# GridFTP w/ Java Authorization User Documentation

## Introduction

Java Authorization is an extension of Globus’ GridFTP server that allows a user to configure and use Java authorization checks. Currently, there are two such checks:

1. SQL Database check
2. Grid Grouper check

As the user, you simply install Globus GridFTP with Java Authorization extensions. Then choose which check 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 plugin for the Java Authorization extension, please refer to the GridFTP w/ Java Authorization Developer Documentation.

## Background and Motivation

Globus GridFTP is a high-performance grid-enabled ftp server. It primarily is used for bulk data transfer: 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

By default, GridFTP support authentication and authorization, but does not perform any authorization checks. That is, the server supports security but doesn’t have security configured out-of-the-box. There are a few C authorization checks, such as Community Authorization Service, that the community provides. These checks can be set up and configured for use, but again they are in C. Since Java is one of the top development languages in use today, having the option of performing authorization in Java is desirable. Up until now, there has been no Java authorization check that could be installed and configured. The Java Authorization extension to GridFTP meets three requirements.

1. The first requirement the Java Authorization extension meets is in bridging the existing C authz infrastructure, implemented by CAS, to Java. Essentially, the Java Authorization callout is a C authz implementation that calls Java code when a GridFTP server request is made.
2. The second requirement the Java Authorization extension meets is a developer API for implementing Java authorization checks. As mentioned above, the GridFTP with Java Authorization Developer Documentation provides further details on developing a custom authorization plugin.
3. The third requirement the Java Authorization extension meets is easy configuration of provided authorization checks. Provided authorization checks, as mentioned above, are the database check and the Grid Grouper check.

This rest of this document presents an overview of each area and finishes with installing and configuring the provided checks. If you simply want to install GridFTP with Java Authorization, please skip to section (TODO section number).

## C authz implementation

The GridFTP Java API provides Java client access to the GridFTP server. Though it provides secure client access, there isn’t currently a way to secure the GridFTP server using Java code, since the server is written in C. The GridFTP Java Authorization API and associated server-side hooks fill this gap. Specifically, the C authz implementation bridges the C authz support in GridFTP to the Java Authorization API. See the GridFTP with Java Authorization Developer Documentation for more information.

## Java Authorization API and Custom Authorization Plugin Development

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

## Java Authorization Installation and Configuration

### Supported 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. We have tested the installation on CentOS 4.

### Requirements

JDK 5 is the minimum Java version that is supported. Ant 1.6.5 or greater is also needed. Globus GT 4 is the tested version of Globus. GT 4 or higher shoulder work.

### Installation

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).
2. Login as the new user.
3. Download the setup package (TODO link).
4. Uncompress the setup package in a directory of your choice.
5. Change to the new directory.
6. Install Java 5 or higher if you do not have it (8)
7. export JAVA\_HOME=/my/java/home
8. Install Ant 1.6.5 or higher if you do not have it (9).
9. export ANT\_HOME=/my/ant/home
10. export PATH=${PATH}:${JAVA\_HOME}/bin:${ANT\_HOME}/bin
11. install.sh
    1. Type the full directory path to where you want to install GridFTP (e.g., /usr/local/gridftp)
    2. Type in your operating system without quotes (e.g., solaris or linux)
12. cd $GLOBUS\_LOCATION/bin
13. ./gridftp

The install script sets up everything you need to install GridFTP. It sets up the database authorization implementation by default. The most important environment variable that you may need to modify is GRIDFTP\_JAVA\_AUTH\_CONF which points to the location of the Java Authorzation configuration file. The file is very simple, with just two parameters: the Java VM classpath to use and the classname of the class implementing the Authorization interface. You will need to modify this file if you implement your own authorization mechanism.

TODO important environment variables and config files. TODO: a note about setting up security certificates properly (certs in $GLOBUS\_AUTH\_DIR/certificates, etc).

### Database Authorization Check Configuration

Database authorization uses a local HSQLDB database. 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. Change to the authorization directory (TODO get the exact name)
   3. Edit properties/org/cagrid/authorization/callout/gridftp/db/db.props

The database needs to have tuples inserted before the authorization check can succeed. To add data to the database, use org.cagrid.authorization.callout.gridftp.db.DBUtil. Refer to the javadoc for the documentation (TODO javadoc reference). Database authorization logs to /tmp/AuthCallout.log (TODO get exact name).

## Grid Grouper Check Configuration

Coming Soon

## Logs

The C authz implementation logs to /tmp/gridftp\_java\_auth\_log\*. The denotes the current system time of when the GridFTP server is started. The current system time is appended to the log so as not to overwrite previous logs. Check this file often to make sure the authz module is calling your Java code properly. You can check the log to make sure there are no classpath or classname configuration issues in your $GRIDFTP\_JAVA\_AUTH\_CONF file.

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

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

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

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

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

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