CASTOR2 with GridFTP v2 server architecture

(brief description)

Victor Kotlyar

Victor.Kotlyar@ihep.ru, Victor.Kotlyar@cern.ch

Institute for High Energy Physics / European Organization for Nuclear Research

Abstract

This article describes the CASTOR2 extension for Globus GridFTP server version 2. GridFTP is a protocol defined by Global Grid Forum Recommendation GFD.020, RFC 959, RFC 2228, RFC 2389, and a draft before the IETF FTP working group. The GridFTP protocol provides for the secure, robust, fast and efficient transfer of (especially bulk) data. CASTOR2 stands for Cern Advanced STORage manager. It is a hierarchical storage system developed at CERN.

1. GridFTP2 server 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 and has a few additional features to meet requirements from data grid projects.

The features of GridFTP2 are:

- A new, complete reimplementation of the server (not wuftp based).
- Support for striping.
- This new implementation will greatly ease new feature additions and modifications of the server (new commands, new data sources such as mass storage devices, etc.), maintainability, and resolves a licensing issue that was discovered.
- 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).
- 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 message 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.

2. GridFTP2 server architecture

GridFTP2 (and normal FTP) has 2 distinct components:

1. Client and server protocol interpreters which handle control channel protocol.
2. Data Transfer Process which handles the access to actual data and their movement via the data channel.

Data Transfer Process has architecturally, 3 distinct pieces:

Protocol handler, Data Storage Interface and Data processing module.

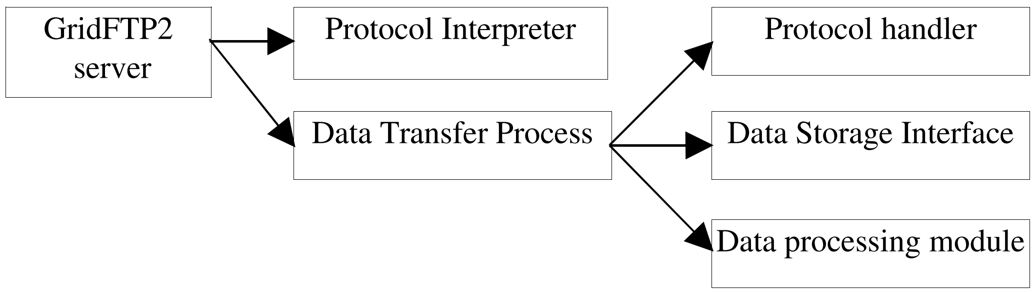


Fig.1 GridFTP2 server architecture.

The protocol handler - talks to the network and understands the data channel protocol;

The Data Storage Interface (DSI) - provides an interface to data sources and sinks;

The data processing module - provides ability to manipulate the data prior to transmission (currently handled via the DSI, in future they plan to make this a separate module).

3. CASTOR2 DSI module

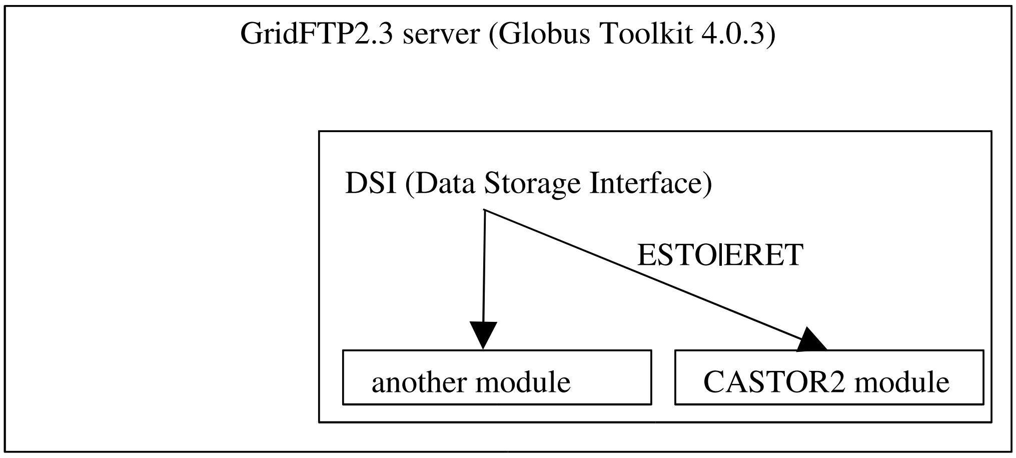
Templates and development instructions on how to build your own DSI module are in the sources of the Globus Toolkit:

```
ls ~globus/gt4.0.3-all-source-installer/source-trees/gridftp/server/src/dsi_bones/
    CVS  generate-stubs.sh  globus_gridftp_server_dsi.c.in  Makefile.in  README.txt
```

There are several systems with DSI module already:

- File systems accessible via standard POSIX API;
- Storage Resource Broker (SRB);
- High Performance Storage System (HPSS);
- NeST from the Condor team.

Current CASTOR2 DSI module is based on the DPM (Disk Pool Manager) DSI module and it uses rfio api to communicate with CASTOR2.



GridFTP2.3 server (Globus Toolkit 4.0.3)
DSI (Data Storage Interface)
ESTO|ERET
another module
CASTOR2 module

Fig. 2 CASTOR2 DSI module.

The following GridFTP commands are currently implemented:

GridFTP command	*DSI module functions*	*CASTOR functions*
LIST	globus_l_gfs_CASTOR2_stat	rfio_stat64, rfio_opendir, rfio_readdir64, rfio_rewinddir, rfio_closedir
MKD	globus_l_gfs_CASTOR2_command	rfio_mkdir
RMD	globus_l_gfs_CASTOR2_command	rfio_rmdir
DELE	globus_l_gfs_CASTOR2_command	rfio_unlink
RNTO	globus_l_gfs_CASTOR2_command	rfio_rename
CHMOD	globus_l_gfs_CASTOR2_command	rfio_chmod
STOR*	globus_l_gfs_CASTOR2_recv, globus_l_gfs_CASTOR2_read_from_net, globus_l_gfs_rfio_net_read_cb	rfio_open64, rfio_lseek64, rfio_write, rfio_close
RETR*	globus_l_gfs_CASTOR2_send, globus_l_gfs_CASTOR2_send_next_to_client, globus_l_gfs_net_write_cb	rfio_open64, rfio_lseek64, rfio_read, rfio_close
ESTO/ERET	same as STOR, RETR for DSI module	

*STOR/RETR commands allow to use parallel connections and changing buffer size from GridFTP clients.

GridFTP server can run in inetd or in daemon mode. All DSI commands are executed under uid/gui of user from the /etc/grid-security/grid-mapfile mapping file.

Example of the xinetd configuration file:

```
service gsiftp
{
instances          = 100
socket_type        = stream
wait               = no
user               = root
env                += GLOBUS_LOCATION=/usr/local/globus-4.0.3
env                += LD_LIBRARY_PATH=/usr/local/globus-4.0.3/lib
env                += RFIO_USE_CASTOR_V2=YES
env                += STAGE_HOST=lxb1387
env                += STAGE_SVCCLASS=default
env                += CNS_HOST=lxb1387
```

```
        server                = /usr/local/globus-4.0.3/sbin/globus-gridftp-server

        server_args           = -i -Z /var/spool/gsiftp/log -L /var/spool/gridftp/gridftp -l /
var/spool/gridftp/log -d 255 -dsi CASTOR2 -allowed-modules CASTOR2 -control-idle-
timeout 3600
        log_on_success        += DURATION
        nice                  = 10
        disable               = no
        }
```

4. GSIFTP as internal CASTOR2 protocol

CASTOR2 allows to use gsiftp as an internal protocol. When the user performs stage_open with gsiftp protocol, scheduler runs stagerJob on the disk server which executes gridftp server in inetd mode. In this mode stagerJob sets up UUID, FULLDESTPATH, ACCESS_MODE environment variables and CASTOR2 DSI module uses them to allow access only for one file by UUID with specific access rights.

There are a few gsiftp variables defined in castor.conf

```
GSIFTP  GLOBUS_LOCATION        /usr/local/globus-4.0.3
GSIFTP  GRIDMAP                 /etc/grid-security/grid-mapfile
GSIFTP  LOGFILE                /var/spool/gridftp/log
GSIFTP  NETLOGFILE             /var/spool/gridftp/netlog
```

Gridftp server must be available to run under stage:st privileges (stagerJob runs under stage:st) than means:

```
chown stage:st /etc/grid-security/hostcert.pem
chwon stage:st /etc/grid-security/hostkey.pem
```

and all log files must have write permissions for stage:st.

Also on the disk server a special grid-map file that does mapping for all grid users to the local stage account must be present.

5. GridFTP2 with CASTOR2

There are two ways to use gridftp v2 with CASTOR2. First, gsiftp can be used as an “external” protocol, see fig. 3.

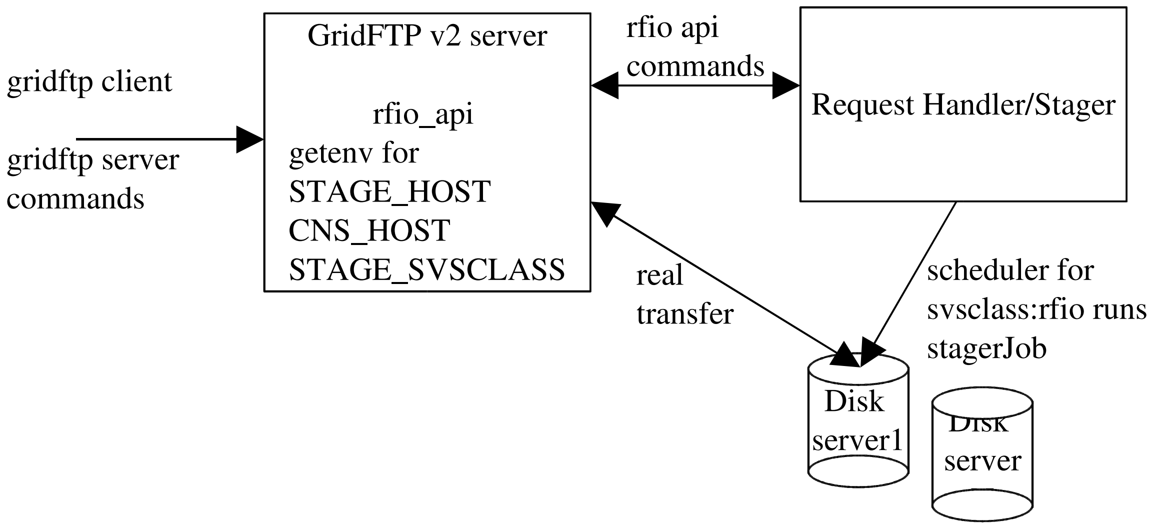


Fig. 3 gsiftp as “external” protocol.

In this way GridFTP v2 server has mapping for all grid users to the internal CASTOR2 users and when grid user performs gsi authentication she/he is mapped to uid/gid of local user and all rfio api commands execute under her/his privileges. By this schema all data transfers between client and CASTOR2 disk servers go through gridftp server.

Second, gsiftp can be used as an “internal” protocol of CASTOR2, see fig. 4.

In this case GridFTP v2 server is started by scheduler on a certain disk server and all file transfers are done between client and gridftp server on the disk server. On the disk server there must be a special mapping for all grid users to the local user “stage” and the gridftp server can only store or receive files from local file system on this disk server.

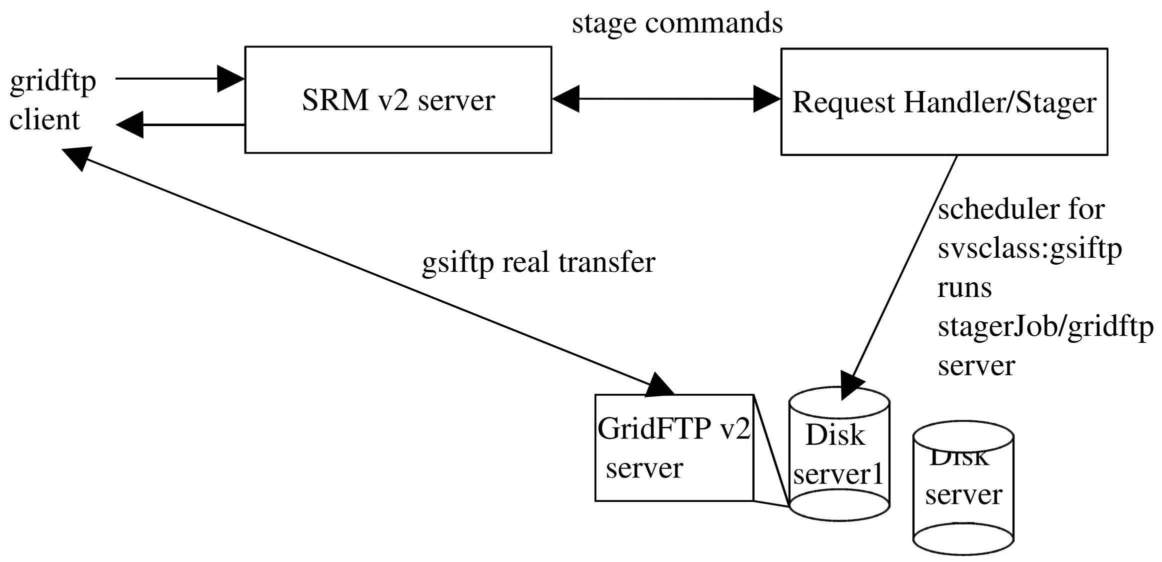


Fig. 4 gsiftp as "internal" protocol.

On each disk server node there is GridFTP2 server with modified DSI module which allows access only for store and receive commands (only transfer file) to the local node file system.

6. Acknowledgments

This work was supported by LCG project. I would also like to thank everyone of the Castor development team, namely Sebastien Ponce, Giuseppe Lo Presti, Rosa Maria Garcia Rioja, Vitaly Motyakov for the useful discussions and suggestions.