energie atomique • energies alternatives

Lustre/HSM Binding

Aurélien Degrémont

aurelien.degremont@cea.fr

Agenda

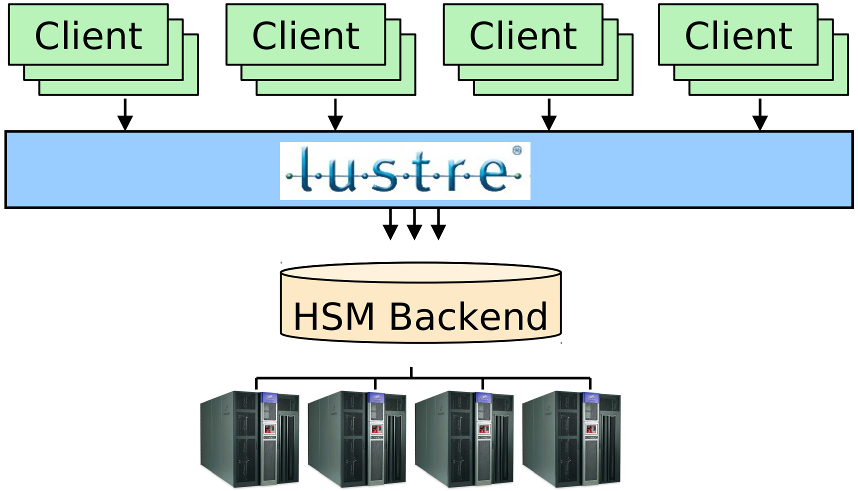
- Project history
- Presentation
- Architecture
- Components
- Performances
- Code Integration

Project History

- **2007 – CFS times**
  - Never ending Architecture
- **2008-2009 – Sun era**
  - Designing and Lustre internals learning
- **2010 – Oracle times**
  - Coding, hard landing
- **2011 – Nowadays**
  - Debugging, Testing, Improving

Presentation (1/2)

- **HSM seamless integration**

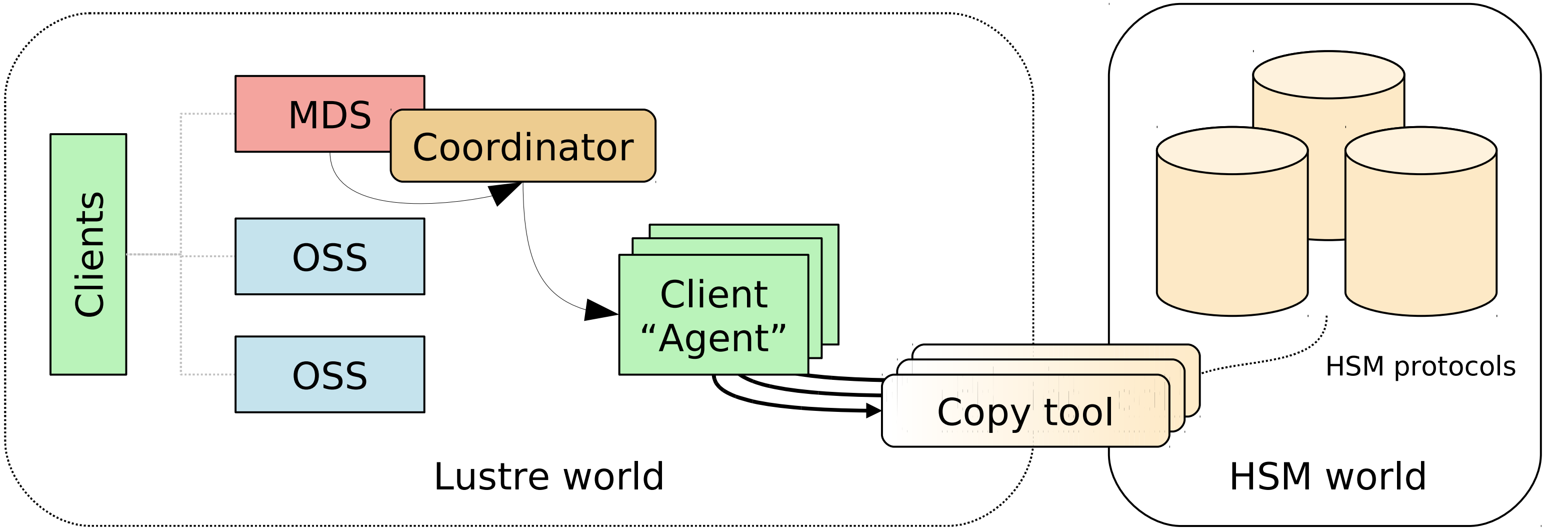


- **Takes the best of each world:**
  - Lustre: high-performance disk cache in front of the HSM
    - parallel cluster filesystem
    - high I/O performance, POSIX access
  - HSM: long-term data storage
    - Manage large number of disks and tapes
    - Huge storage capacity

Presentation (2/2)

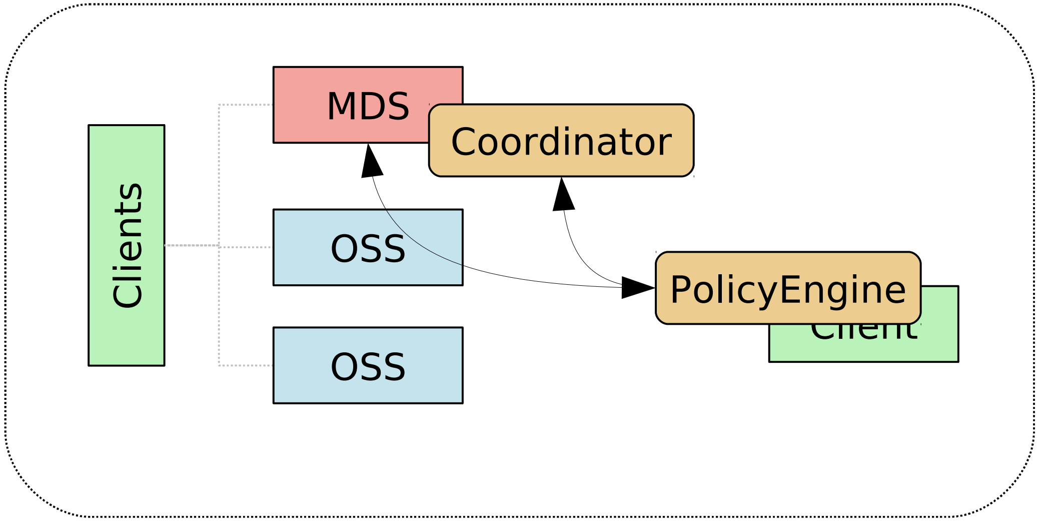
- **Features**
  - Migrate data to the HSM
  - Free disk space when needed
  - Bring back data on cache-miss
  - Policy management (migration, purge, soft rm,…)
  - Import from existing backend
  - Disaster recovery (restore Lustre filesystem from backend)
- **New components**
  - Coordinator
  - Archiving tool (backend specific user-space daemon)
  - Policy Engine (user-space daemon)

Architecture (1/2)



- **New components: Coordinator, Agent and copy tool**
  - The coordinator gathers archiving requests and dispatches them to agents.
  - Agent is a client which runs a copytool which transfers data between Lustre and the HSM.

Architecture (2/2)



- **PolicyEngine manages archive and release policies.**
  - A user-space tool which communicates with the MDT and the coordinator.
  - Watch the filesystem changes.
  - Trigger actions like archive, release and removal in backend.

Component: Copytool

- **It is the interface between Lustre and the HSM.**
- **It reads and writes data between them. It is HSM specific.**
- **It is running on a standard Lustre client (called Agent).**
- **2 of them are already available:**
  - HPSS copytool. (HPSS 7.3+). CEA development which will be freely available to all HPSS sites.
  - Posix copytool. Could be used with any system supporting a posix interface, like SAM/QFS.
- **More supported HSM to come**
  - DMF
  - Enstore

Component: PolicyEngine Robinhood

- **PolicyEngine is the specification**
- **Robinhood is an implementation:**
  - Is originately an user-space daemon for monitoring and purging large filesystems.
  - CEA opensource development: http://robinhood.sf.net
- **Policies:**
  - File class definitions, associated to policies
  - Based on files attributes (path, size, owner, age, xattrs…)
  - Rules can be combined with boolean operators
  - LRU-based migr./purge policies
  - Entries can be white-listed

Component: Coordinator

- **MDS thread which "coordinates" HSM-related actions.**
  - Centralizes HSM-related requests.
  - Ignore duplicate request.
  - Control migration flow.
  - Dispatch request to copytools.
  - Requests are saved and replayed if MDT crashes.

energie atomique • energies alternatives

File states

- **View file states**
  - lfs hsm_state <FILE>

```
$ lfs hsm_state /mnt/lustre/foo
/mnt/lustre/foo
        states: (0x00000009) exists archived
```

- **New file states**
  - Archived
    - A copy exists in the HSM
  - Dirty
    - File in Lustre was modified since it was archived
  - Released
    - No more data, same size, no block

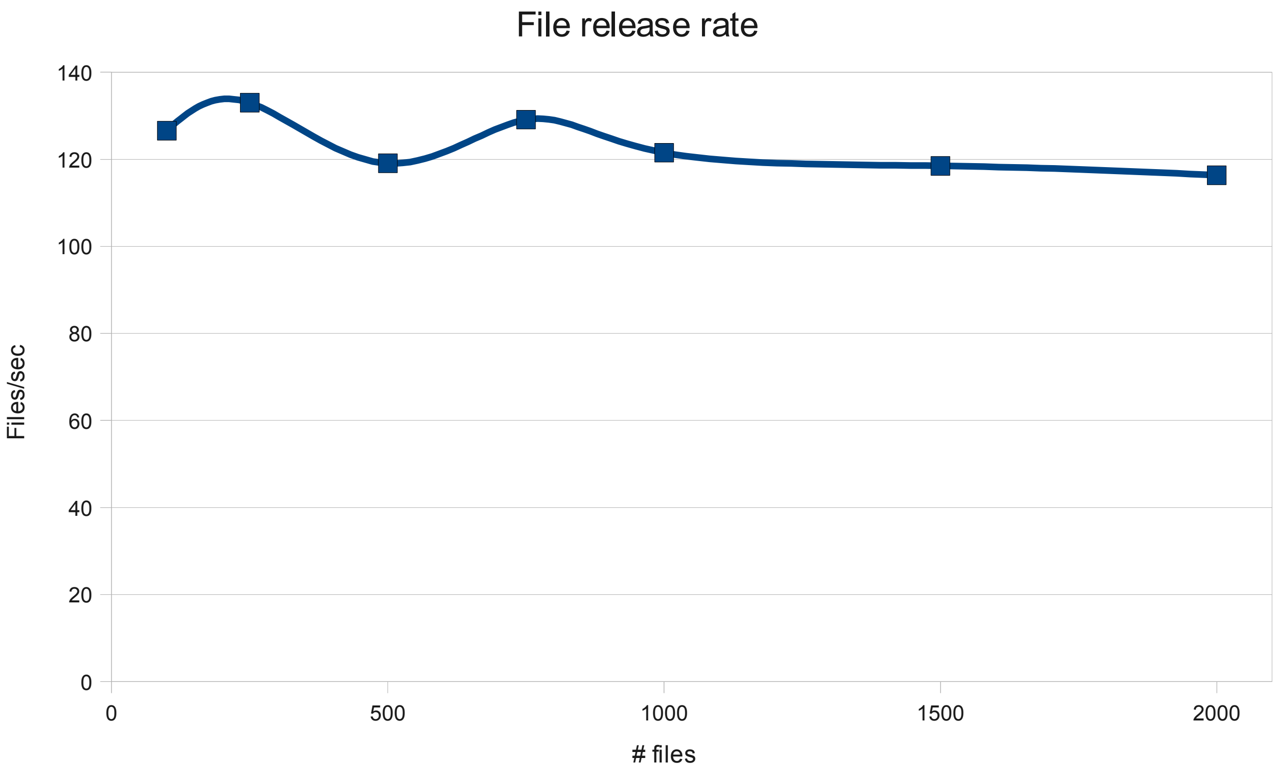
```
# stat /mnt/lustre/foo
  File: `/mnt/lustre/foo
  Size: 409600          Blocks: 0          IO Block: 2097152 …
```

Early performance

- **Standard use**
  - Based on Lustre MDT Changelogs. Low impact on MDT.
  - Introduce Layout lock. Very low RPC overhead.
- **Simple testbed**
  - Plateform
    - HW: 2 Dell R710 – 4 Xeon X5650 @ 2.6 GHz – 40 GB
    - 3 VM – 12 CPU – 8GB
  - Filesystem
    - 1 server: 1 MGS/MDS, 2 OSS
    - Few clients
  - HSM Backend
    - Local /tmp filesystem

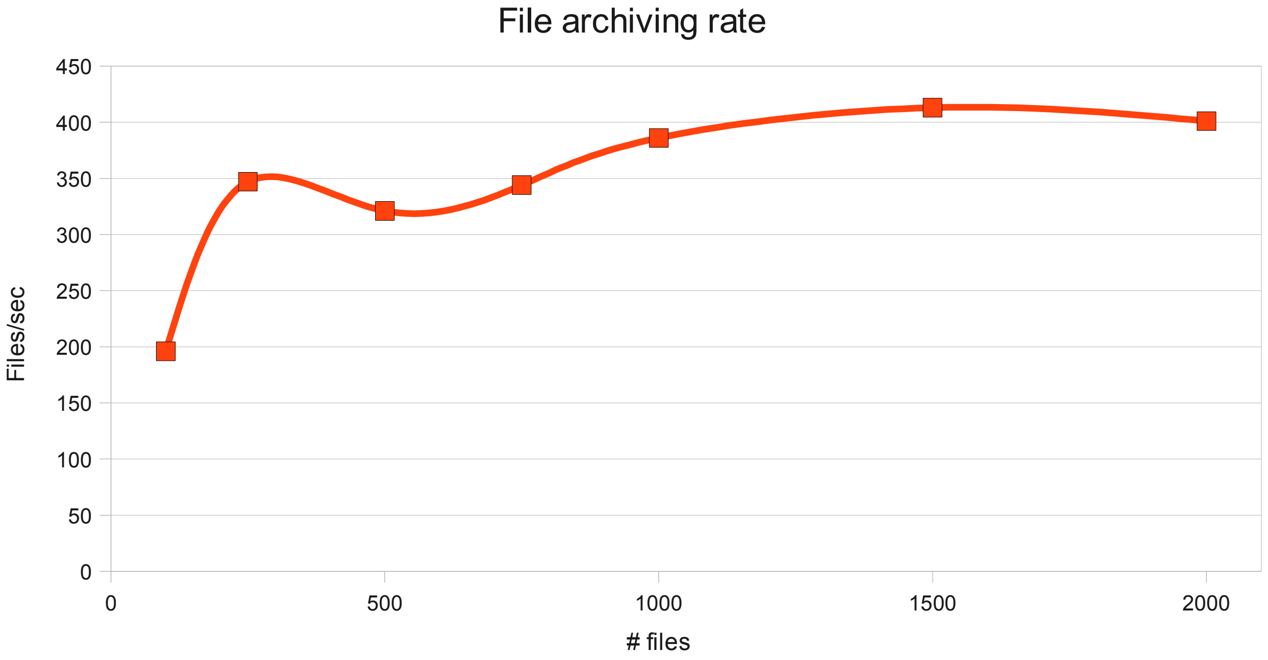
Early performance

- **Release**
  - Synchronous data purge



Early performance

- **Archiving**
  - Copy files from a Lustre client to a local ext3 filesystem
  - More than 1 million archives per hour



Learn Lustre

- **Lustre/HSM project uses a lot of different parts of Lustre code.**
- **To develop this project, we needed to learn each of them**
- **HSM patches:**
  - Infrastructure
    - RPC
  - Layout lock and grouplock
    - LDLM, CLIO
  - Release
    - LDLM, CLIO, OST object management, MDT layering
  - Coordinator
    - Llog

Code Integration

- **Oracle 2.1 branch (R.I.P.)**
  - Few first patches were landed in Oracle 2.1 branch
  - All landing effort was stopped due to Oracle Lustre policy
- **Lustre 2.1 Whamcloud/community release**
  - Target is to released to a stable version quickly
  - Do not take risk to delay it due to HSM patches
- **So when?**
  - Will start code landing as soon as 2.2 branch is available

energie atomique • energies alternatives

Thank you.
Questions ?

Robinhood: example of migration policy

- **File classes:**

```
Filesets {
     FileClass small_files {
         definition { tree == "/mnt/lustre/project“ and size < 1MB }
         migration_hints = "cos=12" ;
…
     }
}
```

- **Policy definitions:**

```
Migration_Policies {
     ignore { size == 0 or xattr.user.no_copy == 1 }
     ignore { tree == “/mnt/lustre/logs” and name==“*.log” }

     policy migr_small {
               target_fileclass = small_files;
               condition { last_mod > 6h or last_copyout > 1d }
     }
     …
     policy default {
               condition { last_mod > 12h }
               migration_hints = “cos=42”;
     }
}
```

Robinhood: example of purge policy

- **Triggers:**

```
Purge_trigger {
   trigger_on = ost_usage;
   high_watermark_pct = 80%;
   low_watermark_pct = 70%;
}
```

- **Policy definitions:**

```
Purge_Policies {
   ignore { size < 1KB }
   ignore { xattr.user.no_release = 1 or owner == “root” }

   policy purge_quickly{
         target_fileclass = classX;
         condition { last_access > 1min }
   }
   …

   policy default {
         condition { last_access > 1h }
   }
}
```