dCache.org

dCache, a QoS/DLC introduction

Paul Millar

INDIGO-DataCloud WP4 virtual meeting (2015-08-28)

Fermilab NDGF Nordic DataGrid Facility DESY HELMHOLTZ ASSOCIATION LSDMA INDIGO DataCloud

Current status: overview

- QoS is controlled by:
  - **directory** file is written into
  - Some **request** influence (SRM)
- QoS choices are limited to:
  - **Disk** or **Tape** (or a sort-of Disk+Tape),
  - Choice of hardware, based on initial data-placement
  - Permanent vs Volatile storage
- DLC is limited to **pinning** "restored" data.

Current status: file data in dCache

- File's data is largely **independent** of namespace entry.
- Two types: **cached** (can be GC) and **precious** (should go to tape).
- Cache data can be protected from GC by adding **sticky records**
- Commonly seen examples:
  - D1T0 (cached, but with infinite sticky record),
  - D0T1 (cached copy exists initially, but without any sticky records),
  - "D1T1"-like (stored on tape; disk copy with infinite sticky record),
  - Cache-copy (cached without stick records, created for load-balancing),
  - Pinned D0T1 (file brought back from tape and protected for some period)

Current status: Volatile storage

- Storage-nodes can participate in volatile storage
- File data written to volatile storage is **not** protected from garbage collection.
- When the last copy of a file's data is removed, the file is removed from the namespace.

Current status: file placement

- Where the data ends up is completely determined by which **directory** the file is written in:
  - Partition available disk storage nodes (ownership, QoS),
  - Write to one of the available HSM systems,
- Supports caching reads on faster media (e.g., SSDs and spinning disks), with two modes:

  Write-around (write to slow media, cache when read)
  Write-back (write into fast media, then copy to slow media)

Current: with SRM

- User can specify, *per file*, whether it should be written to disk or tape.
- Same options as before (disk, tape, disk+tape)

  ONLINE → add permanent sticky record.

  CUSTODIAL → goes to tape.

Simple DLC: tape → tape+disk

- Supported for Tape → Tape+Disk
- On-disk copy is "pinned" for period
- Once pin released, file may still be available with ONLINE latency, but could be GC.

Limitations: edge-triggered

- File is disk+tape and-then storage component hosting file goes down → file available tape only.
- Storage-node with pinned file goes down, nothing happens.
- If pool "dies" and GC-able copies of file exists.
- Admin changes where files should end up

  (e.g., pool decommissioning)
- File stored in "backup" storage-node, then main storage-node comes back online.

New approach: overview

- Three layers:
  - User-interface (upper layer)
  - DLC layer (middle layer)
  - File Placement Engine (low level)

New approach: File Placement Engine

- Files have arbitrary set of tags

  online, replica, atlas:default, ATLASMCDISK, …

- Storage-nodes also have tags

  atlas, tape, rack-12, room-A, fast, …

- Operations have tags:

  write, GridFTP, WAN, swedish-door, …

- Set of rules in propositional logic links these three kinds of tags
- Solver gives possible assignments to propositions (storage-node, priority)

New approach: how FPE will be used

- Given a transfer, select pools
- Given change to rules, which pools/files affected?
- Given change in available pools, what files need moving?
- Given change in file tags, which pools can't host the file?
- Given a new file where should it be located?

New approach: DLC

- List of DLC policies, files with those policies,
- Receives triggers:

  passage of time, (file popularity?, …)
- Updates file according to policy:

  Modify QoS (updates tags on files),

  Updates ACL,

  Triggers transfer? Delete file?

New approach: user-interface

- Accepts requests in CDMI
  - Operates on files (in namespace)
  - Updates the "tags" on files.
  - Modifies DLC for files.
- Depends on outcome from RDA and SNIA
  - RDA defines the abstract model,
  - SNIA defines mechanism via CDMI.

Backup slides