Data Caches on the OSG

Brian Bockelman

WLCG Workshop, June 2017

Why Do We Care?

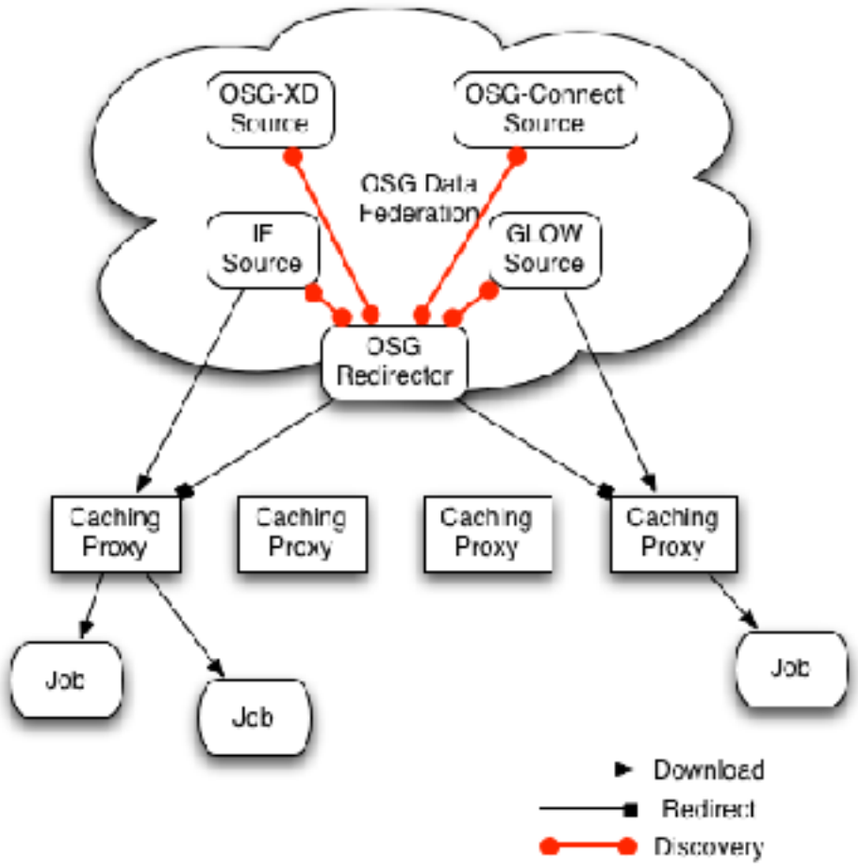
- Storage is hard!
  - Storage elements introduce state (“the files at my site”); state requires significant effort to maintain.
    - SEs require metadata, they require synchronization with external catalogs, file loss is an exceptional event, etc.
- Caches are easy! (maybe)
  - Not expected to maintain state.
  - File loss — aka, eviction — is a normal event

Ok, maybe not so easy!

- There are many hard parts of caching:
  - **How big should the cache be?** A thrashed cache is worse than no cache at all!
  - What is the minimum performance for the cache?
  - How should caches be authenticated?

OSG Work in Caching

- OSG has been working on a caching infrastructure ("StashCache") based on Xrootd's caching proxy.
    - Will retrieve data from a data federation "origin" using an external Xrootd client.
        - Data is exported via any protocol supported by Xrootd (HTTPS or Xrootd).
- Caching infrastructure takes brunt of data distribution, not the origin server.
- Initial version was only for public data. We closely worked with the origins to make sure working set size is smaller than cache size.
    - Solving technical problems via clever definition of scope…



Building on StashCache

- Public StashCache serves out O(PB) / week.
- However, plenty of data (LIGO) is non-public.
- The current version of StashCache adds support for auth'n / auth'z in the namespace.
    - Requires clients to have HTTPS-based authentication to access certain files.
    - In this case, proxies themselves must be authorized to download from the origin.

Building on StashCache

- We provide a command-line client, `stashcp`, for accessing StashCache.

- However, some applications cannot predict which files they need - or can't be adopted to use `stashcp`. They want a POSIX-like interface!
    - Data access is already scalable: need scalable metadata access.

- Solution: CVMFS! We extended CVMFS so it can read metadata from existing CVMFS infrastructure and data via StashCache.
    - Added support for X509-based authorization in CVMFS.
    - Did a demo in 2016 publishing 1PB for CMS.

USCMS Work

- UCSD and Caltech are working on large-scale proxy cache setups (multi-hundred-TB).
    - Goal: host a full set of CMS MINIAOD data tier in Southern California.
    - Uniform cache-based access.
    - Can either run on top of existing SE or independent JBOD-style hosts.
- This project is working on both the *scale* issues and *integration* with the CMS job submission infrastructure.
    - Should be regularly used later this summer.
    - Idea is to scale this down into something for smaller sites.

Conclusions

- We have two interesting demos centering around the Xrootd caching proxy:
  - CMS is probing scale in total volume and integration with complex job infrastructure.
  - OSG is probing CVMFS-based access and multi-VO authentication setups (maybe non-GSI-based in the near future?). Smaller overall working set size. Partnering with LIGO.
- Is this accessible for a generic “small site” with a few tens of TB?
  - Unclear for CMS: can efficiently access via remote IO.
  - Majority of our production workflows are streaming-based (not cache friendly).
  - Non-streaming-based workflows are very IO-intensive: not great for “small sites.”
  - Analysis is a good use case (repeat use, moderate IO requirements), but requires moderately-large working set size (few hundred TB).
  - Tests by the UCSD / Caltech team will provide illumination this year!