Robinhood Experiences: University of Cambridge

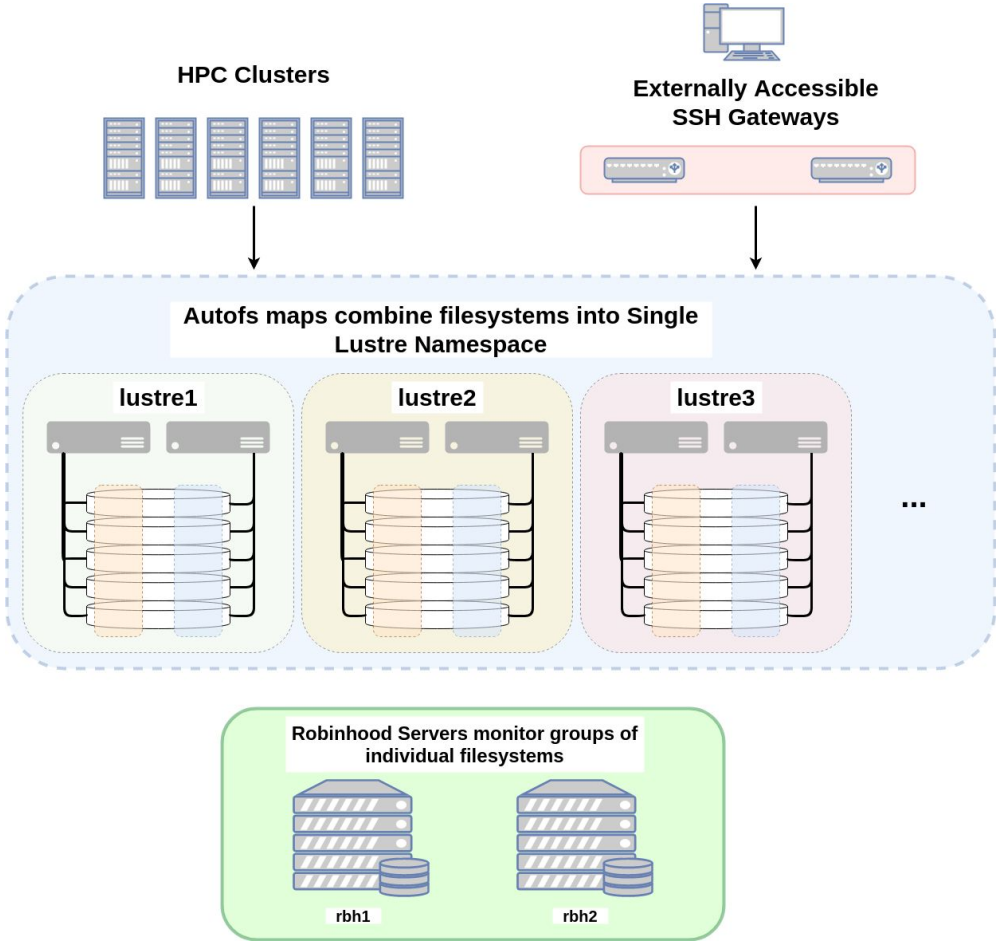
Matt Rásó-Barnett, Wojciech Turek
Research Computing Platforms
University of Cambridge

Site Overview

- ➔ Research Computing Services aims to provide wide-range of Data and Compute services to researchers across the university
  - ◆ Grown out of a HPC-focused team, broader remit
  - ◆ Lustre forms the core storage platform across our services
- ➔ Just over 14PiB of Lustre in production. How can we better utilise this, beyond traditional scratch? How can we make it more resilient?
- ➔ Looking at Data Tiering in two different directions:
  - ◆ ‘Fast’ NVMe SSD storage tier
  - ◆ Tape-based, archival tier
  - ◆ Lustre in the middle as bulk/primary storage tier
- ➔ We’re exploring Robinhood + Lustre HSM for managing tiering data to our Tape-archive
  - ◆ We’ve also explored the idea of archiving a whole filesystem via HSM, as a sort of ‘DR’ recovery copy
- ➔ Robinhood at the centre of our Filesystem analytics → Want to expose information to Users

How we Organise our Lustre

- We create many smaller Filesystems instead of one big Filesystem
  - Majority are between 1 - 2PiB in size, 14 Filesystems currently
  - Filesystem boundaries mostly hidden from users by per-user autofs mounts and symlinks
  - Mix of Lustre versions, trying to move everything to IEEL 3.X (**Lustre 2.7**) currently.
- We can spread Filesystems across multiple Robinhood servers, adding more servers as the filesystems grow
  - Currently have 2x MySQL servers in use by Robinhood, each with SSD-storage and 128GiB RAM
  - Everything on **Robinhood v3**



Robinhood Configuration

- ➔ Everything is on Robinhood v3 from the beginning (no migration, only small experiments with v2.5)
- ➔ Currently still early-experiences:
  - ◆ Only about 1PiB being monitored consistently (spread over only our newer Lustre 2.7 filesystems). Approximately 400M inodes so far.
  - ◆ Haven't yet reached performance problems with multi-PiB, billions-inodes filesystems in Robinhood.
- ➔ MariaDB v5.5.56 - Default release from RHEL7.X. Haven't yet explored whether we should be using something newer?
- ➔ We have 2x DB Servers

- Dell R730xd 2x E5-2630v3 @ 2.40GHz
- 128GiB RAM
- 20x 400GiB Intel SSD DC S3710 in Raid 10

- ➔ Monitor via Graphite+Grafana, trying our best to emulate what others have done here

Cold Storage with HSM

Lustre

IEEL 3.1
Lustre 2.7

Small ~200TiB Filesystem

'User-interface'

Tier 1

Robinhood

HSM Copy Agents

Intel Lemur 0.5.1

Posix-Backend

Run multiple copies as stateless VM instances inside Openstack

QStar Archive Manager

-

Provides Posix Filesystem interface to Tape library along with 300TB Disk cache

QSTAR TECHNOLOGIES

Tier 1.5

Spectra Logic T950

2x Libraries with ~10PB of Tape per DC

QStar copies data to two tapes - one in each DC

Tier 2

HSM Experiences: Lemur Copy Tool

- ➔ Started using Lemur in Oct 2016 after initially using `lhsmtool_posix`
- ➔ Our immediate experience using Lemur were a nice improvement
  - ◆ Lemur multi-threaded architecture enabled us to easily increase data throughput such that we could saturate our HSM backend's ingest rate (~1GB/s)
  - ◆ Lemur HSM job throughput was faster, could run with much higher setting for 'max_requests'
  - ◆ Easier to manage, quality of life improvements. Single lhsmd systemd service also manages Lustre filesystem mounts - made it very easy to deploy as stateless VMs in our Openstack environment
  - ◆ Lemur uses a UUID stored as an extended attribute in the file for structuring the files on the HSM backend instead of using the file's FID which is filesystem dependent.

HSM Experiences: Lemur Copy Tool

Some problems however:

- ➔ Lemur version 0.5.0+ required for any compatibility with Robinhood
- ➔ We've had trouble getting rbh-undelete functionality to work properly with Lemur, and have instead written our own simplistic '`rbh-undelete`' scripts built around the Lemur CLI functionality '`lhsm import`' released in Lemur 0.5.1
- ➔ Worried as Lemur development appears to have stopped, so potentially looking to have to move away or learn some golang :)
- ➔ Lemur has some extremely interesting experimental ideas such as doing in-flight checksums in the copytool - needs development work though to store in the Robinhood DB in some way.

HSM Experiences: MDT / Robinhood

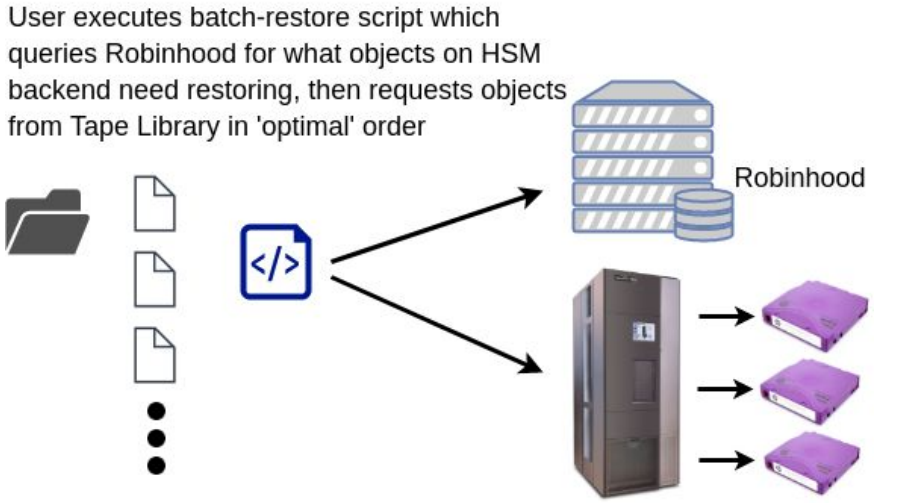
HSM coordinator throughput is our main bottleneck now:

- ➔ 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 dispatched by Robinhood policy run, slows down dramatically. We 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, forcing Robinhood to retry on next policy run instead
- ➔ 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.

HSM Experiences: Batch restore?

- With Tape as HSM backend, we are constantly worried about user's archiving large numbers of small files and then trying to restore them all
- We are looking into how we could develop some batch-restore scripts that would asynchronously restore files from tapes in the most optimal order for the tape library

  Interested in how other sites deal with this problem?

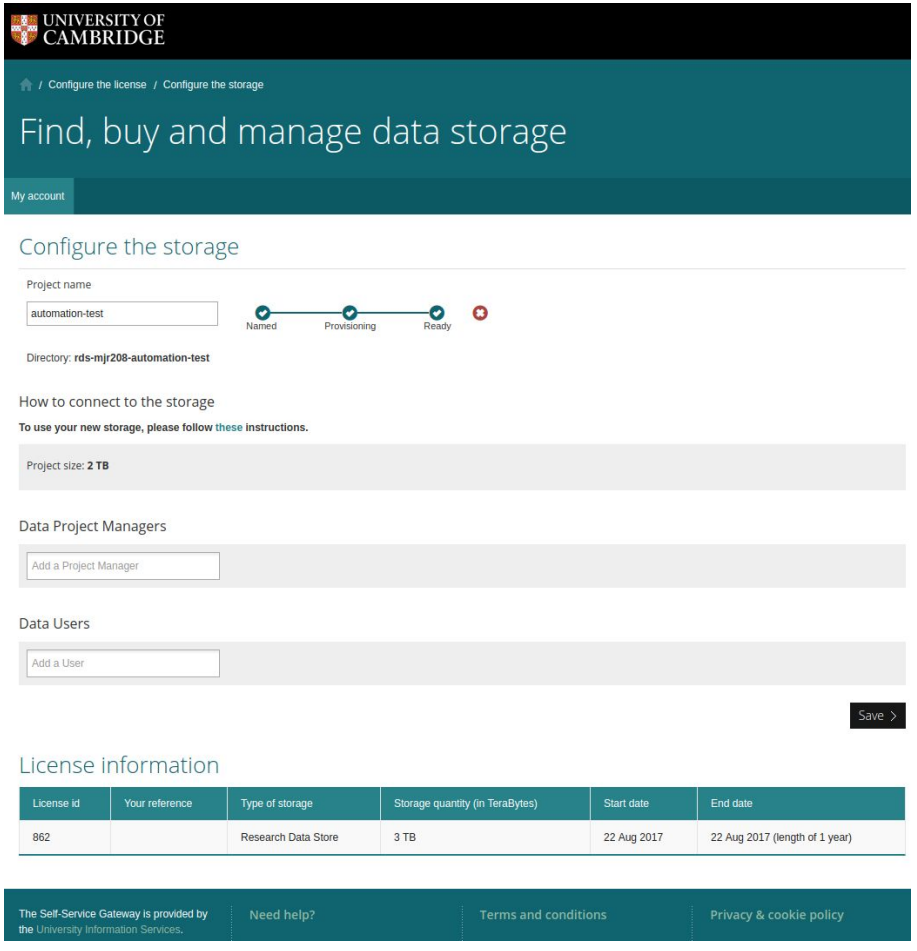


HSM for Disaster Recovery Copy

- ➔ We sell our Lustre to researchers on the basis of 1-copy on disk only. Use other platforms for resilience.
- ➔ We experimented with the idea of using Lustre HSM to make a continual copy of filesystem data as a ‘DR’ copy. If we lost a OST, we could then potentially recover part of the volume however slowly that would be.
- ➔ One of the motivations for the previously mentioned ‘batch-restore’ scripts we are working on
- ➔ It would have helped us if we could have an arbitrary number of HSM backends registered, e.g: one-per OST even, so we could optimise file placement on the HSM backend so files collocated on an OST are always written to the same set of tapes. This isn’t currently possible - upper limit on number of HSM backends
- ➔ Excited about possibility of Lustre 2.11+ File Level Redundancy for this kind of use-case?

Robinhood Stats for Users

- ➔ Other major use-case for Robinhood is filesystem metrics for our Storage portal
- ➔ We've developed a self-service portal for users to purchase Lustre storage space from us and are working to automate the creation of this for users in response to a purchase
- ➔ Working on scripts querying information on a user or project's usage from Robinhood / Lustre quotas and sending this back to the portal
- ➔ Lustre 2.10 project quotas will help a lot with this, but the richer Robinhood information is very valuable to users too



Summary

- ➔ Robinhood and Lustre HSM very important for us
- ➔ Still early-usage, haven't really pushed the limits of the database *yet*
- ➔ Using Robinhood + HSM predominantly in a cold-storage area for Researchers
- ➔ Have struggled a lot with Lustre HSM, still not really where we want to be with it. Have hit a lot of problems along the way, particularly with the MDT coordinator
- ➔ Interested in how other sites using Lustre HSM manage it's weaknesses?
- ➔ Interested in other sites' Robinhood monitoring scripts (collectd,graphite or otherwise)