GT4 GridFTP for Developers: The New GridFTP Server

Bill Allcock, ANL

January 13, 2005

What is GridFTP?

- A secure, robust, fast, efficient, standards based, widely accepted data transfer protocol
- A Protocol
  - Multiple independent implementations can interoperate
    - This works. Both the Condor Project at Uwis and Fermi Lab have home grown servers that work with ours.
    - Lots of people have developed clients independent of the Globus Project.
- We also supply a reference implementation:
  - Server
  - Client tools (globus-url-copy)
  - Development Libraries

GridFTP: The Protocol

- FTP protocol is defined by several IETF RFCs
- Start with most commonly used subset
  - Standard FTP: get/put etc., 3rd-party transfer
- Implement standard but often unused features
  - GSS binding, extended directory listing, simple restart
- Extend in various ways, while preserving interoperability with existing servers
  - Striped/parallel data channels, partial file, automatic & manual TCP buffer setting, progress monitoring, extended restart

GridFTP: The Protocol (cont)

- Existing standards
  - RFC 959: File Transfer Protocol
  - RFC 2228: FTP Security Extensions
  - RFC 2389: Feature Negotiation for the File Transfer Protocol
  - Draft: FTP Extensions
  - GridFTP: Protocol Extensions to FTP for the Grid
    - Grid Forum Recommendation
    - GFD.20
    - http://www.ggf.org/documents/GWD-R/GFD-R.020.pdf

wuftpd based GridFTP

Functionality prior to GT3.2

- Security
- Reliability / Restart
- Parallel Streams
- Third Party Transfers
- Manual TCP Buffer Size
- Partial File Transfer
- Large File Support
- Data Channel Caching
- Integrated Instrumentation
- De facto standard on the Grid

New Functionality in 3.2

- Server Improvements
  - Structured File Info
    - MLST, MLSD
  - checksum support
  - chmod support (client)
- globus-url-copy changes
  - File globbing support
  - Recursive dir moves
  - RFC 1738 support
  - Control of restart
  - Control of DC security

New GT4 GridFTP Implementation

- NOT web services based
- NOT based on wuftpd
- 100% Globus code. No licensing issues.
- Absolutely no protocol change. New server should work with old servers and custom client code.
- Extremely modular to allow integration with a variety of data sources (files, mass stores, etc.)
- Striping support is present.
- Has IPV6 support included (EPRT, EPSV), but we have limited environment for testing.
- Based on XIO
- wuftpd specific functionality, such as virtual domains, will NOT be present

Extensible IO (XIO) system

- Provides a framework that implements a Read/Write/Open/Close Abstraction
- Drivers are written that implement the functionality (file, TCP, UDP, GSI, etc.)
- Different functionality is achieved by building protocol stacks
- GridFTP drivers will allow 3rd party applications to easily access files stored under a GridFTP server
- Other drivers could be written to allow access to other data stores.
- Changing drivers requires minimal change to the application code.

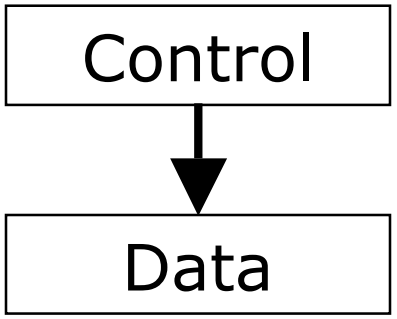
New Server Architecture

- GridFTP (and normal FTP) use (at least) two separate socket connections:
  - A control channel for carrying the commands and responses
  - A Data Channel for actually moving the data
- Control Channel and Data Channel can be (optionally) completely separate processes.
- A single Control Channel can have multiple data channels behind it.
  - This is how a striped server works.
  - In the future we would like to have a load balancing proxy server work with this.

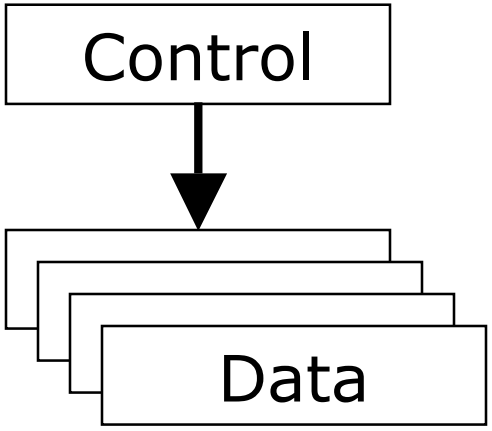
Possible Configurations

Typical Installation

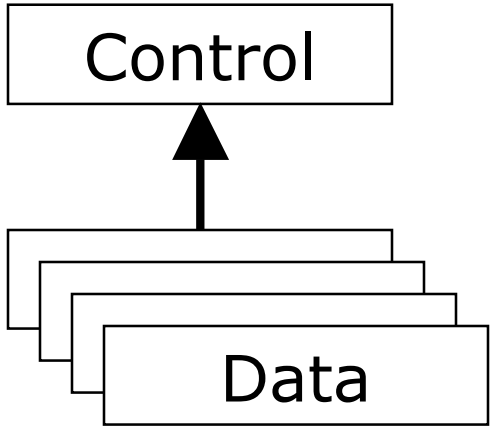
Separate Processes



Striped Server



Striped Server (future)



New Server Architecture

- Data Transport Process (Data Channel) is architecturally, 3 distinct pieces:
  - The protocol handler. This part talks to the network and understands the data channel protocol
  - The Data Storage Interface (DSI). A well defined API that may be re-implemented to access things other than POSIX filesystems
  - ERET/ESTO processing. Ability to manipulate the data prior to transmission.
    - currently handled via the DSI
    - In V4.2 we to support XIO drivers as modules and chaining
- Working with several groups to on custom DSIs
  - LANL / IBM for HPSS
  - UWis / Condor for NeST
  - SDSC for SRB

The Data Storage Interface (DSI)

- Unoriginally enough, it provides an interface to data storage systems.
- Typically, this data storage system is a file system accessible via the standard POSIX API, and we provide a driver for that purpose.
- However, there are many other storage systems that it might be useful to access data from, for instance HPSS, SRB, a database, non-standard file systems, etc..

The Data Storage Interface (DSI)

- Conceptually, the DSI is very simple.
- There are a few required functions (init, destroy)
- Most of the interface is optional, and you can only implement what is needed for your particular application.
- There are a set of API functions provided that allow the DSI to interact with the server itself.
- Note that the DSI could be given significant functionality, such as caching, proxy, backend allocation, etc..

Current Development Status

- GT3.9.4 has a very solid alpha. This code base has been in use for over a year.
- The data channel code, which was the code we added to wuftpd, was re-used and so has been running for several years.
- Initial bandwidth testing is outstanding.
- Stability testing shows non-striped is rock solid
- Striped has a memory leak that we are hunting
- http://dc-master.isi.edu/mrtg/ned.html

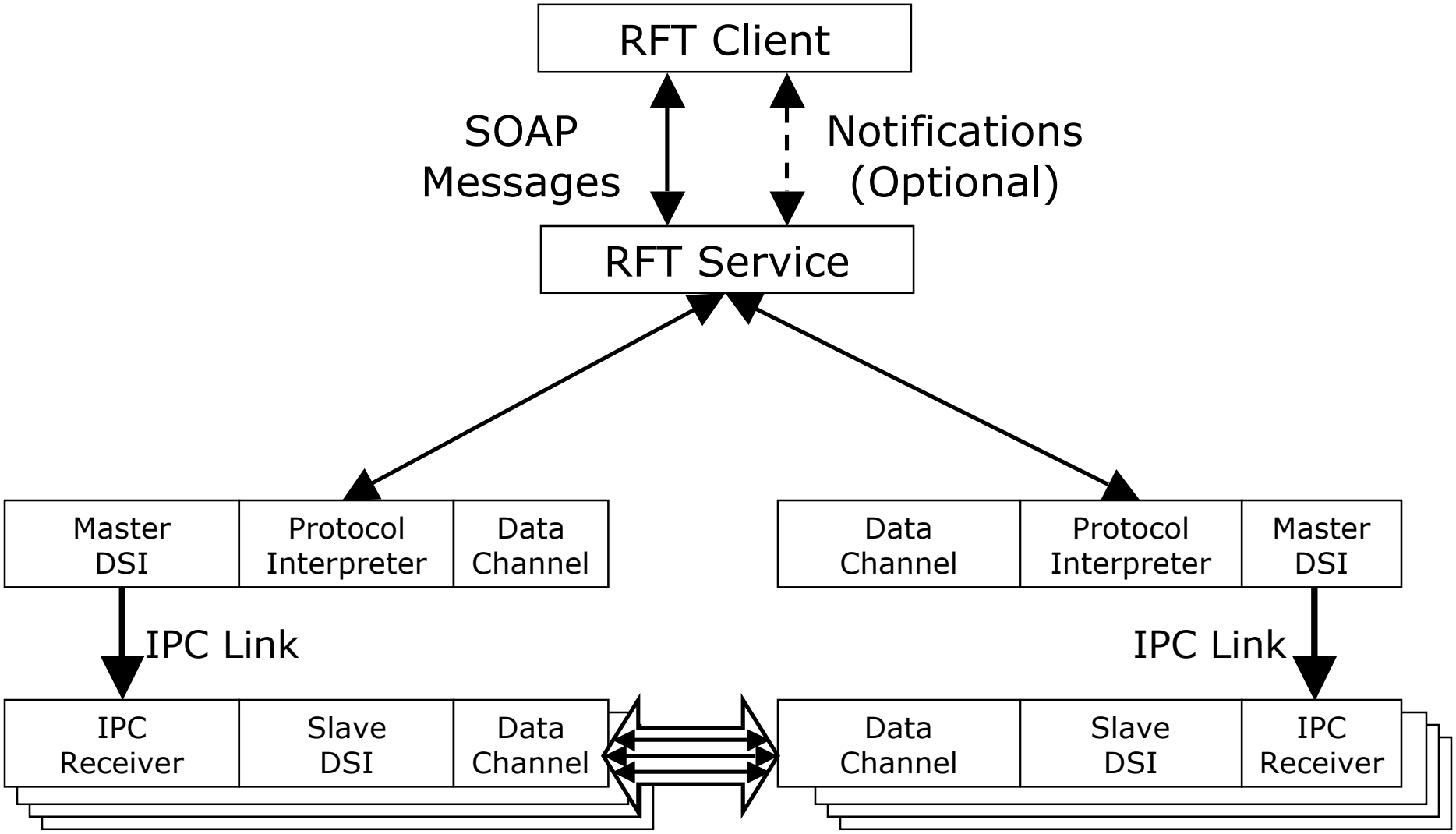
Status continued

- Stability tests to date have been for a single long running transfer
- We are working on sustained load and "job storm" tests
- A usable response in the face of overload is a key goal.
- Completed an external security architecture review
  - Likely to make changes to the "recommended configuration"
  - This is a deployment issue, not a code issue.
- Planning an external code review.

Deployment Scenario under Consideration

- All deployments are striped, i.e. separate processed for control and data channel.
- Control channel runs as a user who can only read and execute executable, config, etc. It can write delegated credentials.
- Data channel is a root setuid process
  - Outside user never connects to it.
  - If anything other than a valid authentication occurs it drops the connection
  - It can be locked down to only accept connections from the control channel machine IP
  - First action after successful authentication is setuid

Third Party Transfer



Striped Server

- Multiple nodes work together and act as a single GridFTP server
- An underlying parallel file system allows all nodes to see the same file system and must deliver good performance (usually the limiting factor in transfer speed)
  - I.e., NFS does not cut it
- Each node then moves (reads or writes) only the pieces of the file that it is responsible for.
- This allows multiple levels of parallelism, CPU, bus, NIC, disk, etc.
  - Critical if you want to achieve better than 1 Gbs without breaking the bank

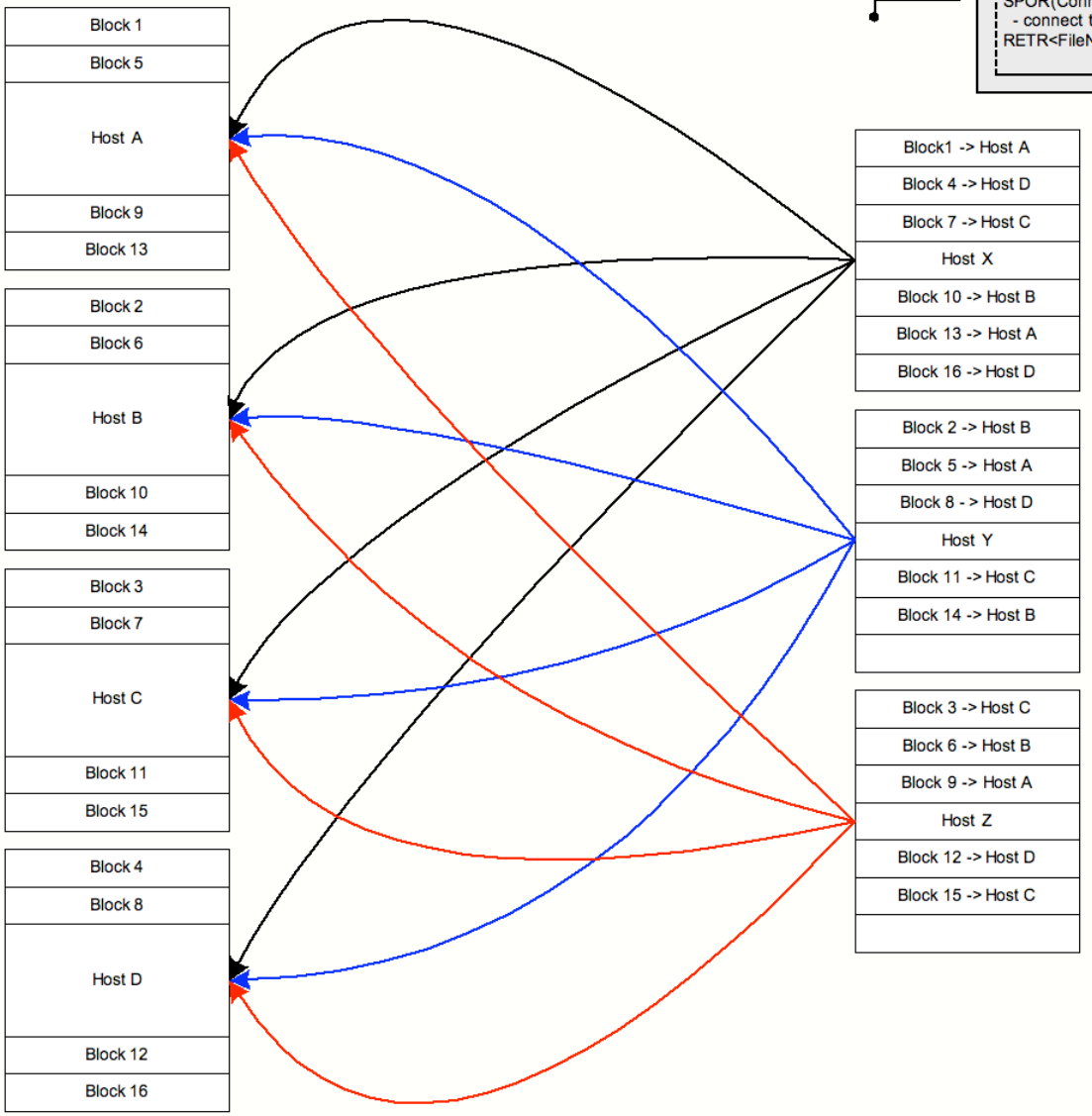
the globus alliance
www.globus.org

18-Nov-03

GridFTP Striped Transfer

MODE E
SPAS (Listen)
- returns list of host:port pairs
STOR<FileName>

MODE E
SPOR(Connect)
- connect to the host-port pairs
RETR<FileName>

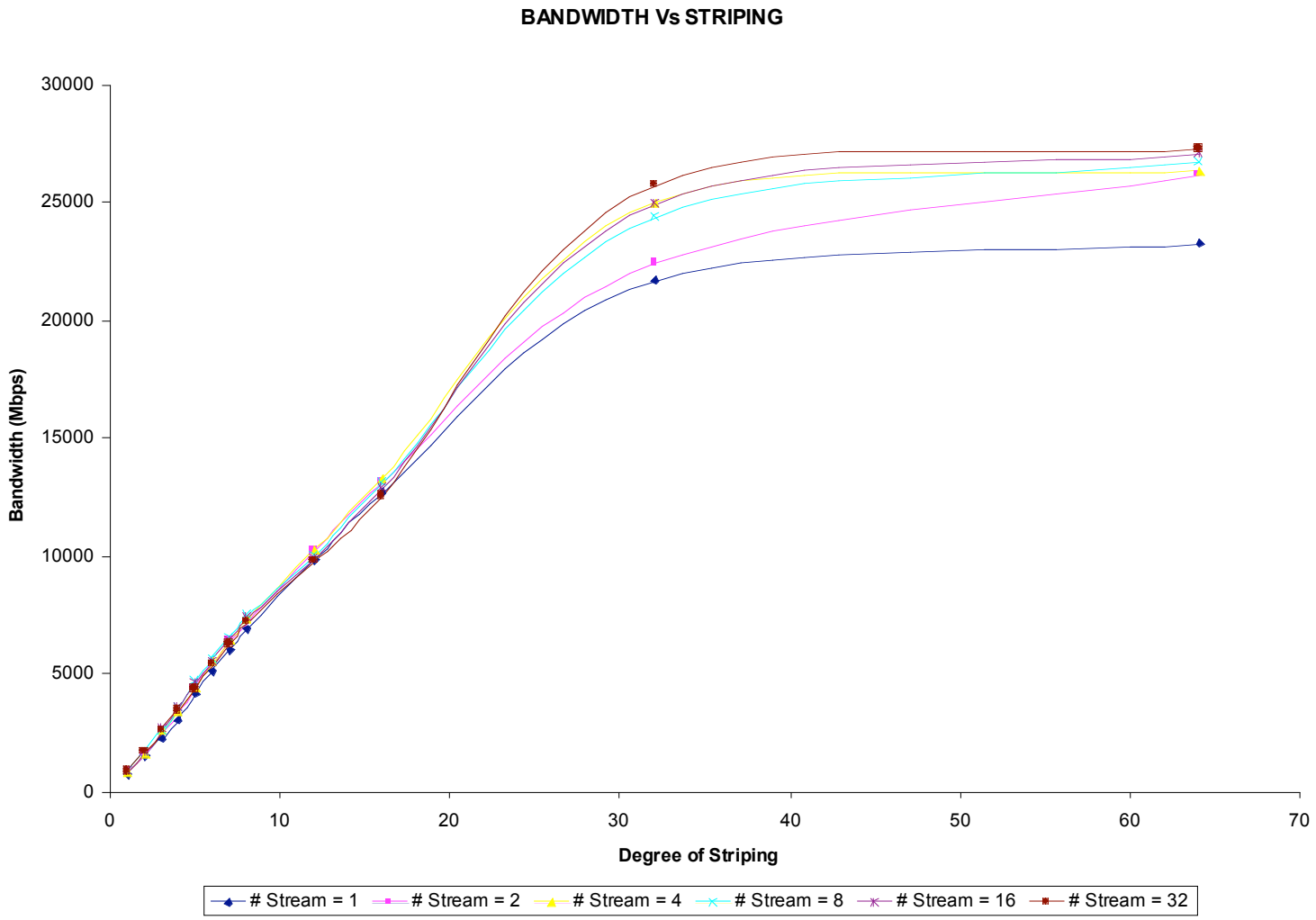


TeraGrid Striping results

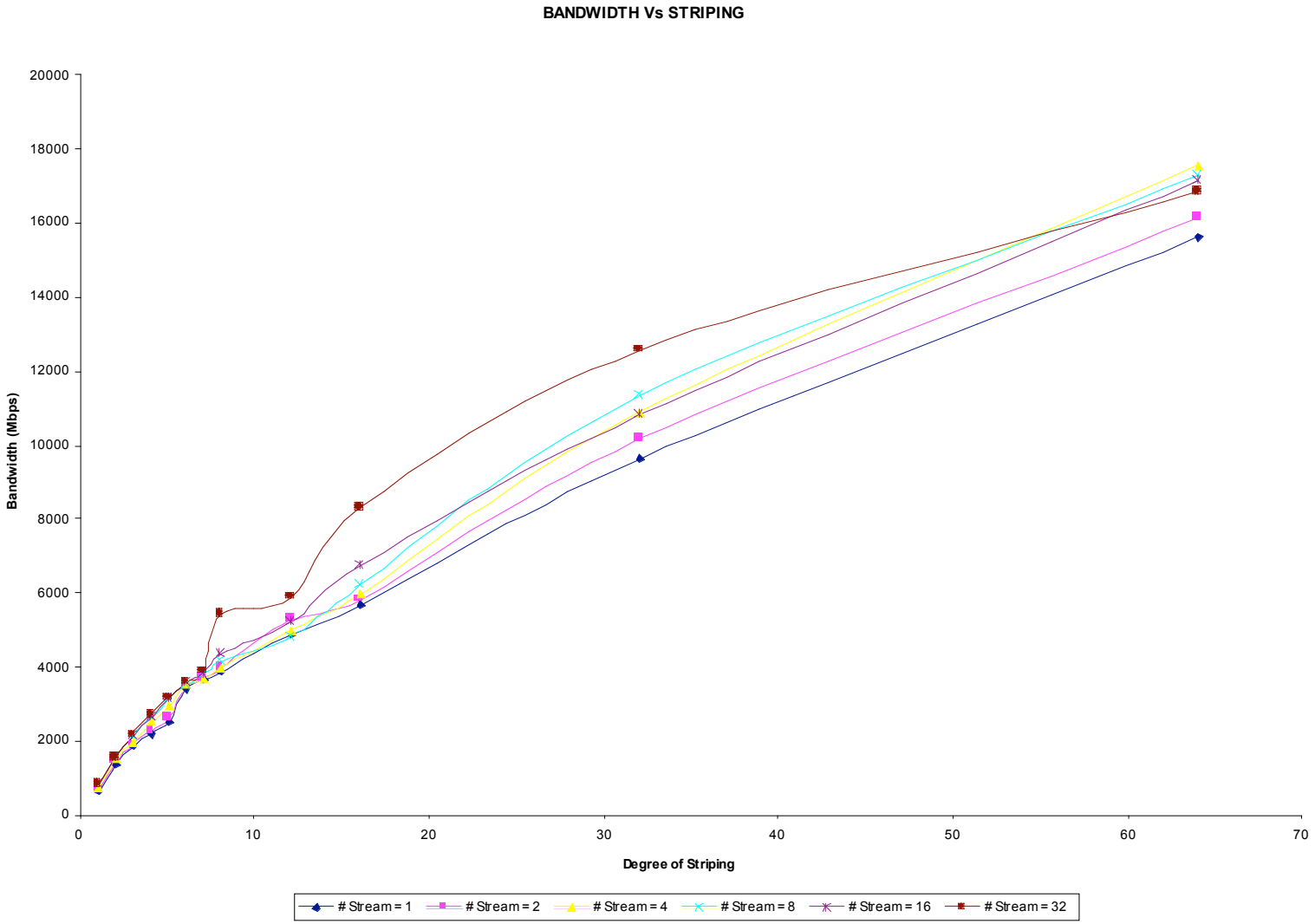
- Ran varying number of stripes
- Ran both memory to memory and disk to disk.
- Memory to Memory gave extremely high linear scalability (slope near 1).
- We achieved 27 Gbs on a 30 Gbs link (90% utilization) with 32 nodes.
- Disk to disk we were limited by the storage system, but still achieved 17.5 Gbs

the globus alliance
www.globus.org

Memory to Memory Striping Performance



Disk to Disk Striping Performance



GridFTP: Caveats

- Protocol requires that the sending side do the TCP connect (possible Firewall issues)
- Client / Server
  - Currently, no simple encapsulation of the server side functionality (need to know protocol), therefore Peer to Peer type apps VERY difficult
    - A library with this encapsulation is on our radar, but no timeframe.
  - Generally needs a pre-installed server
    - Looking at a "dynamically installable" server

So, what about Web Services…

- Web Services access to data movement is available via the Reliable File Transfer Service.
  - WSRF, WS-addressing, WSN, WSI compliant
  - It is reliable. State is persisted in a database. It will retry and either succeed or meet what you defined as ultimate failure criteria.
  - It is a service. Similar to a job scheduler. You can submit your data transfer job and go away.

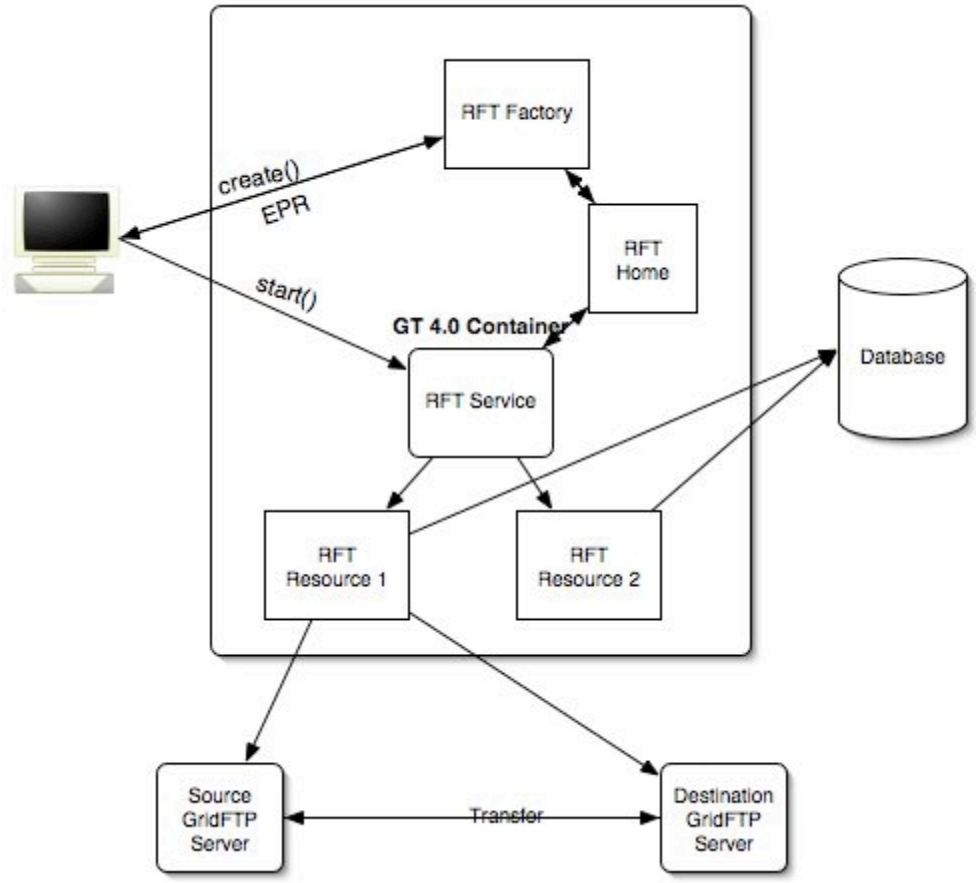
Public Interfaces

- http://www-unix.globus.org/toolkit/docs/development/4.0-drafts/data/rft/RFT_Public_Interfaces.html
- The above URL lists the methods and resource properties.
- It also provides an overview of how the command line client works.
  - Our client is relatively simple
  - No GUI client is provided by Globus

Important Points

- Container wide database connection pool
  - Can either wait for ever or throw an exception
- Container wide RFT thread max
  - Total number of transfer threads limited
- One resource per request
  - Request has a thread pool equal to concurrency
- Resource Lifetime is independent of transfers
  - Needs to exceed transfer lifetime if you want state around to check status.
- URL expansion can be time consuming
  - Currently does not start transfers until fully expanded

RFT Architecture



WS-GRAM Approach