GridFTP Lite Project Summary

The GridFTP protocol provides for the secure, robust, fast, efficient, standards based transport of (especially bulk) data. The Globus Alliance provides the most commonly used implementation of that protocol. This implementation is used in projects all over the world, in many cases in production environments. Given its ubiquity and its heavy use within application research communities that the network researchers are trying to support, it makes a good candidate as a "test application" to determine the impact of the network researcher's work on those communities. However, many of its production features make it difficult to use in non-production scenarios, particularly the security. With this in mind, we set out to make the Globus implementation more suitable for network researchers needs. We decided to approach this problem along three primary avenues: Ease of use and installation, extensibility to ease its use in non-standard network situations, and finally, the addition of a few features specifically requested by network researchers.

Ease of Use and Installation

Early versions of the Globus Toolkit were extremely difficult to install. Its installation used non-standard packaging technology (the Grid Packaging Toolkit or GPT), and the installation was monolithic, you could not pick and choose the components you wanted to build / install. There was already work started to hide the GPT. We piggybacked on this work and added the ability to build specific components. Specifically, the GPT commands are now wrapped in a make file so that the installation now follows the "configure, make, make install" paradigm in common use in *nix systems. There are multiple make targets available for the components. The ones of interest for this project are:

gridftp: builds the client, server, libraries, and various useful associated utilities
globus-gridftp-server: builds just the server
globus-data-management-client: builds just the clients
globus-data-management-sdk: builds just the development libraries

Extensibility

We had already planned for extensibility elements in our new server. This work provided us funding and added a few additional requirements. This extensibility is provided through two primary avenues: the use of Globus XIO to abstract an IO abstraction layer, and the Data Storage Interface which abstracts the entire storage system.

Globus XIO

All IO in our new server implementation is done via XIO. I.e., you will not see calls to fopen, fwrite, or socket calls. You will only see xio_open, xio_write. XIO uses the concept of drivers and a driver stack. You choose the functionality you want to have in your IO, and build a driver stack appropriately. For instance, to have GSI authenticated TCP communication, you would push a TCP driver and a GSI driver onto the stack. Switch TCP for UDT, and now you have a GSI authenticated UDT connection. Examples of drivers that might make sense in the DOE network research community could include protocol drivers (UDT, Tsunami, RBUDP, etc), a DSCP driver might allow trivial integration of Lambda station into GridFTP, a driver for either setting up a dynamic optical path, or claiming a reservation on such a path, etc..

The Data Storage Interface (DSI)

Sometimes, just abstracting the IO is not enough. You might want to interact with something that can source or sink data, but is not a normal disk file system. While you could (and should) abstract the IO to these systems via XIO, they also demand additional functionality such as file information, directory listings, directory creation, etc.. The DSI creates an abstraction for these additional functions. As an example, a DSI could be written that could interact with the Internet Backplane Protocol (IBP) and once written, a GridFTP client (any client, not just ours) could access data stored in IBP and not know or care that it was stored there.

Additional Features

There were a few features added that were specifically requested by members of the DOE network research community. These included:

anonymous access: It is now possible to allow open access. In situations where the testing environment is not accessible by the general internet, this avoids the overhead of having to deal with X.509 certificates.

password access: This is standard FTP cleartext password access. It uses the standard /etc/passwd file format, but you should not use that file and you should not use the same password since the passwords are sent in clear text.

memory-to-memory transfers: Network researchers often do not want disk IO to factor into their tests. You may now transfer between /dev/zero and /dev/null.

time limited transfers: primarily added for use with memory-to-memory transfers, this causes the transfer to run for a specified period of time and then end (similar to iperf). This is required for memory-to-memory transfers since they have no end-of-file and will run forever unless explicitly killed.

Future Plans

Funding for this project ended in FY2005. However, we have identified the following issues that we believe are generally useful and will work on them when and where possible.

Documentation: Globus XIO is fairly well documented, but could use some improvement. The Data Storage Interface is virtually undocumented and this needs to be rectified.

Ease DSI development: We intend to separate the actual movement of data from the other DSI functionality since in most cases they can use our data movement code for fast, efficient, IO.

Workshop: We would like to sponsor a workshop that would include detailed development tutorials and “bring your code” sessions as well as a general discussion of research and issues surrounding end-to-end application performance over the network, particularly for files.

Add additional extensibility points: Our goal is to have a pre and post callout for most if not all events that occur within the server. This would allow extreme customization of the server operation.

More security options: We will investigate the use of ssh keys (though there are possible licensing as well as technical issues) and the introduction of a trivial testbed setup utility that would allow the security of GSI without the associated headaches.