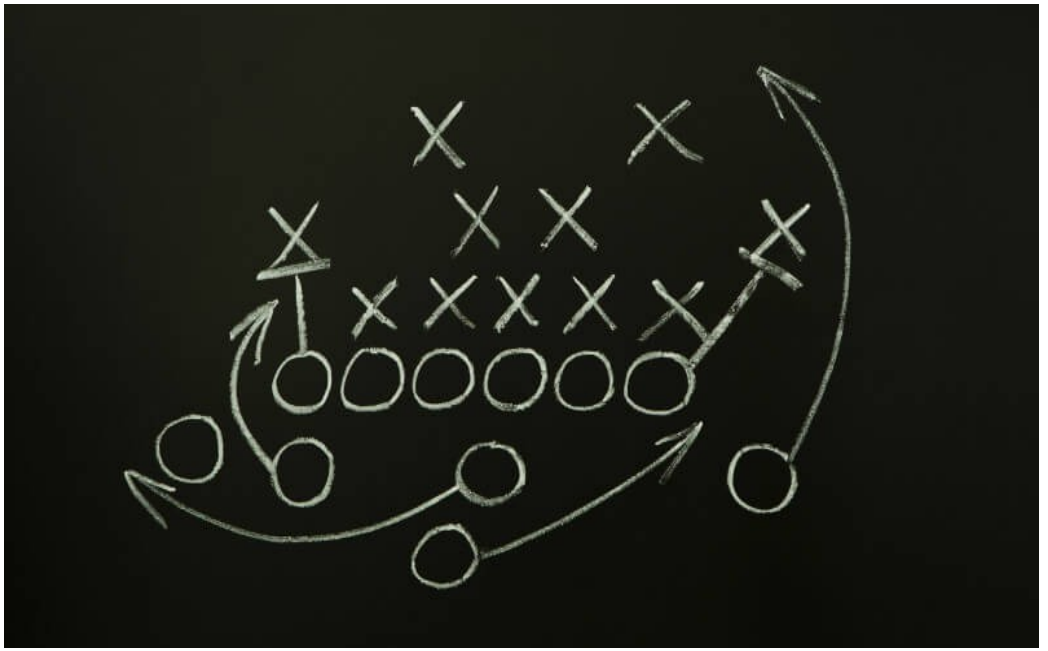
DE LA RECHERCHE À L'INDUSTRIE

www.cea.fr

Storage Strategy for the Exascale: why we have a need for Object Storage

Philippe DENIEL (philippe.deniel@cea.fr)

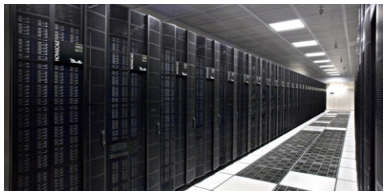


Data Storage: an archeological approach

Data Storage for HPC has evolved a lot since the 90's

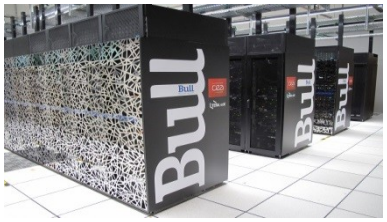
- **90's: The HPC Cavemen's Era**
  - Cray T924 and Cray T3E, ~60Gflops (6.10^7 flops), 80 TB tapes
    - Fun fact: Today's Samsung Galaxy S10+ = 8x13.4 Gflops
- **2000: Terascale period (beginning of the SMP Cluster Age)**
  - Tera1 supercomputer installed at CEA
  - 5 Tflops (5.10^9 flops)
  - 50TB (disks), 1PB (tapes)
    - Fun fact: STK 9840 “Eagle” tapes had 40GB capacity, we had 25000 of them...

- **2005: First Intermediate Period**
  - Tera10 supercomputer @ CEA
  - 60 Tflops (6.10^10 flops), 2 PB (disks, 100 GB/s), 10 PB (tapes)



- **2010: Petascale period (beginning of the “Cluster of clusters” Age)**
  - Tera100 supercomputer @ CEA
  - 1.05 Pflops (10^12 flops), 20 PB (disks, 500 GB/s), 30 PB (tapes)

- **2015: Second Intermediate Period**
  - Tera1000 supercomputer
  - 30 Pflops (3.10^13 flops)
  - 40 PB (HDD, 767 GB/s), 2 PB (SSD, 1 TB/s), 100 PB (tapes)



- **2022: Early Steps of the Exascale**
  - Exa1 supercomputer @ CEA, currently in production
  - 100 PB (SSD+HDD, 2.5TB/s, 25 millions IOPS), 300 PB (tapes)

The Exascale is a major challenge for storage (1/2)

About Scalability

- **System Scalability**
  - Exascale supercomputer will expose hundreds of thousands of clients
    - *This aspect is mitigated by GPU*, for this technology brings many cores per host
  - Memory per core is decreasing : less room for the OS
    - *This aspect is enhanced by GPU*, for this technology brings many cores per host
    - Building complex states machines to deal with so many clients won't be possible

- **Data Scalability**
  - *Metadata Scalability*: Thousands of billions of records (files)
    - Metadata tends to become more complex (via files's extended attributes)
  - *Pure Data Scalability:* about ~10-100EB data stored

The Exascale is a major challenge for storage (2/2)

Other Challenges

- **Data Heterogeneity**
  - different record sizes: from a few bytes – kilobytes to several terabytes
  - Different usage: Video (stream) / Check-point restart (Write Once Read Never) / DB chunks
  - Different data lifetimes and data life cycle

- **Resources Heterogeneity**
  - Different storage devices (NVRAM,SSD, HDD, Tapes), stacked within storage tiers
  - Heterogeneous network performance result from complex network topologies

- **Data Placement**
  - Putting the data at the right place avoids migration
  - Moving data cost resources: CPU and RAM, bu don’t forget network and power

Change the user's point of view on data

Globality vs Locality

- Do we need to have all of the data accessible by everyone, everytime, everywhere?
- The simulation code actually needs a small fraction of data
  - The storage system MUST show this fraction to the code
  - It DOES NOT REQUIRE to show everything else (useless data for this run of the code)
- The data stored and their usages should be considered through new point of views
  - Use local replica (with version management) for read-only data
  - Maybe using local replica for read-write, with a lease management

Introducing Data Nodes

- Some nodes will be dedicated to IO / DATA operations, the **data nodes**
- They are close to compute nodes and are allocated the same way
  - They are managed by the Resources Manager like CPU/GPU and RAM

Introducing Data Accessors and Ephemeral Services

Data Accessors

- Each simulation knows precisely the way it will access data
  - POSIX namespace (mounted file systems) should be “legacy interface”
  - S3/Swift buckets (e.g. TF based applications)
  - MPI-IO / HDF5
  - Other pieces of middleware
- The Data Accessors is associated with a job and run on a datanode (or a set of datanodes)

Ephemeral services

- **Data accessors**, are running on **data nodes**, and are associated with a simulation job
- Ephemeral services are spawn spawned by the Resource Manager

The IO-SEA project

- IO-SEA is a project funded by EuroHPC
  - It involves data nodes and ephemeral services management
  - It involves Hierarchical Storage Management
  - It strongly relies on **Object Storage**
- More details on the IO-SEA project: **https://iosea-project.eu/**

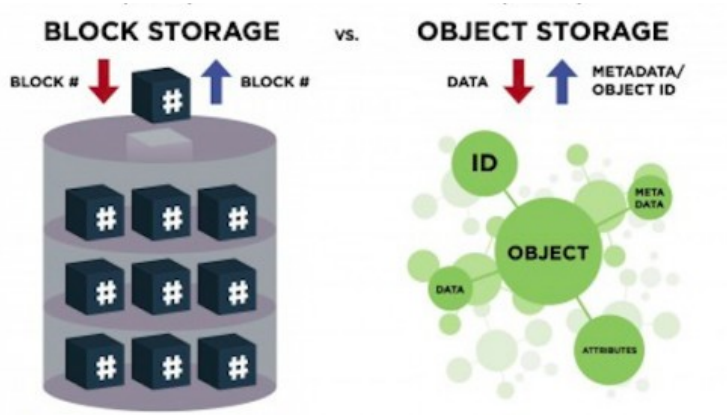
Object Store to implement Data Accessors

Object Storage is based on Associative Addressing

- An amount of data, of variable size, is addressed by a string (a key)
- Knowing the key allows to get a handle to the data content
  - Tools like YouTube makes a large usage of this kind of technology
  - depending on the product, randow R/W and/or PUT/GET access is possible

The Object Store paradigm removes constraints

- It has a simple put/get/delete/update interface with simple semantics (like CRUD)
  - Simple semantics allows more complex semantics to be implemented on top of it
  - Even complex semantic such as POSIX can be implemented
- Object Stores are “data strongholds”
  - They can be easily distributed
  - They can be easily replicated
- No dependencies between objects
  - The number of objects is no longer a limitation



Using Object Store jointly with Key-Value Store

Key Value is required for metadata management

- A key (a string) addresses a string or a small piece of data, read in a single request
- Implementing Data Accessors requires metadata management
  - KVS are designed to handle metadata sets

Using Key Value Stores and Object Stores together

- There is a “natural split” between data and metadata
  - Lustre and NFSv4.1/pNFS have
    - MDS (Metadata Servers) for metadata management
    - Lustre/OSS and pNFS/DS (Data Server) for data management
- But KVS and Object Store should closely work together
  - It’s nice to have transactions that mixes operations to object stores AND KVS

About Intel’s DAOS

- DAOS implements a very fast and efficient Object Storage with KVS overtones
- DAOS has explicit support of transactions
- It has the HPC Storage in its targets,
  - it meets the requirements expressed in the former slides

daos