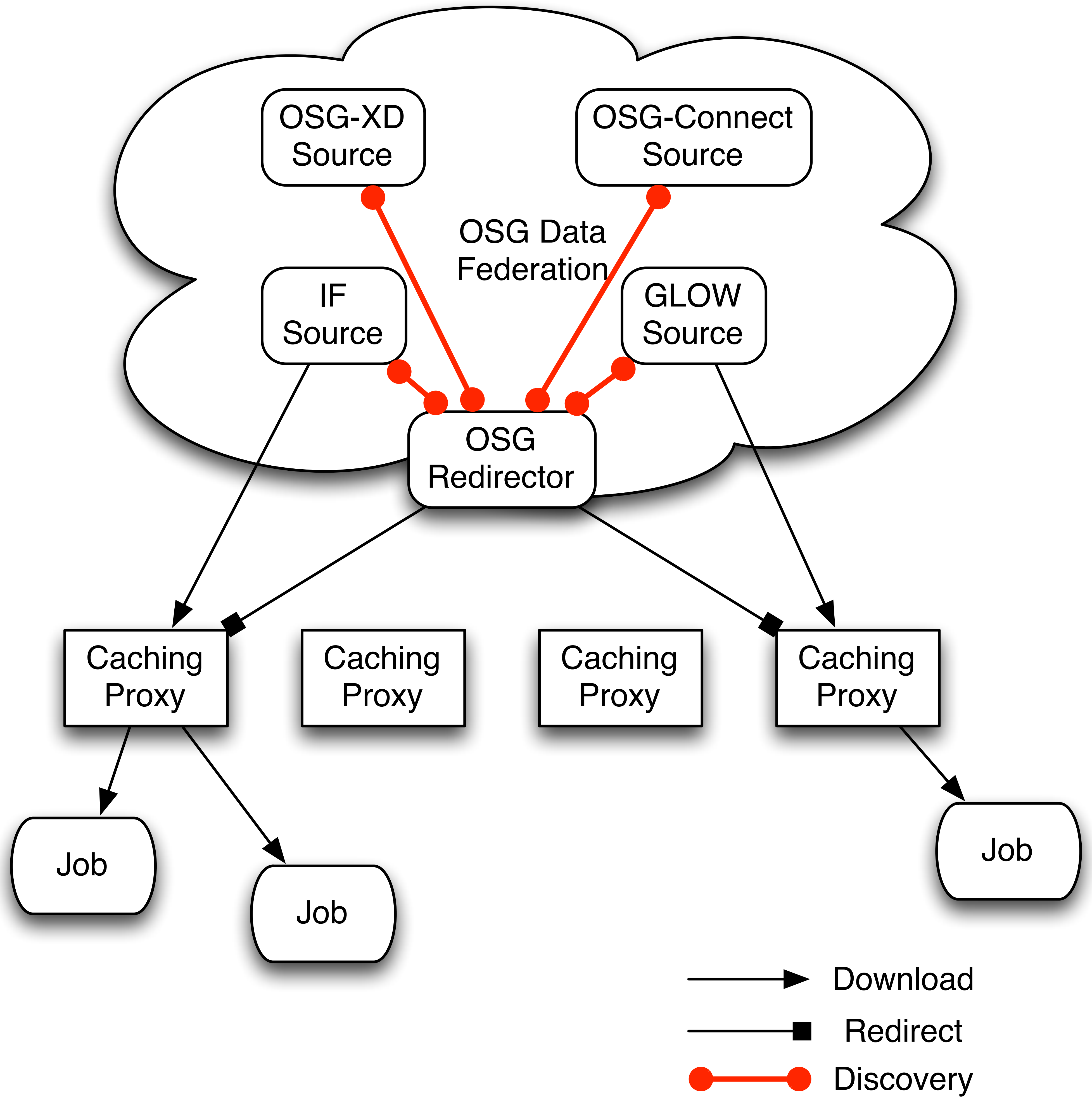
CVMFS for Data Federations

Brian Bockelman
University of Nebraska - Lincoln

Problems with Xrootd-based Data Federations

- No directory listings: more like GET/PUT.
- Foreign command line tools: `xrdcp` (or `stashcp`), not `cp`.
- They are difficult to setup for opportunistic VO's; OSG has already created one StashCache.
- **USERS WANT POSIX!**



File service -> File *system*

- Users, in short, want to turn the global, read-only file service into a **global, read-only filesystem**.
  - Further, we want one safe to mount on 100,000 hosts where users may do profoundly dumb things on the namespace ("The `ls -lhR` Problem").
- Well, this sounds familiar: CVMFS.

Can we use CVMFS to serve the POSIX interface and our data federations to serve the data?

Times They are A Changin’

- Xrootd data federation: the CVMFS FUSE process speaks HTTP. Our federation must export this protocol => **Use Xrootd’s HTTP(S) support**.
- CVMFS:
    - Client must follow HTTP-based redirects.
    - We *cannot* change, alter, or rename the files inside the federation for CVMFS.
        - Files in the data federation are saved by the *logical file name* and *uncompressed*.
        - CVMFS wants files saved by their *content address* (e.g., SHA-1 hash) and *compressed*.
        - CVMFS catalogs now have new file attributes that denote compression type (compressed / uncompressed) and storage type (file name / content address).
    - We need a mechanism to publish files without downloading them to the repository server.

Repositories

- We have a series of 4 repositories we maintain:
  - **nova.osgstorage.org** - Repo from XrootD data source at FNAL
  - **stash.osgstorage.org** - Repo built from user accessible storage at OSG-Connect
  - **cms.osgstorage.org** - Repo of the CMS data federation
  - **ligo.osgstorage.org** - Repo of LIGO data stored at Nebraska

Repositories

- Let’s take a closer look at two of these:
  - **nova.osgstorage.org** - Repo from XrootD data source at FNAL
  - **stash.osgstorage.org** - Repo built from user accessible storage at OSG-Connect
  - **cms.osgstorage.org** - Repo of the CMS data federation
  - **ligo.osgstorage.org** - Repo of LIGO data stored at Nebraska

stash.osgstorage.org

- Goal: publish the "Stash" filesystem at UChicago, exposed via the StashCache data federation, into CVMFS:

  1. A periodic job scans the Stash filesystem at UChicago, recording differences since last scan.

     - This looks at the world-readable contents of `/stash/$USER/public`.

  2. Job puts records files' metadata (size, checksum) into the CVMFS repository server. Data stays on Stash.

  3. CVMFS repository is published with new contents.

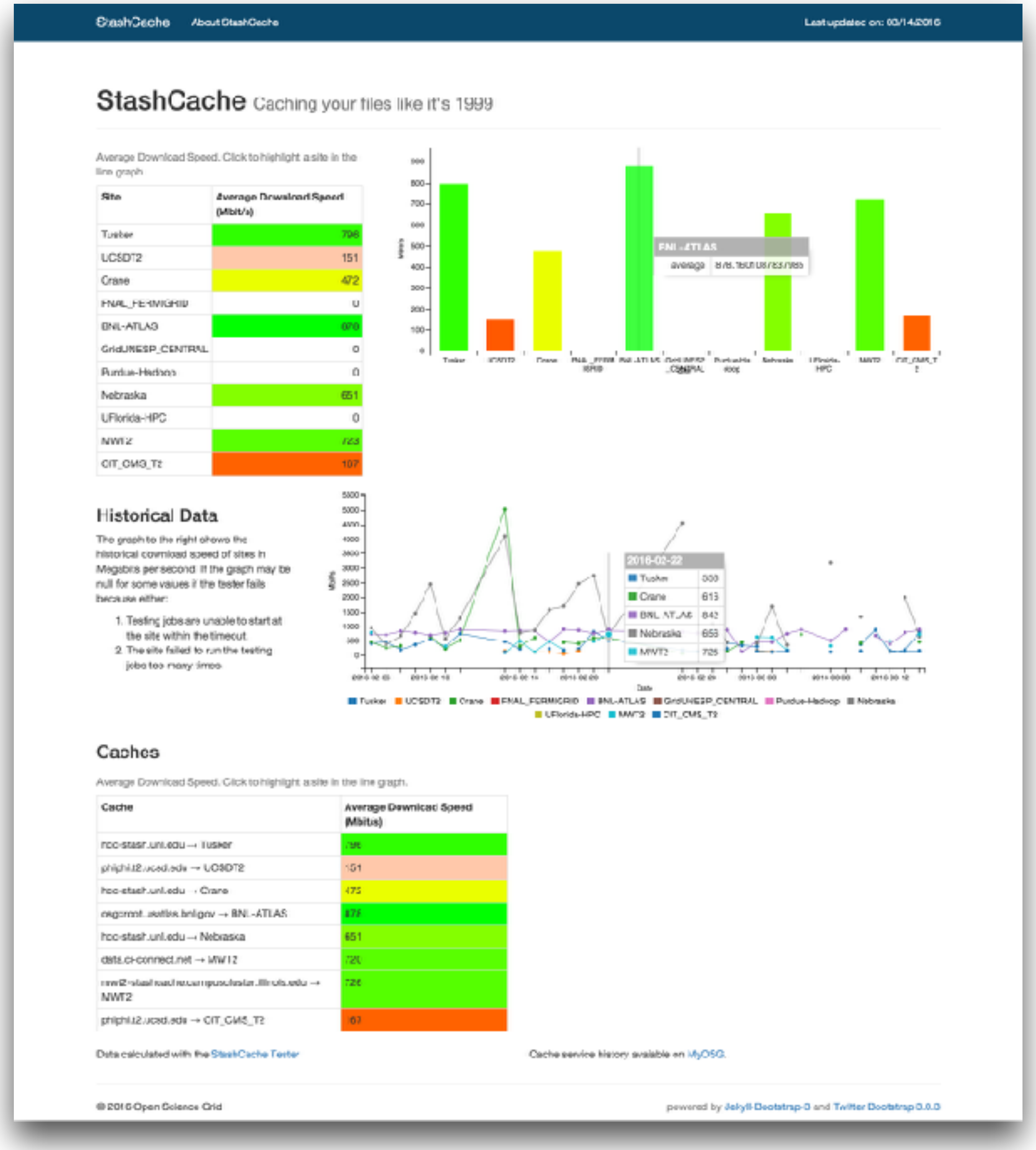
StashCache

- Managing data opportunistically at storage elements requires a CMS- or ATLAS-sized commitment.

- StashCache uses distributed caches across the country.

- Data origin is the Stash service on OSG-Connect.

- Users write data into Stash, and read the data from jobs through StashCache



StashCache About StashCache Last updated on: 03/14/2016

StashCache Caching your files like it's 1999

Average Download Speed. Click to highlight a site in the line graph

Site	Average Download Speed (Mbit/s)
Tusker	796
UCSDT2	151
Crane	472
FNAL_FERMIGRID	0
BNL-ATLAS	878
GridUNESP_CENTRAL	0
Purdue-Hadoop	0
Nebraska	651
UFlorida-HPC	0
NWT2	723
CIT_CMS_T2	107

Historical Data

The graph to the right shows the historical download speed of sites in Megabits per second. If the graph may be null for some values if the tester fails because either:

1. Testing jobs are unable to start at the site within the timeout.
2. The site failed to run the testing jobs too many times.

Caches

Average Download Speed. Click to highlight a site in the line graph.

Cache	Average Download Speed (Mbit/s)
hcc-stash.unl.edu → Tusker	796
phiphi.t2.ucsd.edu → UCSDT2	151
hcc-stash.unl.edu → Crane	472
osgxrootd.usatlas.bnl.gov → BNL-ATLAS	878
hcc-stash.unl.edu → Nebraska	651
data.ci-connect.net → MWT2	723
mwt2-stashcache.campuscluster.illinois.edu → NWT2	726
phiphi.t2.ucsd.edu → CIT_CMS_T2	107

Data calculated with the StashCache Tester

Cache service history available on MyOSG.

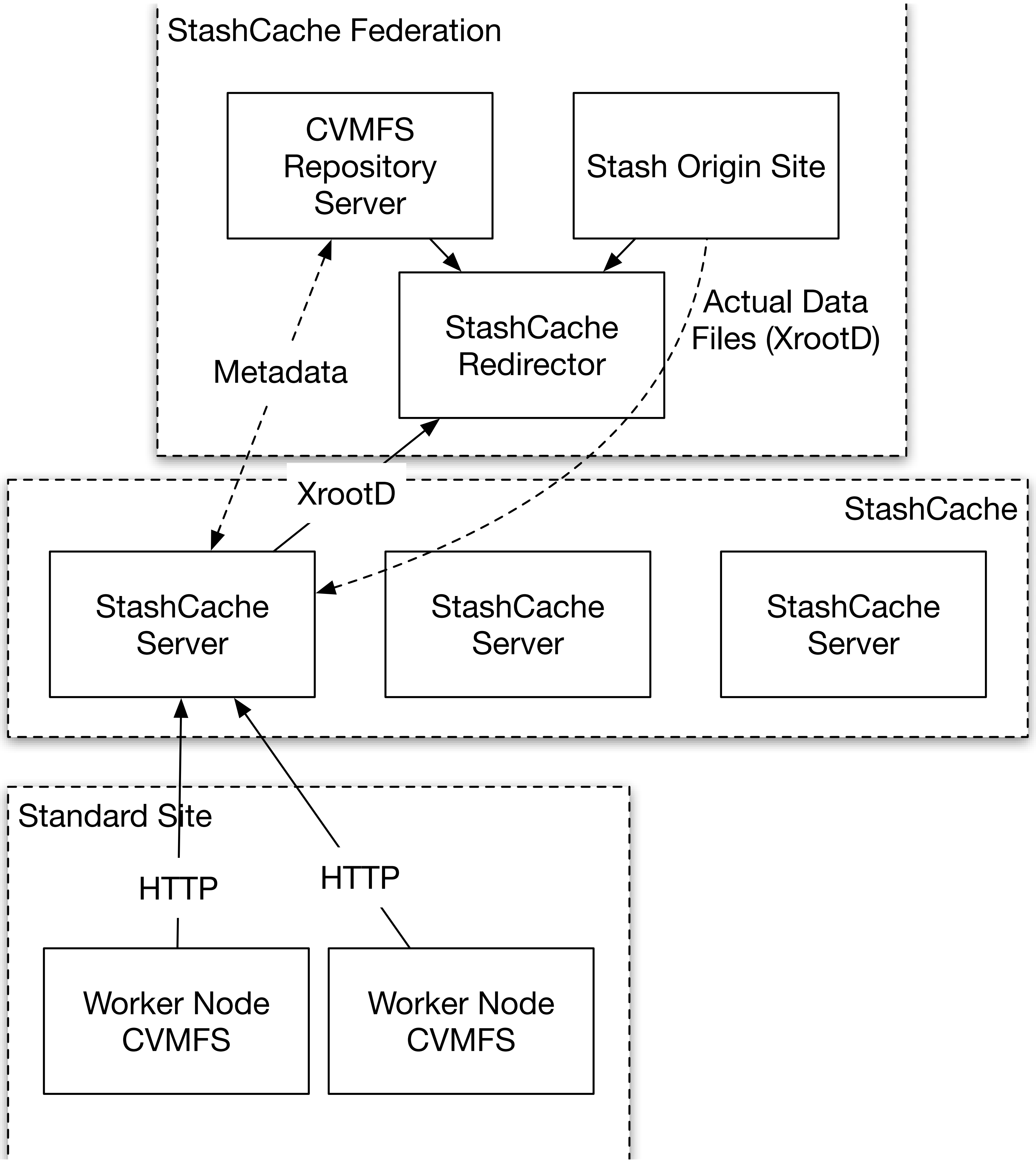
© 2016 Open Science Grid

powered by Jekyll-Bootstrap-3 and Twitter Bootstrap 3.0.3

stashcache.github.io

Overview of CVMFS and StashCache

- Regular XrootD StashCache Federation
- CVMFS contacts the caching servers over HTTP
- Caching servers contact the federation for the data.
  - Note the caching layer protects the origin server from load: very different from the CMS AAA model.
- Worker nodes pull data from the caching servers.



Uses

- Large datasets which cannot be cached within the existing Squid-based caching infrastructure (tuned for working set size of 10GB):
  - Full Blast DB’s
  - Nova Flux Files…
- Targeting working set sizes* from 10GB to 10TB. Will work fine for smaller sizes, but OASIS may be more efficient for software distribution.

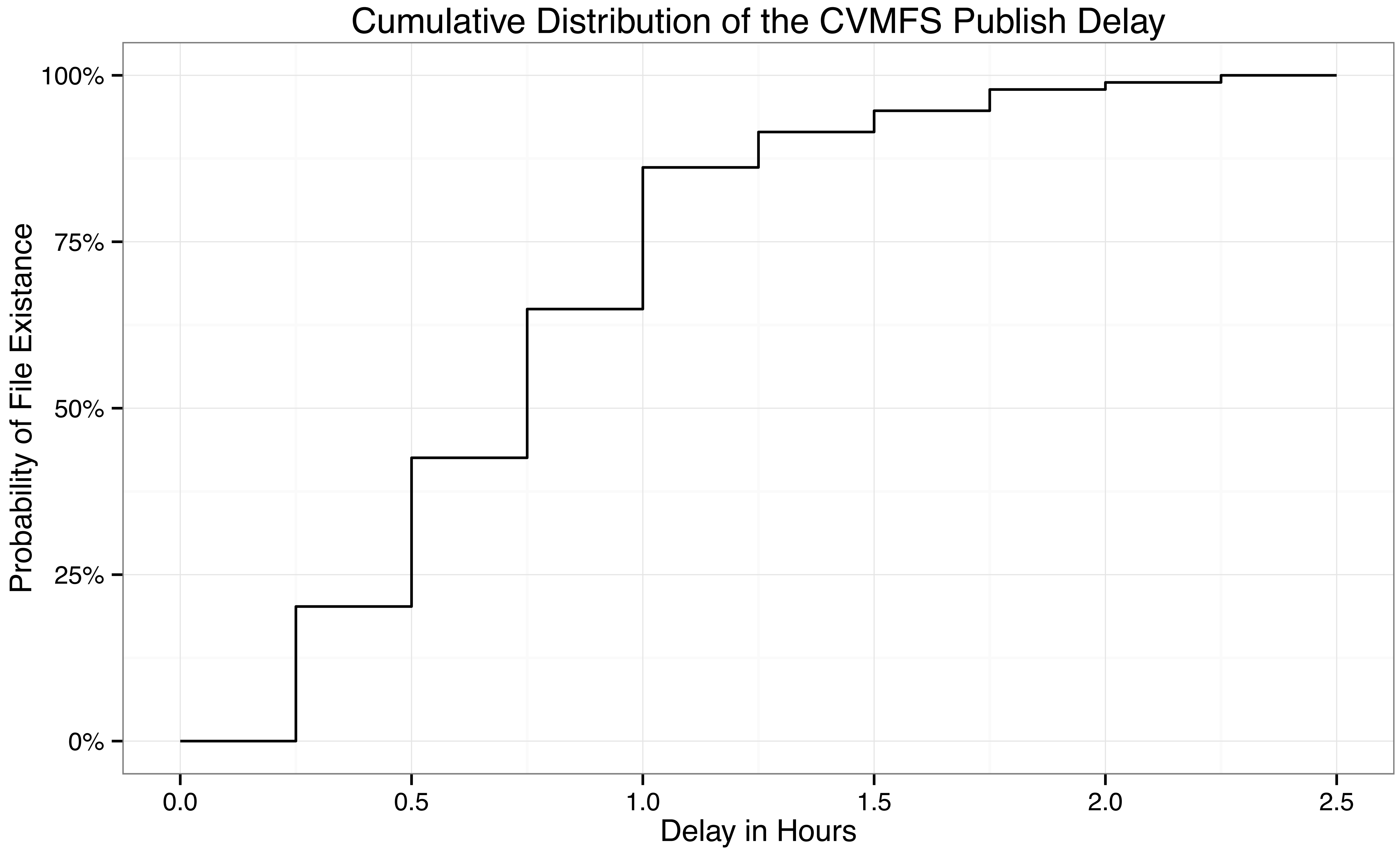
*Number of unique bytes touched by a workflow

User Perspective

- Copies data onto OSG-Connect using `scp`, Globus Online - pick your favorite.
- Put data into `/stash/<user>/public`
- Wait for a while for the data to be published (~1 hr)
- Use data on the worker nodes!

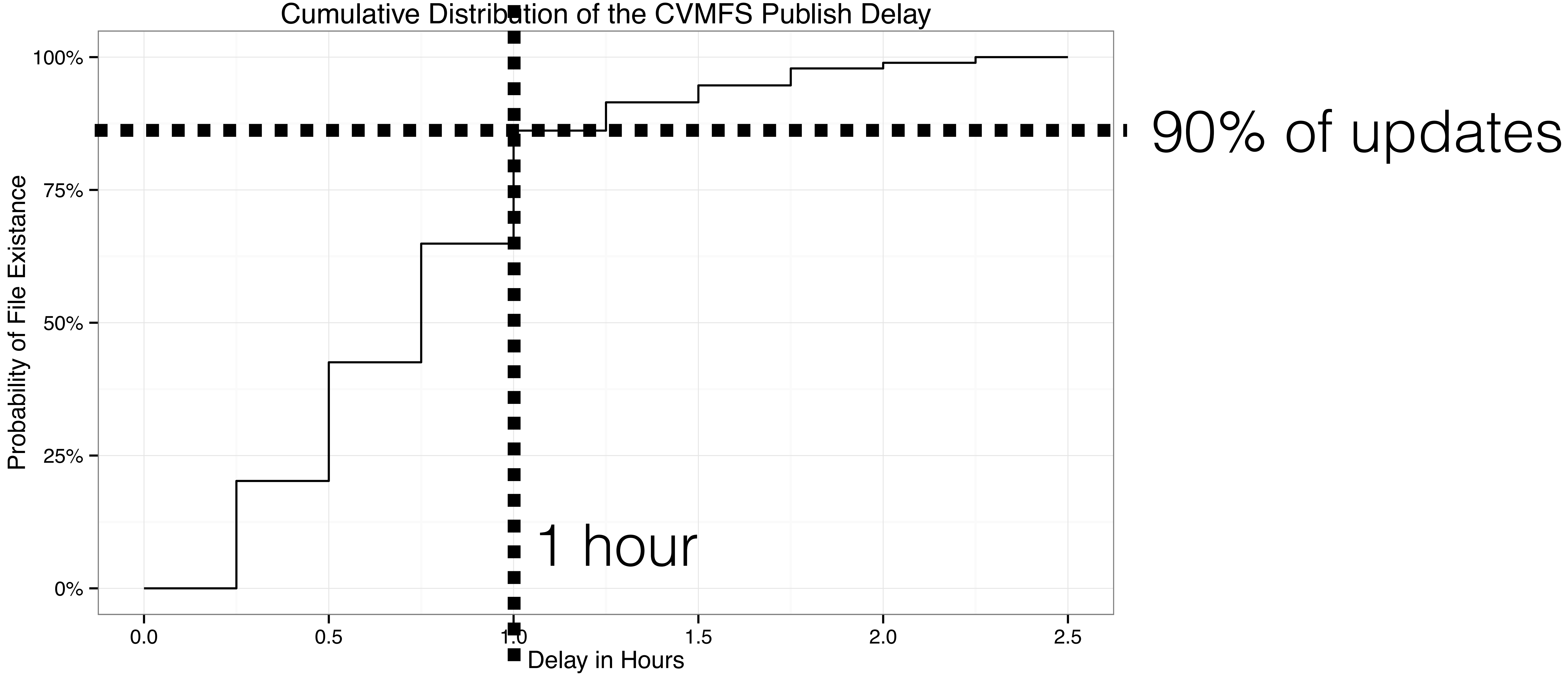
Stash -> CVMFS Delay

- There is a delay between when the file has been created, and when the it appears in CVMFS.



Stash -> CVMFS Delay

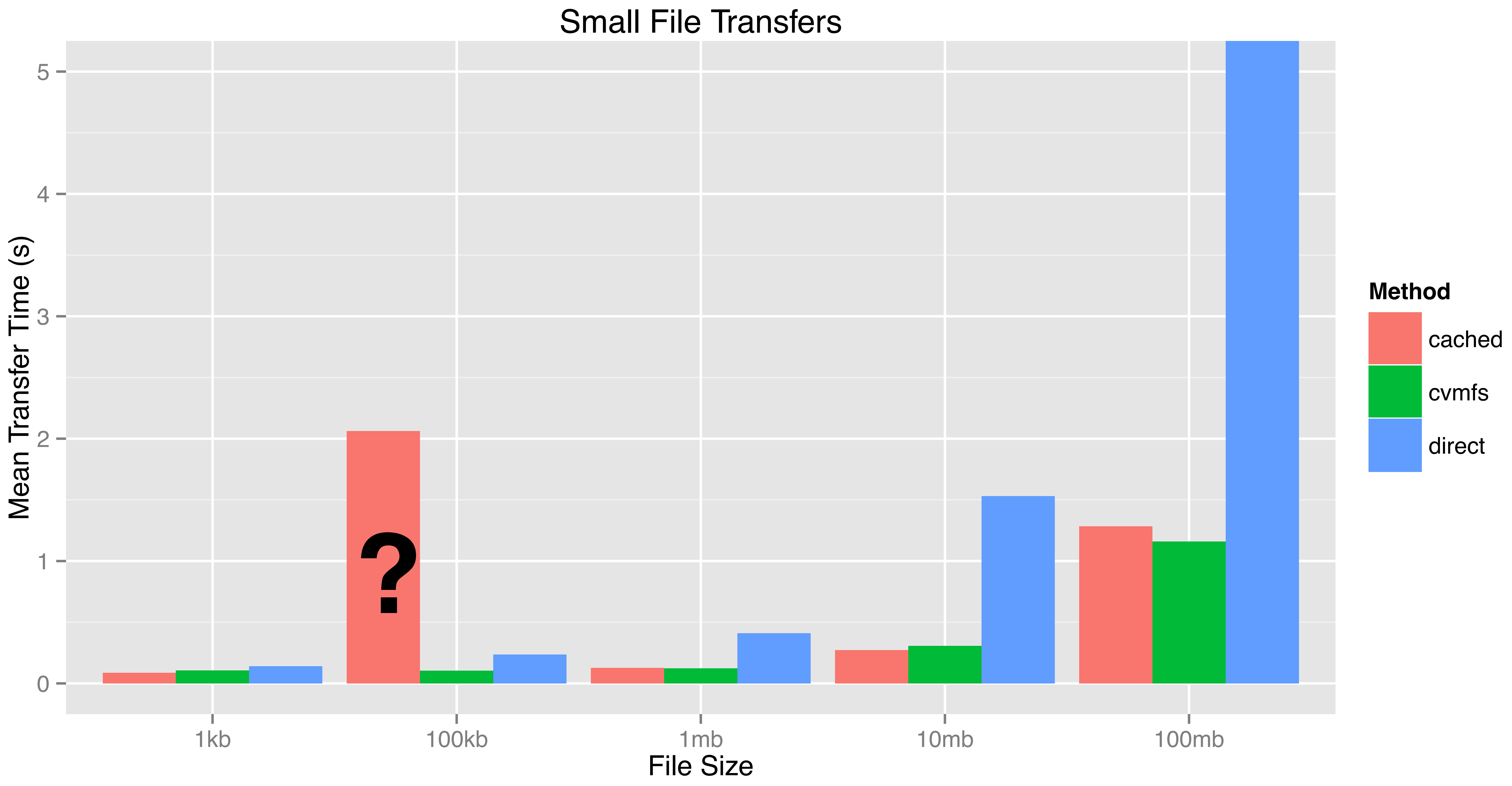
- In 1 hour, the files are largely available



Still opportunity to squeeze this further!

Performance - Small Files

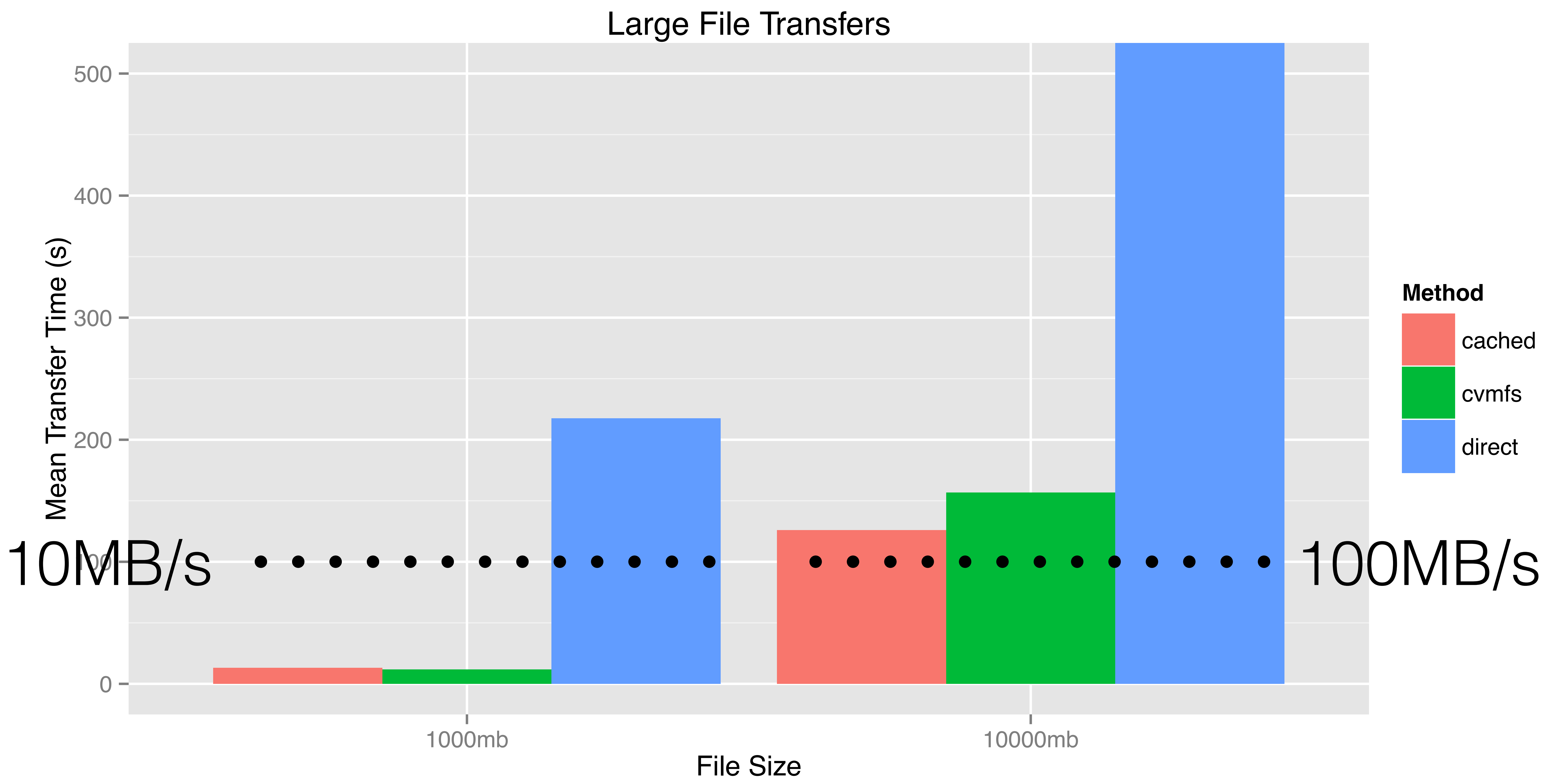
- CVMFS caching is roughly equal to that of using StashCache



Performance - Larger Files

- CVMFS caching is roughly equal to that of using StashCache

Lower is better



10MB/s

100MB/s

CVMFS + StashCache

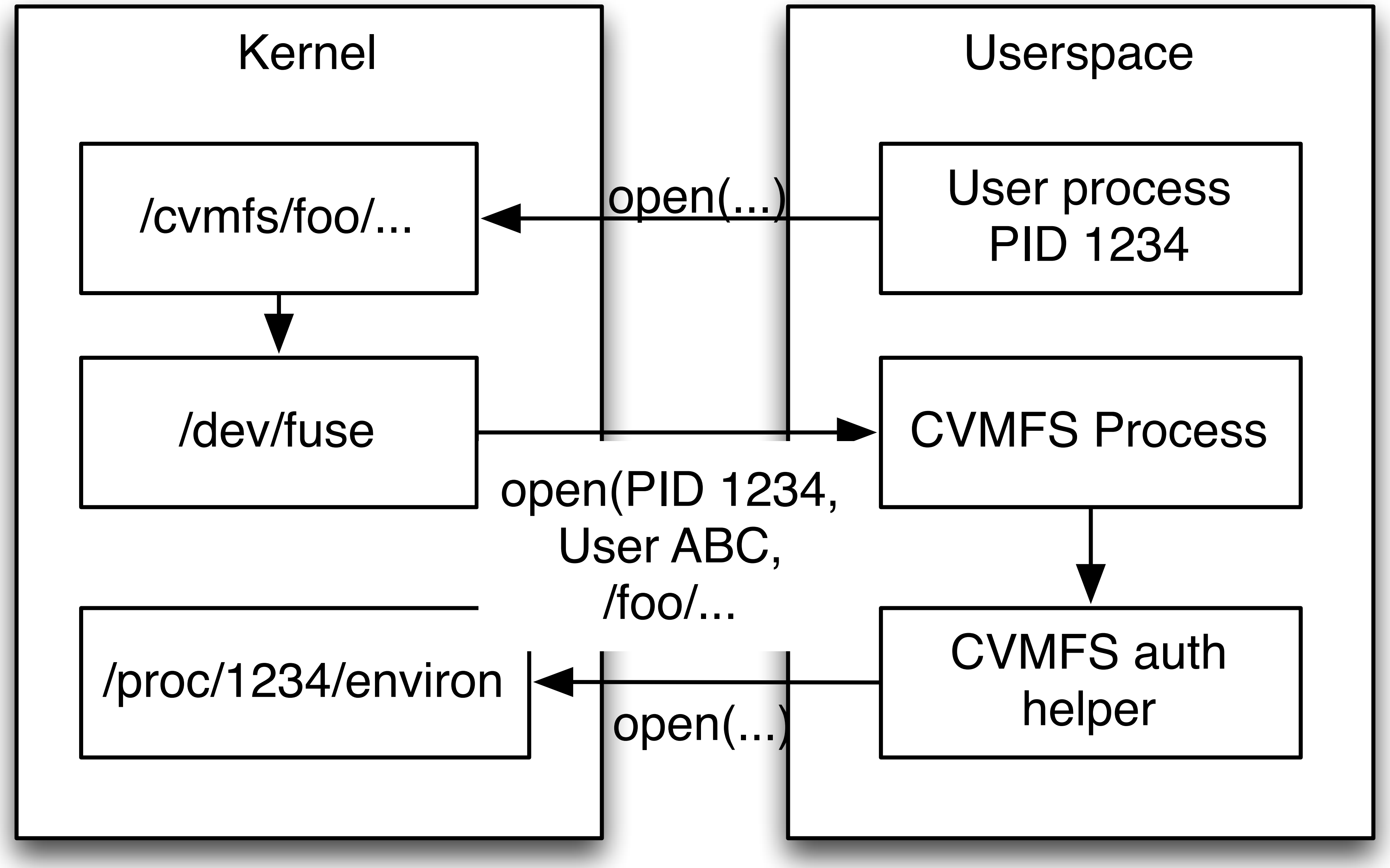
- We finally have a **global, scalable, read-only filesystem**.
- We have analogous setups for the NoVA and DES experiments.
- Writable by all OSG VO users.
- Access at `/cvmfs/stash.osgstorage.org/`.

ligo.osgstorage.org

- Problem: `stash.osgstorage.org` is unauthenticated access to public files
    - Not great for event data!
- LIGO has very specific rules about data access and even namespace visibility
- Therefore, had to develop new features in CVMFS to enable VOMS authentication.

Secure CVMFS

- FUSE provides CVMFS with the PID/UID/GID of the accessing process.
- CVMFS uses a helper process to, in turn, acquire an appropriate credential from the accessing process.
  - Currently, this is an X509 proxy.
  - Helper process also enforces authorization to the repository.
- The proxy is returned to the CVMFS process and used to secure the HTTPS connection.

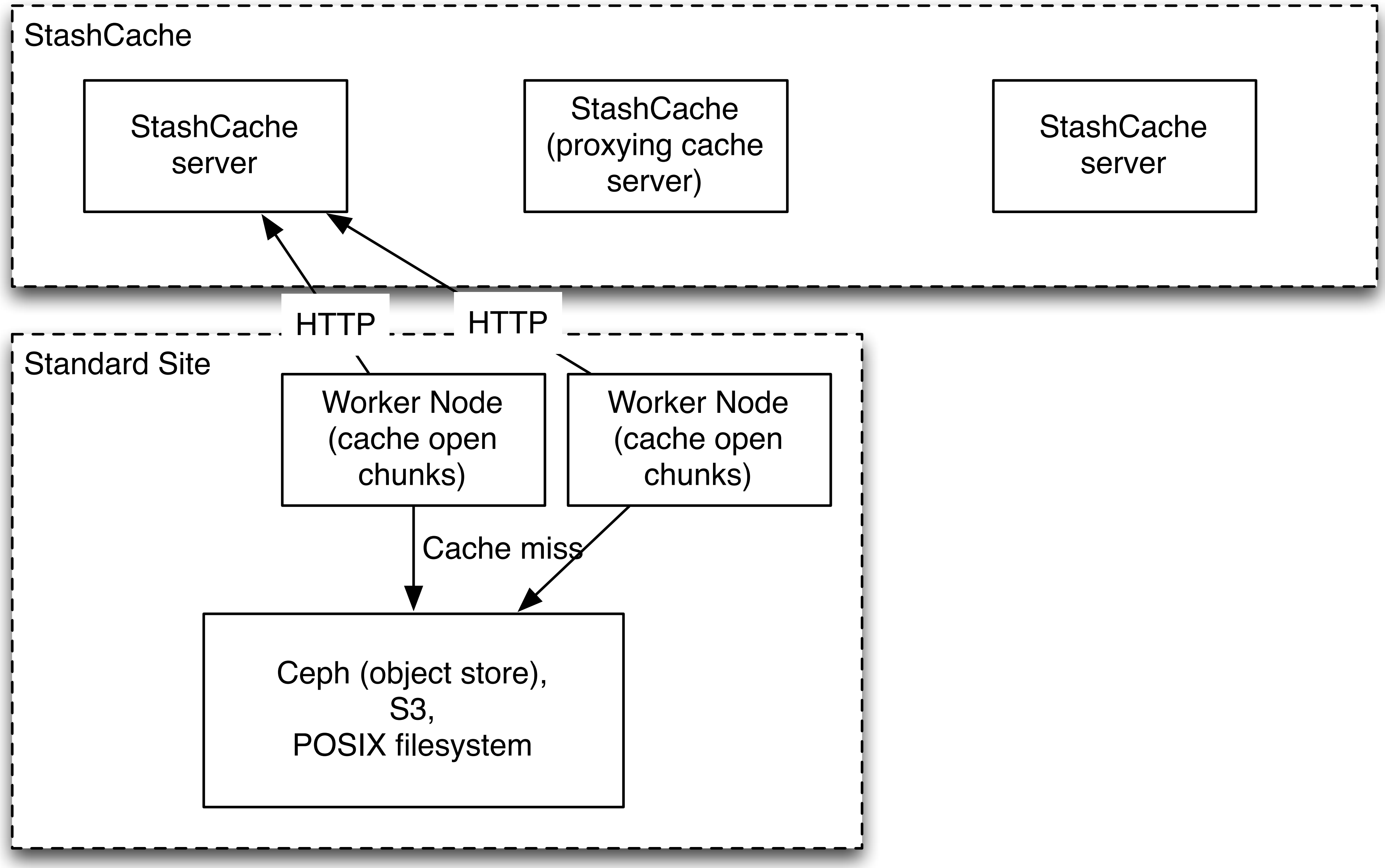


Secure CVMFS

- The HTTPS server must secure access to the data catalog (we use mod_gridsite).
  - Alternately, decide that the filenames are not proprietary and just use HTTP.
- The data federation is responsible for authenticating and authorizing the HTTPS connections from the CVMFS process.
  - Anywhere HTTPS is used, caching is not possible in general.
- Obviously, the root user can always see the parts of the namespace and the files in the worker node cache.

The Long Road Ahead

- The `osgstorage.org` repositories provide a POSIX filesystem: there's a long road to making it look like EXT3!
  - The "global transaction" approach may fit poorly wit some experiments. Looking to add the ability for a remote host to update a subset of the directory tree.
  - Still fits poorly with handling intermediate outputs of workflows. **Everything needs to go faster!**
  - Linux-kernel-side work needed for unprivileged mounting of FUSE.
- Take advantage of storage resources at sites: more intelligent **site-level caching is needed**.


StashCache
StashCache server
StashCache (proxying cache server)
StashCache server
HTTP
HTTP
Standard Site
Worker Node (cache open chunks)
Worker Node (cache open chunks)
Cache miss
Ceph (object store), S3, POSIX filesystem

Question?
Comments?
Heckling?