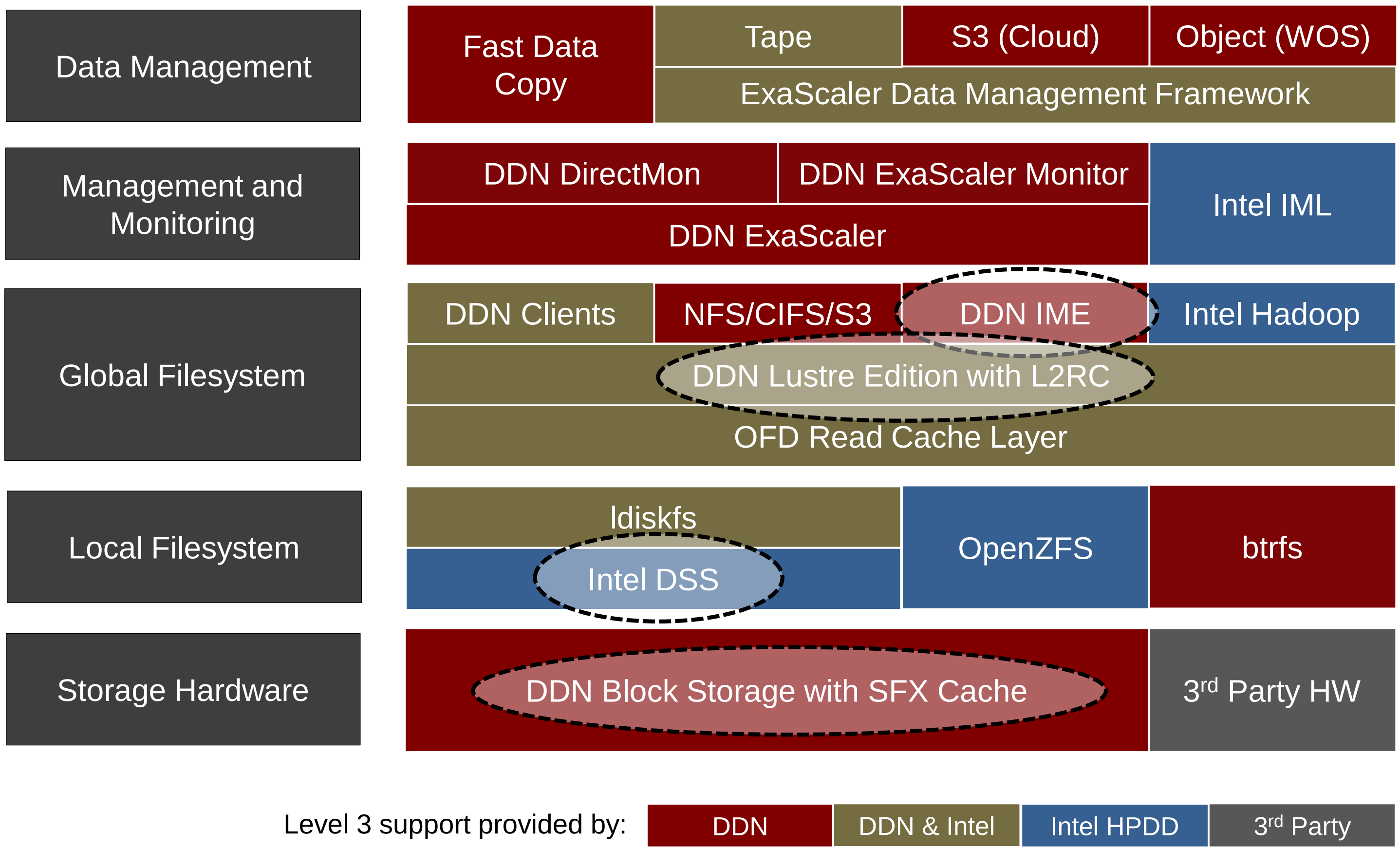
DDN

# Accelerating Lustre with SSDs and NVMe

James Coomer, DDN

# DDN ExaScaler Software Components



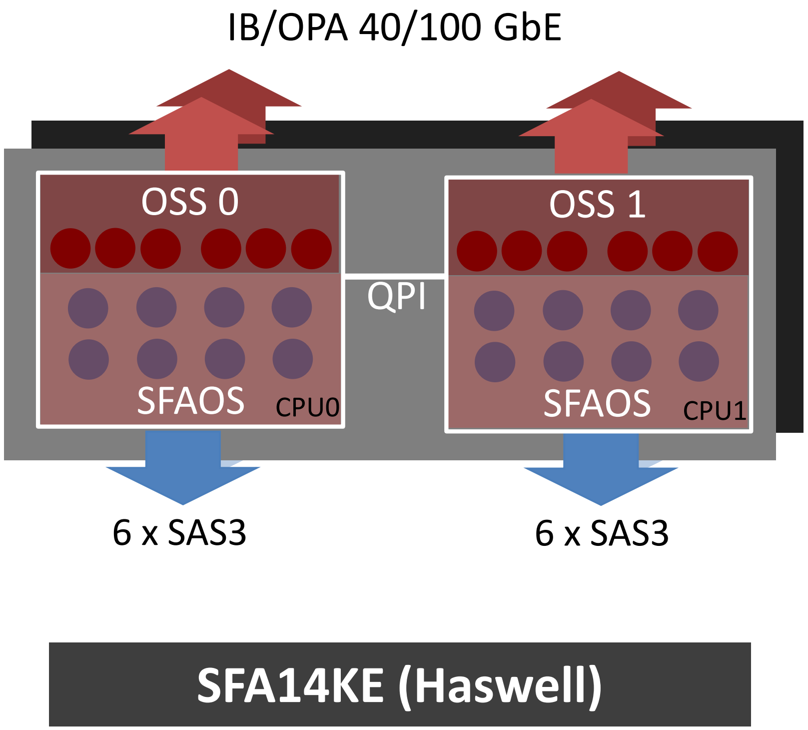
# DDN | ES14K
## Designed for Flash and NVMe

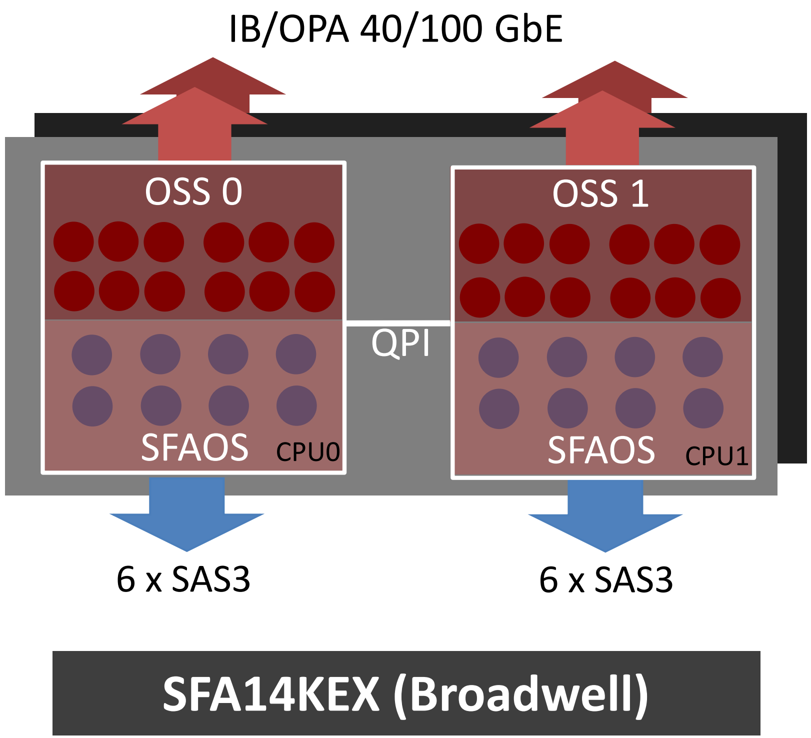
**Configuration Options**
- 72 SAS SSD or 48 NVMe
- SSDs or HDDs only
- HDDs with SSD caching
- SSDs with HDD tier

**Connectivity**
- FDR/EDR
- OmniPath
- 40/100GbE

**Industry Leading Performance in 4U**
- Up to 40 GB/sec throughput
- Up to 6 million IOPS to cache
- Up to 3.5 million IOPS to storage
- 1PB+ capacity (with 16TB SSD)
- 100 millisecond latency

# ES14K Architecture



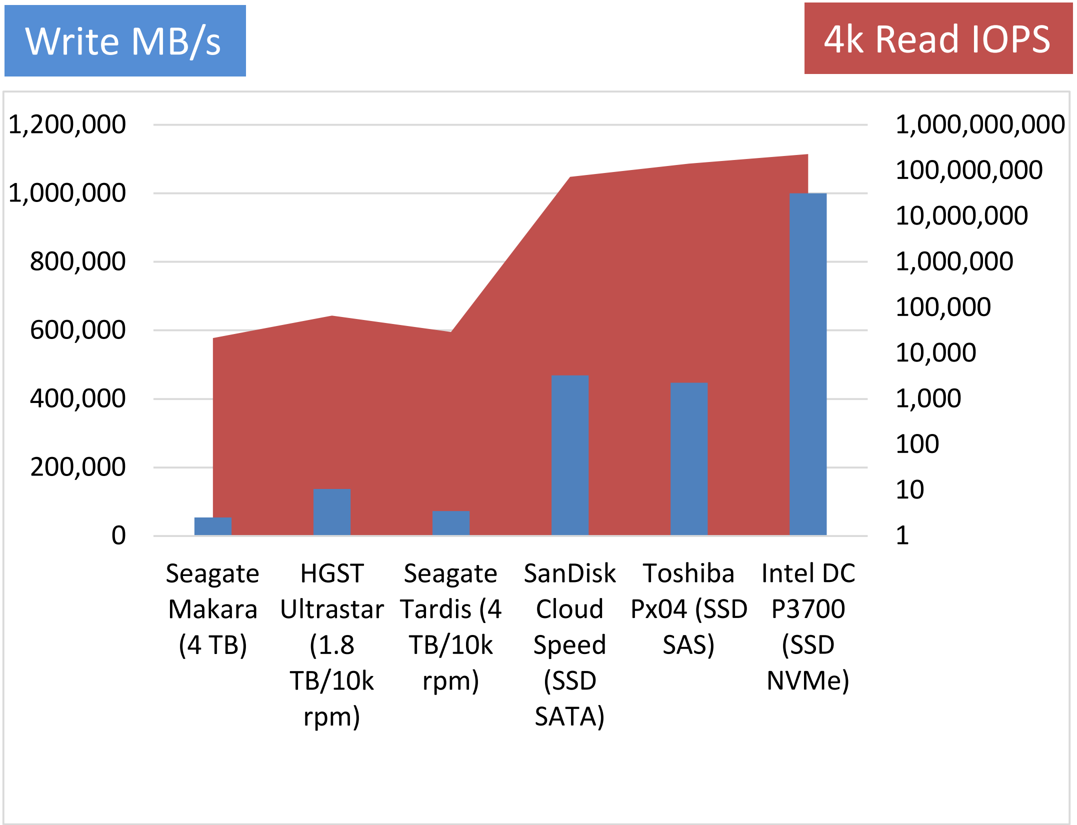



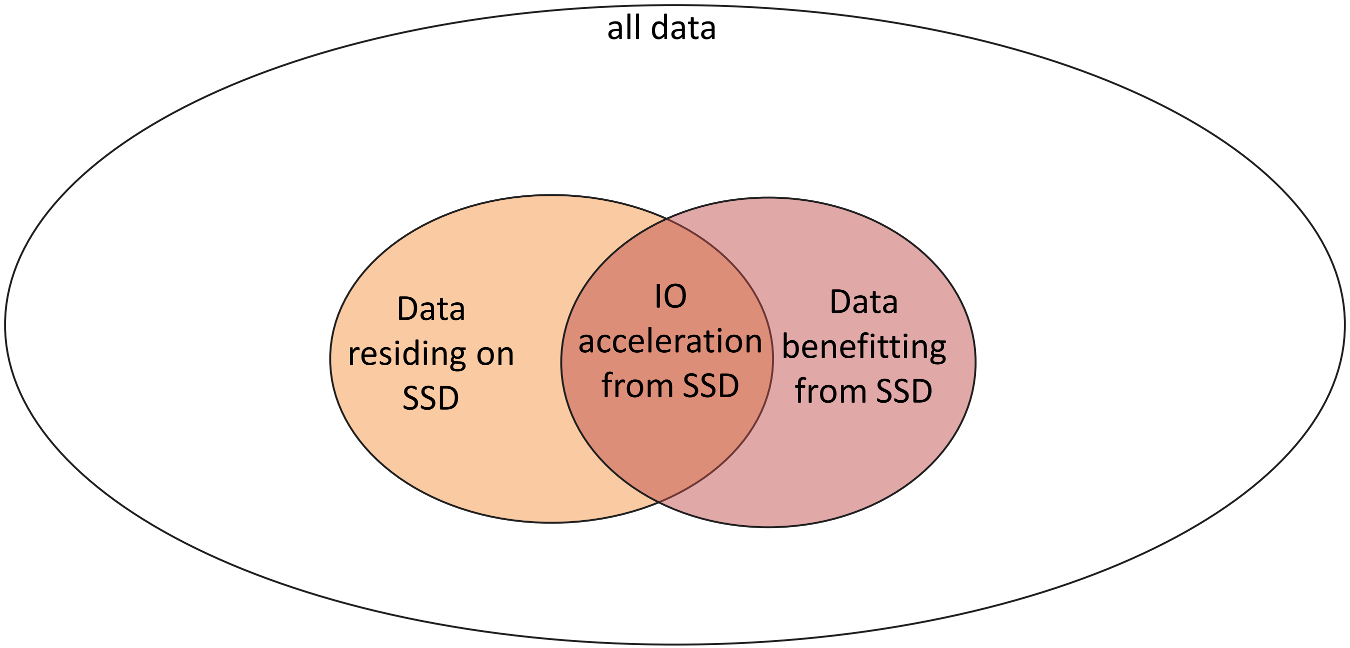
# Why SSD Cache?

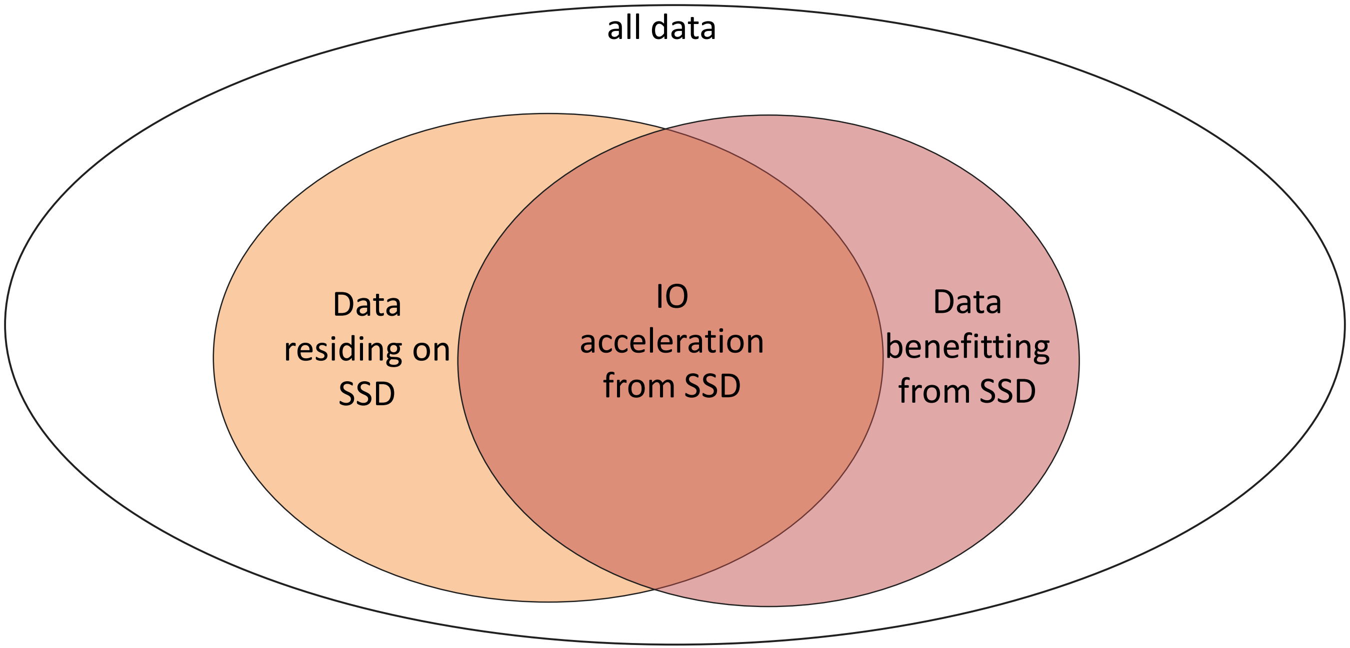
Don't blow the power/space/management with spindles

SSDs still pricey... So

- ▶ Optimise Data for SSDs
- ▶ Optimise SSDs for Data

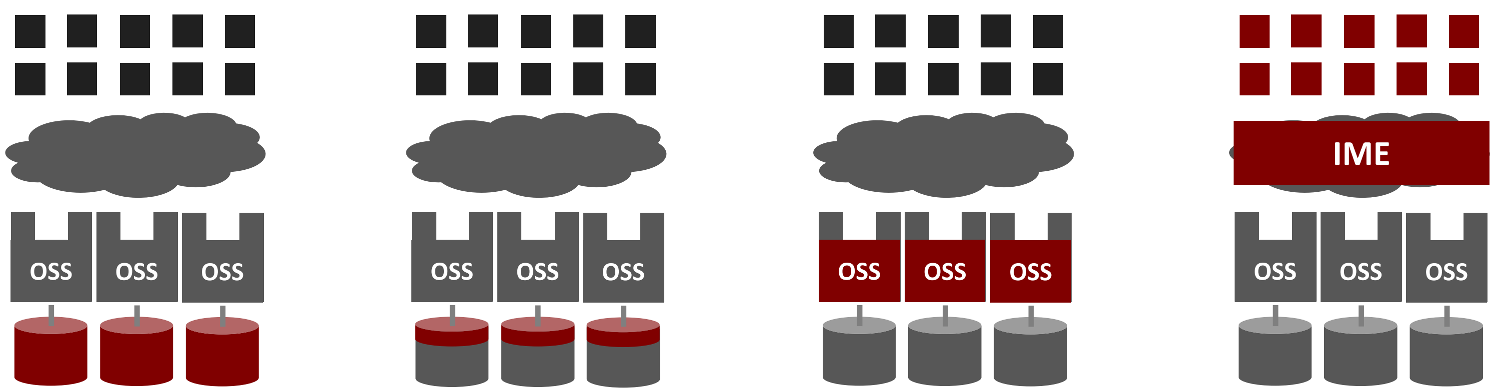



all data
Data residing on SSD
IO acceleration from SSD
Data benefitting from SSD

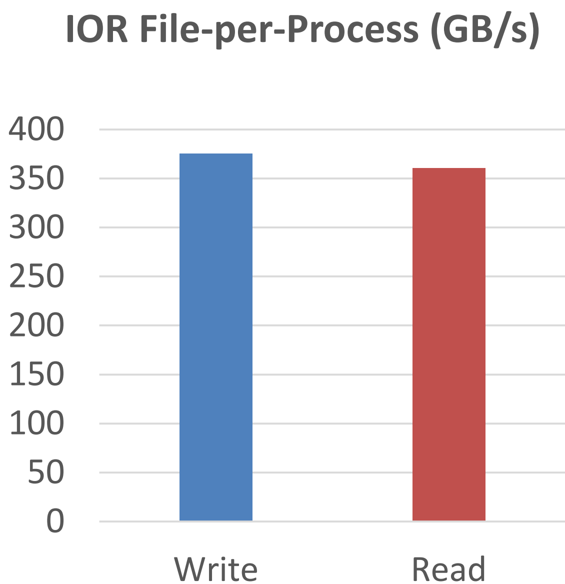

all data
Data
residing on
SSD
IO
acceleration
from SSD
Data
benefitting
from SSD

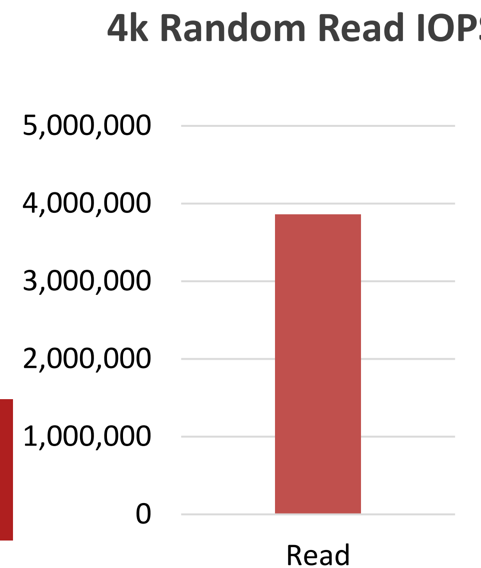
# SSD Options

| 1 | 2 | 3 | 4 |
|---|---|---|---|
| **All SSD Lustre** | **SFX** | **L2RC** | **IME** |
| Lustre HSM for Data Tiering to HDD namespace | Block-Level Read Cache | OSS-level Read cache | I/O level Write and Read Cache |
| Generic Lustre I/O | Instant Commit, DSS, fadvice() | Heuristics with **FileHeat** | Transparent + hints |
| Millions of Read and Write IOPS | Millions of Read IOPS | Millions of Read IOPS | 10s of Millions of Read/Write IOPS |



# 1. Rack Performance: Lustre



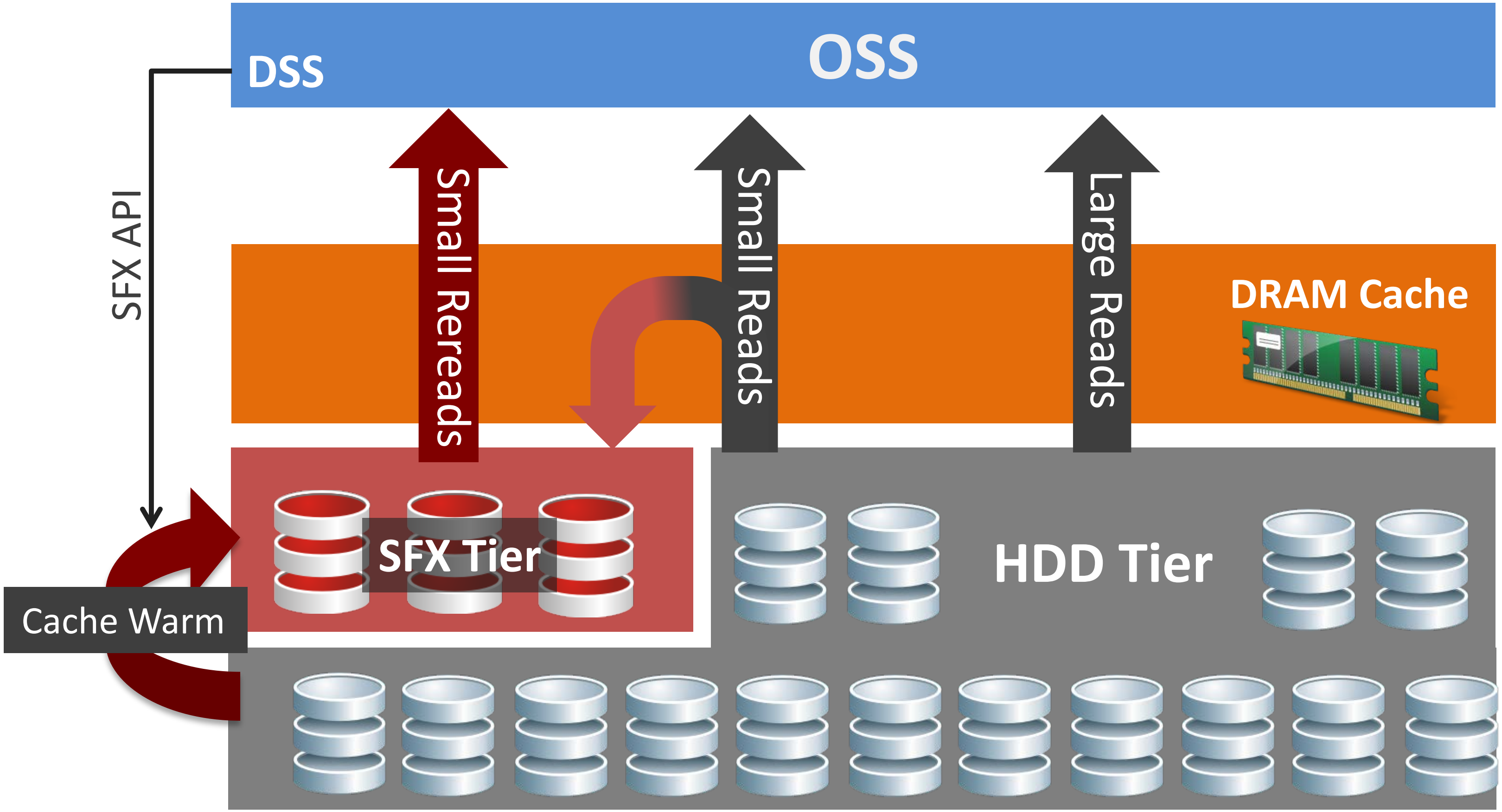

up to 4PB Flash Capacity

4 Million IOPs

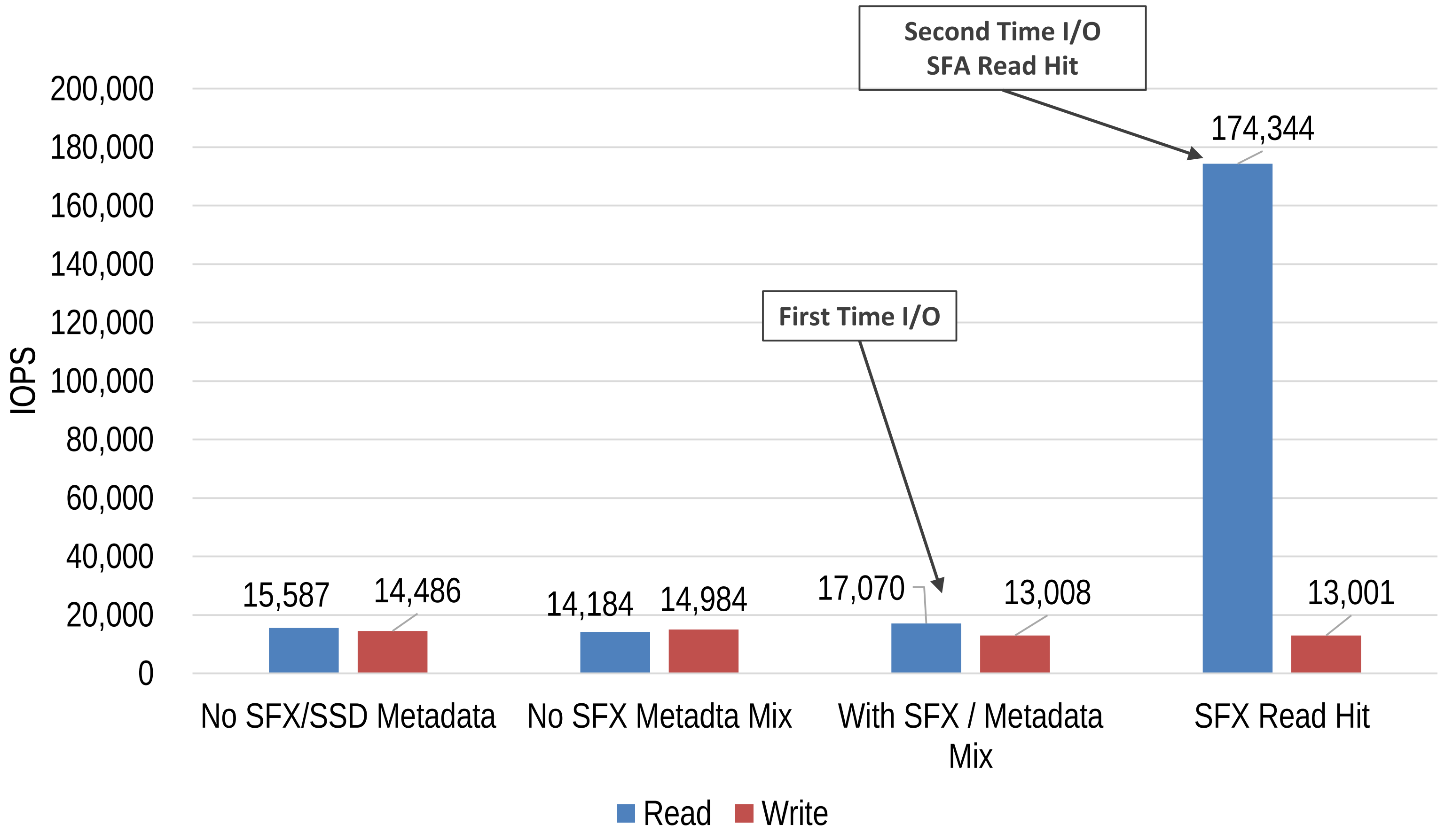
350GB/s Read and Write (IOR)

# 2. SFX & ReACT – Accelerating Reads

Integrated with Lustre DSS



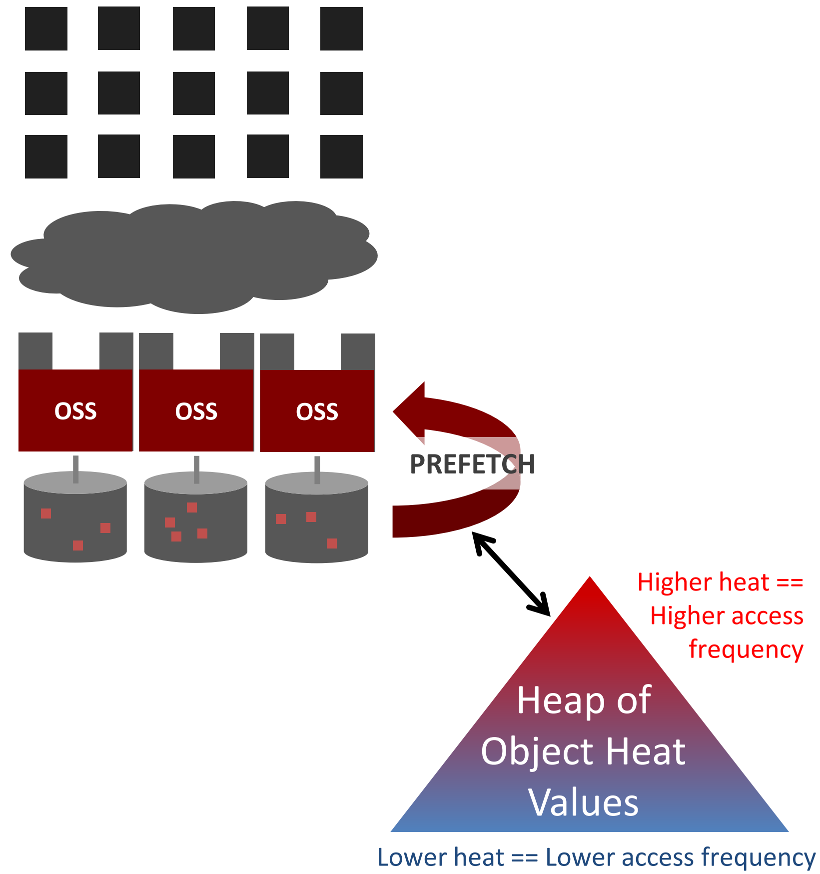
# 2. 4 KiB Random I/O



# 3. Lustre L2RC and File Heat

OSS-based Read Caching

- Uses SSDs (or SFA SSD pools) on the OSS as read cache
- Automatic prefetch management based on file heat
- File-heat is a relative (tunable) attribute that reflects file access frequency
- Indexes are kept in memory (worst case is 1 TB SSD for 10 GB memory)
- Efficient space management for the SSD cache space (4KB-1 MB extends)
- Full support for ladvice in Lustre

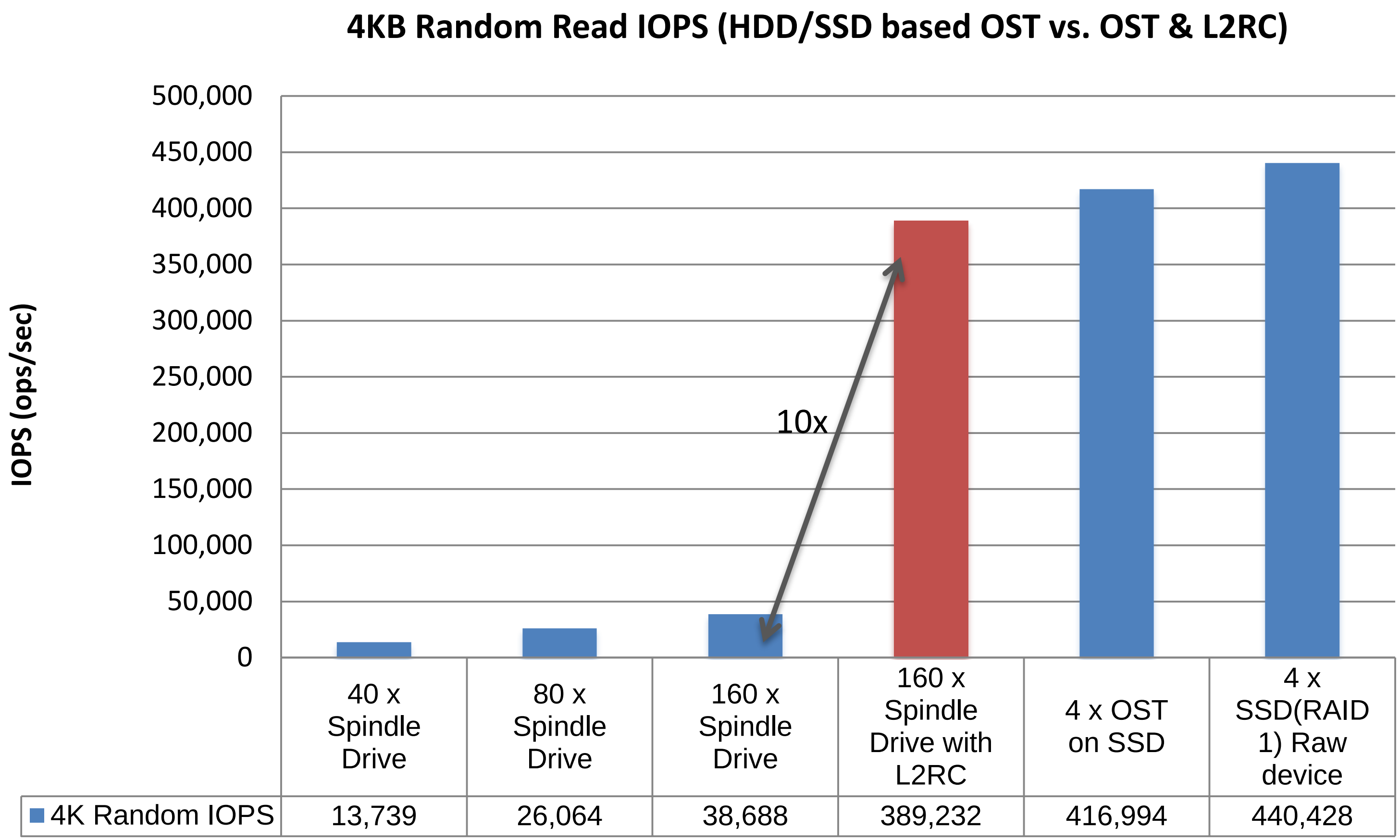


# 3. File Heat Utility

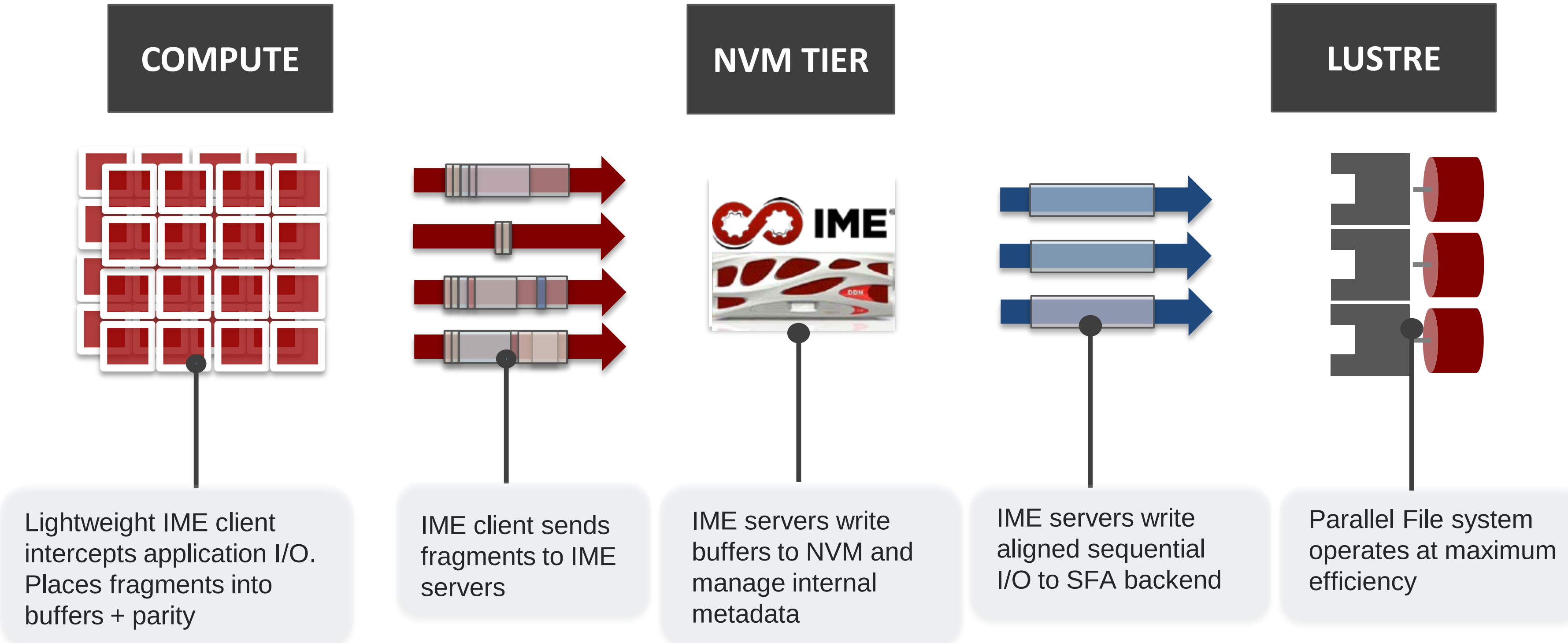
- tune the arguments of file heat with proc interfaces
  ```
  /proc/fs/lustre/heat_period_second
  /proc/fs/lustre/heat_replacement_percentage
  ```
- Utils to get file heat values: `lfs heat_get <file>`
- Utils to set flags for file heat:
  ```
  lfs heat_set [--clear|-c] [--off|-o] [--on|-O] <file>
  ```
- Heat can be cleared by: `lfs heat_set --clear`
- Heat accounting of a file can be turned off by: `lfs heat_set --off`
- Heaps on OSTs which can be used to dump lists of FIDs sorted by heat:
  ```
  [root@server9-Centos6-vm01 cache]# cat
  /proc/fs/lustre/obdfilter/lustre-OST0000/heat_top
  [0x200000400:0x1:0x0] [0x100000000:0x2:0x0]: 0 740 0 775946240
  [0x200000400:0x9:0x0] [0x100000000:0x6:0x0]: 0 300 0 314572800
  [0x200000400:0x8:0x0] [0x100000000:0x5:0x0]: 0 199 0 208666624
  [0x200000400:0x7:0x0] [0x100000000:0x4:0x0]: 0 100 0 104857600
  [0x200000400:0x6:0x0] [0x100000000:0x3:0x0]: 0 100 0 104857600
  ```

# 3. Random Read Performance with L2RC

DDN



# DDN | IME
## Application I/O Workflow



# 4. IME Write Dataflow

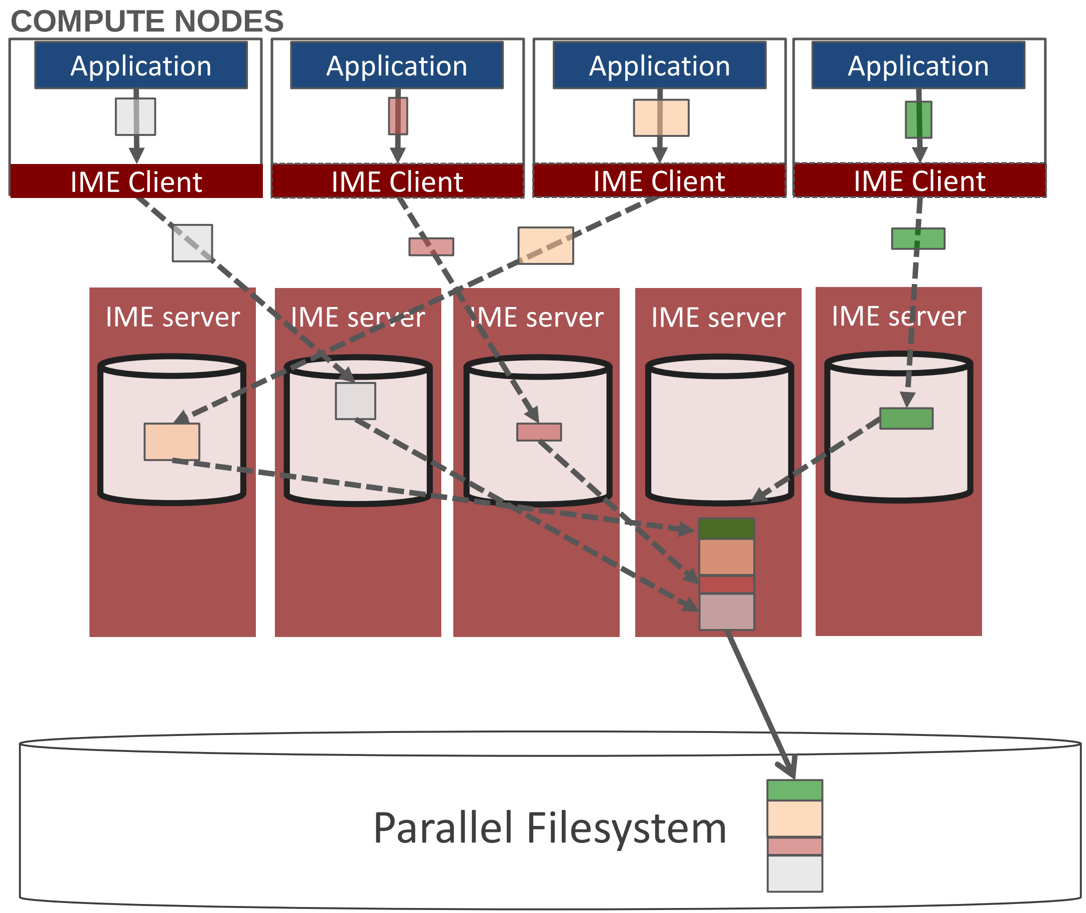
1. Application issues fragmented, misaligned IO

2. IME clients send **fragments** to IME servers

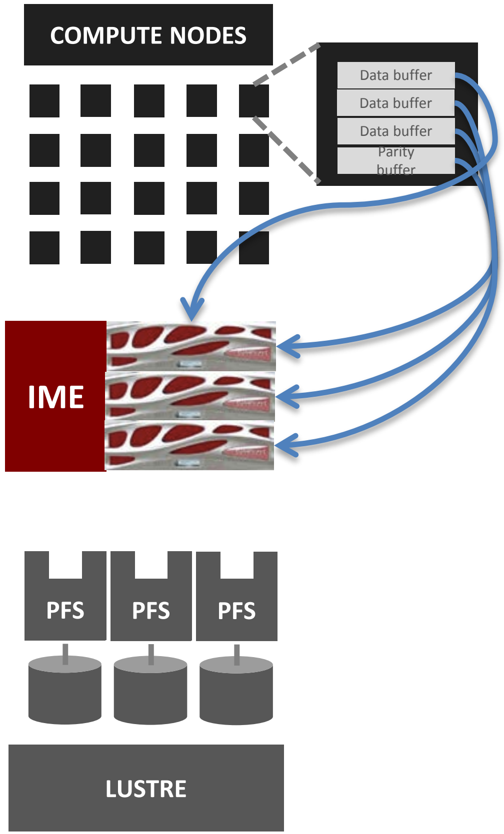
3. Fragments are sent to IME servers and are accessible via DHT to all clients

4. Fragments to be flushed from IME are assembled into PFS stripes

5. PFS receives complete aligned PFS stripe

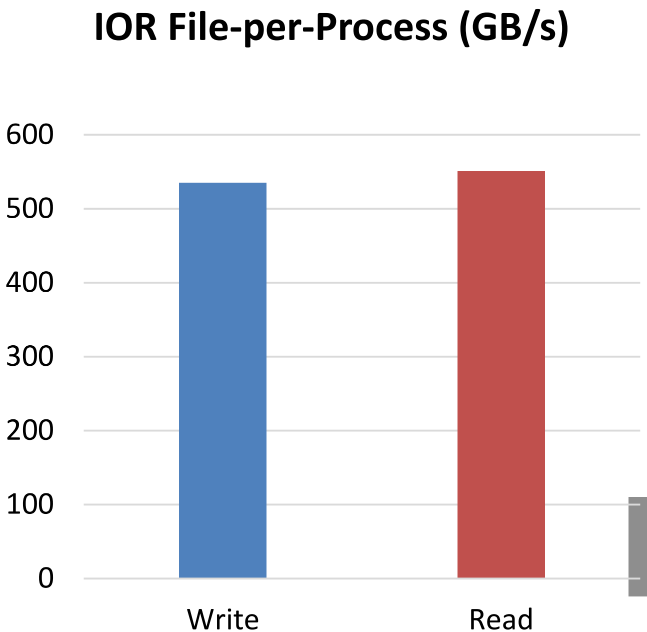


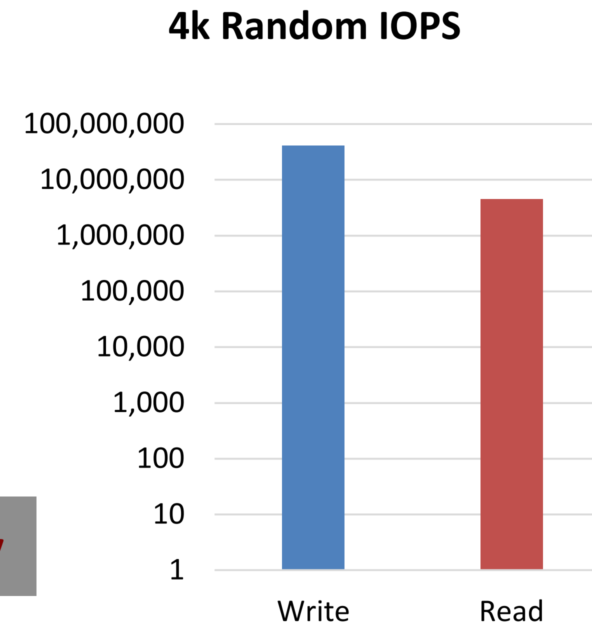
# 4. IME Erasure Coding




- Data protection against IME server or SSD Failure is optional
  - (the lost data is "just cache")
- Erasure Coding calculated at the Client
  - Great scaling with extremely high client count
  - Servers don't get clogged up
- Erasure coding does reduce useable Client bandwidth and useable IME capacity:
  - 3+1: 56Gb → 42Gb
  - 5+1: 56Gb → 47Gb
  - 7+1: 56Gb → 49Gb
  - 8+1: 56Gb → 50Gb

# 4. Rack Performance: IME




768TB Flash Capacity

50 Million IOPs

500GB/s Read and Write

# Summary

- SSDs can today be seamlessly introduced into a Lustre Filesystem
  - **Modest** investment in SSDs
  - **Intelligent** policy-driven data moves the most appropriate blocks/files to SSD cache
  - **Block level and Lustre Object** Level data placement schemes
- IME is a ground-up NVM distributed cache which adds
  - **Write** Performance optimisation (not just read)
  - **Small, random** I/O optimisations
  - **Shared** (many-to-one) file optimisations
  - **Improved** SSD lifetime
  - Back-end **Lustre IO optimisation**