

GT 5.0.2 GridFTP : System Administrator's Guide

GT 5.0.2 GridFTP : System Administrator's Guide

Introduction

This guide contains advanced configuration information for system administrators working with GridFTP. It provides references to information on procedures typically performed by system administrators, including installation, configuring, deploying, and testing the installation. This guide should help you configure and run the GridFTP *server* in some standard configurations.



Important

This information is in addition to the basic Globus Toolkit prerequisite, overview, installation, security configuration instructions in the [Installing GT 5.0.2](#). Read through this guide before continuing!

Table of Contents

1. Building and Installing	1
1. Building only GridFTP and Utilities	1
2. More build options	1
2. Configuring GridFTP	4
1. GridFTP server configuration overview	4
2. Typical configuration	4
3. Firewall requirements	5
4. Configuring Security for GridFTP	6
5. globus-gridftp-server quickstart	10
3. Key Admin Settings and Tuning Recommendations	12
1. Concurrent Instances	12
2. Disk Block Size	13
3. System TCP Buffer Settings	13
4. Data Interface	14
4. Advanced Configuration	15
1. Configuring GridFTP to use UDT instead of TCP	15
2. Configuring GridFTP to enable Netlogger's bottleneck detection	16
3. Separation of Processes (Split Process)	16
4. Configuring GridFTP for Cluster-to-Cluster (or Striped) data movement	16
5. Combining Split Process and Striped configuration	17
6. Running with GFork Master Plugin	17
7. Configuring multicasting/broadcasting	17
8. Accessing data from other data interfaces	18
9. GridFTP Where There Is FTP (GWTFTP)	19
5. Testing	20
6. Debugging	21
1. Logging	21
7. Troubleshooting	22
1. Error Codes in GridFTP	23
2. Establish control channel connection	25
3. Try running globus-url-copy	26
4. If your server starts... ..	26
5. High latency for GridFTP server connections	27
8. Usage statistics collection by the Globus Alliance	28
1. GridFTP-specific usage statistics	28
A. GridFTP Admin Tool	29
globus-gridftp-server	30
B. Setting up SRB for use with GridFTP	42
1. Introduction	42
2. Architecture	42
3. Software	42
4. Building	42
5. Administration	43
6. Running	44
7. See Also	44
C. Running the Globus GridFTP Server With GFork	45
1. Introduction	45
2. Use Cases	45
3. Configuration	45
4. GFork and Striped Servers	47
5. GFork with Memory Management	48

Glossary	50
Index	52

List of Tables

7.1. GridFTP Errors	24
---------------------------	----

Chapter 1. Building and Installing

GridFTP is built and installed as part of a default GT 5.0.2 installation. For basic installation instructions, see [Installing GT 5.0.2](#). No extra installation steps are required for this component.

1. Building only GridFTP and Utilities

If you wish to install GridFTP without installing the rest of the Globus Toolkit, refer to [the Installing GT 5.0.2 section of the System Administrator's Guide](#). Perform steps 1-5, as written with the following exception: instead of running "make" as directed in step 4,

Run:

```
globus$ make gridftp
```

If you wish to have a log file of the build, use **tee**:

```
globus$ make gridftp 2>&1 | tee build.log
```

The syntax above assumes a Bourne shell. If you are using another shell, redirect stderr to stdout and then pipe it to **tee**.

For instructions on building the server alone, client alone or building a static binary of server/client, please see [More build options](#).

2. More build options

2.1. Building only the GridFTP server

If you wish to install only the GridFTP server, refer to [the Installing GT 5.0.2 section of the System Administrator's Guide](#). Perform steps 1-5, as written with the following exception: instead of running "make" as directed in step 4,

Run:

```
globus$ make gpt globus_gridftp_server
```

If you wish to have a log file of the build, use **tee**:

```
globus$ make gpt globus_gridftp_server 2>&1 | tee build.log
```

The syntax above assumes a Bourne shell. If you are using another shell, redirect stderr to stdout and then pipe it to **tee**.

2.2. Building only the GridFTP client

If you wish to install only the GridFTP client, refer to [the Installing GT 5.0.2 section of the System Administrator's Guide](#). Perform steps 1-5, as written with the following exception: instead of running "make" as directed in step 4,

Run:

```
globus$ make globus-data-management-client
```

If you wish to have a log file of the build, use **tee**:

```
globus$ make globus-data-management-client 2>&1 | tee build.log
```

The syntax above assumes a Bourne shell. If you are using another shell, redirect stderr to stdout and then pipe it to **tee**.

2.3. Building only the GridFTP SDK

If you wish to install only the GridFTP SDK, refer to the [the Installing GT 5.0.2 section of the System Administrator's Guide](#). Perform steps 1-5, as written with the following exception: instead of running "make" as directed in step 4,

Run:

```
globus$ make globus-data-management-sdk
```

If you wish to have a log file of the build, use **tee**:

```
globus$ make globus-data-management-sdk 2>&1 | tee build.log
```

The syntax above assumes a Bourne shell. If you are using another shell, redirect stderr to stdout and then pipe it to **tee**.

2.4. Building a combination of GridFTP elements

If you wish to build a combination of GridFTP elements, refer to the [the Installing GT 5.0.2 section of the System Administrator's Guide](#). Perform steps 1-5, as written with the following exception: instead of running "make" as directed in step 4,

Run:

```
globus$ make [any combination of the above commands, each separated by a space]
```

For example, if you just want to install the GridFTP server and client, the command would be:

```
globus$ make gpt globus_gridftp_server globus-data-management-client
```

If you wish to have a log file of the build, use **tee**:

```
globus$ make [any combination of the above commands, each separated by a space] 2>&1 | tee
```

The syntax above assumes a Bourne shell. If you are using another shell, redirect stderr to stdout and then pipe it to **tee**.

2.5. Building and Installing a static GridFTP server

If you wish to build and install a statically linked set of GridFTP binaries, refer to [the Installing GT 5.0.2 section of the System Administrator's Guide](#). Follow steps 1-2 as written. In step 3, however, you should

Run:

```
globus$ export GLOBUS_LOCATION=/usr/local/globus-4.0.0
        globus$ ./configure --prefix=$GLOBUS_LOCATION --with-buildopts="--static"
globus$ make gpt globus_gridftp_server
```

If you wish to have a log file of the build, use **tee**:

```
globus$ make gpt globus_gridftp_server 2>&1 | tee build.log
```

The syntax above assumes a Bourne shell. If you are using another shell, redirect stderr to stdout and then pipe it to **tee**.

Then continue on with steps 4 and 5 as written.

2.6. Switching between threaded and non-threaded flavors

In GT5, the default flavor of the GridFTP server and the client 'globus-url-copy' is non-threaded.

2.6.1. If you are building from the source installer

If you are building from a source installer, both threaded and non-threaded flavors of server and client will be built by default. By default, the server executable in `$GLOBUS_LOCATION/sbin` and the client executable in `$GLOBUS_LOCATION/bin` are non-threaded.

If you built from the source installer, you will find non-default flavors of both the server and client binaries in `$GLOBUS_LOCATION/sbin/[flavor]/shared` and `$GLOBUS_LOCATION/bin/[flavor]/shared`. To use those flavors by default, simply copy those binaries to `$GLOBUS_LOCATION/sbin` or `$GLOBUS_LOCATION/bin`.

2.6.1.1. Examples

Changing **globus-url-copy** from non-threaded to threaded:

```
$ cp $GLOBUS_LOCATION/bin/gcc32dbgpthr/shared/globus-url-copy $GLOBUS_LOCATION/bin
```

Changing **globus-gridftp-server** from non-threaded to threaded:

```
$ cp $GLOBUS_LOCATION/sbin/gcc32dbgpthr/shared/globus-gridftp-server $GLOBUS_LOCATION/sbin
```

Checking whether a binary is a threaded flavor or not with the tool **ldd** (no output means non-threaded):

```
$ ldd $GLOBUS_LOCATION/sbin/gcc32dbgpthr/globus-gridftp-server |grep thread
      libpthread.so.0 => /lib/libpthread.so.0 (0x00cab000)
```

2.6.2. If you are building from the binary installer

If you are using a binary installer, it will only have non-threaded GridFTP server and non-threaded client. If you want threaded flavor of server/client, use source installer.

Chapter 2. Configuring GridFTP

1. GridFTP server configuration overview

The configuration interface for GridFTP is the admin tool, [globus-gridftp-server](#), which can be used with a configuration file and/or run-time options.



Note

Command line options and configuration file options may both be used, but the command line *overrides* the config file.

The configuration file for the GridFTP *server* is read from the following locations, in the given order. Only the first file found will be loaded:

- Path specified with the `-c <configfile>` command line option.
- `$GLOBUS_LOCATION/etc/gridftp.conf`
- `/etc/grid-security/gridftp.conf`

Options are one per line, with the format:

`<option> <value>`

If the value contains spaces, they should be enclosed in double-quotes ("). Flags or boolean options should only have a value of 0 or 1. Blank lines and lines beginning with # are ignored.

For example:

```
port 5000
allow_anonymous 1
anonymous_user bob
banner "Welcome!"
```

For complete command documentation including all options, see [globus-gridftp-server\(1\)](#).

This page includes information about general configuration of the GridFTP server. Security options are discussed [here](#), and more advanced configuration is described [here](#).

2. Typical configuration

The following describes a typical GridFTP configuration of the front end (control channel) and back end (data channels). For other alternatives that provide greater levels of security, see [Advanced Configuration](#).

By default, the data channel and control channel are separate socket connections within the same process. The client sends a command and waits to finish before issuing the next command. This is good for a single host, traditional-type user. If you have a single host and you want an ultra-reliable and light weight file transfer service, this is a good choice. This configuration is also good for testing purposes.

3. Firewall requirements

If the GridFTP server is behind a firewall:

1. Contact your network administrator to open up port 2811 (for GridFTP control channel connection) and a range of ports (for GridFTP data channel connections) for the incoming connections. If the firewall blocks the outgoing connections, open up a range of ports for outgoing connections as well.
2. Set the environment variable `GLOBUS_TCP_PORT_RANGE`:

```
export GLOBUS_TCP_PORT_RANGE=min,max
```

where `min,max` specify the port range that you have opened for the incoming connections on the firewall. This restricts the listening ports of the GridFTP server to this range. Recommended range is 1000 (e.g., 50000-51000) but it really depends on how much use you expect.

3. If you have a firewall blocking the outgoing connections and you have opened a range of ports, set the environment variable `GLOBUS_TCP_SOURCE_RANGE`:

```
export GLOBUS_TCP_SOURCE_RANGE=min,max
```

where `min,max` specify the port range that you have opened for the outgoing connections on the firewall. This restricts the outbound ports of the GridFTP server to this range. Recommended range is twice the range used for `GLOBUS_TCP_PORT_RANGE`, because if parallel TCP streams are used for transfers, the listening port would remain the same for each connection but the connecting port would be different for each connection.



Note

If the server is behind NAT, the `--data-interface <real ip/hostname>` option needs to be used on the server.

If the GridFTP *client* is behind a firewall:

1. Contact your network administrator to open up a range of ports (for GridFTP data channel connections) for the incoming connections. If the firewall blocks the outgoing connections, open up a range of ports for outgoing connections as well.
2. Set the environment variable `GLOBUS_TCP_PORT_RANGE`

```
export GLOBUS_TCP_PORT_RANGE=min,max
```

where `min,max` specify the port range that you have opened for the incoming connections on the firewall. This restricts the listening ports of the GridFTP client to this range. Recommended range is 1000 (e.g., 50000-51000) but it really depends on how much use you expect.

3. If you have a firewall blocking the outgoing connections and you have opened a range of ports, set the environment variable `GLOBUS_TCP_SOURCE_RANGE`:

```
export GLOBUS_TCP_SOURCE_RANGE=min,max
```

where `min,max` specify the port range that you have opened for the outgoing connections on the firewall. This restricts the outbound ports of the GridFTP client to this range. Recommended range is twice the range used for `GLOBUS_TCP_PORT_RANGE`, because if parallel TCP streams are used for transfers, the listening port would remain the same for each connection but the connecting port would be different for each connection.

Additional information on Globus Toolkit Firewall Requirements is available [here](#)¹.

4. Configuring Security for GridFTP

There are many security options in GridFTP ranging from no security to higher security via GSI .

4.1. Anonymous mode

Anonymous mode (using the `-aa` option) allows any user with an FTP client to read and write (and delete) files that the server process can similarly access (it is also a quick way to test that your server works).

```
% globus-gridftp-server -aa
    Server listening at 127.0.0.1:58806
```

Warning

When the server is run in this way, anyone who can connect to the server will possess all the same rights as the user that the process is run as (directly or via `-anonymous-user`). If using this mode intentionally for open access, it is best to run under a dedicated account with limited filesystem permissions. You can also use the option below to disable FTP commands such as STOR, ESTO, DELE, RDEL, RNT0, etc to make sure that users can only read from the server and not write to it.

```
-disable-command-list <string>
```

Where `<string>` represents a comma separated list of client commands that will be disabled. Default: not set.

4.2. Username/password

If you trust your network and want a minimal amount of security, you can run the `globus-gridftp-server` with clear text passwords. This security model is the one originally introduced in RFC959.

Warning

We do not recommend it for long running servers open to the internet.

4.2.1. Create password file

To run the server in clear text password mode, we first need to create a password file dedicated to it. The format of the password file is the same as standard system password files; however, it is ill-advised to use a system password file. To create an entry in a GridFTP password file, run the following commands:

```
% touch pwfile
    % gridftp-password.pl >> pwfile
    Password:
```

This will ask you for a password and then create an entry in the password file for the current user name and the given password. Take a look at the file created. You will notice that the password you typed in is not in the file in a clear text form. We have run it through a one way hash algorithm before storing it in the file.

¹ <http://www.globus.org/toolkit/security/firewalls/>

4.2.2. Run the server in password mode

Simply start the server pointing it at the password file you just created.

```
% globus-gridftp-server -password-file /full/path/of/pwfile
    Server listening at 127.0.0.1:5555
```

4.2.3. Make a transfer

To run `globus-url-copy` with the password, use the following syntax:

```
globus-url-copy file:///etc/group ftp://username:pw@localhost:5000/tmp/group
```

4.3. SSHFTP (GridFTP-over-SSH)

This type of security introduces the `sshftp` control channel (frontend) protocol. This is a very simple means of obtaining strong security on the control channel only (the data channel is *not* authenticated). With this approach, you can run a GridFTP transfer anywhere that you can `ssh`. `sshftp://` leverages the ubiquitous `ssh/ssh` programs to form control channel connections much in the same way that `inetd` forms connections.

4.3.1. Configure Client-Side `sshftp://`

Every `$GLOBUS_LOCATION` must be configured for client-side `sshftp://` connections. In other words, if we wish to use **globus-url-copy** with `sshftp://` URLs, we must first configure the `$GLOBUS_LOCATION` that contains `globus-url-copy` in the following way:

```
% $GLOBUS_LOCATION/setup/globus/setup-globus-gridftp-sshftp
```

4.3.2. Configure Server Side `sshftp://`

Every host that wishes to run a **globus-gridftp-server** which can accept `sshftp://` connections must run the following command as root:

```
% $GLOBUS_LOCATION/setup/globus/setup-globus-gridftp-sshftp -server
```

In the absence of root access, a user can configure the server to allow `sshftp://` connections for that user only with the following command:

```
% $GLOBUS_LOCATION/setup/globus/setup-globus-gridftp-sshftp -server -nonroot
```

The above command creates a file named 'sshftp' in '/etc/grid-security' (if run as root) or in '\$HOME/.globus' (if run as nonroot). The default contents of the 'sshftp' file is shown below. To configure the GridFTP server for `sshftp` transfers, you have to edit this file.

```
export GLOBUS_LOCATION=/sandbox/kettimut/421/INSTALL
. $GLOBUS_LOCATION/etc/globus-user-env.sh

#export GLOBUS_TCP_PORT_RANGE=50000,50100

$GLOBUS_LOCATION/sbin/globus-gridftp-server -ssh
# -data-interface <interface to force data connections>
```

4.3.3. Performing `sshftp://` Transfers

In this case, a `globus-gridftp-server` does not need to be running. The server will be started via the `sshd` program. Therefore, the hostname and port should be that of the `sshd` server. Run `globus-url-copy` just as you have before; simply change `ftp://` to `sshftp://`.

```
% globus-url-copy -v file:/etc/group sshftp://127.0.0.1/tmp/group % globus-url-copy -list
```

4.4. GSIFTP

This security option can be the most involved to set up, but provides the most security. It requires setting up GSI security as described in the GT Installation Guide here: [Basic Security Configuration](#).

Once GSI has been set up (host and user credentials are valid, the gridmap file is updated and you've run `grid-proxy-init` to create a proxy certificate), you simply run the GridFTP server:

```
globus-gridftp-server
```

Note

If run as `root`, it will pick up the host cert; if not, it will pick up the user cert.

Now you are ready to perform a GSI-authenticated transfer:

```
globus-url-copy <-s subject> src_url dst_url
```

Note

The `subject` option is only needed if the server was not started as `root`.

4.4.1. Running in daemon mode

The server should generally be run as `root` in daemon mode, although it is possible to run it as a user (see below). When run as `root` you will need to have a [host certificate](#).

Run the server:

```
globus-gridftp-server < -s | -S > <args>
```

where:

- s Runs in the foreground (this is the default mode).
- S Detaches from the terminal and runs in the background.

The following additional steps may be required when running as a user other than `root` (for more details, review [Basic Security Configuration](#)):

- Create a `~/ .gridmap` file, containing the DNs of any clients you wish to allow, mapped to the current username.
- Create a proxy with `grid-proxy-init`.

4.4.2. Running under inetd or xinetd

Note

We also feature a user-configurable, super-server daemon plugin called GFork. Click [here](#) for more information.

4.4.2.1. Set up xinetd/inetd config file

Note

The service name used (gsift in this case) should be defined in `/etc/services` with the desired port.

Here is a sample GridFTP server xinetd config entry in `/etc/xinetd.conf`:

```
service gsiftp
{
    instances            = 100
    socket_type         = stream
    wait                = no
    user                = root
    env                 += GLOBUS_LOCATION=(globus_location)
    env                 += LD_LIBRARY_PATH=(globus_location)/lib
    server              = (globus_location)/sbin/globus-gridftp-server
    server_args         = -i
    log_on_success      += DURATION
    nice                = 10
    disable             = no
}
```

Here is a sample gridftp server inetd config entry in `/etc/inetd.conf` (read as a single line):

```
gsiftp stream tcp nowait root /usr/bin/env env \
GLOBUS_LOCATION=(globus_location) \
LD_LIBRARY_PATH=(globus_location)/lib \
(globus_location)/sbin/globus-gridftp-server -i
```

Note

On Mac OS X, you must set `DYLD_LIBRARY_PATH` instead of `LD_LIBRARY_PATH` in the above examples.

On IRIX, you may need to set either `LD_LIBRARYN32_PATH` or `LD_LIBRARY64_PATH`.

Note

You should NOT include `USERID` in the log lines. See [Section 5, “High latency for GridFTP server connections”](#) for more information.

4.4.2.2. globus-gridftp-server -i

Use the `-i` commandline option with `globus-gridftp-server`:

```
globus-gridftp-server -i
```

4.5. User permissions

Users are mapped to a local account on the server machine and file permissions are handled by the operating systems. In the anonymous mode, users that connect to the server will possess all the same rights as the user that the server process is run as (directly or via `-anonymous-user`).

In case of username/password authentication, the users are mapped to the uid corresponding to the username in the GridFTP password file and the access permissions for the users is same as that of the UID that they are mapped to. If SSH based authentication is used, upon successful authentication, SSHD maps users to a local account and the GridFTP server is run as the mapped local user. The access permissions are the same as that of the mapped local user.

If GSI is used, upon successful authentication an authorization callout is invoked to (a) verify authorization and (b) determine the local user id as which the request should be executed. This callout is linked dynamically. Globus GridFTP provides an implementation that supports a Globus "gridmapfile". Sites can also provide alternative implementations. Server does a setuid to the local user id as determined by the authorization callout and the access permissions are the same as that of the local user id.

GridFTP server provides an option to disable certain FTP commands:

```
-disable-command-list <string>
```

Where `<string>` represents a comma separated list of client commands that will be disabled. Default: not set.

5. globus-gridftp-server quickstart

The following is a quick guide to running the server and using the client:

Look through the list of options for `globus-gridftp-server`:

```
globus-gridftp-server --help
```

Start the server in anonymous mode (discussed more fully [here](#)):

```
globus-gridftp-server -control-interface 127.0.0.1 -aa -p 5000
```

where:

`-control-interface` is the hostname or IP address of the interface to listen for control connections on. This option is only needed here as a rudimentary means of security for this simple example.

`-aa` enables anonymous mode

`-p` indicates on which port the server listens.

Run a two party transfer with client:

```
globus-url-copy -v file:///etc/group ftp://localhost:5000/tmp/group
```

Run 3rd party transfer:

```
globus-url-copy -v ftp://localhost:port/etc/group ftp://localhost:port/tmp/group2
```

Experiment with `-dbg`, and `-vb` options for debugging and checking the performance of your setup:

```
globus-url-copy -dbg file:///etc/group ftp://localhost:5000/tmp/group
```

```
globus-url-copy -vb file:///dev/zero ftp://localhost:5000/dev/null
```

where:

- dbg A useful option when something is not working. It results in a GridFTP control channel protocol dump (along with other useful information) to stderr. If you understand the GridFTP protocol, or you have ambition to understand it, this can be a very useful tool to discover various problems in your setup such as overloaded servers and firewalls. When submitting a bug report or asking a question on the support email lists one should always send along the -dbg output.
- vb Provides a type of progress bar of the user to observe the rate at which their transfer is progressing.

Ctrl-c - Kill the server.



Note

There are many possible options and configurations with **globus-gridftp-server**. For some guidelines on setting it up for your situation, see [Chapter 3, Key Admin Settings and Tuning Recommendations](#).

Chapter 3. Key Admin Settings and Tuning Recommendations

The **globus-gridftp-server** is a flexible and tunable piece of software. It is easy for an admin to get lost in all of the options it offers. This document intends to highlight some of the more commonly important options related to performance and robustness. It does not intend to account for all of the options but rather to give the system administrator a better perspective into how to set some of the less obvious controls.

1. Concurrent Instances

A very important option for a system administrator to set is the number of simultaneous GridFTP transfers allowed. In other words, the number of clients that are allowed to connect to the server at the same time.

GridFTP is designed to be a high performance, on-demand data transfer service. Quite a bit of system resources (mainly memory) are allocated to each client connection and this is with the assumption that the session will consume even more system resources (CPU, net/disk bandwidth) when performing a high speed data transfer. For this reason, the system administrator must evaluate the resource their host machine has to offer and set a reasonable limit to the number of client connects allowed at one time.

When determining the instance concurrency level, there are two major factors to consider: **system memory** helps determine the upper limit of the instance range and **available I/O bandwidth** helps determine the lower limit.

1.1. System memory considerations

First and foremost is system memory. The recommended instance count based on system memory is:

```
instance count = system memory / 34
```

Each instance of a GridFTP server will require about 2MB of memory just to handle the connection in a sane way. Beyond that is the amount of memory required to handle a fast, TCP-based data transfer. A safe rule of thumb here is 32MB. This allows for a TCP buffer size of 16MB (which is a common client selection for high performance WAN bandwidth delay products) and a user space buffer to match that value. $2\text{MB} + 16\text{MB} + 16\text{MB} = 34\text{MB}$, thus the denominator in the above formula.

1.2. I/O bandwidth considerations

Simultaneous clients share the available I/O resources. Most often it is beneficial to allocate enough bandwidth so that each client can transfer data at an acceptable rate. In a simple model, the higher the instance count, the lower the transfer rates for each client. At some point it is beneficial for the GridFTP server to reject connections in an attempt to provide a higher level of service to its currently connected clients. There is also a point where too many simultaneous clients can cause thrashing and drop network packets. Obviously this situation should be avoided.

While no client wants to be rejected, a higher level service can take advantage of this by either trying again later at a more efficient time, reordering its work load, or finding a replica. RFT provides some of this functionality and other such services are being researched and designed.

1.3. Why More Than One?

When considering the right concurrent instance level it is helpful to consider why there should ever be more than one at a time. There are three major reasons for this:

1. **The other side of the connection is the bottleneck.**

If we assume that each transfer moves as fast as our system can send it, then, when considering overall throughput, having two connections going at half speed is roughly the same thing as having 2 full speed connections run one at a time. However, if the remote end of the connection is the bottleneck, then there is unused local bandwidth from which another simultaneous connection and thus the overall system can benefit.

2. **Hide the overhead.**

Another important aspect of simultaneous connections is that the needed overhead of control messaging can be overlapped with the payload of another sessions data transfer. Hiding this processing and messaging latency makes for a more efficient system with a higher overall throughput.

3. **Provide an interactive service.**

In some case, users may find connection rejections unacceptable and would prefer a slower overall system provided they could connect to it immediately for the purpose of an interactive session.

In the case of #3 the the highest safe level of instance count possible is ideal. In the other two case the ideal number is less deterministic. At least 10 instance is always recommended.

1.4. Setting the instance cap

If using GForge or Xinetd, set the instance cap by adding the following line to the configuration file:

```
instance = <integer>
```

If running the GridFTP server as a daemon, use the following option to set the instance cap:

```
-connections-max <integer>
```

2. Disk Block Size

The **globus-gridftp-server** sits on top of various file systems. Each file system has its own ideal access patterns and I/O buffer sizes. To provide the user with some means of control, we offer the option:

```
-blocksize <number>
```

This number indicates the size of the read requests posted to the disk.

3. System TCP Buffer Settings

The most important setting in achieving high performance, TCP-based transfers is the TCP buffer size. It is our experience that this should be set to at least 16MB. On Linux systems, this is done by editing the file `/etc/sysctl.conf` and adding the following lines:

```
net/core/rmem_max = 16777216
net/core/wmem_max = 16777216

net/ipv4/tcp_rmem = 8192 1048576 16777216
net/ipv4/tcp_wmem = 8192 1048576 16777216
```

This sets the max to 16MB, the default to 1MB, and the min to 8KB. In most cases this will be a good value, but administrators are encouraged to experiment.

4. Data Interface

On systems that have multiple network interfaces, the system admin likely wants to associate data transfers with the fastest possible NIC available. This can be done in the GridFTP server by using the option:

```
--data-interface <ip address>
```

Chapter 4. Advanced Configuration

It is assumed that the toolkit installation was successful. For more information, see the [Installing GT 5.0.2](#). Also be sure to reference [Chapter 2, Configuring GridFTP](#) and [globus-gridftp-server](#).

1. Configuring GridFTP to use UDT instead of TCP

UDT is bundled with Globus starting with Globus v4.2, so downloading UDT separately is no longer needed.

1.1. Prerequisites

1. Threaded build of the Globus GridFTP server. In GT5, the default flavor of the server is non-threaded. Refer to [Section 2.6, “Switching between threaded and non-threaded flavors”](#) for information on how to switch between threaded and non-threaded flavors of GridFTP server.
2. For client-server transfers, threaded build of globus-url-copy. For third-party (server-server) transfers, threaded build of globus-url-copy is not needed. Refer to [Section 2.6, “Switching between threaded and non-threaded flavors”](#) for information on how to switch between threaded and non-threaded flavors of globus-url-copy.

1.2. Steps

1. Build and install UDT

```
globus$ make udt ("make gridftp udt" if gridftp is not built and installed already)
globus$ make install
```

2. Configure GridFTP server

```
If you run the GridFTP server from xinetd, add '-dc-whitelist
    udt,gsi,tcp' to 'server_args' in /etc/xinetd.d/gsiftp
```

Alternatively, you can use the file `$GLOBUS_LOCATION/etc/gridftp.conf` to configure this. Add the following to that file:

```
dc-whitelist udt,gsi,tcp
```

If you run the server from commandline:

```
$GLOBUS_LOCATION/sbin/globus-gridftp-server -dc-whitelist
    udt,gsi,tcp
```

3. Run globus-url-copy with new command line option "-udt"

2. Configuring GridFTP to enable Netlogger's bottleneck detection

For information on enabling bottleneck detection via Netlogger, see the [Gridftp-netlogger](#)¹ page on the CEDPS website.

3. Separation of Processes (Split Process)

The GridFTP server can be separated into front end and data node processes. This is the architecture used to achieve a striped server, but it can also be exploited to achieve a higher level of security.

Running the server as `root` is often desirable because it allows the server to fork and `setuid` on a child process related to an authenticated user. This allows the server to leverage the operating system's file system permissions and other security devices. However, it is not at all desirable to have a `root`-running process listening on a port open to the world. If an attacker were to compromise the process, they could obtain root-level access to the machine.

To overcome this security risk, the GridFTP server can be run in a front end/back end manner. The front end can be run as any user, say user `globus`, that has very limited access to the machine. The front end is the process open to the outside world. If it is compromised, an attacker has only gained access to that limited account. The back end is run as `root`, but configured to only allow connections from the front end.

This does, however, require that a copy of the [host cert and host key](#) be owned by the non-privileged user. If you use this configuration, the non-privileged user should not have write permission to executables, configuration files, etc. This provides greater security and also allows for proxying and load balancing. Many backend data movers can be behind a single point of client contact. Each client is assigned a different backend in a round robin fashion.

To start the front end, run:

```
globus-gridftp-server -p 7000 -r localhost:7001
```

To start the back end, run:

```
globus-gridftp-server -p 7001 -dn -allow-from 127.0.0.1
```

4. Configuring GridFTP for Cluster-to-Cluster (or Striped) data movement

The GridFTP server supports separate front end (client control connection) and back end (data node) processes. In addition, a single front end process may connect to multiple back end data nodes.

When multiple back end data nodes are available, the server is said to be in a *striped* configuration, or simply, is a striped server. In this mode, transfers are divided over all available data nodes, thus allowing the combined bandwidth of all data nodes to be used.

This is recommended for improved performance of large (1GB+) file transfers. This can also be useful if you want to use full data encryption and need to tether together many hosts to handle the processing load.



Note

The connection between the front end and data nodes is referred to as the *IPC channel*.

¹ <http://www.cedps.net/index.php/Gridftp-netlogger>

The ability to use `inetd` or `daemon` execution modes applies to both front end servers and data nodes, and the same certificate and user requirements apply.

To start the front end, run:

```
globus-gridftp-server <args> -r <host:port>[,<host:port>,...]
```

To start the data-node, run:

```
globus-gridftp-server -p <port> -dn
```

The `-p <port>` option used on the data-node is the port that will be used for IPC connections. This is the port that you will register with the front end server.

For example:

```
machineB> globus-gridftp-server -p 6000 -dn
machineC> globus-gridftp-server -p 7000 -dn
machineA> globus-gridftp-server -p 5000 -r machineB:6000,machineC:7000
```

The client would only connect to the front end at `machineA:5000`, for example, using `globus-url-copy` with the `-stripe` option:

```
globus-url-copy -stripe gsiftp://machineA:5000/file file:///destination
or
globus-url-copy -stripe gsiftp://machineA:5000/file gsiftp://machineX/destination
```

Where `machineX` may be another striped server or a standard GridFTP server.

5. Combining Split Process and Striped configuration

Furthermore striped servers and split process can be combined. You can have an 8 node cluster that only uses 2 nodes at a time in a striped server configuration and load balances across the rest of the nodes. TODO: any other details here?

6. Running with GFork Master Plugin

GFork is a service like `inetd` that listens on a TCP port and runs a configurable executable in a child process whenever a connection is made. GFork also creates bi-directional pipes between the child processes and the master service. These pipes are used for interprocess communication between the child process executables and a master process plugin. More information on GFork can be found [here](#).

7. Configuring multicasting/broadcasting

To enable [multicasting](#), you must whitelist the `gridftp_multicast` driver with the `-fs-whitelist file,gridftp_multicast` option:

```
globus-gridftp-server -fs-whitelist file,gridftp_multicast
```

The above command whitelists both the `file` driver and the `gridftp_multicast` driver.

 **Note**

The `file` driver is the default `XIO` driver that handles reading and writing to file systems (disks). By default, this driver is already whitelisted. However, if you use the `-fs-whitelist` option, you must set *all* the drivers you want whitelisted (and the `file` driver will still be needed to allow reads and writes to disk for non-multicast users).

For information about using multicasting, click [here](#).

8. Accessing data from other data interfaces

8.1. GridFTP and DSIs

The following information is helpful if you want to use GridFTP to access data in DSIs (such as HPSS and SRB), and non-POSIX data sources.

Architecturally, the Globus GridFTP *server* can be divided into 3 modules:

- the GridFTP protocol module,
- the (optional) data transform module, and
- the Data Storage Interface (DSI).

In the GT 5.0.2 implementation, the data transform module and the DSI have been merged, although we plan to have separate, chainable, data transform modules in the future.

 **Note**

This architecture does NOT apply to the WU-FTPD implementation (GT3.2.1 and lower).

8.1.1. GridFTP Protocol Module

The GridFTP protocol module is the module that reads and writes to the network and implements the GridFTP protocol. This module should not need to be modified since to do so would make the server non-protocol compliant, and unable to communicate with other servers.

8.1.2. Data Transform Functionality

The data transform functionality is invoked by using the ERET (extended retrieve) and ESTO (extended store) commands. It is seldom used and bears careful consideration before it is implemented, but in the right circumstances can be very useful. In theory, any computation could be invoked this way, but it was primarily intended for cases where some simple pre-processing (such as a partial get or sub-sampling) can greatly reduce the network load. The disadvantage to this is that you remove any real option for planning, brokering, etc., and any significant computation could adversely affect the data transfer performance. Note that the *client* must also support the ESTO/ERET functionality as well.

8.1.3. Data Storage Interface (DSI) / Data Transform module

The Data Storage Interface (DSI) / Data Transform module knows how to read and write to the "local" storage system and can optionally transform the data. We put local in quotes because in a complicated storage system, the storage may not be directly attached, but for performance reasons, it should be relatively close (for instance on the same LAN).

The interface consists of functions to be implemented such as send (get), receive (put), command (simple commands that simply succeed or fail like mkdir), etc..

Once these functions have been implemented for a specific storage system, a client should not need to know or care what is actually providing the data. The server can either be configured specifically with a specific DSI, i.e., it knows how to interact with a single class of storage system, or one particularly useful function for the ESTO/ERET functionality mentioned above is to load and configure a DSI on the fly.

See [Appendix A, Developing DSIs for GridFTP](#) for more information.

8.2. Plugging in a Data Storage Interface (DSI)

GridFTP can be used as a network interface to existing Data Storage Interfaces (DSIs) using the `-dsi` option. With this option the DSI plugs into the backend (compatible with striping) and is transparent to the client or remote party. It can be used with either the GT standard DSI plugins or with [custom-built DSI plugins](#). The standard DSI plugins available in a default GT installation are:

- [Storage Resource Broker \(SRB\)](#)

The above link point to complete information about setting up and running the GridFTP server with these DSIs.

8.3. Accessing data in a non-POSIX file data source that has a POSIX interface

If you want to access data in a non-POSIX file data source that has a POSIX interface, the standard server will do just fine. Just make sure it is really POSIX-like (out of order writes, contiguous byte writes, etc).

9. GridFTP Where There Is FTP (GWTFTP)

GridFTP Where There Is FTP (GWTFTP) is an intermediate program that acts as a proxy between existing FTP clients and GridFTP servers. Users can connect to GWFTP with their favorite standard FTP client, and GWFTP will then connect to a GridFTP server on the client's behalf. To clients, GWFTP looks much like an FTP proxy server. When wishing to contact a GridFTP server, FTP clients instead contact GWTFTP.

Clients tell GWFTP their ultimate destination via the FTP **USER <username>** command. Instead of entering their username, client users send the following:

```
USER <GWTFTP username>::<GridFTP server URL>
```

This command tells GWTFTP the GridFTP endpoint with which the client wants to communicate. For example:

```
USER bresnaha::gsiftp://wiggum.mcs.anl.gov:2811/
```



Note

Requires [GSIC security](#).

Chapter 5. Testing

If the `globus-ftp-client-test` package has been installed, our standard test suite may be run to verify functionality on your platform. Simply set up the `globus` environment, `chdir` to `$GLOBUS_LOCATION/test/globus_ftp_client_test/` and run `./TESTS.pl`.

Chapter 6. Debugging

1. Logging

As of Globus 4.2.0, GridFTP server provides system administration logs in 2 different formats. The CEDPS best practices compliant format is a new format provided by GridFTP server available in Globus 5.0.2. For more details on the CEDPS Logging format, see <http://cedps.net/index.php/LoggingBestPractices>.

1.1. Configuring CEDPS format system administration logs

```
globus-gridftp-server -log-module stdio_ng -log-level info,warn,error -logfile /var/log/gr
```

For more information about the logging options, see [globus-gridftp-server\(1\)](#).

Sample log file: [gridftp.log1](#)¹

1.2. Configuring traditional format system administration logs

```
globus-gridftp-server -log-module stdio -log-level info,warn,error -logfile /var/log/gridf
```

which is the same as

```
globus-gridftp-server -log-level info,warn,error -logfile /var/log/gridftp.log
```

`stdio` is the default log-module.

Sample log file: [gridftp.log2](#)²

1.3. Netlogger-style logging

Apart from the 2 formats mentioned above, GridFTP server can log netlogger style information for each transfer.

```
globus-gridftp-server -log-transfer /var/log/gridftp.log
```

Sample log file: [gridftp.log3](#)³

¹ /toolkit/docs/5.0/5.0.2/data/gridftp/gridftp.log1

² /toolkit/docs/5.0/5.0.2/data/gridftp/gridftp.log2

³ /toolkit/docs/5.0/5.0.2/data/gridftp/gridftp.log3

Chapter 7. Troubleshooting

If you are having problems using the GridFTP *server*, try the steps listed below. If you have an error, try checking the server logs if you have access to them. By default, the server logs to stderr, unless it is running from inetd, or its execution mode is detached, in which case logging is disabled by default.

The command line options `-d`, `-log-level`, `-L` and `-logdir` can affect where logs will be written, as can the configuration file options `log_single` and `log_unique`. See the [globus-gridftp-server\(1\)](#) for more information on these and other configuration options.

For a list of common errors in GT, see [Error Codes](#).

1. Error Codes in GridFTP

Table 7.1. GridFTP Errors

Error Code	Definition	Possible Solutions
<p>globus_ftp_client: the server responded with an error 530 530-globus_xio: Authentication Error 530-OpenSSL Error: s3_srvr.c:2525: in library: SSL routines, function SSL3_GET_CLIENT_CERTIFICATE: no certificate returned 530-globus_gsi_callback_module: Could not verify credential 530-globus_gsi_callback_module: Can't get the local trusted CA certificate: Untrusted self-signed certificate in chain with hash d1b603c3 530 End.</p>	<p>This error message indicates that the GridFTP server doesn't trust the certificate authority (CA) that issued your certificate.</p>	<p>You need to ask the GridFTP server administrator to install your CA certificate chain in the GridFTP server's trusted certificates directory.</p>
<p>globus_ftp_control: gss_init_sec_context failed OpenSSL Error: s3_clnt.c:951: in library: SSL routines, function SSL3_GET_SERVER_CERTIFICATE: certificate verify failed globus_gsi_callback_module: Could not verify credential globus_gsi_callback_module: Can't get the local trusted CA certificate: Untrusted self-signed certificate in chain with hash d1b603c3</p>	<p>This error message indicates that your local system doesn't trust the certificate authority (CA) that issued the certificate on the resource you are connecting to.</p>	<p>You need to ask the resource administrator which CA issued their certificate and install the CA certificate in the local trusted certificates directory.</p>

Error Code	Definition	Possible Solutions
530-globus_xio: Authentication Error 530-globus_gsi_callback_module: Could not verify credential 530-globus_gsi_callback_module: Could not verify credential 530-globus_gsi_callback_module: Invalid CRL: The available CRL has expired 530 End.	This error message indicates one of the following: Certificate Revocation List (CRL) for the source or destination server CA at the client has expired or CRL for client CA has expired at source or destination server or CRL for source (destination) server CA has expired at destination (source) server. CRL is a file {CA_hash}.r0 in /etc/grid-security/certificates or \${USER_HOME}/.globus/certificates or \${X509_CERT_DIR}	The tool available at http://dist.eu-gridpma.info/distribution/util/fetch-crl/ can be run in a crontab to keep the CRLs up to date.

2. Establish control channel connection

Verify that you can establish a control channel connection and that the server has started successfully by telnetting to the port on which the server is running:

```
% telnet localhost 2811
      Trying 127.0.0.1...
      Connected to localhost.
      Escape character is '^]'.
      220 GridFTP Server mldev.mcs.anl.gov 2.0 (gcc32dbg, 1113865414-1) ready.
```

If you see anything other than a 220 banner such as the one above, the server has not started correctly.

Verify that there are no configuration files being unexpectedly loaded from /etc/grid-security/gridftp.conf or \$GLOBUS_LOCATION/etc/gridftp.conf. If those files exist, and you did not intend for them to be used, rename them to .save, or specify -c none on the command line and try again.

If you can log into the machine where the server is, try running the server from the command line with only the -s option:

```
$GLOBUS_LOCATION/sbin/globus-gridftp-server -s
```

The server will print the port it is listening on:

```
Server listening at gridftp.mcs.anl.gov:57764
```

Now try and telnet to that port. If you still do not get the banner listed above, something is preventing the socket connection. Check firewalls, tcp-wrapper, etc.

If you now get a correct banner, add -p 2811 (you will have to disable (x)inetd on port 2811 if you are using them or you will get port already in use):

```
$GLOBUS_LOCATION/sbin/globus-gridftp-server -s -p 2811
```

Now telnet to port 2811. If this does not work, something is blocking port 2811. Check firewalls, tcp-wrapper, etc.

If this works correctly then re-enable your normal server, but remove all options but -i, -s, or -S.

Now telnet to port 2811. If this does not work, something is wrong with your service configuration. Check /etc/services and (x)inetd config, have (x)inetd restarted, etc.

If this works, begin adding options back one at a time, verifying that you can telnet to the server after each option is added. Continue this till you find the problem or get all the options you want.

At this point, you can establish a control connection. Now try running `globus-url-copy`.

3. Try running `globus-url-copy`

Once you've verified that you can establish a control connection, try to make a transfer using `globus-url-copy`.

If you are doing a *client*/server transfer (one of your URLs has `file:` in it) then try:

```
globus-url-copy -vb -dbg gsiftp://host.server.running.on/dev/zero file:///dev/null
```

This will run until you control-c the transfer. If that works, reverse the direction:

```
globus-url-copy -vb -dbg file:///dev/zero gsiftp://host.server.running.on/dev/null
```

Again, this will run until you control-c the transfer.

If you are doing a *third party transfer*, run this command:

```
globus-url-copy -vb -dbg gsiftp://host.server1.on/dev/zero gsiftp://host.server2.on/dev/nu
```

Again, this will run until you control-c the transfer.

If the above transfers work, try your transfer again. If it fails, you likely have some sort of file permissions problem, typo in a file name, etc.

4. If your server starts...

If the server has started correctly, and your problem is with a security failure or gridmap lookup failure, verify that you have security configured properly [here](#).

If the server is running and your client successfully authenticates but has a problem at some other time during the session, please ask for help on gt-user@globus.org¹. When you send mail or submit bugs, please always include as much of the following information as possible:

- Specs on all hosts involved (OS, processor, RAM, etc).
- `globus-url-copy -version`
- `globus-url-copy -versions`
- Output from the telnet test above.
- The actual command line you ran with `-dbg` added. Don't worry if the output gets long.
- Check that you are getting a FQDN and `/etc/hosts` that is sane.
- The server configuration and setup (`/etc/services` entries, `(x)inetd` configs, etc.).
- Any relevant lines from the server logs (not the entire log please).

¹ http://dev.globus.org/wiki/Mailing_Lists

5. High latency for GridFTP server connections

If you run GridFTP servers via Xinetd and notice high latency for connections and/or transfers, check if `/etc/xinetd.conf` or the `gsiftp` service configuration inside `/etc/xinetd.d` is set to log `USERID` as follows:

```
log_on_success += USERID
log_on_failure += USERID
```

Such a configuration tells Xinetd to log the remote user using the method defined in RFC 1413, which causes an ident client to attempt to query the machine that the connection is coming from before the service will run. Even when this succeeds, the response can't be trusted, and more often than not it is rejected or simply dropped (which results in the longest delays) by the remote firewall.

Latency can be reduced by making sure Xinetd does *not* log the `USERID`.

Chapter 8. Usage statistics collection by the Globus Alliance

1. GridFTP-specific usage statistics

The following GridFTP-specific usage statistics are sent in a UDP packet at the end of each transfer, in addition to the standard header information described in the [Usage Stats](#)¹ section.

- Start time of the transfer
- End time of the transfer
- Version string of the server
- TCP buffer size used for the transfer
- Block size used for the transfer
- Total number of bytes transferred
- Number of parallel streams used for the transfer
- Number of stripes used for the transfer
- Type of transfer (STOR, RETR, LIST)
- FTP response code -- Success or failure of the transfer



Note

The client (`globus-url-copy`) does NOT send any data. It is the *servers* that send the usage statistics.

We have made a concerted effort to collect only data that is not too intrusive or private and yet still provides us with information that will help improve and gauge the usage of the GridFTP server. Nevertheless, if you wish to disable this feature for GridFTP only, use the `-disable-usage-stats` option of `globus-gridftp-server`. Note that you can disable transmission of usage statistics globally for all C components by setting "GLOBUS_USAGE_OPTOUT=1" in your environment.

Also, please see our [policy statement](#)² on the collection of usage statistics.

¹ /toolkit/docs/5.0/5.0.2/Usage_Stats.html

² /toolkit/docs/latest-stable/Usage_Stats.html

Appendix A. GridFTP Admin Tool

Name

globus-gridftp-server -- Configures the GridFTP Server

globus-gridftp-server

Tool description

globus-gridftp-server configures the GridFTP server using a config file and/or commandline options.



Note

Command line options and configuration file options may both be used, but the command line *overrides* the config file.

The configuration file for the GridFTP *server* is read from the following locations, in the given order. Only the first file found will be loaded:

- Path specified with the `-c <configfile>` command line option.
- `$GLOBUS_LOCATION/etc/gridftp.conf`
- `/etc/grid-security/gridftp.conf`

Options are one per line, with the format:

```
<option> <value>
```

If the value contains spaces, they should be enclosed in double-quotes ("). Flags or boolean options should only have a value of 0 or 1. Blank lines and lines beginning with # are ignored.

For example:

```
port 5000
allow_anonymous 1
anonymous_user bob
banner "Welcome!"
```

Developer notes

The Globus implementation of the GridFTP *server* draws on:

- three IETF RFCs:
 - RFC 959
 - RFC 2228
 - RFC 2389
- an IETF Draft: MLST-16
- the GridFTP protocol specification, which is Global Grid Forum (GGF) Standard GFD.020.

The command line tools and the *client* library completely hide the details of the protocol from the user and the developer. Unless you choose to use the control library, it is not necessary to have a detailed knowledge of the protocol.

Command syntax

The basic syntax for **globus-gridftp-server** is:

```
globus-gridftp-server [optional command line switches]
```

To use **globus-gridftp-server** with a config file, make sure to use the `-c <configfile>` option.

Command line options

The table below lists config file options, associated command line options (if available) and descriptions.



Note

Any boolean option can be negated on the command line by preceding the specified option with '-no-' or '-n'.
example: -no-cas or -nf.

Informational Options

<code>help <0 1></code> , <code>-h</code> , <code>-help</code>	Show usage information and exit. Default value: FALSE
<code>version <0 1></code> <code>,</code> <code>-v</code> , <code>-version</code>	Show version information for the server and exit. Default value: FALSE
<code>versions</code> <code><0 1></code> , <code>-v</code> , <code>-versions</code>	Show version information for all loaded globus libraries and exit. Default value: FALSE

Modes of Operation

<code>inetd <0 1></code> , <code>-i</code> , <code>-inetd</code>	Run under an inetd service. Default value: FALSE
<code>daemon <0 1></code> , <code>-s</code> , <code>-daemon</code>	Run as a daemon. All connections will fork off a new process and setuid if allowed. See Section 4.4.1, “Running in daemon mode” for more information. Default value: TRUE
<code>detach <0 1></code> , <code>-S</code> , <code>-detach</code>	Run as a background daemon detached from any controlling terminals. See Section 4.4.1, “Running in daemon mode” for more information. Default value: FALSE
<code>ssh</code> , <code>-ssh</code>	Run over a connected ssh session. Default value: not set

exec <string> , -exec <string>	For statically compiled or non-GLOBUS_LOCATION standard binary locations, specify the full path of the server binary here. Only needed when run in <u>daemon mode</u> . Default value: not set
chdir <0 1>, -chdir	Change directory when the server starts. This will change directory to the dir specified by the chdir_to option. Default value: TRUE
chdir_to <string>, -chdir-to <string>	Directory to chdir to after starting. Will use / if not set. Default value: not set
fork <0 1>, -f, -fork	Server will fork for each new connection. Disabling this option is only recommended when debugging. Note that non-forked servers running as 'root' will only accept a single connection and then exit. Default value: TRUE
single <0 1>, -1, -single	Exit after a single connection. Default value: FALSE

Authentication, Authorization, and Security Options

auth_level <number>, -auth-level <number>	<ul style="list-style-type: none"> • 0 = Disables all authorization checks. • 1 = Authorize identity only. • 2 = Authorize all file/resource accesses. <p>If not set, the GridFTP Server uses level 2 for front ends and level 1 for data nodes.</p> <p>Default value: not set</p>
ipc_allow_from <string>, -ipc-allow-from <string>	<p>Only allow IPC connections (applicable for backend servers in a striped configuration) from these source IP addresses. Specify a comma-separated list of IP address fragments. A match is any IP address that starts with the specified fragment. Example: '192.168.1.' will match and allow a connection from 192.168.1.45. Note that if this option is used, any address not specifically allowed will be denied.</p> <p>Default value: not set</p>
ipc_deny_from <string>, -ipc-deny-from <string>	<p>Deny IPC connections (applicable for backend servers in a striped configuration) from these source IP addresses. Specify a comma-separated list of IP address fragments. A match is any IP address that starts with the specified fragment. Example: '192.168.2.' will match and deny a connection from 192.168.2.45.</p> <p>Default value: not set</p>
allow_from <string>, -allow-from <string>	<p>Only allow connections from these source IP addresses. Specify a comma-separated list of IP address fragments. A match is any IP address that starts with the specified fragment. Example: '192.168.1.' will match and allow a connection from 192.168.1.45. Note that if this option is used, any address not specifically allowed will be denied.</p>

	Default value: not set
deny_from <string> , -deny-from <string>	Deny connections from these source IP addresses. Specify a comma-separated list of IP address fragments. A match is any IP address that starts with the specified fragment. Example: '192.168.2.' will match and deny a connection from 192.168.2.45. Default value: not set
secure_ipc <0 1> , -si , -secure-ipc	Use GSI security on the IPC channel. Default value: TRUE
ipc_auth_mode <string> , -ia <string> , -ipc-auth- mode <string>	Set GSI authorization mode for the IPC connection. Options are one of the following: <ul style="list-style-type: none"> • none • host • self • subject:[subject] Default value: host
allow_anonym- ous <0 1> , -aa , -allow- anonymous	Allow cleartext anonymous access. If server is running as root, anonymous_user must also be set. Disables IPC security. Default value: FALSE
anonym- ous_names_al- lowed <string> , -an- onymous- names-allowed <string>	Comma-separated list of names to treat as anonymous users when allowing anonymous access. If not set, the default names of 'anonymous' and 'ftp' will be allowed. Use '*' to allow any user-name. Default value: not set
anonym- ous_user <string> , -an- onymous-user <string>	User to setuid to for an anonymous connection. Only applies when running as root. Default value: not set
anonym- ous_group <string> , -an- onymous-group <string>	Group to setgid to for an anonymous connection. If not set, the default group of anonymous_user will be used. Default value: not set
pw_file <string> , -password- file <string>	Enable cleartext access and authenticate users against this /etc/passwd formatted file. Default value: not set
connec- tions_max	Maximum concurrent connections allowed. Only applies when running in <u>daemon mode</u> . Unlimited if not set.

<code><number></code> , <code>-connections-</code> <code>max <number></code>	Default value: not set
<code>connec-</code> <code>tions_dis-</code> <code>abled <0 1></code> , <code>-connections-</code> <code>disabled</code>	Disable all new connections. Does not affect ongoing connections. This must be set in the configuration file and then a SIGHUP issued to the server in order to reload the configuration. Default value: FALSE
<code>offline_msg</code> <code><string></code> , <code>-offline-msg</code> <code><string></code>	Custom message to be displayed to clients when the server is offline via the <code>connections_disabled</code> or <code>connections_max = 0</code> options. Default value: not set
<code>disable_com-</code> <code>mand_list</code> <code><string></code> , <code>-disable-com-</code> <code>mand-list</code> <code><string></code>	A comma separated list of client commands that will be disabled. Default value: not set
<code>authz_cal-</code> <code>louts</code> , <code>-cas</code> , <code>-authz-cal-</code> <code>louts</code>	Enable the GSI authorization callout framework, for callouts such as CAS. Default value: TRUE
<code>acl</code> , <code>-em</code> , <code>-acl</code>	A comma separated list of ACL or event modules to load. Default value: not set

Logging Options

<code>log_level</code> <code><string></code> , <code>-d</code> <code><string></code> , <code>-log-level</code> <code><string></code>	Log level. A comma-separated list of levels from the following: <ul style="list-style-type: none"> • ERROR • WARN • INFO • DUMP • ALL For example: <pre>globus-gridftp-server -d error,warn,info</pre> You may also specify a numeric level of 1-255. Default value: ERROR
<code>log_module</code> <code><string></code> ,	Indicates the <code>globus_logging</code> module that will be loaded. If not set, the default <code>stdio</code> module will be used and the logfile options (see next option) will apply.

<pre>-log-module <string></pre>	<p>Built-in modules are <code>stdio</code> and <code>syslog</code>. Log module options may be set by specifying <code>module:opt1=val1:opt2=val2</code>. Available options for the built-in modules are:</p> <ul style="list-style-type: none"> • <code>interval</code> - Indicates buffer flush interval. Default is 5 seconds. A 0 second flush interval will disable periodic flushing, and the buffer will only flush when it is full. • <code>buffer</code> - Indicates buffer size. Default is 64k. A value of 0k will disable buffering and all messages will be written immediately. <p>Example:</p> <pre>-log-module stdio:buffer=4096:interval=10</pre> <p>Default value: not set</p>
<pre>log_single <string>, -l <string>, -logfile <string></pre>	<p>Indicates the path of a single file to which you want to log all activity. If neither this option nor <code>log_unique</code> is set, logs will be written to <code>stderr</code>, unless the execution mode is detached, or <code>inetd</code>, in which case logging will be disabled.</p> <p>Default value: not set</p>
<pre>log_unique <string>, -L <string>, -logdir <string></pre>	<p>Partial path to which <code>gridftp.(pid).log</code> will be appended to construct the log filename.</p> <p>Example:</p> <pre>-L /var/log/gridftp/</pre> <p>will create a separate log (<code>/var/log/gridftp/gridftp.xxxx.log</code>) for each process (which is normally each new <i>client</i> session). If neither this option nor <code>log_single</code> is set, logs will be written to <code>stderr</code>, unless the execution mode is detached, or <code>inetd</code>, in which case logging will be disabled.</p> <p>Default value: not set</p>
<pre>log_transfer <string>, -Z <string>, -log-transfer <string></pre>	<p>Log NetLogger-style info for each transfer into this file.</p> <p>Default value: not set</p> <p>Example: <code>DATE=20050520163008.306532 HOST=localhost PROG=globus-gridftp-server NL.EVT=FTP_INFO START=20050520163008.305913 USER=ftp FILE=/etc/group BUFFER=0 BLOCK=262144 NBYTES=542 VOLUME=/ STREAMS=1 STRIPES=1 DEST=[127.0.0.1] TYPE=RETR CODE=226</code></p> <p>Time format is <code>YYYYMMDDHHMMSS.UUUUUU</code> (microsecs).</p> <ul style="list-style-type: none"> • <code>DATE</code>: time the transfer completed. • <code>START</code>: time the transfer started. • <code>HOST</code>: hostname of the server. • <code>USER</code>: username on the host that transferred the file. • <code>BUFFER</code>: tcp buffer size (if 0 system defaults were used). • <code>BLOCK</code>: the size of the data block read from the disk and posted to the network. • <code>NBYTES</code>: the total number of bytes transferred.

- **VOLUME:** the disk partition where the transfer file is stored.
- **STREAMS:** the number of parallel TCP streams used in the transfer.
- **STRIPES:** the number of stripes used on this end of the transfer.
- **DEST:** the destination host.
- **TYPE:** the transfer type, RETR is a send and STOR is a receive (ftp 959 commands).
- **CODE:** the FTP rfc959 completion code of the transfer. 226 indicates success, 5xx or 4xx are failure codes.

`log_filemode` File access permissions of log files. Should be an octal number such as 0644 (the leading 0 is required),
`<string>`,

`-log_filemode`
`<string>` Default value: not set

`disable_usage_stats` Disable transmission of per-transfer usage statistics. See the [Usage Statistics](#)¹ section in the online documentation for more information.

`<0|1>`, `-disable-usage-stats`
`stats` Default value: FALSE

`usage_stats_target` Comma-separated list of contact strings for usage statistics listeners. The format of `<string>` is `host:port`.

`<string>`,
`-usage_stats_target`
`<string>` Default value: `usage-stats.globus.org:4810`

Example:

```
-usage-stats-target usage-stats.globus.org:4810,usage-stats.uc.teragrid.org
```

In this example, the usage statistics will be transmitted to the default Globus target (`usage-stats.globus.org:4810`) and another target (`usage-stats.uc.teragrid.org:5920`).

The usage stats sent to a particular receiver may be customized by configuring it with a taglist (`host:port!taglist`) The taglist is a list of characters that each correspond to a usage stats tag. When this option is unset, stats are reported to `usage-stats.globus.org:4810`. If you set your own receiver, and wish to continue reporting to the Globus receiver, you will need to add it manually. The list of available tags follow. Tags marked * are reported by default.

- *(e) START - start time of transfer
- *(E) END - end time of transfer
- *(v) VER - version string of gridftp server
- *(b) BUFFER - tcp buffer size used for transfer
- *(B) BLOCK - disk blocksize used for transfer
- *(N) NBYTES - number of bytes transferred

¹ ../../Usage_Stats.html

- *(s) STREAMS - number of parallel streams used
- *(S) STRIPES - number of stripes used
- *(t) TYPE - transfer command: RETR, STOR, LIST, etc
- *(c) CODE - ftp result code (226 = success, 5xx = fail)
- *(D) DSI - DSI module in use
- *(A) EM - event modules in use
- *(T) SCHEME - ftp, gsiftp, sshftp, etc. (client supplied)
- *(a) APP - guc, rft, generic library app, etc. (client supplied)
- *(V) APPVER - version string of above. (client supplied)
- (f) FILE - name of file/data transferred
- (i) CLIENTIP - ip address of host running client (control channel)
- (I) DATAIP - ip address of source/dest host of data (data channel)
- (u) USER - local user name the transfer was performed as
- (d) USERDN - DN that was mapped to user id
- (C) CONFID - ID defined by -usage-stats-id config option
- (U) SESSID - unique id that can be used to match transfers in a session and transfers across source/dest of a third party transfer. (client supplied)

us- Identifying tag to include in usage statistics data.

age_stats_id
<string>, -us- Default value: not set
age-stats-id
<string>

Single and Striped Remote Data Node Options

remote_nodes Comma-separated list of remote node contact strings. See [Remote data-nodes and striped operations](#) and [Separation of processes for higher security](#) for examples of using this option.

<string>, -r
<string>, -re- Default value: not set
mote-nodes
<string>

data_node This server is a back end data node. See [Separation of processes for higher security](#) for an example of using this option.

<0|1>, -dn,
-data-node Default value: FALSE

stripe_blocks- Size in bytes of sequential data that each stripe will transfer.

ize <number>,
-sbs <number> Default value: 1048576
, -stripe-

blocksize
<number>

stripe_count Number of stripes to use per transfer when this server controls that number. If remote nodes are statically configured (via -r or remote_nodes), this will be set to that number of nodes, otherwise the default is 1.

<number> ,
-stripe-count
<number>

Default value: not set

stripe_layout Stripe layout. 1 = Partitioned, 2 = Blocked.

<number> , -sl
<number> ,
-stripe-lay-
out <number>

Default value: 2

stripe_blocksize_locked Do not allow client to override stripe blocksize with the **OPTS RETR** command.

<0|1> ,
-stripe-block-
size-locked;

Default value: FALSE

stripe_layout_locked Do not allow client to override stripe layout with the **OPTS RETR** command.

<0|1> ,
-stripe-lay-
out-locked

Default value: FALSE

Disk Options

blocksize Size in bytes of data blocks to read from disk before posting to the network.

<number> , -bs
<number> ,
-blocksize
<number>

Default value: 262144

sync_writes Flush disk writes before sending a restart marker. This attempts to ensure that the range specified in the restart marker has actually been committed to disk. This option will probably impact performance and may result in different behavior on different storage systems. See the man page for **sync()** for more information.

<0|1> , -sync-
writes

Default value: FALSE

use_home_dirs Set the startup directory to the authenticated users home dir.

, -use-home-
dirs

Default value: TRUE

perms Set the default permissions for created files. Should be an octal number such as 0644. The default is 0644. Note: If umask is set it will affect this setting -- i.e. if the umask is 0002 and this setting is 0666, the resulting files will be created with permissions of 0664.

<string> ,
-perms
<string>

Default value: not set

file_timeout Timeout in seconds for all disk accesses. A value of 0 disables the timeout.

<number> ,

Default value: not set

-file-timeout
<number>

Network Options

port <number> Port on which a front end will listen for client control channel connections or on which a data node will listen for connections from a front end. If not set, a random port will be chosen and printed via the logging mechanism. See [Remote data-nodes and striped operations](#) and [Separation of processes for higher security](#) for examples of using this option.

Default value: not set

control_interface <string> Hostname or IP address of the interface to listen for control connections on. If not set, will listen on all interfaces.

, -control-interface
<string>

Default value: not set

data_interface <string> Hostname or IP address of the interface to use for data connections. If not set will use the current control interface.

, -data-interface
<string>

Default value: not set

ipc_interface <string>, Hostname or IP address of the interface to use for IPC connections. If not set, will listen on all interfaces.

-ipc-interface
<string>

Default value: not set

hostname <string>, Effectively sets the above control_interface, data_interface and ipc_interface options.

-hostname
<string>

Default value: not set

ipc_port <number>, Port on which the front end will listen for data node connections.

-ipc-port
<number>

Default value: not set

Timeouts

control_preauth_timeout <number>, Time in seconds to allow a client to remain connected to the control channel without activity before authenticating.

-control-preauth-timeout
<number>

Default value: 120

control_idle_timeout <number>; Time in seconds to allow a client to remain connected to the control channel without activity.

-control-idle-timeout
<number>

Default value: 600

ipc_idle_timeout <number>, Idle time in seconds before an unused IPC connection will close.

<number> ,

`-ipc-idle-` Default value: 600
`timeout <num-`
`ber>`

`ipc_con-` Time in seconds before cancelling an attempted IPC connection.
`nect_timeout`
`<number>`, Default value: 60
`-ipc-connect-`
`timeout <num-`
`ber>`

User Messages

`banner` Message that is displayed to the client before authentication.
`<string>`,
`-banner` Default value: not set
`<string>`

`banner_file` Read banner message from this file.
`<string>`,
`-banner-file` Default value: not set
`<string>`

`banner_terse` When this is set, the minimum allowed banner message will be displayed to unauthenticated
`<0|1>`, `-ban-` clients.
`ner-terse` Default value: FALSE

`banner_append` When this is set, the message set in the 'banner' or 'banner_file' option will be appended to the
`<0|1>`, `-ban-` default banner message rather than replacing it.
`ner-append` Default value: FALSE

`login_msg` Message that is displayed to the client after authentication.
`<string>`, `-lo-`
`gin-msg` Default value: not set
`<string>`

`lo-` Read login message from this file.
`gin_msg_file`
`<string>`, `-lo-` Default value: not set
`gin-msg-file`
`<string>`

Module Options

`load_dsi_mod-` Load this Data Storage Interface module. File and remote modules are defined by the server. If
`ule <string>`, not set, the file module is loaded, unless the `remote` option is specified, in which case the remote
`-dsi <string>` module is loaded. An additional configuration string can be passed to the DSI using the format
[module name]:[configuration string]. The format of the configuration string is
defined by the DSI being loaded.

Default value: not set

<code>allowed_modules <string></code> <code>, -allowed-modules <string></code>	Comma-separated list of ERET/ESTO modules to allow and, optionally, specify an alias for. Example: <code>-allowed-modules module1,alias2:module2,module3</code> (module2 will be loaded when a client asks for alias2). Default value: not set
<code>dc_whitelist <string></code> <code>-dc-whitelist <string></code>	A comma separated list of drivers allowed on the network stack. Default value: not set
<code>fs_whitelist <string></code> <code>-fs-whitelist <string></code>	A comma separated list of drivers allowed on the disk stack. Default value: not set
<code>popen_whitelist <string></code> <code>-popen-whitelist <string></code>	A comma separated list of programs that the popen driver is allowed to execute, when used on the network or disk stack. An alias may also be specified, so that a client does not need to specify the full path. Format is [alias:]prog,[alias:]prog. example: /bin/gzip,tar:/bin/tar Default value: not set

Other Options

<code>configfile <string></code> <code>, -c <string></code>	Path to configuration file that should be loaded. Otherwise will attempt to load \$GLOBUS_LOCATION/etc/gridftp.conf and /etc/grid-security/gridftp.conf. Default value: not set
<code>debug <0 1></code> <code>, -debug</code>	Set options that make the server easier to debug. Forces no-fork, no-chdir, and allows core dumps on bad signals instead of exiting cleanly. Not recommended for production servers. Note that non-forked servers running as root will only accept a single connection and then exit. Default value: FALSE

Limitations

For transfers using parallel data transport streams and for transfers using multiple computers at each end, the direction of the connection on the data channels must go from the sending to the receiving side. For more information about this limitations see <http://www.ogf.org/documents/GFD.20.pdf>.

Globus GridFTP server does not run on windows

Appendix B. GT 5.0.2 GridFTP: Setting up Storage Resource Broker (SRB)

1. Introduction

The Storage Resource Broker Data Storage Interface (SRB-DSI) is an extension to the GridFTP server that allows it to interact with SRB. Plugging this extension into a GridFTP server allows the GridFTP server to access a SRB resource and serve it to any GridFTP client as though it were a filesystem.

2. Architecture



The above image shows the architecture of the system. There are 4 major components:

- *SRB Server* - This is where the data is stored. It is accessed by the GridFTP server via the standard SRB protocols using GSI_AUTH.
- *SRB-DSI* - This component is the bridge between GridFTP and SRB. All operation requests and data are routed through this component. The GridFTP server makes requests of it, then it translates these requests into SRB client commands.
- *GridFTP Server* - A standard GridFTP 5.0.2 server is loaded with the SRB-DSI. Clients contact this server to access data in a SRB resource. The server passes the request to the SRB-DSI which, as described above, passes the request on to the SRB server. The responses to the requests return along the same path.
- *GridFTP Client* - A stock GridFTP client (like **globus-url-copy**). No modifications to the client are needed.

3. Software

You need the following items to use the SRB-DSI:

- *Globus Toolkit* - You need the GridFTP distributed with GT (compatible with 4.0.1 or later). You can find that [here](#)¹.
- *SRB Client 3.4.0* - You only need the client libraries to build the SRB-DSI, but you will need access to a running SRB server and resource. You can find the client libraries [here](#)².
- *SRB-DSI* - You can find the SRB-DSI [here](#)³.

4. Building

Instructions for building [Globus](#) and [SRB](#)⁴ are well documented in the above links. The following sections describe one way of building these two packages. However, if any questions or errors are discovered, the reader should look to the above links for solutions.

¹ <http://www.globus.org/toolkit/downloads/5.0.2/>

² <http://www.sdsc.edu/srb/tarfiles/main.html>

³ http://www-unix.mcs.anl.gov/~bresnaha/SRB_DSI_Doc/globus_srb_dsi-latest.tar.gz

⁴ <http://www.sdsc.edu/srb>

4.1. Building Globus

Download the source installer, choose a path on your filesystem for your GLOBUS_LOCATION, and run the following:

```
% bunzip2 gt5.0.2-all-source-installer.tar.bz2
% tar -xvf gt5.0.2-all-source-installer.tar
% export GLOBUS_LOCATION=<path you chose for your GLOBUS_LOCATION>
% ./configure --prefix=$GLOBUS_LOCATION
% make gridftp globus_gridftp_server-thr
% source $GLOBUS_LOCATION/etc/globus-user-env.sh
```

4.2. Building SRB

```
% ./configure --enable-gsi-auth --enable-globus-location=$GLOBUS_LOCATION --enable-globus
% make
```

4.3. Building SRB-DSI

The SRB-DSI is a GPT package. More information about GPT package installation can be found [here](#)⁵. Most users should simply need:

```
gpt-build -force CONFIGOPTS_GPTMACRO="--with-srb-path=<location of SRB source tree>" glob
```

5. Administration

Before you can run the GridFTP server with the SRB-DSI, you need to set up some files.

5.1. Creating and setting up the SRB configuration file

A configuration file must be created at:

```
$GLOBUS_LOCATION/etc/gridftp_srb.conf
```

The following values must be set in this file:

```
srb_hostname <host>:<port>
srb_hostname_dn <domain name to expect from SRB server>
srb_default_resource <default srb resource to use>
```

5.2. Setting up the gridmap file

Additionally, the gridmap file must be special for this DSI. Along with the subject name and username, the SRB-DSI needs to know the SRB domain name for the user. This is handled by adding an additional value to the gridmap file:

```
"<user security DN>" <srb user name>@<domain name>
```

⁵ <http://www.gridpackagingtools.org/>

6. Running

Once you have the configuration files in place, you can run the server.

Important

All options of the server apply, but the parameter `-dsi srb -auth-level 4` *must* also be used.

For more information on setting these values and running the GridFTP server see [Chapter 2, Configuring GridFTP](#).

Most users can run with:

```
$GLOBUS_LOCATION/sbin/globus-gridftp-server -p <port> -dsi srb -auth-level 4
```

7. See Also

See the [README](#)⁶ file for more information.

⁶ http://www-unix.mcs.anl.gov/~bresnaha/SRB_DSI_Doc/README.txt

Appendix C. Running the Globus GridFTP Server With GFork

1. Introduction

GFork is a user-configurable super-server daemon very similar to xinetd in that it listens on a TCP port. When clients connect to a port, it runs an administrator-defined program which services that client connection, just as xinetd does.

An unfortunate drawback to xinetd is that there is no way to maintain or share long-term information. Every time a client connects, a new process is created; and every time that client disconnects, the process is destroyed. All of the information regarding the specific interactions with a given client is lost with these transient processes. A further disadvantage is that there is no way for these service instances to share service-specific information with each other while they are running.

There are times when it is useful for a service to maintain long-term service-specific state, or for a service to share state across client connections. GFork is designed to address this situation. GFork runs a long term master program (that is user-defined) and forms communication links via UNIX pipes between this process and all client connection child processes. This allows long-term state to be maintained in memory and allows for communication between all nodes.

Associated with a GFork instance is a master process. When GFork starts, it runs a user-defined master program and opens up bi-directional pipes to it. The master program runs for the lifetime of the GFork daemon. The master is free to do whatever it wants; it is a user-defined program. Some master programs listen on alternative TCP connections to have state remotely injected. Others monitor system resources, such as memory, in order to best share resources. As clients connect to the TCP listener, child processes are forked which then service the client connection. Bi-directional pipes are opened up to the child processes as well. These pipes allow for communication between the master program and all child processes. The master program and the child programs have their own protocol for information exchange over these links. GFork is just a framework for safely and quickly creating these links.

2. Use Cases

The creation of GFork was motivated by the Globus GridFTP server. GridFTP can be run as a striped server where there is a frontend and several backends. The backends run in tandem to transfer files faster by tying together many NICs. The frontend is the contact point for the client where transfer requests are made. When the frontend is run out of inetd, the list of possible backends must be statically configured. Unfortunately, backends tend to come and go. Sometimes backends fail, and sometimes backends are added to a pool. We needed a way to have a [fixme good synopsis: dynamic pool of backends for use in live transfers]. To accomplish this we created GFork.

3. Configuration

A major difference between GFork configuration and xinetd is that GFork only runs one service per instance, where xinetd runs many services per instance all associated with many different ports. GFork takes a single configuration file and handles a single service. If there is demand, GFork will be enhanced to handle many services in the way that xinetd does.

Running the globus-gridftp-server under GFork is almost identical to running it under xinetd. First, you need a configuration file:

```
service gridftp2
{
env += GLOBUS_LOCATION=<path to GL>
env += LD_LIBRARY_PATH=<path to GL>/lib
server = <path to GL>/sbin/globus-gridftp-server
server_args = -i
server_args += -d ALL -l <path to GL>/var/gridftp.log
port = 5000
}
```

That portion is identical to xinetd. In fact, an existing xinetd configuration file should work.

When running GridFTP out of GFork, the server should be run with a master program. The master program provides enhanced functionality such as dynamic backend registration for striped servers, managed system memory pools and internal data monitoring for both striped and non-striped servers.

To run with a master program, the following two lines are needed in the config file.

```
master = <path to GL>/libexec/gfs-gfork-master
master_args = <options>
```

These last two options relate to the master program and work in the same way that `server` and `server_args` do. The first line tells GFork what master program to use (for the GridFTP server, we use **gfs-gfork-master**). The second line provides options to the master program.

The full list of master options are as follows (this is to date only, run the program with `--help` for newer options):

<code>-b --reg-cs <contact string></code>	Contact to the frontend registry. This option makes it a data node.
<code>-df --dn-file <path></code>	Path to a file containing the list of acceptable DNs. Default is system gridmap file.
<code>-G --gsi <bool></code>	Enable or disable GSI. Default is on.
<code>-h --help</code>	Print the help message.
<code>-l --logfile <path></code>	Path to the logfile.
<code>-p --port <int></code>	Port where the server listens for connections.
<code>-s --stripe-count <int></code>	The maximum number of stripes to give to each server. A value of 0 indicates all stripes are available.
<code>-u --update-interval <int></code>	Number of seconds between registration updates.

The following is an example GFork configuration file:

```
service gridftp2
{
instances = 100
env += GLOBUS_LOCATION=/home/bresnaha/Dev/Globus-gfork3/GL
env += LD_LIBRARY_PATH=/home/bresnaha/Dev/Globus-gfork3/GL/lib
server = /home/bresnaha/Dev/Globus-gfork3/GL/sbin/globus-gridftp-server
server_args = -i -aa
server_args += -d ALL -l /home/bresnaha/tst.log
}
```

```
server_args += -dsi remote -repo-count 1
nice = 10
port = 5000
master = /home/bresnaha/Dev/Globus-gfork3/GL/libexec/gfs-gfork-master
master_args = -port 6065 -l /home/bresnaha/master.log -G n
master_args += -dn /home/bresnaha/master_gridmap
}
```

Once you have a configuration file, run GFork with:

```
% gfork -c <path to config file>
```

4. GFork and Striped Servers

As mentioned in [Section 4, “Configuring GridFTP for Cluster-to-Cluster \(or Striped\) data movement”](#), GridFTP offers a powerful enhancement called striped servers. In this mode a GridFTP server is set up with a single frontend and one or more backends. All of the backends work in concert to transfer a single file and thereby achieve high throughput rates. Here we describe how to configure one frontend and multiple backends for use as a striped server with GFork.

4.1. Frontend Configuration

The frontend server described here is run using dynamic backends. We need additional options for both the GridFTP server and the master program. The following lines are added to the config file:

```
server_args += -dsi remote
master_args = -port 8588
master_args += -df <path to gridmap file>
```

The first line is an additional argument to the GridFTP server. It tells the server that it will be operating in split mode (separate frontend and backend processes) and that it will be using the frontend. (Specifically it tells the server to use the 'remote' DSI).

The second line tells the master program on which port it should listen for backend registrations. Backend services can then connect to this port to notify the frontend of their existence. By default, a registration is good for 10 minutes, but a backend is free to refresh its registration. In this way, a frontend is provided with the list of possible backends (stripes) which may be used for a transfer.

The third line provides the master program with a list of authorized DNs. Each line in the file must contain a GSI DN (certificate subject). In order to register, the backend must authenticate and provide its DN. The provided DN is checked against this file. In other words, the file is a list of DNs that may register with the frontend. If the master program is not given a `-df` option and is given the `-G` option, then there is no registration security at all.

4.2. Backend Configuration

Any striped server setup can have more than one backend service. Furthermore, any one computer can run multiple backends. The following explains how to set up a backend server. These steps should be repeated for each needed backend instance.

A backend server may also be run with GFork, it just needs different options for both the GridFTP server and the master program. A sample backend config file is shown here:

```
service gridftp2
{
```

```
env += GLOBUS_LOCATION=<path to GL>
env += LD_LIBRARY_PATH=<path to GL>/lib

server = <path to GL>/sbin/globus-gridftp-server
server_args = -i
server_args += -dn
master = <path to GL>/libexec/gfs-gfork-master
master_args = -b localhost:8588
}
```

Notable additions to this file are:

```
server_args += -dn
master_args = -b localhost:8588
```

The first line tells the GridFTP server that it will be a 'data node', which is another name for a backend.

The second line tells the master program two things, first that it will be a master of a data node, and second what the frontend's registration contact point is. Note that in our example we have a hostname of 'localhost' and a port of '8588'. 8588 is (and must be) the same port that was provided to the frontend's master program in the previous step.

Once the configuration file is complete, run GFork again as follows:

```
% gfork -c <conf file>
```

This will start up the data node and the master program will register itself to the frontend and refresh its registration every 5 minutes (default setting).

5. GFork with Memory Management

Another feature of the GridFTP GFork plugin is memory usage limiting. Under extreme client loads, it is possible that GridFTP servers require more memory than the system has available. Due to a common kernel memory allocation scheme known as optimistic provisioning, this situation can lead to a full consumption of memory resources and thus trigger the out of memory handler. The OOM handler will kill processes in a difficult-to-predict way in order to free up memory. This will leave the system in an unpredictable and unstable state; obviously, this is a situation that we want to avoid.

To control this situation, the GridFTP GFork plugin has a memory limiting option. This will attempt to limit memory usage to a given value or to the maximum amount of RAM in the system. Most of the memory is given to the first few connections, but when the plugin detects that it is overloaded, each session is limited to half the available memory.

To enable this feature, one of two options must be passed to the master program via the `master_args` in the config file:

<code>-m</code>	Limits memory consumption to amount of RAM in the system.
<code>-M <formatted int></code>	Limits memory to the given value.

Another important option should be provided in the GFork config file: `instance`. When a client connects to GFork, a GridFTP server instance is executed. This instance requires a certain amount of RAM. If connections are coming in too fast, this can act as a DOS attack. Limiting the number of allowed simultaneous connections will help the memory management algorithm do its job. This limit is set with:

```
instance = <int>
```

We recommend a value of 100 or $\lfloor \text{RAM} / 2\text{M} \rfloor$, whichever is smaller.

The following is an example of a GFork configuration file with memory limiting enabled:

```
service gridftp2
{
instance = 100
env += GLOBUS_LOCATION=<path to GL>
env += LD_LIBRARY_PATH=<path to GL>/lib
server = <path to GL>/sbin/globus-gridftp-server
server_args = -i
server_args += -dn
master = <path to GL>/libexec/gfs-gfork-master
master_args = -M 512M
}
```

Glossary

some terms not in the docs but wanted in glossary: *scheduler client/server transfer*

C

- client** A process that sends commands and receives responses. Note that in GridFTP, the client may or may not take part in the actual movement of data.
- client/server transfer** In a client/server transfer, there are only two entities involved in the transfer, the client entity and the server entity. We use the term entity here rather than process because in the implementation provided in GT5, the server entity may actually run as two or more separate processes.
- The client will either move data from or to his local host. The client will decide whether or not he wishes to connect to the server to establish the data channel or the server should connect to him (MODE E dictates who must connect).
- If the client wishes to connect to the server, he will send the PASV (passive) command. The server will start listening on an ephemeral (random, non-privileged) port and will return the IP and port as a response to the command. The client will then connect to that IP/Port.
- If the client wishes to have the server connect to him, the client would start listening on an ephemeral port, and would then send the PORT command which includes the IP/Port as part of the command to the server and the server would initiate the TCP connect. Note that this decision has an impact on traversing firewalls. For instance, the client's host may be behind a firewall and the server may not be able to connect.
- Finally, now that the data channel is established, the client will send either the RETR "filename" command to transfer a file from the server to the client (GET), or the STOR "filename" command to transfer a file from the client to the server (PUT).

S

- scheduler** Term used to describe a job scheduler mechanism to which GRAM interfaces. It is a networked system for submitting, controlling, and monitoring the workload of batch jobs in one or more computers. The jobs or tasks are scheduled for execution at a time chosen by the subsystem according to an available policy and availability of resources. Popular job schedulers include Portable Batch System (PBS), Platform LSF, and IBM LoadLeveler.
- server** A process that receives commands and sends responses to those commands. Since it is a server or service, and it receives commands, it must be listening on a port somewhere to receive the commands. Both FTP and GridFTP have IANA registered ports. For FTP it is port 21, for GridFTP it is port 2811. This is normally handled via inetd or xinetd on Unix variants. However, it is also possible to implement a daemon that listens on the specified port. This is described more fully in the Architecture section of the GridFTP Developer's Guide.

T

third party transfers

In the simplest terms, a third party transfer moves a file between two GridFTP servers.

The following is a more detailed, programmatic description.

In a third party transfer, there are three entities involved. The client, who will only orchestrate, but not actually take place in the data transfer, and two servers one of which will be sending data to the other. This scenario is common in Grid applications where you may wish to stage data from a data store somewhere to a super-computer you have reserved. The commands are quite similar to the client/server transfer. However, now the client must establish two control channels, one to each server. He will then choose one to listen, and send it the PASV command. When it responds with the IP/port it is listening on, the client will send that IP/port as part of the PORT command to the other server. This will cause the second server to connect to the first server, rather than the client. To initiate the actual movement of the data, the client then sends the RETR “filename” command to the server that will read from disk and write to the network (the “sending” server) and will send the STOR “filename” command to the other server which will read from the network and write to the disk (the “receiving” server).

See Also [client/server transfer](#).

Index

A

- accessing data
 - non-POSIX data source, 18
 - non-POSIX file data source that has a POSIX interface, 19
- admin scenarios
 - running in daemon mode, 8
- administrative settings, recommended, 12

B

- building and installing
 - general instructions, 1
- building and installing GridFTP
 - only a combination of certain GridFTP elements, 2
 - only a static GridFTP server, 2
 - only GridFTP and Utilities, 1
 - only the GridFTP client, 1
 - only the GridFTP SDK, 2
 - only the GridFTP server, 1

C

- commandline tool
 - globus-gridftp-server, 30
- configuration interface for GridFTP, 4
- configuring GridFTP, 4
 - multicasting, 17
 - overview, 4
 - run with GFork, 17
 - run with UDT for third party transfers, 15
- security
 - anonymous mode, 6
 - gsiftp, 8
 - over SSH, 7
 - username/password, 6
- separation of processes, 16
- split process, 16
- striped servers, 16
- typical, 4
- with DSI, 19

D

- deploying
 - running under inetd or xinetd, 9

E

- errors, 23

G

- globus-gridftp-server, 30

L

- logging, 21

M

- moving files
 - existing FTP, 19

P

- performance, 12

R

- running in daemon mode, 8

T

- testing, 20
- troubleshooting for GridFTP, 22
- tuning, 12

U

- usage statistics for GridFTP, 28

GT 5.0.2 GridFTP: User's Guide

GT 5.0.2 GridFTP: User's Guide

Introduction

The GridFTP User's Guide provides general end user-oriented information.

Table of Contents

1. Managing Files on a Grid (GridFTP Quickstart)	1
1. Building and installing the GridFTP client	1
2. Java Client API Download	1
3. Configuring the GridFTP client	1
4. Basic procedure for using GridFTP (globus-url-copy)	2
5. Using standard FTP clients with GridFTP server	4
6. Advanced Features	4
2. GridFTP Client Tool	8
globus-url-copy	9
globus-url-sync	21
3. Graphical User Interface	24
1. Globus GridFTP GUI (pre-alpha)	24
2. UberFTP	24
4. Troubleshooting	25
1. Error Codes in GridFTP	26
2. Establish control channel connection	28
3. Try running globus-url-copy	29
4. If your server starts... ..	29
5. High latency for GridFTP server connections	30
5. Usage statistics collection by the Globus Alliance	31
1. GridFTP-specific usage statistics	31
Glossary	32
Index	34

List of Figures

2.1. Effect of Parallel Streams in GridFTP	20
--	----

List of Tables

2.1. URL formats	11
2.2. URL formats	22
4.1. GridFTP Errors	27

Chapter 1. Managing Files on a Grid (GridFTP Quickstart)

1. Building and installing the GridFTP client

If the GridFTP client is not installed and that is all you need, follow the instructions [here](#) to build only the GridFTP client.

2. Java Client API Download

GT 5.0 does not include any of the CoG JGlobus Java APIs that were included in the GT4 release series. But, the JGlobus APIs can still be used with the GT5 services. You can get them directly from the CoG JGlobus releases; see the following link:

http://dev.globus.org/wiki/CoG_jglobus

Consider the following when determining which version of CoG JGlobus to use:

- The GRAM development team used CoG JGlobus version 1.6.0 for performance testing.
- The BIRN project used CoG JGlobus version 1.6.0 (plus patches) for GridFTP testing. All patches are included in 1.8.0.
- At the time of the GT 5.0.2 release, 1.8.0 was the recommended version. In general, the latest recommended CoG JGlobus version should be used.

3. Configuring the GridFTP client

3.1. If client is behind a firewall

If the GridFTP *client* is behind a firewall:

1. Contact your network administrator to open up a range of ports (for GridFTP data channel connections) for the incoming connections. If the firewall blocks the outgoing connections, open up a range of ports for outgoing connections as well.

2. Set the environment variable GLOBUS_TCP_PORT_RANGE

```
export GLOBUS_TCP_PORT_RANGE=min,max
```

where min,max specify the port range that you have opened for the incoming connections on the firewall. This restricts the listening ports of the GridFTP client to this range. Recommended range is 1000 (e.g., 50000-51000) but it really depends on how much use you expect.

3. If you have a firewall blocking the outgoing connections and you have opened a range of ports, set the environment variable GLOBUS_TCP_SOURCE_RANGE:

```
export GLOBUS_TCP_PORT_RANGE=min,max
```

where min,max specify the port range that you have opened for the outgoing connections on the firewall. This restricts the outbound ports of the GridFTP client to this range. Recommended range is twice the range used for GLOBUS_TCP_PORT_RANGE, because if parallel TCP streams are used for transfers, the listening port would remain the same for each connection but the connecting port would be different for each connection.

Additional information on Globus Toolkit Firewall Requirements is available [here](#)¹.

3.2. Configuring security

3.2.1. SSH Security

There is no additional configuration required to use GridFTP in conjunction with SSH.

3.2.2. GSI Security

In order to use GSI security for the transfers, you need to obtain and install a user certificate from a certificate authority trusted by the GridFTP servers that you wish to move data in and out of, and configure the client to trust the certificate authority that signed the certificates of the GridFTP server(s)

- [Obtaining user certificates](#)
- [Configuring the client to trust a particular certificate authority](#)
- [Creating a proxy credential](#)

4. Basic procedure for using GridFTP (globus-url-copy)

If you just want the "rules of thumb" on getting started (without all the details), the following options using `globus-url-copy` will normally give acceptable performance:

For a single file transfer:

```
globus-url-copy -vb -tcp-bs 1048576 -p 4 source_url destination_url
```

where:

`-vb` specifies verbose mode and displays:

- number of bytes transferred,
- performance since the last update (currently every 5 seconds), and
- average performance for the whole transfer.

`-tcp-bs` specifies the size (in bytes) of the TCP buffer to be used by the underlying ftp data channels. This is critical to good performance over the WAN.

[How do I pick a value?](#)

¹ <http://www.globus.org/toolkit/security/firewalls/>

-p Specifies the number of parallel data connections that should be used. This is one of the most commonly used options.

How do I pick a value?

For a directory transfer:

```
globus-url-copy -vb -tcp-bs 1048576 -p 4 -r -cd - cc 4 source_url destination_url
```

where:

-vb specifies verbose mode and displays:

- number of bytes transferred,
- performance since the last update (currently every 5 seconds), and
- average performance for the whole transfer.

-tcp-bs specifies the size (in bytes) of the TCP buffer to be used by the underlying ftp data channels. This is critical to good performance over the WAN.

How do I pick a value?

-p Specifies the number of parallel data connections that should be used. This is one of the most commonly used options.

How do I pick a value?

-cc Specifies the number of concurrent FTP connections to use for multiple transfers.

-cd Creates destination directories, if needed.

-r Copies files in subdirectories.

The source/destination URLs will normally be one of the following:

- *file:///path/to/my/file* if you are accessing a file on a file system accessible by the host on which you are running your *client*.
- *gsiftp://hostname/path/to/remote/file* if you are accessing a file from a GridFTP *server*.

4.1. Putting files

One of the most basic tasks in GridFTP is to "put" files, i.e., moving a file from your file system to the server. So for example, if you want to move the file `/tmp/foo` from a file system accessible to the host on which you are running your client to a file name `/tmp/bar` on a host named `remote.machine.my.edu` running a GridFTP server, you would use this command:

```
globus-url-copy -vb -tcp-bs 2097152 -p 4 file:///tmp/foo gsiftp://remote.machine.my.edu/tmp
```



Note

In theory, `remote.machine.my.edu` could be the same host as the one on which you are running your client, but that is normally only done in testing situations.

4.2. Getting files

A `get`, i.e. moving a file from a server to your file system, would just reverse the source and destination URLs:

Tip

Remember `file:` always refers to your file system.

```
globus-url-copy -vb -tcp-bs 2097152 -p 4 gsiftp://remote.machine.my.edu/tmp/bar file:///tm
```

4.3. Third party transfers

Finally, if you want to move a file between two GridFTP servers (a *third party transfer*), both URLs would use `gsiftp:` as the protocol:

```
globus-url-copy -vb -tcp-bs 2097152 -p 4 gsiftp://other.machine.my.edu/tmp/foo gsiftp://re
```

4.4. For more information

If you want more information and details on URLs and the [command line options](#), the [Key Concepts](#) gives basic definitions and an overview of the GridFTP protocol as well as our implementation of it.

5. Using standard FTP clients with GridFTP server

You can use any standard FTP client to communicate with the GridFTP server in the following cases:

- GridFTP server is configured to allow anonymous access or username/password based authentication. Note that this method is not secure but if the data on the GridFTP server is world readable or if the GridFTP server is accessible only to the clients on a trusted internal network, the GridFTP server may be configured to allow anonymous access or username/password based authentication
- Your local system administrator has installed "GridFTP Where There is FTP (GWTFTP)", which acts as a proxy between standard FTP clients and GridFTP servers. More information on GWTFTP is available at [Section 9](#), "[GridFTP Where There Is FTP \(GWTFTP\)](#)".

6. Advanced Features

6.1. Failures and retries

To retry a transfer after a server or network failure, use the `-rst` option. To store the untransferred urls for restarting the transfer after a client failure, use the `-df` option. More information about these options is available [here](#).

For example, `globus-url-copy` can be invoked in a loop for long running transfers, as shown in the script below:

```
#!/bin/sh
STATEFILE=/path/to/statefile;
while [ ! -e $STATEFILE -o -s $STATEFILE ];
do
```

```
globus-url-copy -rst -p 4 -cc 4 -cd -vb -r -df $STATEFILE gsiftp://srchost/srcdirpat
sleep 10;
done;
```

6.2. Bottleneck detection

To determine whether the disk or the network is the bottleneck for the file transfer, use the `-nlb` option. This option uses NetLogger to estimate speeds of disk and network read/write system calls, and attempt to determine the bottleneck component.



Note

In order to use this, the server must be configured to [enable netlogger bottleneck detection](#)².

Example:

```
globus-url-copy -p 2 -nlb -vb gsiftp://host1:port/path/myfile gsiftp://host2:port/path/my
```

This will output something like the following:

```
Total instantaneous throughput:
disk read      = 17022.2 Mbits/s
disk write     = 26630.8 Mbits/s
net read       = 509.0 Mbits/s
net write      = 1053.4 Mbits/s
Bottleneck: network
```

6.3. Using UDT as an alternative transport protocol for TCP

UDT is an application-level protocol that uses UDP for data transport. It addresses some of the limitations of TCP in high-bandwidth and high-delay networks and achieves better performance than TCP on those networks. To use UDT as the underlying transport protocol for the GridFTP transfers, use the `-udt` option.



Note

Note: In order to use this for a third-party transfer, the server must be configured to [enable UDT](#). In order to use this for a client-server transfer, you need a threaded flavor of **globus-url-copy**. See [Switching between threaded and non-threaded flavors](#) for instructions on how to change the flavor.

6.4. Encryption and Integrity protection

The data channel is authenticated by default. Integrity protection and encryption are optional. To integrity protect the data, use the `-dcsafe` option. For encrypted data transfer, use the `-dcpriv` option.

6.5. Striping

The striping functionality enables one to use a set of computers at both ends of a network to transfer data. At both the source and destination ends, the computers need to have a shared file system so that the dataset is accessible from any computer.

² <http://www.cedps.net/index.php/Gridftp-netlogger>

This feature is especially useful in configurations where individual nodes at the source and destination clusters have significantly less network capacity when compared to the network capacity available between the clusters. An example would be clusters with the individual nodes connected by 1 Gbit/s Ethernet connections to a switch that is itself connected to the external network at 10 Gbit/s or faster.

To perform striped data movement, use the `-stripe` option.

 **Note**

This option is useful only if the server is configured for striped data movement.

6.6. Multicasting

To transfer a single file to many destinations in a multicast/broadcast, use the new `-mc` option.

 **Note**

To use this option, the admin must enable multicasting. Click [here](#) for more information.

```
globus-url-copy -vb -tcp-bs 2097152 -p 4 -mc filename source_url
```

The *filename* must contain a line-separated list of destination urls. For example:

```
gsiftp://localhost:5000/home/user/tst1
gsiftp://localhost:5000/home/user/tst3
gsiftp://localhost:5000/home/user/tst4
```

For more flexibility, you can also specify a single destination url on the command line in addition to the urls in the file. Examples are:

```
globus-url-copy -MC multicast.file gsiftp://localhost/home/user/src_file
```

or

```
globus-url-copy -MC multicast.file gsiftp://localhost/home/user/src_file gsiftp://localhos
```

6.6.1. Advanced multicasting options

Along with specifying the list of destination urls in a file, a set of options for each url can be specified. This is done by appending a `?` to the resource string in the url followed by semicolon-separated key value pairs. For example:

```
gsiftp://dst1.domain.com:5000/home/user/tst1?cc=1;tcpbs=10M;P=4
```

This indicates that the receiving host `dst1.domain.com` will use 4 parallel stream, a tcp buffer size of 10 MB, and will select 1 host when forwarding on data blocks. This url is specified in the `-mc` file as described above.

The following is a list of key=value options and their meanings:

`P=integer` The number of parallel streams this node will use when forwarding.

`cc=integer` The number of urls to which this node will forward data.

`tcpbs=format-integer` The TCP buffer size this node will use when forwarding.

`urls=string` The list of urls that must be children of this node when the spanning tree is complete.
`list`

`local_write=boolean:` Determines if this data will be written to a local disk, or just forwarded on to the next hop. This
`y/n` is explained more in the [Network Overlay](#) section.

`subject=string` The DN name to expect from the servers this node is connecting to.

6.6.2. Network Overlay

In addition to allowing multicast, this function also allows for creating user-defined network routes.

If the `local_write` option is set to `n`, then no data will be written to the local disk, the data will only be forwarded on.

If the `local_write` option is set to `n` and is used with the `cc=1` option, the data will be forwarded on to exactly one location.

This allows the user to create a network overlay of data hops using each GridFTP server as a router to the ultimate destination.

Chapter 2. GridFTP Client Tool

Name

globus-url-copy -- Multi-protocol data movement

globus-url-copy

Tool description

globus-url-copy is a scriptable command line tool that can do multi-protocol data movement. It supports gsiftp:// (GridFTP), ftp://, http://, https://, and file:/// protocol specifiers in the URL. For GridFTP, globus-url-copy supports all implemented functionality. Versions from GT 3.2 and later support file globbing and directory moves.

- [Before you begin](#)
- [Command syntax](#)
- [Command line options](#)
 - [Informational options](#)
 - [Utility options](#)
 - [Reliability options](#)
 - [Performance options](#)
 - [Security-related options](#)
- [Default usage](#)
- [MODES in GridFTP](#)
- [If you run a GridFTP server by hand](#)
- [How do I choose a value for the TCP buffer size \(-tcp-bs\) option?](#)
- [How do I choose a value for the parallelism \(-p\) option?](#)
- [Limitations](#)
- [Interactive clients for GridFTP](#)

Before you begin

Important

To use gsiftp:// and https:// protocols in globus-url-copy, you must have a [certificate](#). However, you may use ftp://, http:// or sshftp:// protocols without a certificate.

1. First, as with all things Grid, you *must* have a valid proxy certificate to run globus-url-copy in certain protocols (gsiftp:// and https://, as noted above). If you are using ftp://, http:// or sshftp:// protocols, you may skip ahead to [Command syntax](#)

If you do not have a certificate, you must [obtain one](#).

If you are doing this for testing in your own environment, the [SimpleCA](#) provided with the Globus Toolkit should suffice.

If not, you must contact the Virtual Organization (VO) with which you are associated to find out whom to ask for a certificate.

One common source is the [DOE Science Grid CA](#)¹, although you must confirm whether or not the resources you wish to access will accept their certificates.

Instructions for proper installation of the certificate should be provided from the source of the certificate.

Please note when your certificates expire; they will need to be renewed or you may lose access to your resources.

- Now that you have a certificate, you must generate a temporary proxy. Do this by running:

```
grid-proxy-init
```

Further documentation for **grid-proxy-init** can be found [here](#).

- You are now ready to use **globus-url-copy**! See the following sections for syntax and command line options and other considerations.

Command syntax

The basic syntax for **globus-url-copy** is:

```
globus-url-copy [optional command line switches] Source_URL Destination_URL
```

where:

[optional command line switches]	See Command line options below for a list of available options.
<i>Source_URL</i>	Specifies the original URL of the file(s) to be copied. If this is a directory, all files within that directory will be copied.
<i>Destination_URL</i>	Specifies the URL where you want to copy the files. If you want to copy multiple files, this must be a directory.



Note

Any url specifying a directory must end with */*.

URL prefixes

Versions from GT 3.2 and later support the following URL prefixes:

- file://** (on a local machine only)
- ftp://**
- gsiftp://**
- http://**

¹ <http://www.doe grids.org/pages/cert-request.htm>

- `https://`

Versions from GT 4.2 and later support the following URL prefix (in addition to the above-mentioned URL prefixes):

- `sshftp://`



Note

We do *not* provide an interactive client similar to the generic FTP client provided with Linux. See the [Interactive Clients](#) section below for information on an interactive client developed by NCSA/NMI/TeraGrid.

URL formats

URLs can be any valid URL as defined by RFC 1738 that have a [protocol](#) we support. In general, they have the following format: `protocol://host:port/path`.



Note

If the path ends with a trailing / (i.e. `/path/to/directory/`) it will be considered to be a directory and all files in that directory will be moved. If you want a recursive directory move, you need to add the `-r/-recurse` switch described below.

Table 2.1. URL formats

<code>gsiftp://myhost.mydomain.com:2812/data/foo.dat</code>	Fully specified.
<code>http://myhost.mydomain.com/mywebpage/default.html</code>	Port is not specified; therefore, GridFTP uses protocol default (in this case, 80).
<code>file:///foo.dat</code>	Host is not specified; therefore, GridFTP uses your local host. Port is not specified; therefore, GridFTP uses protocol default (in this case, 80).
<code>file:/foo.dat</code>	This is also valid but is not recommended because, while many servers (including ours) accept this format, it is <i>not</i> RFC conformant and is not recommended.



Important

For GridFTP (`gsiftp://`) and FTP (`ftp://`), it is legal to specify a user name and password in the the URL as follows:

```
gsiftp://myname:[mypassword]@myhost.mydomain.com/foo.dat
```

If you are using GSI security, then you may specify the username (but you may *not* include the `:` or the password) and the grid-mapfile will be searched to see if that is a valid account mapping for your distinguished name (DN). If it is found, the [server](#) will setuid to that account. If not, it will fail. It will **NOT** fail back to your default account.

If you are using anonymous FTP, the username *must* be one of the usernames listed as a valid anonymous name and the password can be anything.

If you are using password authentication, you must specify both your username and password. **THIS IS HIGHLY DISCOURAGED, AS YOU ARE SENDING YOUR PASSWORD IN THE CLEAR ON THE NETWORK.** This is worse than no security; it is a false illusion of security.

Command line options

Informational Options

-help -usage	Prints help.
-version	Prints the version of this program.
-versions	Prints the versions of all modules that this program uses.
-q -quiet	Suppresses all output for successful operation.
-vb -verbose	During the transfer, displays: <ul style="list-style-type: none">• number of bytes transferred,• performance since the last update (currently every 5 seconds), and• average performance for the whole transfer.
-dbg -debugftp	Debugs FTP connections and prints the entire control channel protocol exchange to STDERR. Very useful for debugging. Please provide this any time you are requesting assistance with a globus-url-copy problem.
-list <url>	This option will display a directory listing for the given url.
-nl-bottleneck -nlb	This option uses NetLogger to estimate speeds of disk and network read/write system calls, and attempt to determine the bottleneck component.



Note

In order to use this, the server must be configured to [enable netlogger bottleneck detection](#)².

Utility Ease of Use Options

-a -ascii	Converts the file to/from ASCII format to/from local file format.
-b -binary	Does not apply any conversion to the files. This option is turned on by default.
-cd -create-dest	Create destination directories, if needed
-f <i>filename</i>	Reads a list of URL pairs from a filename. Each line should contain: <i>sourceURL destURL</i>

² <http://www.cedps.net/index.php/Gridftp-netlogger>

Enclose URLs with spaces in double quotes ("). Blank lines and lines beginning with the hash sign (#) will be ignored.

-r | -recurse

Copies files in subdirectories.

-notpt | -no-third-party-transfers

Turns third-party transfers off (on by default).

Site firewall and/or software configuration may prevent a connection between the two servers (a *third party transfer*). If this is the case, globus-url-copy will "relay" the data. It will do a GET from the source and a PUT to the destination.

This obviously causes a performance penalty but will allow you to complete a transfer you otherwise could not do.

Reliability Options

-rst | -restart

Restarts failed FTP operations.

-rst-retries <retries>

Specifies the maximum number of times to retry the operation before giving up on the transfer.

Use 0 for infinite.

The default value is 5.

-rst-interval <seconds>

Specifies the interval in seconds to wait after a failure before retrying the transfer.

Use 0 for an exponential backoff.

The default value is 0.

-rst-timeout <seconds>

Specifies the maximum time after a failure to keep retrying.

Use 0 for no timeout.

The default value is 0.

-df <filename> | -dumpfile <filename>

Specifies path to the file where untransferred urls will be saved for later restarting. The resulting file is the same format as the -f input file. If the file exists, it will be read and all other url input will be ignored.

-stall-timeout | -st <seconds>

Specifies how long before cancelling/restarting a transfer with no data movement. Set to 0 to disable. Default is 600 seconds.

Performance Options

-tcp-bs <size> | -tcp-buffer-size <size>

Specifies the size (in bytes) of the TCP buffer to be used by the underlying ftp data channels.



Important

This is critical to good performance over the WAN.

[How do I pick a value?](#)

`-p <parallelism> | -parallel <parallelism>` Specifies the number of parallel data connections that should be used.



Note

This is one of the most commonly used options.

How do I pick a value?

`-bs <block size> | -block-size <block size>` Specifies the size (in bytes) of the buffer to be used by the underlying transfer methods.

`-pp` **(New starting with GT 4.1.3)** Allows pipelining. GridFTP is a command response protocol. A client sends one command and then waits for a "Finished response" before sending another. Adding this overhead on a per-file basis for a large data set partitioned into many small files makes the performance suffer. Pipelining allows the client to have many outstanding, unacknowledged transfer commands at once. Instead of being forced to wait for the "Finished response" message, the client is free to send transfer commands at any time.

`-mc filename source_url` > Transfers a single file to many destinations. Filename is a line-separated list of destination urls. For more information on this option, click [here](#).

Multicasting must be enabled for use on the server side.



Warning

This option is EXPERIMENTAL.

`-concurrency | -cc` Specifies the number of concurrent FTP connections to use for multiple transfers.

`-udt` Uses UDT, a reliable UDP-based transport protocol, for data transfers.

`-fast` Recommended when using GridFTP servers. Use MODE E for all data transfers, including reusing data channels between list and transfer operations.

Note: In order to use this option, the server must be configured to use UDT. For third party transfers, no change is required on the client side. For client-server transfers, you need the threaded flavor of the client. Refer to [Switching between threaded and non-threaded flavors](#) for information on how to switch between threaded and non-threaded flavors of globus-url-copy.

Security Related Options

`-s <subject> | -subject <subject>` Specifies a subject to match with both the source and destination servers.



Note

Used when the server does not have access to the host certificate (usually when you are running the server as a user). See [the section called "If you run a GridFTP server by hand..."](#).

`-ss <subject> | -source-subject <subject>` Specifies a subject to match with the source server.

**Note**

Used when the server does not have access to the host certificate (usually when you are running the server as a user). See [the section called “If you run a GridFTP server by hand...”](#).

`-ds <subject> | -dest-subject <subject>`

Specifies a subject to match with the destination server.

**Note**

Used when the server does not have access to the host certificate (usually when you are running the server as a user). See [the section called “If you run a GridFTP server by hand...”](#).

`-nodcau | -no-data-channel-authentication`

Turns off data channel authentication for FTP transfers (the default is to authenticate the data channel).

**Warning**

We do *not* recommend this option, as it is a security risk.

`-dcsafe | -data-channel-safe`

Sets data channel protection mode to SAFE.

Otherwise known as *integrity* or *checksumming*.

Guarantees that the data channel has not been altered, though a malicious party may have observed the data.

**Warning**

Rarely used as there is a substantial performance penalty.

`-dcpriv | -data-channel-private`

Sets data channel protection mode to PRIVATE.

The data channel is encrypted and checksummed.

Guarantees that the data channel has not been altered and, if observed, it won't be understandable.

**Warning**

VERY rarely used due to the VERY substantial performance penalty.

Advanced Options


`-stripe`



Enables striped transfers on supported servers.

`-striped-block-size | -sbs`

Sets layout mode and blocksize for striped transfers.

If not set, the server defaults will be used.

	If set to 0, partitioned mode will be used.
	If set to >0, blocked mode will be used, with this setting used as the blocksize.
-t <transfer time in seconds>	Runs the transfer for the specified number of seconds and then ends. Useful for performance testing or forced restart loops.
-ipv6	Uses ipv6 when available.
 Warning	
This option is EXPERIMENTAL. Use at your own risk.	
-dp -delayed-pasv	Enables delayed passive.
-g2 -gridftp2	Uses GridFTP v2 protocol enhancements when possible.
-mn -module-name <gridftp storage module name>	Specifies the backend storage module to use for both the source and destination in a GridFTP transfer.
-mp -module-parameters <gridftp storage module parameters>	Specifies the backend storage module arguments to use for both the source and destination in a GridFTP transfer.
-smn -src-module-name <gridftp storage module name>	Specifies the backend storage module to use for the source file in a GridFTP transfer.
-smp -src-module-parameters <gridftp storage module parameters>	Specifies the backend storage module arguments to use for the source file in a GridFTP transfer.
-dmn -dst-module-name <gridftp storage module name>	Specifies the backend storage module to use for the destination file in a GridFTP transfer.
-dmp -dst-module-parameters <gridftp storage module parameters>	Specifies the backend storage module arguments to use for the destination file in a GridFTP transfer.
-aa -authz-assert <authorization assertion file>	Uses the assertions in the specified file to authorize access to both the source and destination servers.
-saa -src-authz-assert <authorization assertion file>	Uses the assertions in the specified file to authorize access to the source server.
-daa -dst-authz-assert <authorization assertion file>	Uses the assertions in the specified file to authorize access to the destination server.
-cache-aa -cache-authz-assert	Caches the authorization assertion for subsequent transfers.
-cache-saa -cache-src-authz-assert	Caches the source authorization assertion for subsequent transfers.
-cache-daa -cache-dst-authz-assert	Caches the destination authorization assertion for subsequent transfers.
-nl-bottleneck -nlb	Uses NetLogger to estimate speeds of disk and network read/write system calls, and attempt to determine the bottleneck component.
	Note: In order to use this, the server must be configured to enable netlogger bottleneck detection.

<code>-src-pipe -SP <command line></code>	Sets the source end of a remote transfer to use piped-in input with the given command line.
	 Warning Do not use with the <code>-fsstack</code> option.
<code>-dst-pipe -DP <command line></code>	Sets the destination end of a remote transfer to write data to then standard input of the program run via the given command line.
	 Warning Do not use with the <code>-fsstack</code> option.
<code>-pipe <command line></code>	Sets both <code>-src-pipe</code> and <code>-dst-pipe</code> to the same value.
<code>-dcstack -data-channel-stack</code>	Specifies the XIO driver stack for the network on both the source and the destination. Both must be GridFTP servers.
<code>-fsstack -file-system-stack</code>	Specifies the XIO driver stack for the disk on both the source and the destination. Both must be GridFTP servers.
<code>-src-dcstack -source-data-channel-stack</code>	Specifies the XIO driver stack for the network on the source GridFTP server.
<code>-src-fsstack -source-file-system-stack</code>	Specifies the XIO driver stack for the disk on the source GridFTP server.
<code>-dst-dcstack -dest-data-channel-stack</code>	Specifies the XIO driver stack for the network on the destination GridFTP server.
<code>-dst-fsstack -dest-file-system-stack</code>	Specifies the XIO driver stack for the disk on the destination GridFTP server.
<code>-cred <path to credentials or proxy file>, -src-cred -sc <path to credentials or proxy file>, -dst-cred -dc <path to credentials or proxy file></code>	Specifies the credentials to use for source, destination, or both FTP connections.
<code>-af <filename> -alias-file <filename></code>	Specifies a file that maps logical host aliases to lists of physical hosts. When used with multiple concurrent connections, each connection uses the next host in the list. Each line should either be an alias (noted with the <code>@</code> symbol), or a <code>hostname[:port]</code> . Currently, only the aliases <code>@source</code> and <code>@destination</code> are valid, and they are used for every source or destination url.

Synchronization Options

<code>-sync</code>	Only transfer files where the destination does not exist or differs from the source. <code>-sync-level</code> controls how to determine if files differ.
<code>-sync-level <number></code>	Choose criteria for determining if files differ when performing a sync transfer. Level 0 will only transfer if the destination does not exist. Level 1 will transfer if the size of the destination does not match the size of the source. Level 2 will transfer if the timestamp of the destination is older than the timestamp of the source. Level 3 will

perform a checksum of the source and destination and transfer if the checksums do not match. The default sync level is 2.

Default globus-url-copy usage

A **globus-url-copy** invocation using the **gsiftp** protocol with no options (i.e., using all the defaults) will perform a transfer with the following characteristics:

- binary
- stream mode (which implies no parallelism)
- host default TCP buffer size
- encrypted and checksummed control channel
- an authenticated data channel

MODES in GridFTP

GridFTP (as well as normal FTP) defines multiple wire protocols, or MODES, for the data channel.

Most normal FTP servers only implement *stream mode* (MODE S), i.e. the bytes flow in order over a single TCP connection. GridFTP defaults to this mode so that it is compatible with normal FTP servers.

However, GridFTP has another MODE, called Extended Block Mode, or *MODE E*. This mode sends the data over the data channel in blocks. Each block consists of 8 bits of flags, a 64 bit integer indicating the offset from the start of the transfer, and a 64 bit integer indicating the length of the block in bytes, followed by a payload of length bytes. Because the offset and length are provided, out of order arrival is acceptable, i.e. the 10th block could arrive before the 9th because you know explicitly where it belongs. This allows us to use multiple TCP channels. If you use the `-p 1` | `-parallelism` option, **globus-url-copy** automatically puts the servers into MODE E.



Note

Putting `-p 1` is not the same as no `-p` at all. Both will use a single stream, but the default will use stream mode and `-p 1` will use MODE E.

If you run a GridFTP server by hand...

If you run a GridFTP server by hand, you will need to explicitly specify the subject name to expect. The subject option provides **globus-url-copy** with a way to validate the remote servers with which it is communicating. Not only must the server trust **globus-url-copy**, but **globus-url-copy** must trust that it is talking to the correct server. The validation is done by comparing host DNs or subjects.

If the GridFTP server in question is running under a host certificate then the client assumes a subject name based on the server's canonical DNS name. However, if it was started under a user certificate, as is the case when a server is started by hand, then the expected subject name must be explicitly stated. This is done with the `-ss`, `-sd`, and `-s` options.

`-ss` Sets the `sourceURL` subject.

`-ds` Sets the `destURL` subject.

`-s` If you use this option alone, it will set both urls to be the same. You can see an example of this usage under the Troubleshooting section.

**Note**

This is an *unusual* use of the client. Most times you need to specify both URLs.

How do I choose a value?

How do I choose a value for the TCP buffer size (`-tcp-bs`) option?

The value you should pick for the TCP buffer size (`-tcp-bs`) depends on how fast you want to go (your bandwidth) and how far you are moving the data (as measured by the Round Trip Time (RTT) or the time it takes a packet to get to the destination and back).

To calculate the value for `-tcp-bs`, use the following formula (this assumes that Mega means 1000^2 rather than 1024^2 , which is typical for bandwidth):

$$-tcp-bs = \text{bandwidth in Megabits per second (Mbs)} * \text{RTT in milliseconds (ms)} * 1000 / 8$$

As an example, if you are using fast ethernet (100 Mbs) and the RTT was 50 ms it would be:

$$-tcp-bs = 100 * 50 * 1000 / 8 = 625,000 \text{ bytes.}$$

So, how do you come up with values for bandwidth and RTT? To determine RTT, use either ping or traceroute. They both list RTT values.

**Note**

You must be on one end of the transfer and ping the other end. This means that if you are doing a third party transfer you have to run the ping or traceroute between the two server hosts, not from your client.

The bandwidth is a little trickier. Any point in the network can be the bottleneck, so you either need to talk with your network engineers to find out what the bottleneck link is or just assume that your host is the bottleneck and use the speed of your network interface card (NIC).

**Note**

The value you pick for `-tcp-bs` limits the top speed you can achieve. You will NOT get bandwidth any higher than what you used in the calculation (assuming the RTT is actually what you specified; it varies a little with network conditions). So, if for some reason you want to limit the bandwidth you get, you can do that by judicious choice of `-tcp-bs` values.

So where does this formula come from? Because it uses the bandwidth and the RTT (also known as the latency or delay) it is called the *bandwidth delay product*. The very simple explanation is this: TCP is a reliable protocol. It must save a copy of everything it sends out over the network until the other end acknowledges that it has been received.

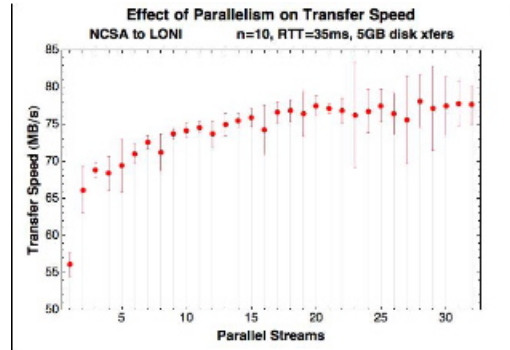
As a simple example, if I can put one byte per second onto the network, and it takes 10 seconds for that byte to get there, and 10 seconds for the acknowledgment to get back (RTT = 20 seconds), then I would need at least 20 bytes of storage. Then, hopefully, by the time I am ready to send byte 21, I have received an acknowledgement for byte 1 and I can free that space in my buffer. If you want a more detailed explanation, try the following links on TCP tuning:

- http://www.psc.edu/networking/perf_tune.html
- <http://www-didc.lbl.gov/TCP-tuning/>
- <http://www.ncne.nlanr.net/research/tcp/>

How do I choose a value for the parallelism (`-p`) option?

For most instances, using 4 streams is a very good rule of thumb. Unfortunately, there is not a good formula for picking an exact answer. The shape of the graph shown here is very characteristic.

Figure 2.1. Effect of Parallel Streams in GridFTP



You get a strong increase in bandwidth, then a sharp knee, after which additional streams have very little impact. Where this knee is depends on many things, but it is generally between 2 and 10 streams. Higher bandwidth, longer round trip times, and more congestion in the network (which you usually can only guess at based on how applications are behaving) will move the knee higher (more streams needed).

In practice, between 4 and 8 streams are usually sufficient. If things look really bad, try 16 and see how much difference that makes over 8. However, anything above 16, other than for academic interest, is basically wasting resources.

Limitations

There are no limitations for `globus-url-copy` in GT 5.0.2.

Interactive clients for GridFTP

The Globus Project does *not* provide an interactive client for GridFTP. Any normal FTP client will work with a GridFTP server, but it cannot take advantage of the advanced features of GridFTP. The interactive clients listed below take advantage of the advanced features of GridFTP.

There is no endorsement implied by their presence here. We make no assertion as to the quality or appropriateness of these tools, we simply provide this for your convenience. We will *not* answer questions, accept bugs, or in any way shape or form be responsible for these tools, although they should have mechanisms of their own for such things.

UberFTP was developed at the NCSA under the auspices of NMI and TeraGrid:

- NCSA Uberftp only download: <http://dims.ncsa.uiuc.edu/set/uberftp/download.html>
- UberFTP User's Guide: <http://dims.ncsa.uiuc.edu/set/uberftp/userdoc.html>

Name

`globus-url-sync` -- Used in conjunction with `globus-url-copy` to synchronize directories.

`globus-url-sync`

Tool description

globus-url-sync is a command line tool which provides a list of files to be transferred, in order to synchronize two directories. It currently supports `gsiftp://` (GridFTP) and `sshftp://` protocol specifiers in the URL.

The program **globus-url-sync** compares two endpoints, using GridFTP, and prints a list of GSI file transfers that should be performed using **globus-url-copy**.

The current implementation of **globus-url-sync** supports very basic features for directory synchronization. It includes comparators for existence checks, file size checks, modification timestamp checks, but not checksum comparison.

- [Before you begin](#)
- [Command syntax](#)
- [Command line options](#)
- [Limitations](#)

Before you begin

1. First, as with **globus-url-copy**, you must have a valid proxy certificate to run **globus-url-sync** using protocol "`gsiftp://`".

If you do not have a certificate, you must [obtain one](#).

If you are doing this for testing in your own environment, the [SimpleCA](#) provided with the Globus Toolkit should suffice.

If not, you must contact the Virtual Organization (VO) with which you are associated to find out whom to ask for a certificate.

One common source is the [DOE Science Grid CA](#)¹, although you must confirm whether or not the resources you wish to access will accept their certificates.

Instructions for proper installation of the certificate should be provided from the source of the certificate.

Please note when your certificates expire; they will need to be renewed or you may lose access to your resources.

2. Now that you have a certificate, you must generate a temporary proxy. Do this by running:

```
grid-proxy-init
```

Further documentation for **grid-proxy-init** can be found [here](#).

¹ <http://www.doe grids.org/pages/cert-request.htm>

Command syntax

The basic syntax for **globus-url-sync** is:

```
globus-url-sync [optional command line switches] Source_URL Destination_URL
```

where:

[optional command line switches]	See Command line options below for a list of available options.
<i>Source_URL</i>	Specifies the original URL of the file(s) to be copied. If this is a directory, all files within that directory that need to be synchronized will be listed.
<i>Destination_URL</i>	Specifies the URL where you want to copy the files. The types of the source and the destination must match. In other words, if the source is a file, the destination must be a file, and if the source is a directory, the destination must be a directory.

Note

Any url specifying a directory must end with `/`.

URL prefixes

The following URL prefixes are supported:

- **gsiftp://**
- **sshftp://**

URL formats

URLs can be any valid URL as defined by RFC 1738 that have a [protocol](#) we support. In general, they have the following format: ***protocol://host:port/path***.

Note

If the path ends with a trailing `/` (i.e. `/path/to/directory/`) it will be considered to be a directory and all files in that directory that are not synchronized will be listed.

Table 2.2. URL formats

<code>gsiftp://myhost.mydomain.com//tmp/file1</code>	File name, absolute path.
<code>gsiftp://myhost.mydomain.com/~file1</code>	File name, absolute path.
<code>gsiftp://myhost.mydomain.com/file1</code>	File name, relative path.
<code>gsiftp://myhost.mydomain.com//tmp/dir1/</code>	Directory, absolute path.

Command line options

-help -usage	Print help text.
-version	Print the version of this program.
-d -debug -v -verbose	Print additional detail.
-r -recursive-dir-copy	Output directory names, when an entire directory is to be copied recursively.
-n -newer	File is to be transferred, if the source timestamp is newer than the destination timestamp.
-o -older	File is to be transferred, if the source timestamp is older than the destination timestamp.
-s -size	File is to be transferred, if the sizes of the source and the destination are not the same.

Limitations

- This is an early version of **globus-url-sync**. In the event that unexpected results are returned, please re-run the command with the **-verbose** option.
- **globus-url-copy** should be invoked with the **-r** (copy files in subdirectories) **-cd** (create directory) options, so that directories can be copied recursively (for "globus-url-sync -r"), and so that directories at the destination can be created.
- Authentication errors may be erroneously be reported as though a file is missing.
- Order of options does not currently effect order in which matching criteria are evaluated.

Chapter 3. Graphical User Interface

1. Globus GridFTP GUI (pre-alpha)

The Globus GridFTP GUI is Java web start application. Users can get it by clicking a link; the program will be downloaded and started automatically. A pre-alpha version of the GUI is available now.

- [Download the GUI client](#)¹

The GUI client provides an easy-to-use interface for connecting to GridFTP servers and transferring files. It has the following features:

- Allows you to browse the local file system and transfer files and directories between the local system and remote GridFTP servers and between two remote GridFTP servers (third-party transfers).
- Supports file system operations such as creating, deleting and renaming files and directories.

Prerequisites:

- JDK 1.5.0+

Supported Platforms:

- Windows
- Linux
- MAC

The GUI provides two ways for generating a proxy credential required for the data transfer:

1. Creating a proxy credential using a locally stored key pair.
2. Obtaining a proxy from a MyProxy Server. For more information about MyProxy, please visit: <http://myproxy.ncsa.uiuc.edu/>.

[A demo of using the GridFTP GUI is available here](#)². Open the file ending in .htm with any browser with the Flash plugin to start the Flash demo - then just click the green arrows to progress through each screen.

2. UberFTP

NCSA, as part of their TeraGrid activity, produces a text based interactive client called UberFTP, which you may want to check out. See [the section called “Interactive clients for GridFTP”](#) for more information.

¹ <http://www-unix.globus.org/cog/demo/ogce/ftp.jnlp>

² [../demo.tar.gz](#)

Chapter 4. Troubleshooting

If you are having problems using the GridFTP *server*, try the steps listed below. If you have an error, try checking the server logs if you have access to them. By default, the server logs to stderr, unless it is running from inetd, or its execution mode is detached, in which case logging is disabled by default.

The command line options `-d`, `-log-level`, `-L` and `-logdir` can affect where logs will be written, as can the configuration file options `log_single` and `log_unique`. See the [globus-gridftp-server\(1\)](#) for more information on these and other configuration options.

For a list of common errors in GT, see [Error Codes](#).

1. Error Codes in GridFTP

Table 4.1. GridFTP Errors

Error Code	Definition	Possible Solutions
<p>globus_ftp_client: the server responded with an error 530 530-globus_xio: Authentication Error 530-OpenSSL Error: s3_srvr.c:2525: in library: SSL routines, function SSL3_GET_CLIENT_CERTIFICATE: no certificate returned 530-globus_gsi_callback_module: Could not verify credential 530-globus_gsi_callback_module: Can't get the local trusted CA certificate: Untrusted self-signed certificate in chain with hash d1b603c3 530 End.</p>	<p>This error message indicates that the GridFTP server doesn't trust the certificate authority (CA) that issued your certificate.</p>	<p>You need to ask the GridFTP server administrator to install your CA certificate chain in the GridFTP server's trusted certificates directory.</p>
<p>globus_ftp_control: gss_init_sec_context failed OpenSSL Error: s3_clnt.c:951: in library: SSL routines, function SSL3_GET_SERVER_CERTIFICATE: certificate verify failed globus_gsi_callback_module: Could not verify credential globus_gsi_callback_module: Can't get the local trusted CA certificate: Untrusted self-signed certificate in chain with hash d1b603c3</p>	<p>This error message indicates that your local system doesn't trust the certificate authority (CA) that issued the certificate on the resource you are connecting to.</p>	<p>You need to ask the resource administrator which CA issued their certificate and install the CA certificate in the local trusted certificates directory.</p>

Error Code	Definition	Possible Solutions
530-globus_xio: Authentication Error 530-globus_gsi_callback_module: Could not verify credential 530-globus_gsi_callback_module: Could not verify credential 530-globus_gsi_callback_module: Invalid CRL: The available CRL has expired 530 End.	This error message indicates one of the following: Certificate Revocation List (CRL) for the source or destination server CA at the client has expired or CRL for client CA has expired at source or destination server or CRL for source (destination) server CA has expired at destination (source) server. CRL is a file {CA_hash}.r0 in /etc/grid-security/certificates or \${USER_HOME}/.globus/certificates or \${X509_CERT_DIR}	The tool available at http://dist.eu-gridpma.info/distribution/util/fetch-crl/ can be run in a crontab to keep the CRLs up to date.

2. Establish control channel connection

Verify that you can establish a control channel connection and that the server has started successfully by telnetting to the port on which the server is running:

```
% telnet localhost 2811
      Trying 127.0.0.1...
      Connected to localhost.
      Escape character is '^]'.
      220 GridFTP Server mldev.mcs.anl.gov 2.0 (gcc32dbg, 1113865414-1) ready.
```

If you see anything other than a 220 banner such as the one above, the server has not started correctly.

Verify that there are no configuration files being unexpectedly loaded from /etc/grid-security/gridftp.conf or \$GLOBUS_LOCATION/etc/gridftp.conf. If those files exist, and you did not intend for them to be used, rename them to .save, or specify -c none on the command line and try again.

If you can log into the machine where the server is, try running the server from the command line with only the -s option:

```
$GLOBUS_LOCATION/sbin/globus-gridftp-server -s
```

The server will print the port it is listening on:

```
Server listening at gridftp.mcs.anl.gov:57764
```

Now try and telnet to that port. If you still do not get the banner listed above, something is preventing the socket connection. Check firewalls, tcp-wrapper, etc.

If you now get a correct banner, add -p 2811 (you will have to disable (x)inetd on port 2811 if you are using them or you will get port already in use):

```
$GLOBUS_LOCATION/sbin/globus-gridftp-server -s -p 2811
```

Now telnet to port 2811. If this does not work, something is blocking port 2811. Check firewalls, tcp-wrapper, etc.

If this works correctly then re-enable your normal server, but remove all options but -i, -s, or -S.

Now telnet to port 2811. If this does not work, something is wrong with your service configuration. Check /etc/services and (x)inetd config, have (x)inetd restarted, etc.

If this works, begin adding options back one at a time, verifying that you can telnet to the server after each option is added. Continue this till you find the problem or get all the options you want.

At this point, you can establish a control connection. Now try running `globus-url-copy`.

3. Try running `globus-url-copy`

Once you've verified that you can establish a control connection, try to make a transfer using `globus-url-copy`.

If you are doing a *client*/server transfer (one of your URLs has `file:` in it) then try:

```
globus-url-copy -vb -dbg gsiftp://host.server.running.on/dev/zero file:///dev/null
```

This will run until you control-c the transfer. If that works, reverse the direction:

```
globus-url-copy -vb -dbg file:///dev/zero gsiftp://host.server.running.on/dev/null
```

Again, this will run until you control-c the transfer.

If you are doing a *third party transfer*, run this command:

```
globus-url-copy -vb -dbg gsiftp://host.server1.on/dev/zero gsiftp://host.server2.on/dev/nu
```

Again, this will run until you control-c the transfer.

If the above transfers work, try your transfer again. If it fails, you likely have some sort of file permissions problem, typo in a file name, etc.

4. If your server starts...

If the server has started correctly, and your problem is with a security failure or gridmap lookup failure, verify that you have security configured properly [here](#).

If the server is running and your client successfully authenticates but has a problem at some other time during the session, please ask for help on gt-user@globus.org¹. When you send mail or submit bugs, please always include as much of the following information as possible:

- Specs on all hosts involved (OS, processor, RAM, etc).
- `globus-url-copy -version`
- `globus-url-copy -versions`
- Output from the telnet test above.
- The actual command line you ran with `-dbg` added. Don't worry if the output gets long.
- Check that you are getting a FQDN and `/etc/hosts` that is sane.
- The server configuration and setup (`/etc/services` entries, `(x)inetd` configs, etc.).
- Any relevant lines from the server logs (not the entire log please).

¹ http://dev.globus.org/wiki/Mailing_Lists

5. High latency for GridFTP server connections

If you run GridFTP servers via Xinetd and notice high latency for connections and/or transfers, check if `/etc/xinetd.conf` or the `gsiftp` service configuration inside `/etc/xinetd.d` is set to log `USERID` as follows:

```
log_on_success += USERID
log_on_failure += USERID
```

Such a configuration tells Xinetd to log the remote user using the method defined in RFC 1413, which causes an ident client to attempt to query the machine that the connection is coming from before the service will run. Even when this succeeds, the response can't be trusted, and more often than not it is rejected or simply dropped (which results in the longest delays) by the remote firewall.

Latency can be reduced by making sure Xinetd does *not* log the `USERID`.

Chapter 5. Usage statistics collection by the Globus Alliance

1. GridFTP-specific usage statistics

The following GridFTP-specific usage statistics are sent in a UDP packet at the end of each transfer, in addition to the standard header information described in the [Usage Stats](#)¹ section.

- Start time of the transfer
- End time of the transfer
- Version string of the server
- TCP buffer size used for the transfer
- Block size used for the transfer
- Total number of bytes transferred
- Number of parallel streams used for the transfer
- Number of stripes used for the transfer
- Type of transfer (STOR, RETR, LIST)
- FTP response code -- Success or failure of the transfer



Note

The client (`globus-url-copy`) does NOT send any data. It is the *servers* that send the usage statistics.

We have made a concerted effort to collect only data that is not too intrusive or private and yet still provides us with information that will help improve and gauge the usage of the GridFTP server. Nevertheless, if you wish to disable this feature for GridFTP only, use the `-disable-usage-stats` option of `globus-gridftp-server`. Note that you can disable transmission of usage statistics globally for all C components by setting "GLOBUS_USAGE_OPTOUT=1" in your environment.

Also, please see our [policy statement](#)² on the collection of usage statistics.

¹ /toolkit/docs/5.0/5.0.2/Usage_Stats.html

² /toolkit/docs/latest-stable/Usage_Stats.html

Glossary

C

client A process that sends commands and receives responses. Note that in GridFTP, the client may or may not take part in the actual movement of data.

E

extended block mode (MODE E) MODE E is a critical GridFTP components because it allows for out of order reception of data. This in turn, means we can send the data down multiple paths and do not need to worry if one of the paths is slower than the others and the data arrives out of order. This enables parallelism and striping within GridFTP. In MODE E, a series of “blocks” are sent over the data channel. Each block consists of:

- an 8 bit flag field,
- a 64 bit field indicating the offset in the transfer,
- and a 64 bit field indicating the length of the payload,
- followed by length bytes of payload.

Note that since the offset and length are included in the block, out of order reception is possible, as long as the receiving side can handle it, either via something like a seek on a file, or via some application level buffering and ordering logic that will wait for the out of order blocks.

S

server A process that receives commands and sends responses to those commands. Since it is a server or service, and it receives commands, it must be listening on a port somewhere to receive the commands. Both FTP and GridFTP have IANA registered ports. For FTP it is port 21, for GridFTP it is port 2811. This is normally handled via inetd or xinetd on Unix variants. However, it is also possible to implement a daemon that listens on the specified port. This is described more fully in the Architecture section of the GridFTP Developer's Guide.

stream mode (MODE S) The only mode normally implemented for FTP is MODE S. This is simply sending each byte, one after another over the socket in order, with no application level framing of any kind. This is the default and is what a standard FTP server will use. This is also the default for GridFTP.

T

third party transfers In the simplest terms, a third party transfer moves a file between two GridFTP servers.

The following is a more detailed, programmatic description.

In a third party transfer, there are three entities involved. The client, who will only orchestrate, but not actually take place in the data transfer, and two servers one of which will be sending data to the other. This scenario is common in Grid applications where you may wish to stage data from a data store somewhere to a super-computer you have reserved. The commands are quite similar to the client/server transfer. However, now the client must establish two control channels, one to each server. He will then choose one to listen, and send it the PASV command. When it responds with the IP/port it is listening on, the client will send that IP/port as part of the PORT command to the other server. This will cause the second server to connect to the first server, rather than the client. To initiate the actual movement of the data, the client then sends the RETR “filename” command to the server that will read from disk and write to the network (the “sending” server) and will send the STOR “filename” command to the other server which will read from the network and write to the disk (the “receiving” server).

See Also [client/server transfer](#).

Index

C

commandline tool
 globus-url-copy, 9
 globus-url-sync, 21

E

errors, 26

G

globus-url-copy, 9
globus-url-sync, 21
GUI information for GridFTP, 24

I

interactive clients
 UberFTP, 20

M

moving files
 basic procedure, 2
 between two GridFTP servers (a third party transfer),
 4
 from a server to your file system, 4
 from your file system to the server, 3
 single file to many destinations, 6
 advanced options, 6
 user-defined network routes, 7

T

troubleshooting for GridFTP, 25

U

usage statistics for GridFTP, 31

GT 5.0.2 GridFTP : Developer's Guide

GT 5.0.2 GridFTP : Developer's Guide

Introduction

This guide contains information of interest to developers working with GridFTP. It provides reference information for application developers, including APIs, architecture, procedures for using the APIs and code samples.

Table of Contents

1. Before you begin	1
1. Feature Summary	1
2. Tested platforms	2
3. Backward compatibility summary	3
4. Technology dependencies	3
2. Usage Scenarios	4
3. Tutorials	5
4. Architecture and design overview	6
1. GridFTP Listening	6
2. GridFTP Transfer	6
3. GridFTP Server and DSIs	6
5. API Summary	9
1. Programming Model Overview	9
2. Component API	9
I. GridFTP Commands	11
globus-url-copy	12
globus-url-sync	24
globus-gridftp-server	27
6. Graphical User Interface	39
1. Globus GridFTP GUI (pre-alpha)	39
2. UberFTP	39
7. Configuring GridFTP	40
1. GridFTP server configuration overview	40
2. Typical configuration	40
3. Firewall requirements	41
4. Configuring Security for GridFTP	42
5. globus-gridftp-server quickstart	46
8. Environment variable interface	48
1. Environment variables for GridFTP	48
9. Debugging	49
10. Troubleshooting	50
1. Error Codes in GridFTP	51
2. Establish control channel connection	53
3. Try running globus-url-copy	54
4. If your server starts... ..	54
5. High latency for GridFTP server connections	55
11. Related Documentation	56
A. Developing DSIs for GridFTP	57
1. DSI Interface	57
2. DSI utility API	57
3. Implementation	57
4. DSI Bones	59
B. GridFTP Multicast	60
1. Architecture	60
2. Globus XIO	60
3. Network Overlay	61
4. Results	61
5. Protocol Details	62
6. Usage	62
Glossary	64
Index	67

List of Figures

1. Effect of Parallel Streams in GridFTP	23
B.1. GridFTP Spanning Tree	60
B.2. From client to server through Internet	61
B.3. Routing data through network via multicast	61
B.4.	62
B.5.	62
B.6.	62

List of Tables

1. URL formats	14
2. URL formats	25
10.1. GridFTP Errors	52

Chapter 1. Before you begin

1. Feature Summary

Features new in GT 5.0.2:

- Improved failure restart capability in globus-url-copy:

A new option to store untransferred urls for later restarting is available. In case of any failures, this option allows users to restart transfers from a checkpoint rather than restarting from scratch. This option can be used for directory transfers, single file transfer or a list of files transfer. See [Section 6.1, “Failures and retries”](#) for more details.

- Stall detection:

It is possible that the transfer hangs due to filesystem errors, network errors or GridFTP server errors. A new option is available in globus-url-copy to specify how long before canceling/restarting a transfer with no data movement. See [the section called “Reliability Options”](#) for more details.

- Client-side host aliasing:

This allows [concurrent transfers](#) to be extended to multiple different hosts rather than multiple connections to the same host, without relying on DNS.

- Initial release of command line tool globus-url-sync.

Features that continue to be supported from previous versions

- GridFTP over UDT
- SSH security for GridFTP control channel
- Running the GridFTP server with GFork GridFTP
- Multicasting / Network overlays
- Netlogger's bottleneck detection for GridFTP transfers
- GSI security: This is the PKI based, de facto standard security system used in Grid applications. Kerberos is also possible but is not supported and can be difficult to use due to divergence in the capabilities of GSI and Kerberos.
- Third-party transfers: Very common in Grid applications, this is where a *client* mediates a transfer between two servers (both likely at remote sites) rather than between the server and itself (called a *client/server transfer*).
- Cluster-to-cluster data movement or Striping: GridFTP can do coordinated data transfer by using multiple computer nodes at the source and destination.
- Partial file access: Regions of a file may be accessed by specifying an offset into the file and the length of the block desired.
- Reliability/restart: The receiving server periodically (the default is 5 seconds, but this can be changed) sends “restart markers” to the client. This marker is a messages specifying what bytes have been successfully written to the disk. If the transfer fails, the client may restart the transfer and provide these markers (or an aggregated equivalent marker), and the transfer will pick up where it left off. This can include “holes” in the file.

- Large file support: All file sizes, lengths, and offsets are 64 bits in length.
- Data channel reuse: Data channel can be held open and reused if the next transfer has the same source, destination, and credentials. This saves the time of connection establishment, authentication, and delegation. This can be a huge performance difference when moving lots of small files.
- Integrated instrumentation (Performance Markers).
- Logging/audit trail (Extensive Logging in the server).
- Parallel transfers (Multiple TCP streams between a pair of hosts).
- TCP Buffer size control (Protocol supports Manual and Automatic; Only Manual Implemented).
- Server-side computation (Extended Retrieve (ERET) / Extended Store (ESTO) commands).
- Based on Standards: RFC 959, RFC 2228, RFC 2389, IETF Draft MLST-16 , GGF GFD.020.

Other Supported Features

- On the client side we provide a scriptable tool called globus-url-copy. This tool can take advantage of all the GridFTP protocol features and can also do protocol translation between FTP, HTTP, HTTPS, and POSIX file IO on the client machine.
- We also provide a set of development libraries and APIs for developers wishing to add GridFTP functionality to their application.

Deprecated Features

- None

2. Tested platforms

Tested platforms for GridFTP

- i386 Linux
- ia64 Linux (TeraGrid)
- AIX 5.2
- Solaris 9
- PA-RISC HP/UX 11.11
- ia64 HP/UX 11.22
- Tru64 Unix
- Mac OS X

While the above list includes platforms on which we have tested GridFTP, it does not imply support for a specific platform. However, we are interested in hearing reports of success or bug reports on any platform.

3. Backward compatibility summary

Protocol changes since GT 5.0.2

- None

API changes since GT 5.0.2

- None

Exception changes since GT 5.0.2

- Not Applicable (GridFTP is not Java-based)

Schema changes since GT 5.0.2

- Not Applicable (GridFTP is not SOAP-based)

4. Technology dependencies

GridFTP depends on the following GT components:

- Non-WS (General) Authentication & Authorization
- C Common Libraries
- XIO

GridFTP depends on the following 3rd party software:

- OpenSSL (version is included in release)

Chapter 2. Usage Scenarios

Do you have a need to move large quantities of data rapidly and reliably to remote locations? Globus GridFTP is a software suite optimized for the gamut of data access issues — from bulk file transfer to the details of getting data out of complex storage systems within sites.

Are you concerned about authenticating and authorizing the users? Globus GridFTP supports various authentication and authorization mechanisms. In fact, it is easy to plugin in different authorization mechanism.

Do you need to move data in and out of a complex storage systems? In addition to the POSIX file systems, Globus GridFTP can move data in and out of HPSS and SRB. Capability to access other storage systems can be easily added by implementing a well-defined pluggable interface called Data Storage Interface (DSI).

Do you need transfer huge volume of data and you do not want to babysit the transfer? Use the GridFTP client 'Reliable File Transfer Service'.

Do you need to protect the data while moving it to the remote location? Globus GridFTP provides support for data integrity and data encryption.

Do you need move your data to remote location that is far away and TCP limits the performance? Globus GridFTP supports UDT as an alternate transport protocol for UDT. GridFTP also allows you to use parallel TCP streams.

Do you want to move data from one location to many locations efficiently? Multicasting capability in Globus GridFTP allows you to do it.

Do you have a network link between source and destination that supports much higher data rate than the data rate supported by individual nodes at either end? GridFTP supports cluster-to-cluster data movement – coordinated data transfer using multiple computer nodes at source and destination.

Do you want to limit the resources used for each client connection on the server node? Globus GridFTP can be configured to run with GFork to limit the resource usage. Customized resource control modules can be plugged in easily.

Globus GridFTP has pervasive use in the e-Science Grid community. The high energy physics community in particular has been a huge user from the start. The Relativistic Heavy Ion Collider (RHIC) community in Brookhaven used Globus GridFTP to sustain 600 megabytes per second of data transfer (from Long Island, New York, to Japan) over 11 days.

Frequent large file transfer demands for the British Broadcasting Corporation (BBC) are met by GridFTP. Typical broadcast hour today requires 280 GB for all pre-processed media streams. "Everything in Gridcast is built using Globus Toolkit," said Terry Harmer, Technical Director at the Belfast e-Science Centre, in an interview (<http://www.globusconsortium.org/journal/20050816/harmer.html>) in '05 with the Globus Consortium Journal. "We use it as a means by which we create, define, and deploy services. We are big users of GridFTP."

Recently, US Department of Energy's Advanced Photon Source user facility at Argonne transferred more than a terabyte of data (partitioned into lots of small files) to Australia using GridFTP at a rate 30 times faster than traditional data transfer mechanisms such as SCP. The Laser Interferometer Gravitational Wave Observatory (LIGO) project moved 1.5 Terabytes of data from University of Wisconsin at Milwaukee, USA to Hannover, Germany at a sustained rate of 80 MB/s.

Chapter 3. Tutorials

There is an online tutorial available at: <http://gridftp.globus.org/tutorials/>.

Chapter 4. Architecture and design overview

GridFTP represents a service that a host is providing. Therefore, the service must be listening on a port waiting for *client* to request access to that service. This is generally handled one of two ways:

- Either an application daemon is running listening for connections, or
- inetd/xinetd is used.

1. GridFTP Listening

The following list describes the process between the service listening for connection and an exchange of data taking place:

1. These services (application daemon or inetd/xinetd) listen for connections.
2. When a connection is received on a "well known" port such as 2811 for GridFTP, inetd does a fork/exec to start up a GridFTP *server* process and then does a Switch User (SU) so that the server is running in a user account rather than as root for security reasons. At this point, the client has established a control channel to the server.
3. The client will then send a series of commands to configure or describe the transfer that it wants to take place.

2. GridFTP Transfer

There are basically four important components of the exchange:

1. The first is security. You must authenticate, and for GridFTP, you must establish encryption on the control channel. The control channel is encrypted by default, though it can be switched off (see the security section for more detail).
2. The second is setup and informational exchanges. The client may specify the type of the file (Binary or ASCII), the *MODE* of the transfer, he might request the size of a file before transferring it, etc..
3. Third, the information and negotiation for the data channel must be done. How this is handled, depends on whether you are doing a *client/server transfer* or *third party transfer*.
4. Finally, a store (STOR), retrieve (RETR), extended store (ESTO) or extended retrieve (ERET) to indicate direction of the transfer and to start data moving.

3. GridFTP Server and DSIs

3.1. GridFTP and DSIs

The following information is helpful if you want to use GridFTP to access data in DSIs (such as HPSS and SRB), and non-POSIX data sources.

Architecturally, the Globus GridFTP *server* can be divided into 3 modules:

- the GridFTP protocol module,

- the (optional) data transform module, and
- the Data Storage Interface (DSI).

In the GT 5.0.2 implementation, the data transform module and the DSI have been merged, although we plan to have separate, chainable, data transform modules in the future.

Note

This architecture does NOT apply to the WU-FTPD implementation (GT3.2.1 and lower).

3.1.1. GridFTP Protocol Module

The GridFTP protocol module is the module that reads and writes to the network and implements the GridFTP protocol. This module should not need to be modified since to do so would make the server non-protocol compliant, and unable to communicate with other servers.

3.1.2. Data Transform Functionality

The data transform functionality is invoked by using the ERET (extended retrieve) and ESTO (extended store) commands. It is seldom used and bears careful consideration before it is implemented, but in the right circumstances can be very useful. In theory, any computation could be invoked this way, but it was primarily intended for cases where some simple pre-processing (such as a partial get or sub-sampling) can greatly reduce the network load. The disadvantage to this is that you remove any real option for planning, brokering, etc., and any significant computation could adversely affect the data transfer performance. Note that the *client* must also support the ESTO/ERET functionality as well.

3.1.3. Data Storage Interface (DSI) / Data Transform module

The Data Storage Interface (DSI) / Data Transform module knows how to read and write to the "local" storage system and can optionally transform the data. We put local in quotes because in a complicated storage system, the storage may not be directly attached, but for performance reasons, it should be relatively close (for instance on the same LAN).

The interface consists of functions to be implemented such as send (get), receive (put), command (simple commands that simply succeed or fail like mkdir), etc..

Once these functions have been implemented for a specific storage system, a client should not need to know or care what is actually providing the data. The server can either be configured specifically with a specific DSI, i.e., it knows how to interact with a single class of storage system, or one particularly useful function for the ESTO/ERET functionality mentioned above is to load and configure a DSI on the fly.

See [Appendix A, Developing DSIs for GridFTP](#) for more information.

3.2. Latest information about HPSS

Last Update: August 2005

Working with Los Alamos National Laboratory and the High Performance Storage System (HPSS) collaboration (<http://www.hpss-collaboration.org>), we have written a Data Storage Interface (DSI) for read/write access to HPSS. This DSI would allow an existing application that uses a GridFTP compliant client to utilize an HPSS data resources.

This DSI is currently in testing. Due to changes in the HPSS security mechanisms, it requires HPSS 6.2 or later, which is due to be released in Q4 2005. Distribution for the DSI has not been worked out yet, but it will *probably* be available from both Globus and the HPSS collaboration. While this code will be open source, it requires underlying HPSS libraries which are NOT open source (proprietary).



Note

This is a purely server side change, the client does not know what DSI is running, so only a site that is already running HPSS and wants to allow GridFTP access needs to worry about access to these proprietary libraries.

3.3. Latest information about SRB

Last Update: August 2005

Working with the SRB team at the San Diego Supercomputing Center, we have written a Data Storage Interface (DSI) for read/write access to data in the Storage Resource Broker (SRB) (<http://www.npaci.edu/DICE/SRB>). This DSI will enable GridFTP compliant clients to read and write data to an SRB server, similar in functionality to the sput/sget commands.

This DSI is currently in testing and is not yet publicly available, but will be available from both the SRB web site ([here](#)) and the Globus web site ([here](#)). It will also be included in the next stable release of the toolkit. We are working on performance tests, but early results indicate that for wide area network (WAN) transfers, the performance is comparable.

When might you want to use this functionality:

- You have existing tools that use GridFTP clients and you want to access data that is in SRB
- You have distributed data sets that have some of the data in SRB and some of the data available from GridFTP servers.

Chapter 5. API Summary

1. Programming Model Overview

The Globus FTP Client library provides a convenient way of accessing files on remote FTP servers. In addition to supporting the basic FTP protocol, the FTP Client library supports several security and performance extensions to make FTP more suitable for Grid applications. These extensions are described in the [GridFTP Protocol document](#)¹.

In addition to protocol support for grid applications, the FTP Client library provides a [plugin architecture](#)² for installing application or grid-specific fault recovery and performance tuning algorithms within the library. Application writers may then target their code toward the FTP Client library and, by simply enabling the appropriate plugins, easily tune their application to run it on a different grid.

All applications which use the Globus FTP Client API must include the header file `globus_ftp_client.h` and activate the `GLOBUS_FTP_CLIENT_MODULE`³.

To use the Globus FTP Client API, one must create an [FTP Client handle](#)⁴. This structure contains:

- context information about FTP operations which are being executed,
- a cache of FTP control and data connections, and
- information about plugins which are being used.

The specifics of the connection caching and plugins are found in the "[Handle Attributes](#)"⁵ section of the API documentation.

Once the handle is created, one may begin transferring files or doing other FTP operations by calling the functions in the "[FTP Operations](#)"⁶ section of the API documentation. In addition to whole-file transfers, the API supports partial file transfers, restarting transfers from a known point, and various FTP directory management commands. All FTP operations may have a set of attributes, defined in the `operationattr` section, associated with them to tune various FTP parameters. The data structures and functions needed to restart a file transfer are described in the "[Restart Markers](#)"⁷ section of the API documentation. For operations which require the user to send to or receive data from an FTP *server* they must call the functions described in the "`globus_ftp_client_data`" section of the manual.

The `globus_ftp_control` library provides low-level services needed to implement FTP clients and servers. The API provided is protocol specific. The data transfer portion of this API provides support for the standard data methods described in the FTP Specification as well as extensions for parallel, striped, and partial data transfer.

2. Component API

- [C Client Library API](#)⁸
- [C Control Library API](#)⁹

¹ <http://www.globus.org/alliance/publications/papers/GFD-R.0201.pdf>

² http://www.globus.org/api/c-globus-5.0.2/globus_ftp_client/html/group_globus_ftp_client_plugins.html

³ http://www.globus.org/api/c-globus-5.0.2/globus_ftp_client/html/group_globus_ftp_client_activation.html

⁴ http://www.globus.org/api/c-globus-5.0.2/globus_ftp_client/html/group_globus_ftp_client_handle.html

⁵ http://www.globus.org/api/c-globus-5.0.2/globus_ftp_client/html/group_globus_ftp_client_handleattr.html

⁶ http://www.globus.org/api/c-globus-5.0.2/globus_ftp_client/html/group_globus_ftp_client_operations.html

⁷ http://www.globus.org/api/c-globus-5.0.2/globus_ftp_client/html/group_globus_ftp_client_restart_marker.html

⁸ http://www.globus.org/api/c-globus-3.9.x/globus_ftp_client/html/index.html

⁹ http://www.globus.org/api/c-globus-3.9.x/globus_ftp_control/html/index.html

For information on the internationalization API, see [Chapter 1, APIs](#).

GridFTP Commands

Name

globus-url-copy -- Multi-protocol data movement

globus-url-copy

Tool description

globus-url-copy is a scriptable command line tool that can do multi-protocol data movement. It supports gsiftp:// (GridFTP), ftp://, http://, https://, and file:/// protocol specifiers in the URL. For GridFTP, globus-url-copy supports all implemented functionality. Versions from GT 3.2 and later support file globbing and directory moves.

- [Before you begin](#)
- [Command syntax](#)
- [Command line options](#)
 - [Informational options](#)
 - [Utility options](#)
 - [Reliability options](#)
 - [Performance options](#)
 - [Security-related options](#)
- [Default usage](#)
- [MODES in GridFTP](#)
- [If you run a GridFTP server by hand](#)
- [How do I choose a value for the TCP buffer size \(-tcp-bs\) option?](#)
- [How do I choose a value for the parallelism \(-p\) option?](#)
- [Limitations](#)
- [Interactive clients for GridFTP](#)

Before you begin

Important

To use gsiftp:// and https:// protocols in globus-url-copy, you must have a [certificate](#). However, you may use ftp://, http:// or sshftp:// protocols without a certificate.

1. First, as with all things Grid, you *must* have a valid proxy certificate to run globus-url-copy in certain protocols (gsiftp:// and https://, as noted above). If you are using ftp://, http:// or sshftp:// protocols, you may skip ahead to [Command syntax](#)

If you do not have a certificate, you must [obtain one](#).

If you are doing this for testing in your own environment, the [SimpleCA](#) provided with the Globus Toolkit should suffice.

If not, you must contact the Virtual Organization (VO) with which you are associated to find out whom to ask for a certificate.

One common source is the [DOE Science Grid CA](#)¹, although you must confirm whether or not the resources you wish to access will accept their certificates.

Instructions for proper installation of the certificate should be provided from the source of the certificate.

Please note when your certificates expire; they will need to be renewed or you may lose access to your resources.

- Now that you have a certificate, you must generate a temporary proxy. Do this by running:

```
grid-proxy-init
```

Further documentation for **grid-proxy-init** can be found [here](#).

- You are now ready to use **globus-url-copy**! See the following sections for syntax and command line options and other considerations.

Command syntax

The basic syntax for **globus-url-copy** is:

```
globus-url-copy [optional command line switches] Source_URL Destination_URL
```

where:

[optional command line switches]	See Command line options below for a list of available options.
<i>Source_URL</i>	Specifies the original URL of the file(s) to be copied. If this is a directory, all files within that directory will be copied.
<i>Destination_URL</i>	Specifies the URL where you want to copy the files. If you want to copy multiple files, this must be a directory.



Note

Any url specifying a directory must end with */*.

URL prefixes

Versions from GT 3.2 and later support the following URL prefixes:

- file://** (on a local machine only)
- ftp://**
- gsiftp://**
- http://**

¹ <http://www.doe grids.org/pages/cert-request.htm>

- **https://**

Versions from GT 4.2 and later support the following URL prefix (in addition to the above-mentioned URL prefixes):

- **sshftp://**



Note

We do *not* provide an interactive client similar to the generic FTP client provided with Linux. See the [Interactive Clients](#) section below for information on an interactive client developed by NCSA/NMI/TeraGrid.

URL formats

URLs can be any valid URL as defined by RFC 1738 that have a [protocol](#) we support. In general, they have the following format: ***protocol://host:port/path***.



Note

If the path ends with a trailing / (i.e. `/path/to/directory/`) it will be considered to be a directory and all files in that directory will be moved. If you want a recursive directory move, you need to add the `-r/ -recurse` switch described below.

Table 1. URL formats

<code>gsiftp://myhost.mydomain.com:2812/data/foo.dat</code>	Fully specified.
<code>http://myhost.mydomain.com/mywebpage/default.html</code>	Port is not specified; therefore, GridFTP uses protocol default (in this case, 80).
<code>file:///foo.dat</code>	Host is not specified; therefore, GridFTP uses your local host. Port is not specified; therefore, GridFTP uses protocol default (in this case, 80).
<code>file:/foo.dat</code>	This is also valid but is not recommended because, while many servers (including ours) accept this format, it is <i>not</i> RFC conformant and is not recommended.



Important

For GridFTP (`gsiftp://`) and FTP (`ftp://`), it is legal to specify a user name and password in the the URL as follows:

```
gsiftp://myname:[mypassword]@myhost.mydomain.com/foo.dat
```

If you are using GSI security, then you may specify the username (but you may *not* include the `:` or the password) and the grid-mapfile will be searched to see if that is a valid account mapping for your distinguished name (DN). If it is found, the *server* will be setuid to that account. If not, it will fail. It will **NOT** fail back to your default account.

If you are using anonymous FTP, the username *must* be one of the usernames listed as a valid anonymous name and the password can be anything.

If you are using password authentication, you must specify both your username and password. **THIS IS HIGHLY DISCOURAGED, AS YOU ARE SENDING YOUR PASSWORD IN THE CLEAR ON THE NETWORK.** This is worse than no security; it is a false illusion of security.

Command line options

Informational Options

-help -usage	Prints help.
-version	Prints the version of this program.
-versions	Prints the versions of all modules that this program uses.
-q -quiet	Suppresses all output for successful operation.
-vb -verbose	During the transfer, displays: <ul style="list-style-type: none"> • number of bytes transferred, • performance since the last update (currently every 5 seconds), and • average performance for the whole transfer.
-dbg -debugftp	<p>Debugs FTP connections and prints the entire control channel protocol exchange to STDERR.</p> <p>Very useful for debugging. Please provide this any time you are requesting assistance with a globus-url-copy problem.</p>
-list <url>	This option will display a directory listing for the given url.
-nl-bottleneck -nlb	This option uses NetLogger to estimate speeds of disk and network read/write system calls, and attempt to determine the bottleneck component.



Note

In order to use this, the server must be configured to [enable netlogger bottleneck detection](#)².

Utility Ease of Use Options

-a -ascii	Converts the file to/from ASCII format to/from local file format.
-b -binary	Does not apply any conversion to the files. This option is turned on by default.
-cd -create-dest	Create destination directories, if needed
-f <i>filename</i>	<p>Reads a list of URL pairs from a filename.</p> <p>Each line should contain:</p> <p><i>sourceURL destURL</i></p>

² <http://www.cedps.net/index.php/Gridftp-netlogger>

Enclose URLs with spaces in double quotes ("). Blank lines and lines beginning with the hash sign (#) will be ignored.

-r | -recurse

Copies files in subdirectories.

-notpt | -no-third-party-transfers

Turns third-party transfers off (on by default).

Site firewall and/or software configuration may prevent a connection between the two servers (a *third party transfer*). If this is the case, globus-url-copy will "relay" the data. It will do a GET from the source and a PUT to the destination.

This obviously causes a performance penalty but will allow you to complete a transfer you otherwise could not do.

Reliability Options

-rst | -restart

Restarts failed FTP operations.

-rst-retries <retries>

Specifies the maximum number of times to retry the operation before giving up on the transfer.

Use 0 for infinite.

The default value is 5.

-rst-interval <seconds>

Specifies the interval in seconds to wait after a failure before retrying the transfer.

Use 0 for an exponential backoff.

The default value is 0.

-rst-timeout <seconds>

Specifies the maximum time after a failure to keep retrying.

Use 0 for no timeout.

The default value is 0.

-df <filename> | -dumpfile <filename>

Specifies path to the file where untransferred urls will be saved for later restarting. The resulting file is the same format as the -f input file. If the file exists, it will be read and all other url input will be ignored.

-stall-timeout | -st <seconds>

Specifies how long before cancelling/restarting a transfer with no data movement. Set to 0 to disable. Default is 600 seconds.

Performance Options

-tcp-bs <size> | -tcp-buffer-size <size>

Specifies the size (in bytes) of the TCP buffer to be used by the underlying ftp data channels.



Important

This is critical to good performance over the WAN.

[How do I pick a value?](#)

-p <parallelism> | -parallel <parallelism> Specifies the number of parallel data connections that should be used.

 **Note**

This is one of the most commonly used options.

How do I pick a value?

-bs <block size> | -block-size <block size> Specifies the size (in bytes) of the buffer to be used by the underlying transfer methods.

-pp **(New starting with GT 4.1.3)** Allows pipelining. GridFTP is a command response protocol. A client sends one command and then waits for a "Finished response" before sending another. Adding this overhead on a per-file basis for a large data set partitioned into many small files makes the performance suffer. Pipelining allows the client to have many outstanding, unacknowledged transfer commands at once. Instead of being forced to wait for the "Finished response" message, the client is free to send transfer commands at any time.

-mc *filename source_url* > Transfers a single file to many destinations. Filename is a line-separated list of destination urls. For more information on this option, click [here](#).

Multicasting must be enabled for use on the server side.

 **Warning**

This option is EXPERIMENTAL.

-concurrency | -cc Specifies the number of concurrent FTP connections to use for multiple transfers.

-udt Uses UDT, a reliable UDP-based transport protocol, for data transfers.

-fast Recommended when using GridFTP servers. Use MODE E for all data transfers, including reusing data channels between list and transfer operations.

Note: In order to use this option, the server must be configured to use UDT. For third party transfers, no change is required on the client side. For client-server transfers, you need the threaded flavor of the client. Refer to Switching between threaded and non-threaded flavors for information on how to switch between threaded and non-threaded flavors of globus-url-copy.

Security Related Options

-s <subject> | -subject <subject> Specifies a subject to match with both the source and destination servers.

 **Note**

Used when the server does not have access to the host certificate (usually when you are running the server as a user). See the section called "If you run a GridFTP server by hand...".

-ss <subject> | -source-subject <subject> Specifies a subject to match with the source server.



Note

Used when the server does not have access to the host certificate (usually when you are running the server as a user). See [the section called “If you run a GridFTP server by hand...”](#).

-ds <subject> | -dest-subject <subject>

Specifies a subject to match with the destination server.



Note

Used when the server does not have access to the host certificate (usually when you are running the server as a user). See [the section called “If you run a GridFTP server by hand...”](#).

-nodcau | -no-data-channel-authentication

Turns off data channel authentication for FTP transfers (the default is to authenticate the data channel).



Warning

We do *not* recommend this option, as it is a security risk.

-dcsafe | -data-channel-safe

Sets data channel protection mode to SAFE.

Otherwise known as *integrity* or *checksumming*.

Guarantees that the data channel has not been altered, though a malicious party may have observed the data.



Warning

Rarely used as there is a substantial performance penalty.

-dcpriv | -data-channel-private

Sets data channel protection mode to PRIVATE.

The data channel is encrypted and checksummed.

Guarantees that the data channel has not been altered and, if observed, it won't be understandable.



Warning

VERY rarely used due to the VERY substantial performance penalty.

Advanced Options


-stripe



Enables striped transfers on supported servers.

-striped-block-size | -sbs

Sets layout mode and blocksize for striped transfers.

If not set, the server defaults will be used.

	If set to 0, partitioned mode will be used.
	If set to >0, blocked mode will be used, with this setting used as the blocksize.
-t <transfer time in seconds>	Runs the transfer for the specified number of seconds and then ends. Useful for performance testing or forced restart loops.
-ipv6	Uses ipv6 when available.
 Warning	
This option is EXPERIMENTAL. Use at your own risk.	
-dp -delayed-pasv	Enables delayed passive.
-g2 -gridftp2	Uses GridFTP v2 protocol enhancements when possible.
-mn -module-name <gridftp storage module name>	Specifies the backend storage module to use for both the source and destination in a GridFTP transfer.
-mp -module-parameters <gridftp storage module parameters>	Specifies the backend storage module arguments to use for both the source and destination in a GridFTP transfer.
-smn -src-module-name <gridftp storage module name>	Specifies the backend storage module to use for the source file in a GridFTP transfer.
-smp -src-module-parameters <gridftp storage module parameters>	Specifies the backend storage module arguments to use for the source file in a GridFTP transfer.
-dmn -dst-module-name <gridftp storage module name>	Specifies the backend storage module to use for the destination file in a GridFTP transfer.
-dmp -dst-module-parameters <gridftp storage module parameters>	Specifies the backend storage module arguments to use for the destination file in a GridFTP transfer.
-aa -authz-assert <authorization assertion file>	Uses the assertions in the specified file to authorize access to both the source and destination servers.
-saa -src-authz-assert <authorization assertion file>	Uses the assertions in the specified file to authorize access to the source server.
-daa -dst-authz-assert <authorization assertion file>	Uses the assertions in the specified file to authorize access to the destination server.
-cache-aa -cache-authz-assert	Caches the authorization assertion for subsequent transfers.
-cache-saa -cache-src-authz-assert	Caches the source authorization assertion for subsequent transfers.
-cache-daa -cache-dst-authz-assert	Caches the destination authorization assertion for subsequent transfers.
-nl-bottleneck -nlb	Uses NetLogger to estimate speeds of disk and network read/write system calls, and attempt to determine the bottleneck component.
	Note: In order to use this, the server must be configured to enable netlogger bottleneck detection.

-src-pipe -SP <command line>	<p>Sets the source end of a remote transfer to use piped-in input with the given command line.</p> <p> Warning</p> <p>Do not use with the <code>-fsstack</code> option.</p>
-dst-pipe -DP <command line>	<p>Sets the destination end of a remote transfer to write data to then standard input of the program run via the given command line.</p> <p> Warning</p> <p>Do not use with the <code>-fsstack</code> option.</p>
-pipe <command line>	Sets both <code>-src-pipe</code> and <code>-dst-pipe</code> to the same value.
-dcstack -data-channel-stack	Specifies the XIO driver stack for the network on both the source and the destination. Both must be GridFTP servers.
-fsstack -file-system-stack	Specifies the XIO driver stack for the disk on both the source and the destination. Both must be GridFTP servers.
-src-dcstack -source-data-channel-stack	Specifies the XIO driver stack for the network on the source GridFTP server.
-src-fsstack -source-file-system-stack	Specifies the XIO driver stack for the disk on the source GridFTP server.
-dst-dcstack -dest-data-channel-stack	Specifies the XIO driver stack for the network on the destination GridFTP server.
-dst-fsstack -dest-file-system-stack	Specifies the XIO driver stack for the disk on the destination GridFTP server.
-cred <path to credentials or proxy file>, -src-cred -sc <path to credentials or proxy file>, -dst-cred -dc <path to credentials or proxy file>	Specifies the credentials to use for source, destination, or both FTP connections.
-af <filename> -alias-file <filename>	Specifies a file that maps logical host aliases to lists of physical hosts. When used with multiple concurrent connections, each connection uses the next host in the list. Each line should either be an alias (noted with the @ symbol), or a hostname[:port]. Currently, only the aliases @source and @destination are valid, and they are used for every source or destination url.

Synchronization Options

-sync	Only transfer files where the destination does not exist or differs from the source. -sync-level controls how to determine if files differ.
-sync-level <number>	Choose criteria for determining if files differ when performing a sync transfer. Level 0 will only transfer if the destination does not exist. Level 1 will transfer if the size of the destination does not match the size of the source. Level 2 will transfer if the timestamp of the destination is older than the timestamp of the source. Level 3 will

perform a checksum of the source and destination and transfer if the checksums do not match. The default sync level is 2.

Default globus-url-copy usage

A **globus-url-copy** invocation using the **gsiftp** protocol with no options (i.e., using all the defaults) will perform a transfer with the following characteristics:

- binary
- stream mode (which implies no parallelism)
- host default TCP buffer size
- encrypted and checksummed control channel
- an authenticated data channel

MODES in GridFTP

GridFTP (as well as normal FTP) defines multiple wire protocols, or MODES, for the data channel.

Most normal FTP servers only implement *stream mode* (MODE S), i.e. the bytes flow in order over a single TCP connection. GridFTP defaults to this mode so that it is compatible with normal FTP servers.

However, GridFTP has another MODE, called Extended Block Mode, or *MODE E*. This mode sends the data over the data channel in blocks. Each block consists of 8 bits of flags, a 64 bit integer indicating the offset from the start of the transfer, and a 64 bit integer indicating the length of the block in bytes, followed by a payload of length bytes. Because the offset and length are provided, out of order arrival is acceptable, i.e. the 10th block could arrive before the 9th because you know explicitly where it belongs. This allows us to use multiple TCP channels. If you use the `-p 1` | `-parallelism` option, **globus-url-copy** automatically puts the servers into MODE E.



Note

Putting `-p 1` is not the same as no `-p` at all. Both will use a single stream, but the default will use stream mode and `-p 1` will use MODE E.

If you run a GridFTP server by hand...

If you run a GridFTP server by hand, you will need to explicitly specify the subject name to expect. The subject option provides **globus-url-copy** with a way to validate the remote servers with which it is communicating. Not only must the server trust **globus-url-copy**, but **globus-url-copy** must trust that it is talking to the correct server. The validation is done by comparing host DNs or subjects.

If the GridFTP server in question is running under a host certificate then the client assumes a subject name based on the server's canonical DNS name. However, if it was started under a user certificate, as is the case when a server is started by hand, then the expected subject name must be explicitly stated. This is done with the `-ss`, `-sd`, and `-s` options.

`-ss` Sets the `sourceURL` subject.

`-ds` Sets the `destURL` subject.

`-s` If you use this option alone, it will set both urls to be the same. You can see an example of this usage under the Troubleshooting section.



Note

This is an *unusual* use of the client. Most times you need to specify both URLs.

How do I choose a value?

How do I choose a value for the TCP buffer size (`-tcp-bs`) option?

The value you should pick for the TCP buffer size (`-tcp-bs`) depends on how fast you want to go (your bandwidth) and how far you are moving the data (as measured by the Round Trip Time (RTT) or the time it takes a packet to get to the destination and back).

To calculate the value for `-tcp-bs`, use the following formula (this assumes that Mega means 1000^2 rather than 1024^2 , which is typical for bandwidth):

$$-tcp-bs = \text{bandwidth in Megabits per second (Mbs)} * \text{RTT in milliseconds (ms)} * 1000 / 8$$

As an example, if you are using fast ethernet (100 Mbs) and the RTT was 50 ms it would be:

$$-tcp-bs = 100 * 50 * 1000 / 8 = 625,000 \text{ bytes.}$$

So, how do you come up with values for bandwidth and RTT? To determine RTT, use either ping or traceroute. They both list RTT values.



Note

You must be on one end of the transfer and ping the other end. This means that if you are doing a third party transfer you have to run the ping or traceroute between the two server hosts, not from your client.

The bandwidth is a little trickier. Any point in the network can be the bottleneck, so you either need to talk with your network engineers to find out what the bottleneck link is or just assume that your host is the bottleneck and use the speed of your network interface card (NIC).



Note

The value you pick for `-tcp-bs` limits the top speed you can achieve. You will NOT get bandwidth any higher than what you used in the calculation (assuming the RTT is actually what you specified; it varies a little with network conditions). So, if for some reason you want to limit the bandwidth you get, you can do that by judicious choice of `-tcp-bs` values.

So where does this formula come from? Because it uses the bandwidth and the RTT (also known as the latency or delay) it is called the *bandwidth delay product*. The very simple explanation is this: TCP is a reliable protocol. It must save a copy of everything it sends out over the network until the other end acknowledges that it has been received.

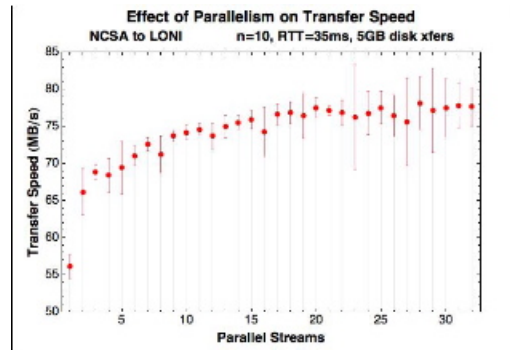
As a simple example, if I can put one byte per second onto the network, and it takes 10 seconds for that byte to get there, and 10 seconds for the acknowledgment to get back (RTT = 20 seconds), then I would need at least 20 bytes of storage. Then, hopefully, by the time I am ready to send byte 21, I have received an acknowledgement for byte 1 and I can free that space in my buffer. If you want a more detailed explanation, try the following links on TCP tuning:

- http://www.psc.edu/networking/perf_tune.html
- <http://www-didc.lbl.gov/TCP-tuning/>
- <http://www.ncne.nlanr.net/research/tcp/>

How do I choose a value for the parallelism (`-p`) option?

For most instances, using 4 streams is a very good rule of thumb. Unfortunately, there is not a good formula for picking an exact answer. The shape of the graph shown here is very characteristic.

Figure 1. Effect of Parallel Streams in GridFTP



You get a strong increase in bandwidth, then a sharp knee, after which additional streams have very little impact. Where this knee is depends on many things, but it is generally between 2 and 10 streams. Higher bandwidth, longer round trip times, and more congestion in the network (which you usually can only guess at based on how applications are behaving) will move the knee higher (more streams needed).

In practice, between 4 and 8 streams are usually sufficient. If things look really bad, try 16 and see how much difference that makes over 8. However, anything above 16, other than for academic interest, is basically wasting resources.

Limitations

There are no limitations for `globus-url-copy` in GT 5.0.2.

Interactive clients for GridFTP

The Globus Project does *not* provide an interactive client for GridFTP. Any normal FTP client will work with a GridFTP server, but it cannot take advantage of the advanced features of GridFTP. The interactive clients listed below take advantage of the advanced features of GridFTP.

There is no endorsement implied by their presence here. We make no assertion as to the quality or appropriateness of these tools, we simply provide this for your convenience. We will *not* answer questions, accept bugs, or in any way shape or form be responsible for these tools, although they should have mechanisms of their own for such things.

UberFTP was developed at the NCSA under the auspices of NMI and TeraGrid:

- NCSA Uberftp only download: <http://dims.ncsa.uiuc.edu/set/uberftp/download.html>
- UberFTP User's Guide: <http://dims.ncsa.uiuc.edu/set/uberftp/userdoc.html>

Name

`globus-url-sync` -- Used in conjunction with `globus-url-copy` to synchronize directories.

`globus-url-sync`

Tool description

globus-url-sync is a command line tool which provides a list of files to be transferred, in order to synchronize two directories. It currently supports `gsiftp://` (GridFTP) and `sshftp://` protocol specifiers in the URL.

The program **globus-url-sync** compares two endpoints, using GridFTP, and prints a list of GSI file transfers that should be performed using **globus-url-copy**.

The current implementation of **globus-url-sync** supports very basic features for directory synchronization. It includes comparators for existence checks, file size checks, modification timestamp checks, but not checksum comparison.

- [Before you begin](#)
- [Command syntax](#)
- [Command line options](#)
- [Limitations](#)

Before you begin

1. First, as with **globus-url-copy**, you must have a valid proxy certificate to run **globus-url-sync** using protocol "`gsiftp://`".

If you do not have a certificate, you must [obtain one](#).

If you are doing this for testing in your own environment, the [SimpleCA](#) provided with the Globus Toolkit should suffice.

If not, you must contact the Virtual Organization (VO) with which you are associated to find out whom to ask for a certificate.

One common source is the [DOE Science Grid CA](#)¹, although you must confirm whether or not the resources you wish to access will accept their certificates.

Instructions for proper installation of the certificate should be provided from the source of the certificate.

Please note when your certificates expire; they will need to be renewed or you may lose access to your resources.

2. Now that you have a certificate, you must generate a temporary proxy. Do this by running:

```
grid-proxy-init
```

Further documentation for **grid-proxy-init** can be found [here](#).

¹ <http://www.doe grids.org/pages/cert-request.htm>

Command syntax

The basic syntax for **globus-url-sync** is:

```
globus-url-sync [optional command line switches] Source_URL Destination_URL
```

where:

[optional command line switches]	See Command line options below for a list of available options.
<i>Source_URL</i>	Specifies the original URL of the file(s) to be copied. If this is a directory, all files within that directory that need to be synchronized will be listed.
<i>Destination_URL</i>	Specifies the URL where you want to copy the files. The types of the source and the destination must match. In other words, if the source is a file, the destination must be a file, and if the source is a directory, the destination must be a directory.

Note

Any url specifying a directory must end with `/`.

URL prefixes

The following URL prefixes are supported:

- **gsiftp://**
- **sshftp://**

URL formats

URLs can be any valid URL as defined by RFC 1738 that have a [protocol](#) we support. In general, they have the following format: ***protocol://host:port/path***.

Note

If the path ends with a trailing `/` (i.e. `/path/to/directory/`) it will be considered to be a directory and all files in that directory that are not synchronized will be listed.

Table 2. URL formats

<code>gsiftp://myhost.mydomain.com//tmp/file1</code>	File name, absolute path.
<code>gsiftp://myhost.mydomain.com/~file1</code>	File name, absolute path.
<code>gsiftp://myhost.mydomain.com/file1</code>	File name, relative path.
<code>gsiftp://myhost.mydomain.com//tmp/dir1/</code>	Directory, absolute path.

Command line options

-help -usage	Print help text.
-version	Print the version of this program.
-d -debug -v -verbose	Print additional detail.
-r -recursive-dir-copy	Output directory names, when an entire directory is to be copied recursively.
-n -newer	File is to be transferred, if the source timestamp is newer than the destination timestamp.
-o -older	File is to be transferred, if the source timestamp is older than the destination timestamp.
-s -size	File is to be transferred, if the sizes of the source and the destination are not the same.

Limitations

- This is an early version of **globus-url-sync**. In the event that unexpected results are returned, please re-run the command with the **-verbose** option.
- **globus-url-copy** should be invoked with the **-r** (copy files in subdirectories) **-cd** (create directory) options, so that directories can be copied recursively (for "globus-url-sync -r"), and so that directories at the destination can be created.
- Authentication errors may be erroneously be reported as though a file is missing.
- Order of options does not currently effect order in which matching criteria are evaluated.

Name

globus-gridftp-server -- Configures the GridFTP Server

globus-gridftp-server

Tool description

globus-gridftp-server configures the GridFTP server using a config file and/or commandline options.



Note

Command line options and configuration file options may both be used, but the command line *overrides* the config file.

The configuration file for the GridFTP *server* is read from the following locations, in the given order. Only the first file found will be loaded:

- Path specified with the `-c <configfile>` command line option.
- `$GLOBUS_LOCATION/etc/gridftp.conf`
- `/etc/grid-security/gridftp.conf`

Options are one per line, with the format:

```
<option> <value>
```

If the value contains spaces, they should be enclosed in double-quotes ("). Flags or boolean options should only have a value of 0 or 1. Blank lines and lines beginning with # are ignored.

For example:

```
port 5000
allow_anonymous 1
anonymous_user bob
banner "Welcome!"
```

Developer notes

The Globus implementation of the GridFTP *server* draws on:

- three IETF RFCs:
 - RFC 959
 - RFC 2228
 - RFC 2389
- an IETF Draft: MLST-16
- the GridFTP protocol specification, which is Global Grid Forum (GGF) Standard GFD.020.

The command line tools and the *client* library completely hide the details of the protocol from the user and the developer. Unless you choose to use the control library, it is not necessary to have a detailed knowledge of the protocol.

Command syntax

The basic syntax for **globus-gridftp-server** is:

```
globus-gridftp-server [optional command line switches]
```

To use **globus-gridftp-server** with a config file, make sure to use the `-c <configfile>` option.

Command line options

The table below lists config file options, associated command line options (if available) and descriptions.



Note

Any boolean option can be negated on the command line by preceding the specified option with '-no-' or '-n'.
example: -no-cas or -nf.

Informational Options

help <0 1>, -h, -help	Show usage information and exit. Default value: FALSE
version <0 1> , -v, -version	Show version information for the server and exit. Default value: FALSE
versions <0 1>, -v, -versions	Show version information for all loaded globus libraries and exit. Default value: FALSE

Modes of Operation

inetd <0 1>, -i, -inetd	Run under an inetd service. Default value: FALSE
daemon <0 1>, -s, -daemon	Run as a daemon. All connections will fork off a new process and setuid if allowed. See Section 4.4.1, “Running in daemon mode” for more information. Default value: TRUE
detach <0 1>, -S, -detach	Run as a background daemon detached from any controlling terminals. See Section 4.4.1, “Running in daemon mode” for more information. Default value: FALSE
ssh, -ssh	Run over a connected ssh session. Default value: not set

exec <string> , -exec <string>	For statically compiled or non-GLOBUS_LOCATION standard binary locations, specify the full path of the server binary here. Only needed when run in <u>daemon mode</u> . Default value: not set
chdir <0 1>, -chdir	Change directory when the server starts. This will change directory to the dir specified by the chdir_to option. Default value: TRUE
chdir_to <string>, -chdir-to <string>	Directory to chdir to after starting. Will use / if not set. Default value: not set
fork <0 1>, -f, -fork	Server will fork for each new connection. Disabling this option is only recommended when debugging. Note that non-forked servers running as 'root' will only accept a single connection and then exit. Default value: TRUE
single <0 1>, -1, -single	Exit after a single connection. Default value: FALSE

Authentication, Authorization, and Security Options

auth_level <number>, -auth-level <number>	<ul style="list-style-type: none"> • 0 = Disables all authorization checks. • 1 = Authorize identity only. • 2 = Authorize all file/resource accesses. <p>If not set, the GridFTP Server uses level 2 for front ends and level 1 for data nodes.</p> <p>Default value: not set</p>
ipc_allow_from <string>, -ipc-allow-from <string>	<p>Only allow IPC connections (applicable for backend servers in a striped configuration) from these source IP addresses. Specify a comma-separated list of IP address fragments. A match is any IP address that starts with the specified fragment. Example: '192.168.1.' will match and allow a connection from 192.168.1.45. Note that if this option is used, any address not specifically allowed will be denied.</p> <p>Default value: not set</p>
ipc_deny_from <string>, -ipc-deny-from <string>	<p>Deny IPC connections (applicable for backend servers in a striped configuration) from these source IP addresses. Specify a comma-separated list of IP address fragments. A match is any IP address that starts with the specified fragment. Example: '192.168.2.' will match and deny a connection from 192.168.2.45.</p> <p>Default value: not set</p>
allow_from <string>, -allow-from <string>	<p>Only allow connections from these source IP addresses. Specify a comma-separated list of IP address fragments. A match is any IP address that starts with the specified fragment. Example: '192.168.1.' will match and allow a connection from 192.168.1.45. Note that if this option is used, any address not specifically allowed will be denied.</p>

	Default value: not set
deny_from <string> , -deny-from <string>	Deny connections from these source IP addresses. Specify a comma-separated list of IP address fragments. A match is any IP address that starts with the specified fragment. Example: '192.168.2.' will match and deny a connection from 192.168.2.45. Default value: not set
secure_ipc <0 1> , -si, -secure-ipc	Use GSI security on the IPC channel. Default value: TRUE
ipc_auth_mode <string> , -ia <string> , -ipc-auth- mode <string>	Set GSI authorization mode for the IPC connection. Options are one of the following: <ul style="list-style-type: none"> • none • host • self • subject:[subject] Default value: host
allow_anonym- ous <0 1> , -aa , -allow- anonymous	Allow cleartext anonymous access. If server is running as root, anonymous_user must also be set. Disables IPC security. Default value: FALSE
anonym- ous_names_al- lowed <string> , -an- onymous- names-allowed <string>	Comma-separated list of names to treat as anonymous users when allowing anonymous access. If not set, the default names of 'anonymous' and 'ftp' will be allowed. Use '*' to allow any user-name. Default value: not set
anonym- ous_user <string> , -an- onymous-user <string>	User to setuid to for an anonymous connection. Only applies when running as root. Default value: not set
anonym- ous_group <string> , -an- onymous-group <string>	Group to setgid to for an anonymous connection. If not set, the default group of anonymous_user will be used. Default value: not set
pw_file <string> , -password- file <string>	Enable cleartext access and authenticate users against this /etc/passwd formatted file. Default value: not set
connec- tions_max	Maximum concurrent connections allowed. Only applies when running in <u>daemon mode</u> . Unlimited if not set.

<code><number> , -connections- max <number></code>	Default value: not set
<code>connec- tions_dis- abled <0 1> , -connections- disabled</code>	Disable all new connections. Does not affect ongoing connections. This must be set in the configuration file and then a SIGHUP issued to the server in order to reload the configuration. Default value: FALSE
<code>offline_msg <string> , -offline-msg <string></code>	Custom message to be displayed to clients when the server is offline via the <code>connections_disabled</code> or <code>connections_max = 0</code> options. Default value: not set
<code>disable_com- mand_list <string> , -disable-com- mand-list <string></code>	A comma seperated list of client commands that will be disabled. Default value: not set
<code>authz_cal- louts , -cas , -authz-cal- louts</code>	Enable the GSI authorization callout framework, for callouts such as CAS. Default value: TRUE
<code>acl , -em , -acl</code>	A comma seperated list of ACL or event modules to load. Default value: not set

Logging Options

<code>log_level <string> , -d <string> , -log-level <string></code>	Log level. A comma-separated list of levels from the following: <ul style="list-style-type: none"> • ERROR • WARN • INFO • DUMP • ALL For example: <pre>globus-gridftp-server -d error,warn,info</pre> You may also specify a numeric level of 1-255. Default value: ERROR
---	---

<code>log_module <string> ,</code>	Indicates the <code>globus_logging</code> module that will be loaded. If not set, the default <code>stdio</code> module will be used and the logfile options (see next option) will apply.
--	--

<p><code>-log-module</code> <code><string></code></p>	<p>Built-in modules are <code>stdio</code> and <code>syslog</code>. Log module options may be set by specifying <code>module:opt1=val1:opt2=val2</code>. Available options for the built-in modules are:</p> <ul style="list-style-type: none"> • <code>interval</code> - Indicates buffer flush interval. Default is 5 seconds. A 0 second flush interval will disable periodic flushing, and the buffer will only flush when it is full. • <code>buffer</code> - Indicates buffer size. Default is 64k. A value of 0k will disable buffering and all messages will be written immediately. <p>Example:</p> <pre>-log-module stdio:buffer=4096:interval=10</pre> <p>Default value: not set</p>
<p><code>log_single</code> <code><string></code>, <code>-l</code> <code><string></code>, <code>-logfile</code> <code><string></code></p>	<p>Indicates the path of a single file to which you want to log all activity. If neither this option nor <code>log_unique</code> is set, logs will be written to <code>stderr</code>, unless the execution mode is detached, or <code>inetd</code>, in which case logging will be disabled.</p> <p>Default value: not set</p>
<p><code>log_unique</code> <code><string></code>, <code>-L</code> <code><string></code>, <code>-logdir</code> <code><string></code></p>	<p>Partial path to which <code>gridftp.(pid).log</code> will be appended to construct the log filename.</p> <p>Example:</p> <pre>-L /var/log/gridftp/</pre> <p>will create a separate log (<code>/var/log/gridftp/gridftp.xxxx.log</code>) for each process (which is normally each new <i>client</i> session). If neither this option nor <code>log_single</code> is set, logs will be written to <code>stderr</code>, unless the execution mode is detached, or <code>inetd</code>, in which case logging will be disabled.</p> <p>Default value: not set</p>
<p><code>log_transfer</code> <code><string></code>, <code>-Z</code> <code><string></code>, <code>-log-transfer</code> <code><string></code></p>	<p>Log NetLogger-style info for each transfer into this file.</p> <p>Default value: not set</p> <p>Example: <code>DATE=20050520163008.306532 HOST=localhost PROG=globus-gridftp-server NL.EVT=FTP_INFO START=20050520163008.305913 USER=ftp FILE=/etc/group BUFFER=0 BLOCK=262144 NBYTES=542 VOLUME=/ STREAMS=1 STRIPES=1 DEST=[127.0.0.1] TYPE=RETR CODE=226</code></p> <p>Time format is <code>YYYYMMDDHHMMSS.UUUUUU</code> (microsecs).</p> <ul style="list-style-type: none"> • <code>DATE</code>: time the transfer completed. • <code>START</code>: time the transfer started. • <code>HOST</code>: hostname of the server. • <code>USER</code>: username on the host that transferred the file. • <code>BUFFER</code>: tcp buffer size (if 0 system defaults were used). • <code>BLOCK</code>: the size of the data block read from the disk and posted to the network. • <code>NBYTES</code>: the total number of bytes transferred.

- VOLUME: the disk partition where the transfer file is stored.
- STREAMS: the number of parallel TCP streams used in the transfer.
- STRIPES: the number of stripes used on this end of the transfer.
- DEST: the destination host.
- TYPE: the transfer type, RETR is a send and STOR is a receive (ftp 959 commands).
- CODE: the FTP rfc959 completion code of the transfer. 226 indicates success, 5xx or 4xx are failure codes.

log_filemode <string> ,
-log_filemode <string>

File access permissions of log files. Should be an octal number such as 0644 (the leading 0 is required).

Default value: not set

disable_usage_stats <0|1> , -disable-usage-stats

Disable transmission of per-transfer usage statistics. See the [Usage Statistics](#)¹ section in the online documentation for more information.

Default value: FALSE

usage_stats_target <string> ,
-usage_stats_target <string>

Comma-separated list of contact strings for usage statistics listeners. The format of <string> is host:port.

Default value: usage-stats.globus.org:4810

Example:

```
-usage-stats-target usage-stats.globus.org:4810,usage-stats.uc.teragrid.org
```

In this example, the usage statistics will be transmitted to the default Globus target (usage-stats.globus.org:4810) and another target (usage-stats.uc.teragrid.org:5920).

The usage stats sent to a particular receiver may be customized by configuring it with a taglist (host:port!taglist) The taglist is a list of characters that each correspond to a usage stats tag. When this option is unset, stats are reported to usage-stats.globus.org:4810. If you set your own receiver, and wish to continue reporting to the Globus receiver, you will need to add it manually. The list of available tags follow. Tags marked * are reported by default.

- *(e) START - start time of transfer
- *(E) END - end time of transfer
- *(v) VER - version string of gridftp server
- *(b) BUFFER - tcp buffer size used for transfer
- *(B) BLOCK - disk blocksize used for transfer
- *(N) NBYTES - number of bytes transferred

¹ ../../Usage_Stats.html

- *(s) STREAMS - number of parallel streams used
- *(S) STRIPES - number of stripes used
- *(t) TYPE - transfer command: RETR, STOR, LIST, etc
- *(c) CODE - ftp result code (226 = success, 5xx = fail)
- *(D) DSI - DSI module in use
- *(A) EM - event modules in use
- *(T) SCHEME - ftp, gsiftp, sshftp, etc. (client supplied)
- *(a) APP - guc, rft, generic library app, etc. (client supplied)
- *(V) APPVER - version string of above. (client supplied)
- (f) FILE - name of file/data transferred
- (i) CLIENTIP - ip address of host running client (control channel)
- (I) DATAIP - ip address of source/dest host of data (data channel)
- (u) USER - local user name the transfer was performed as
- (d) USERDN - DN that was mapped to user id
- (C) CONFID - ID defined by -usage-stats-id config option
- (U) SESSID - unique id that can be used to match transfers in a session and transfers across source/dest of a third party transfer. (client supplied)

us- Identifying tag to include in usage statistics data.
age_stats_id Default value: not set
<string>, -us-
age-stats-id
<string>

Single and Striped Remote Data Node Options

remote_nodes Comma-separated list of remote node contact strings. See [Remote data-nodes and striped operations](#) and [Separation of processes for higher security](#) for examples of using this option.
<string>, -r
<string>, -re-
mote-nodes Default value: not set
<string>

data_node This server is a back end data node. See [Separation of processes for higher security](#) for an example of using this option.
<0|1>, -dn,
-data-node Default value: FALSE

stripe_blocks- Size in bytes of sequential data that each stripe will transfer.
ize <number>,
-sbs <number> Default value: 1048576
, -stripe-

blocksize

<number>

stripe_count Number of stripes to use per transfer when this server controls that number. If remote nodes are statically configured (via -r or remote_nodes), this will be set to that number of nodes, otherwise the default is 1.

<number> ,
-stripe-count
<number>

Default value: not set

stripe_layout Stripe layout. 1 = Partitioned, 2 = Blocked.

<number> , -sl
<number> ,
-stripe-lay-
out <number>

Default value: 2

stripe_blocksize_locked Do not allow client to override stripe blocksize with the **OPTS RETR** command.

<0|1> ,
-stripe-block-
size-locked;

Default value: FALSE

stripe_layout_locked Do not allow client to override stripe layout with the **OPTS RETR** command.

<0|1> ,
-stripe-lay-
out-locked

Default value: FALSE

Disk Options

blocksize Size in bytes of data blocks to read from disk before posting to the network.

<number> , -bs
<number> ,
-blocksize
<number>

Default value: 262144

sync_writes Flush disk writes before sending a restart marker. This attempts to ensure that the range specified in the restart marker has actually been committed to disk. This option will probably impact performance and may result in different behavior on different storage systems. See the man page for **sync()** for more information.

<0|1> , -sync-
writes

Default value: FALSE

use_home_dirs Set the startup directory to the authenticated users home dir.

, -use-home-
dirs

Default value: TRUE

perms Set the default permissions for created files. Should be an octal number such as 0644. The default is 0644. Note: If umask is set it will affect this setting -- i.e. if the umask is 0002 and this setting is 0666, the resulting files will be created with permissions of 0664.

<string> ,
-perms
<string>

Default value: not set

file_timeout Timeout in seconds for all disk accesses. A value of 0 disables the timeout.

<number> ,

Default value: not set

-file-timeout
<number>

Network Options

port <number> Port on which a front end will listen for client control channel connections or on which a data node will listen for connections from a front end. If not set, a random port will be chosen and printed via the logging mechanism. See [Remote data-nodes and striped operations](#) and [Separation of processes for higher security](#) for examples of using this option.

Default value: not set

control_interface <string> Hostname or IP address of the interface to listen for control connections on. If not set, will listen on all interfaces.

, -control-interface
<string> Default value: not set

data_interface <string> Hostname or IP address of the interface to use for data connections. If not set will use the current control interface.

, -data-interface
<string> Default value: not set

ipc_interface <string>, Hostname or IP address of the interface to use for IPC connections. If not set, will listen on all interfaces.

-ipc-interface
<string> Default value: not set

hostname <string>, Effectively sets the above control_interface, data_interface and ipc_interface options.

-hostname
<string> Default value: not set

ipc_port <number>, Port on which the front end will listen for data node connections.

-ipc-port
<number> Default value: not set

Timeouts

control_preauth_timeout Time in seconds to allow a client to remain connected to the control channel without activity before authenticating.

<number>,
-control-preauth-timeout
<number> Default value: 120

control_idle_timeout Time in seconds to allow a client to remain connected to the control channel without activity.

<number>;,
-control-idle-timeout
<number> Default value: 600

ipc_idle_timeout Idle time in seconds before an unused IPC connection will close.

<number> ,

-ipc-idle-
timeout <num-
ber> Default value: 600

ipc_con-
nect_timeout
<number> , Time in seconds before cancelling an attempted IPC connection.
-ipc-connect-
timeout <num-
ber> Default value: 60

User Messages

banner Message that is displayed to the client before authentication.
<string> ,
-banner Default value: not set
<string>

banner_file Read banner message from this file.
<string> ,
-banner-file Default value: not set
<string>

banner_terse When this is set, the minimum allowed banner message will be displayed to unauthenticated
<0|1> , -ban- clients.
ner-terse Default value: FALSE

banner_append When this is set, the message set in the 'banner' or 'banner_file' option will be appended to the
<0|1> , -ban- default banner message rather than replacing it.
ner-append Default value: FALSE

login_msg Message that is displayed to the client after authentication.
<string> , -lo-
gin-msg Default value: not set
<string>

lo-
gin_msg_file Read login message from this file.
<string> , -lo- Default value: not set
gin-msg-file
<string>

Module Options

load_dsi_mod- Load this Data Storage Interface module. File and remote modules are defined by the server. If
ule <string> , not set, the file module is loaded, unless the `remote` option is specified, in which case the remote
-dsi <string> module is loaded. An additional configuration string can be passed to the DSI using the format
 [module name]:[configuration string]. The format of the configuration string is
 defined by the DSI being loaded.

 Default value: not set

- allowed_modules <string> Comma-separated list of ERET/ESTO modules to allow and, optionally, specify an alias for. Example:
 , -allowed-modules <string> -allowed-modules module1,alias2:module2,module3
 (module2 will be loaded when a client asks for alias2).
 Default value: not set
- dc_whitelist <string> , -dc-whitelist <string> A comma separated list of drivers allowed on the network stack.
 Default value: not set
- fs_whitelist <string> , -fs-whitelist <string> A comma separated list of drivers allowed on the disk stack.
 Default value: not set
- popen_whitelist <string> , -popen-whitelist <string> A comma separated list of programs that the popen driver is allowed to execute, when used on the network or disk stack. An alias may also be specified, so that a client does not need to specify the full path. Format is [alias:]prog,[alias:]prog. example: /bin/gzip,tar:/bin/tar
 Default value: not set

Other Options

- configfile <string> , -c <string> Path to configuration file that should be loaded. Otherwise will attempt to load \$GLOBUS_LOCATION/etc/gridftp.conf and /etc/grid-security/gridftp.conf.
 Default value: not set
- debug <0|1> , -debug Set options that make the server easier to debug. Forces no-fork, no-chdir, and allows core dumps on bad signals instead of exiting cleanly. Not recommended for production servers. Note that non-forked servers running as root will only accept a single connection and then exit.
 Default value: FALSE

Limitations

For transfers using parallel data transport streams and for transfers using multiple computers at each end, the direction of the connection on the data channels must go from the sending to the receiving side. For more information about this limitations see <http://www.ogf.org/documents/GFD.20.pdf>.

Globus GridFTP server does not run on windows

Chapter 6. Graphical User Interface

1. Globus GridFTP GUI (pre-alpha)

The Globus GridFTP GUI is Java web start application. Users can get it by clicking a link; the program will be downloaded and started automatically. A pre-alpha version of the GUI is available now.

- [Download the GUI client](#)¹

The GUI client provides an easy-to-use interface for connecting to GridFTP servers and transferring files. It has the following features:

- Allows you to browse the local file system and transfer files and directories between the local system and remote GridFTP servers and between two remote GridFTP servers (third-party transfers).
- Supports file system operations such as creating, deleting and renaming files and directories.

Prerequisites:

- JDK 1.5.0+

Supported Platforms:

- Windows
- Linux
- MAC

The GUI provides two ways for generating a proxy credential required for the data transfer:

1. Creating a proxy credential using a locally stored key pair.
2. Obtaining a proxy from a MyProxy Server. For more information about MyProxy, please visit: <http://myproxy.ncsa.uiuc.edu/>.

[A demo of using the GridFTP GUI is available here](#)². Open the file ending in .htm with any browser with the Flash plugin to start the Flash demo - then just click the green arrows to progress through each screen.

2. UberFTP

NCSA, as part of their TeraGrid activity, produces a text based interactive client called UberFTP, which you may want to check out. See [the section called "Interactive clients for GridFTP"](#) for more information.

¹ <http://www-unix.globus.org/cog/demo/ogce/ftp.jnlp>

² [../demo.tar.gz](#)

Chapter 7. Configuring GridFTP

1. GridFTP server configuration overview

The configuration interface for GridFTP is the admin tool, [globus-gridftp-server](#), which can be used with a configuration file and/or run-time options.



Note

Command line options and configuration file options may both be used, but the command line *overrides* the config file.

The configuration file for the GridFTP *server* is read from the following locations, in the given order. Only the first file found will be loaded:

- Path specified with the `-c <configfile>` command line option.
- `$GLOBUS_LOCATION/etc/gridftp.conf`
- `/etc/grid-security/gridftp.conf`

Options are one per line, with the format:

`<option> <value>`

If the value contains spaces, they should be enclosed in double-quotes ("). Flags or boolean options should only have a value of 0 or 1. Blank lines and lines beginning with # are ignored.

For example:

```
port 5000
allow_anonymous 1
anonymous_user bob
banner "Welcome!"
```

For complete command documentation including all options, see [globus-gridftp-server\(1\)](#).

This page includes information about general configuration of the GridFTP server. Security options are discussed [here](#), and more advanced configuration is described [here](#).

2. Typical configuration

The following describes a typical GridFTP configuration of the front end (control channel) and back end (data channels). For other alternatives that provide greater levels of security, see [Advanced Configuration](#).

By default, the data channel and control channel are separate socket connections within the same process. The client sends a command and waits to finish before issuing the next command. This is good for a single host, traditional-type user. If you have a single host and you want an ultra-reliable and light weight file transfer service, this is a good choice. This configuration is also good for testing purposes.

3. Firewall requirements

If the GridFTP server is behind a firewall:

1. Contact your network administrator to open up port 2811 (for GridFTP control channel connection) and a range of ports (for GridFTP data channel connections) for the incoming connections. If the firewall blocks the outgoing connections, open up a range of ports for outgoing connections as well.
2. Set the environment variable `GLOBUS_TCP_PORT_RANGE`:

```
export GLOBUS_TCP_PORT_RANGE=min,max
```

where `min,max` specify the port range that you have opened for the incoming connections on the firewall. This restricts the listening ports of the GridFTP server to this range. Recommended range is 1000 (e.g., 50000-51000) but it really depends on how much use you expect.

3. If you have a firewall blocking the outgoing connections and you have opened a range of ports, set the environment variable `GLOBUS_TCP_SOURCE_RANGE`:

```
export GLOBUS_TCP_SOURCE_RANGE=min,max
```

where `min,max` specify the port range that you have opened for the outgoing connections on the firewall. This restricts the outbound ports of the GridFTP server to this range. Recommended range is twice the range used for `GLOBUS_TCP_PORT_RANGE`, because if parallel TCP streams are used for transfers, the listening port would remain the same for each connection but the connecting port would be different for each connection.



Note

If the server is behind NAT, the `--data-interface <real ip/hostname>` option needs to be used on the server.

If the GridFTP *client* is behind a firewall:

1. Contact your network administrator to open up a range of ports (for GridFTP data channel connections) for the incoming connections. If the firewall blocks the outgoing connections, open up a range of ports for outgoing connections as well.
2. Set the environment variable `GLOBUS_TCP_PORT_RANGE`

```
export GLOBUS_TCP_PORT_RANGE=min,max
```

where `min,max` specify the port range that you have opened for the incoming connections on the firewall. This restricts the listening ports of the GridFTP client to this range. Recommended range is 1000 (e.g., 50000-51000) but it really depends on how much use you expect.

3. If you have a firewall blocking the outgoing connections and you have opened a range of ports, set the environment variable `GLOBUS_TCP_SOURCE_RANGE`:

```
export GLOBUS_TCP_SOURCE_RANGE=min,max
```

where `min,max` specify the port range that you have opened for the outgoing connections on the firewall. This restricts the outbound ports of the GridFTP client to this range. Recommended range is twice the range used for `GLOBUS_TCP_PORT_RANGE`, because if parallel TCP streams are used for transfers, the listening port would remain the same for each connection but the connecting port would be different for each connection.

Additional information on Globus Toolkit Firewall Requirements is available [here](#)¹.

4. Configuring Security for GridFTP

There are many security options in GridFTP ranging from no security to higher security via GSI .

4.1. Anonymous mode

Anonymous mode (using the `-aa` option) allows any user with an FTP client to read and write (and delete) files that the server process can similarly access (it is also a quick way to test that your server works).

```
% globus-gridftp-server -aa
    Server listening at 127.0.0.1:58806
```

Warning

When the server is run in this way, anyone who can connect to the server will possess all the same rights as the user that the process is run as (directly or via `-anonymous-user`). If using this mode intentionally for open access, it is best to run under a dedicated account with limited filesystem permissions. You can also use the option below to disable FTP commands such as `STOR`, `ESTO`, `DELE`, `RDEL`, `RNTO`, etc to make sure that users can only read from the server and not write to it.

```
-disable-command-list <string>
```

Where `<string>` represents a comma separated list of client commands that will be disabled. Default: not set.

4.2. Username/password

If you trust your network and want a minimal amount of security, you can run the `globus-gridftp-server` with clear text passwords. This security model is the one originally introduced in RFC959.

Warning

We do not recommend it for long running servers open to the internet.

4.2.1. Create password file

To run the server in clear text password mode, we first need to create a password file dedicated to it. The format of the password file is the same as standard system password files; however, it is ill-advised to use a system password file. To create an entry in a GridFTP password file, run the following commands:

```
% touch pwfile
    % gridftp-password.pl >> pwfile
    Password:
```

This will ask you for a password and then create an entry in the password file for the current user name and the given password. Take a look at the file created. You will notice that the password you typed in is not in the file in a clear text form. We have run it through a one way hash algorithm before storing it in the file.

¹ <http://www.globus.org/toolkit/security/firewalls/>

4.2.2. Run the server in password mode

Simply start the server pointing it at the password file you just created.

```
% globus-gridftp-server -password-file /full/path/of/pwfile
    Server listening at 127.0.0.1:5555
```

4.2.3. Make a transfer

To run `globus-url-copy` with the password, use the following syntax:

```
globus-url-copy file:///etc/group ftp://username:pw@localhost:5000/tmp/group
```

4.3. SSHFTP (GridFTP-over-SSH)

This type of security introduces the `sshftp` control channel (frontend) protocol. This is a very simple means of obtaining strong security on the control channel only (the data channel is *not* authenticated). With this approach, you can run a GridFTP transfer anywhere that you can `ssh`. `sshftp://` leverages the ubiquitous `ssh/sshd` programs to form control channel connections much in the same way that `inetd` forms connections.

4.3.1. Configure Client-Side `sshftp://`

Every `$GLOBUS_LOCATION` must be configured for client-side `sshftp://` connections. In other words, if we wish to use **globus-url-copy** with `sshftp://` URLs, we must first configure the `$GLOBUS_LOCATION` that contains `globus-url-copy` in the following way:

```
% $GLOBUS_LOCATION/setup/globus/setup-globus-gridftp-sshftp
```

4.3.2. Configure Server Side `sshftp://`

Every host that wishes to run a **globus-gridftp-server** which can accept `sshftp://` connections must run the following command as root:

```
% $GLOBUS_LOCATION/setup/globus/setup-globus-gridftp-sshftp -server
```

In the absence of root access, a user can configure the server to allow `sshftp://` connections for that user only with the following command:

```
% $GLOBUS_LOCATION/setup/globus/setup-globus-gridftp-sshftp -server -nonroot
```

The above command creates a file named `'sshftp'` in `'/etc/grid-security'` (if run as root) or in `'$HOME/.globus'` (if run as nonroot). The default contents of the `'sshftp'` file is shown below. To configure the GridFTP server for `sshftp` transfers, you have to edit this file.

```
export GLOBUS_LOCATION=/sandbox/kettimut/421/INSTALL
. $GLOBUS_LOCATION/etc/globus-user-env.sh

#export GLOBUS_TCP_PORT_RANGE=50000,50100

$GLOBUS_LOCATION/sbin/globus-gridftp-server -ssh
# -data-interface <interface to force data connections>
```

4.3.3. Performing `sshftp://` Transfers

In this case, a `globus-gridftp-server` does not need to be running. The server will be started via the `sshd` program. Therefore, the hostname and port should be that of the `sshd` server. Run `globus-url-copy` just as you have before; simply change `ftp://` to `sshftp://`.

```
% globus-url-copy -v file:/etc/group sshftp://127.0.0.1/tmp/group % globus-url-copy -list
```

4.4. GSIFTP

This security option can be the most involved to set up, but provides the most security. It requires setting up GSI security as described in the GT Installation Guide here: [Basic Security Configuration](#).

Once GSI has been set up (host and user credentials are valid, the gridmap file is updated and you've run `grid-proxy-init` to create a proxy certificate), you simply run the GridFTP server:

```
globus-gridftp-server
```



Note

If run as `root`, it will pick up the host cert; if not, it will pick up the user cert.

Now you are ready to perform a GSI-authenticated transfer:

```
globus-url-copy <-s subject> src_url dst_url
```



Note

The `subject` option is only needed if the server was not started as `root`.

4.4.1. Running in daemon mode

The server should generally be run as `root` in daemon mode, although it is possible to run it as a user (see below). When run as `root` you will need to have a [host certificate](#).

Run the server:

```
globus-gridftp-server < -s | -S > <args>
```

where:

- s Runs in the foreground (this is the default mode).
- S Detaches from the terminal and runs in the background.

The following additional steps may be required when running as a user other than `root` (for more details, review [Basic Security Configuration](#)):

- Create a `~/ .gridmap` file, containing the DNs of any clients you wish to allow, mapped to the current username.
- Create a proxy with `grid-proxy-init`.

4.4.2. Running under inetd or xinetd

Note

We also feature a user-configurable, super-server daemon plugin called GFork. Click [here](#) for more information.

4.4.2.1. Set up xinetd/inetd config file

Note

The service name used (gsiftp in this case) should be defined in `/etc/services` with the desired port.

Here is a sample GridFTP server xinetd config entry in `/etc/xinetd.conf`:

```
service gsiftp
{
    instances            = 100
    socket_type         = stream
    wait                = no
    user                = root
    env                 += GLOBUS_LOCATION=(globus_location)
    env                 += LD_LIBRARY_PATH=(globus_location)/lib
    server              = (globus_location)/sbin/globus-gridftp-server
    server_args         = -i
    log_on_success      += DURATION
    nice                = 10
    disable              = no
}
```

Here is a sample gridftp server inetd config entry in `/etc/inetd.conf` (read as a single line):

```
gsiftp stream tcp nowait root /usr/bin/env env \
GLOBUS_LOCATION=(globus_location) \
LD_LIBRARY_PATH=(globus_location)/lib \
(globus_location)/sbin/globus-gridftp-server -i
```

Note

On Mac OS X, you must set `DYLD_LIBRARY_PATH` instead of `LD_LIBRARY_PATH` in the above examples.

On IRIX, you may need to set either `LD_LIBRARYN32_PATH` or `LD_LIBRARY64_PATH`.

Note

You should NOT include `USERID` in the log lines. See [Section 5, “High latency for GridFTP server connections”](#) for more information.

4.4.2.2. globus-gridftp-server -i

Use the `-i` commandline option with `globus-gridftp-server`:

```
globus-gridftp-server -i
```

4.5. User permissions

Users are mapped to a local account on the server machine and file permissions are handled by the operating systems. In the anonymous mode, users that connect to the server will possess all the same rights as the user that the server process is run as (directly or via `-anonymous-user`).

In case of username/password authentication, the users are mapped to the uid corresponding to the username in the GridFTP password file and the access permissions for the users is same as that of the UID that they are mapped to. If SSH based authentication is used, upon successful authentication, SSHD maps users to a local account and the GridFTP server is run as the mapped local user. The access permissions are the same as that of the mapped local user.

If GSI is used, upon successful authentication an authorization callout is invoked to (a) verify authorization and (b) determine the local user id as which the request should be executed. This callout is linked dynamically. Globus GridFTP provides an implementation that supports a Globus "gridmapfile". Sites can also provide alternative implementations. Server does a setuid to the local user id as determined by the authorization callout and the access permissions are the same as that of the local user id.

GridFTP server provides an option to disable certain FTP commands:

```
-disable-command-list <string>
```

Where `<string>` represents a comma separated list of client commands that will be disabled. Default: not set.

5. globus-gridftp-server quickstart

The following is a quick guide to running the server and using the client:

Look through the list of options for `globus-gridftp-server`:

```
globus-gridftp-server --help
```

Start the server in anonymous mode (discussed more fully [here](#)):

```
globus-gridftp-server -control-interface 127.0.0.1 -aa -p 5000
```

where:

`-control-interface` is the hostname or IP address of the interface to listen for control connections on. This option is only needed here as a rudimentary means of security for this simple example.

`-aa` enables anonymous mode

`-p` indicates on which port the server listens.

Run a two party transfer with client:

```
globus-url-copy -v file:///etc/group ftp://localhost:5000/tmp/group
```

Run 3rd party transfer:

```
globus-url-copy -v ftp://localhost:port/etc/group ftp://localhost:port/tmp/group2
```

Experiment with `-dbg`, and `-vb` options for debugging and checking the performance of your setup:

```
globus-url-copy -dbg file:///etc/group ftp://localhost:5000/tmp/group
```

```
globus-url-copy -vb file:///dev/zero ftp://localhost:5000/dev/null
```

where:

- dbg A useful option when something is not working. It results in a GridFTP control channel protocol dump (along with other useful information) to stderr. If you understand the GridFTP protocol, or you have ambition to understand it, this can be a very useful tool to discover various problems in your setup such as overloaded servers and firewalls. When submitting a bug report or asking a question on the support email lists one should always send along the -dbg output.
- vb Provides a type of progress bar of the user to observe the rate at which their transfer is progressing.

Ctrl-c - Kill the server.



Note

There are many possible options and configurations with **globus-gridftp-server**. For some guidelines on setting it up for your situation, see [Chapter 3, Key Admin Settings and Tuning Recommendations](#).

Chapter 8. Environment variable interface

1. Environment variables for GridFTP

The GridFTP *server* or *client* libraries do not read any environment variable directly, but the security and networking related variables described below may be useful.

- Non-WS (General) Authentication & Authorization Environment Variables.
- XIO Network Driver Environment Variables.

Chapter 9. Debugging

You can find information on sys admin logs in [Chapter 6, Debugging](#).

Chapter 10. Troubleshooting

If you are having problems using the GridFTP *server*, try the steps listed below. If you have an error, try checking the server logs if you have access to them. By default, the server logs to stderr, unless it is running from inetd, or its execution mode is detached, in which case logging is disabled by default.

The command line options `-d`, `-log-level`, `-L` and `-logdir` can affect where logs will be written, as can the configuration file options `log_single` and `log_unique`. See the [globus-gridftp-server\(1\)](#) for more information on these and other configuration options.

For a list of common errors in GT, see [Error Codes](#).

1. Error Codes in GridFTP

Table 10.1. GridFTP Errors

Error Code	Definition	Possible Solutions
<p>globus_ftp_client: the server responded with an error 530 530-globus_xio: Authentication Error 530-OpenSSL Error: s3_srvr.c:2525: in library: SSL routines, function SSL3_GET_CLIENT_CERTIFICATE: no certificate returned 530-globus_gsi_callback_module: Could not verify credential 530-globus_gsi_callback_module: Can't get the local trusted CA certificate: Untrusted self-signed certificate in chain with hash d1b603c3 530 End.</p>	<p>This error message indicates that the GridFTP server doesn't trust the certificate authority (CA) that issued your certificate.</p>	<p>You need to ask the GridFTP server administrator to install your CA certificate chain in the GridFTP server's trusted certificates directory.</p>
<p>globus_ftp_control: gss_init_sec_context failed OpenSSL Error: s3_clnt.c:951: in library: SSL routines, function SSL3_GET_SERVER_CERTIFICATE: certificate verify failed globus_gsi_callback_module: Could not verify credential globus_gsi_callback_module: Can't get the local trusted CA certificate: Untrusted self-signed certificate in chain with hash d1b603c3</p>	<p>This error message indicates that your local system doesn't trust the certificate authority (CA) that issued the certificate on the resource you are connecting to.</p>	<p>You need to ask the resource administrator which CA issued their certificate and install the CA certificate in the local trusted certificates directory.</p>

Error Code	Definition	Possible Solutions
530-globus_xio: Authentication Error 530-globus_gsi_callback_module: Could not verify credential 530-globus_gsi_callback_module: Could not verify credential 530-globus_gsi_callback_module: Invalid CRL: The available CRL has expired 530 End.	This error message indicates one of the following: Certificate Revocation List (CRL) for the source or destination server CA at the client has expired or CRL for client CA has expired at source or destination server or CRL for source (destination) server CA has expired at destination (source) server. CRL is a file {CA_hash}.r0 in /etc/grid-security/certificates or \${USER_HOME}/.globus/certificates or \${X509_CERT_DIR}	The tool available at http://dist.eu-gridpma.info/distribution/util/fetch-crl/ can be run in a crontab to keep the CRLs up to date.

2. Establish control channel connection

Verify that you can establish a control channel connection and that the server has started successfully by telnetting to the port on which the server is running:

```
% telnet localhost 2811
      Trying 127.0.0.1...
      Connected to localhost.
      Escape character is '^]'.
      220 GridFTP Server mldev.mcs.anl.gov 2.0 (gcc32dbg, 1113865414-1) ready.
```

If you see anything other than a 220 banner such as the one above, the server has not started correctly.

Verify that there are no configuration files being unexpectedly loaded from /etc/grid-security/gridftp.conf or \$GLOBUS_LOCATION/etc/gridftp.conf. If those files exist, and you did not intend for them to be used, rename them to .save, or specify -c none on the command line and try again.

If you can log into the machine where the server is, try running the server from the command line with only the -s option:

```
$GLOBUS_LOCATION/sbin/globus-gridftp-server -s
```

The server will print the port it is listening on:

```
Server listening at gridftp.mcs.anl.gov:57764
```

Now try and telnet to that port. If you still do not get the banner listed above, something is preventing the socket connection. Check firewalls, tcp-wrapper, etc.

If you now get a correct banner, add -p 2811 (you will have to disable (x)inetd on port 2811 if you are using them or you will get port already in use):

```
$GLOBUS_LOCATION/sbin/globus-gridftp-server -s -p 2811
```

Now telnet to port 2811. If this does not work, something is blocking port 2811. Check firewalls, tcp-wrapper, etc.

If this works correctly then re-enable your normal server, but remove all options but -i, -s, or -S.

Now telnet to port 2811. If this does not work, something is wrong with your service configuration. Check /etc/services and (x)inetd config, have (x)inetd restarted, etc.

If this works, begin adding options back one at a time, verifying that you can telnet to the server after each option is added. Continue this till you find the problem or get all the options you want.

At this point, you can establish a control connection. Now try running `globus-url-copy`.

3. Try running `globus-url-copy`

Once you've verified that you can establish a control connection, try to make a transfer using `globus-url-copy`.

If you are doing a *client/server* transfer (one of your URLs has `file:` in it) then try:

```
globus-url-copy -vb -dbg gsiftp://host.server.running.on/dev/zero file:///dev/null
```

This will run until you control-c the transfer. If that works, reverse the direction:

```
globus-url-copy -vb -dbg file:///dev/zero gsiftp://host.server.running.on/dev/null
```

Again, this will run until you control-c the transfer.

If you are doing a *third party transfer*, run this command:

```
globus-url-copy -vb -dbg gsiftp://host.server1.on/dev/zero gsiftp://host.server2.on/dev/nu
```

Again, this will run until you control-c the transfer.

If the above transfers work, try your transfer again. If it fails, you likely have some sort of file permissions problem, typo in a file name, etc.

4. If your server starts...

If the server has started correctly, and your problem is with a security failure or gridmap lookup failure, verify that you have security configured properly [here](#).

If the server is running and your client successfully authenticates but has a problem at some other time during the session, please ask for help on gt-user@globus.org¹. When you send mail or submit bugs, please always include as much of the following information as possible:

- Specs on all hosts involved (OS, processor, RAM, etc).
- `globus-url-copy -version`
- `globus-url-copy -versions`
- Output from the telnet test above.
- The actual command line you ran with `-dbg` added. Don't worry if the output gets long.
- Check that you are getting a FQDN and `/etc/hosts` that is sane.
- The server configuration and setup (`/etc/services` entries, `(x)inetd` configs, etc.).
- Any relevant lines from the server logs (not the entire log please).

¹ http://dev.globus.org/wiki/Mailing_Lists

5. High latency for GridFTP server connections

If you run GridFTP servers via Xinetd and notice high latency for connections and/or transfers, check if `/etc/xinetd.conf` or the `gsiftp` service configuration inside `/etc/xinetd.d` is set to log `USERID` as follows:

```
log_on_success += USERID
log_on_failure += USERID
```

Such a configuration tells Xinetd to log the remote user using the method defined in RFC 1413, which causes an ident client to attempt to query the machine that the connection is coming from before the service will run. Even when this succeeds, the response can't be trusted, and more often than not it is rejected or simply dropped (which results in the longest delays) by the remote firewall.

Latency can be reduced by making sure Xinetd does *not* log the `USERID`.

Chapter 11. Related Documentation

- [The Globus Striped GridFTP Framework and Server](http://www.globus.org/alliance/publications/papers/gridftp_final.pdf)¹

¹ http://www.globus.org/alliance/publications/papers/gridftp_final.pdf

Appendix A. Developing DSIs for GridFTP

The GridFTP server provides high speed remote access to data stores. There are many different types of data storage systems from standard file systems to arrays of magnetic tape. To allow GridFTP to be a transfer interface to as many data storage systems as possible the Data Storage Interface (DSI) was created.

The DSI presents a modular abstraction layer to a storage system. It consists of several function signatures and a set of semantics.

- When a new DSI is created, a programmer implements the functions to provide the semantics associated with them.
- DSIs can be loaded and switched at runtime.
- When the server requires action from the storage system (be it data, meta-data, directory creation, etc), it passes a request to the loaded DSI module.
- The DSI then services that request and tells the function when it is finished.

This document provides an introduction to the DSI and how to create one.

1. DSI Interface

The set of interface functions that define the DSI can be found in `globus_gridftp_server.h`.

All type definitions starting with `globus_gfs_storage_*` are part of the DSI interface.

2. DSI utility API

An API is provided to the DSI author to assist in implementation. The most interesting parts of this API provide functions that abstract away the details of sending data across the data channel. The DSI author is not expected to know the intimate details of the data channel protocols involved in a GridFTP transfer. Instead this API provides functions for reading and writing data to and from the net.

FIXME - link to api doc

3. Implementation

The following is a brief description of part of the DSI implementation process.

An FTP session is defined from the time a client is authorized to use the server until the time the connection is disconnected (disconnect can happen due to the client sending QUIT, error, or timeout, etc). In the lifetime of the session, the client issues various commands to the FTP server. Some of these commands require access to the storage system, and thus require action by the DSI. Whenever such a command is received, the server calls out to the appropriate DSI interface function requesting that the specific operation be performed.

The server passes a

`globus_gfs_operation_t`

data type as a parameter to all DSI request functions. When the DSI is finished performing that operation, it calls a corresponding

```
globus_gridftp_server_finished_<type>()
```

function, passing it this `globus_gfs_operation_t` structure (and whatever other data is needed for any given operation). This lets the server know that the operation is completed and it can respond to the client appropriately.

As an example we will look at how a simple unix file system DSI would implement the `stat` function.

The DSI's function signature for `stat` is:

```
void
(*globus_gfs_storage_stat_t)(
    globus_gfs_operation_t      op,
    globus_gfs_stat_info_t *    stat_info,
    void *                      user_arg);
```

When it is called, the DSI is expected to:

- determine all information associated with the path: `stat_info->pathname`,
- fill in a `globus_gfs_stat_t` with that information,
- and then call `globus_gridftp_server_finished_stat()` with that structure.

```
static
void
globus_gfs_storage_example_stat(
    globus_gfs_operation_t      op,
    globus_gfs_stat_info_t *    stat_info,
    void *                      user_arg)
{
    globus_gfs_stat_t           stat_out;
    struct stat                 stat_in;

    stat(stat_info->pathname, &stat_in);

    stat_out.mode               = stat_in.st_mode;
    stat_out.nlink              = stat_in.st_nlink;
    stat_out.uid                = stat_in.st_uid;
    stat_out.gid                = stat_in.st_gid;
    stat_out.size               = stat_in.st_size;
    stat_out.mtime              = stat_in.st_mtime;
    stat_out.atime              = stat_in.st_atime;
    stat_out.ctime              = stat_in.st_ctime;
    stat_out.dev                = stat_in.st_dev;
    stat_out.ino                = stat_in.st_ino;

    stat_out.name = strdup(stat_info->pathname);

    globus_gridftp_server_finished_stat(op, GLOBUS_SUCCESS, &stat_out, 1);
}
```

This is obviously a very basic example but it should serve for the purposes of understanding.

4. DSI Bones

Every DSI must register itself with the Globus extensions module properly. This can be a tedious task yet must be done properly. For this reason, we created a distribution that provides a skeleton DSI upon which a developer can build.

The distribution includes a script to generate C stubs for a DSI with all of the proper shared library hooks and names needed to work with the **globus-gridftp-server**. The DSI implementor must fill in the stubbed-out functions with the necessary code specific to their needs.

```
% ./generate-stubs.sh dsi name flavor
```

This command will generate the c source file. "dsi name" is the string that will be associated with the DSI. It must be unique to your Globus installation. To load it into the server use the `-dsi dsi name` option to the server.

```
% make
```

This will compile the DSI and create the dynamically loadable library. To include additional compile dependencies or libraries, open `Makefile` and add them to the appropriate `MACRO` line.

```
% make install
```

This will copy the library to `$GLOBUS_LOCATION/lib`, thereby making it ready for use.

Appendix B. GridFTP Multicast

GridFTP is a well known, extremely fast and efficient protocol for transferring data from one destination to another. Here we present how GridFTP can be used to transfer a single file to many destinations in a multicast/broadcast.

1. Architecture

The purpose of this work is to efficiently transfer a single data set to many locations. Our goal is to use all available network cycles, effectively moving the total amount of data (destinations * file size) at network speeds. Ideally, we would like to transfer to all destinations in the same time it takes to transfer to a single destination.

Distributing the data to many endpoints is not difficult, and has been a feature of **globus-url-copy** (a popular GridFTP client) for some time. It would be quite simple to have the client loop over the destination set and send to each destination in series. Of course, this would be quite slow and the transfer time would scale linearly with the data set.

In our architecture, destination servers are configured so that they can forward packets along to other endpoints. Thus, the GridFTP servers act as both servers receiving data and clients sending data. The server is set up as a tree such that the first destination writes the data to disk and then forwards it on to N more hosts (by default N is 2). This process is repeated until all destinations have received the data.

For further explanation, the architecture can be thought of as a directed graph. Each destination is a vertex with N edges connecting it to N other vertices. A spanning tree is formed connecting all vertices. The degree of any one vertex is a client-side configuration option.

The following image illustrates this:

Figure B.1. GridFTP Spanning Tree



In the image, data blocks are purple and are sent first from the client to a root destination. The root destination then forwards it on to two more servers.

2. Globus XIO

The figure in the previous section shows 4 colored boxes. Orange represents client logic, blue is server logic, and as stated above, purple is for data block. The final box type is yellow and it is used to show globus XIO drivers.

More information on Globus XIO can be found [here](#). For our purposes we can think of each XIO driver as a modular protocol interpreter that can be plugged in to an IO stack without involving the application using it. In this way, we can add functionality to an existing application without disturbing its tested code base.

Because the **globus-gridftp-server** uses Globus XIO for all of its IO, we are able to forward data at the block level. We achieve this by allowing the client to add a new XIO driver, the `gridftp_multicast` driver, to the GridFTP server's disk stack. Because of the modular driver abstraction that Globus XIO provides as the GridFTP server writes data blocks to its file system, the data blocks are first passed through the `gridftp_multicast` driver. As the `gridftp_multicast` driver passes the data block on to be written to disk, it also forwards the block on to other GridFTP servers in the tree.

Using this approach to add the multicast functionality is minimally invasive to the tested and robust GridFTP server and is entirely modular. The driver is written to a well defined and clean abstraction. Enabling this feature is a simple matter of inserting the driver in the disk stack and passing the driver stack the destination list.

3. Network Overlay

In addition to allowing for multicast, the `gridftp_multicast` driver and this architecture allow us to create a network overlay where many GridFTP servers act as routers forwarding packets along to each other until they get to the final destination where there are written to disk. The advantage of this type of system is actively researched by [Phoebus](http://e2epi.internet2.edu/phoebus.html)¹.

The `gridftp_multicast` driver can be configured to only forward data along to the next server, and to not write it to disk. Furthermore, it can be told to only forward to a single endpoint. When configured in this way, we achieve the network overlay described above.

The following images illustrate this. In the first image we show the standard case where data is sent from a client to a server through the Internet. The routing is done by the Internet outside of the clients control.

Figure B.2. From client to server through Internet



In the next image we show how the `gridftp_multicast` driver can route data through the network via GridFTP servers. This allows the user to have greater control over the network path which the data takes.

Figure B.3. Routing data through network via multicast



4. Results

To show the effectiveness of this architecture we ran experiments on the [UC TeraGrid](http://www.uc.teragrid.org/tg-docs/)². We show some of those results here.

4.1. Experiment 1

In the first experiment, we leased nodes from the UC TeraGrid. 29 hosts were designated as destinations and we ran `gridftp_multicast`-enabled GridFTP servers on them. 1 node was designated as the client node and from it all transfers were started.

All transfers were performed with `globus-url-copy` and a tcp buffer size of 128KB. As a control group, we ran a transfer using `globus-url-copy` with the `-f` option. This caused the source file to be sent to each endpoint in serial. We then transferred the source to all destinations using this architecture.

The first graph shows the completion time of a multicast session against the number of destinations. The first line is when the transfers are performed in a serial fashion from the client. All other lines are multicast sessions performed using this architecture. Each line represents a different vertex degree in the spanning tree (ie, each server forwarding to a different number of destinations).

¹ <http://e2epi.internet2.edu/phoebus.html>

² <http://www.uc.teragrid.org/tg-docs/>

Figure B.4.

As expected, the results for the serial transfer scales linearly while the multicast sessions very slowly increase with more destinations.

The next graph shows the same experiment; but instead of graphing completion time, we graph the clients sending throughput. This is the size of the files being sent (1GB) divided by the time it takes for this file to reach all destinations.

Figure B.5.

The final graph shows the collective bandwidth of a transfer. The graphing function is (# of destinations * file size) / time.

Figure B.6.

4.2. Future Experiments

We created another XIO driver that allows each endpoint in the session to buffer the data. This prevents stalls in sending data transfers due to the latency required to reach a leaf node, and gets data to the disks of nodes higher in the tree faster. Early results show slight improvements on a LAN, but we expect greater results when broadcasting across WANs.

5. Protocol Details

The additions to the protocol are exceptionally minor. Every server in the tree (except for leaf nodes) becomes a client to another server, but that client speaks the standard GridFTP protocol. The only change needed is a command to add the driver to the file system stack, and that command has existed in the GridFTP server for some time.

The command is:

```
SITE SETDISKSTACK 1*{driver name[:driver options]},
```

The second parameter to the site command is a comma-separated list of driver names optionally followed by a colon (:) and a set of driver-specific URL-encoded options. From left to right, the driver names form a stack from bottom to top.

Adding the `gridftp_multicast` driver to this list will enable the multicast functionality. The set of options are the same as those specified in the previous section. The only difference is that each url in the `urls=` options must be url encoded.

6. Usage

The broadcast functionality can be used with **globus-url-copy**. We added the following option:

```
-mc filename
```

The file must contain a line separated list of destination urls. For example:

```
gsiftp://localhost:5000/home/user/tst1
gsiftp://localhost:5000/home/user/tst2
gsiftp://localhost:5000/home/user/tst4
```

The source url is specified on the command line as always. A single destination url may also be specified on the command line in addition to the urls in the file. An example globus-url-copy command is:

```
% globus-url-copy -MC multicast.file gsiftp://localhost/home/user/src_file
```

6.1. Advanced multicasting options

Along with specifying the list of destination urls in a file, a set of options for each url can be specified. This is done by appending a ? to the resource string in the url followed by semicolon-separated key value pairs. For example:

```
gsiftp://dst1.domain.com:5000/home/user/tst1?cc=1;tcpbs=10M;P=4
```

This indicates that the receiving host `dst1.domain.com` will use 4 parallel stream, a tcp buffer size of 10 MB, and will select 1 host when forwarding on data blocks. This url is specified in the `-mc` file as described above.

The following is a list of key=value options and their meanings:

<code>P=<i>integer</i></code>	The number of parallel streams this node will use when forwarding.
<code>cc=<i>integer</i></code>	The number of urls to which this node will forward data.
<code>tcpbs=<i>format-integer</i></code>	The TCP buffer size this node will use when forwarding.
<code>urls=<i>string list</i></code>	The list of urls that must be children of this node when the spanning tree is complete.
<code>local_write=<i>boolean:Y/n</i></code>	Determines if this data will be written to a local disk, or just forwarded on to the next hop. This is explained more in the Network Overlay section.
<code>subject=<i>string</i></code>	The DN name to expect from the servers this node is connecting to.

6.2. Required Server Options

For security reasons the GridFTP server does not allow clients to load arbitrary xio drivers into the server. The GridFTP server admin must whitelist the driver individually. White-listing the mlink driver is done with the following parameter to the server:

```
-fs-whitelist file,gridftp_multicast
```

Notice that `file` must also be specified. Without this option, the `file` driver is the default; however, if used, you must specifically list it.

Glossary

some terms not in the docs but wanted in glossary: client/server transfer stream mode (MODE S) improved extended block mode (MODE X)

C

client	A process that sends commands and receives responses. Note that in GridFTP, the client may or may not take part in the actual movement of data.
client/server transfer	<p>In a client/server transfer, there are only two entities involved in the transfer, the client entity and the server entity. We use the term entity here rather than process because in the implementation provided in GT5, the server entity may actually run as two or more separate processes.</p> <p>The client will either move data from or to his local host. The client will decide whether or not he wishes to connect to the server to establish the data channel or the server should connect to him (MODE E dictates who must connect).</p> <p>If the client wishes to connect to the server, he will send the PASV (passive) command. The server will start listening on an ephemeral (random, non-privileged) port and will return the IP and port as a response to the command. The client will then connect to that IP/Port.</p> <p>If the client wishes to have the server connect to him, the client would start listening on an ephemeral port, and would then send the PORT command which includes the IP/Port as part of the command to the server and the server would initiate the TCP connect. Note that this decision has an impact on traversing firewalls. For instance, the client's host may be behind a firewall and the server may not be able to connect.</p> <p>Finally, now that the data channel is established, the client will send either the RETR "filename" command to transfer a file from the server to the client (GET), or the STOR "filename" command to transfer a file from the client to the server (PUT).</p>

E

extended block mode (MODE E)	<p>MODE E is a critical GridFTP components because it allows for out of order reception of data. This in turn, means we can send the data down multiple paths and do not need to worry if one of the paths is slower than the others and the data arrives out of order. This enables parallelism and striping within GridFTP. In MODE E, a series of "blocks" are sent over the data channel. Each block consists of:</p> <ul style="list-style-type: none">• an 8 bit flag field,• a 64 bit field indicating the offset in the transfer,• and a 64 bit field indicating the length of the payload,• followed by length bytes of payload.
------------------------------	--

Note that since the offset and length are included in the block, out of order reception is possible, as long as the receiving side can handle it, either via something like a seek on a file, or via some application level buffering and ordering logic that will wait for the out of order blocks.

M

MODE command

In reality, GridFTP is not one protocol, but a collection of several protocols. There is a protocol used on the control channel, but there is a range of protocols available for use on the data channel. Which protocol is used is selected by the MODE command. Four modes are defined: STREAM (S), BLOCK (B), COMPRESSED (C) in RFC 959 for FTP, and EXTENDED BLOCK (E) in GFD.020 for GridFTP. There is also a new data channel protocol, or mode, being defined in the GGF GridFTP Working group which, for lack of a better name at this point, is called MODE X.

See also [extended block mode \(MODE E\)](#)⁶.

See also [stream mode \(MODE S\)](#)⁷.

S

server

A process that receives commands and sends responses to those commands. Since it is a server or service, and it receives commands, it must be listening on a port somewhere to receive the commands. Both FTP and GridFTP have IANA registered ports. For FTP it is port 21, for GridFTP it is port 2811. This is normally handled via inetd or xinetd on Unix variants. However, it is also possible to implement a daemon that listens on the specified port. This is described more fully in in the Architecture section of the GridFTP Developer's Guide.

stream mode (MODE S)

The only mode normally implemented for FTP is MODE S. This is simply sending each byte, one after another over the socket in order, with no application level framing of any kind. This is the default and is what a standard FTP server will use. This is also the default for GridFTP.

T

third party transfers

In the simplest terms, a third party transfer moves a file between two GridFTP servers.

The following is a more detailed, programmatic description.

In a third party transfer, there are three entities involved. The client, who will only orchestrate, but not actually take place in the data transfer, and two servers one of which will be sending data to the other. This scenario is common in Grid applications where you may wish to stage data from a data store somewhere to a super-computer you have reserved. The commands are quite similar to the client/server transfer. However, now the client must establish two control channels, one to each server. He will then choose one to listen, and send it the PASV command. When it responds with the IP/port it is listening on, the client will send that IP/port as

⁶ #extended-block-mode

⁷ #stream-mode

part of the PORT command to the other server. This will cause the second server to connect to the first server, rather than the client. To initiate the actual movement of the data, the client then sends the RETR “filename” command to the server that will read from disk and write to the network (the “sending” server) and will send the STOR “filename” command to the other server which will read from the network and write to the disk (the “receiving” server).

See Also [client/server transfer](#).

Index

A

- accessing data
 - HPSS, 7
 - non-POSIX data source, 6
 - SRB, 8
- admin scenarios
 - running in daemon mode, 44
- API information for GridFTP, 9
- architecture (GridFTP), 6

C

- commandline tool
 - globus-gridftp-server, 27
 - globus-url-copy, 12
 - globus-url-sync, 24
- configuration interface for GridFTP, 40
- configuring GridFTP, 40
 - overview, 40
 - security
 - anonymous mode, 42
 - gsiftp, 44
 - over SSH, 43
 - username/password, 42
 - typical, 40

D

- deploying
 - running under inetd or xinetd, 45

E

- environment variable interface for GridFTP, 48
- errors, 51

G

- globus-gridftp-server, 27
- globus-url-copy, 12
- globus-url-sync, 24
- GUI information for GridFTP, 39

I

- interactive clients
 - UberFTP, 23

L

- logging, 49

M

- moving files

- single file to many destinations
 - advanced options, 63

R

- running in daemon mode, 44

T

- troubleshooting for GridFTP, 50
- tutorial, 5

GT 5.0.2 Migrating Guide for GridFTP

Table of Contents

1. Migrating GridFTP from GT4.2	1
2. Migrating GridFTP from GT3 or GT4.0	1
3. Migrating GridFTP from GT2	2
Glossary	3

<titleabbrev>Migrating Guide</titleabbrev>

The following provides available information about migrating from previous versions of the Globus Toolkit.

1. Migrating GridFTP from GT4.2

FIXME

2. Migrating GridFTP from GT3 or GT4.0

If you are running GT 3.0.0 or greater, the migration for GridFTP is relatively painless. There were only new features added. No changes were made to the existing protocol or APIs, so any existing *client* or application built using our client APIs will work unchanged. You may install new clients and servers on an as-opportunity-permits basis and will have no problems. Any combination of old/new client/server will work.

To upgrade your server, either install it in a \$GLOBUS_LOCATION different than the GT 3 installation (either as part of an entire GT 4 installation or by doing just make `gridftp`). Alternately, you can statically link the new server to avoid versioning issues and replace the existing executable. The configuration files are very different, so you will need to update the configuration.

To upgrade your client, simply build the client and use the new client as you would the old one.

Below is a list of new functionality available in GT 3.2 and higher (note that the MLST/MLSD feature is used by RFT in GT 3.2 and higher and is required if you want to be able to specify a directory to move):

New Functionality in 3.2:

- Server Improvements
 - Structured File Info
 - MLST, MLSD
 - checksum support
 - chmod support
- globus-url-copy changes
 - File globbing support
 - Recursive dir moves

- RFC 1738 support
- Control of restart
- Control of DC security

3. Migrating GridFTP from GT2

If you are running a version 2.2 or earlier, it is deprecated, unsupported, has major bugs leading to stability problems, has known potential security exploits via the wuftp *server*, and has a protocol incompatibility with later versions due to an error in the security code. Your GT 2.2 *clients* will not work with newer servers (GT 2.4.0 and greater) and new clients will not work with GT 2.2 servers. You should immediately upgrade to GT 5.0.2.

If you are running GT 2.4.0 or greater, the migration for GridFTP is relatively painless. There were only new features added. No changes were made to the existing protocol or APIs, so any existing client or application built using our client APIs will work unchanged. You may install new clients and servers on an as-opportunity-permits basis and will have no problems. Any combination of old/new client/server will work.

To upgrade your server, either install it in a \$GLOBUS_LOCATION different than the GT 2 installation (either as part of an entire GT 4 installation or by just doing `make gridftp`). Alternately, you can statically link the new server to avoid versioning issues and replace the existing executable. The configuration files are very different, so you will need to update the configuration.

To upgrade your client, simply build the client and use the new client as you would the old one.

Below is a list of new functionality available in GT 3.2 and higher (note that the MLST/MLSD feature is used by RFT in GT 3.2 and higher and is required if you want to be able to specify a directory to move):

New Functionality in 3.2:

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

Glossary

C

client A process that sends commands and receives responses. Note that in GridFTP, the client may or may not take part in the actual movement of data.

S

server A process that receives commands and sends responses to those commands. Since it is a server or service, and it receives commands, it must be listening on a port somewhere to receive the commands. Both FTP and GridFTP have IANA registered ports. For FTP it is port 21, for GridFTP it is port 2811. This is normally handled via `inetd` or `xinetd` on Unix variants. However, it is also possible to implement a daemon that listens on the specified port. This is described more fully in the Architecture section of the GridFTP Developer's Guide.

GT 5.0.2 GridFTP: Quality Profile

Table of Contents

1. Test coverage reports	1
2. Code analysis reports	1
3. Known Bugs	1
4. Bug Fixes	1
5. Performance reports	2

<titleabbrev>Quality Profile</titleabbrev>

1. Test coverage reports

There are no test coverage reports available at this time.

2. Code analysis reports

There are no code analysis reports available at this time.

3. Known Bugs

The following problems are known to exist for GridFTP at the time of the 5.0.2 release:

- GridFTP Server
 - There are some small memory leaks, though they should not grow much.
 - Some error responses are unclear.

3.1. Limitations

In order to use custom XIO drivers with GridFTP and globus-url-copy, the flavor of the driver must match the flavor of globus-gridftp-server and globus-url-copy (for 2-party transfers).

4. Bug Fixes

- [Bug GRIDFTP-68](http://jira.globus.org/browse/GRIDFTP-68)¹ - sshftp broken when password/passphrase entry is required
- [Bug GRIDFTP-57](http://jira.globus.org/browse/GRIDFTP-57)² - SITE commands not individually disableable
- [Bug GRIDFTP-62](http://jira.globus.org/browse/GRIDFTP-62)³ - Disk read for CKSM uses unexpected block size
- [Bug GRIDFTP-54](http://jira.globus.org/browse/GRIDFTP-54)⁴ - Displaying the hostname while asking for password in SSH GridFTP

¹ <http://jira.globus.org/browse/GRIDFTP-68>

² <http://jira.globus.org/browse/GRIDFTP-57>

³ <http://jira.globus.org/browse/GRIDFTP-62>

⁴ <http://jira.globus.org/browse/GRIDFTP-54>

- [Bug 6928⁵](#) - AF_LOCAL vs. AF_UNIX

5. Performance reports

- [Performance of Globus Striped GridFTP Server on TeraGrid⁶](#)
- [GridFTP Scalability and Performance Results⁷](#)

⁵ https://bugzilla.mcs.anl.gov/globus/show_bug.cgi?id=6928

⁶ ../gridftp_performance.doc

⁷ ../gridftp_scalability.doc

GT 5.0.2 Release Notes: GridFTP

Table of Contents

1. Component Overview	1
2. Feature Summary	1
3. Summary of Changes in GridFTP	3
4. Bug Fixes	3
5. Known Problems	3
6. Technology dependencies	3
7. Tested platforms	4
8. Backward compatibility summary	4
9. Associated Standards	4
10. For More Information	5
Glossary	5

<titleabbrev>Release Notes</titleabbrev>

1. Component Overview

GridFTP is a high-performance, secure, reliable data transfer protocol optimized for high-bandwidth wide-area networks. The GridFTP protocol is based on FTP, the highly-popular Internet file transfer protocol. We have selected a set of protocol features and extensions defined already in IETF RFCs and added a few additional features to meet requirements from current data grid projects.

2. Feature Summary

Features new in GT 5.0.2:

- Improved failure restart capability in globus-url-copy:

A new option to store untransferred urls for later restarting is available. In case of any failures, this option allows users to restart transfers from a checkpoint rather than restarting from scratch. This option can be used for directory transfers, single file transfer or a list of files transfer. See [Section 6.1, “Failures and retries”](#) for more details.

- Stall detection:

It is possible that the transfer hangs due to filesystem errors, network errors or GridFTP server errors. A new option is available in globus-url-copy to specify how long before canceling/restarting a transfer with no data movement. See [the section called “Reliability Options”](#) for more details.

- Client-side host aliasing:

This allows [concurrent transfers](#) to be extended to multiple different hosts rather than multiple connections to the same host, without relying on DNS.

- Initial release of command line tool globus-url-sync.

Features that continue to be supported from previous versions

- GridFTP over UDT

- SSH security for GridFTP control channel
- Running the GridFTP server with GFork GridFTP
- Multicasting / Network overlays
- Netlogger's bottleneck detection for GridFTP transfers
- GSI security: This is the PKI based, de facto standard security system used in Grid applications. Kerberos is also possible but is not supported and can be difficult to use due to divergence in the capabilities of GSI and Kerberos.
- Third-party transfers: Very common in Grid applications, this is where a *client* mediates a transfer between two servers (both likely at remote sites) rather than between the server and itself (called a *client/server transfer*).
- Cluster-to-cluster data movement or Striping: GridFTP can do coordinated data transfer by using multiple computer nodes at the source and destination.
- Partial file access: Regions of a file may be accessed by specifying an offset into the file and the length of the block desired.
- Reliability/restart: The receiving server periodically (the default is 5 seconds, but this can be changed) sends “restart markers” to the client. This marker is a messages specifying what bytes have been successfully written to the disk. If the transfer fails, the client may restart the transfer and provide these markers (or an aggregated equivalent marker), and the transfer will pick up where it left off. This can include “holes” in the file.
- Large file support: All file sizes, lengths, and offsets are 64 bits in length.
- Data channel reuse: Data channel can be held open and reused if the next transfer has the same source, destination, and credentials. This saves the time of connection establishment, authentication, and delegation. This can be a huge performance difference when moving lots of small files.
- Integrated instrumentation (Performance Markers).
- Logging/audit trail (Extensive Logging in the server).
- Parallel transfers (Multiple TCP streams between a pair of hosts).
- TCP Buffer size control (Protocol supports Manual and Automatic; Only Manual Implemented).
- Server-side computation (Extended Retrieve (ERET) / Extended Store (ESTO) commands).
- Based on Standards: RFC 959, RFC 2228, RFC 2389, IETF Draft MLST-16 , GGF GFD.020.

Other Supported Features

- On the client side we provide a scriptable tool called globus-url-copy. This tool can take advantage of all the GridFTP protocol features and can also do protocol translation between FTP, HTTP, HTTPS, and POSIX file IO on the client machine.
- We also provide a set of development libraries and APIs for developers wishing to add GridFTP functionality to their application.

Deprecated Features

- None

3. Summary of Changes in GridFTP

- New `globus-url-sync` command for syncing individual files or directories
- New server option to control the default permissions of created files
- New server option to time out on slow or hanging filesystems
- New server logging level to include transfer statistics

4. Bug Fixes

- [Bug GRIDFTP-68](#)¹ - sshftp broken when password/passphrase entry is required
- [Bug GRIDFTP-57](#)² - SITE commands not individually disableable
- [Bug GRIDFTP-62](#)³ - Disk read for CKSM uses unexpected block size
- [Bug GRIDFTP-54](#)⁴ - Displaying the hostname while asking for password in SSH GridFTP
- [Bug 6928](#)⁵ - AF_LOCAL vs. AF_UNIX

5. Known Problems

The following problems and limitations are known to exist for GridFTP at the time of the 5.0.2 release:

5.1. Known Bugs

The following problems are known to exist for GridFTP at the time of the 5.0.2 release:

- GridFTP Server
 - There are some small memory leaks, though they should not grow much.
 - Some error responses are unclear.

5.1.1. Limitations

In order to use custom XIO drivers with GridFTP and `globus-url-copy`, the flavor of the driver must match the flavor of `globus-gridftp-server` and `globus-url-copy` (for 2-party transfers).

6. Technology dependencies

GridFTP depends on the following GT components:

- Non-WS (General) Authentication & Authorization

¹ <http://jira.globus.org/browse/GRIDFTP-68>

² <http://jira.globus.org/browse/GRIDFTP-57>

³ <http://jira.globus.org/browse/GRIDFTP-62>

⁴ <http://jira.globus.org/browse/GRIDFTP-54>

⁵ https://bugzilla.mcs.anl.gov/globus/show_bug.cgi?id=6928

- C Common Libraries
- XIO

GridFTP depends on the following 3rd party software:

- OpenSSL (version is included in release)

7. Tested platforms

Tested platforms for GridFTP

- i386 Linux
- ia64 Linux (TeraGrid)
- AIX 5.2
- Solaris 9
- PA-RISC HP/UX 11.11
- ia64 HP/UX 11.22
- Tru64 Unix
- Mac OS X

While the above list includes platforms on which we have tested GridFTP, it does not imply support for a specific platform. However, we are interested in hearing reports of success or bug reports on any platform.

8. Backward compatibility summary

Protocol changes since GT 5.0.2

- None

API changes since GT 5.0.2

- None

Exception changes since GT 5.0.2

- Not Applicable (GridFTP is not Java-based)

Schema changes since GT 5.0.2

- Not Applicable (GridFTP is not SOAP-based)

9. Associated Standards

Associated standards for GridFTP:

