# Storage Development Progress

Robert Illingworth

International Computing Advisory Committee

9 February 2022

# Storage Development Progress Topics

- CTA
- Improving tape media migration
- Staging performance
- Contributions to common storage projects
- Data management and Rucio
- Ceph
  - I won't discuss this one further, except to say we now have effort available and we have recently started work to investigate possible uses

# New effort

- New hires for storage R&D effort
- Ren Bauer – August 2021
  - CTA development and Enstore support
- Alison Peisker – January 2022
  - Starting with Ceph evaluation
- Primary Enstore developer is retiring (with relatively short notice)
  - Replacing by internal transfer (phasing in over time)
- Have an open posting for another developer
- Experienced member of storage/tape operations retired
  - Open posting for replacement

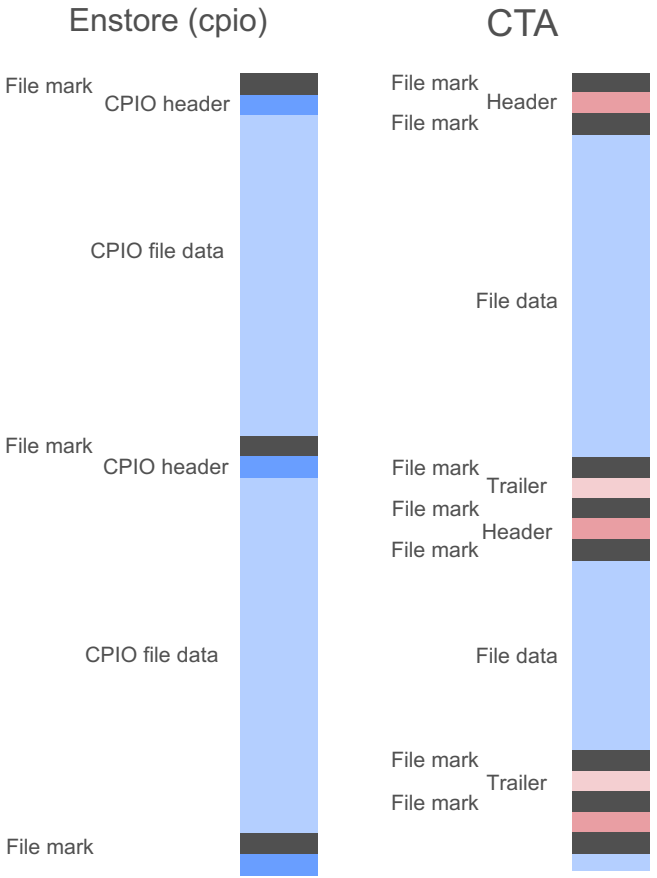
# Moving from Enstore to CTA

- The retirement of the primary Enstore developer makes transitioning to CTA an even higher priority
  - This both emphasizes why we need to move to something new (and with a larger pool of developers), and has also caused a short-term slowing of the CTA effort as we transfer all the knowledge we can
- The most significant challenge to switching tape management systems is the 270+ PB of data we already have
  - Not practical to copy everything just to switch to CTA, so need a way to convert in place
- Two aspects to this – the metadata and the data format
- The metadata is the database holding all the information on the tape contents, file sizes, checksums, ownership (storage group), etc
  - A slight complication is that the Enstore uses the pnfs namespace from dCache, while CTA utilizes the EOS namespace, but converting from one to the other shouldn't be an issue
- The data format is the structure of the data on the tapes themselves

# Tape data format

- A written tape consists of multiple data regions (“files”) separated by tape marks (or “file marks”)
  - File marks are physically distinct on the media; not just software defined

- Enstore uses two different data formats
  - The majority of the data is in “cpio” format
    - This makes each file self describing by wrapping each file in a cpio archive and storing it as a single file on tape
  - Files over 8 GB use the same format as CASTOR & CTA
    - This is a header + file mark + file data + file mark + trailer






Fermilab

# CTA: Work so far on reading cpio data

- Set up CTA test environment with mhVTL virtual tape library (not connected to a real tape library yet)
- Modifying CTA code to decode cpio header and return only the wrapped data to the higher levels
  - Also need to correctly handle the size & checksum (the raw data read from tape includes the cpio header, so doesn't match the unwrapped file size and checksum)
  - Have successfully read back very small cpio wrapped files from virtual tapes
    - Larger files require stitching multiple buffers together
- Still to come: modify seek code not to expect headers & trailers
  - Advancing by one cpio wrapped file requires advancing one file mark. Advancing one CTA file requires skipping three file marks.

# CTA: Other practical problems to solve

- dCache or EOS as the frontend
  - CTA uses EOS to provide namespace and read/write buffer
  - Currently access to Enstore is via dCache
    - CMS has disk only dCache + separate tape r/w instance
    - Public dCache has large tape backed cache with direct access by users
  - For CMS, could replace “tape dCache” with EOS + CTA with little change to access method
    - Such a change to public dCache would be much more disruptive to end users
  - The dCache project is testing an interface to make dCache function as a direct CTA frontend
    - dCache pools are used as the read/write buffer, just like now
    - This approach would have less impact on public dCache end users
    - dCache is a product we have more influence over than EOS
  - Have to keep in mind the effort required to manage a more heterogenous environment
    - We have an EOS instance for the LPC already, so it’s not a completely new system

# CTA: Other practical problems to solve (2)

- Small file aggregation (SFA)
  - This was an attempt to improve the efficiency of storing “small” (10’s of MB) sized files on tape by transparently combining them into GB sized tar files and putting that on the tape
    - Personal view: it did improve the efficiency, but with hindsight – although it seemed like a good idea at the time – it has encouraged bad habits among the experiments. Like most transparent things it’s only transparent up to a point.
  - We have large numbers of these aggregated small files on tape, so we will need some way to keep reading them
  - I think restoring them could be done within dCache (so no changes to CTA), although some method would be needed to map the dCache pnfs namespace entries to the containing tar file
  - Question is what to do about future small files given the experiments now expect to be able to do this
    - Some of the reasons that originally motivated this feature may have been addressed (“buffered file marks”), so maybe we don’t really need it?

# CTA: Deployment considerations

- We need to run CTA in the FNAL environment, which requires some changes from CERN (and RAL)
  - Most significant is that existing installations at CERN and RAL use Oracle as the backend database
  - We no longer use Oracle for scientific databases
  - CTA can use PostgreSQL, but this is less well tested and tuned for large scale

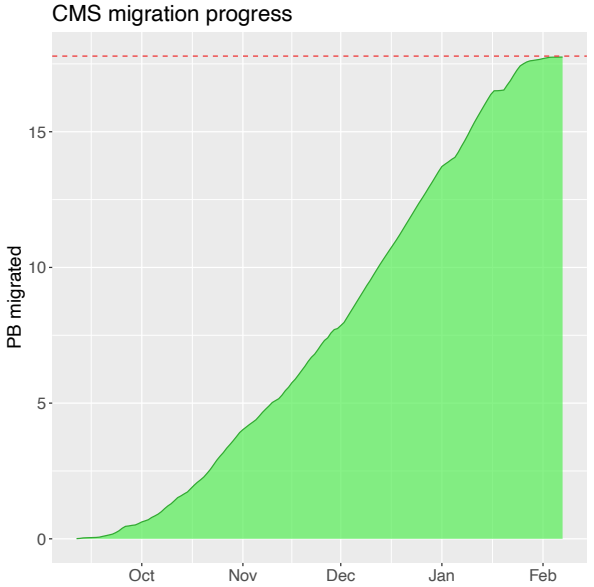
- Want to leverage same platforms already used locally where possible
  - Containerized deployment, OKD (OpenShift)
    - CERN production deployment is not containerized
  - CTA uses Ceph for a work queue, so will need that too

# Tape migration

- Migration to new tape media was identified as a problem in the last ICAC review
  - Not scaling well; too labor intensive
- Have done work to address this over the last year
- The existing migration process was derived from the tape cloning process
  - It went to great lengths to preserve all the Enstore metadata exactly, and had paranoid checks for this – but users don't care about internal Enstore IDs
  - Ran on dedicated "migration stations" that required specific hardware and software installations

# New tape migration

- A new process has been implemented that leverages dCache to perform the migration
  - Uses dCache pools; can move them between normal usage or migration as required
    - Maximizing throughput required tuning I/O load much more than for normal read write activity (or just use ultra-fast NVMe drives)
    - Control process doesn't do data transfer itself (all third-party copies); can run on any system
  - More automated workflow; less load on operators
  - Simpler (few 100s of lines of Python)
  - Benefits from all dCache error handling/recovery (same as used for initial upload)




- Tested on migrating Tevatron data (~20 PB) – simpler problem
- Used to migrate all the remaining CMS data from the Oracle tape library to the new Spectra Logic library (17 PB, most of it in Oct-Jan)
- Now moving on to public migration, which is the most complex
  - Still a few things to add; for example, the best way to handle SFA files

# Improving tape staging efficiency

- A current problem with the efficiency is poor ordering of stage requests between dCache and Enstore
  - Each dCache pool maintains a separate restore queue
  - dCache has no knowledge of tape locations
  - Enstore can handle a limited total queue size, so each pool queue must have a relatively low maximum number of active restore requests
  - Consequence is that when the system is busy, restore requests are passed to Enstore scattered across many tapes
  - More noticeable for public than CMS, because it frequently gets heavily loaded with uncorrelated activity from experiments and users

- Leads to too much tape spooling back and forth – takes time and wears out the tapes – and too many mounts and remounts – takes time and wears out the tapes

# Improving tape staging efficiency (2)

- Tuning the new migration process demonstrated that grouping requests by tape on the same dCache pool can improve the throughput significantly
  - Done externally in that case – by controlling the input tapes used – but proof of concept
- Will be able to build it into dCache internal staging/queuing
  - Retrieve tape info from Enstore (Should be able to get the same info from CTA)
- Migration got a factor 4 improvement in throughput by grouping requests this way
  - This was in a tape library that wasn't getting much other activity at the time
- See also "Improving Performance of Tape Restore Request Scheduling in the Storage System dCache" from CHEP 2021

# Contributions to common storage projects

- As a part of our contribution to the dCache project, SCD developers are involved in a number of other storage and data transfer projects for the HEP community

- Transition from gridftp to WebDAV (including SRM with https)
  - Participating in the process of transitioning away from gridftp to https based file transfers
  - dCache has supported SRM+https for a very long time, but nobody was using it
    - Now we need to move away from gridftp, and without a tape staging API we have to continue with SRM, so the combination with https is finally seeing real use and testing

# Contributions to common storage projects (2)

- Common WLCG Tape REST API
  - A common API for dCache/CTA/STORM/etc tape access (replaces SRM)
  - Playing a major role in the design – looks like an extension of the existing dCache API
  - This interface will help ease the transition from Enstore to CTA; clients can use the same interface regardless of the various combinations of dCache, EOS, Enstore, and CTA

- Token support for dCache xrootd and WLCG interoperability tests
  - Implemented token support for xrootd within dCache
  - Participated in the interoperability tests to ensure all the different sites/implementations can work together
    - 3rd party xrootd is fully implemented in dCache, but we don't expect it to be a primary use case

# Data management/Rucio

- CMS is now fully using Rucio (since Nov 2020)
- DUNE moving forward with transition from legacy SAM data management
  - Rucio used exclusively for data placement outside FNAL & CERN
  - Plan is that upcoming protoDUNE run (October 2022) will be entirely Rucio managed rather than a hybrid like the last run
- Some use by other IF experiments (i.e. ICARUS) for data movement
- Rubin Observatory is adopting Rucio for their “data backbone”
  - Has provided additional funding for Rucio DM work

- Arranging work so everybody is involved on at least two of the three projects (CMS, IF/DUNE, Rubin)
  - Leverage using a common tool to develop a more general pool of expertise
  - Feeding contributions back to the core project
    - Site consistency (from CMS)
    - Containerized deployment (CMS & DUNE)

# Summary

- CTA
  - Ongoing work on being able to import cpio wrapped data into CTA
    - DESY want us to solve this problem for them too
  - Considering options for the optimal approach to integrate it with existing systems

- Migration
  - New migration system implemented: more automation, less human effort

- Tape staging efficiency
  - Tuning the new migration provided a proof of principle of methods to group requests better to improve file restore efficiency

- Also contributing to various other community storage, data management & transfer projects