Cache and storageless sites

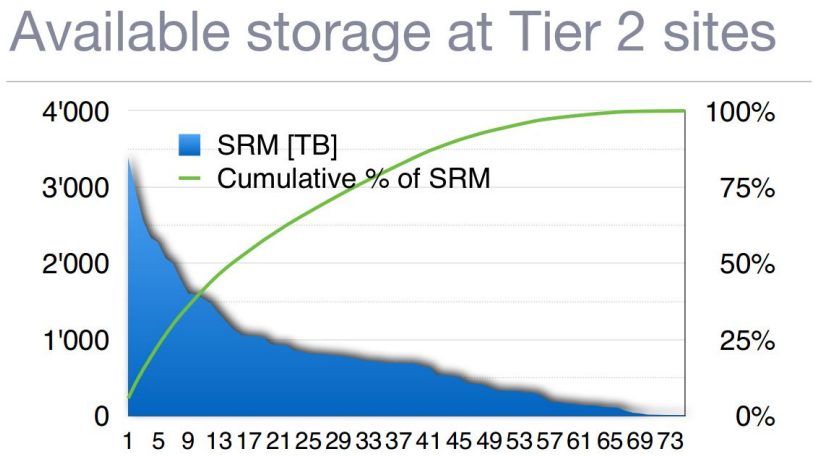
Site Jamboree
Cedric Serfon (Oslo)

Introduction

- Some sites have very limited storage/manpower
- From operational experience, the small sites are the ones that generate most of the problems/operational load :
  - Lost files
  - More Dark Data than bigger sites
  - Sometimes have to reduce space, change SE hostname/path...
  - Sometimes not very responsive
- The effort to operate these storages isn't worth the benefit

Storage less cache, why ?

- ATLAS wants to avoid sites with small disk space (<300 TB) for the operational reason mentioned on the last slide
- See Eric's talk yesterday :



Different type of cache

- Secondary files :
  - Files residing on normal Rucio Storage Element but can be deleted whenever space is needed
  - Deletion based on LRU logic done by Rucio
  - Not the purpose of this talk
- “Internal cache”, i.e. cache that is only accessible from the site.
  - For local jobs
  - Not registered in Rucio
- Cache site
  - Can be accessed from the WAN
  - Needs to have the data registered in Rucio. Will help for brokering
  - There might be some inconsistencies between the cache and the catalog

Cache in ARC

- Cache is used for a long time in Nordugrid
- ARC CE has a built-in cache
  - It stores files that are used as input on a given CE
  - The cache is usually on a shared file-system accessible from the WN
  - The space is managed by ARC CE, deleting least recently accessed files when space is needed
  - Typical size for an NDGF-T1 site: 100TB
- By default “internal cache”, i.e. not accessible from the outside world
  - For the moment completely hidden from ATLAS systems
- This cache model can only be used with the “full” ARC CE setup
  - i.e. aCT submitting pre-defined payloads and ARC CE doing data staging
  - Not appropriate for “ARC as a pilot gateway”

Rucio cache

- Rucio allows to have cache storages, i.e. a storage area that might not be as consistent as a normal Rucio Storage Element.
- Rucio keeps the list of files that are likely to be in the cache.
- Some process needs to update Rucio regularly with the files in the cache :
  - For insertion of new files
  - For deletion of files
- The synchronization is based on activeMQ messages sent by the cache that are consumed by a Rucio daemon
  - The daemons and clients to update the cache information are already there.
  - Missing the part on the cache site that report to Rucio

Rucio cache

- Possibility to define a validity of the replicas (e.g. the replicas on Rucio cache sites expires after x days)
- No Rucio cache sites available yet
- Will be defined in AGIS :
  - Marked as volatile
  - Name convention the sites to be defined (e.g. sitexyz_CACHE).

Integration of ARC cache in Rucio

- Implemented recently.
  - After an initial bootstrap, one process that runs regularly on the CE send the diff to the Rucio cache
- Will be tested then put into production in the coming weeks
  - Will allow brokering of panda jobs to sites where files are cached
  - No problem if files are deleted by the time the job gets there, ARC will download again

Xrootd cache

Xrootd Cache R&D at MWT2

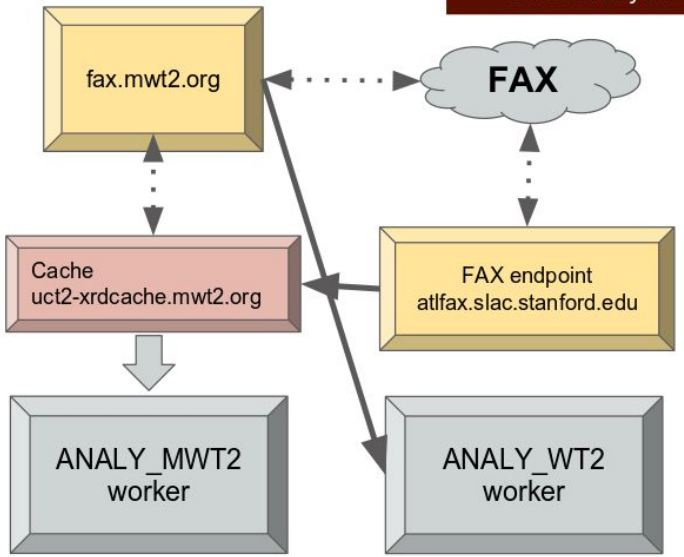
Ilija Vukotic
Wei Yang
Andy Hanushevsky
Lincoln Bryant

Caching machine:

- 26TB of RAID6 HDD
- 318GB SSDs
- 10 Gb NIC
- 12GB RAM

Test Configuration:

- Changed translation service so remote accesses use MWT2 caching server. When cache missed, uses FAX endpoint to get the data.
- Overflow jobs running at other sites still use our FAX endpoint.
- Failover jobs work as before.



Xrootd cache

Caching algorithm features

- Partial file (block based) caching, on-demand
- Only downloads the requested fixed-size blocks of a file (we set it to 1MB)
  - Serves them to client from memory.
  - Does not begin prefetching until a read request is actually received; a check is made if data required to fulfill the request exists on disk; if it doesn't, the required blocks get queued for download.
- Downloaded blocks are stored with the same filename as the original file. The cached file is sparse.
- For each downloaded file there is a metadata file containing:
  - Position of blocks already cached
  - Number of times file was accessed
  - Times of first and last access
  - Bytes requested

Squid cache

- Possibility to deploy a squid cache on the site :
  - If a job needs a file it submit a GET request the rucio redirector that will redirect it to the file.
  - The requests are intercepted by Squid and all responses are cached.
- AFAIK, never tested. Volunteers ?

SRMless access to the cache

- Don’t need to setup SRM to access caches.
- 2 prefered protocols :
  - Xrootd
  - WebDAV

Storage-less sites

- Simplest solution : one site with only CPU getting/exporting data from a remote well connected site.
- As Alastair mentioned yesterday, many T2Ds in UK will decommission their storage.
- DDM ops will help in defining procedures to ease the decommissioning of these sites