Any Data, Anytime, Anywhere

Dan Bradley <dan@hep.wisc.edu>
representing the AAA Team
At OSG All Hands Meeting
March 2013, Indianapolis

AAA Project

Goal

- Use resources more effectively through remote data access in CMS

Sub-goals

- Low-ceremony/latency access to any single event
- Reduce data access error rate
- Overflow jobs from busy sites to less busy ones
- Use opportunistic resources
- Make life at T3s easier

xrootd: Federating Storage Systems

- Step 1: deploy seamless global storage interface
- But preserve site autonomy:
  - xrootd plugin maps from global logical filename to physical filename at site
    - Mapping is typically trivial in CMS:
      /store/* → /store/*
  - xrootd plugin reads from site storage system
    - Example: HDFS
  - User authentication also pluggable
    - But we use standard GSI + lcmaps + GUMS

Status of CMS Federation

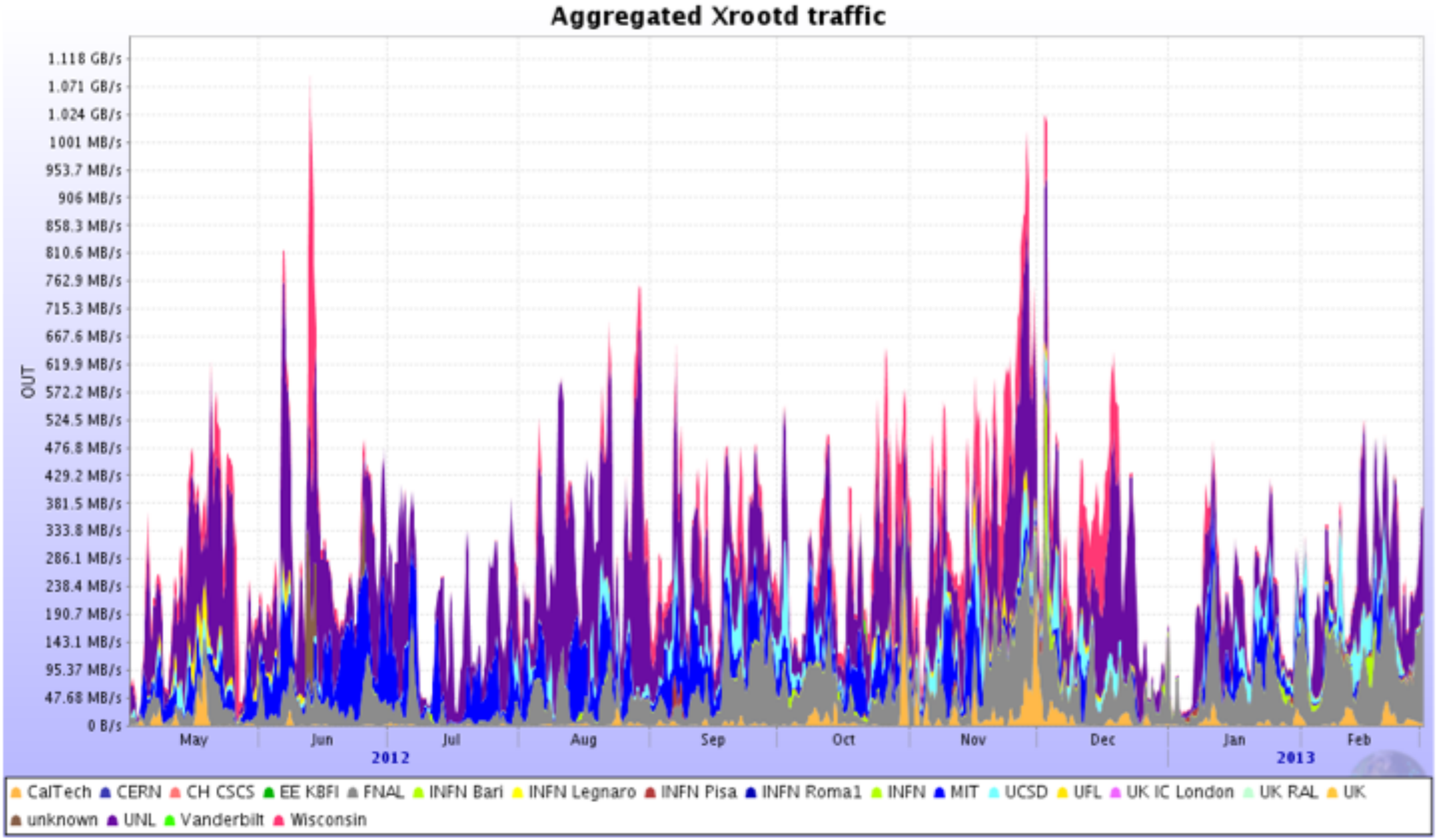
US

- T1 (disk) + 7/7 T2s federated
- Covers 100% of the data for analysis
- Does not cover files only on tape

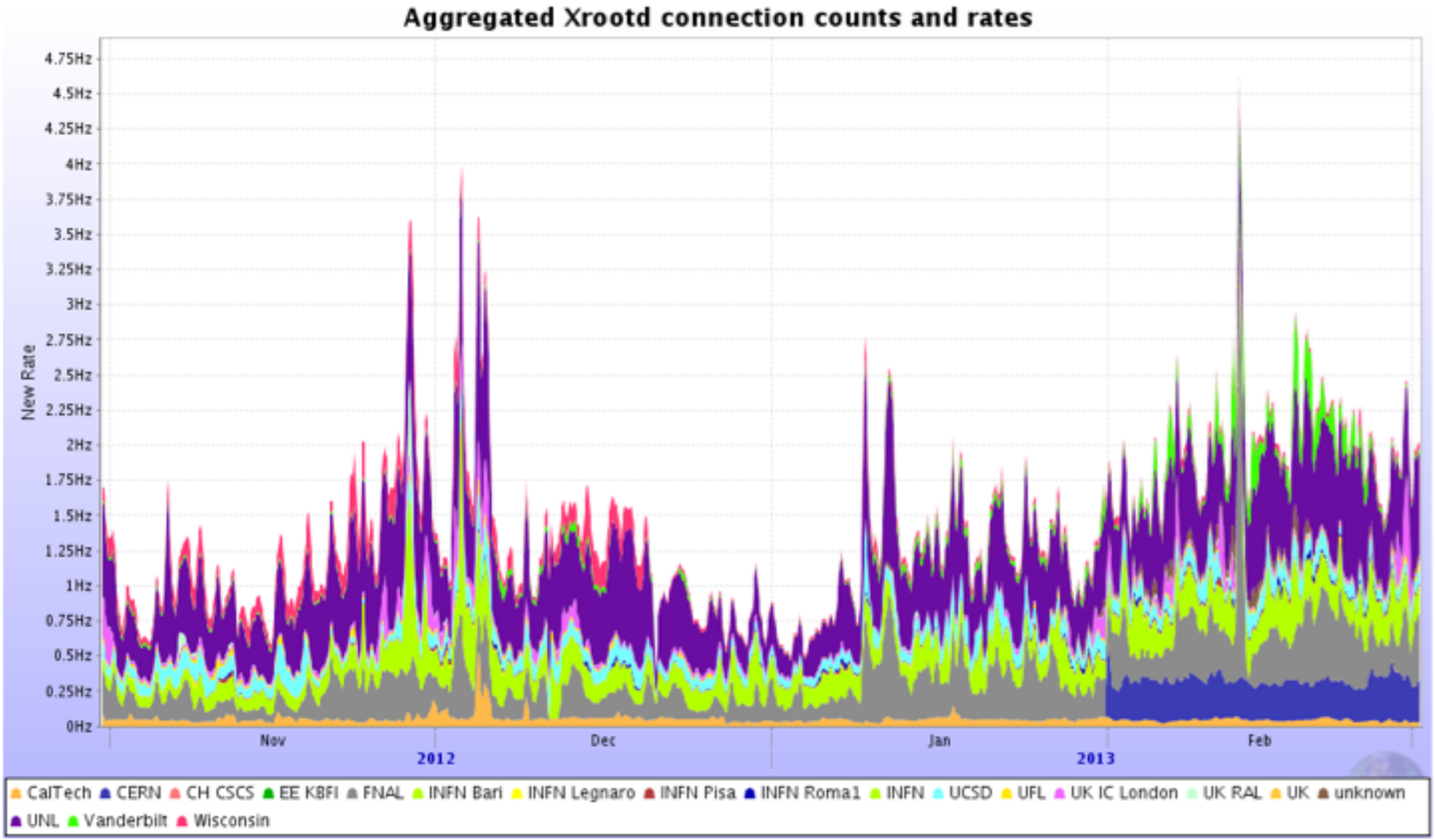
World

- 2 T1s + 1/3 T2s accessible
- Monitored but not a “turns your site red” service (yet)

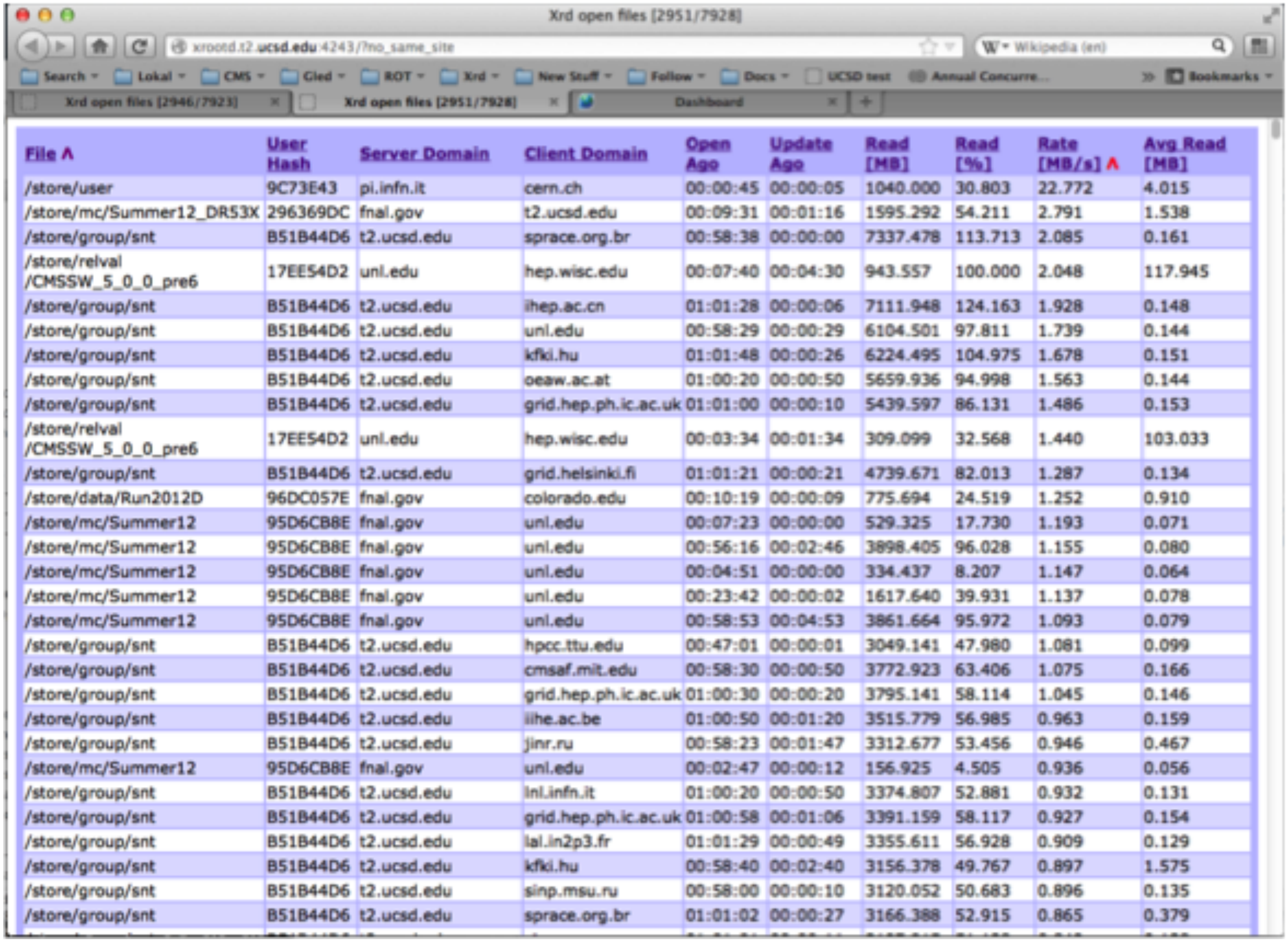
WAN xrootd traffic



Opening Files



Microscopic View



File ∧	User Hash	Server Domain	Client Domain	Open Ago	Update Ago	Read [MB]	Read [%]	Rate [MB/s] ∧	Avg Read [MB]
/store/user	9C73E43	pi.infn.it	cern.ch	00:00:45	00:00:05	1040.000	30.803	22.772	4.015
/store/mc/Summer12_DR53X	296369DC	fnal.gov	t2.ucsd.edu	00:09:31	00:01:16	1595.292	54.211	2.791	1.538
/store/group/snt	B51B44D6	t2.ucsd.edu	sprace.org.br	00:58:38	00:00:00	7337.478	113.713	2.085	0.161
/store/relval /CMSSW_5_0_0_pre6	17EE54D2	unl.edu	hep.wisc.edu	00:07:40	00:04:30	943.557	100.000	2.048	117.945
/store/group/snt	B51B44D6	t2.ucsd.edu	ihep.ac.cn	01:01:28	00:00:06	7111.948	124.163	1.928	0.148
/store/group/snt	B51B44D6	t2.ucsd.edu	unl.edu	00:58:29	00:00:29	6104.501	97.811	1.739	0.144
/store/group/snt	B51B44D6	t2.ucsd.edu	kfki.hu	01:01:48	00:00:26	6224.495	104.975	1.678	0.151
/store/group/snt	B51B44D6	t2.ucsd.edu	oeaw.ac.at	01:00:20	00:00:50	5659.936	94.998	1.563	0.144
/store/group/snt	B51B44D6	t2.ucsd.edu	grid.hep.ph.ic.ac.uk	01:01:00	00:00:10	5439.597	86.131	1.486	0.153
/store/relval /CMSSW_5_0_0_pre6	17EE54D2	unl.edu	hep.wisc.edu	00:03:34	00:01:34	309.099	32.568	1.440	103.033
/store/group/snt	B51B44D6	t2.ucsd.edu	grid.helsinki.fi	01:01:21	00:00:21	4739.671	82.013	1.287	0.134
/store/data/Run2012D	96DC057E	fnal.gov	colorado.edu	00:10:19	00:00:09	775.694	24.519	1.252	0.910
/store/mc/Summer12	95D6CB8E	fnal.gov	unl.edu	00:07:23	00:00:00	529.325	17.730	1.193	0.071
/store/mc/Summer12	95D6CB8E	fnal.gov	unl.edu	00:56:16	00:02:46	3898.405	96.028	1.155	0.080
/store/mc/Summer12	95D6CB8E	fnal.gov	unl.edu	00:04:51	00:00:00	334.437	8.207	1.147	0.064
/store/mc/Summer12	95D6CB8E	fnal.gov	unl.edu	00:23:42	00:00:02	1617.640	39.931	1.137	0.078
/store/mc/Summer12	95D6CB8E	fnal.gov	unl.edu	00:58:53	00:04:53	3861.664	95.972	1.093	0.079
/store/group/snt	B51B44D6	t2.ucsd.edu	hpcc.ttu.edu	00:47:01	00:00:01	3049.141	47.980	1.081	0.099
/store/group/snt	B51B44D6	t2.ucsd.edu	cmsaf.mit.edu	00:58:30	00:00:50	3772.923	63.406	1.075	0.166
/store/group/snt	B51B44D6	t2.ucsd.edu	grid.hep.ph.ic.ac.uk	01:00:30	00:00:20	3795.141	58.114	1.045	0.146
/store/group/snt	B51B44D6	t2.ucsd.edu	iihe.ac.be	01:00:50	00:01:20	3515.779	56.985	0.963	0.159
/store/group/snt	B51B44D6	t2.ucsd.edu	jinr.ru	00:58:23	00:01:47	3312.677	53.456	0.946	0.467
/store/mc/Summer12	95D6CB8E	fnal.gov	unl.edu	00:02:47	00:00:12	156.925	4.505	0.936	0.056
/store/group/snt	B51B44D6	t2.ucsd.edu	lnl.infn.it	01:00:20	00:00:50	3374.807	52.881	0.932	0.131
/store/group/snt	B51B44D6	t2.ucsd.edu	grid.hep.ph.ic.ac.uk	01:00:58	00:01:06	3391.159	58.117	0.927	0.154
/store/group/snt	B51B44D6	t2.ucsd.edu	lal.in2p3.fr	01:01:29	00:00:49	3355.611	56.928	0.909	0.129
/store/group/snt	B51B44D6	t2.ucsd.edu	kfki.hu	00:58:40	00:02:40	3156.378	49.767	0.897	1.575
/store/group/snt	B51B44D6	t2.ucsd.edu	sinp.msu.ru	00:58:00	00:00:10	3120.052	50.683	0.896	0.135
/store/group/snt	B51B44D6	t2.ucsd.edu	sprace.org.br	01:01:02	00:00:27	3166.388	52.915	0.865	0.379

Problem

- Access via xrootd overloads site storage system
  - Florida to Federation, “We are seceding!”
- Terms of the Feb 2013 treaty:
  - Addition of local xrootd I/O load monitoring
  - Site can configure automatic throttles
    - When load too high, rejects new transfer requests
    - End-user only sees error if file unavailable elsewhere in federation
- But these policies are intended for the exception, not the norm, because ...

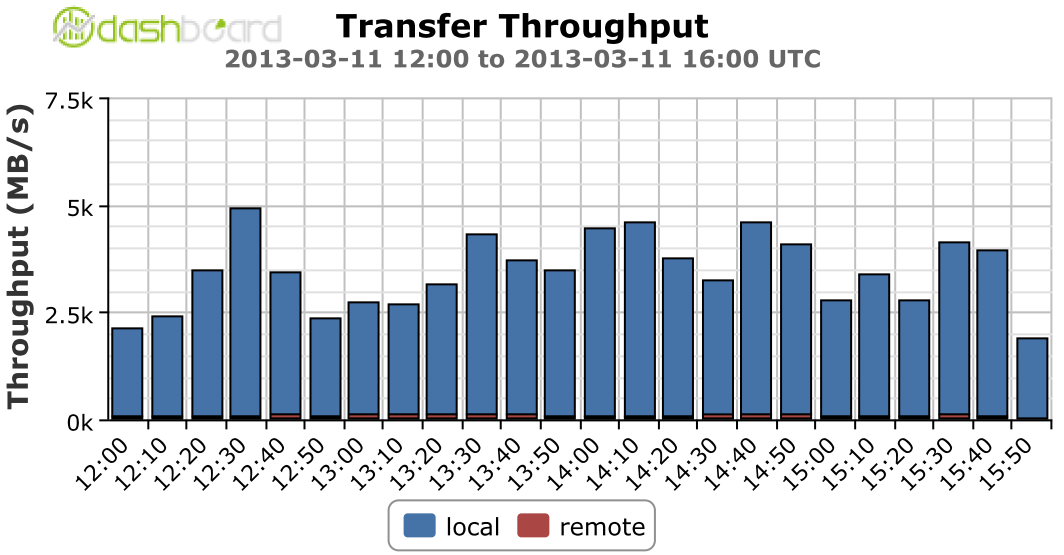
Regulation of Requests

- To 1st order, jobs still run at sites with the data
  - ~0.25 GB/s average remote read rate
  - O(10) GB/s average local read rate
  - ~1.5 GB/s PhEDEx transfer rate
- Cases where data is read remotely:
  - Interactive - limited by # humans
  - Fallback - limited by error rate opening files
  - Overflow - limited by scheduling policy
  - Opportunistic - limited by scheduling policy
  - T3 - watching this

At the Campus Scale

Some sites are using xrootd for access to data from across a campus grid

- Examples: Nebraska, Purdue, Wisconsin



More on Fallback

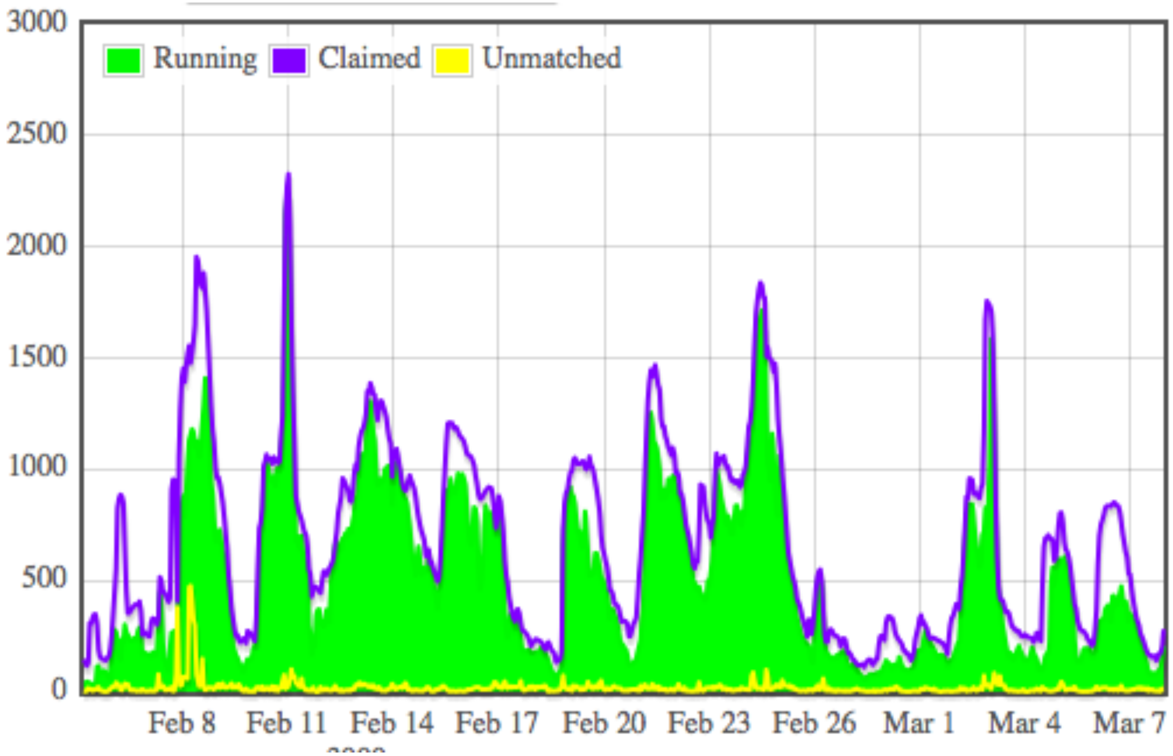
- On file open error, CMS software can retry via alternate location/protocol
  - Configured by site admin
  - We fall back to regional xrootd federation
    - US, EU
    - Could also have inter-region fallback
      - Have not configured this … yet
  - Can recover from missing file error, but not missing block within file error (more on this later)
  - Has more uses than just error recovery ...

More about Overflow

- GlideinWMS scheduling policy
  - Candidates for overflow:
    - Idle jobs with wait time above threshold (6h)
    - Desired data available in a region supporting overflow
  - Regulation of overflow:
    - Limited number of overflow glideins submitted per source site
- Data access
  - No reconfiguration of job required
    - Uses fallback mechanism
    - Try local access, fall back to remote access on failure

Overflow

- Small but steady overflow in US region



Running Opportunistically

- To run CMS jobs at non-CMS sites, we need
  - Outbound network access
  - Access to CMS datafiles
    - Xrootd remote access
  - Access to conditions data
    - http proxy
  - Access to CMS software
    - CVMFS (also needs http proxy)

CVMFS Anywhere

But non-CMS sites might not happen to mount the CMS CVMFS repository

→ Run the job under Parrot (from cctools)

- Can now access CVMFS without FUSE mount
- Also gives us identity boxing
  - Privilege separation between glidein and user job
- Has worked well for guinea pig analysis users
  - Working on extending it to more users

What about in the cloud?

- If you control the VM image, just mount CVMFS

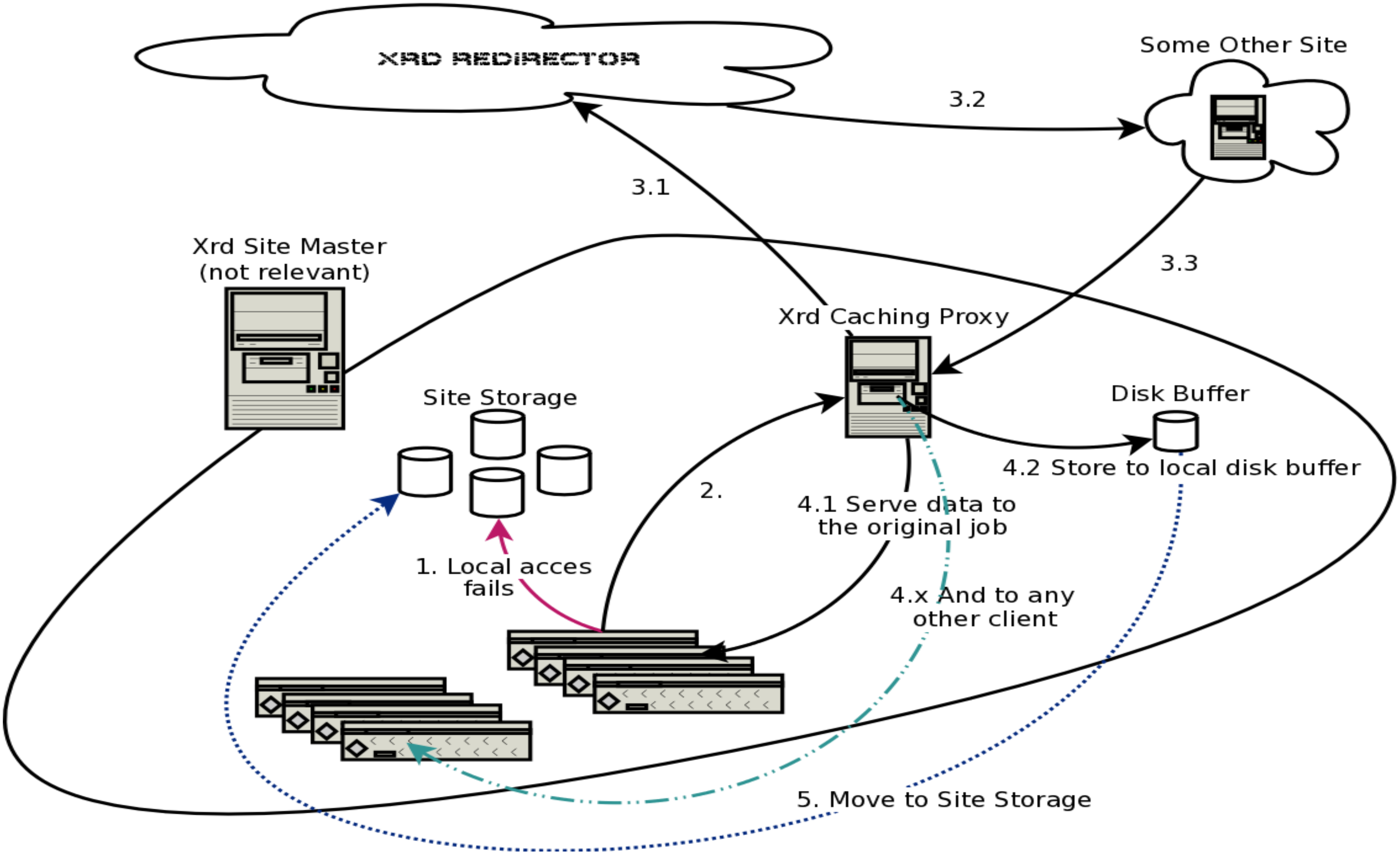
Fallback++

- Today we can recover when file is missing from local storage system
- But missing blocks within files cause jobs to fail
  - And job may come back and fail again ...
  - Admin may need to intervene to recover the data
  - User may need to resubmit the job
- Can we do better?

Yes, We Hope

- Concept
  - Fall back on read error
  - Cache remotely read data
  - Insert downloaded data back into storage system

File Healing



File Healing Status

- Currently have it working via whole-file caching
  - Still only triggered by file open error
- Plans to support partial-file healing
  - Will need to fall back to local xrootd proxy on all read failures
  - Current implementation is HDFS-specific
    - Modifies HDFS client to do the fallback to xrootd
    - But it's not CMS-specific

Cross-site Replication

- Once we have partial-file healing …
  - Could reduce HDFS replication level from 2 to 1 and use cross-site redundancy instead
    - Would need to enforce the replication policy at higher level
    - May not be good idea for hot data
    - Need to consider impact on performance

Performance

Mostly CMS application-specific stuff

- Improved remote read performance by combining multiple reads into vector reads
  - Eliminates many round-trips
- Working on bit-torrent-like capabilities in CMS application
  - Read from multiple xrootd sources
  - Balance load away from slower source
  - React in O(1) minute time frame

HTCondor Integration

- Improved vanilla universe file transfer scheduling and monitoring in 7.9
  - Used to have one file transfer queue
    - One misconfigured workflow could starve everything else
    - Difficult to diagnose
  - Now one per user
    - Or per arbitrary attribute (e.g. target site)
  - Equal sharing between transfer queues in case of contention
  - Reporting transfer status, bandwidth usage, disk load, and network load
  - And now you can condor_rm those malformed jobs that are transferring GBs of files :)

Summary

- xrootd storage federation rapidly expanding and proving useful within CMS
- We hope to do more
  - Automatic error recovery
  - Opportunistic usage
  - Improving performance