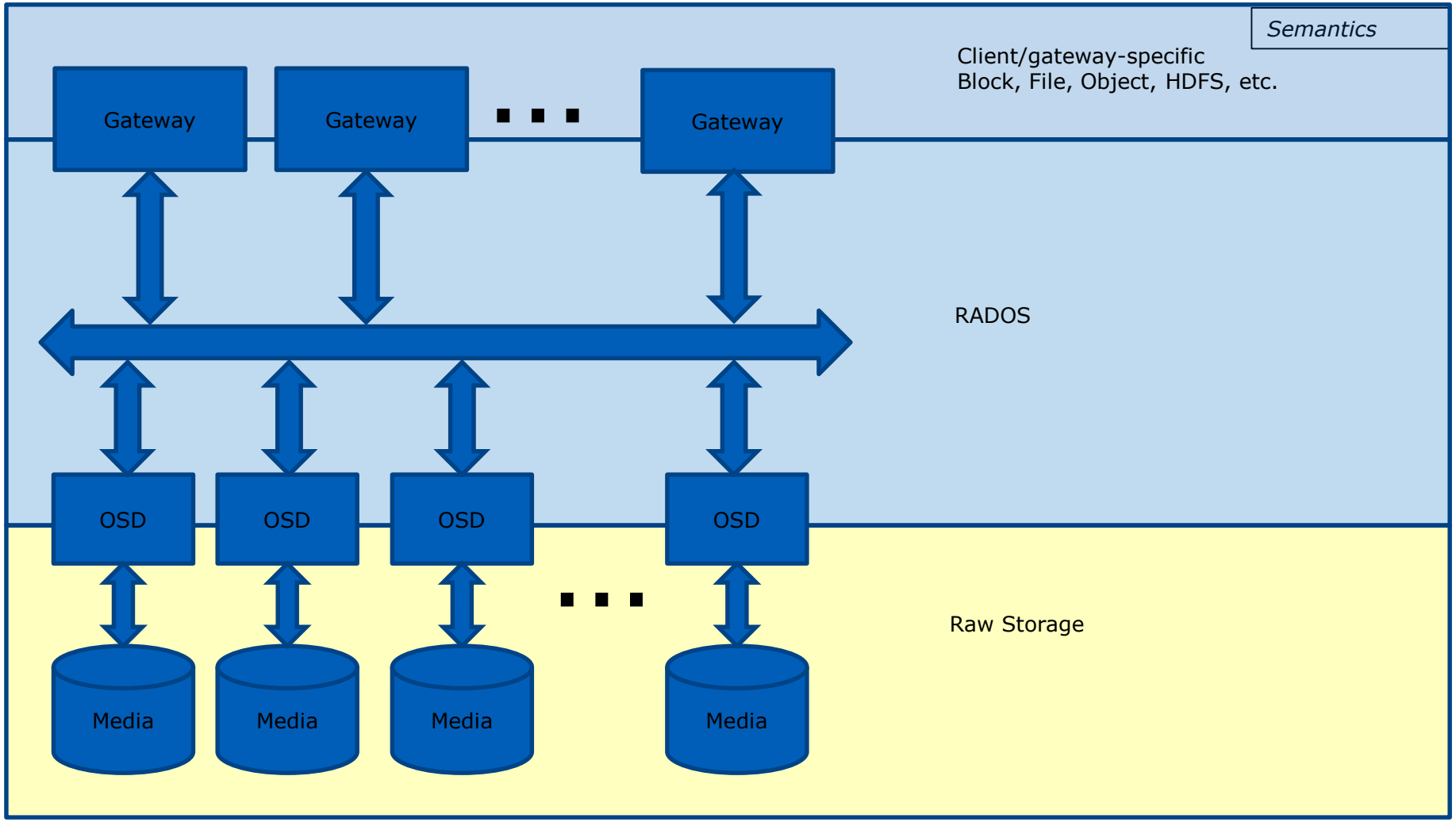
Western Digital.

Bluestore: A new storage engine for Ceph

Allen Samuels, Engineering Fellow

March 4, 2017

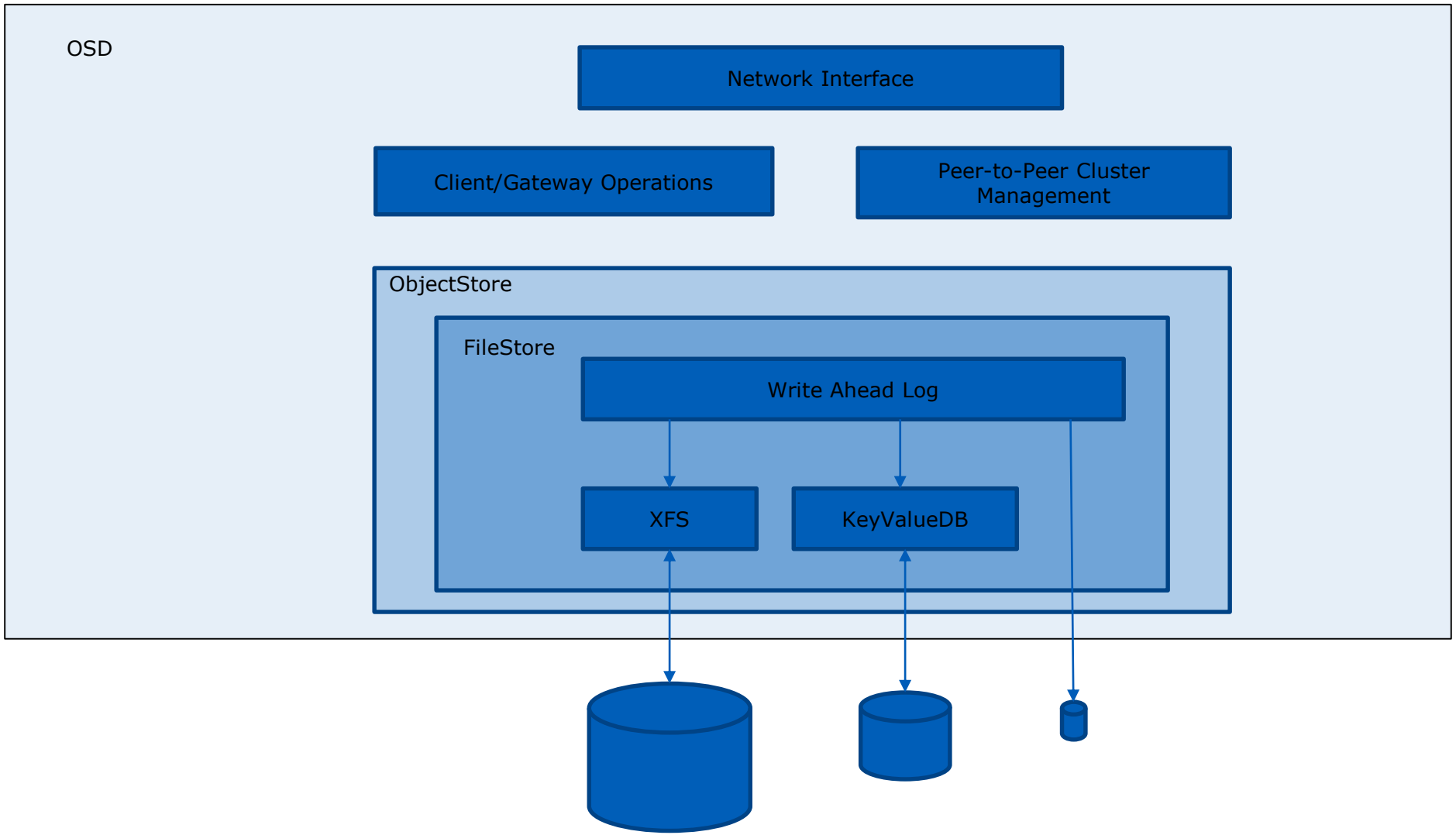
Conceptual Ceph System Model



Terminology

- Gateway – implements client protocol using RADOS
  - LibRBD, KRBD, RGW, CephFS, etc.
- RADOS – cluster-wide storage protocol
  - Transactional, Durable and Available storage
- OSD – Object Storage Daemon
  - Raw object storage for RADOS
  - Limited durability and availability

Inside the OSD



Ceph Deployment Options

- Ceph Journal on Flash
- Ceph Metadata on Flash
- All Flash

Ceph Journal on Flash

- Journal consumes only a tiny fraction of one SSD
  - Constrained by spills to HDD through XFS
    - Average SSD BW is much less than 100 MB/Sec
    - Space consumption is much less than < 10GB
- Typical usage aggregates multiple OSDs / SSD
  - Partitioning of SSD is straightforward
  - New failure domain affects durability
  - Resource planning is simple

SSD Provisioning/Selection

- Multiplexing OSDs means random writes for SSD
  - Journal write size is 4K + RADOS transaction size
  - Overall rate still limited by background destage to HDD
- Right-size the SSD logs
  - ~1 minute of max throughput is only 6-8GB
  - Small log wraparound is implicit “trim”
  - SSD Garbage collection is minimized
- Should see best-case endurance for SSD
  - Minimal write amplification due to garbage collection

Ceph Metadata on Flash

- Not much value for RBD
  - Ceph xattrs generally stored in inode
- Will improve Object (S3/Swift) throughput
  - But still have XFS metadata on HDD
  - Difficult to estimate improvement
- Provisioning harder to estimate
  - Bucket sharding can help with space allocation

Optimizing Ceph for the future

- With the vision of an all flash system, SanDisk engaged with the Ceph community in 2013
- Self-limited to no wire or storage format changes
- Result: Jewel release is up to 15x vs. Dumpling
  - Read IOPS are decent, Write IOPS still suffering
- Further improvements require breaking storage format compatibility

What's wrong with FileStore?

- Metadata split into two disjoint environments
  - Ugly logging required to meet transactional semantics
- Posix directories are poor indexes for objects
- Missing virtual copy and merge semantics
  - Virtual copies become actual copies
- BTRFS hope didn't pan out
  - Snapshot/rollback overhead too expensive for frequent use
  - Transaction semantics aren't crash proof

What's wrong with FileStore?

- Bad Write amplification
  - Write ahead logging for everything
  - levelDB (LSM)
  - Journal on Journal
- Bad jitter due to unpredictable file system flushing
  - Binge/purge cycle is very difficult to ameliorate
- Bad CPU utilization
  - syncfs is VERY expensive

BlueStore a rethink of ObjectStore

- Original implementation written by Sage late in ‘15
- Tech preview available in Jewel Release
- First full release in Kraken Release
- Preserves wire compatibility
- Storage Format incompatible
- Target Write performance ≥ 2x FileStore
- Target Read performance ≥ FileStore

BlueStore a rethink of ObjectStore

- Efficiently Support current and future HW types
  - SCM, Flash, PMR and SMR hard drives, standalone or hybrid combinations
- Improve performance
  - Eliminate double write when unneeded
  - Better CPU utilization through simplified structure and tailored algorithms
- Much better code stability

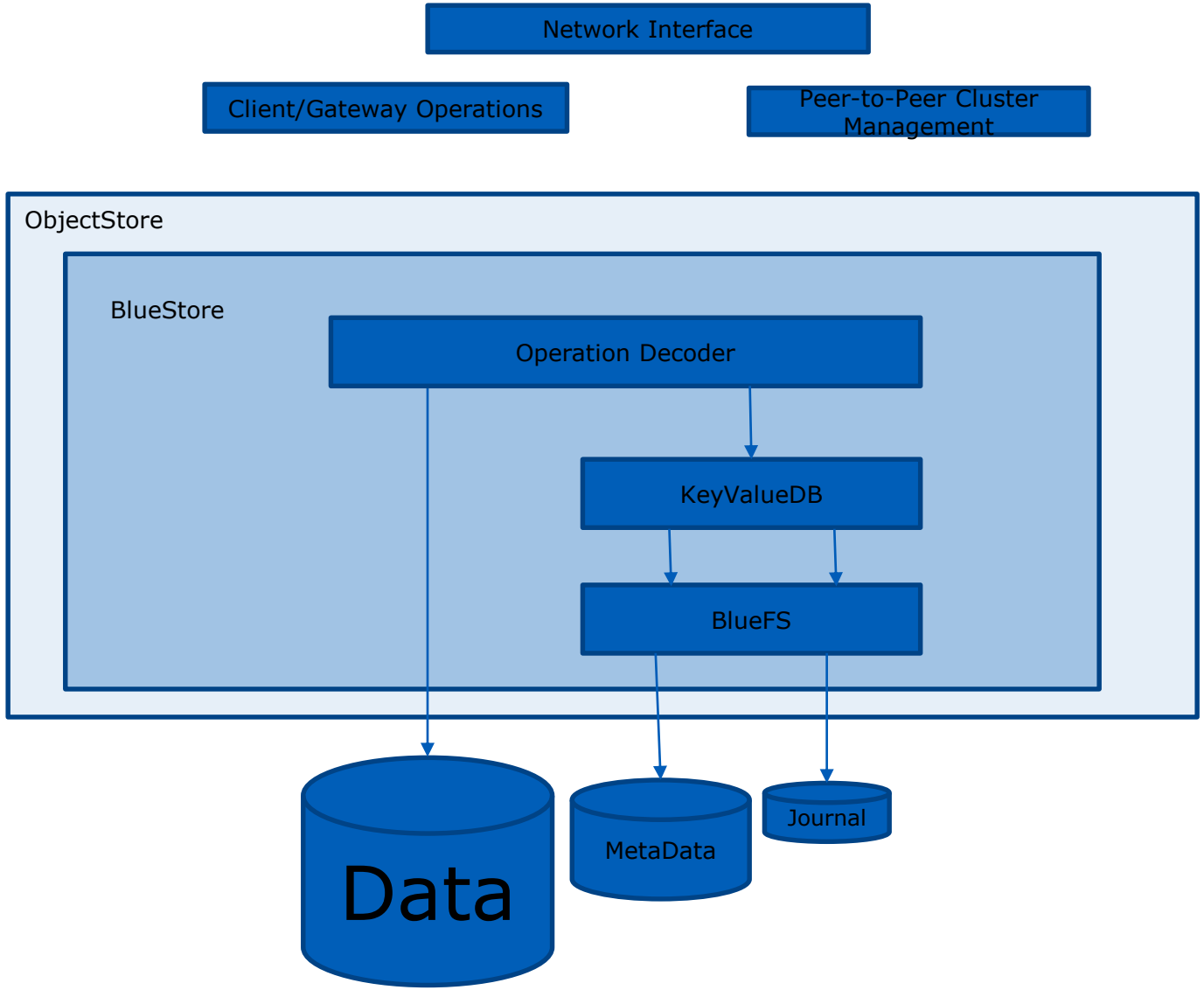
BlueStore a rethink of ObjectStore

- Wire compatible but not data format compatible
  - Mixed FileStore/Bluestore nodes in a cluster transparently supported
  - FileStore continues for legacy systems
  - In place upgrade/conversion supported via node rebuild

BlueStore Enhanced Functionality

- Checksum on all read operations
  - SW defined data integrity
- Inline Compression
  - Pluggable, Snappy and Zlib initially
- Virtual clone
  - Efficient implementation of snapshots and clones
- Virtual move
  - Enables RBD/CephFS to directly use erasure coded pools

BlueStore


Network Interface
Client/Gateway Operations
Peer-to-Peer Cluster Management
ObjectStore
BlueStore
Operation Decoder
KeyValueDB
BlueFS
Data
MetaData
Journal

BlueStore Architecture

- One, Two or Three raw block devices
  - Data, Metadata/WAL and KV Journaling
  - When combined no fixed partitioning is needed
- Use a single transactional KV store for all metadata
  - Semantics are well matched to ObjectStore transactions
- Use raw block device for data storage
  - Support Flash, PMR and SMR HDD

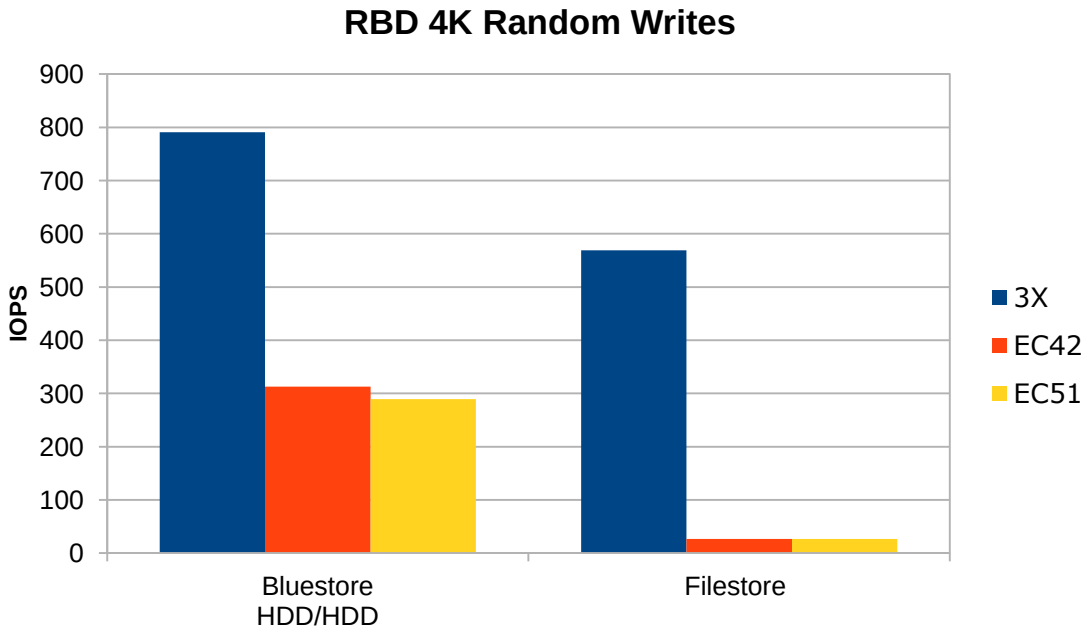
Two Write Path Options

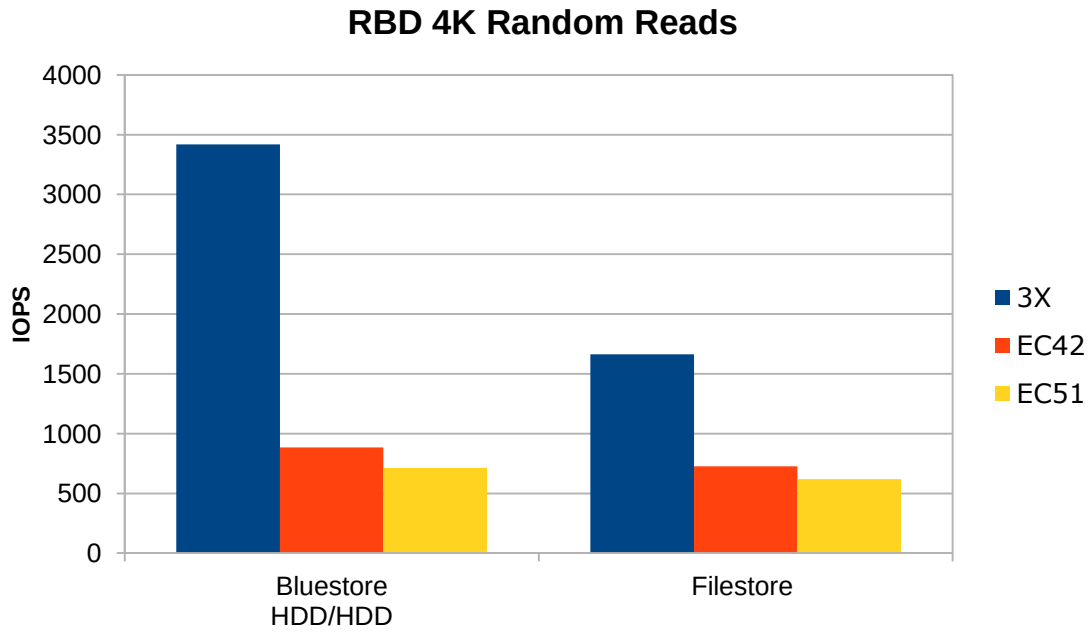
- Direct Write
  - (1) Write data to unused space (Copy-on-write style)
    - Trivially crash-proof
  - (2) Modify metadata through single KV transaction
    - Transaction semantics of KV store shine here!
  - (3) Send client completion signal

Two Write Path Options

- Write-ahead Log (WAL)
  - (1) Commit data and metadata into single KV transaction
  - (2) Send client completion signal
  - <later>
  - (3) Move data from KV into destination (Idempotent and crash restartable)
  - (4) Update KV to remove data and WAL operation

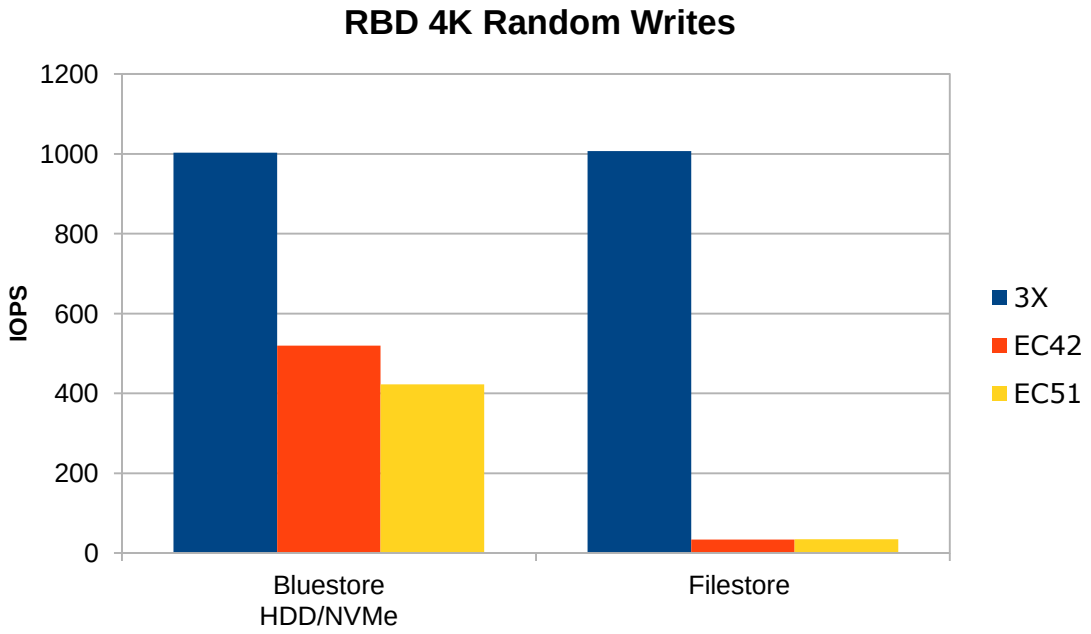
BlueStore vs FileStore (HDD)

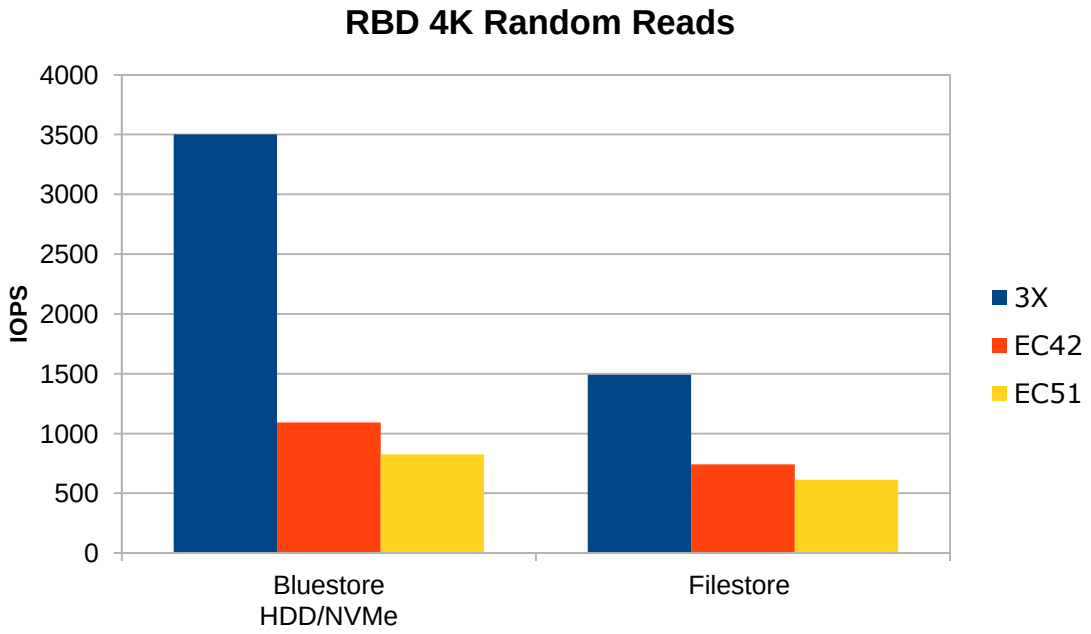




* Mark Nelson (RedHat) email 3-3-17, Master, 4 nodes of: 2xE5-2650v3, 64GB, 40GbE, 4xP3700, 8x1TB Constellation ES.2

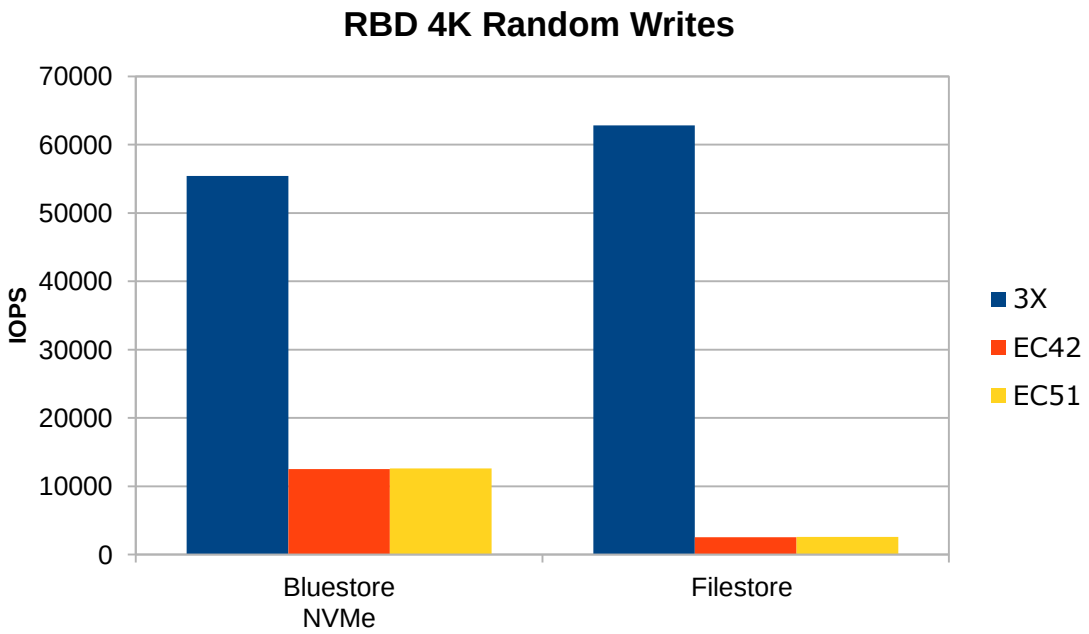
BlueStore vs FileStore (Hybrid)

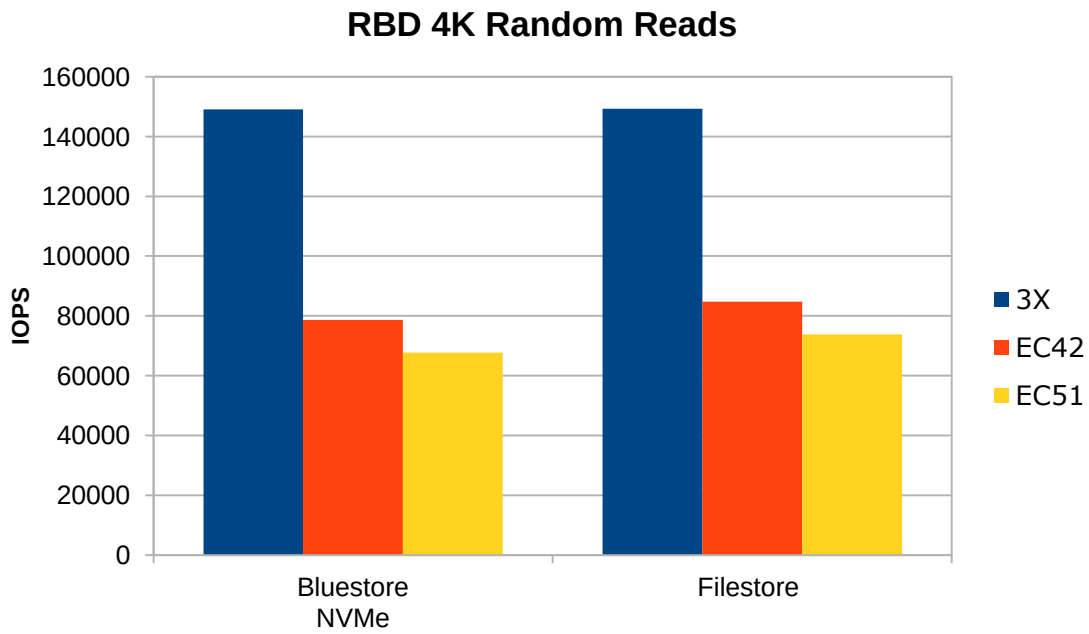




* Mark Nelson (RedHat) email 3-3-17, Master, 4 nodes of: 2xE5-2650v3, 64GB, 40GbE, 4xP3700, 8x1TB Constellation ES.2

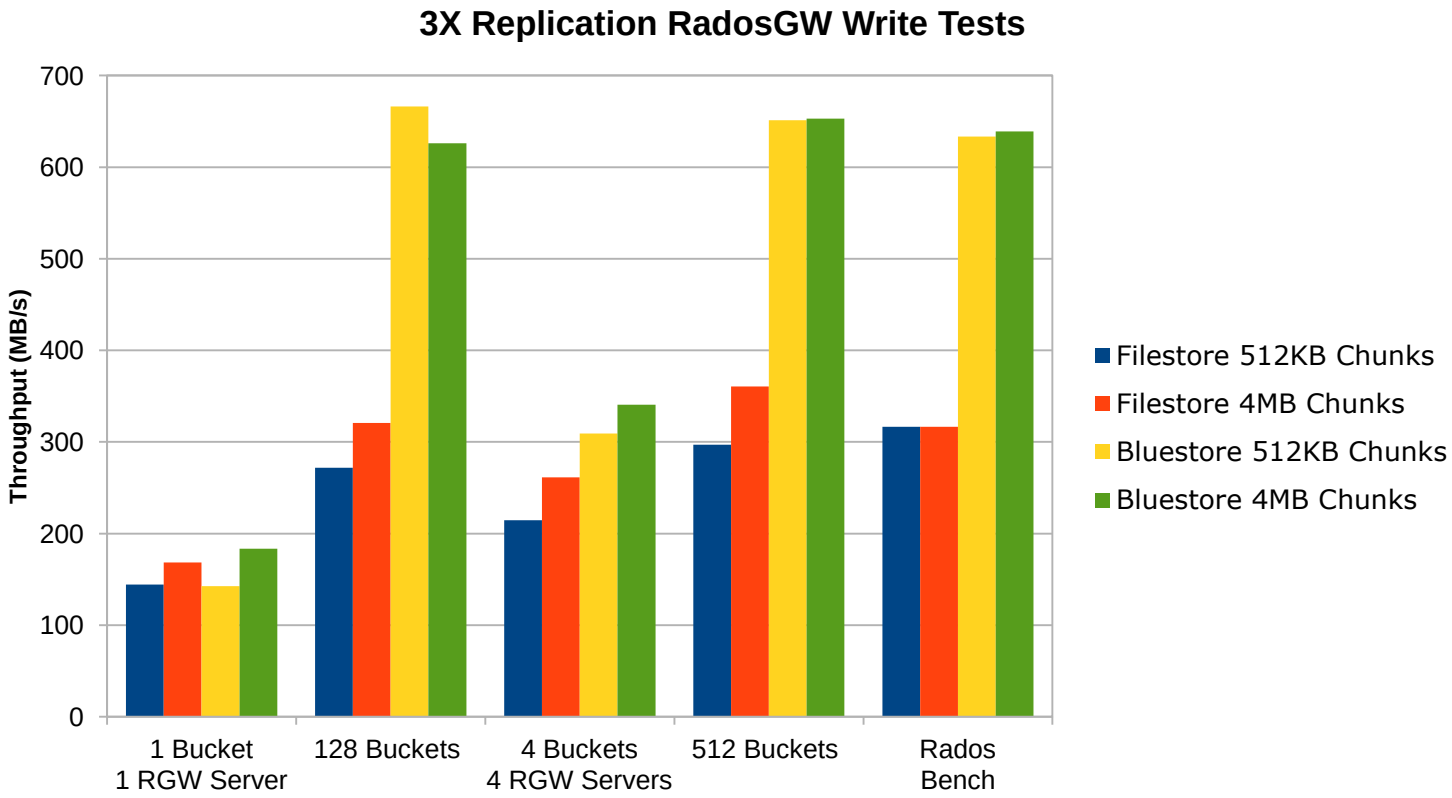
BlueStore vs FileStore (Flash)





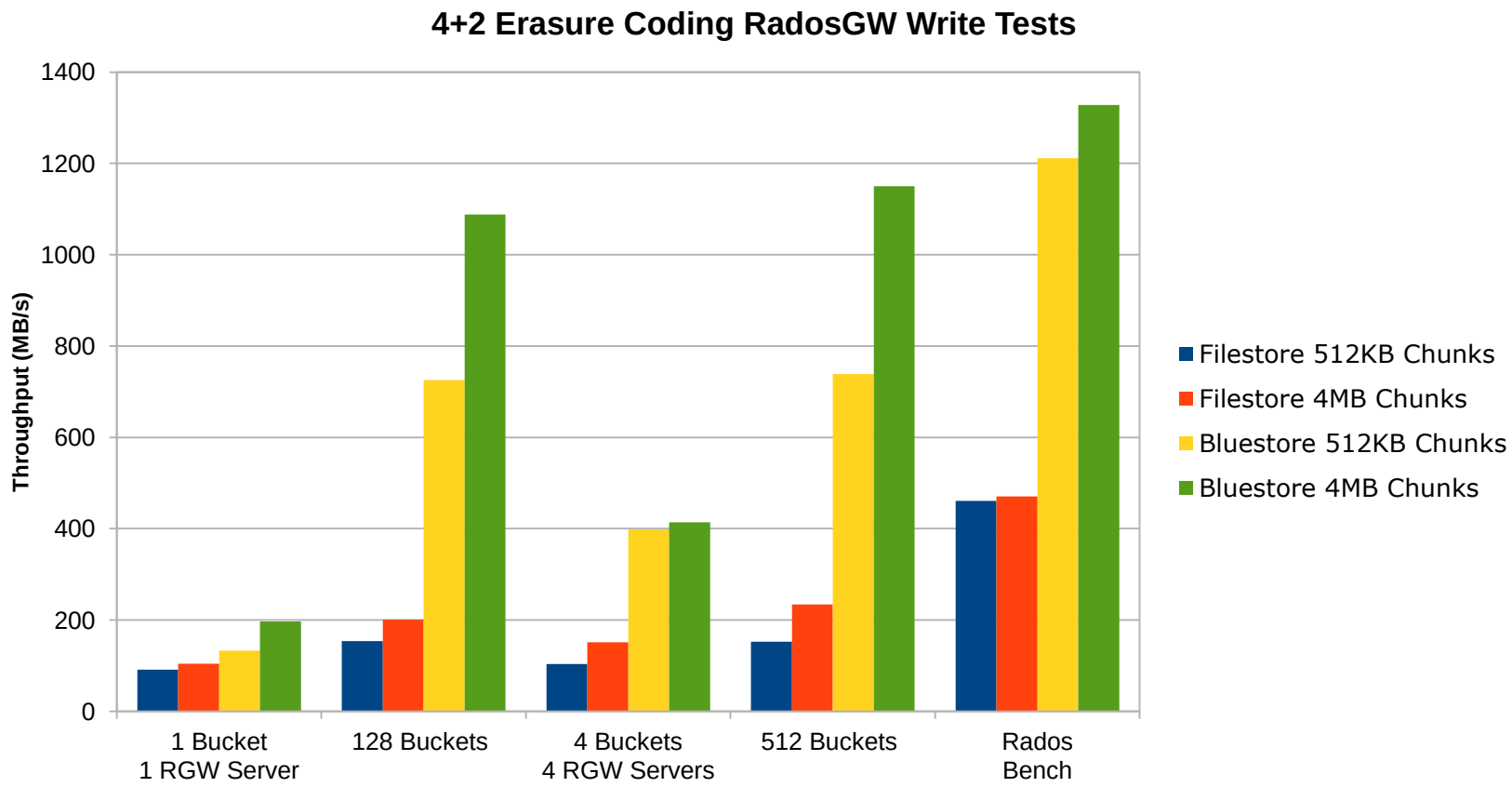
* Mark Nelson (RedHat) email 3-3-17, Master, 4 nodes of: 2xE5-2650v3, 64GB, 40GbE, 4xP3700, 8x1TB Constellation ES.2

BlueStore vs FileStore



* Mark Nelson (RedHat) email 3-3-17, Master, 4 nodes of: 2xE5-2650v3, 64GB, 40GbE, 4xP3700, 8x1TB Constellation ES.2

BlueStore vs FileStore

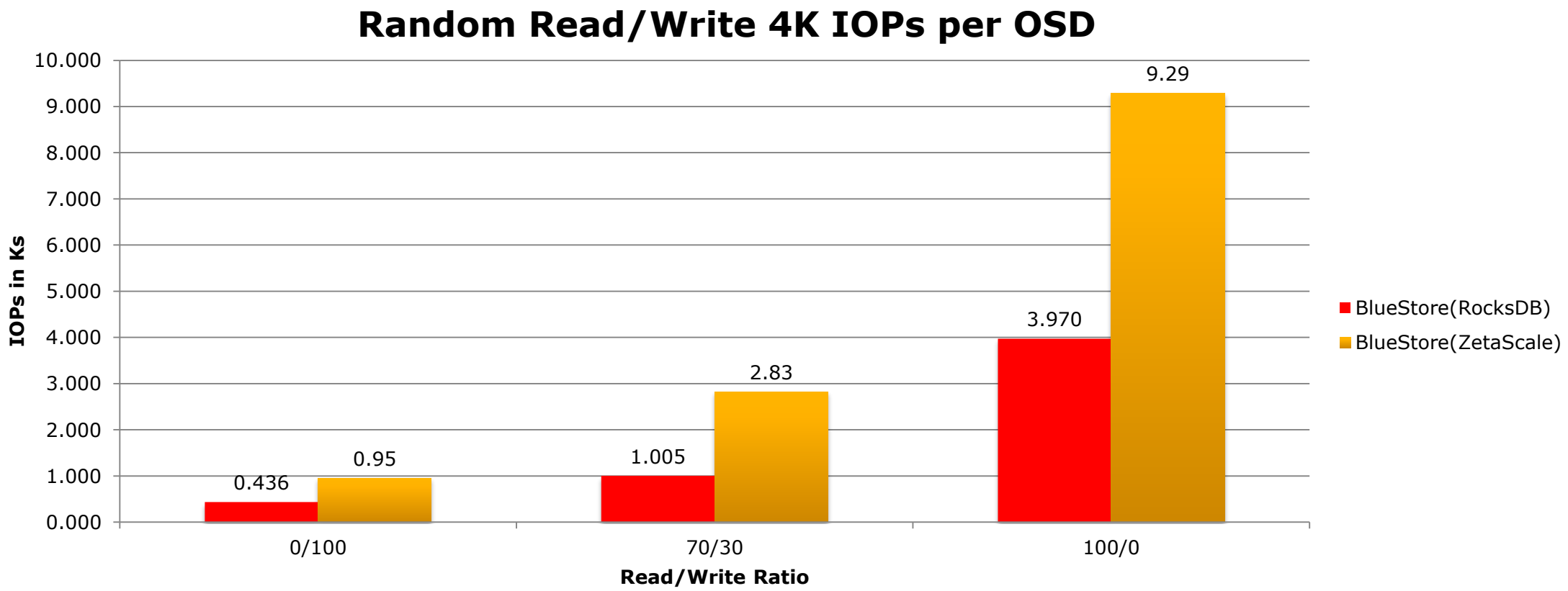


* Mark Nelson (RedHat) email 3-3-17, Master, 4 nodes of: 2xE5-2650v3, 64GB, 40GbE, 4xP3700, 8x1TB Constellation ES.2

KV Store Options

- RocksDB is a Facebook extension of levelDB
  - Log Structured Merge (LSM) based
  - Ideal when metadata is on HDD
  - Merge is effectively host-based GC when run on flash
- ZetaScale™ from SanDisk® now open sourced
  - B-tree based
  - Ideal when metadata is on Flash
  - Uses device-based GC for max performance

BlueStore ZetaScale v RocksDB Performance



Test Setup:
1 OSD, 8TB SAS SSD, 10GB ram, Intel(R) Xeon(R) CPU E5-2680 v2 @ 2.80GHz , fio, 32 thds, 64 iodepth, 6TB dataset, 30 min

BlueStore Status

- Exceeding design goal of 2x write performance
- Available in Kraken Release
  - Ready for experimentation
  - Not ready for production
- Targeted as default in Luminous Release
  - FileStore NOT going away