
whamcloud

whamcloud

Lustre User Group
Austin, Tx
April, 2012

Leveraging Lustre to address I/O Challenges of Exascale

- Eric Barton

  CTO
  Whamcloud, Inc.
  eeb@whamcloud.com

Agenda

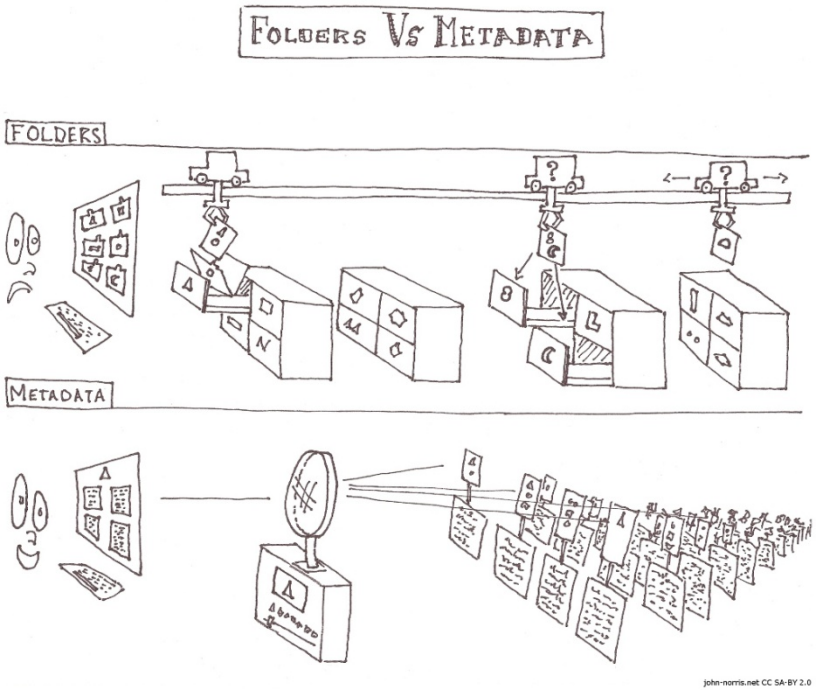
- Forces at work in exascale I/O
  - Technology drivers
  - I/O requirements
  - Software engineering issues
- Proposed exascale I/O model
  - Filesystem
  - Application I/O
  - Components

Exascale I/O technology drivers

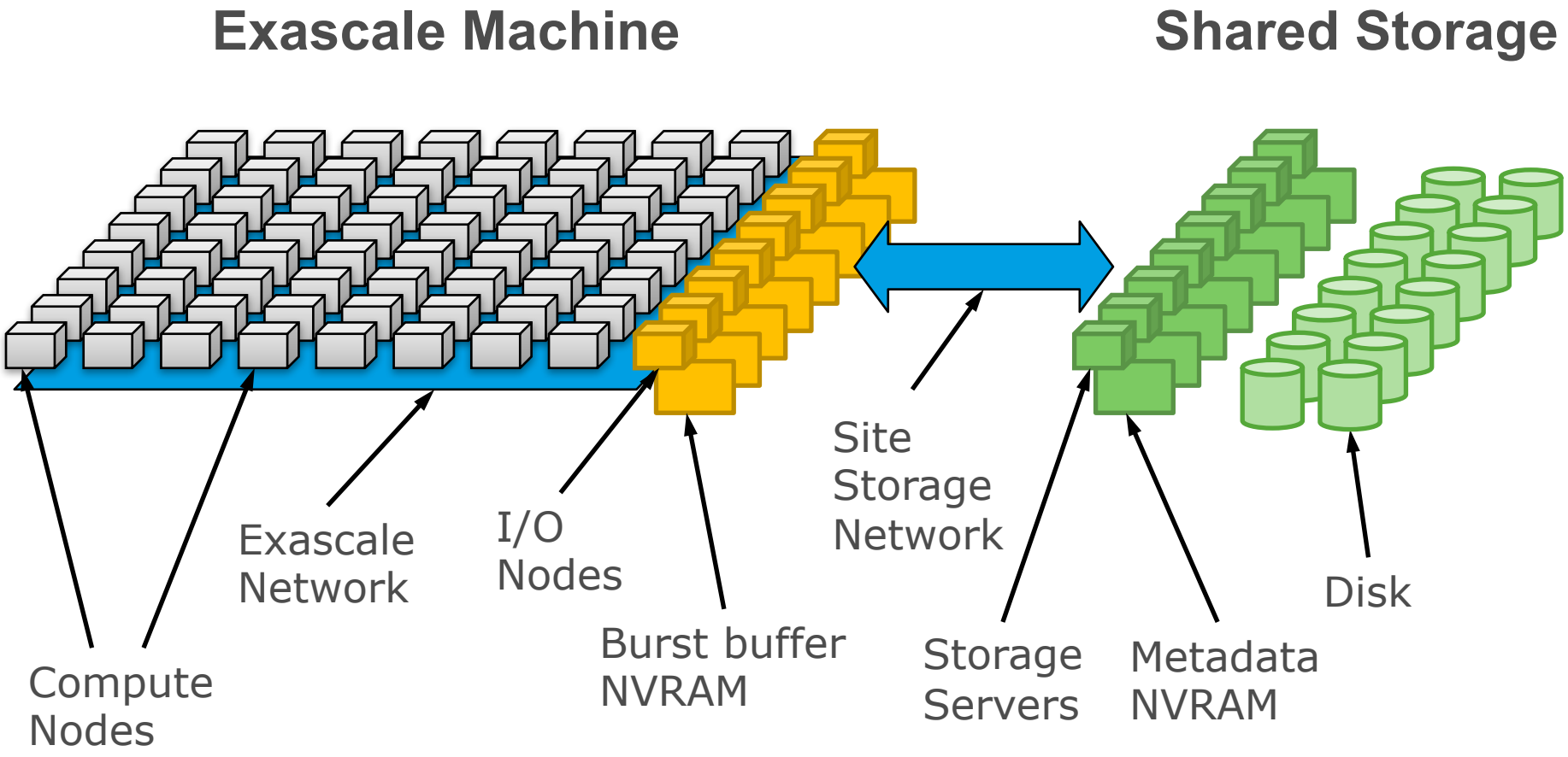
	2012	2020
Nodes	10-100K	100K-1M
Threads/node	~10	~1000
Total concurrency	100K-1M	100M-1B
Memory	1-4PB	30-60PB
FS Size	10-100PB	600-3000PB
MTTI	1-5 Days	6 Hours
Memory Dump	< 2000s	< 300s
Peak I/O BW	1-2TB/s	100-200TB/s
Sustained I/O BW	10-200GB/s	20TB/s
Object create	100K/s	100M/s

Exascale I/O technology drivers

- (Meta)data explosion
  - Many billions of entities
    - Mesh elements
    - Graph nodes
    - Timesteps
  - Complex relationships
  - UQ ensemble runs
- OODB
  - Read/Write -> Instantiate/Persist
  - Index / Search
    - Where's the 100 year wave
  - Data provenance + quality
- Storage Management
  - Migration / Archive



Exascale I/O Architecture



Exascale I/O requirements

- Concurrency
  - Death by 1,000M cuts
    - Scattered un-aligned variable size data structures
    - Asynchronous I/O
  - I/O Staging
    - Aggregate ~100 compute nodes x ~100-1000 threads
    - Burst buffer / pre-staging
    - "Laminar" data flow to global file system
      - Object-per-staging process
- Search & Analysis
  - Multiple indexes
  - Ad-hoc index creation
  - Pre-stage data for analysis
    - Subset determined by ad-hoc query

Exascale I/O requirements

- (Meta)data consistency + integrity
  - Metadata at one level is data in the level below
  - Foundational component of system resilience
  - Required end-to-end
- Balanced recovery strategies
  - Transactional models
    - Fast cleanup up failure
    - Filesystem always available
    - Filesystem always exists in a defined state
  - Scrubbing
    - Repair / resource recovery that may take days-weeks

Exascale I/O requirements

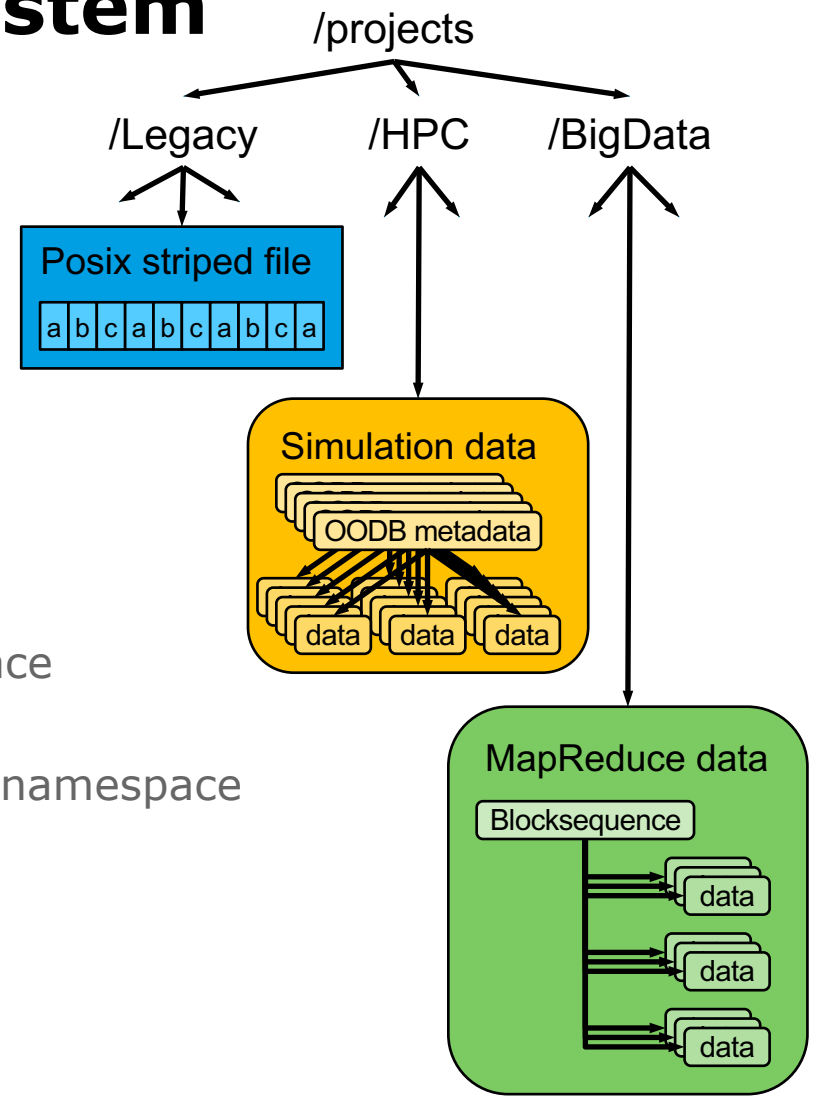
- Global v. local storage
  - Global looks like a filesystem
  - Local looks like
    - Cache / storage tier
      - How transparent?
    - Something more specific?
- Automated / policy / scheduler driven migration
  - Pre-staging from global F/S
  - Post-writeback to global F/S
- Fault isolation
  - Massive I/O node failure cannot affect shared global F/S
- Performance isolation
  - I/O staging nodes allocated per job
  - Qos requirements for shared global F/S

Software engineering

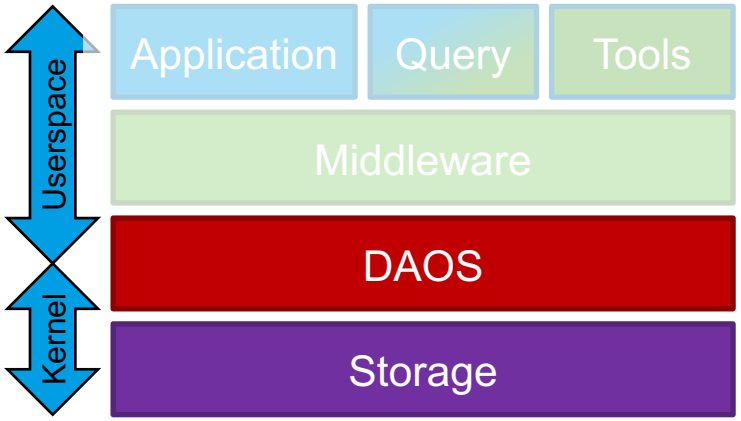
- Stabilization effort required non-trivial
  - Expensive/scarce scale development and test resources
- Build on existing components when possible
  - LNET (network abstraction), OSD API (backend storage abstraction)
- Implement new subsystems when required
  - Distributed Application Object Storage (DAOS)
- Clean stack
  - Common base features in lower layers
  - Application-domain-specific features in higher layers
  - APIs that enable concurrent development

Exascale shared filesystem

- Conventional namespace
  - Works at human scale
  - Administration
    - Security & accounting
  - Legacy data and applications
- DAOS Containers
  - Work at exascale
  - Embedded in conventional namespace
  - Scalable storage objects
  - App/middleware determined object namespace
- Storage pools
  - Quota
  - Streaming v. IOPS

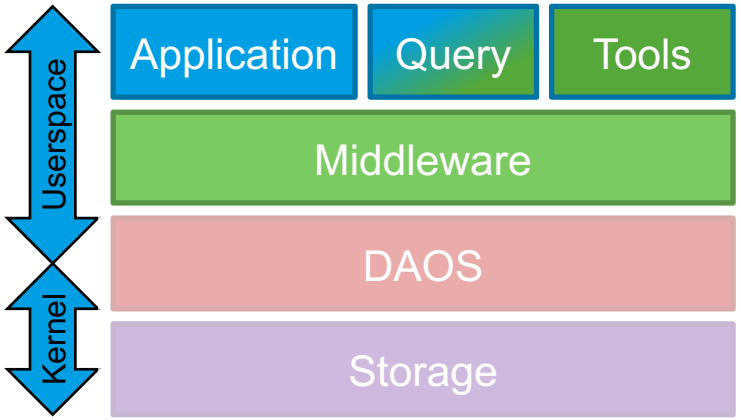


I/O stack



- DAOS Containers
  - Application data and metadata
  - Object resilience
    - N-way mirrors / RAID6
  - Data management
    - Migration over pools / between containers
  - 10s of billions of objects distributed over thousands of OSSs
    - Share-nothing create/destroy, read/write
    - Millions of application threads
  - ACID transactions on objects and containers
    - Defined state on any/all combinations of failures
    - No scanning on recovery

I/O stack



- Userspace
  - Easier development and debug
  - Low latency / OS bypass
- Middleware
  - Domain-specific API style
    - Collective / independent
    - Transaction model
    - OODB, Hadoop, HDF5, Posix…
  - I/O staging / burst buffers
- Applications and tools
  - Backup and restore
  - Query, search and analysis
  - Data browsers, visualisers and editors
  - General purpose or application specific according to target APIs

Thank You

- Eric Barton
  CTO
  Whamcloud, Inc.
  eeb@whamcloud.com