Towards ICN-based Data-Locality in HPC

IRTF ICNRG-IETF93 July 20, 2015

Michael Alexander, Vienna Scientific Cluster, TU Wien

Can NDN supplemental HPC storage be an ICN use case?

- HPC Range
- Challenges
- Parallel File Systems
- POSIX
- Plurality of Datastores
- Ongoing Changes
- Constants
- A NDN Storage API?
- Enriching Storage with NDN Elements - Implementation Options

HPC Range

- Classic HPC (mesthes, MPI, PDEs, …)
    - Parallel file systems
    - Opportunities for I/O optimization persist
- Data intensive computing
    - Operate on externally generated data
- High throughput computing
    - e.g. Genomics pipeline
    - Loosely coupled processes
    - Distributed storage

Challenges

- Parallel File System Limitations
  - Large aggregate throughput balanced with metadata performance
  - Fault tolerance-resilience-recovery
  - Physics of spinning disks, solid state economics dictate constraints
  - Function/Analysis shipping rare
  - Moving data to compute will remain
  - Small file, mixed workloads metadata performance

Challenges

- HPC I/O Research emphasises exascale context
  - Fast-Forward I/O stack
- Spinning disk hierarchical storage - post tape
- Disconnect between code I/O properties and storage stack
  - Storage stack has to discover behavior at runtime
  - Immutable data, coherence (write back/through)

Parallel File Systems

- Single namespace
- CAP Consistency-focused
- Frequently object store based
- Some similarities to NDN
- Central metadata, may resolve by name within metadata tree

POSIX File Systems

- POSIX semantics not not well attuned to HPC
  - Bytestream paradigm
    - Defined for hierarchical file systems
    - Strong consistency
- Yet everybody programs against it

Plurality of Datastores

- Object Stores
    - Hashes, indexes, limited CRUD operations
    - Non-POSIX drawback
- HDF5
    - Hierarchical, relates to POSIX
    - Elaborate data model
    - Index datasets
    - Highly Optimized: MPI-IO VFD
- (Parallel) netCDFN
- Entity-Attribute-Value DBs
- SQL

Ongoing Changes

- Multi-level storage hierarchies
  - Flash, Buffered DRAM
  - Burst buffers
    - campaign store
- Fast-Forward I/O and Storage Stack
  - Exascale context
- Can transparent POSIX continue as main pattern?

Constants

- Preference for node-local storage
  - Performance
  - Availability
- Codes wanting to
  - Avoid programming towards a storage stack
  - Stay close to POSIX semantics

A NDN Storage API?

- Tuples {IOPS, streaming throughput, latency, non-contiguous performance, …} could describe storage layers, instances
  - as classes
  - mapped to arrays, storage targets
    - POSIX, HDF5/(Parallel) netCDF, Objectstore, Entity-Attribute-Value, SQL
- /my/data?ssdcache=writeback
- Acceptable higher latencies in cloud HPC
- As network functions?
- Folding in storage API?

Enriching Storage with NDN Elements - Implementation Options

- Abstract API
- Modify Objectstore or Parallel Filesystem
- Internal or **Overlay**
- HDF5/PnetCDF (overlay, internal)
- What would a demonstrator stack need to be able to do?

References

Michael Alexander. A Reconciled Data Warehouse Layer Based on CCNx. CCNxCon PARC, Palo Alto. Sept. 2013.

Michael Alexander. Datastore Aware Programming. IBM Developer Days, Vienna, 2014.

Salman Habib. Extreme Scaling and Performance across Diverse Architectures. ACM sighpc Webinar, March 31, 2015.

Jeff Layton and Eric Barton. Fast Forward Storage & I/O. 30th International Conferenceon Massive Storage Systems and Technology, 2014.