Global Xrootd Demonstrator

Brian Bockelman
January 2011 GDB

About this Presentation

- I'm going to be talking about three things:
  - Review of what we're doing. Largely a short-version of the last GDB presentation.
  - Summary of accomplishments.
  - Discussion of what's missing.

Summary Vision

- Provide a service which breaks the job/data locality dependency currently in LHC.
- Provide a remote I/O capability allowing data access outside LHC data centers at an acceptable loss of efficiency.

Goals

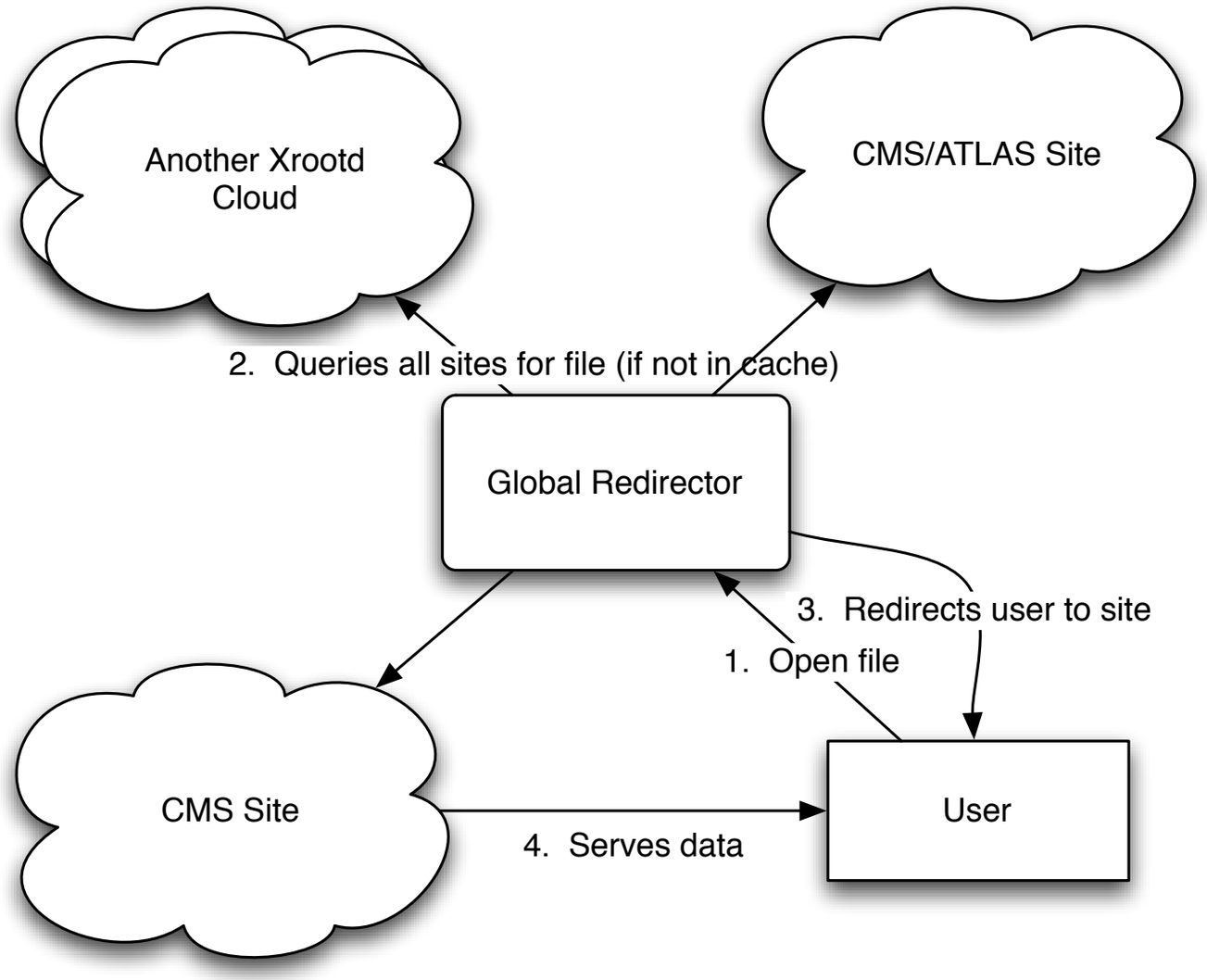
- **Reliability**: Seamless failover, even between sites.
- **Transparency**: Open the same filename regardless of source site.
- **Usability**: Must be native to ROOT.
- **Global**: Cover as many CMS files as possible.

Targets

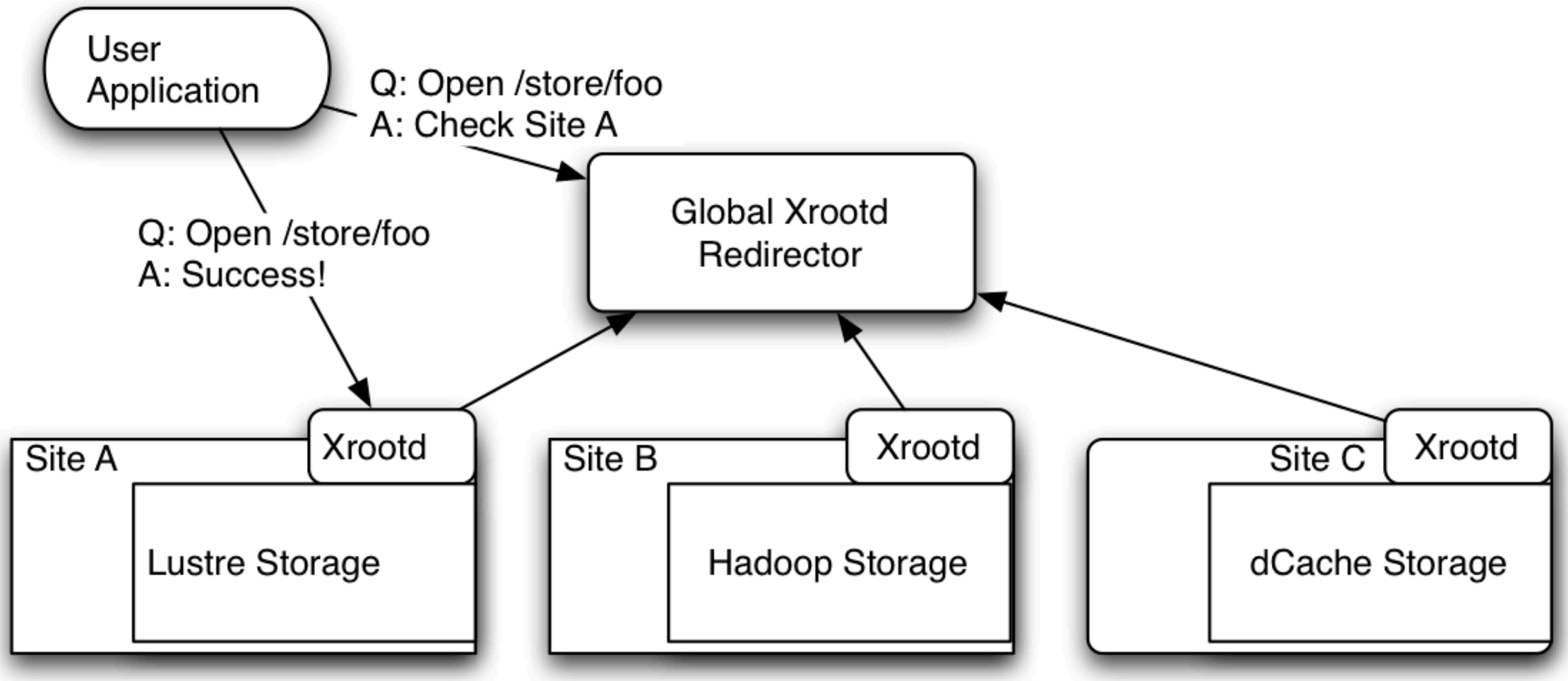
- The target users of this project:
  - *End-users*: event viewers, running on a few CMS files, sharing files with a group.
  - *T3s*: Running medium-scale ntuple analysis
- None of these users are well-represented by CMS tools right now.

NOT A TARGET: Wholesale replacement of the LHC data management infrastructure.

Global Xrootd Federation



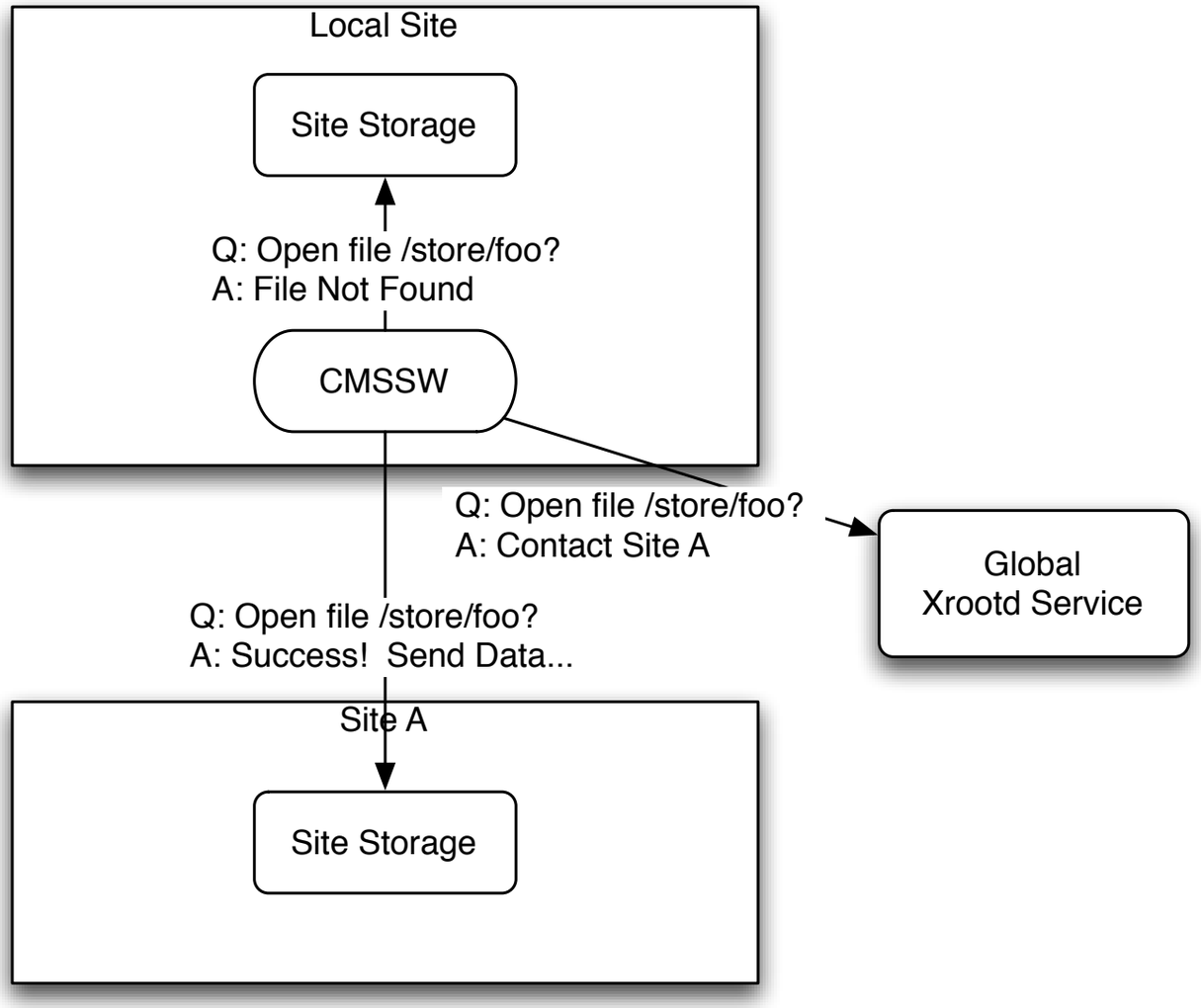
Xrootd Architecture



Xrootd layers on top of *existing storage element.*
Think of it as a proxy (or a door) to the site's data

“Fallback” Case

- CMSSW_3_9_x includes the ability to open a file remotely if the local file is missing.



Xrootd Prototype

- Have a global redirector users can contact for all files.
  - Can use any ROOT-based app to access the prototype infrastructure! Each file only has one possible URL
- Each participating site deploys at least 1 xrootd server that acts like a proxy/door to the external world.

Xrootd Prototype

- We don't change the site storage.
- Rather, we just provide a proxy that exports data in the xrootd protocol and:
  - Federates itself with the other xrootd instances.
  - Secures everything with X509/GSI/ VOMS.
  - Requires the SE to have a stable C API.

Again

- This is used for access *external* to the site.
- It is agnostic to the access protocol for the jobs inside the site.
  - My personal preference is POSIX-like (FUSE/NFSv4.1) internal to the site.
- Designed for new user *capabilities*, not new *capacity*. Proxies won't set Gbps records.

Demonstrator Accomplishments

- Improve performance of CMSSW under modestly high latency. Loss in efficiency is notable, but acceptable.
- Running of a test infrastructure. One T1, multiple T2s (US and EU), 2 T3s
- Demonstrate the functionality of a "disk-free" T3.
- Adoption by a group of physics.

Accomplishments

- Integrated dCache (explored 3 different mechanisms), HDFS, Lustre,
- Map the CMS namespace to local storage in Xrootd.
- Integrate XACML authentication into xrootd.
- Packaging into RPMs / config templates.

Very Recently

- The dCache.org team has their independent xrootd protocol implementation.
- Most recent versions add GSI security.
- I've tested remote CMSSW access with GSI; only one (minor) issue remains.
- Not yet integrated in the global namespace.

Namespace Propagation Issues

- It is easy to imagine this infrastructure would allow an innocent user to launch a DDoS against site namespaces.
- There is a positive/negative response cache at the redirector.
  - The intelligence of this cache is low, and must get better in the future. Key short-term requirement.
- Improved redirector placement helps.

Prototype to Production

- A demonstrator doesn't become a production service off the bat.
- Needs integration into the larger monitoring context.
- Improve our reliability metrics. How pleasant is the user experience?
- Decide on number and location of redirectors.

Prototype to Production

- We need a reasonable strategy for integrating dCache into the system. At least three different approaches are on the table.
- Data behavior changes => network behavior changes. Plan appropriately.
- Improved marketing and acceptance.
  - Would change user habits (hard).
  - Would need site buy-in (easier, but still hard).

To Cache or not To Cache?

- This work demonstrates Xrootd could be turned into a viable cache-based infrastructure for T3s.
  - I'm not worried about showing scaling to T2 levels, although this must be done.
- This work does not address whether we *should* cache, or if caching would be efficient for LHC. See ATLAS's work though...

Conclusions

- "Any data, Anywhere": We have performed a viable, medium-scale demonstration of a user-friendly remote-data infrastructure.
  - There are issues remaining, but none appear to be blockers.
- It would be possible to continue to grow this project *alongside* the production infrastructure.
  - I think data caching can be achieved with the same technology.
  - However, I think data caching and remote data access can be done in two separate decisions.