Moving Data from Campus to and from the OSG

Tanya Levshina

Talk Outline

- Data movement methods and limitations
- Storage Architecture for Open Science Grid (OSG) sites
- OSG Public Storage
- Summary

Acknowledgments

This presentation is compiled from multiple sources:

- Brian Bockelman & Derek Weitzel's lectures at the OSG Summer Grid Schools
- Conversations with the experts (Dave Dykstra, Marko Slyz, Gabriele Garzoglio, Parag Mhashilkar, Derek Weitzel and others)

Data Movement: Planning ahead

Open Science Grid

- You have to be able to answer to the following questions:
  - How many jobs do you want to run?
  - What should the submission rate be?
  - What is the duration of a job?
  - How much total storage you will need for input and output data?
  - Input data:
    - What are the number and size of input files per job?
    - Do jobs use a common set of files? What is the size of this set? What is the job access pattern to this set?
    - What is a current location of these files?
  - Output data:
    - What are the number and size of output files per job?
    - How long do you want to keep this data in transient storage?
    - What should be the transient and final storage destination for these files?

What doesn't work well on OSG

- Transferring big files >10GB
- Transferring a lot of small files
- Expecting POSIX-mounted storage accessible from worker nodes
- Expecting to have > 20 GB scratch space on a worker node

Data Movement Methods (I)

There are multiple ways for moving data to and from a grid job via:

- HTCondor (job sandbox) – data sent with a job
- GridFTP server on top POSIX-mounted storage area(Classic SE) – pre-staging data
- OSG SRM EndPoint – pre-staging and output data
- Squid server – caching data while job is running
- Parrot, XrootD (remote I/O) – streaming data upon job request

Data Movement Methods (II)

Common services built on top:

- Globus Online
- OSG Public Storage (iRODS) – OSG-XSEDE Campus Grid
- AAA (CMS) ; FAX (ATLAS)
- PhEDEx, FTS2 (CMS); DDM (ATLAS)
- SAMGrid/IFDH (Fermi FIFE Experiments)
- STASH – OSG Connect Campus Grid

You will need to decide which approach is the most efficient in your case.

GridFTP server on Compute Element

- OSG sites provide shared storage area:
  - accessible via GridFTP
  - POSIX-mounted storage (typically NFS) in most cases.
  - is mounted and writable on the CE head node.
  - readable and sometimes writable from a worker node.
  - limited by size of the shared area allocated for your VO.
- Usage pattern:
  - First, data is pre-staged into a shared area on a head node, then accessed from the worker nodes. In the majority of cases the access is POSIX compliant.
- Keep in mind:
  - not all the sites provide this storage.
  - simultaneous access may produce a heavy load on a shared area server.
  - unknown policy of space management.

Storage Element on OSG

- A SE is a cluster of nodes where data is stored and accessed: physical file systems, disk caches, hierarchical mass storage systems.
- Most sites have at least one SE. The SE has a SRM EndPoint. SRM (Storage Resource Management) is a web-services-based protocol that allows you to:
  - enforce authorization policies.
  - handle metadata.
  - do load balancing.
- Scalability and capacity of a SE significantly varies from site to site.
- A user interacts with a SE via a get or put of the file.
- A SE doesn't necessarily provide POSIX access.
- Usage Pattern:
  - First, pre-stage data into SE then download to a worker node by using srm-client or fuse mount. Upload output data into SE.

Squid

- Squid is a web caching service:
  - downloads requests from http servers
  - improves response times by caching and reusing frequently-requested web pages
  - installed on several OSG Sites. Mostly used on the OSG: for CRL downloads, to download common configuration files used by a VO, for software caching (CVMFS).
- Usage pattern:
  - A job running on a WN issues wget, curl to get a file. The file may be pulled from the web or be in the squid cache already.
- Keep in mind:
  - somebody needs to maintain public http server(s).
  - allowed file size is set in configuration. It controls the size of the largest HTTP message body that will be sent to a cache client for one request.
  - caching policy is determined by a site.
  - in general, there is no security and checksum verification.
  - useful only for frequently requested, relatively small files.

Globus Online

- GO is a Software-as-a-Service facility that provides file transfer functionality. It does a third-party transfer on your behalf:
  - performs transfers of files.
  - retries in case of failures.
  - optimizes gridftp parameters for transfer: concurrency, threading, pipelining
  - provides CLI, Rest API, WEB UI
- Usage pattern:

  Can be used to reliably pre-stage data to OSG SEs, or download files from OSG SEs back to home institution.
- Keep in mind:
  - doesn't support SRM (have to list explicitly all gridftp servers)
  - load balancing could interfere with SRM load balancing.
  - free service has strict limit for simultaneous transfers for the same user.

Remote I/O

- Usage Pattern:
- A job can seamlessly access the subset of data from the file located remotely.
- Parrot
  - allows unmodified applications access remote distributed storage transparently. It traps program system calls through ptrace.
  - can use various software drivers to communicate with different storage devices such as HTTP, GridFTP, HDFS, Chirp and others.
- Keep in mind:
  - The additional services (e.g SQUID, Chirp) are required for scalability
- Any Data Anytime Anywhere (CMS) and Federating ATLAS storage systems using XrootD
  - provides seamless access to a national-scale data access infrastructure.
    - Reliability: automatic retries and rerouting in search of files
    - Transparency: catalog lookups, redirections, reconnections are hidden from user
  - Based on XrootD
- Keep in mind:
  - Currently designed to only export CMS/ATLAS namespace.
  - Can be used only for reading data

OSG Storage Statistics

Service Name	# Instances
CE GridFTP	58
SRM EndPoint	86
SQUID	84

SRM EndPoint	DFS
BeStMan-Gateway	NFS
	HDFS
	XrootD
	Lustre
	ReddNet
dCache	

Motivation for the OSG Public Storage

Goals:

- Manage opportunistic storage provided by OSG sites.
- Help small Virtual Organizations with grid jobs data handling.

- Problems:
  - The common tools for automatic management of allocated storage do not exist.
  - Small VOs have difficulties finding appropriate storage, verifying its availability, and monitoring its utilization.
  - The involvement of a Production Manager, site administrators, and VO support personnel is required to allocate or rescind storage space

Use Cases

- SNOWMASS (Simulate hundreds of millions of high-energy proton-proton collisions, which mimic the collisions expected at future hadron colliders).
  - Need to pre-stage big files (3 – 15 GB) to selected SEs.
  - Need to download these files on a worker node during job execution.
- EIC (Electron Ion Collider at BNL: Modeling the performance and optimizing the design)
  - Pattern A: Pre-stage files (1 GB) to OSG_DATA and copy files from $OSG_DATA to a worker node during job execution.
  - Pattern B: Pre-stage a file to "SRM" SEs then copy file to all worker nodes.
- DetectorDesign (Medical Imaging, University of New Mexico: Investigating how different simulated SPECT system geometries can affect reconstructed images)
  - Upload output files to a local/remote storage from a worker node.
  - Download all the files from various SEs to user's laptop.

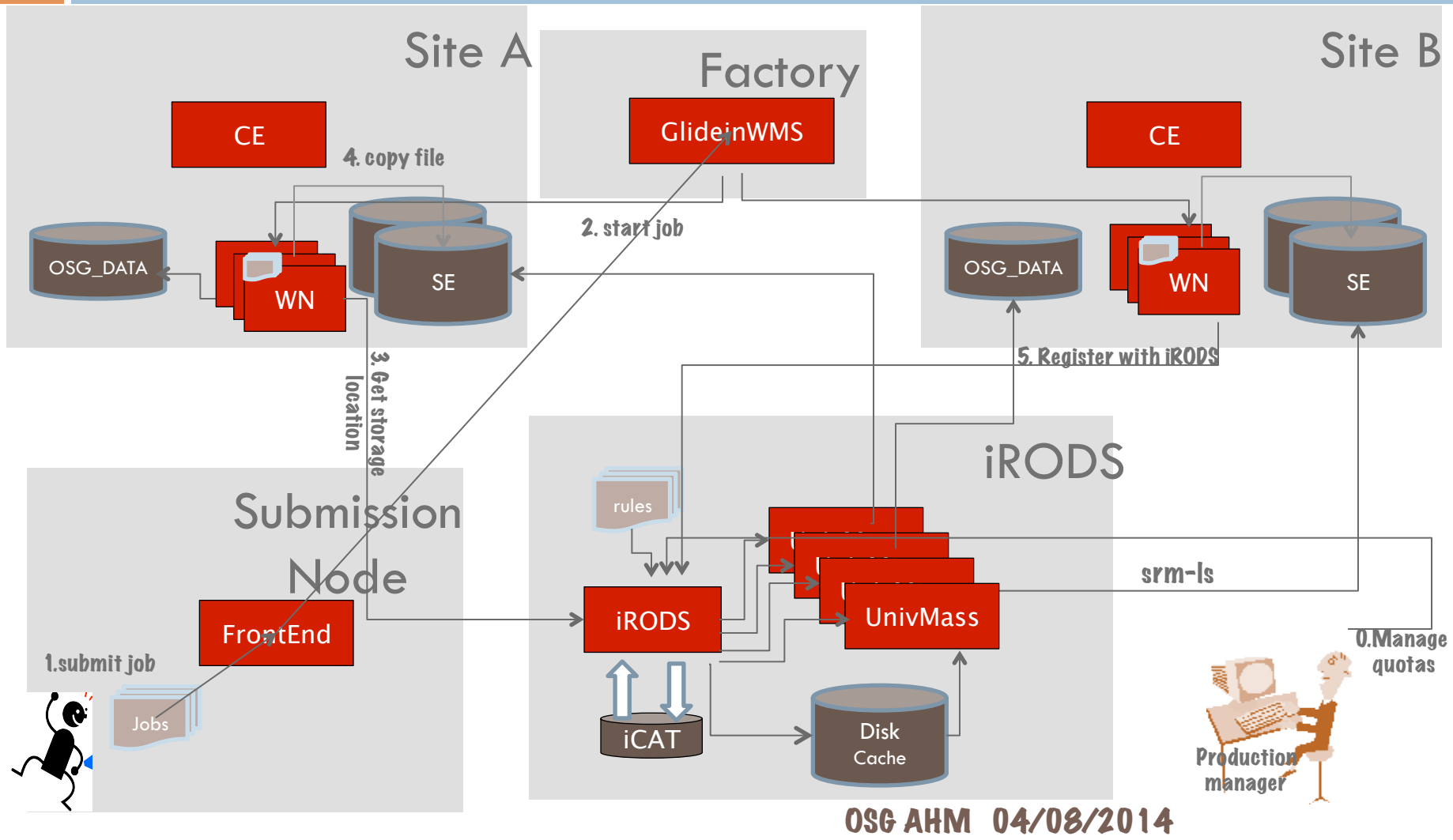
Why iRODS?

We have been looking for a storage service that:

- Enables small non-LHC VOs with "large" data requirements to use OSG sites with less effort.
- Allows the OSG Production manager to administer public storage allocation across all the participating sites.
- Imposes minimal burden on participating sites.
- We don't want to start from scratch:
- The service should be integrated with existing OSG middleware. It should use available Storage Elements.
- Has a strong community support.
- We are currently exploring the feasibility of integrating the Integrated Rule-Oriented Data System (iRODS) with the OSG SEs for providing the OSG Public Storage

OSG/iRODS Integration

(running grid job)



iRODS integration pros and cons

- Advantages:
  - Allows to register different type of remote resources
  - Allows a user to pre-stage data to OSG_DATA and SRM SEs via iRODS without dealing with sites, gathering scattered information about site resources, worrying about storage location and end path.
  - Provides a global namespace that has information about files location, size, etc.
  - Manages quota per VO/resource.
  - Doesn't impose any burden on the sites
- Disadvantages:
  - File pre-staging/download happens in two hops.
  - One cannot utilize iRODS features fully because of the architecture we are using:
    - We need to write and maintain custom scripts
    - Cannot achieve same performance

Summary

- There are multiple ways to handle data movement. The selection depends on the needs defined by your workflow.
- None of the existing methods provides a comprehensive solution for all use cases.
- There is pressing need to provide a data handling tools for small VOs. Integration with iRODS seems to provide a feasible solution for accessing and managing public storage at the OSG sites.
- The OSG Public storage doesn't address all the issues related to efficient data movement.

References and Contacts

- OSG iRODS Docs and Tutorial:

  https://twiki.grid.iu.edu/bin/view/VirtualOrganizations/IRODSOSG

- iRODS Home Page

  https://www.irods.org/index.php/IRODS:Data_Grids,_Digital_Libraries,_Persistent_Archives,_and_Real-time_Data_Systems

- iRODS-Chat google group:

  https://groups.google.com/forum/?fromgroups#!forum/iROD-Chat

- Contacts:
  - user-support@opensciencegrid.org
  - tlevshin@fnal.gov