Data Management
Summary of Experiment Prospective

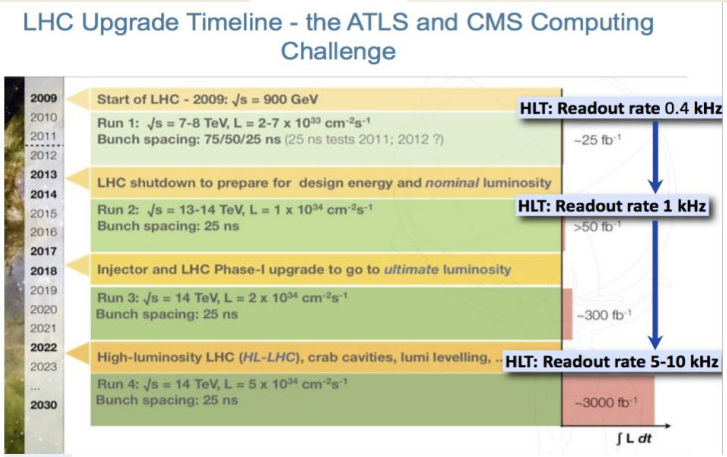
D. Benjamin, B. Bockelman, P. Charpentier, V. Garonne,
M. Lassnig, M. Litmaath, W. Yang,
F. Wuerthwein

U.S. DEPARTMENT OF ENERGY
Office of Science

SLAC NATIONAL ACCELERATOR LABORATORY

The Situation Room

- Run 2/3 (→ 2023):
  - » Flat budget
  - » Computing model changes impact resource utilization
    - ALICE: analysis train, nanoAOD
    - ATLAS xAOD, train model, RAWtoALL
    - CMS MiniAOD
    - But expect no miracle
  - » In DM:
    - Need step-by-step, large and small improvements everywhere
- Run 4 ?
  - » $$$ is the best solution

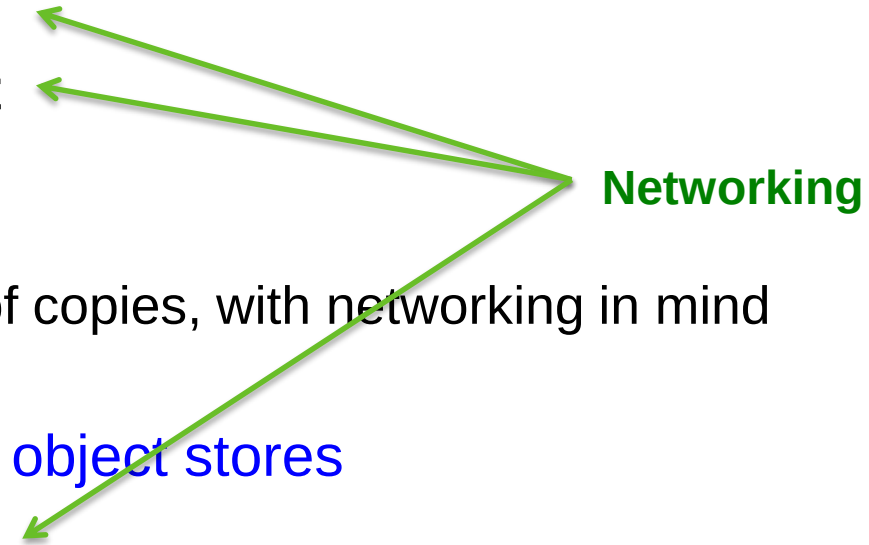


LHCb medium term

- Run2 conditions:
  - 12.5 ++ kHz of HLT output rate
  - Small pileup (μ=1.1)
  - Throughput just below 1 GB/s
    - In 2015: up to 1.3 GB/s
  - Online reconstruction = offline reconstruction
    - Allows direct analysis from online data (TURBO stream)
- LHCb major upgrade is for Run3 (2020 horizon)!
  - Luminosity x 5 (2 10^{33})
  - Trigger rate… x 5 (at least)
  - Throughput between 6 and 10 GB/s!
  - Trigger (SW only) = offline selection
    - Stripping is no longer effective (all events are for physics!)
    - Possibly directly export reconstructed data only

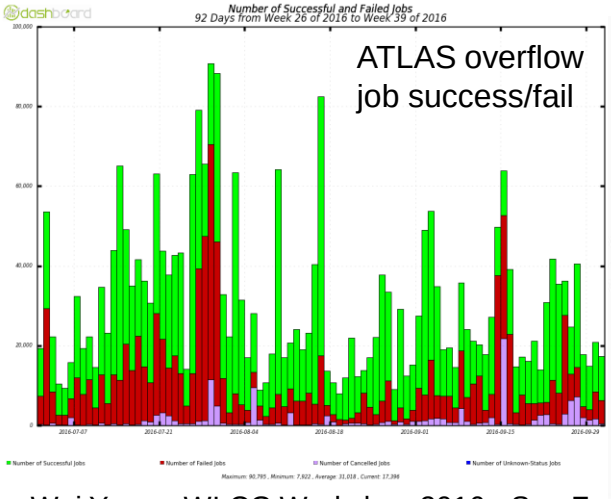
Interesting Areas in Data Management

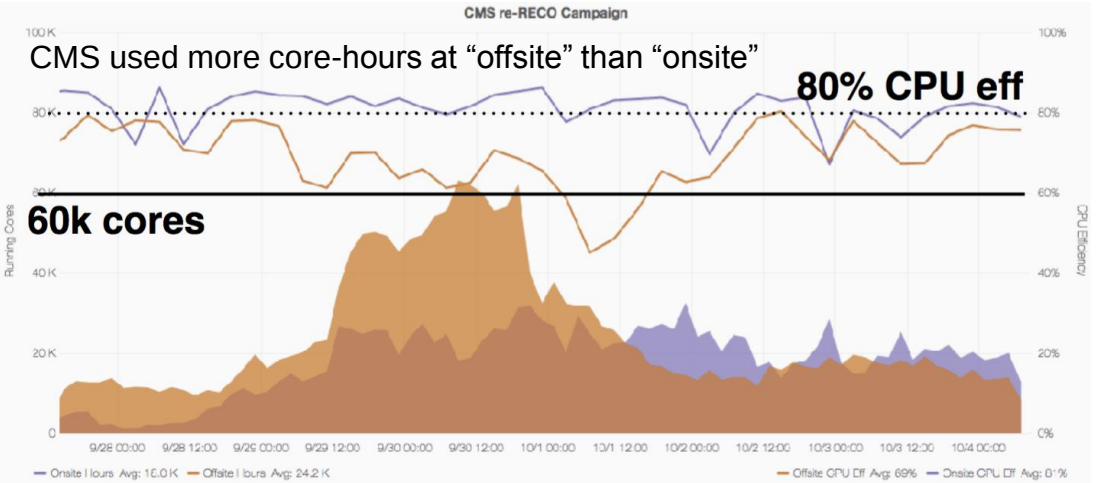
- **Remote access**
- **Improved data placement**
  - » Fast data movement
  - » Smart placement
  - » Assess requirement on # of copies, with networking in mind
- Protocols: old / good / new
- Storage: traditional stores / object stores
- Federations
- Caching technologies
- Tape, Cloud, HPC, etc.



Remote access – everyone's favorite – with caveats

- ALICE: remote access can be tolerated at a level of 15%
- ATLAS: (analysis) failover jobs and overflow jobs via FAX, ~10% level. Will also try metalink (anything on the software side?)
- CMS: Aggressively invested in sw&c for remote access.
  - » In Run2, more data-intensive workflows are "WAN ready", efficiency dropped but manageable.
- Mostly ROOT/xroot protocol so far
- WAN/latency awareness in software is the key, hard to do this for user analysis





Improve Data Placement / Distribution / Remote access

SLAC

- Network bandwidth and reliability vastly improved in many places
- We are building analytic platform, ML capabilities:
  - Data access logs, data transfer logs, network weather, etc.
  - ALICE: sophisticated storage monitoring down to individual disk servers
  - We monitor data popularity
- Can we do:
  - Smart data placement
  - Fast movement - will this save space / data copies?
  - Remote IO
- Dilemma – How much should we increase network usage?
  - ATLAS is getting complaints from two directions:
    - Why not fully utilized my site's fat network pipe
    - How to conserve my site's precious network resource
  - Be "Smart" ?

What do we do?

Protocols

- Data access protocols
  - » Posix file and ROOT/xroot are popular
    - xroot is well tested. HTTP is playing catch up, both for local and remote access
  - » Download to WN or direct IO – not a protocol issue but…
    - ATLAS & CMS: direct IO puts higher stress on storage - delegate the decision to site
    - Caveat: download to WN to merge?
- Data transfer protocols
  - » SRM+GridFTP / xroot / http
- Old / Good / New:
  - » SRM / GridFTP, xroot, http / S3

Protocols – making SRM optional

- The SRM issue / solution is well understood
- All SRM functions we are using can be replaced by other protocols such as GridFTP
  - » Except tape staging (specially if tape and disk are mixed), bulk deletion
  - » Each experiment is taking their own approach on this
    - ALICE: xroot only, **never used SRM**
    - ATLAS: actively working on SRM+GridFTP → GridFTP only.
      - Adler32, space reporting, file-by-file deletion, load balancing
        - still technical/coordination work to do, no road blocker
      - Actively testing xroot and http for file deletion and 3rd party transfer.
    - CMS: proved to be able to go GridFTP only – sites' choice
    - LHCb: in the work, avoid SRM as much as possible

Protocols – where are they heading

- SRM → becoming less used
- GridFTP: likely stay for long term for data transfer
- xroot, http:
  - primary focus could be data access
    - Improve efficiency
      - not completely in the scope of DM, some in the software domain
    - Improve reliability (see Federation slides)
- Make room for new protocol to entry the game
  - They aren't just a new protocol for GET/PUT, etc.
  - They represent a new way of using storage (for good or bad)
  - They are an large eco-system
  - ATLAS: tried it in AWS cloud; ALICE: to be able to use S3, require major development.

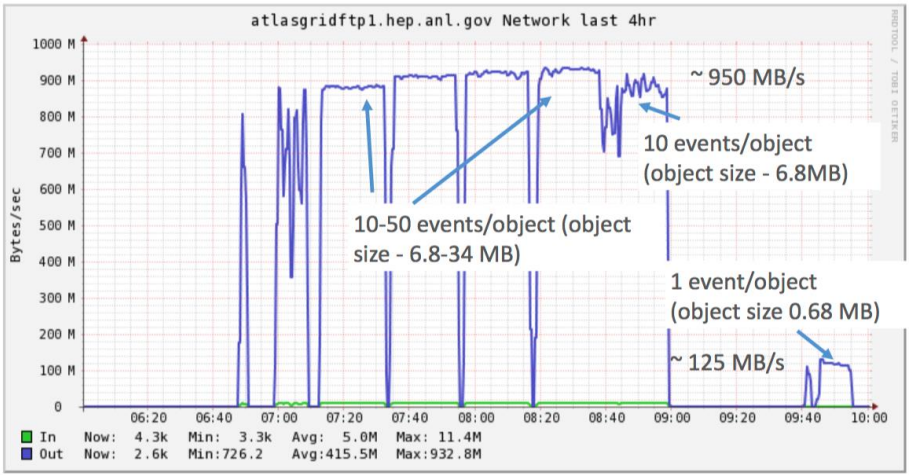
Storage: traditional vs object stores

- Experiment shouldn't care about the brand name
  - Should care: performance, cost, how to utilize
  - CMS: Unlikely that the "storage element" model will go away anytime soon for large sites – probably speak for ATLAS too.
  - Should have a good understanding on the strength and limitation of the object stores
    - Performance gain come from simplified name space operation, client has to share responsibility on name space
    - Good for some applications. Unlikely to be a universal silver bullet
    - Gain in redundancy (and cost, not performance) with erasure coding
    - Authentication model
  - CERN, BNL, RAL have setup CEPH storage for ATLAS
  - ALL four tried AWS S3 storage in production ?
- Our DM systems are built upon files ~ O(1000) events.
  - Conventional storages work well.
  - Completely switch to pure object based will stress our DM systems
    - Are there ways to avoid that?
  - ATLAS is using object store in smaller scale:
    - Event service - great way to explore opportunistic resources
    - Log files
- Deletion is costly (at least for ATLAS). Object stores won't do better.

Storage: CEPH / object stores

OS Performance. Bandwidth vs Object Size

- Achieved ~7.2GiB/sec writing speed from ANL to BNL



Issues and fixes

- Large number of small transfers can saturate Object Stores
  - » Initially Yoda was sending outputs one at a time directly from the compute nodes
  - » Fixed this (on HPC) by asynchronous sending of pre-merged outputs (tar-balls)
  - » What to do with output of large number of Event Service jobs on the Grid?
- Prefer few large transfers to the Object Store to many small transfers
  - » Network stack and IO system limits

We need to gain more experience, learn how to best using object stores

Federations

- Multiple dimensions
  - Storage level, regional federations
    - NDGF-T1, MWT2, AGLT2 dCache,
    - Regional (transparent) federation is seen as a way to consolidate storage and reduce operational cost
      - Reliable and efficient due to people working together, have less diversity, and short RTT
  - Overlay on existing DM:
    - AAA, FAX, DynFed
    - Site/Infrastructure reliability has great impact on the usability of the federations
  - DM level, catalog driven “federations”.
    - ALICE: Jobs get a sorted list of SEs per file – based on network topology and SE availability
    - ATLAS: RUCIO metalink – a list of multi-protocol pfns for client to choose
    - LHCb: Gaudi data federation for analysis, works very well
- Fault tolerate?
  - FAX (and others that I know of) were built to tolerate faults. But experiments used it in a rigid way.
  - Can Catalog Driven Federations handle fault in-tolerate usage?
- Global Name Space
  - Production job doesn’t care
  - Users like this. They don’t always want to download to their laptop!
  - Also important for caching to work well

Caching

- Where DM doesn't fully track the contents, have self clearing policy
- Caching as a way of using storage:
  - » ALICE: Tier 2s if they are self-healing
  - » LHCb: laptop space can be your cache
  - » Nothing special in terms of technology – but in usage pattern and policy
- Caching as a technology
  - » CMS / ATLAS Xrootd cache, ARC, DMP-lite, CVMFS
- In between:
  - » ATLAS RUCIO cache – managed transfer "in"
    - ATLAS xrootd sites that also run auto cleaning (probably other type of storage system too).

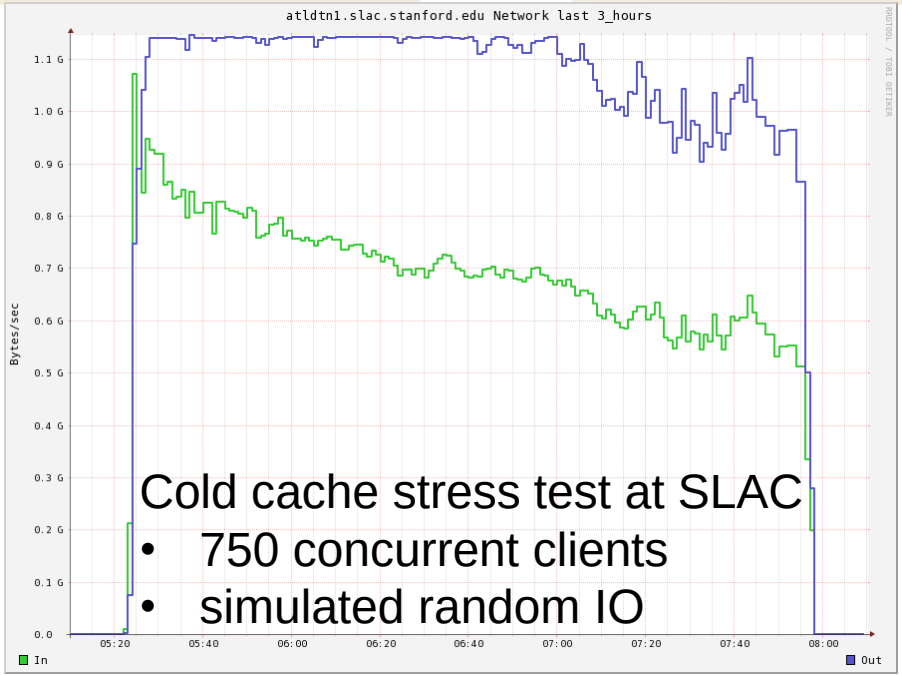
Xrootd proxy cache

SLAC

- A CMS project that ATLAS heavily participated
  - **Squid-like**, static file, high performance, file block level
  - Use Xrootd as core software stack
    - Speak xroot protocol
      - Can easily extend to http between client and cache.
      - Focus on xroot protocol between cache and data source
  - In final development phase, interested in broad collaboration
- Plan:
  - Operational cost evaluation during a 3-6 months period before 2017 data taking.
  - CMS vision: host part of CMS namespace (miniAOD?); default access by all jobs at CMS SoCal (Caltech & UCSD)
  - ATLAS vision: Cache; unmanaged Tier 3 storage; cloudy Tier 2

Xrootd proxy cache - evaluation

- 300+ TB in SoCal (Caltech & UCSD)
  - » 30-50 Gbps IO performance for simultaneous read/write with up to 20,000 clients reading.
    - 100+ disks, diverse hardware & filesystems
    - estimate to need ~20 disks to fill 10Gbps pipe
- ATLAS single machine cache test:
  - » 12x 2TB, 10Gbps cache at SLAC for stress test, data source @BNL
  - » 80TB, 10Gbps cache at Univ. Chicago for real analysis jobs
- Detail: see Rob Gardner's CHEP 2016 presentation



Cloud, Tape, HPC

SLAC

- Commercial cloud storage:
  - » AWS is still more expensive then running our own tape systems
    - Free power, cooling, space, etc., and perhaps free manpower☺
    - But for how long?
  - » Note: AWS and pay-per-use is not the only cloud model.
- Tape? No news (is good news)
  - » Can we use a mechanism other than SRM to stage files?
- HPC
  - » Temporary storage
  - » May involve manual steps to move data in and out – how to integrate into the automated DM systems?