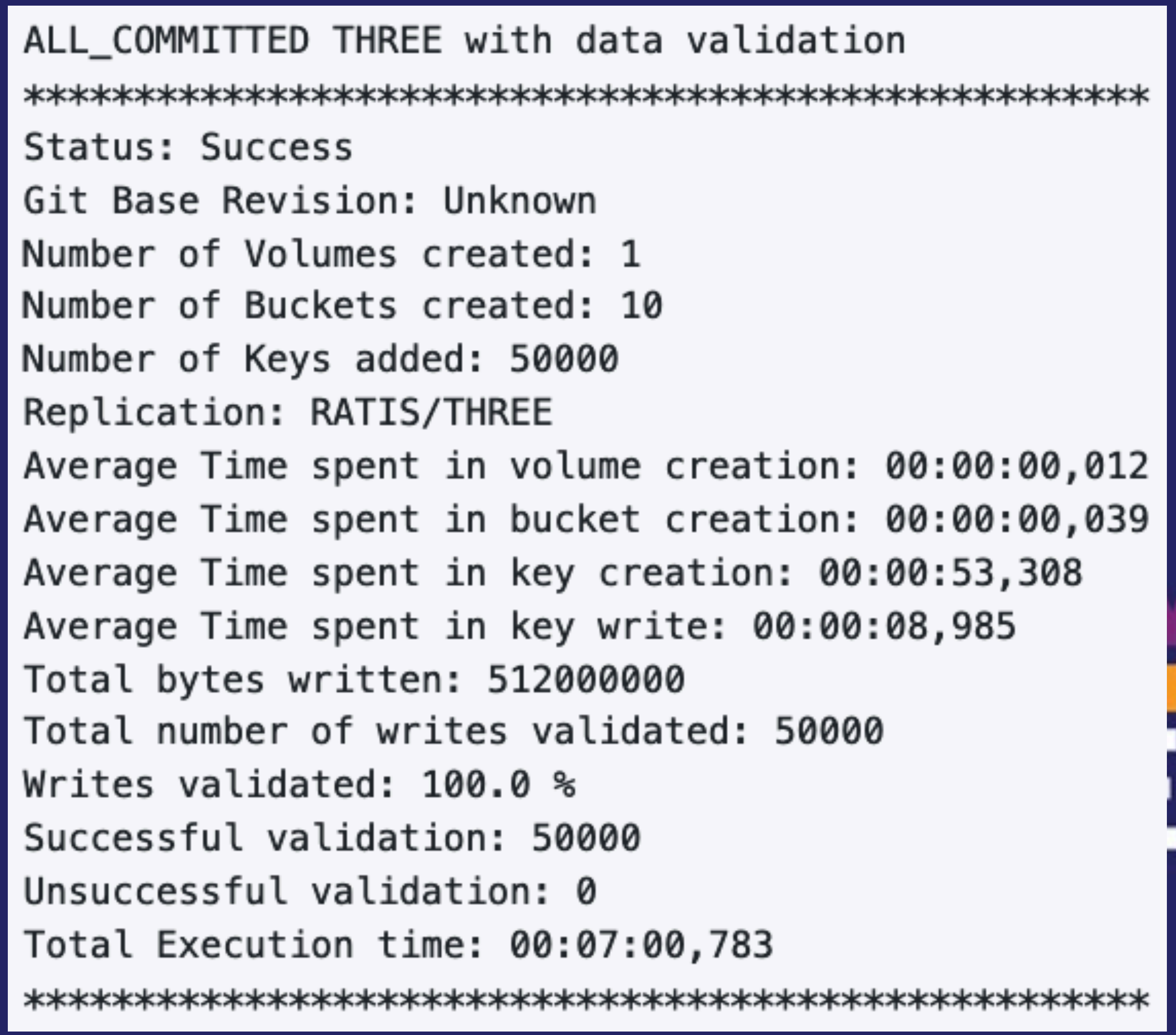
- write succeeds after majority(2) datanode's confirmation

- data of 3 replica can have different length right after the write success

- "ozone freon" shows 15% write performance improvement



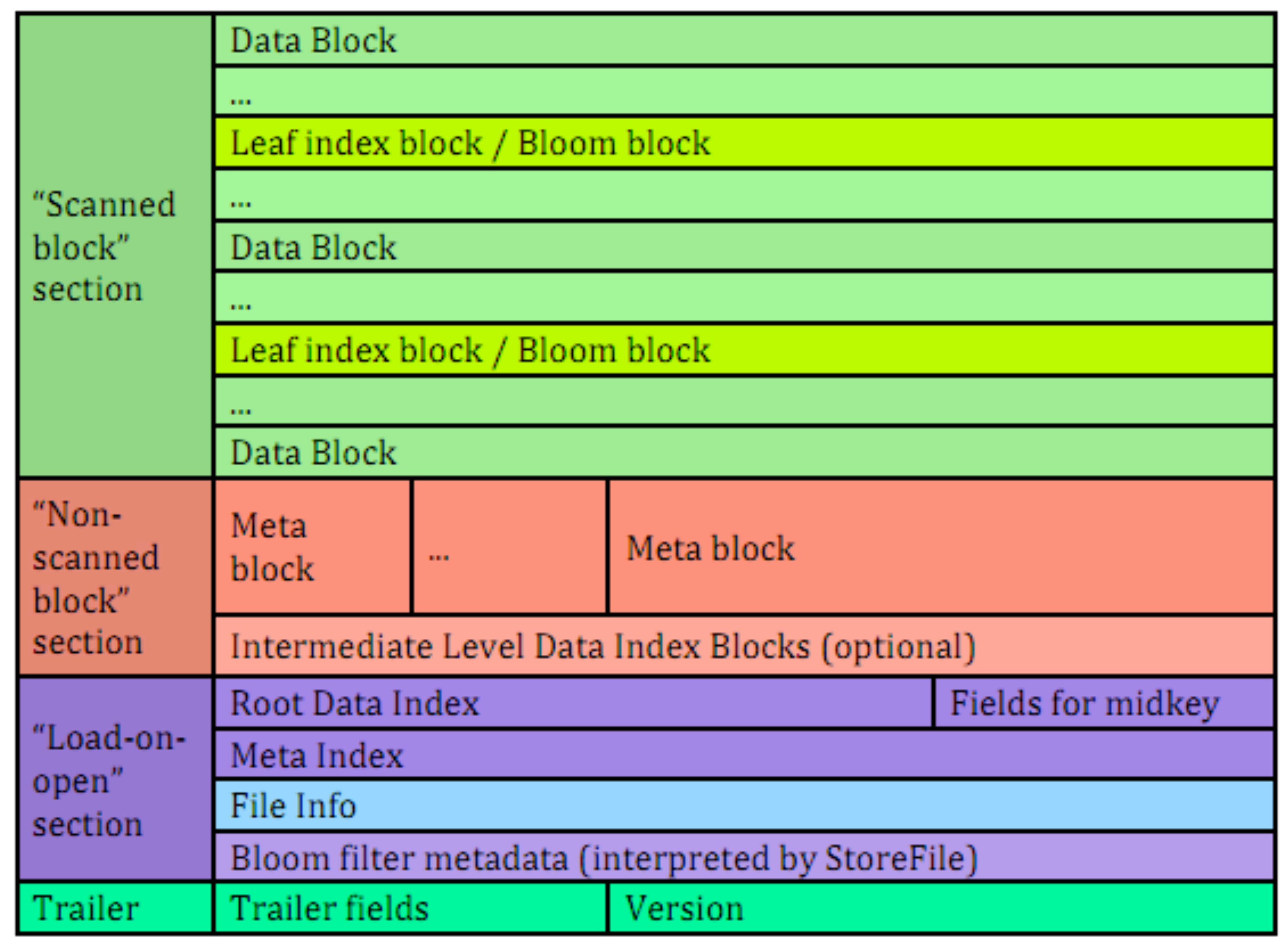
```
MAJORITY_COMMITTED THREE with data validation
***************************************************
Status: Success
Git Base Revision: Unknown
Number of Volumes created: 1
Number of Buckets created: 10
Number of Keys added: 50000
Replication: RATIS/THREE
Average Time spent in volume creation: 00:00:00,015
Average Time spent in bucket creation: 00:00:00,037
Average Time spent in key creation: 00:00:49,065
Average Time spent in key write: 00:00:10,035
Total bytes written: 512000000
Total number of writes validated: 50000
Writes validated: 100.0 %
Successful validation: 50000
Unsuccessful validation: 0
Total Execution time: 00:05:45,658
***************************************************
```



```
ALL_COMMITTED THREE with data validation
***************************************************
Status: Success
Git Base Revision: Unknown
Number of Volumes created: 1
Number of Buckets created: 10
Number of Keys added: 50000
Replication: RATIS/THREE
Average Time spent in volume creation: 00:00:00,012
Average Time spent in bucket creation: 00:00:00,039
Average Time spent in key creation: 00:00:53,308
Average Time spent in key write: 00:00:08,985
Total bytes written: 512000000
Total number of writes validated: 50000
Writes validated: 100.0 %
Successful validation: 50000
Unsuccessful validation: 0
Total Execution time: 00:07:00,783
***************************************************
```

Read Performance

- ✓ Fewer sequential reads. HBase is not analytics engine. It doesn’t ‘scan’ a lot
- ✓ Random reads are small. (HBase block size in HFile recommendation 8KB to 1MB)
- ✓ Ozone Client by default reads 1MB data each time



“Scanned block” section	Data Block		
	...		
	Leaf index block / Bloom block		
	...		
	Data Block		
	...		
	Leaf index block / Bloom block		
	...		
	Data Block		
“Non-scanned block” section	Meta block	...	Meta block
	Intermediate Level Data Index Blocks (optional)		
“Load-on-open” section	Root Data Index	Fields for midkey	
	Meta Index		
	File Info		
	Bloom filter metadata (interpreted by StoreFile)		
Trailer	Trailer fields	Version	

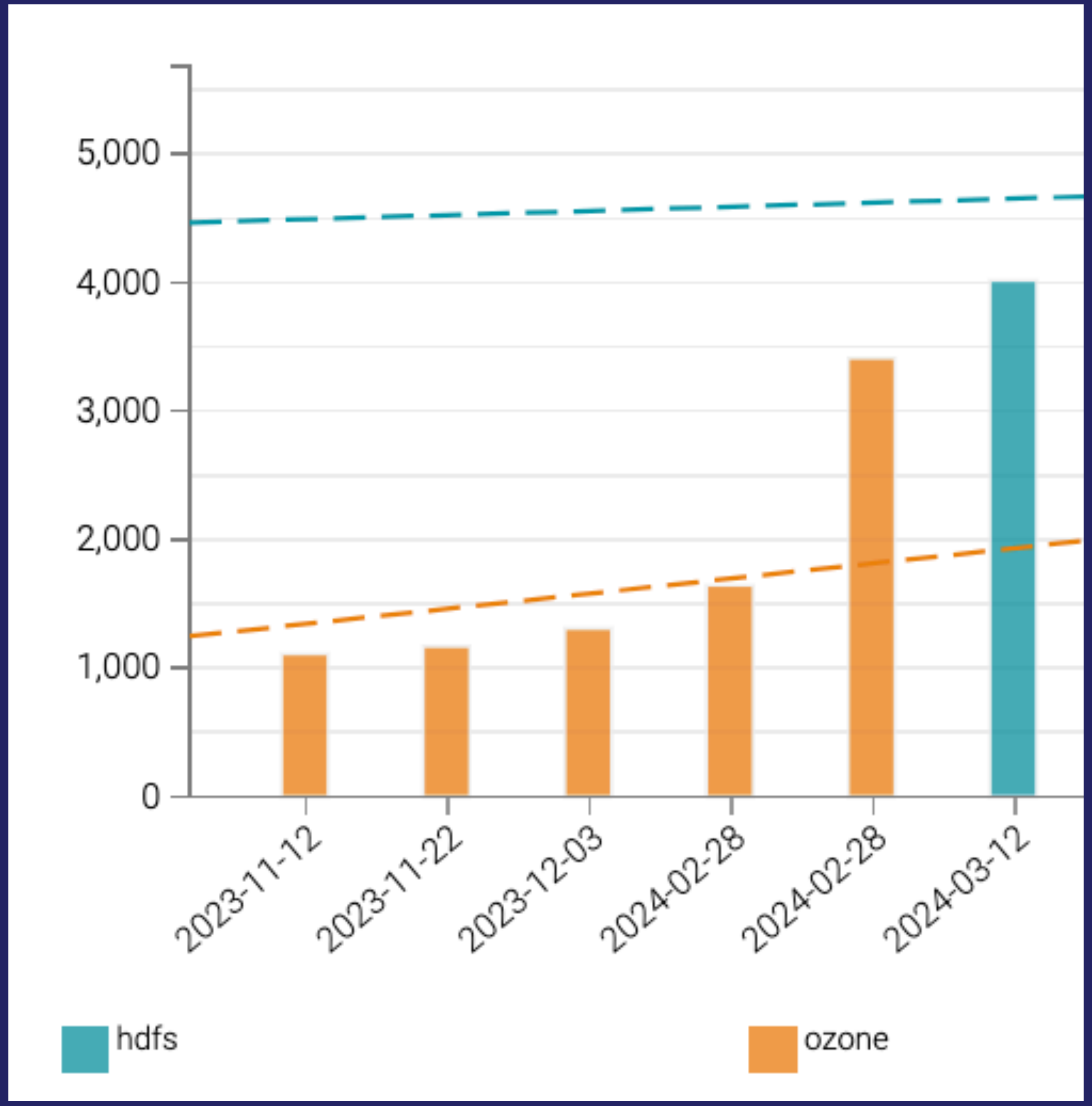
HFile Format (v2)

http://svn.apache.org/repos/asf/hbase/hbase.apache.org/trunk/0.94/book/apes03.html

Read Performance Optimization

✓ Reduce unnecessary data read by Ozone client, changing “ozone.client.bytes.per.checksum” from default 1MB to 16KB (HDDS-10465)

✓ Short circuit read support in Ozone (HBASE-27982) - In progress

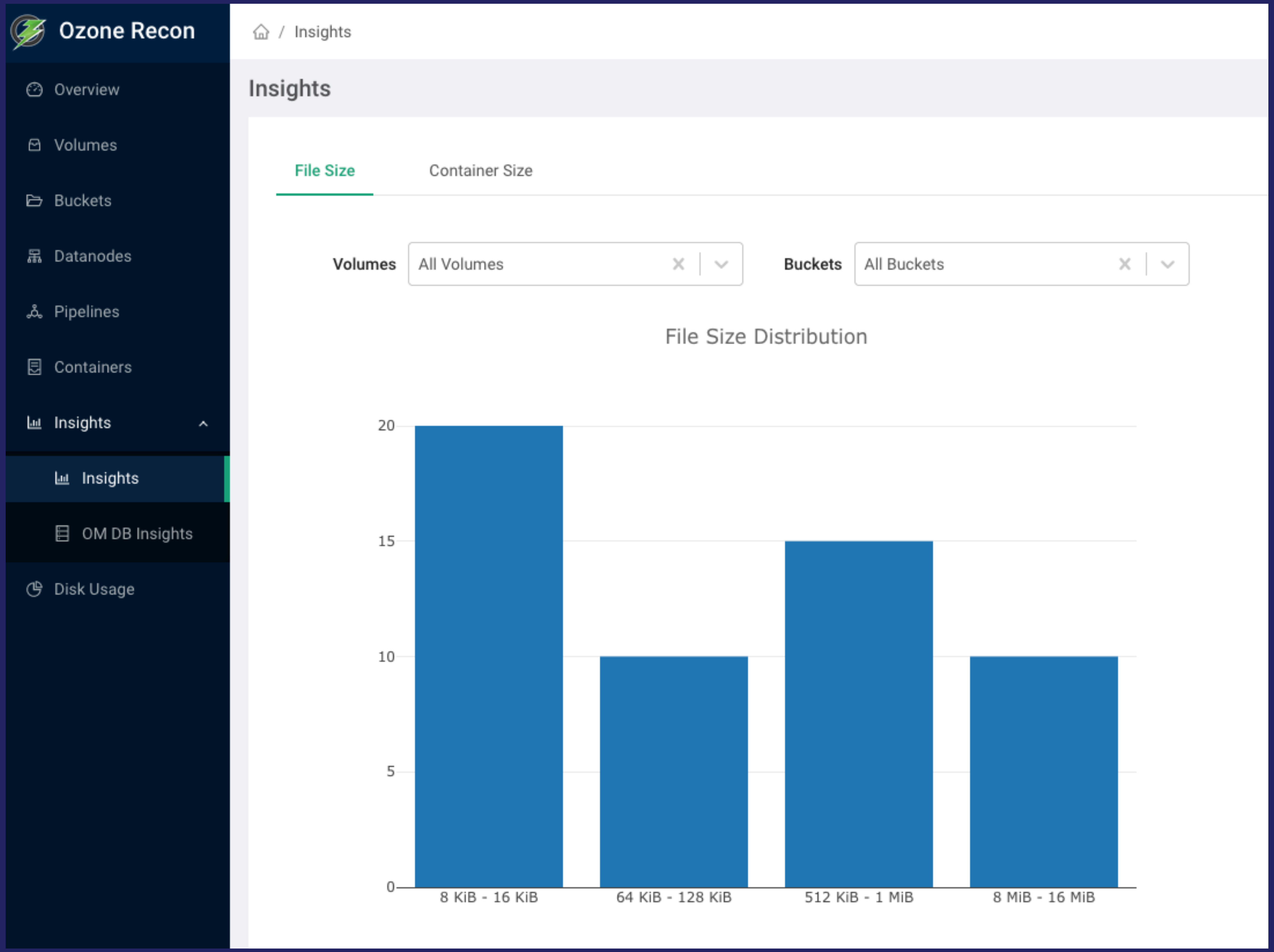


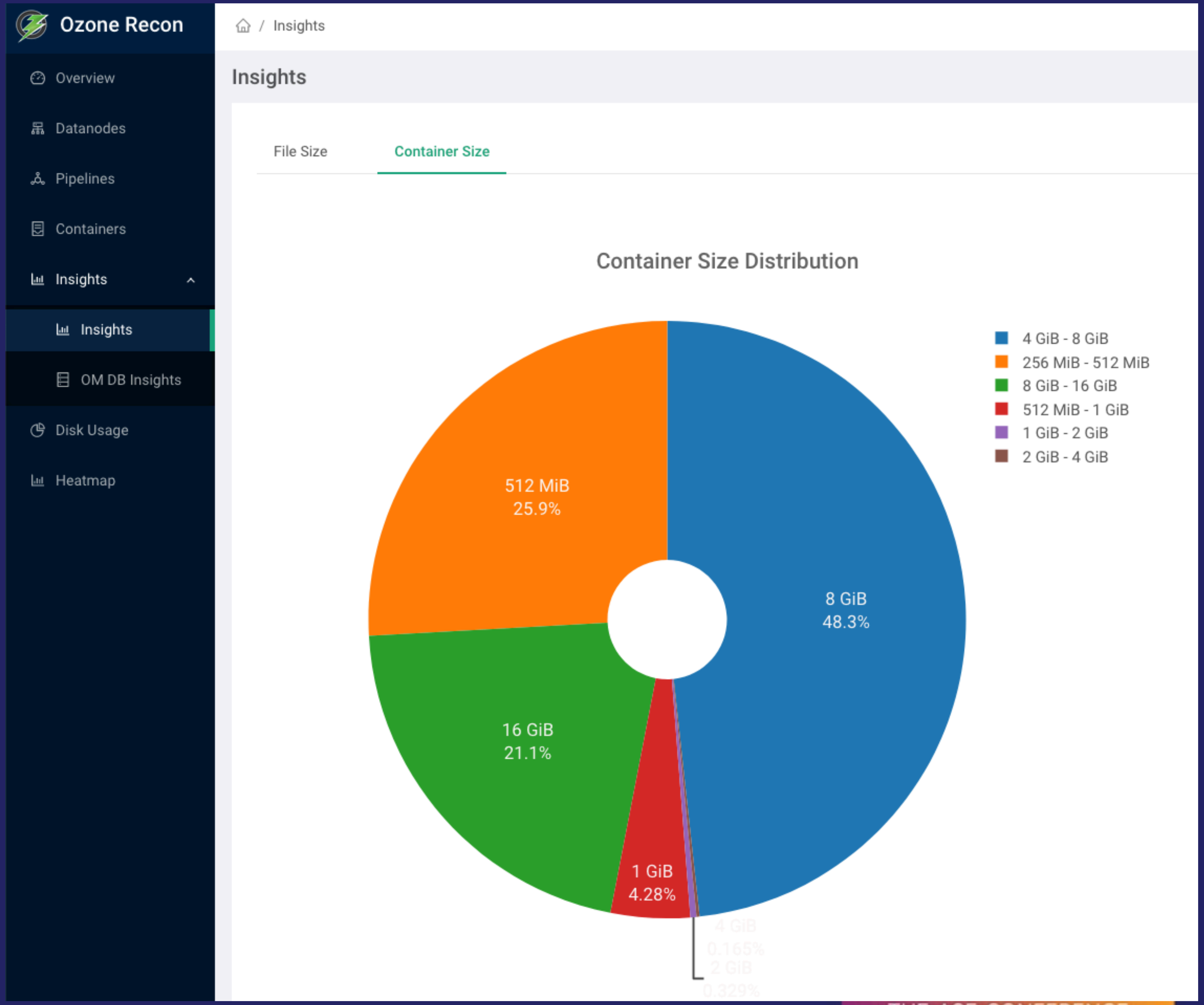
Performance after change bytes.per.checksum

Major JIRAs

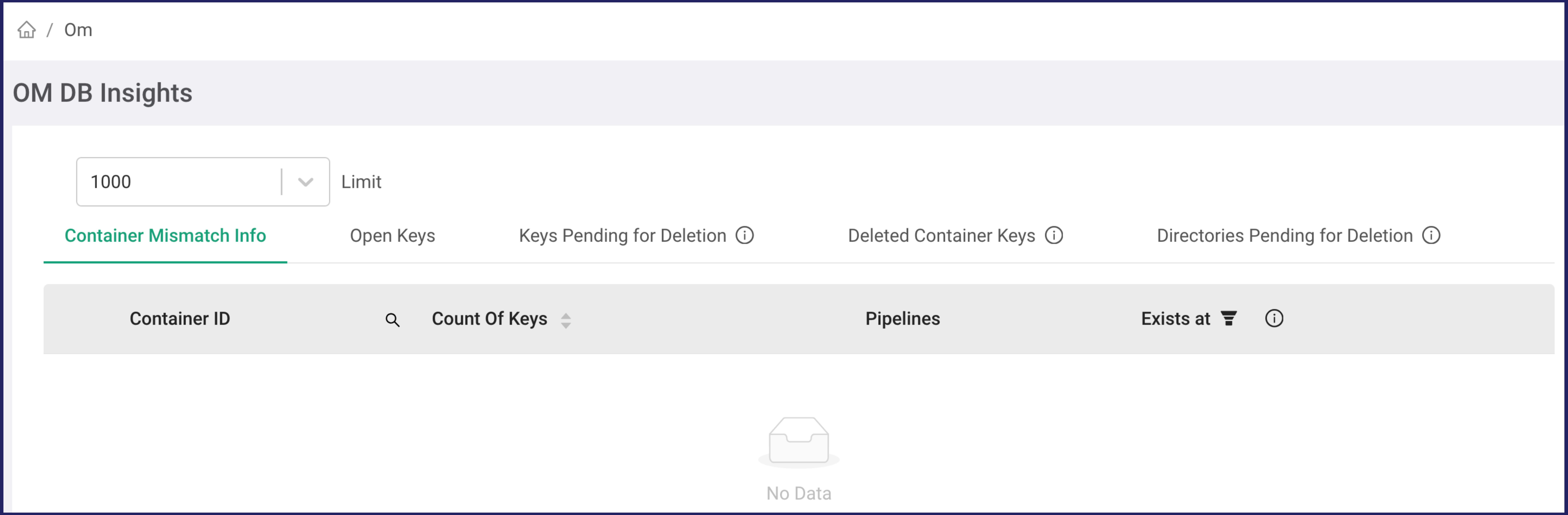
- ✓ HBASE-27740, Support Ozone as a WAL backing storage
- ✓ HDDS-7593, Supporting HSync and lease recovery
- ✓ HDDS-10685, Short circuit read support in Ozone
- ✓ HDDS-8047, Incremental ChunkList in PutBlock
- ✓ HDDS-10442, Add a Freon tool to measure client to DataNode round-trip latency
- ✓ HDDS-10820, Freon tool DN-Echo to test GRPC and Ratis read/write mode performance
- ✓ HDDS-8830, Add admin CLI to list open files
- ✓ HDDS-9365, DataNode to deserialize Ratis transaction only once
- ✓ HDDS-9387, Reduce updating block length times at OM during hsync
- ✓ HDDS-10361, Output stream should support direct byte buffer
- ✓ HDDS-10511, OzoneFSInputStream to support ByteBufferPositionedReadable
- ✓ HDDS-9918, Remove block token from Ratis log once verified
- ✓ HDDS-10890, Increase default value for hdds.container.ratis.log.appender.queue.num-elements
- ✓ HDDS-9842, Checking disk capacity at every write request is expensive for HBase
- ✓ HDDS-9844, De-synchronize hsync API

Recon New Functions





Recon New Functions



⌂ / Om

OM DB Insights

1000 ⌄ Limit

Container Mismatch Info
Open Keys
Keys Pending for Deletion ⓘ
Deleted Container Keys ⓘ
Directories Pending for Deletion ⓘ

Container ID	Count Of Keys	Pipelines	Exists at ⓘ

No Data

Data Tiering (HDDS-10656 Atomic Key Overwrite and Key Replacement)

Potential Usages

✓ Bi-direction conversion of 3 replica with erasure coding format

✓ Compaction of small containers

✓ Storage polices

Information

- ✓ Web site, https://ozone.apache.org
- ✓ Github repo, https://github.com/apache/ozone/
- ✓ Community discussions, https://github.com/apache/ozone/discussions
- ✓ US and APAC Community meetings, https://cwiki.apache.org/confluence/display/OZONE/Ozone+Community+Calls
- ✓ WeChat group “Ozone 技术交流群”

Thanks