Namespace, authorization, quotas and catalogues

Philippe Charpentier
LHCb / CERN

Disclaimer

- This presentation is here for triggering discussion
  - It's not a requirements' document!
- As the outcome of the other topics is not known yet, it's hard to present a fully consistent proposal
  - WARNING: DM in itself is not sufficient. DM is strongly coupled to WM!
    - We need a full consistent Grid architecture (after 10 years…)
- Statements are (strongly) biased by our experience (LHCb) and implementation
  - We would like to push to the DM middleware stuff we had to implement ourselves
    - Can serve as examples / prototypes
- *Acknowledgements to Andrew Smith and Jean-Philippe Baud for very fruitful discussions*

Files, datasets, GUIDs

- Need for a “logical” namespace
  - Hierarchical: path-like
    - Advantages: data clusterisation, “directory” handling, authorization inheritance…
  - Flat space: needed for file reference within files
    - Typically GUID is fine
- Logical namespace → catalog (see later)
  - Currently namespace linked to archive tapeset
- Datasets should be integrated from the beginning
  - Definition as a set of files (or directories)
  - Is it enough to have directories (possibly with “symlinks”) in catalog?
- Still need to partition storage space?
  - Should current service classes be retained
    - Completely decoupled from namespace (just different SEs)
      - Coupling implies change of name when changing service class
  - Don’t want excess of ESD to preclude raw data storage, or user files to prevent AODs: use quotas?

Catalog

- Purposes
  - Keep track of where files are
    - Needed whatever the technology is!
  - Map logical namespace to physical namespace
- Physical namespace : URL
  - `<protocol>://<endpoint>/<SAPath><path>`
  - Currently the whole URL is stored (e.g. SURL)
  - Rather record: `<protocol>, <SEName>` and `<path>`
  - `<SEName>` should define `<endpoint>, <SAPath>` and the URL construction should be simple (depending on `<protocol>`)
  - Should multiple {protocols, endpoint} be implemented?
  - Should archive information be stored (archiveset)?

Catalog (cont'd)

- Keep it simple!
  - Minimum set of metadata, e.g.
    - File size, check sum, creation date
    - For replicas: SEs, creation date, flags
  - Datasets catalog outside the scope (too much expt-dependent)
- Replica flags
  - File availability (for temporary outages), accessibility cost, master replica…
- SE flags
  - SE availability (easy to "hide" replicas when an SE is down)
- Datasets
  - Not dataset catalog! Only dataset composition (files / directrories)
  - Easy to implement with symlinks to files or directories
- Scalability problem:
  - Is there a need for hierarchical catalogs?
  - How to guarantee their consistency?
  - Consistency with storage
    - Automatic notification when files are unreachable!

Authorization

- Assume X509 is _the_ credential mechanism
- Access Control Lists (ACLs)
  - Should be implemented once! (at the catalog level?)
  - Important implication: no backdoor file access!
  - Warning! Implementation should scale…
- Regular ACL structure
  - File ACL: restricts usual operations (r,w,d), possibly also more complex operations: replicate, recall from tape (useless if no tapes ;-)
  - Directory ACL: file ACL + default ACL for files
- ACL to whom?
  - Individuals: DN
  - Specific groups: use VOMS roles (FQAN)
- For those who remember: OpenVMS ACL's were just great!

Quota

- Depends on the WAN file system
  - Users should only be accounted (quotas) for what they have control of!
  - Quota on space for files? Space for replicas?
- If global file system: implement quota on catalog!
- Quotas based on DN (user files) or FQAN (general use files)

Data management and jobs

- Strong coupling between DM and WM!
  - Currently
    - DM imposes sites for running jobs
    - Need for a prestaging system (tape recall prior to submitting jobs)
    - Jobs run where a file is “online”
  - With a WDM (Wordwide Data Management)
    - WM should evaluate the cost of replicating files w.r.t. running jobs where files are replicated
      - What is the cost metrics?
    - File caching policy
    - Multiple replication at a single site for hot files

Are we so far from that now?

- Conceptually the answer is probably NO
- Operationally, the answer is probably YES, but…
- Catalog requirements
  - LFC and AliEn-FC seem to meet most requirements
  - Missing in LFC: easy SE redefinition
    - Worked around in LHCb with a static SE definition (no real use of the SURL)
  - ACLs are there, but multiple backdoors exist and should be closed
    - SRM action (and ACLs!)
    - Storage action (and ACLs or equivalent)
      - nsrm can ruin a catalog!

Far? (cont'd)

- Quotas are not there but…
  - Easy to implement on top of catalogs (should be embedded)
    - LHCb has a quota system for user files (based on replicas from the LFC)
- Desperately missing:
  - Information on unavailable files
    - E.g. file server offline, files "lost"
  - SE accounting
    - du like utility on Ses
    - Can be implemented on top of FC, but costly

Conclusion?

- Global namespace
  - Hierarchical and flat
- Central catalog for file location, metadata
  - ACLs, quotas
  - Can we shut the backdoors?
- Current system (almost) allows this
  - What's not so good is the implementation
    - SRM (too heavy for little add-on)
    - Hardware implementation
      - Number of spindles, servers for matching CPU
    - Failure recovery (application access layer)