

Scalla/xrootd

Andrew Hanushevsky
SLAC National Accelerator Laboratory
Stanford University
29-October-09
ATLAS Tier 3 Meeting at ANL

<http://xrootd.slac.stanford.edu/>

Outline

System Overview

- What's it made of and how it works

Opportunistic Clustering

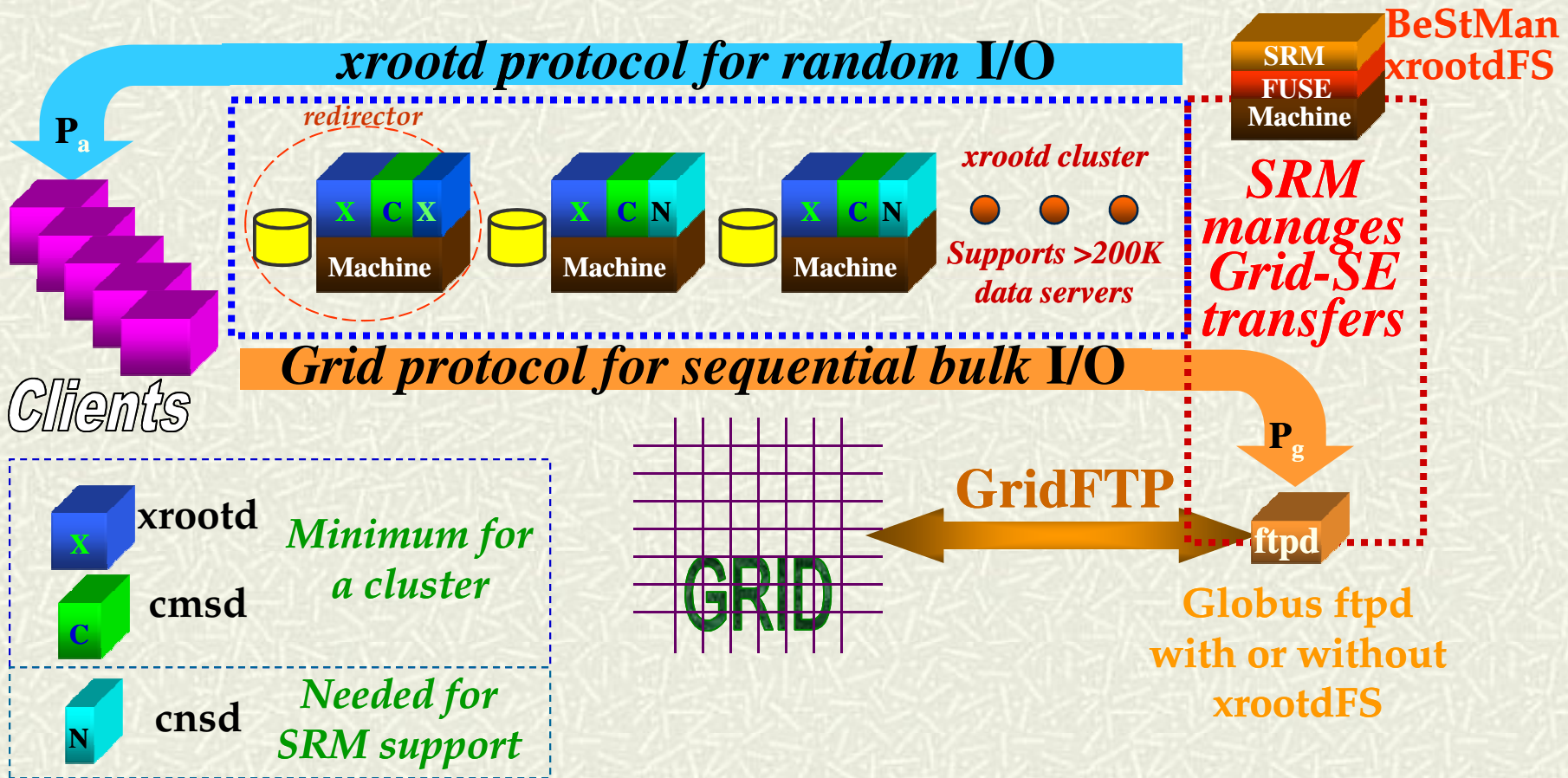
- Batch nodes as data providers

Expansive Clustering

- Federation for speed and fault tolerance
 - The Virtual Mass Storage System

Fullness vs Simplification

Full Scalla/xrootd Overview



The Components

- # xrootd
 - Provides actual data access
- # cmsd
 - Glues multiple xrootd's into a cluster
- # cnsd
 - Glues multiple name spaces into one name space
- # BeStMan
 - Provides SRM v2+ interface and functions
- # FUSE
 - Exports xrootd as a file system for BeStMan
- # GridFTP
 - Grid data access either via FUSE or POSIX Preload Library

*This might not be needed
for typical Tier 3 sites!*

Getting to xrootd hosted data

Via the root framework

- Automatic when files named root://....
- Manually, use TXNetFile() object
 - Note: identical TFile() object will not work with xrootd!

xrscp

- The native copy command

POSIX preload library

- Allows POSIX compliant applications to use xrootd

gridFTP

BeStMan (SRM add-on)

- *srmcp for srm-to-srm copies*

FUSE

- Linux only: xrootd as a mounted file system

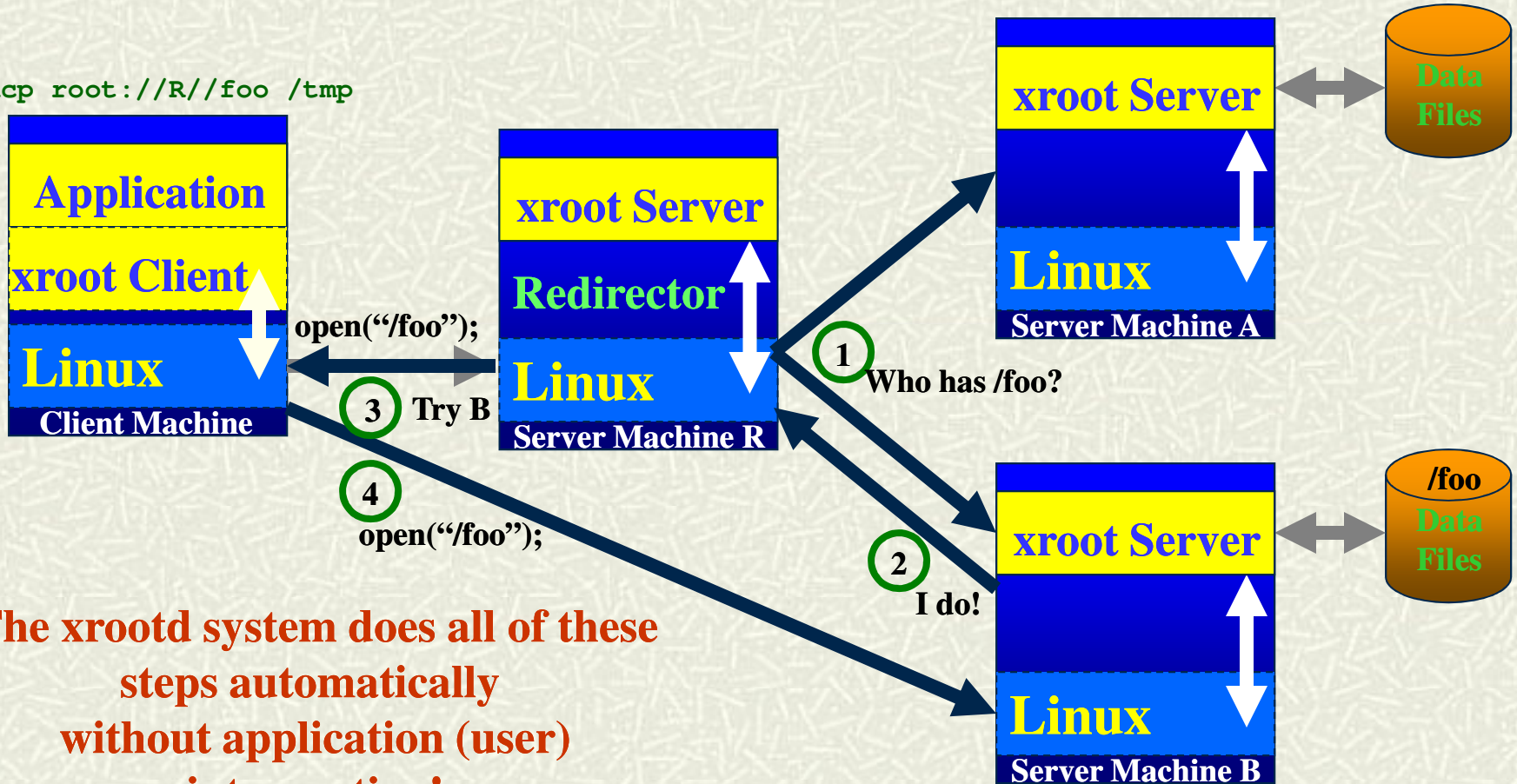
Native Set

Simple Add

*Intensive
Full Grid Set*

Cluster Maneuvering

```
xrdcp root://R//foo /tmp
```



The xrootd system does all of these steps automatically without application (user) intervention!

Corresponding Configuration File

```
# General section that applies to all servers
#
all.export /atlas

if redirector.slac.stanford.edu
all.role manager
else
all.role server
fi
all.manager redirector.slac.stanford.edu 3121

# Cluster management specific configuration
#
cms.allow *.slac.stanford.edu

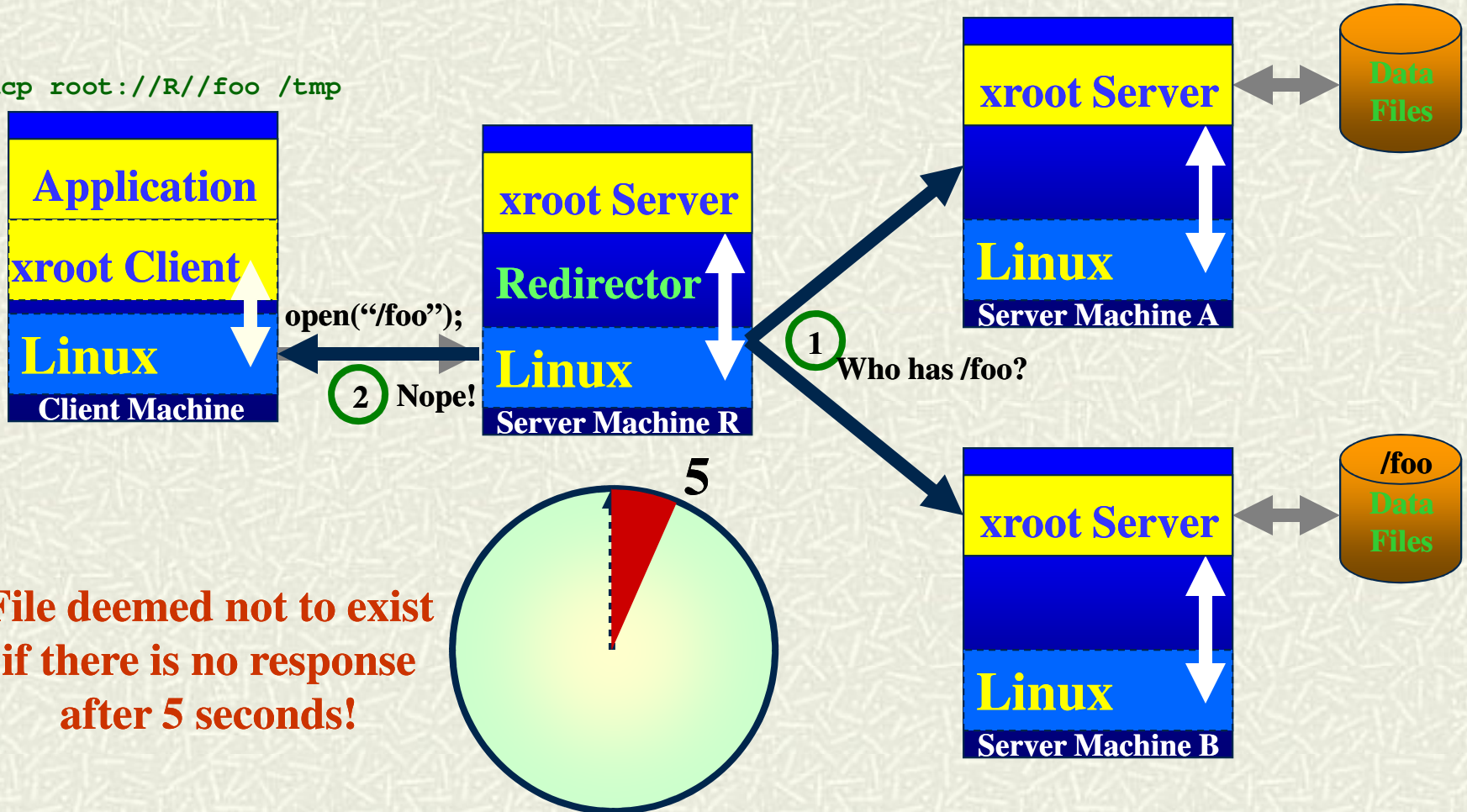
# xrootd specific configuration
#
xrootd.fslib /opt/xrootd/prod/lib/libXrdOfs.so
xrootd.port 1094
```

File Discovery Considerations

- # The redirector does not have a catalog of files
 - It always asks each server, and
 - Caches the answers in memory for a “while”
 - So, it won’t ask again when asked about a past lookup
- # Allows real-time configuration changes
 - Clients never see the disruption
- # Does have some side-effects
 - The lookup takes less than a millisecond when files exist
 - Much longer when a requested file does not exist!

Handling Missing Files

```
xrdcp root://R//foo /tmp
```



File deemed not to exist
if there is no response
after 5 seconds!

Missing File Considerations

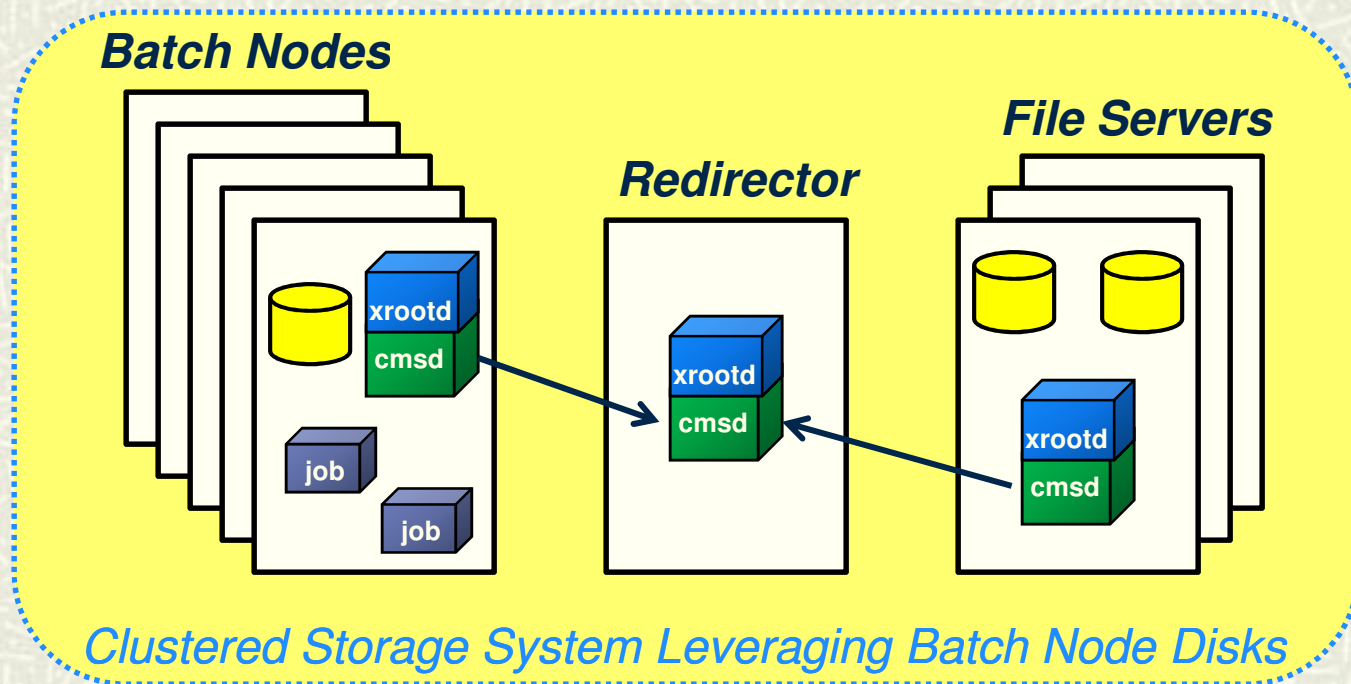
- # System optimized for “file exists” case!
 - This is the classic analysis situation
 - Penalty for going after missing files
- # Aren't *new* files, by definition, missing?
 - Yes, but that involves writing data!
 - The system is optimized for reading data
 - So, creating a new file *will* suffer a 5 second delay
 - Can minimize the delay by using the **xprep** command
 - Primes the redirector's file memory cache ahead of time

Why Do It This Way?

- # Simple, lightweight, and ultra-scalable
 - Ideal for opportunistic clustering
 - E.g., leveraging batch worker disk space
 - Ideal fit with PROOF analysis
- # Has the R³ property (Real-Time Reality Representation)
 - Allows for ad hoc changes
 - Add and remove servers and files without fussing
 - Restart anything in any order at any time
 - Ideal for expansive clustering
 - E.g., cluster federation & globalization
 - Virtual mass storage systems and torrent transfers

Opportunistic Clustering

- # Xrootd *extremely* efficient of machine resources
 - Ultra low CPU usage with a memory footprint $20 \approx 80\text{MB}$
 - Ideal to cluster just about anything



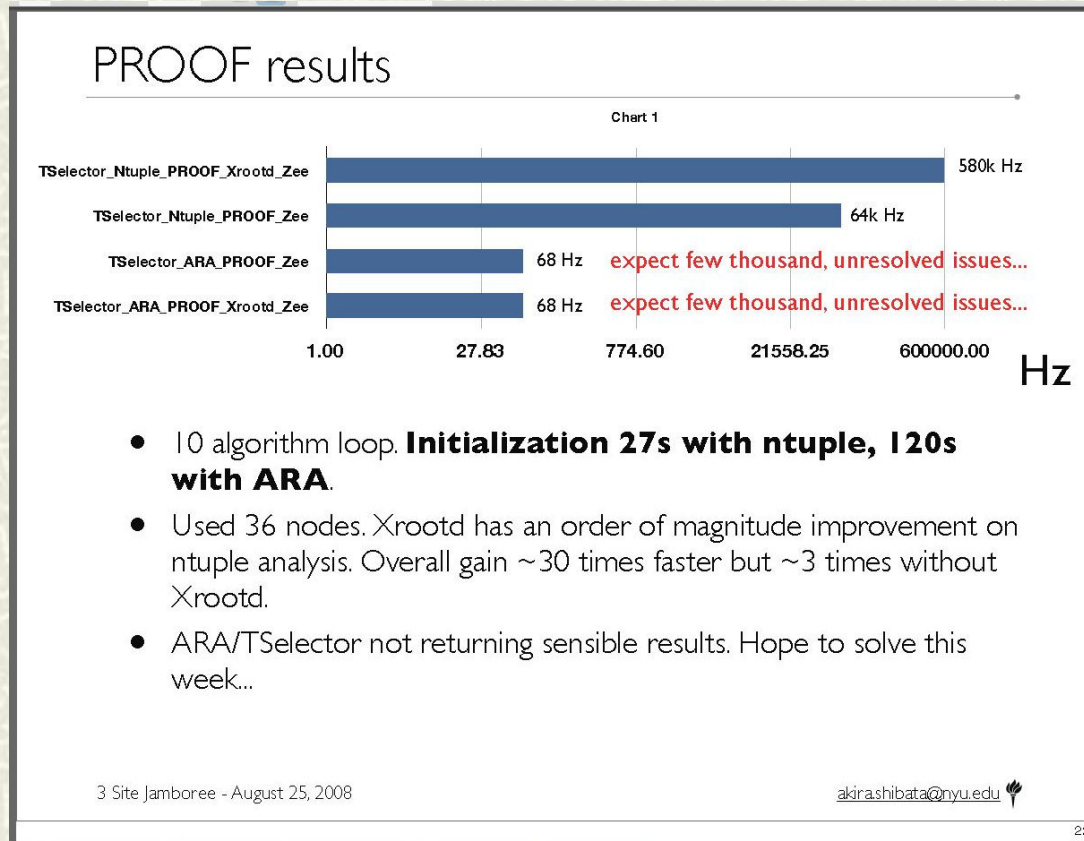
Opportunistic Clustering Caveats

- # Using batch worker node storage is problematic
 - Storage services must compete with actual batch jobs
 - At best, may lead to highly variable response time
 - At worst, may lead to erroneous redirector responses
- # Additional tuning will be required
 - Normally need to renice the cmsd and xrootd
 - As root: `renice -n -10 -p cmsd_pid`
 - As root: `renice -n -5 -p xroot_pid`
- # You must not overload the batch worker node
 - Especially true if exporting local work space

Opportunistic Clustering & PROOF

- # Parallel Root Facility layered on xrootd
 - Good architecture for “map/reduce” processing
- # Batch-nodes provide PROOF infrastructure
 - Reserve and use for interactive PROOF
 - Batch scheduler must have a drain/reserve feature
 - Use nodes as a parallel batch facility
 - Good for co-locating application with data
 - Use nodes as data providers for other purposes

PROOF Analysis Results



Akira's talk about "Panda oriented" ROOT analysis comparison at the Jamboree

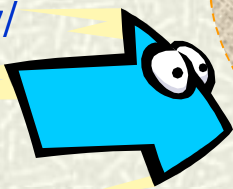
<http://indico.cern.ch/getFile.py/access?contribId=10&sessionId=0&resId=0&materialId=slides&confId=38991>

Expansive Clustering

- # Xrootd can create ad hoc cross domain clusters
 - Good for easily federating multiple sites
 - This is the ALICE model of data management
 - Provides a mechanism for “regional” data sharing
 - Get missing data from close by before using dq2get
 - Architecture allows this to be automated & demand driven
 - This implements a Virtual Mass Storage System

Virtual Mass Storage System

root://atlas.bnl.gov/
includes
SLAC, UOM, UTA
xroot clusters

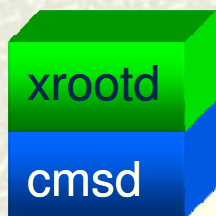


BNL



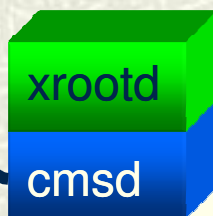
all.role meta manager
all.manager meta atlas.bnl.gov:1312

Meta Managers can be
geographically replicated!



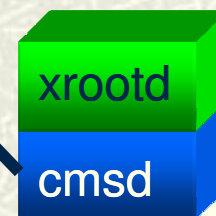
SLAC

all.role manager
all.manager meta atlas.bnl.gov:1312



UOM

all.role manager
all.manager meta atlas.bnl.gov:1312



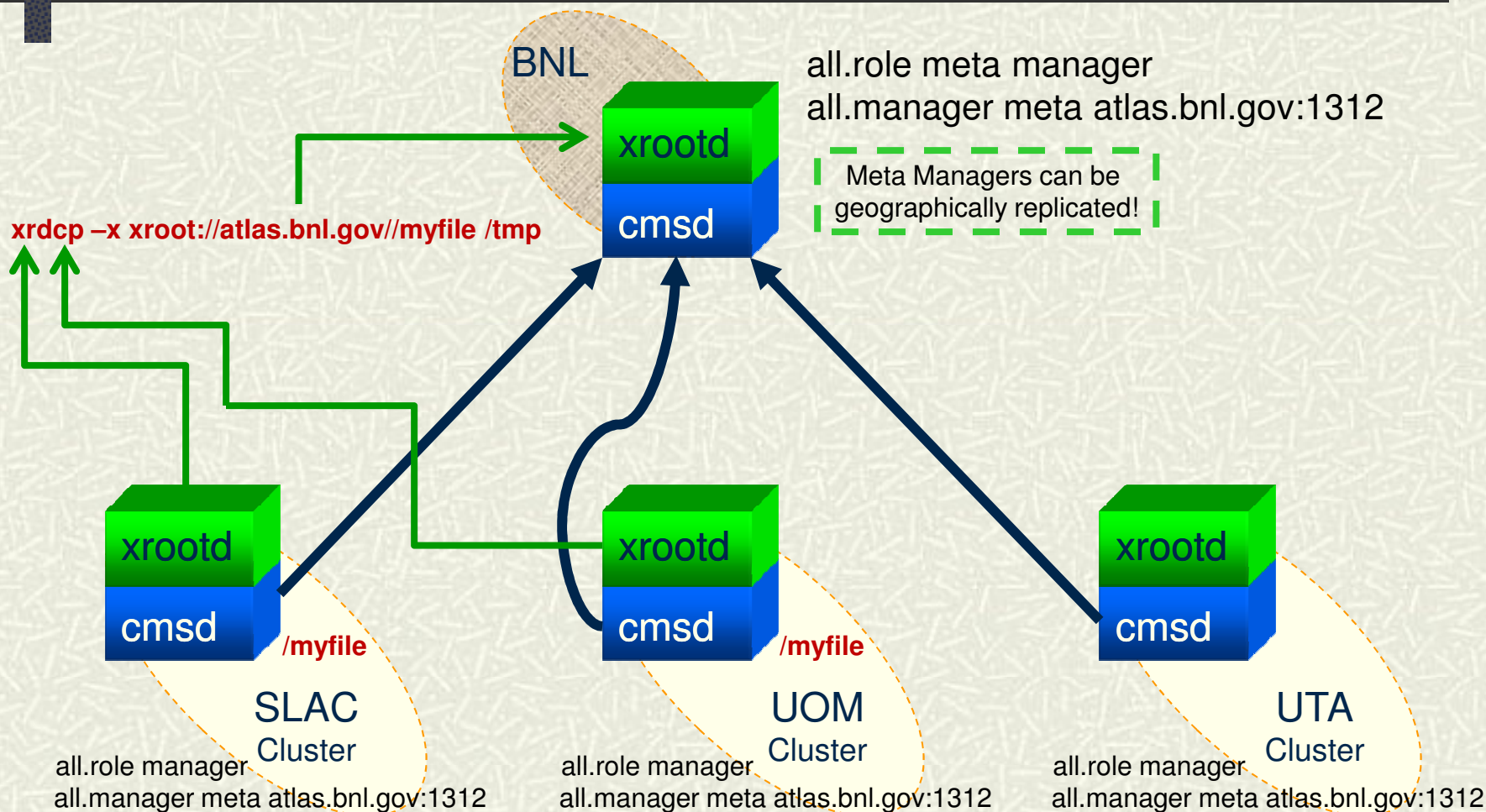
UTA

all.role manager
all.manager meta atlas.bnl.gov:1312

What's Good About This?

- # Fetch missing files in a timely manner
 - Revert to dq2get when file not in regional cluster
- # Sites can participate in an ad hoc manner
 - The cluster manager sorts out what's available
- # Can use R/T WAN access when appropriate
- # Can significantly increase WAN xfer rate
 - Using torrent-style copying

Torrents & Federated Clusters



Improved WAN Transfer

- # The xrootd already supports parallel TCP paths
 - Significant improvement in WAN transfer rate
 - Specified as `xrdcp -S num`
- # Xtreme copy mode uses multiple data sources
 - Specified as `xrdcp -x`
 - Transfers to CERN; examples:
 - 1 source (.de): 12MB/sec (1 stream)
 - 1 source (.us): 19MB/sec (15 streams)
 - 4 sources (3 x .de + .ru): 27MB/sec (1 stream each)
 - 4 sources + || streams: 42MB/Sec (15 streams each)
 - 5 sources (3 x .de + .it + .ro): 54MB/Sec (15 streams each)

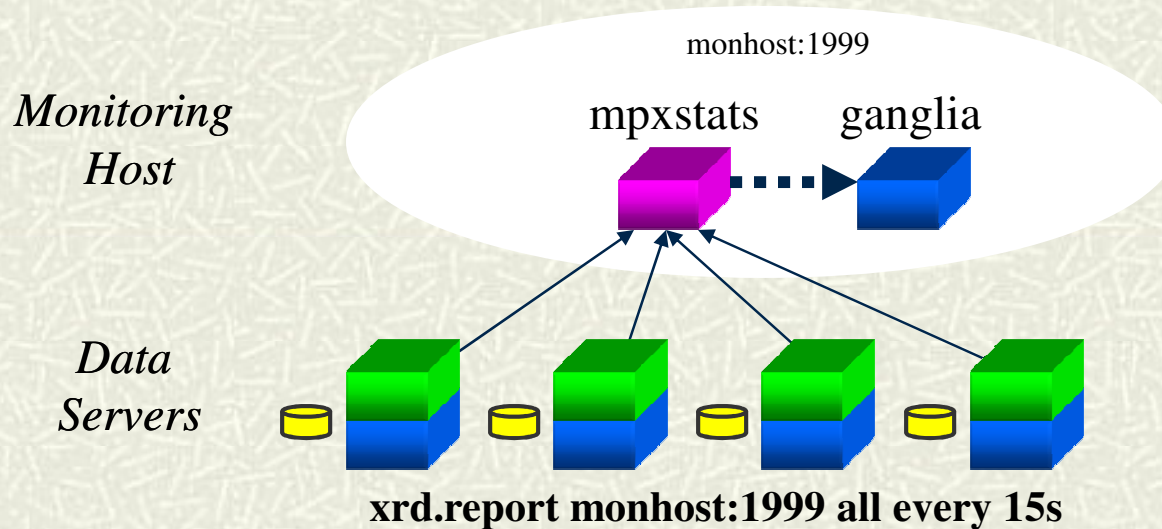
Expansive Clustering Caveats

- # Federation & Globalization are easy if
 - Federated servers are *not* blocked by a firewall
 - No ALICE xroot servers are behind a firewall
- # There are alternatives
 - Implement firewall exceptions
 - Need to fix all server ports
 - Use proxy mechanisms
 - Easy for some services, more difficult for others
- # All of these have been tried in various forms
 - Site's specific situation dictates appropriate approach

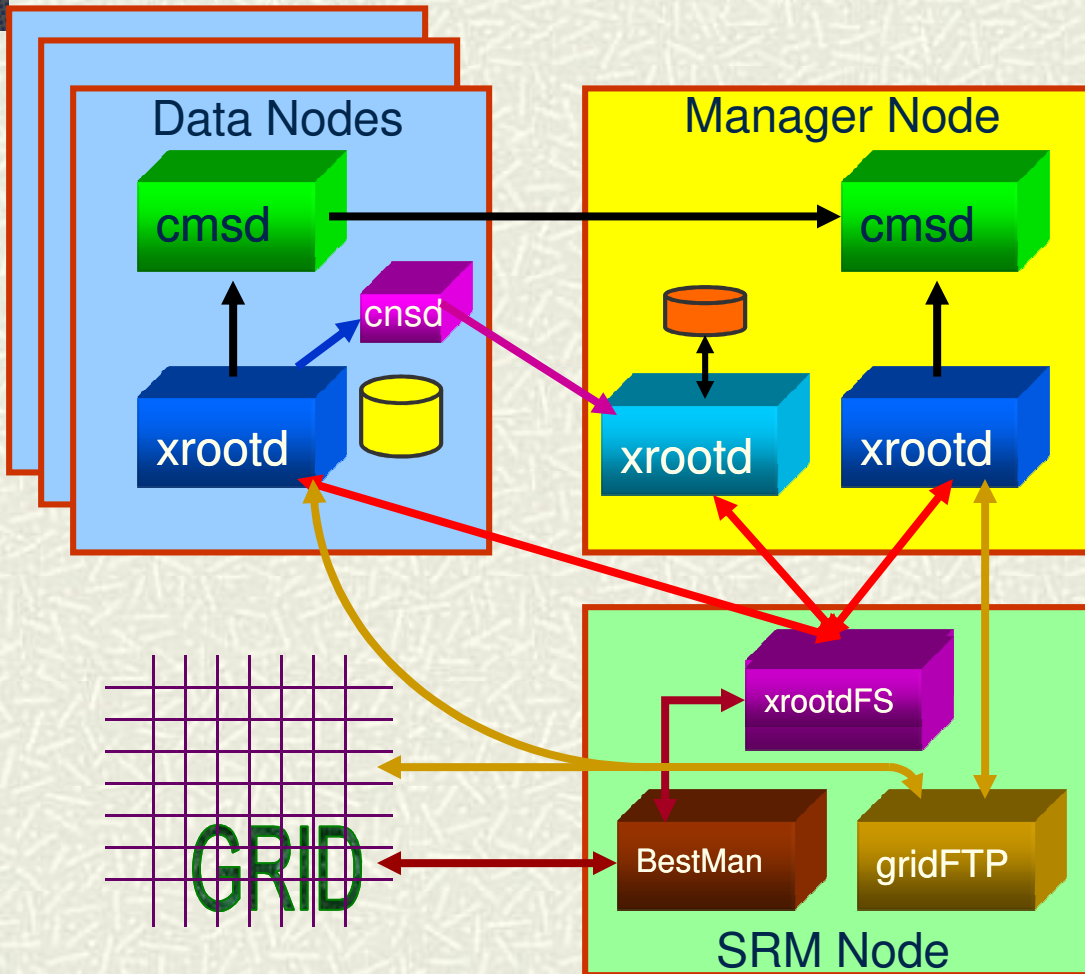
Summary Monitoring

- # Needed information in almost any setting
- # Xrootd can auto-report summary statistics
 - Specify **xrd.report** configuration directive
- # Data sent to one or two locations
 - Use provided **mpxstats** as the feeder program
 - Multiplexes streams and parses xml into key-value pairs
 - Pair it with any existing monitoring framework
 - Ganglia, GRIS, Nagios, MonALISA, and perhaps more

Summary Monitoring Setup



Putting It All Together

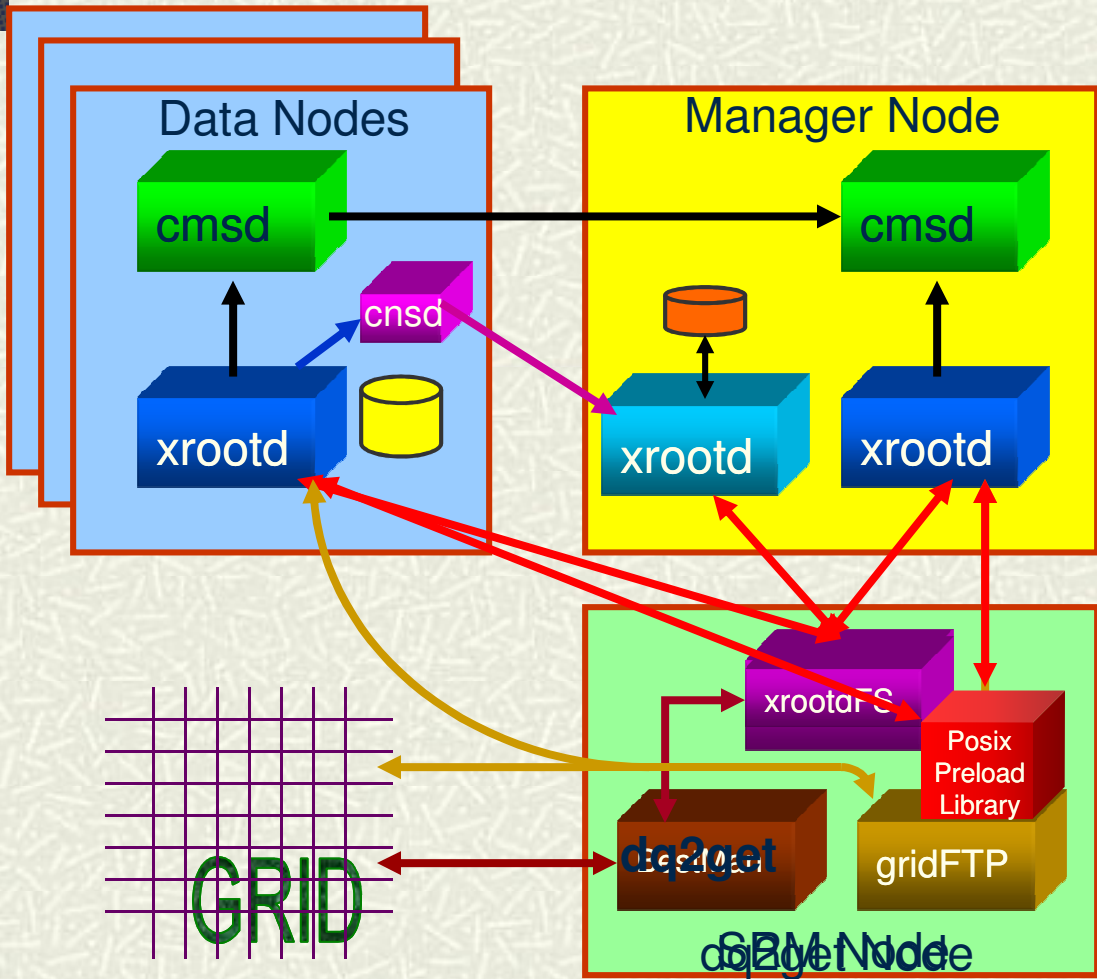


Basic **xrootd** Cluster
+
Name Space **xrootd**
+
cnsd
+
SRM Node
(BestMan, **xrootdFS**, gridFTP)
=
LHC Grid Access

Can't We Simplify This?

- # The **cnsd** present for **XrootdFS** support
 - Provide composite name space for “**ls**” command
- # **FUSE** present for **XrootdFS** support
- # **XrootdFS** & **FUSE** for **BeSTMan** support
- # **BeSTMan** for **SRM** support
- # **SRM** for push-type grid data management
 - **dq2get** is a pull function and only needs **gridFTP**
- # *Answer:* **Yes!** This can be simplified.

Tearing It All Apart



Basic `xrootd` Cluster
 +
 dq2get Node
 (gridFTP + POSIX Preload Lib)
 =
 Simple Grid Access

*Even more effective
 if using a VMSS*

In Conclusion. . .

- # Xrootd is a lightweight data access system
 - Suitable for resource constrained environments
 - Human as well as hardware
 - Geared specifically for efficient data analysis
- # Supports various clustering models
 - E.g., PROOF, batch node clustering and WAN clustering
 - Has potential to greatly simplify Tier 3 deployments
- # Distributed as part of the OSG VDT
 - Also part of the CERN root distribution
- # Visit <http://xrootd.slac.stanford.edu/>

Acknowledgements

Software Contributors

- Alice: Derek Feichtinger
- CERN: Fabrizio Furano , Andreas Peters
- Fermi/GLAST: Tony Johnson (Java)
- Root: Gerri Ganis, Beterand Bellenet, Fons Rademakers
- SLAC: Tofigh Azemmoon, Jacek Becla, Andrew Hanushevsky, Wilko Kroeger
- LBNL: Alex Sim, Junmin Gu, Vijaya Natarajan (BeStMan team)

Operational Collaborators

- BNL, CERN, FZK, IN2P3, RAL, SLAC, UVIC, UTA

Partial Funding

- US Department of Energy
 - Contract DE-AC02-76SF00515 with Stanford University