Implementing a Hierarchical Storage Management system in a large-scale Lustre and HPSS environment

Brett Bode, Michelle Butler, Sean Stevens, Jim Glasgow
National Center for Supercomputing Applications/University of Illinois
Nate Schumann, Frank Zago
Cray Inc.
Background

• What is Hierarchical Storage Management?
  • A single namespace view for multiple physical/logical storage systems
  • Automated movement of data to the lower tiers, usually based on configurable policies
  • Data is returned to the top tier based on request or access.
Traditional HSM

- HSM environments have been used for many years to front-end tape archives with a "disk cache".
  - Usually the environment was isolated, the only actions on data are to transfer to/from the system.
  - All data is expected to be written to the back-end.
  - Most data is accessed infrequently.

User accesses single namespace via explicit data transfer to/from the environment.

Disk cache: Automatically migrate by policy; Dual resident and release as needed.

Tape Archive: Released data requested is retrieved.
Blue Waters Computing System

Aggregate Memory – 1.66 PB

Scuba Subsystem - Storage Configuration for User Best Access

120+ GB/sec

10/40/100 Gb Ethernet Switch – 66 GB/sec

External Servers

IB Switch – >1 TB/sec

120+ GB/sec

400+ Gbps WAN

Spectra Logic: 200 usable PB

Sonexion: 26 usable PB
Cray XE6/XK7 - 288 Cabinets

- XE6 Compute Nodes - 5,659 Blades – 22,636 Nodes – 362,176 FP (bulldozer) Cores – 724,352 Integer Cores
- XK7 GPU Nodes - 1057 Blades – 4,228 Nodes 33,824 FP Cores – 4,228 GPUs

Gemini Fabric (HSN)

- DSL 48 Nodes
- Resource Manager (MOM) 64 Nodes
- XE6 Compute Nodes - 5,659 Blades – 22,636 Nodes – 362,176 FP (bulldozer) Cores – 724,352 Integer Cores
- Unassigned 74 Nodes
- RKIP 12 Nodes
- Network GW 8 Nodes
- Unassigned 74 Nodes
- LNET Routers 582 Nodes

Supporting systems: LDAP, RSA, Portal, JIRA, Globus CA, Bro, test systems, Accounts/Allocations, Wiki

NPCF
Today's Data Management

User manages file location between scratch and Nearline

- scratch
- 30 day Purge
- Nearline

Globus commands
Hierarchical Storage Management (HSM) Vision:

- Purge no longer employed
- Allows policy parameters to manage filesystem free space
- Users limited by lustre quota and back-end quota (out of band)
HSM Design

- Lustre 2.5 or later required for HSM support – NCSA completed upgrade for all file systems in August 2016 (Sonexion Neo 2.0)
- Lustre copy tool provided via co-design & development with Cray Tiered Adaptive Storage Connector product
  - Cray provides bulk of the copy tool with a plugin architecture for various back-ends.
  - NCSA develops a plugin to interface with HPSS tape systems
  - Specifications created for resiliency, quotas, disaster recovery, etc.
Lustre HSM Support

- Lustre (2.5 and later) provides tracking for files regardless of the location of the data blocks.
  - File information remains on the MDS, but none on OSTs
- Commands are provided to initiate data movement and to release the lustre data blocks
  - `lsf hsm_[archive|release|restore|etc]`
  - Lustre tracks the requests, but does not provide any built in data movement.
- A copy tool component is required to register with Lustre in order for any data movement to occur.
HSM Policies

- Policy engine provided by Robinhood
  - Policies drive scripts that execute the `lsf hsm_*` commands.
- It is very early in the policy development process so these are early thoughts.
  - Files will be copied to the back end after 7 days
    - Estimated based on a review of the churn in the scratch file system.
  - Files will be released when the file system reaches 80% full based on age and size.
  - Files below 16MB will never be released and will be copied to a secondary file system rather than HPSS
Cray’s Connector

- Cray’s connector consists of two components
  - Connector Migration Manager (CMM)
    - registers with the Lustre file system as a copy tool.
    - It is responsible for queuing and scheduling Lustre HSM requests that are received from the MDT across one or more CMAs.
  - Connector Migration Agents (CMA)
    - Perform all data movement within the Connector, and are also responsible for removing archive files from back-end storage if requested.
CMA Plugins

- The CMA plugin architecture allows multiple back-ends to interface with the CMM.
- The CMA also allows threading transfers across multiple agents.
- Several sample CMAs are provided to copy data to a secondary file system.
The NCSA HPSS CMA plugin is called HTAP.

- Will be released as open source soon.
- Utilizes HPSS API to provide authentication and data transfer to/from the HPSS environment.
- Transfers can be parallelized to match HPSS COS.
Data IO

• Data IO for all file transfers is done at the native stripe width for the selected HPSS class of service
  • this is generally smaller than the Lustre stripe width however, files are restored to their original Lustre stripe width

• All file archive and retrieve operations are further verified by full checksums
  • checksums are kept with the files in user extended attributes and HPSS UDAs
  • the CMA provides parallel and inline checksum capabilities in a variety of standard methods
User application writes files to Lustre

“scratch”

Robinhood updated via changelogs

Robinhood DB

Data Mover CMM/CMA

Nearline

CUG 2017
Time Passes

Nearline

Data Mover
CMM/CMA

Robinhood DB

Ifs hsm_state afile
afile: (0x00000000)  <--- new file not archived
Policy Triggers Copy to Back End

Policy engine issues Lsf hsm_archive commands

Robinhood policy selects file based on size/age/etc file attributes

“scratch”

Nearline

Robinhood DB

Data Mover
CMM/CMA

CUG 2017
Lustre issues copy request to the copy tool (CMM)

“scratch”

Data Mover CMM/CMA

Robinhood DB

Nearline
CMM kicks off CMA(s) which copy the file to the back end and then store the backend metadata as extended lustre attrs.

`lfs hsm_state somefile`

somefile: (0x00000009) exists archived, archive_id:1<--- file archived, not released
At a later time Robinhood policies choose a list of files to release in order to free file system space. Data blocks are freed, but metadata remains.

```bash
lfs hsm_state somefile
somefile: (0x0000000d) released exists archived, archive_id:1 <--- file archived and released
```
File Restore

User makes request to stage files (or issues a file open)
User makes request to stage files (or issues a file open)

```bash
lfs hsm_state somefile
somefile: (0x00000009) exists archived, archive_id:1<--- file archived, not released
```
Backup?

- HSM is NOT equivalent to a backup!
  - One site uses HSM functions to dual-copy data - sort of a backup in the case of a fault in the primary storage.
  - However, deleting a file in the file system quickly results in it being deleted from the back-end.
    - Thus, HSM does not protect against user mistakes.
    - One could create a backend that did file versioning and delayed deletes, but that goes well beyond the current work.
Initial Testing

- 900 files/min
  - Ok rate. Unclear bottleneck.
- 900 MB/s
  - Good rate, reasonable fraction of the resources for the single data mover node/lustre client
- The Lustre hsm.max_requests setting must be tuned. In the limited test system increasing it from 3 to 6 gave good results.
Future Work

- Much more testing, particularly testing at scale.
- Development and scaled testing of HPSS plugin
- Create HA Robinhood (policy engine) setup
- Workload manager integration
- Cray, Blue Waters, site team is investigating Lustre Bug that requires MDS failover to clear hung transfers
Conclusions

• The initial development and testing of the HPSS/Cray TAS HSM implementation is showing full functionality and good initial performance.
• Challenges remain in crafting effective policies.
• Effective production use will require user education and assistance.
  • Data must be staged before use in a batch job!
  • The lack of a unified quota system will confuse users.
Questions/Acknowledgements

• Supported by:
  • The National Science Foundation through awards OCI-0725070 and ACI-1238993
  • The State and University of Illinois