# Java Authorization Extension for GridFTP User Documentation

## Introduction

Java Authorization is an extension of Globus’ GridFTP server that allows a user to configure and use Java authorization plug-ins. The extension provides two authorization plug-ins:

1. SQL Database plug-in
2. Grid Grouper plug-in (1)

As the user, you simply install Globus GridFTP with Java Authorization extensions using the setup package. Then choose which plug-in you would like to use for authorization and configure it. This document describes the Java Authorization extension, including installation and configuration. If you are a developer who would like to write an authorization plug-in for the Java Authorization extension, please refer to the GridFTP with Java Authorization Developer Documentation.

## Background and Motivation

Globus (2) GridFTP (3) is a high-performance grid-enabled ftp server. GridFTP is used for secure bulk data transfer (BDT): transferring large blocks of data as files (or compressed archives of files) over high-bandwidth networks.

GridFTP is written in C, primarily for performance reasons. When GridFTP was developed, the NIO APIs in the Java platform did not exist. Today, GridFTP has top-notch performance, stability, and security. It is supported on a number of platforms (3).

### Host-centric GridFTP usage scenario

Most GridFTP users utilize GridFTP as a grid-accessible FTP service. Each user accessing GridFTP is mapped to a system user. Then all files that user has permissions to access are remotely accessible. This type of access is a “host-centric” access policy. Note that in this policy there are two main requirements:

1. Authenticate grid users
2. Map grid users to a local system user and use file system permissions to perform all authorization

After a default installation of GridFTP, the server is configured to trust one or more Certificate Authorities (CAs). When a user makes a GridFTP request, he presents his grid identity in the form of a X.509 certificate. The GridFTP authentication process verifies a) that the certificate is valid (e.g., not expired) and b) that the CA that issued the certificate is one of the CAs the GridFTP server trusts. After authentication, authorization is performed. Authorization consists of mapping the grid identity to a local user according to the contents of the gridmap file. At this point, the user is authorized for any request for which the local user has the required file system permissions. As an example, if the requester wants to retrieve file x, and they are mapped to a user that can read file x, then the request is authorized. This host-centric authentication and authorization policy works well for sites that want to use GridFTP as a grid-accessible FTP service. However, some sites have a need for a true Grid BDT service.

### Grid-centric GridFTP usage scenario

When using GridFTP as a grid BDT service, there are three main requirements:

1. Authenticate grid users
2. Authorize grid users according to grid permissions
3. Develop custom authorization policies to define grid permissions

Note that requirement matches requirement from the grid-accessible FTP service scenario described above. This requirement is currently satisfied by GridFTP, and is the fundamental feature that enables the service to be called GridFTP instead of FTP.

Requirement does not match requirement . In the host-centric scenario, file system permissions determine whether a request is authorized. In this grid-centric scenario, we want to use grid permissions to determine authorization. This requirement is partially satisfied by the Community Authorization Service (4). CAS provides easy configuration and enforcement of grid permissions, but only if the permissions are based on a file system model. CAS is a C authorization plug-in that builds directly on the authz framework that Globus provides.

Requirement is supported by the C authz framework added to GridFTP (as part of the CAS work). A developer programming in C can write their own authz extension to GridFTP to enforce a custom authorization scheme.

Thus, support for the grid-centric usage scenario is fairly complete if the language of choice is C and if the permissions model is a file system model. Because Java is one of the top development languages in use today, and there is much ongoing grid development in Java (e.g., caGrid (5)), having the option of performing authorization in Java is desirable. Up until now, there was no Java authorization extension for GridFTP. The Java Authorization extension to GridFTP meets requirements and for Java developers.

## Java Authorization extension overview

There are three distinct requirements the Java Authorization meets:

1. The first requirement is to bridge the existing C authz infrastructure in GridFTP to Java. The first part of the Java Authorization extension is a C authz implementation that calls Java authorization code when a GridFTP server request is made.
2. The second requirement is to provide authorization plug-ins for a user to set up GridFTP authorization to meet requirement in the Java language. The second part of the Java Authorization extension is a developer API designed for developers implementing custom Java authorization plug-ins. As mentioned above, the GridFTP with Java Authorization Developer Documentation provides details about developing a custom authorization plug-in.
3. The third requirement is to provide an easy way for Java developers to implement a custom authorization policy to meet requirement in the Java language. The third part of the Java Authorization extension is a set of plug-ins, each of which is easy to set up and use. The current authorization plug-in set, as mentioned above, includes a database plug-in and a Grid Grouper plug-in.

The rest of this document presents an overview of each requirement, describing how the Java Authorization extension meets each requirement. If you simply want to install GridFTP with Java Authorization and configure one of the provided plug-ins, please skip to the section entitled “”.

### C authz GridFTP Extension

The C authz implementation bridges the C authz support in GridFTP to the Java Authorization API. The new authz implementation is defined in a new library called “java\_callout”. The functions implemented in the library are added to the Globus installation as a libjava\_callout dynamically-linked library. One step in the GridFTP w/ Java Authorization extension installation process is to configure a Globus GSI configuration file that defines the functions that should be called at each step in the authorization process. There are four abstract types that the Java Authorization extension implements as functions defined in the libjava\_callout library:

1. Map *globus\_mapping* abstract type to the *java\_gridmap* function
2. Map *GLOBUS\_GSI\_AUTHZ\_SYSTEM\_INIT* abstract type to the *java\_system\_init\_callout* function
3. Map *GLOBUS\_GSI\_AUTHZ\_SYSTEM\_DESTROY* abstract type to the *java\_system\_destroy\_callout* function
4. Map *GLOBUS\_GSI\_AUTHORIZE\_ASYNC* abstract type to the *java\_callout* function

The next section reviews each of the abstract types listed above and defines for each function: a) when the function is called, and b) what the function does.

The *globus\_mapping* abstract type is called during login. The GridFTP control process authenticates the requester and then maps the requester to a system user. The user account is used to start the data process, spawned by the control process, that carries out the request. In the Java Authorization extension, every requester is mapped to the same user, the user that started the GridFTP process. Note that this is why the installation recommends a new user with minimal permissions is used to start the GridFTP service.

The *GLOBUS\_GSI\_AUTHZ\_SYSTEM\_INIT* abstract type is called during GridFTP service startup. The function reads in the configuration file set up during installation to know what Java VM classpath to use and what class to call during authorization. If the configuration file doesn’t exist or has a bad format, the GridFTP server will not successfully start.

The *GLOBUS\_GSI\_AUTHZ\_SYSTEM\_DESTROY* abstract type is called during GridFTP service shutdown. The function cleans up the java callout.

The *GLOBUS\_GSI\_AUTHORIZE\_ASYNC* abstract type is called during GridFTP authorization. The java class specified in the configuration file must implement the Authorize interface defined in the Java API. The Authorize method is called, performing authorization. If the method returns true, authorization is successful and the request succeeds. Otherwise, authorization fails and the request is denied.

## GridFTP with Java Authorization Installation and Configuration

### Supported Operating Systems

Currently Solaris and Linux are supported. Since the installation compiles source code on the target machines, all versions of Linux should work just fine. Testing is performed on CentOS 4 (6).

### Installation Requirements

1. Operating system supporting the Bash shell.
2. Sun JDK 5 (7) or greater is required.
3. Ant 1.6.5 (8) or greater is required.
4. GCC version 3.2 or greater is required to build Globus.
5. Working internet connection. Globus GT 4.0.4 (9) is downloaded during installation, so there is no need to install Globus separately.
6. wget to download Globus during installation. This is a common program on a modern linux system.

### Installation

Commands that should be executed in a shell are in Courier New font.

1. Create a new user to run the GridFTP server. NOTE: it is strongly recommended that you use a new user with a minimum of privileges. In particular, do NOT use root to run the GridFTP server. See the section entitled “C authz GridFTP Extension” for an explanation. You should create a new user because the installation overwrites the user’s ~/.bashrc file and installs some convenience scripts in the user’s home directory.
2. Configure the user to use the Bash shell.
3. Login as the new user.
4. Install Java 5 or higher if you do not have it (7)
5. export JAVA\_HOME=<path to java directory>
6. Install Ant 1.6.5 or higher if you do not have it (8).
7. export ANT\_HOME=<path to ant directory>
8. export PATH=${PATH}:${JAVA\_HOME}/bin:${ANT\_HOME}/bin
9. Download the setup package (TODO link).
10. Uncompress the setup package to a directory of your choice.
11. Change to the new directory.
12. ./install.sh
    1. Type the full directory path to where you want to install GridFTP (e.g., /usr/local/gridftp)
    2. Type the operating system platform (without quotes, e.g., solaris or linux)

At this point, GridFTP is installed and configured. However, you must install trusted CAs and configure the host certificate and host key. The default certificate directory is X509\_CERT\_DIR. The default host certificate is X509\_USER\_CERT. The default host key is X509\_USER\_KEY. Refer to the Globus installation or your site’s Globus setup documentation for more information. You can check ~/.bashrc for important variables used by the Java Authorization extension.

1. cd $GLOBUS\_LOCATION/bin
2. ./gridftp

The install script sets up everything you need to run GridFTP with the Java Authorization extension configured. By default, the extension is configured to use the database authorization plug-in. See the section entitled “” for more information.

The most important environment variable that you may need to know about is GRIDFTP\_JAVA\_AUTH\_CONF. This variable is set to the location of the Java Authorization extension configuration file. The file contains just two parameters: the Java VM classpath to use and the classname of the authorization class to use, which must implement the Authorization interface. You will need to modify this file if you implement your own authorization mechanism or if you want to change the default authorization plug-in.

### Database Authorization Plug-in Configuration

Database authorization uses a local HSQLDB database (10). Commands to start, stop, and connect to the database are in your home directory. After installation, it is highly recommended that you change the default database password. To do so, follow these steps:

1. Connect to the database
   1. ~/connectdb.sh
   2. set password “mypassword”;
   3. \q
2. Modify ~/sqltool.rc
   1. Update the password to the password you just set so you can use the connectdb.sh script to connect to the database.
3. Update password in the DB connection parameters
   1. cd $GRIDFTP\_ROOT
   2. cd gridftpauthz
   3. Edit properties/org/cagrid/gridftp/authorization/plugin /db/db.props

The database needs to have tuples inserted before the authorization plug-in can succeed. To add data to the database, refer to the javadoc for the DBUtil class. Database authorization logs to /tmp/AuthCallout.log (TODO get exact name).

## Grid Grouper Plug-in Configuration

Coming Soon

## Logs

The C authz implementation logs to /tmp/gridftp\_java\_auth\_callout.log\*. The \* denotes the current system time of when the GridFTP server is started. An example log filename is /tmp/gridftp\_java\_auth\_callout.log1174512520. Check the log if you encounter any authorization errors. An example error is if you develop your own custom authorization plug-in and misconfigured the java authorization config file.

The Database Authorization Plug-in logs to /tmp/AuthCallout.?.log. Here “?” refers to a number that is determined by Java. Java ensures the plug-in will log to a new file. An example log filename is /tmp/AuthCallout.0.log.

GridFTP output is logged to $GLOBUS\_LOCATION/gridftp\_log/gridftp\_log.txt

### Java Authorization API and Custom Authorization Plug-in Development

Please refer to the GridFTP with Java Authorization Developer Documentation for information.

# Works Cited

1. Grid Grouper. *caGrid Wiki.* [Online] http://www.cagrid.org/mwiki/index.php?title=GridGrouper:Main.

2. Globus. *Globus.* [Online] http://www.globus.org.

3. GridFTP. *Globus Toolkit.* [Online] http://www.globus.org/toolkit/docs/4.0/data/gridftp/.

4. Globus. *Community Authorization Service for Group Collaboration.* [Online] http://www.globus.org/alliance/publications/papers/CAS\_2002\_Revised.pdf.

5. caGrid. *caGrid.* [Online] http://www.cagrid.org.

6. CentOS. *CentOS.* [Online] http://www.centos.org/.

7. Java Downloads. *Sun Microsystems, Inc.* [Online] http://java.sun.com/javase/downloads/index.jsp.

8. Apache Ant Binary Distributions. *Apache.* [Online] http://ant.apache.org/bindownload.cgi.

9. Globus Toolkit Downloads. *Globus.* [Online] http://www.globus.org/toolkit/downloads/.

10. HSQLDB. *HSQLDB.* [Online] http://hsqldb.org/.

11. Java GridFTP Client. *Globus Toolkit.* [Online] http://www.globus.org/cog/jftp/guide.html.