CERN / CASTOR Tape Archive (CTA)

Motivations and context

2015 Program of work

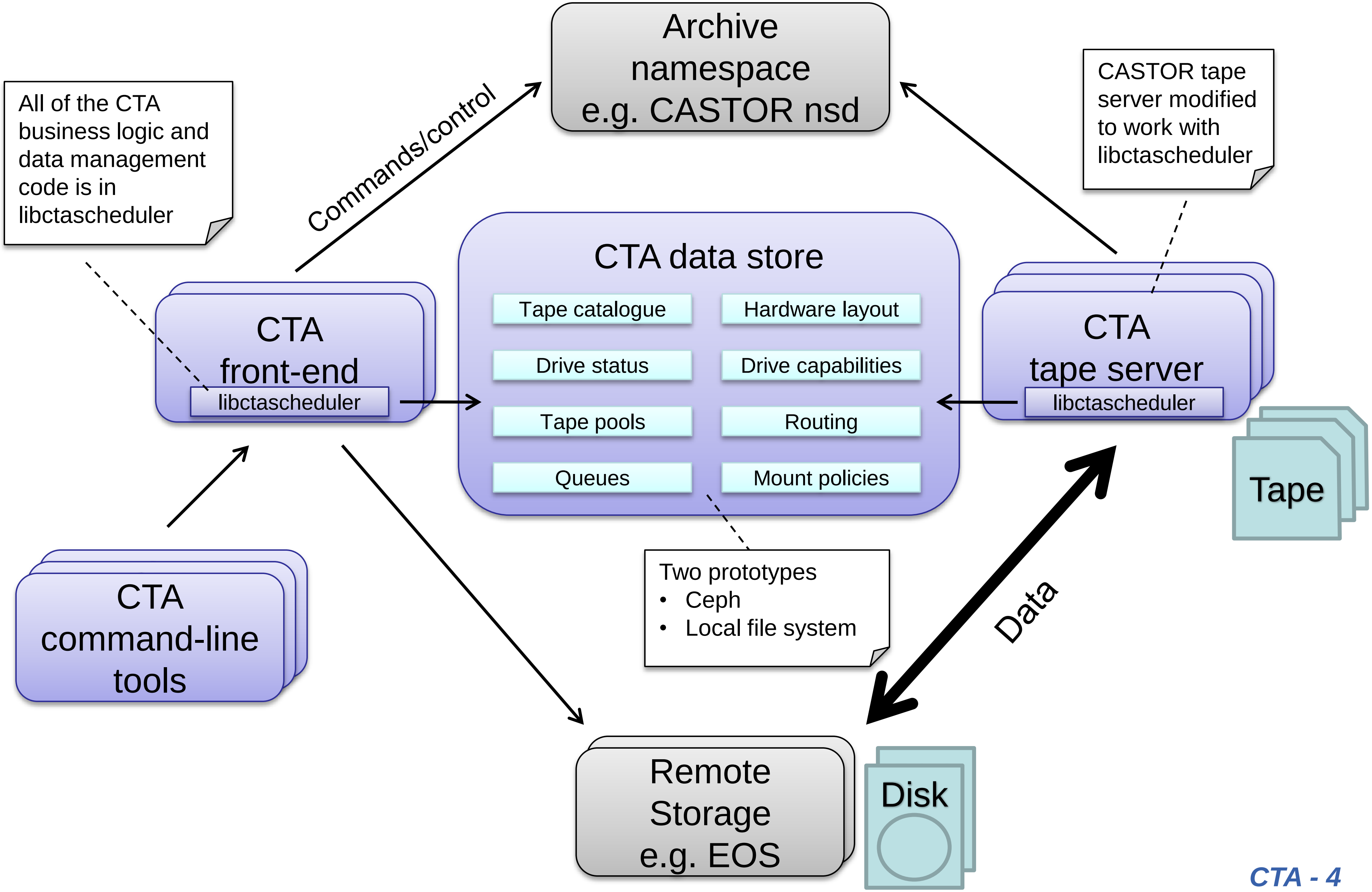
What is CTA?

- A third party copy engine for tape that provides:
  - A simple interface to tape
  - A tape resource scheduler
- CTA has no disk storage and will not stage files
- Remote storage will provide adequate bandwidth for tape
- CTA does not depend on Oracle
- There will be a separate architecture presentation

Motivations

- Simplicity
- Profit from the evolution of tape access
  - Away from random access
  - Towards bulk archival and retrieval
- Provide EOS with direct access to tape
- All of tape in one place
  - Hardware catalogue, queues, policies, scheduling, ...
  - Global and consistent view
  - No scheduling decisions based on partial information
  - Simple and powerful platform for future improvements
  - Self-contained unit within the DSS toolbox

CTA architecture



Archive namespace e.g. CASTOR nsd

All of the CTA business logic and data management code is in libctascheduler

CASTOR tape server modified to work with libctascheduler

Commands/control

CTA data store

- Tape catalogue
- Hardware layout
- Drive status
- Drive capabilities
- Tape pools
- Routing
- Queues
- Mount policies

CTA front-end
libctascheduler

CTA tape server
libctascheduler

Tape

CTA command-line tools

Two prototypes
- Ceph
- Local file system

Data

Remote Storage e.g. EOS

Disk

Problem dimensions

- Peak performance of CASTOR tape sustained over 2 months
  - 83 million files, 25 PetaBytes per month
  - Average file size = 300 MegaBytes
  - Average user request rate = 34 Hz
  - Peak user request rate = 70 Hz

- Current data rate of CASTOR is ≈ 5 times less than peak
  - 6 PetaBytes per month

- Target user request rate
  - 350 Hz average
  - 700 Hz peak

CASTOR and CTA

- CTA will run alongside the current CASTOR
- Share tape file metadata (CASTOR nsd)
- No need to migrate the actual tape files
- Several options for co-existence

Questions for DSS

- Namespaces?

- Data movers and garbage collectors?

Namespaces?

- A technical meeting is being organized
- Which technology?
  - EOS mgm
  - CASTOR nsd
  - Something else?
- How many?
  - One super namespace for multiple tiers of storage
  - Many EOS and one archive
  - Many EOS and many archive
- What is the recovery strategy?

Data movers and garbage collectors?

- What do we have?
  - HSM - CASTOR
    - Automatic recall – Is anybody asking for this?
    - Automatic archive
    - Automatic garbage collection
  - Assisted - EOS archiver
    - Driven by power users
    - An archive is a namespace tree within EOS
    - Users can freeze, archive, purge and retrieve archives
  - You’re on your own - ATLAS and CMS
    - Third party copies between EOS and CASTOR
- What are we missing?
  - Automatic archive from EOS to CASTOR / CTA

Reserve slides

Reserve slides beyond this point

CASTOR tape server

- It is a TCP/IP server with respect to receiving mount commands from the CASTOR drive scheduler (VDQM)
- It is a TCP/IP client with respect to the CASTOR stagers from where it gets the lists of files to be transferred
- It is a TCP/IP client with respect to the remote file systems with which it transfers the contents of files to and from memory.
- Runs on a standard PC connected to a tape drive and the tape library in which the drive is installed
- Controls the mounting, loading, positioning, unloading and un-mounting of the tape even in the event of hardware and software failures
- Transfers data from remote file systems to memory to tape and vice versa. Memory is used to smooth bursty disks transfers and to parallelise slow disk transfers
- Memory is used to map the multi-stream approach of disk with the sequential one file after another format of tape

Tape scheduling

- Storing the physical layout and capabilities of the tape hardware
  - Which tapes and drives are located in which libraries
  - Which tapes are compatible with which drives
- Keeping track of the current status of the tape drives
  - FREE or BUSY
- Queuing user requests
- Mapping tapes to user tape pools
- Routing user files to user tape pools
- Limiting the number of tape mounts
  - Reduce inefficiency due to costly mounts/un-mounts
  - Reduce ware and tare on the tape hardware

Possible CASTOR improvements

- Merge the tape drive scheduler and tape catalogue daemons
  - One daemon instead of two
  - A single tape resource database
  - Move all drive scheduler and tape catalogue logic to PL/SQL procedures
  - Queues and prioritise tape write requests based on tape pools as opposed to tapes
  - Atomically schedule of tapes and drives when writing to tape
  - Heartbeat mechanism to identify unreachable tape servers and automatically stop scheduling them
  - Pre-emptive scheduling to enable true background repack and tape verification activities
  - Support libraries that contain drives with asymmetric read/write capabilities
  - Replace CUPV for tape authorisation requirements
- The stager/tapegateway should no longer control the fseq of individual write requests
  - The tape server should simply report where files have been written (VID + fseq + block ID)