Federated storage with xrootd

THE UK CMS EXPERIENCE

Outline

- UK status
- Imperial setup
- Storage federation testing
- UK CMS xrootd plans

UK xrootd status

- Installed at Imperial, RAL and RALPP
- Possible performance implications of relatively old hardware at RAL
- Plans for installation at Brunel soon

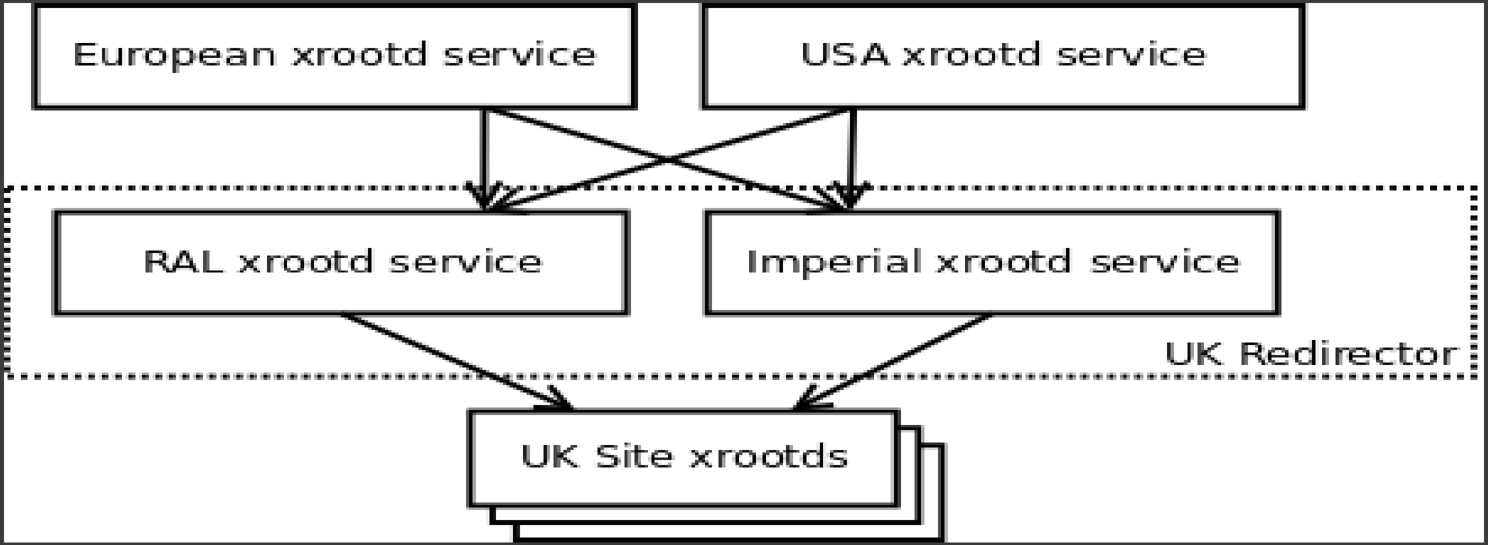
Details of Imperial xrootd I

- Chose to configure xrootd so that incoming requests are passed directly to the dCache xrootd door
- Requests do not pass through a proxy server first
- This is simpler and there’s less to go wrong
- The namespace runs in a chroot

Details of Imperial xrootd II

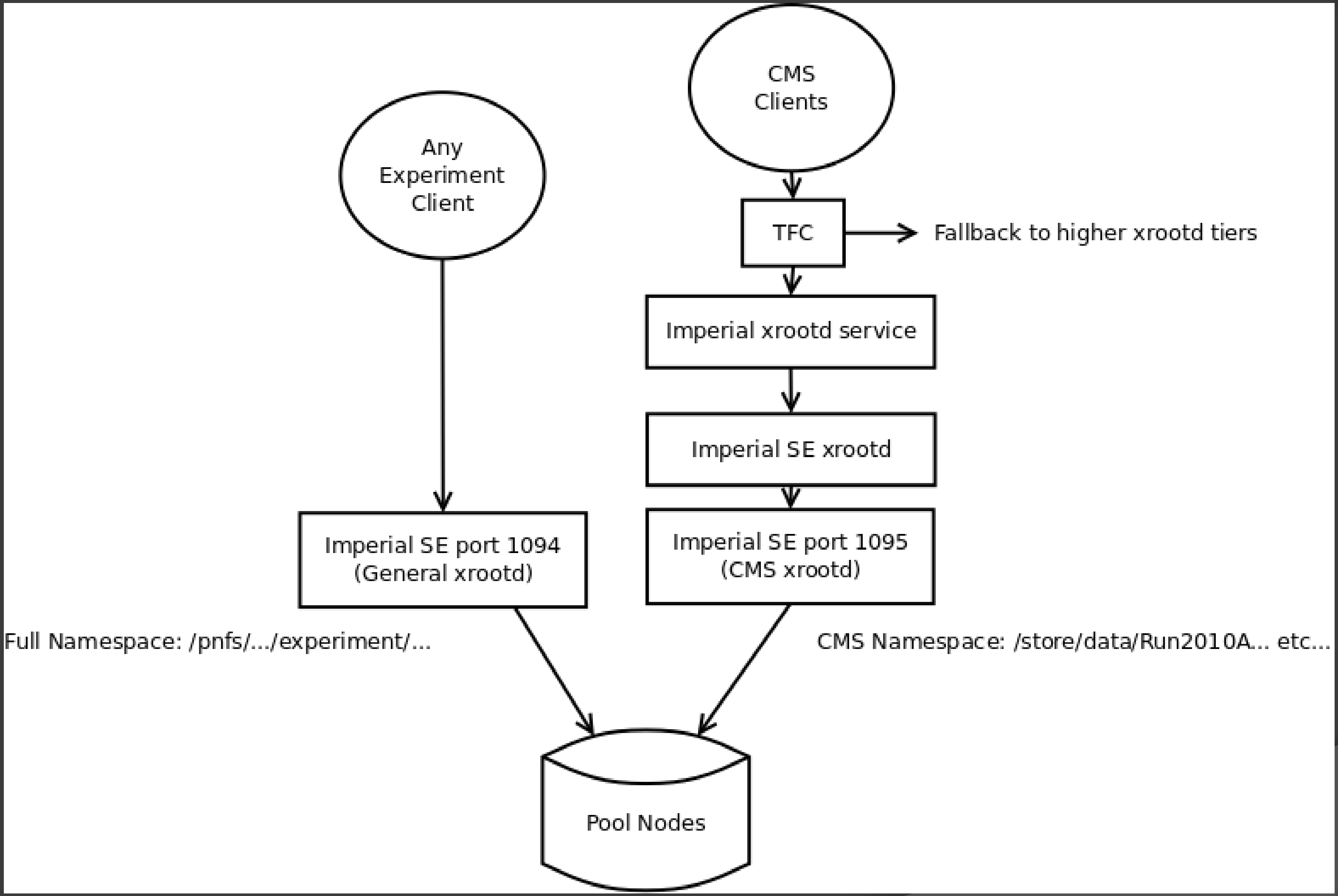
- Took about a day to install
- Our exact configuration wasn't covered in the documentation
- Person who installed it described it as "not too bad"

Details of Imperial xrootd III



- UK redirectors not currently used because a limitation in CMSSW means only 1 redirection is allowed
- Fallback to CERN global redirector, which includes the European and USA ones
- Plan to convert UK redirectors to triage role, using UK sites preferentially, then redirecting to global if the file isn't found there

Details of Imperial xrootd IV


CMS Clients
Any Experiment Client
TFC
Fallback to higher xrootd tiers
Imperial xrootd service
Imperial SE xrootd
Imperial SE port 1094 (General xrootd)
Imperial SE port 1095 (CMS xrootd)
Full Namespace: /pnfs/.../experiment/...
CMS Namespace: /store/data/Run2010A... etc...
Pool Nodes

Storage federation aims

- Explore new tradeoffs:
  - Accept reduced % CPU efficiency
  - Increased data availability
  - Reduced bulk data transfers
- In essence, trade some raw performance for much more flexibility and convenience.
- Does this work in practice?

Test procedure

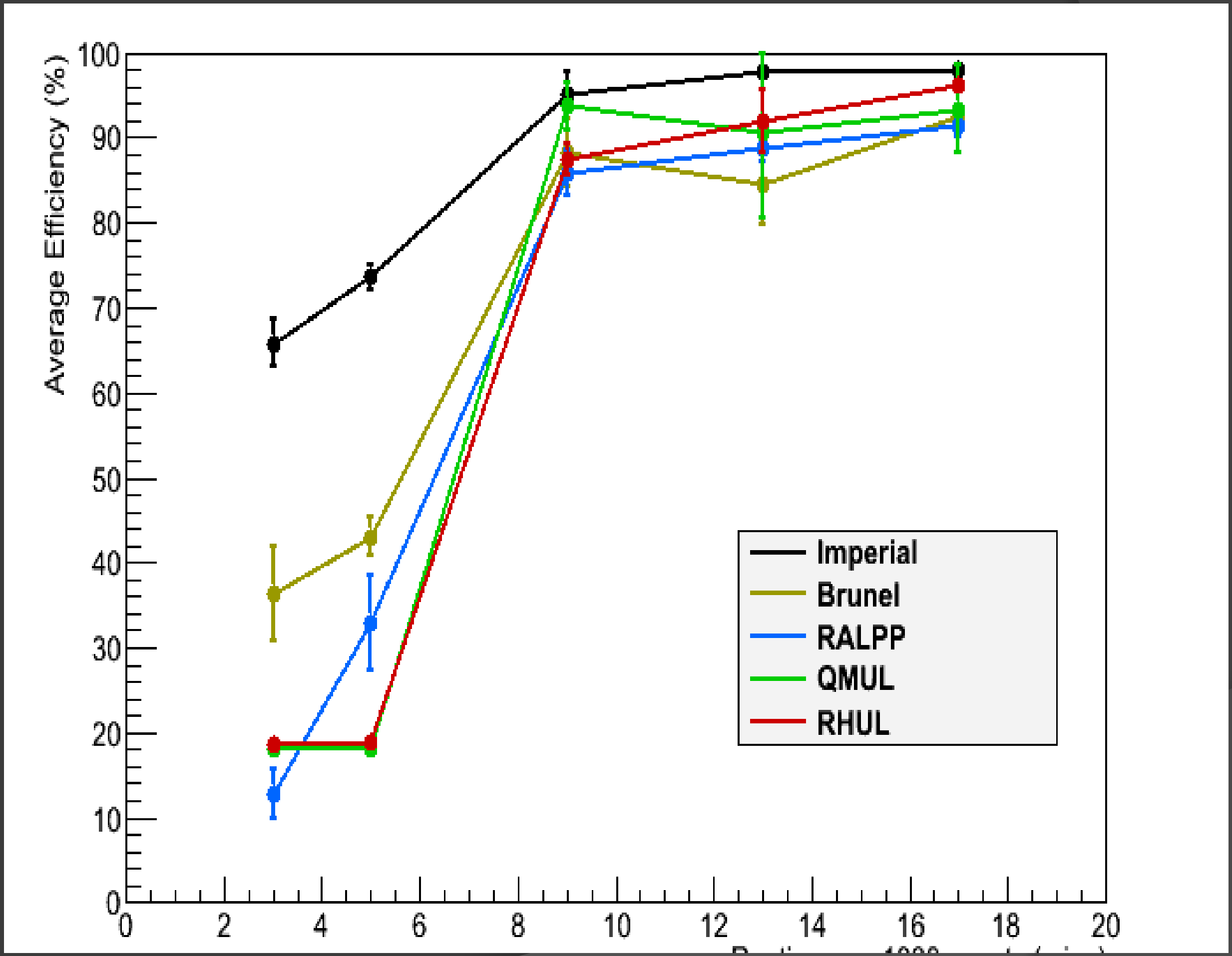
- Idea is to emulate a typical user submitting jobs via crab, running over data files based at a particular site.
- Using the files from the dataset /DoubleMu/Run2012B-PromptReco-v1/AOD based at one site (Imperial), study relative performance of running jobs at alternative sites.
- Jobs of differing complexity and computing intensity (quantified by approximate runtime for 1000 events on local PC) studied. 5 jobs ranging from a highly simplistic job selecting pairs of muons and plotting histograms to a more 'typical' user job creating an ntuple containing multiple objects.
- The performance of these jobs is studied at some UK T2 and T3 sites relative to the performance of running them at Imperial where the files are local.
- The number of events is varied between 1 500 000 for the simplest job and 15 000 for the most complex job, to ensure a runtime of around a few hours for all the jobs.
- Then the more intensive (and arguably more user-typical) ntuple creation jobs are further tested at various European and US T2 sites for comparison.

Testing different types of jobs

- 5 different types of job tested with a runtime of ~3, 5, 9, 13 and 17 minutes for 1000 events.
  - The 3 and 17 minute jobs are the dimuon histograms and ntuple production respectively.
  - The first 2 of the intermediate 3 are generated by increasing the complexity of the dimuon histogram code and the third by decreasing the complexity of the ntuple code.
- For each of the 5 types of jobs, 10 identical jobs running on different files from the same dataset are submitted to each site at once. This is repeated 2 or 3 times at different times depending on the successful operation of the site CEs during the testing period.

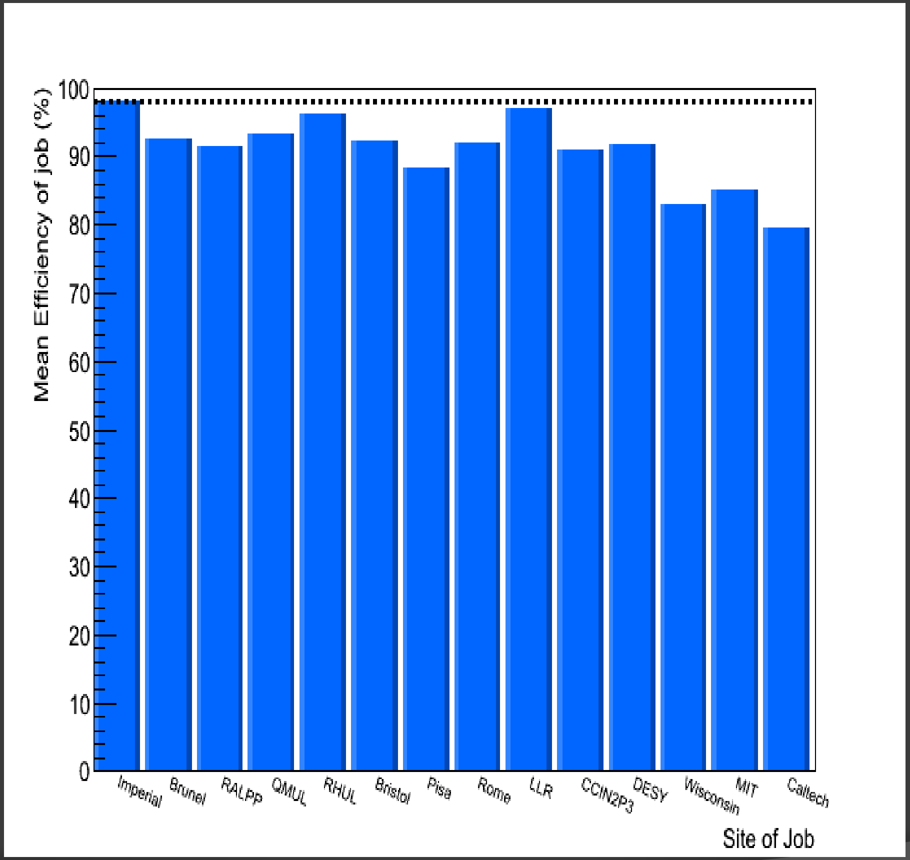
UK Results for all job types

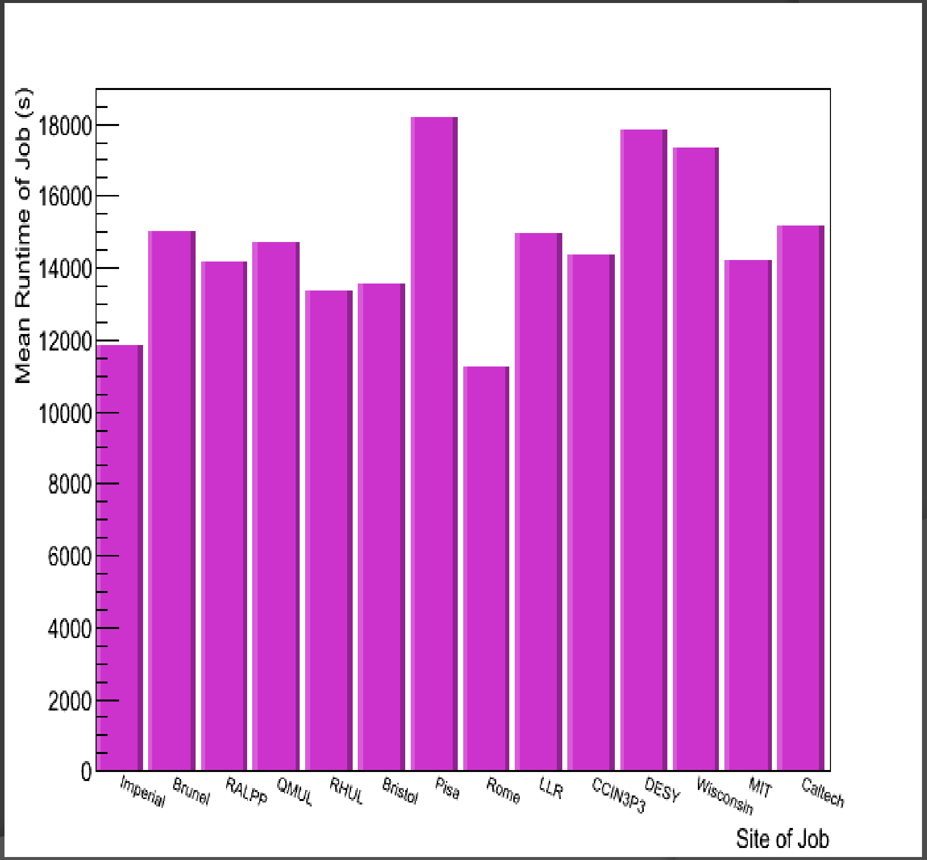
The mean efficiencies of the tested UK sites for the 5 different job types. The error bars indicate the range amongst the 10 separate jobs and their repeats.



Further studies on the more ‘typical’ jobs

- Arguably the more intensive ntuple production code is the type of job more typically submitted to the grid by users for analysis. From the UK sites it can be seen there is a small drop in efficiency for running the jobs at different sites. The plots below show the mean efficiency and mean total runtime for the jobs at all the sites tested, including some European and US sites:





Xrootd testing conclusions I

- Running jobs on remote data possible with some performance loss.
- A greater performance loss is seen for less CPU-intensive jobs (more I/O intensive, so effect of using remote data larger).
- For more CPU-intensive jobs such as ntuple production, relatively small drop in efficiency across different sites in the UK, Europe or US.
- The total runtime for such a job (including I/O timings) varies between 3 and 5 hours across the different sites.

Xrootd testing conclusions II

- Rebecca’s tests have shown that it is feasible to trade some computational efficiency for much more flexibility.
- In most cases the performance impact is not too bad.

UK CMS xrootd plans I

- Mitigate load spikes e.g when Imperial is busy, run jobs at Brunel using Imperial storage via xrootd, and vice versa.
  - Recent networking upgrades via DRI should help here e.g. all Imperial storage now using 10Gbit and there is a “dedicated” 20Gbit uplink for HEP.
- T3 sites tend to lack good storage (compared with T2s).
  - Use them as compute resources, drawing on T2 storage via xrootd.

UK CMS xrootd plans II

- UK is contributing to plans for CMS to use the HLT farm via Cloud APIs.
  - Hardware and networking not designed with this usage pattern in mind.
  - Xrootd federation could be a way of mitigating the limitations of the HLT hardware.
- Aiming for UK-wide xrootd storage federation by end of year, aligning with similar worldwide goal.

Acknowledgements

- Stuart Wakefield for information on xrootd usage.
- Rebecca Lane for the xrootd testing, write-up and graphs.
- Simon Fayer for UK xrootd setup diagrams.

Thank You

- a.huffman@imperial.ac.uk
- http://www.hep.ph.ic.ac.uk