Research Computing Services @ Cambridge

- University-wide service with broad remit to provide research computing and data services to the whole University
- 700 active users from 42 University departments
- Housed within University’s purpose-built DC - ~100 racks for Research computing
Infrastructure @ Cambridge

- **Darwin**: 600 node (9600 core) sandy bridge
- **Wilkes**: 128 node 256 card Nvidia K20 GPU cluster
- **5 PB** Scratch Lustre (v2.1)
- **20PB** of Tape archival storage
- **CSD3**: New services for Oct 2017
  - **Wilkes2**: 90 node 360 Nvidia Tesla P100
  - **New 768 node** Skylake cluster with Omnipath
  - **342 KNL nodes** - Dell C6320p with KNL7210
  - **5PB** of new Lustre
- **Hadoop & spark**: 200 nodes – Broadwell FDR IB
- **OpenStack**: 80 hypervisor nodes, 3 controller nodes, 1PB replicated storage
Research Data Store

- Persistent storage, designed to provide capacity-oriented, low-cost area for active research data to complement a smaller, performance-optimised scratch area

- 1.4PB Lustre filesystem, with Robinhood for filesystem management

- Tape library with disk-cache as HSM backend
  
  - We run a periodic HSM archival policy over the whole filesystem as a disaster recovery copy
Research Cold Store

- Archival storage area for long term storage
- 200TB Lustre filesystem with continual HSM archival and release policies
  - More aggressive release policy to keep Lustre ‘cache’ from filling up
  - Larger Tape-filesystem disk-cache to improve recovery responsiveness
## HSM Design

<table>
<thead>
<tr>
<th>Tier 1</th>
<th>Tier 1.5</th>
<th>Tier 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lustre</td>
<td>QStar Archive Manager</td>
<td>Spectra Logic T950</td>
</tr>
<tr>
<td>IEEL 3.0.1</td>
<td>- Provides Posix Filesystem interface to Tape library along with 300TB Disk cache</td>
<td>2x Libraries with ~10PB of Tape per DC</td>
</tr>
<tr>
<td>Provisioned through Intel IML</td>
<td></td>
<td>QStar copies data to two tapes - one in each DC</td>
</tr>
<tr>
<td></td>
<td>HSM Copy Agents</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Intel Lemur 0.5.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Run multiple copies as stateless VM instances inside Openstack</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Lustre Hardware

<table>
<thead>
<tr>
<th>Type</th>
<th>Quantity</th>
<th>Specs</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDS Server</td>
<td>2</td>
<td>Dell R630 - Dual E5-2667v3 3.2GHz 8 Core</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128GB RAM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FDR IB, 10GB Ethernet</td>
</tr>
<tr>
<td>MDT Storage</td>
<td>1</td>
<td>Dell PowerVault MD3420</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20x 300GB SAS 15K HDDs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dual RAID controller with 8GB cache</td>
</tr>
<tr>
<td>OSS Server</td>
<td>6</td>
<td>Dell R630 - Dual E5-2623v3 3.0GHz 4 Core</td>
</tr>
<tr>
<td></td>
<td></td>
<td>64GB RAM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FDR IB, 10GB Ethernet</td>
</tr>
<tr>
<td>OSS Storage</td>
<td>6</td>
<td>Dell PowerVault MD3460</td>
</tr>
<tr>
<td></td>
<td></td>
<td>60x 6TB NL-SAS HDDs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dual RAID controller with 8GB cache</td>
</tr>
<tr>
<td>Network</td>
<td>Infiniband</td>
<td>Mellanox SX6036 FDR IB switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>36x 56Gb FDR</td>
</tr>
<tr>
<td>Ethernet</td>
<td></td>
<td>Mellanox SN2410 Spectrum switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>48x 10GbE + 8x 100GbE</td>
</tr>
</tbody>
</table>

- Intel Enterprise Edition for Lustre v3.0.1
- All storage configured in HA-failover pairs
Lustre HSM with Lemur

- Moved over to Lemur in Oct 2016 after initial problem with scaling up lhsmtool_posix
- Our immediate experience using Lemur were a major improvement
  - Lemur multi-threaded architecture enabled us to easily increase data throughput such that we could saturate our HSM backend (~1GB/s)
  - Lemur HSM job throughput was also much faster, could run with much higher setting for ‘max_requests’
  - Very pleasant to manage, single systemd service also manages Lustre filesystem mounts - made it very easy to deploy as stateless Lustre copy agent VMs in our Openstack environment
Lemur Feature Wishlist

• Our main hope is for greater integration with Robinhood in future

• Lemur version 0.5.0 brought compatibility with Robinhood v3 by changing how it stores the file HSM UUID EA - so Robinhood now correctly stores the UUID from HSM_ARCHIVE operations

• We had trouble getting rbh-undelete functionality to work, and have instead written our own simplistic ‘rbh-undelete’ scripts built around the Lemur CLI functionality ‘lhsm import’ released in Lemur 0.5.1

• Lemur has some extremely interesting experimental features such as doing in-flight checksums in the copytool - perhaps this could be stored in the Robinhood DB in some way?
HSM Experiences

(rds-d1): Volume by HSM State
- Current: 150.2 TB
- Synchro: 150.2 TB
- New: 2.0 TB
- Modified: 2.2 TB
- Archiving: 4.8 TB

(rds-d1): Files by HSM State
- Current: 31.32 M
- Synchro: 31.32 M
- Modified: 270
- New: 6 K
- Archiving: 23 K

(rcs1): Volume by HSM State
- Current: 46.3 TB
- Synchro: 46.3 TB
- New: 3.6 TB
- Modified: 73 GB
- Archiving: 685 GB

(rcs1): Files by HSM State
- Current: 8.80 M
- Synchro: 8.80 M
- Modified: 306
- New: 2.14 M
- Archiving: 163 K
HSM Experiences

• Lemur resolved our issues with HSM copytool performance

• HSM coordinator throughput is our main bottleneck now
  • As the number of jobs in the HSM coordinator queue grows large (~few hundred thousand jobs) both the rate at which jobs are fulfilled by the copytools, and the rate at which jobs are ingested, slows down

  • Requires us to closely monitor the size of the coordinator queue and tune Robinhood policy runs as necessary

  • Believe the issue has been noticed last year in LU-7988 and LU-8626

  • We would definitely welcome the suggestion in LU-8626 for a configurable limit on the number of jobs that can submitted to the coordinator queue
HSM Coordinator Wishlist

- As previous LUG/LAD presentations have noted, reducing the value for `max_requests` in coordinator can cause number of `active_requests` to grow unbounded (LU-7920)

- More controls, manipulation options for the coordinator’s queue would be really welcome such as:
  - Not just FIFO - allow configuring ratios of different types of HSM operation so that some amount of high-priority operations (RESTORE) can be submitted to copytools when there is a large backlog of ARCHIVE operations (LU-8324 has a patch for this that we are planning to test soon)
  - Perhaps allow HSM agents to inform the coordinator how many jobs it can process at a time (eg: per-agent `max_requests`) - allow for more dynamic scaling up and down the number of HSM agents as needed.
Status and Future of Lustre* HSM

John L. Hammond
Intel HPDD

May 2017

*Other names and brands may be claimed as the property of others.
Outline

- Review of Hierarchical Storage Management
- Coordinator Issues
- Copytool Issues
- Lemur vs lhsmtool_posix
HSM Coordinator (CDT)

- Part of a Lustre MDT
- Maintains a log on disk of HSM actions (waiting, started, done…)
- Manages layout locking (LDLM) for restores
- Maintains list of registered copytools
- Sends HSM actions (archive, restore, …) to copytools
- Receives progress RPCs from copytools
Coordinator Issue 1: Action Log Scalability

• Submitting new actions is slow when log is already large
  • Adding an action involves linear scan of log for existing compatible action
  • TODO Put in memory hash in front of log (easy)
  • TODO Use a better data structure like index (harder)

• Log update requires linear scan
  • Cache action record locations (LU-9312)

• Compound ID prevents batching
  • Same compound ID assigned to all requests in a list at submit
  • Only requests with same Compound ID submitted together to CT
Coordinator Issue 2: LDLM Locking

LDLM locking used unnecessarily (and inconsistently):

- HSM state needs protection but local locking would suffice
- Avoid interactions between HSM and xattr caching (for example)
- Clients access HSM state through specialized RPCs (not xattrs)
- LDLM Layout Locking still needed for release and restore

Benefits:

- Faster updates (archive start, archive complete, set dirty)
- Updates not blocked by unresponsive clients
(MDT and) HSM Coordinator Progress

- LU-7988 CDT scalability and stability (2 changes in 2.10, 8 in progress)
- LU-9803 handle early requests vs CT registering
- LU-9927 bypass quota enforcement for HSM release (in 2.10)
- LU-9312 HSM coordinator llog scalability (3 in 2.10, 1 in progress)
- LU-9338 cache agent record locations (1 in progress)
- LU-9385 remove XATTR locking from mdt_add_dirty_flag() (in 2.10)
- LU-9403 prevent HSM leak on re-archive (in 2.10)
- LU-9404 set HSM xattr only when needed (in 2.10)
- LU-9464 use OBD_ALLOC_LARGE() for hsm_scan_data (in 2.10)
- LU-9540 cache HSM actions in memory by FID (in progress)
Copytools

lhsmtool_posix
- The “reference” copytool
- Included in lustre-release
- Integrated into Lustre* test framework, HPDD CI

lemur
- New copytool, pluggable, flexible
- Available at https://github.com/intel-hpdd/lemur
- Not integrated into Lustre* test framework
- Does not support archives created by lhsmtool_posix
Copytools

- Attaches to a Lustre client mount point
- Serves one or more backend archives
- Receives HSM action requests from CDT
- Archives file data from Lustre FS to archive
- Restores file data from archive back to Lustre FS

Using reference implementation and llapi you too can write a copytool!
Copytool Issue 1: FID use

- lhsmtool_posix uses file FID as primary id in archive
- An expedient short cut
- Make disaster recovery from RobinHood + archive hard
- Makes import hard (requires rename in archive)
- Incompatible with MDT file migration

- Fixed in lemur by storing HSM archive ID in file extended attribute
Copytool Issue 2: HAL, KUC

CDT sends actions to CT through KUC

- Inflexible message format
- Complicated
  
  LDLM lock cancel -> FIFO write

  CT reads from FIFO

- Actions broadcast to all CTs running on a node

Affects lhsmtool_posix and lemur
Copytool Issue 3: Flow Control

- CT starts a new thread for each request
- No in-CT limits on thread count
- No in-CDT limits on active requests per CT

Too many threads degrades performance...
Many HSM stability improvements in Lustre 2.10 release
More will come in subsequent 2.10.x maintenance releases
Multiple options under consideration for Coordinator performance
Lemur copytool available for community evaluation and input
No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request. Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

© Intel Corporation