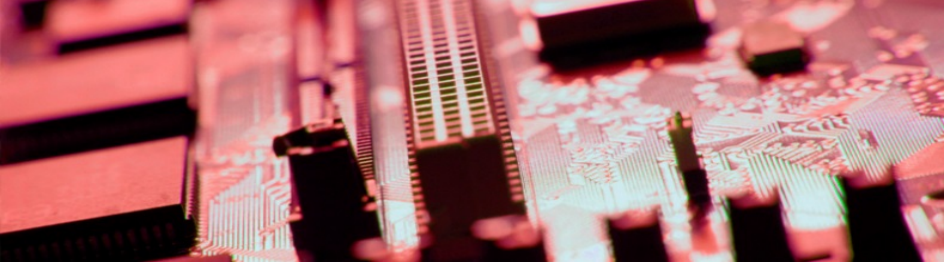


LCI workshop
on high performance clustered computing

Linux Clusters Institute: Tiered Storage

Georgia Tech, August 15th – 18th 2017

J.D. Maloney | Storage Engineer
National Center for Supercomputing Applications (NCSA)
malone12@illinois.edu

NCSA

Tiered Storage: What is it & Why

The Concept - Tiers

- A storage tier is a group of storage media with the same properties and characteristics, some examples:
  - SSDs as a Fast Tier
  - HDDs for a slower Tier
  - Tape for an archive Tier
- Different tiers of data host different types or classifications of data, for example:
  - Certain directories live in certain tiers
  - Data with certain properties live in certain tiers
- Each of the tiers offers a different set of characteristics that's beneficial in different circumstances

The Concept - HSM

- While data movement between tiers can be done manually, automatic movement is most desired
- In order for automated migrations to take place the file system needs to support an HSM (Hierarchical Storage Manager)
- Based on given policies data is migrated between the different storage tiers automatically
  - Ready on fast tiers when the data is needed promptly
  - Pushed down to slower levels of storage for files that are rarely used
- Both Spectrum Scale (GPFS) and Lustre support HSM
  - Built into Spectrum Scale
  - Built into Lustre, but driven from external Robinhood System

What's the appeal?

- So what's the motivation behind this tactic, what's the benefit?...It's two fold:
  - Performance
  - Cost
- You can get the performance of fast disk such as SSDs for a lot of your I/O
- Since most of your capacity is in slower media the total system price can be kept lower
- Gets you close to the best of both worlds!

What's Old is New

- We've seen this before, but in the context of a node's storage access

Location	register	L1	L2	L3	DRAM	HDD
Size	64 bits	64 bytes	128 bytes		4GB	1TB
Latency cycles	1	4	12	50	10^2	10^4

The Challenges

- If it was easy, everyone would be doing it
- Getting data in the right place at the right time can be tricky
  - Easier to handle on two tier SSD/HDD implementations
  - Harder to work with when you add in another tier...especially tape
- With ultra slow, but really low cost tiers, staging requests need to be given well in advance of need for the data
- Data needs to be able to flush out of the fastest tier quick enough to accommodate next round of data
- Depending on size of the fast tier, may not be able to hold all necessary data – can lead to data flapping back and forth between two tiers

The Challenges (cont)

- Highest level of the system needs to know about where all the data is
  - Source of truth needs to be kept somewhere
  - Can be a large burden on that tier of storage to track what tier each file is in
- Ensuring QOS for other I/O on the system
  - We want this movement to stay out of the way of production work as much as possible
- Maintaining data integrity as files move between different systems
- Handling crashes gracefully when they occur so system doesn't lose track of where data is living

List of Benefits

- Gives your system the performance (bandwidth, IOPs) of high speed media
- Keeps system costs lower/GB by leveraging cheaper capacity tiers for bulk data storage and archive
- Efficient use of resources
- Unlocks access to features of multiple different storage technologies within the same system
- Increases resources for users so they can increase their use of the system to aid in their work

List of Side-Effects

- Increased system complexity, and sometimes points of failure
- Need to use file system bandwidth for data movement unrelated to the workload(s) being run
- Mandate to coordinate data needs to ensure data availability at proper times
- Need to track file tier location at all times
- Additional level of acumen to manage different system types that are combined together

Current Implementations

Locally with ZFS

- While in general HPC is interested in tier storage at the macro level, important to note ZFS does this on a single machine level
  - Improved capability than XFS or EXT4 file systems in terms of performance
- Memory flushing to disk sequentially behaves like a Burst Buffer does on large parallel file systems
- One of the reasons it's a growingly popular FS to put underneath file systems such as Lustre
  - That benefit from ZFS will show itself too in the Lustre performance in certain cases

Locally with ZFS

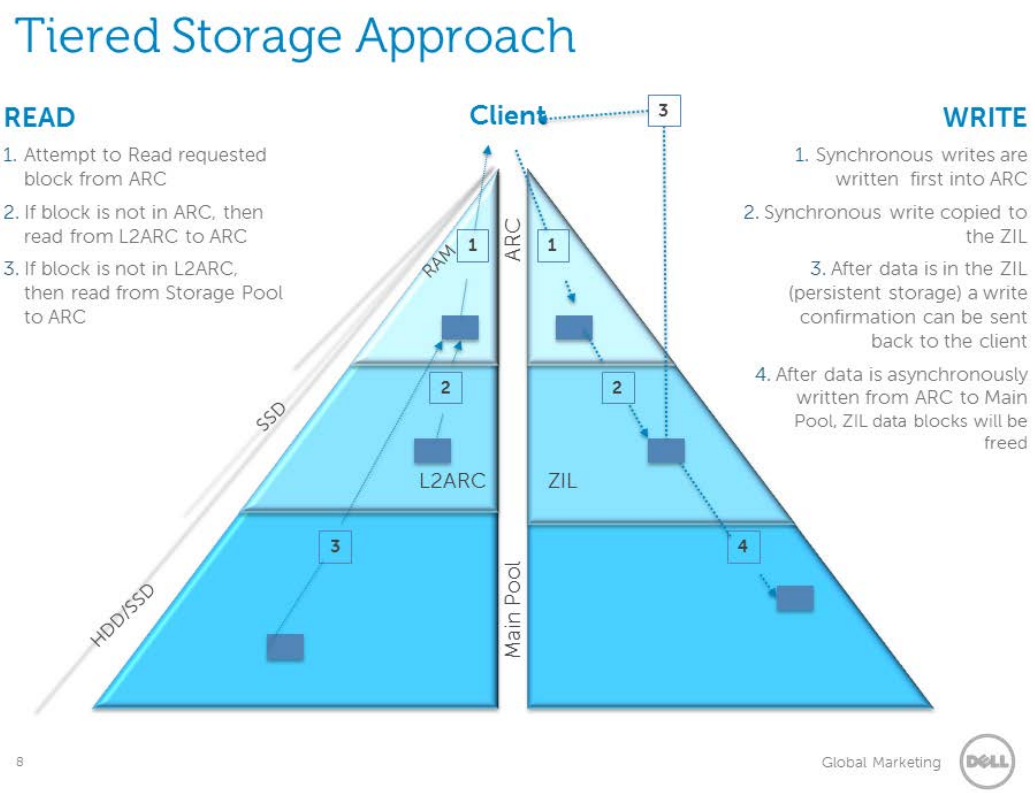


Image Credit: Dell

Spectrum Scale

- Support for tiered storage in Spectrum Scale (GPFS) has been around for a long time in the file system
- Spectrum Scale supports the notion of “storage pools”
- NSDs are each assigned to a given storage pool that represents the characteristics of the NSD
- The movement of data between the tiers of disk can be driven manually
  - Through normal restripe procedures, marking NSDs read-only, etc.
- Big ability is to drive data movement through the built in policy engine

Spectrum Scale

- Support for tiered storage in Spectrum Scale (GPFS) has been around for a long time in the file system
- Spectrum Scale supports the notion of “storage pools”
- NSDs are each assigned to a given storage pool that represents the characteristics of the NSD
- The movement of data between the tiers of disk can be driven manually
  - Through normal restripe procedures, marking NSDs read-only, etc.
- Big ability is to drive data movement through the build in policy engine

Spectrum Scale

Client Based Storage Tiering

- LROC
  - Local Read-Only Cache
  - Sits in client node, can be single non-redundant flash device
  - Expansion of page pool in read respects
    - Does consume some page pool by existing but overall big gain in capacity
    - Not meant to replace page pool
  - Very helpful in repeated-read workloads
- HAWC
  - Highly Available Write Cache
  - Sits in client node, should be redundant flash device
    - Failure of HAWC could lead to loss of data not flushed to PFS
  - Absorbs write bursts to the file system

Spectrum Scale

- Tons of flexibility
- Well implemented into the file system and fairly easy to manage with written policies
- Integrates with other IBM protects such as Spectrum Protect (formerly known as TSM)
- Supports tape tiers, per above, which is not a trivial task to handle
  - Scale and file churn can be burdensome here, depending on file system environment, may not be a good option
- One of the most well included HSM options available in HPC at this time

Lustre

- Integrated into the file system natively, check out the lfs hsm command set
- Lustre keeps track of the archive state of all objects on the file system so it knows where they are
- Requires a copy tool to interface with the external storage target, hooks allow Lustre to push and pull data
- Coordination is done with the file system MDS to notify of data and call out to get data that is needed by the file system
- Benefit is that Lustre doesn't need to know underlying external file store, just needs to be able to talk to it

Lustre HSM Architecture

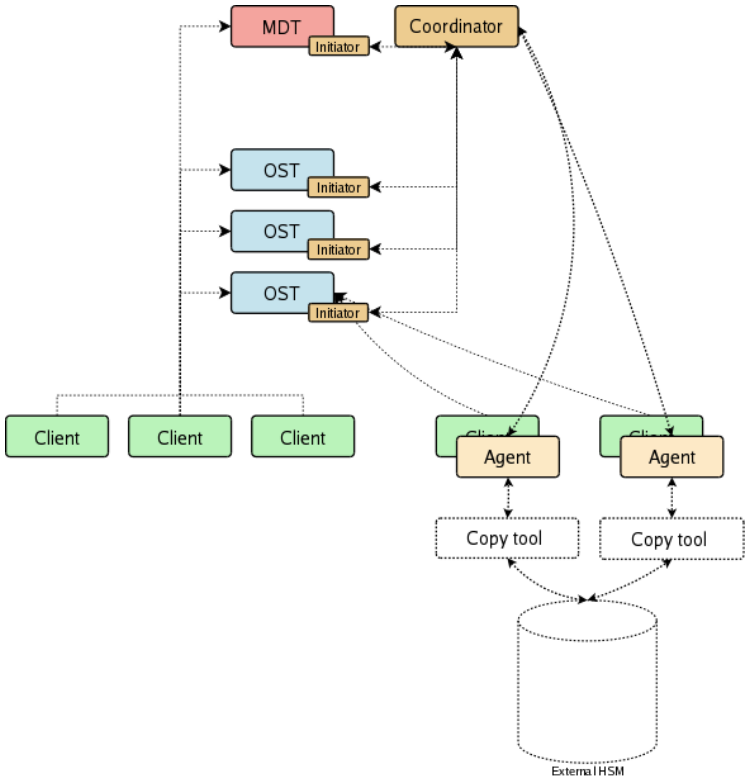


Image Credit: lustre.org

Lustre

- Driving the data between the tiers can be done manually from the CLI
  - See the lfs hsm commands
- While there is no policy engine built into Lustre itself, Robinhood can be used to drive these migrations
- Requires change logs to be enabled on the file system and they need to function reliably to prevent the need of a full scan to find all the files
- No support for tiers within the Lustre file system
  - Eg. some OSTs being SSD vs PMR HDD vs SMR HDD

Support in Other File Systems

- Ceph
  - Hits this with different pools
  - Can have a fast (probably all SSD pool) that acts as a cache tier for a slower pool beneath it
  - Works best for workloads with high read rates of recently written objects
  - Built in support
- GlusterFS
  - Built in tier support for single volumes
  - Volume can have fast bricks and slow bricks and data is moved between the two based on their use frequency and last access time
  - Done automatically by the file system, but can be manipulated by tunable parameters
  - File location is transparent to the application, appears as though it's in normal spot

Support in Other File Systems

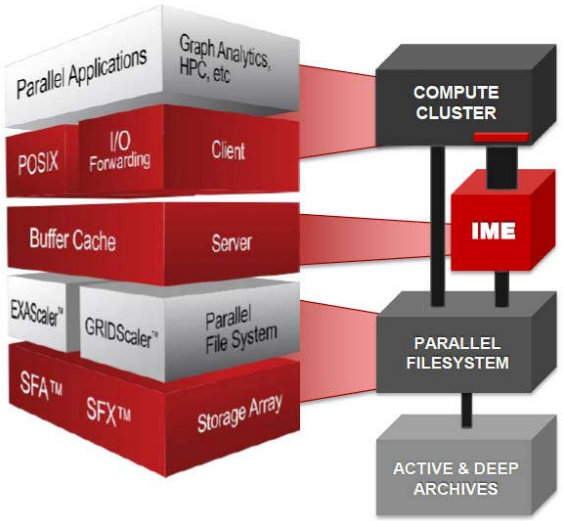
- Panasas
  - Feature of PanFS for a while
  - Available via hybrid storage tiers
  - Uses SSDs to accelerate rotational media
- BeeGFS
  - No current support for Tiered Storage...yet
  - Support set to come later this year (2017)
  - Will come in the form of storage pools similar to how Spectrum Scale handles tiered data

Putting HSM into the Hardware

- Dealing with putting the logic into the file systems themselves can be challenging and time consuming
  - Puts all the effort on the software side of things
- Second approach is putting tier intelligence in the hardware appliances without really telling the file system
- This approach is used in products like Cray's DataWarp, DDN's IME, and Seagate with their Nytro accelerated appliances
- Is really best for moving data between flash and spinning hard drive tiers
  - Not really for pushing data into and out of slower tiers like tape

Putting HSM into the Hardware

- DDN IME leverages a separate storage appliance filled with flash to buffer I/O between the compute system and the parallel file system
- Has native IME libraries to offer optimized I/O performance
- Works with both Lustre and Spectrum Scale



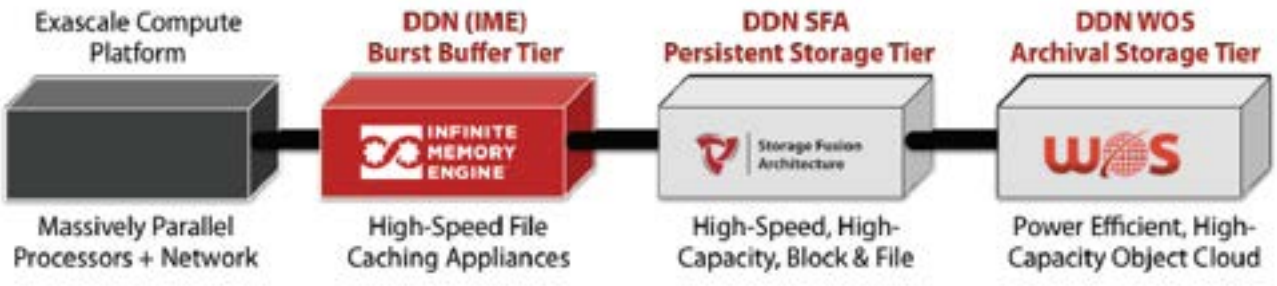
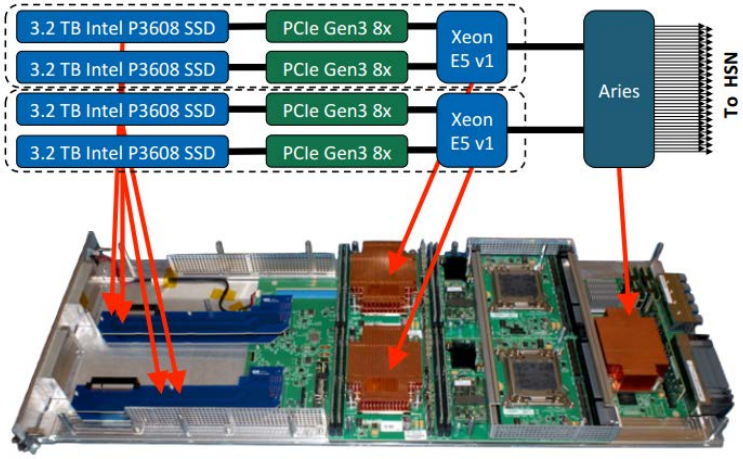


Image Credit: ddn.com

Putting HSM into the Hardware

- Cray Burst buffer technology puts flash accelerator nodes into the system to intercept I/O bursts and smooth it out on it's way to the parallel file system
- Works with Lustre, Spectrum Scale, and PanFS



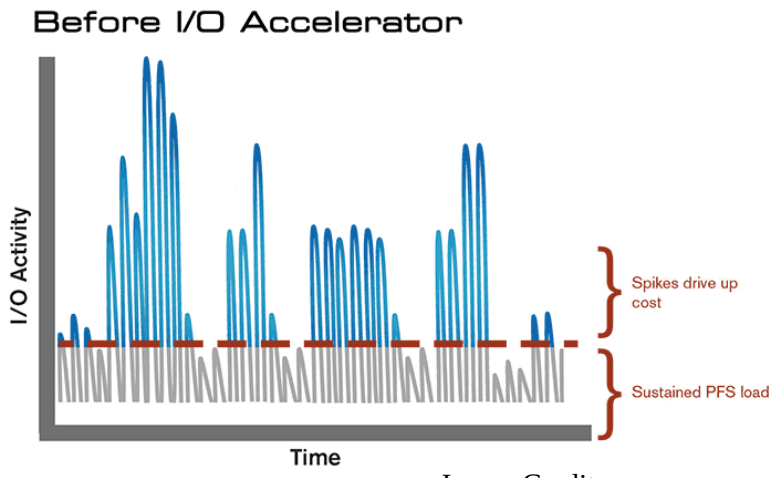


Image Credit: cray.com

Putting HSM into the Hardware

- Seagate Nytro based appliances (L300N and G300N) integrate both rotational media and flash into the appliance itself
- Uses the Nytro Intelligent I/O manager to absorb high I/O and write it to disk and a consistent sequential manner
  - Gives disks best I/O possible to improve their performance
  - Handles applications with poor I/O patterns
- Again just a Flash/HDD tiered system, but one that is transparent to filesystem and user application

Looking Toward the Future

Tackling the Challenges

- Using job scheduling tools to coordinate data movement between tiers
  - Job dependencies that run staging requests for data that have to be completed before scheduler launches the actual job
- Ensuring robust bandwidth between tiers to reduce the ability of data migrations constraining I/O bandwidth
- Fine tuning data movement policies to ensure data is moved only when necessary and in an efficient manner as possible

Tackling the Challenges

- Leveraging new storage medias
  - Even within rotational media there are different technologies with different characteristics: PMR, SMR, HAMR
  - NVME Flash
  - 3D XPoint Technology
- Leveraging more distributed name space abilities
  - DNE 2 in Lustre improved metadata performance and thus HSM performance abilities
  - New GlusterFS core engine will improve performance and scalability
- More software development from companies
  - Companies are putting effort in, especially slow tier vendors to write bridge code to allow file systems like Lustre or Spectrum Scale to talk to their deep storage appliance

HSM Projections

- Increase in the number of storage tiers in the stack, we're already seeing it
  - Memory
  - In-node flash
  - File System Flash
  - HDD
  - SMR
  - Object
  - Tape
- With tier count increasing their will be an increase in complexity in terms of tracking where data sits

HSM Projections

- Migration away from tape based storage tiers for all but some of the largest systems
  - Tape while low in cost, requires specialized skills to administer and maintain
  - Finding talent to run the systems is becoming more difficult
  - Economies of scale with falling disk prices means that more data is necessary to justify the cost of tape
- More file systems gaining features related to HSM
  - Very popular feature and is a big bonus to a file system that can support it well and reliably
  - Some of the ”smaller” file systems will/have start seeing better adoption

HSM Projections

- Flipping the paradigm over and having the lower storage tier hold file state of all objects in the file system
- Larger capacity tier could have easier time handling the file state information
- When data is needed up on the fast storage tier, necessary files are pushed up into hot storage space
  - Once job is completed, new and changed files are then merged back into the capacity tier below
  - Conceptually similar to how git control works
  - Simplifies tracking of the files state, but could be complex in the ability to merge files back into lower level correctly

Acknowledgements

- Members of the SET group at NCSA for slide content & review
- Members of the steering committee for slide review

Questions