Grid Data Management

Open Science Grid

Data Management

- Distributed community of users to access and analyze large amounts of data

Fusion community's International ITER project

- Requirement arises in both simulation and experimental science

Data Management

- Huge raw volume of data
  - Measured in terabytes, petabytes, and further …
  - Data sets can be partitioned as small number of large files or large number of small files
  - Store it long term in appropriate places (e.g., tape silos)
  - Move input to where your job is running
  - Move output data from where your job ran to where you need it (eg. your workstation, long term storage)

Data Management on the Grid

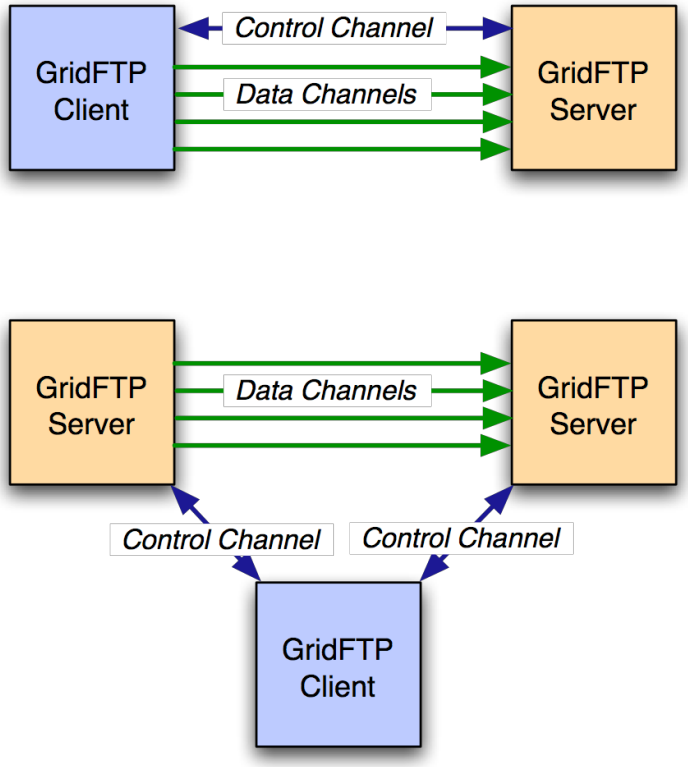
- Data sets replicated for reliability and faster access
- Files have logical names
- Service that maps logical file names to physical locations
  - Replica Location Service (RLS)
  - Where are the files I want?
- How to move data/files to where I want?
  - GridFTP

GridFTP

- High performance, secure, and reliable data transfer protocol based on the standard FTP
  - http://www.ogf.org/documents/GFD.20.pdf
- Extensions include
  - Strong authentication, encryption via Globus GSI
  - Multiple transport protocols - TCP, UDT
  - Parallel transport streams for faster transfer
  - Cluster-to-cluster or striped data movement
  - Multicasting and overlay routing
  - Support for reliable and restartable transfers
  - Server side processing, command pipelining

Basic Definitions

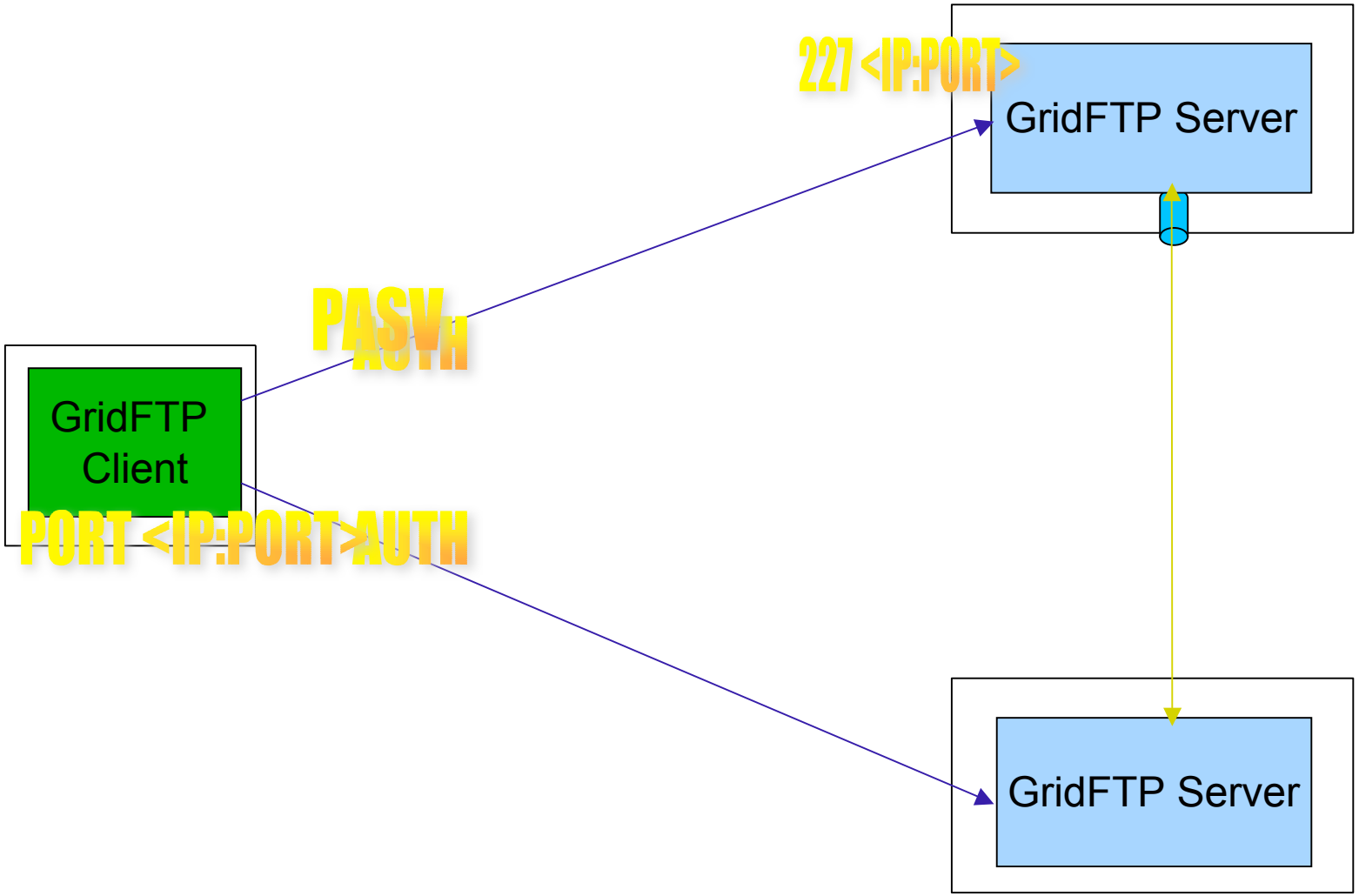
- **Control Channel**
  - TCP link over which **commands** and **responses** flow
  - Low bandwidth; encrypted and integrity protected by default
- **Data Channel**
  - Communication link(s) over which the actual **data** of interest flows
  - High Bandwidth; authenticated by default; encryption and integrity protection optional



Control Channel Establishment

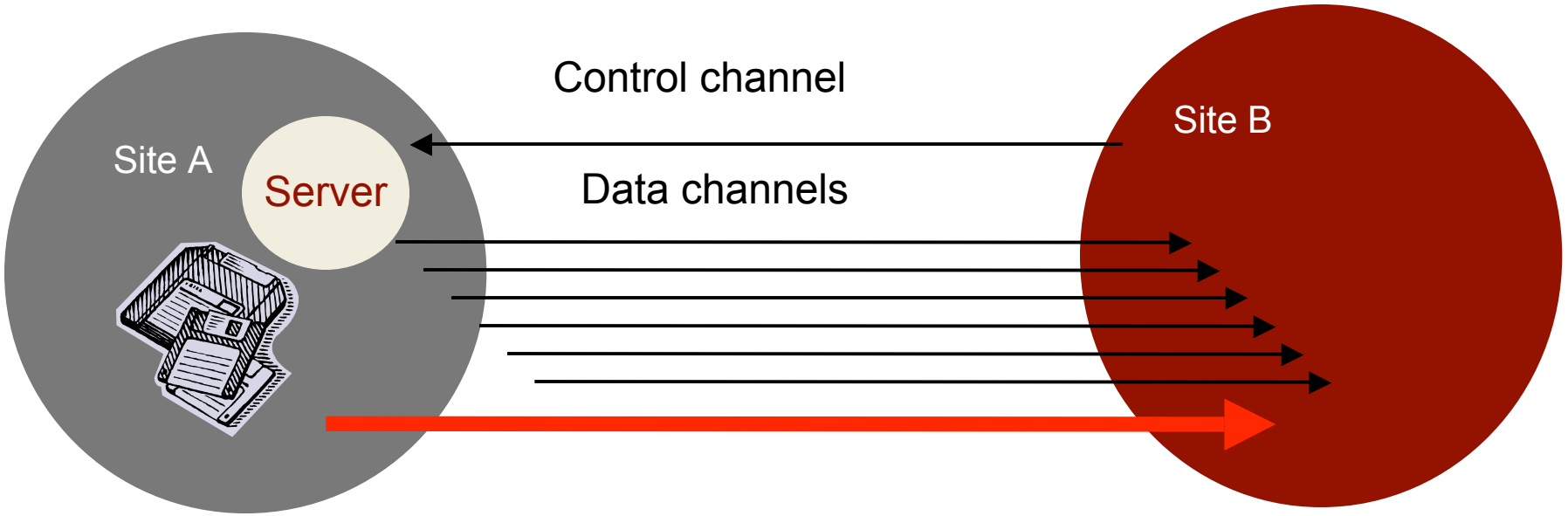
- Server listens on a well-known port (2811)
- Client form a TCP Connection to server
- Authentication
  - Anonymous
  - Clear text USER <username>/PASS <pw>
  - Base 64 encoded GSI handshake

Data Channel Establishment

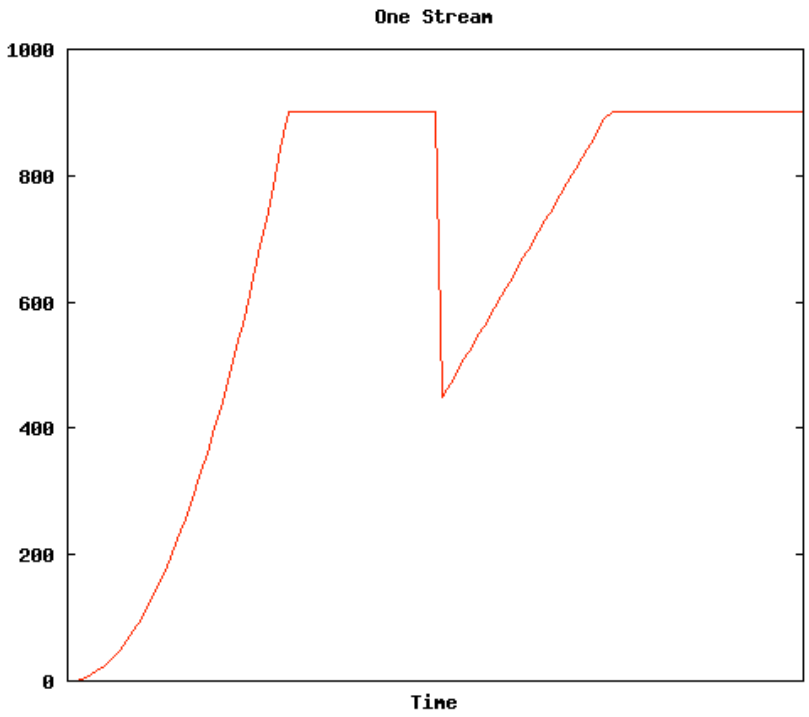


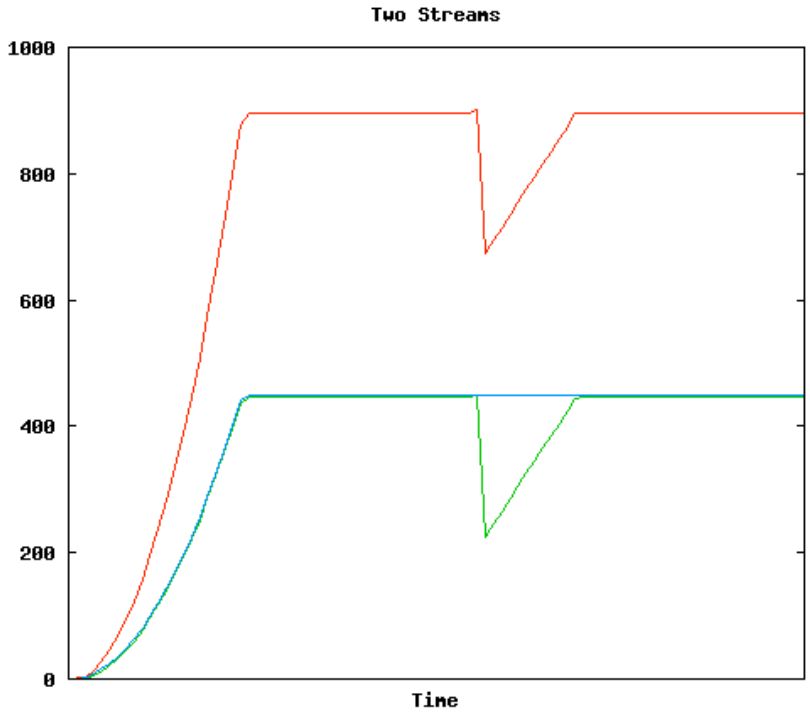
Going fast – parallel streams

- Use several data channels
- TCP - default transport protocol used by GridFTP
- TCP has limitations on high bandwidth wide area networks



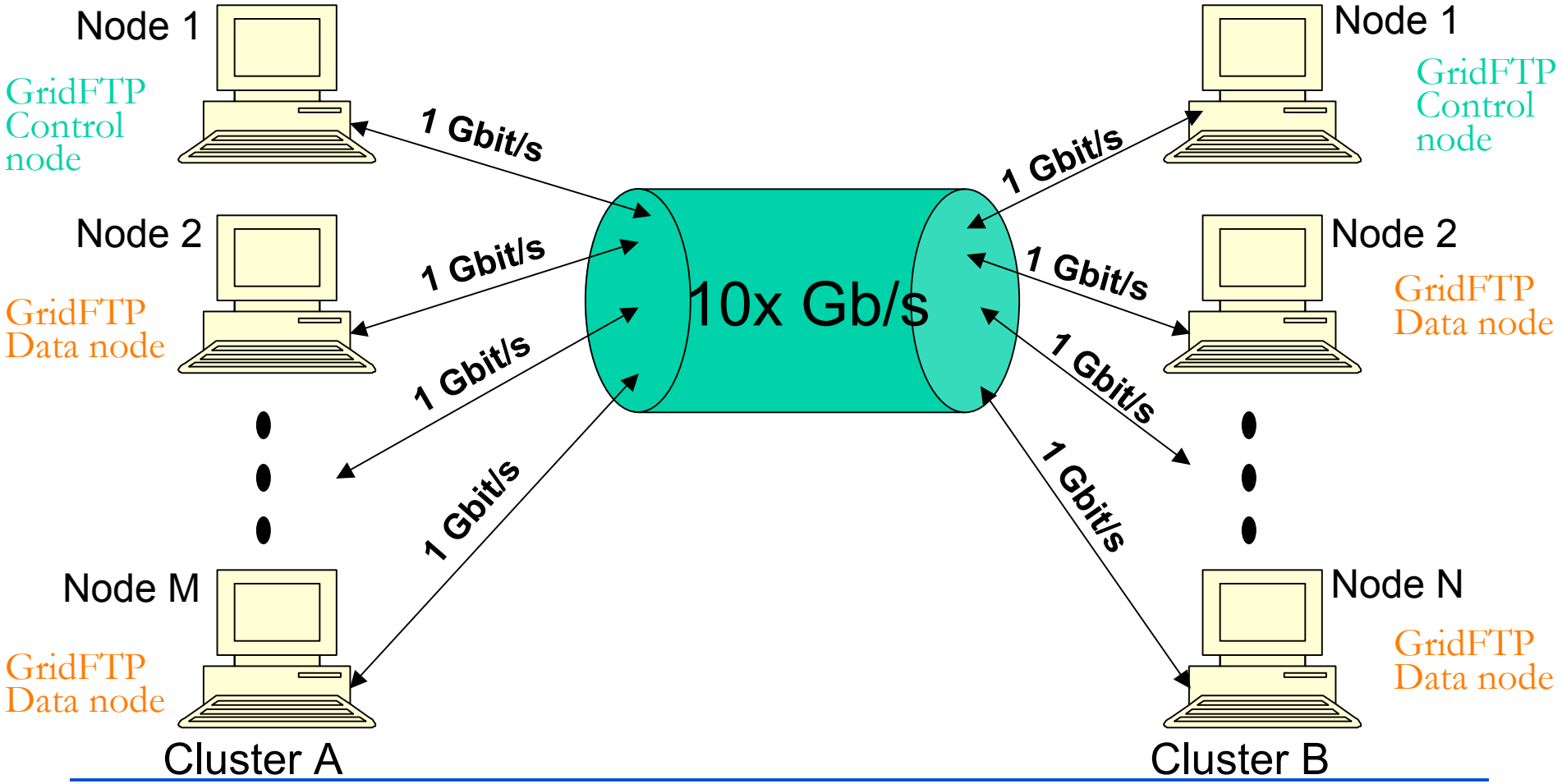
Parallel Streams





Cluster-to-cluster data transfer

GridFTP can do coordinated data transfer utilizing multiple computer nodes at source and destination



GridFTP usage

- **globus-url-copy - commonly used GridFTP client**
  - **Usage: globus-url-copy [options] srcurl dsturl**
- Conventions on URL formats:
  - `file:///home/YOURLOGIN/dataex/largefile`
    - a file called **largefile** on the local file system, in directory **/home/YOURLOGIN/dataex/**
  - `gsiftp://osg-edu.cs.wisc.edu/scratch/YOURLOGIN/`
    - a directory accessible via gsiftp on the host called `osg-edu.cs.wisc.edu` in directory **/scratch/YOURLOGIN**.

GridFTP transfers using globus-url-copy

- **globus-url-copy**
  file:///home/YOURLOGIN/dataex/myfile
  gsiftp://osg-edu.cs.wisc.edu/nfs/osgedu/YOURLOGIN/ex1

- **globus-url-copy**

gsiftp://osg-edu.cs.wisc.edu/nfs/osgedu/YOURLOGIN/ex2

gsiftp://tp-osg.ci.uchicago.edu/YOURLOGIN/ex3

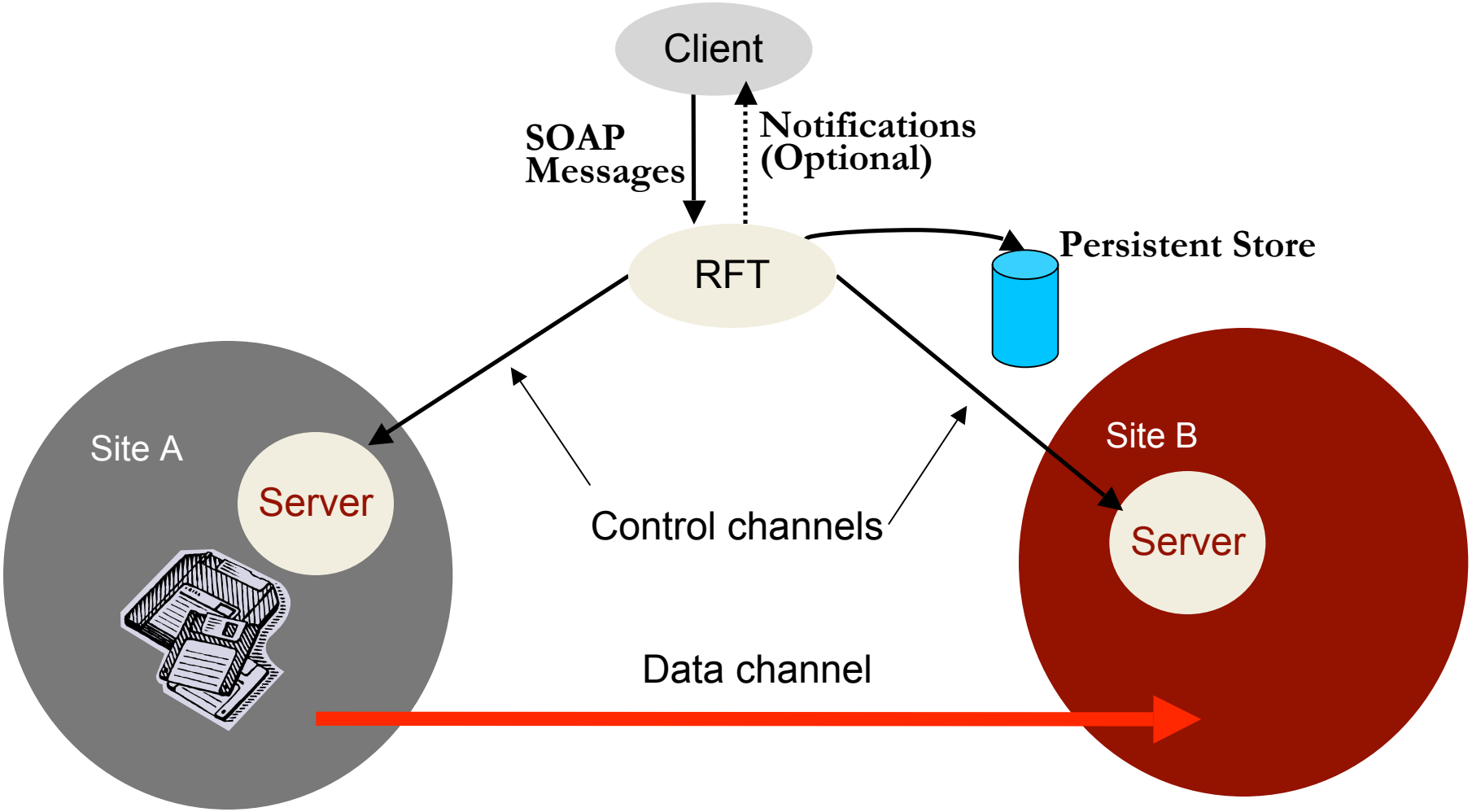
Handling failures

- GridFTP server sends restart and performance markers periodically
  - Restart markers are helpful if there is any failure
  - No need to transfer the entire file again
  - Use restart markers and transfer only the missing pieces
- GridFTP supports partial file transfers
  - Globus-url-copy has a retry option
  - Recover from transient server and network failures
  - What if the client (globus-url-copy) fails in the middle of a transfer?

RFT = Reliable file transfer

- GridFTP client that provides more reliability and fault tolerance for file transfers
  - Part of the Globus Toolkit
- RFT acts as a client to GridFTP, providing management of a large number of transfer jobs (same as Condor to GRAM)
- RFT can
  - keep track of the state of each job
  - run several transfers at once
  - deal with connection failure, network failure, failure of any of the servers involved.

RFT


Client
SOAP Messages
Notifications (Optional)
RFT
Persistent Store
Site A
Server
Control channels
Site B
Server
Data channel

RFT example

- Use the rft command with a .xfr file
- cp /soft/globus-4.0.3-r1/share/globus_wsrf_rft_client/transfer.xfr rft.xfr
- Edit rft.xfr to match your needs
- rft -h terminable.ci.uchicago.edu -f ./rft.xfr

RLS -Replica Location Service

- RLS
  - component of the data grid architecture (Globus component)
  - It provides access to mapping information from logical names to physical names of items
  - Its goal is to

    reduce access latency, improve data locality, improve robustness, scalability and performance for distributed applications

- RLS produces replica catalogs (LRCs), which represent mappings between logical and physical files scattered across the storage system.
  - For better performance, the LRC can be indexed.

RLS -Replica Location Service

- RLS maps logical filenames to physical filenames.
- Logical Filenames (LFN)
  - Names a file with interesting data in it
  - Doesn't refer to location (which host, or where in a host)
- Physical Filenames (PFN)
  - Refers to a file on some filesystem somewhere
  - Often use `gsiftp://` URLs to specify
- Two RLS catalogs:
  - Local Replica Catalog (LRC) and
  - Replica Location Index (RLI)

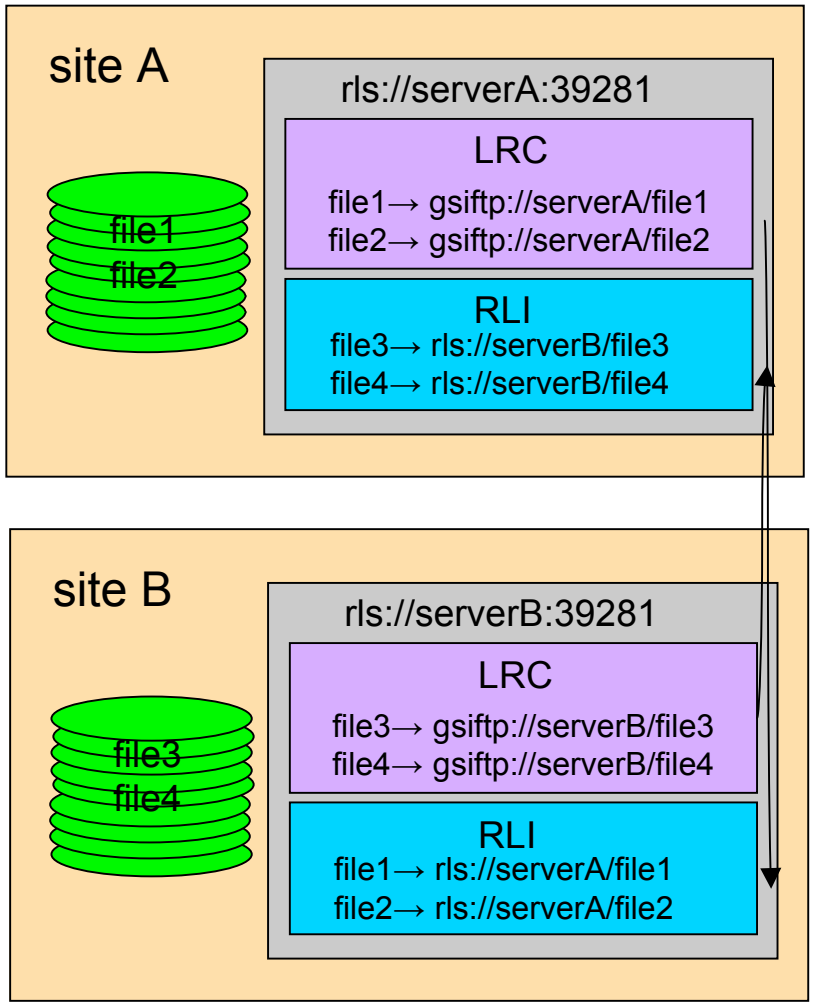
Local Replica Catalog (LRC)

- stores mappings from LFNs to PFNs.
- Interaction:
  - *Q:* Where can I get filename 'experiment_result_1'?
  - *A:* You can get it from gsiftp://gridlab1.ci.uchicago.edu/home/benc/r.txt
- Undesirable to have one of these for whole grid
  - Lots of data
  - Single point of failure

Replica Location Index (RLI)

- stores mappings from LFNs to LRCs.
- Interaction:
  - *Q:* Who can tell me about filename 'experiment_result_1'.
  - *A:* You can get more info from the LRC at gridlab1
  - (Then go to ask that LRC for more info)
- Failure of one RLI or LRC doesn't break everything
- RLI stores reduced set of information, so can cope with many more mappings

Globus RLS


site A
rls://serverA:39281
LRC
file1→ gsiftp://serverA/file1
file2→ gsiftp://serverA/file2
RLI
file3→ rls://serverB/file3
file4→ rls://serverB/file4
file1
file2
site B
rls://serverB:39281
LRC
file3→ gsiftp://serverB/file3
file4→ gsiftp://serverB/file4
RLI
file1→ rls://serverA/file1
file2→ rls://serverA/file2
file3
file4

Globus RLS

- Quick Review
  - LFN → logical filename (think of as simple filename)
  - PFN → physical filename (think of as a URL)
  - LRC → your local catalog of maps from LFNs to PFNs
    - H-R-792845521-16.gwf → gsiftp://dataserver.phys.uwm.edu/LIGO/H-R-792845521-16.gwf
  - RLI → your local catalog of maps from LFNs to LRCs
    - H-R-792845521-16.gwf → LRCs at MIT, PSU, Caltech, and UW-M
  - LRCs inform RLIs about mappings known
- Can query for files is a 2-step process: find files on your Grid by
  - querying RLI(s) to get LRC(s)
  - then query LRC(s) to get URL(s)

Globus RLS: Server Perspective

- Mappings LFNs → PFNs kept in database
  - Uses generic ODBC interface to talk to any (good) RDBM
  - MySQL, PostgreSQL, Oracle, DB2,...
  - All RDBM details hidden from administrator and user
    - well, not quite
    - RDBM may need to be "tuned" for performance
    - but one can start off knowing very little about RDBMs

Globus RLS: Server Perspective

Mappings LFNs → LRCs stored in 1 of 2 ways

- **table in database**
  - full, complete listing from LRCs that update your RLI
  - requires each LRC to send your RLI full, complete list
    - as number of LFNs in catalog grows, this becomes substantial
    - 10^8 filenames at 64 bytes per filename ~ 6 GB
- **in memory in a special hash called Bloom filter**
  - 10^8 filenames stored in as little as 256 MB
    - easy for LRC to create Bloom filter and send over network to RLIs
  - can cause RLI to lie when asked if knows about a LFN
    - only false-positives
    - tunable error rate
    - acceptable in many contexts
  - Wild carding not possible with Bloom Filters

RLS command line tools

- **globus-rls-admin**
  - administrative tasks
    - ping server
    - connect RLIs and LRCs together
- **globus-rls-cli**
  - end user tasks
    - query LRC and RLI
    - add mappings to LRC

Globus RLS: Client Perspective

Two ways for clients to interact with RLS Server

- **globus-rls-cli** simple command-line tool
  - query
  - create new mappings
- "roll your own" client by coding against API
  - Java
  - C
  - Python

Globus-rls-cli

Simple query to LRC to find a PFN for LFN

- Note more then one PFN may be returned

```
$ globus-rls-cli query lrc lfn some-file.jpg rls://dataserver:39281

some-file.jpg : file://localhost/netdata/s001/S1/R/H/714023808-
  714029599/some-file.jpg
some-file.jpg : file://medusa-
  slave001.medusa.phys.uwm.edu/data/S1/R/H/714023808-714029599/some-
  file.jpg
some-file.jpg :
  gsiftp://dataserver.phys.uwm.edu:15000/data/gsiftp_root/cluster_storage/
  data/s001/S1/R/H/714023808-714029599/some-file.jpg
```

- Server and client sane if LFN not found

```
$ globus-rls-cli query lrc lfn foo rls://dataserver
LFN doesn't exist: foo
$ echo $?
1
```

Globus-rls-cli

Wildcard searches of LRC supported

- probably a good idea to quote LFN wildcard expression

```
$ globus-rls-cli query wildcard lrc lfn H-R-7140242*-16.gwf
   rls://dataserver:39281
  H-R-714024208-16.gwf:
   gsiftp://dataserver.phys.uwm.edu:15000/data/gsiftp_root/cluster_stor
   age/data/s001/S1/R/H/714023808-714029599/H-R-714024208-16.gwf
  H-R-714024224-16.gwf:
   gsiftp://dataserver.phys.uwm.edu:15000/data/gsiftp_root/cluster_stor
   age/data/s001/S1/R/H/714023808-714029599/H-R-714024224-16.gwf
```

Globus-rls-cli

Bulk queries also supported

- obtain PFNs for more then one LFN at a time

```
$ globus-rls-cli bulk query lrc lfn H-R-714024224-16.gwf
   H-R-714024320-16.gwf rls://dataserver
   H-R-714024320-16.gwf:
    gsiftp://dataserver.phys.uwm.edu:15000/data/gsiftp_root/
    cluster_storage/data/s001/S1/R/H/714023808-714029599/H-
    R-714024320-16.gwf
   H-R-714024224-16.gwf:
    gsiftp://dataserver.phys.uwm.edu:15000/data/gsiftp_root/
    cluster_storage/data/s001/S1/R/H/714023808-714029599/H-
    R-714024224-16.gwf
```

Globus-rls-cli

Simple query to RLI to locate a LFN -> LRC mapping

- then query that LRC for the PFN

```
$ globus-rls-cli query rli lfn example-file.gwf
  rls://dataserver

  example-file.gwf: rls://ldas-cit.ligo.caltech.edu:39281

$ globus-rls-cli query lrc lfn example-file.gwf rls://ldas-
  cit.ligo.caltech.edu:39281

  example-file: gsiftp://ldas-
  cit.ligo.caltech.edu:15000/archive/S1/L0/LHO/H-R-7140/H-R-
  714024224-16.gwf
```

Globus-rls-cli

- Bulk queries to RLI also supported

```
$ globus-rls-cli bulk query rli lfn H-R-714024224-16.gwf H-R-714024320-
   16.gwf rls://dataserver
  H-R-714024320-16.gwf: rls://ldas-cit.ligo.caltech.edu:39281
  H-R-714024224-16.gwf: rls://ldas-cit.ligo.caltech.edu:39281
```

- Wildcard queries to RLI may not be supported!
  - no wildcards when using Bloom filter updates

```
$ globus-rls-cli query wildcard rli lfn "H-R-7140242*-16.gwf"
   rls://dataserver
Operation is unsupported: Wildcard searches with Bloom filters
```

Globus-rls-cli

Create new LFN → PFN mappings

- use **`create`** to create 1st mapping for a LFN

```
$ globus-rls-cli create file1 gsiftp://dataserver/file1
   rls://dataserver
```

- use **`add`** to add more mappings for a LFN

```
$ globus-rls-cli add file1 file://dataserver/file1
   rls://dataserver
```

- use **`delete`** to remove a mapping for a LFN
  - when last mapping is deleted for a LFN the LFN is also deleted
  - cannot have LFN in LRC without a mapping

```
$ globus-rls-cli delete file1 file://file1 rls://dataserver
```

Globus-rls-cli

LRC can also store attributes about LFN and PFNs

- size of LFN in bytes?
- md5 checksum for a LFN?
- ranking for a PFN or URL?
- extensible...you choose attributes to create and add
- can search catalog on the attributes
- attributes limited to
  - strings
  - integers
  - floating point (double)
  - date/time

Globus-rls-cli

- Create attribute first then add values for LFNs

```
$ globus-rls-cli attribute define md5checksum lfn string
   rls://dataserver
$ globus-rls-cli attribute add file1 md5checksum lfn
   string 42947c86b8a08f067b178d56a77b2650 rls://dataserver
```

- Then query on the attribute

```
$ globus-rls-cli attribute query file1 md5checksum lfn
   rls://dataserver
  md5checksum: string: 42947c86b8a08f067b178d56a77b2650
```

Bloom filters

- LRC-to-RLI flow can happen in two ways:
  - LRC sends list of all its LFNs (but not PFNs) to the RLI. RLI stores whole list.
    - Answer accurately: "Yes I know" / "No I don't know"
    - Expensive to move and store large list
  - Bloom filters
    - LRC generates a Bloom filter of all of its LFNs
    - Bloom filter is a bitmap that is much smaller than whole list of LFNs
    - Answers less accurately: "Maybe I know" / "No I don't know". Might end up querying LRCs unnecessarily (but we won't ever get wrong answers)
    - can't do a wildcard search

Storage and Grid

- Grid applications need to reserve and schedule
  - Compute resources
  - Network resources
  - Storage resources
- Furthermore, they need
  - Monitor progress status
  - Release resource usage when done
- For storage resources, they need
  - To put/get files into/from storage spaces
  - Unlike compute/network resources, storage resources are not available when jobs are done
  - files in spaces need to be managed as well
    - Shared, removed, or garbage collected

Motivation & Requirements (1)

- Suppose you want to run a job on your **local machine**
  - Need to allocate space
  - Need to bring all input files
  - Need to ensure correctness of files transferred
  - Need to monitor and recover from errors
  - What if files don't fit space?
    - Need to manage file streaming
  - Need to remove files to make space for more files

Motivation & Requirements (2)

- Now, suppose that the machine and storage space is a **shared resource**
  - Need to do the above for many users
  - Need to enforce quotas
  - Need to ensure fairness of space allocation and scheduling

Motivation & Requirements (3)

- Now, suppose you want to run **a job on a Grid**
  - Need to access a variety of storage systems
  - mostly remote systems, need to have access permission
  - Need to have special software to access mass storage systems

Motivation & Requirements (4)

- Now, suppose you want to **run distributed jobs on the Grid**
  - Need to allocate remote spaces
  - Need to move files to remote sites
  - Need to manage file outputs and their movement to destination sites

What is SRM?

- Storage Resource Managers (SRMs) are middleware components
  - whose function is to provide
    - dynamic space allocation
    - file management

    on shared storage resources on the Grid
  - Different **implementations** for underlying storage systems are based on the same SRM **specification**

SRMs role in grid

- SRMs role in the data grid architecture
  - Shared storage space allocation & reservation
    - important for data intensive applications
  - Get/put files from/into spaces
    - archived files on mass storage systems
  - File transfers from/to remote sites, file replication
  - Negotiate transfer protocols
  - Interoperate with other SRMs
  - File and space management with lifetime

Site URL and Transfer URL

- Provide: Site URL (SURL)
  - URL known externally – e.g. in Replica Catalogs
  - e.g. srm://ibm.cnaf.infn.it:8444/dteam/test.10193
- Get back: Transfer URL (TURL)
  - Path can be different from SURL – SRM internal mapping
  - Protocol chosen by SRM based on request protocol preference
  - e.g. gsiftp://ibm139.cnaf.infn.it:2811//gpfs/sto1/dteam/test.10193
- One SURL can have many TURLs
  - Files can be replicated in multiple storage components
  - Files may be in near-line and/or on-line storage
  - In a light-weight SRM (a single file system on disk)
    - SURL may be the same as TURL except protocol
- File sharing is possible
  - Same physical file, but many requests
  - Needs to be managed by SRM implementation

Transfer protocol negotiation

- Negotiation
  - Client provides an ordered list of preferred transfer protocols
  - SRM returns first protocol from the list it supports
  - Example
    - Client provided protocols list: bbftp, gridftp, ftp
    - SRM returns: gridftp
- Advantages
  - Easy to introduce new protocols
  - User controls which transfer protocol to use
- How it is returned?
  - The protocol of the Transfer URL (TURL)
  - Example: bbftp://dm.slac.edu//temp/run11/File678.txt

Types of storage and spaces

- Access latency
  - On-line
    - Storage where files are moved to before their use
  - Near-line
    - Requires latency before files can be accessed
- Retention quality
  - Custodial (High quality)
  - Output (Middle quality)
  - Replica (Low Quality)
- Spaces can be reserved in these storage components
  - Spaces can be reserved for a lifetime
  - Space reference handle is returned to client – space token
  - Total space of each type are subject to local SRM policy and/or VO policies
- Assignment of files to spaces
  - Files can be assigned to any space, provided that their lifetime is shorter than the remaining lifetime of the space

Managing spaces

- Default spaces
  - Files can be put into an SRM without explicit reservation
  - Default spaces are not visible to client
- Files already in the SRM can be moved to other spaces
  - By srmChangeSpaceForFiles
- Files already in the SRM can be pinned in spaces
  - By requesting specific files (srmPrepareToGet)
  - By pre-loading them into online space (srmBringOnline)
- Updating space
  - Resize for more space or release unused space
  - Extend or shorten the lifetime of a space
- Releasing files from space by a user
  - Release all files that user brought into the space whose lifetime has not expired
  - Move permanent and durable files to near-line storage if supported
  - Release space that was used by user

Space reservation

- Negotiation
  - Client asks for space: Guaranteed_C, MaxDesired
  - SRM return: Guaranteed_S <= Guaranteed_C, best effort <= MaxDesired
- Types of spaces
  - Specified during srmReserveSpace
  - Access Latency (Online, Nearline)
  - Retention Policy (Replica, Output, Custodial)
  - Subject to limits per client (SRM or VO policies)
  - Default: implementation and configuration specific
- Lifetime
  - Negotiated: Lifetime_C requested
  - SRM return: Lifetime_S <= Lifetime_C
- Reference handle
  - SRM returns space reference handle (space token)
  - Client can assign Description
  - User can use srmGetSpaceTokens to recover handles on basis of ownership

Directory management

- Usual unix semantics
  - srmLs, srmMkdir, srmMv, srmRm, srmRmdir
- A single directory for all spaces
  - No directories for each file type
  - File assignment to spaces is virtual

OSG & Data management

- OSG relies on GridFTP protocol for the raw transport of the data using Globus GridFTP in all cases except where interfaces to storage management systems (rather than file systems) dictate individual implementations.

- OSG supports the SRM interface to storage resources to enable management of space and data transfers to prevent unexpected errors due to running out of space, to prevent overload of the GridFTP services, and to provide capabilities for pre-staging, pinning and retention of the data files. OSG currently provides reference implementations of two storage systems the (BeStMan) and dCache

Credits

Bill Allcock allcock@mcs.anl.gov

Ben Clifford benc@ci.uchicago.edu

Scott Koranda skoranda@uwm.edu

Alex Sim asim@lbl.gov

Open Science Grid

Open Science Grid