DE LA RECHERCHE À L'INDUSTRIE

SHORT AND MID-TERM ROADMAP FOR ROBINHOOD

Henri DOREAU | CEA/DAM
henri.doreau@cea.fr

CEA, DAM, DIF, F-91297 Arpajon France

www.cea.fr

RUG2015

OUTLINE

Improvements and changes to expect in the future

- v3.1 Beyond generic policies: enhanced workflows

- V3.x Roadmap overview
  - Improved modularity
  - Performance improvements
  - Richer plugins ecosystem
  - Support object stores

- V4 Alternative model for a distributed policy engine

ENHANCED WORKFLOWS
V3.1 (Q2 2016)

ASYNCHRONOUS ACCOUNTING

Non-atomic accounting updates

- Currently: atomic update of accounting with inode information
  - Noticeable performance impact (if accounting enabled)
  - Prevents from using batching & multithreading together (deadlock)

- In 3.1: deferred processing
  - Change descriptors enqueued into a FIFO table (nolock)
  - Accounting not updated right away
  - Records dequeued asynchronously
  - Processed by a dedicated thread pool (synclet)

=> Significant speedup
=> Ability to distribute this work (remote synclet)
=> Ability to use a different backend (remote system)

PLUGINIFY ALL THE THINGS!

Generalized use of plugins

- Criteria, statistics, metrics and associated reporting
  - Usage and patterns per job
  - Activity tracking per user / group
  - …
- Triggers (run policies when a condition is met)
  - Existing ones converted to new modules
  - Allow vendors/users to develop/configure custom ones
  - *e.g.: run data integrity checking when other policies are idle*
- Alerts
  - *e.g.: raise alert if name contains non-printable characters*
  - Multi-steps alerts (*NOTICE / WARNING / CRITICAL…)*

ENHANCED LUSTRE/HSM SUPPORT

Significant HSM improvements

- Action rate smoothing/leveling
  - Avoid huge bursts of action per pass
  - Rate-limited actions
- Cray’s HSM/Migrate support (**LU-6081**)
- Disaster recovery
  - When losing OST
    - Identify the impacted files
    - Take appropriate actions
    - Reimport from archive (delete / recreate / rebind if needed)
    - Indicate which files have been restored to latest version, to an older version or definitely lost
  - When losing MDT
    - Re-create the namespace from robinhood database
    - Re-import files from backend
  - Make Robinhood able to control the copytools for *rebind* operation

MID-TERM ROADMAP
V3.x (2016~2017)

3.x CODE IMPROVEMENTS

Code structure improvements

- Group lustre-specific code
- Adapt robinhood to various object stores
- Improve test suites
  - Merge the existing ones
  - Provide plugins a way to nicely integrate dedicated tests
- Switch to liblustre (the new LGPL one)
  - API cleanup
  - Perf improvements (keep open FDs on filesystem root and OBF directory…)
  - Work by CRAY tracked under **LU-5969**
  - See: https://github.com/fzago-cray/liblustre

3.x PERFORMANCE IMPROVEMENTS

Initial scan speedup

- Low-level MDT scans (lustre-side change)
  - Posix scans against billions of inodes require patience…
  - Iterator over the MDT inodes table, exposed as a pseudo changelog stream
  - Goal is to significantly reduce scan duration
  - Work by CEA

1. Enable changelogs
2. Issue changelog_start w/ a special CHANGELOG_FLAG_SCAN flag
3. MDS uses an OSD iterator to retrieve inodes from inodes table
4. Packed and delivered as pseudo-changelog records
5. Current scan state saved using a CHANGELOG_USER_REC record (i.e.: the scan can stop/resume)
6. Process regular records to include changes that happened since 1)

POSTGRESQL SUPPORT

Make the list manager compatible with postgreSQL

- New flavor of list manager
- Aim to have a complete, production-grade support
- Should ease experiments with clustering solutions (rbhv4)
  - Bi-Directional Replication (BDR)
  - pg_shard

DISTRIBUTED BACKEND

Two-phases roadmap

- Modular list managers
  - Redefine / refactor the list manager API
  - Convert the existing (MySQL) list manager to that API
  - Distribute it as a shared library (like rbhv3 plugins)

- Distributed backends
  - Identify suitable technology (proper consistency guarantees!)
  - Implement a new backend

Again, modular design to allow vendors/users to implement and distribute custom adapters for the backend of their choice

WHICH DISTRIBUTED DBMS TO CHOOSE?

Identified candidates

- PostgreSQL (BDR or pg_shard)
  - Based on extremely mature technology
  - Interesting features being developed (JSON as a native type…)
  - … but we have not tested it yet

- Elasticsearch / HBase
  - Excellent scalability
  - Relatively low ingest rate for ES / Very high for HBase
  - Heavy machinery

- MongoDB
  - Document oriented (a row is a JSON document)
  - Transparent sharding
  - Supports concurrent writes
  - … but has shady corner cases (**can loose acknowledged writes on a network partition!**)

ALTERNATIVE MODEL V4

DISTRIBUTED STREAM PROCESSOR

Experiments with a new approach

- Existing model: large database
  - Static data
  - Distributed storage and processing

- Studied model: stream processing
  - Moving records
  - Data in memory
  - On the fly processing (graph of operators)

- Requires a compact representation of the system (Flajolet-Martin Sketch, StreamSamp, DenStream clustering…)

- Ongoing experiments with Apache SPARK framework

PROSPECTIVES

Expected improvements in virtually all areas

- Flexibility
  - Modular components as much as possible
  - Encouragement for contributors to develop their own modules
- Performance
  - Keep the product ready for next generations of machines
  - Vertical / Horizontal scalability effort
- Stability
  - Sustained effort on stability
  - High code quality standards (gerrithub reviews, discussed design changes)…
  - Regression testing
  - Early deployment of beta versions

WANT TO HELP?

https://github.com/cea-hpc/robinhood

robinhood-devel@lists.sf.net

THANK YOU!

ANY QUESTION?

Commissariat à l’énergie atomique et aux énergies alternatives
CEA/DAM Ile de France | 91297 Arpajon Cedex
T. +33 (0)1 69 26 40 00

Etablissement public à caractère industriel et commercial | RCS Paris B 775 685 019

DAM Ile de France