

Installing GT 4.2.0

DRAFT

Installing GT 4.2.0

Introduction

This guide is the starting point for everyone who wants to install Globus Toolkit 4.2.0. It will take you through a basic installation that installs Java WS Core and the following Base Services: a security infrastructure (GSI), GridFTP, Execution Services (GRAM), and Information Services (WS MDS).

This guide is also available as a [PDF](#)¹. However, each component includes online reference material, which this guide sometimes links to.

DRAFT

¹ installingGT.pdf

Table of Contents

1. Before you begin	1
2. Software Prerequisites	2
1. Required software	2
2. Optional software	2
3. Platform Notes	4
1. Apple MacOS X	4
2. Debian	4
3. Fedora Core	4
4. FreeBSD	4
5. HP/UX	4
6. IBM AIX	4
7. Red Hat	5
8. SGI Altix (IA64 running Red Hat)	5
9. Sun Solaris	5
10. SuSE Linux	6
11. Tru64 Unix	6
12. Windows	6
4. Installing GT 4.2.0	7
1. Basic Installation	7
2. Advanced Installation	9
5. Basic Security Configuration	10
1. Set environment variables	10
2. Obtain host certificates	10
3. Make the host credentials accessible by the container	11
4. Add authorization	12
5. Verify Basic Security	12
6. Firewall configuration	13
6. Basic Setup for GT 4.2.0	14
A. Packaging details	15
1. The makefile	15
2. The Grid Packaging Toolkit	15
3. Picking a flavor for a source installation	16
B. Environmental Variables in GT 4.2.0	17
1. Common Runtime Environmental Variables	17
2. Security Environmental Variables	19
3. Data Management Environmental Variables	24
4. Execution Management Environmental Variables	25
C. Installing SimpleCA	27
1. Create users	27
2. Run the setup script	27
3. Host certificates	30
4. User certificates	31
5. Verify the SimpleCA certificate installation	31
6. Configure SimpleCA for multiple machines	32
D. Deploying in GT 4.2.0	33
1. Deploying into the Java WS Core container	33
2. Deploying into Tomcat	33
3. Deploying into JBoss	38
E. Troubleshooting your installation	40
F. Detailed Configuration by Component	42
G. Security Considerations in GT 4.2.0	44

1. Common Runtime	44
2. Security	46
3. Data Management	48
4. Information Services	51
5. Execution Management	52
H. Usage Statistics	53
1. Common Runtime Usage Statistics	53
2. Data Management Usage Statistics	54
3. Execution Management Usage Statistics	57
Glossary	58

DRAFT

List of Tables

B.1. Globus standard environment variables	17
B.2. Launch script specific environment variables	17
B.3. Options supported by the GLOBUS_OPTIONS environment property	18
B.4. Environment variables	23
C.1. CA Name components	27

DRAFT

Chapter 1. Before you begin

Before you start installing the Globus Toolkit 4.2.0, there are a few things you should consider. The toolkit contains many subcomponents, and you may only be interested in some of them.

There are non-web services implementations of:

- Security (Non-WS Authentication and Authorization)
- File Transfers GridFTP
- Resource Management (GRAM2)
- Replica Location Service
- and Information Services (MDS2).

Important

These all run on Unix platforms only.

Additionally, there are WSRF implementations of:

- Security (Delegation Service, ...?)
- Resource Management (GRAM4)
- Reliable File Transfer (RFT)
- and Information Services (Index Service).

Important

All the Java clients to these services run on both Windows and Unix. The WSRF GRAM service requires infrastructure that only runs on Unix systems.

Therefore, if you are new to the toolkit and want to experiment with all of the components, you may want to use a Unix system. If you are interested in the Windows development, you may restrict yourself to the Java-based software.

Chapter 2. Software Prerequisites

1. Required software

- Globus Toolkit installer, from Globus Toolkit [download page](#)¹
- Make sure Java is installed: J2SE 1.5.0+ SDK from [Sun](#)², [IBM](#)³, [HP](#)⁴, or [BEA](#)⁵ (do not use [GCJ](#)⁶).
- [Ant 1.6.2+](#)⁷. Do not use the Ant distributed with Fedora Core 2, or other recent RedHat/Fedora distributions (RHEL3, Scientific Linux 3.0.x, FC3/4).
- C compiler. If [gcc](#)⁸, avoid version 3.2. 3.2.1 and 2.95.x are okay.
- [GNU tar](#)⁹
- [GNU sed](#)¹⁰
- [zlib 1.1.4+](#)¹¹
- [GNU Make](#)¹²
- [sudo](#)¹³ for basic GRAM4 functionality. For a more complete list of prerequisites, see [Prerequisites for installing GRAM4](#).
- Openssl 0.9.7 or later.
- gpt-3.2autotools2004 (shipped with the installers, but required if building standalone GPT bundles/packages)

2. Optional software

- [IODBC](#)¹⁴ (compile requirement for RLS) For a more complete list of RLS prerequisites, see [Prerequisites for RLS](#).
 - A Relational Database Server (RDBMS) that supports ODBC (we provide instructions for both PostgreSQL and MySQL [olink to appendix]):
 - If you use PostgreSQL, you'll also need [psqlODBC](#) (the ODBC driver for PostgreSQL).
 - If you use MySQL, you'll also need the [MyODBC](#) (Connector/ODBC) packages. MySQL's top level installation directory must be specified. By default these are assumed to be in `$GLOBUS_LOCATION`.

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

² <http://java.sun.com/j2se>

³ <http://www.ibm.com/developerworks/java/jdk>

⁴ <http://www.hp.com/java>

⁵ <http://www.bea.com/framework.jsp?CNT=index.htm&FP=/content/products/jrocket>

⁶ <http://gcc.gnu.org/java/>

⁷ <http://jakarta.apache.org/ant>

⁸ <http://gcc.gnu.org>

⁹ <http://www.gnu.org/software/tar/tar.html>

¹⁰ <http://www.gnu.org/software/sed/sed.html>

¹¹ <http://www.zip.org/zlib/>

¹² <http://www.gnu.org/software/make/>

¹³ <http://www.courtesan.com/sudo/>

¹⁴ <http://www.iodbc.org/>

- The package is used to interface to the ODBC layer of the RDBMS. The location of iODBC and the `odbc.ini` file must be specified before installing the RLS server.
- Tomcat¹⁵ (required by WebMDS, optional for other services) - Make sure to download it directly from the Apache web site.
- gLite¹⁶ Java VOMS parsing libraries - binary available¹⁷ (compile requirement for Workspace Service)

DRAFT

¹⁵ <http://jakarta.apache.org/tomcat/>

¹⁶ <http://glite.web.cern.ch/glite/security/>

¹⁷ <http://www.mcs.anl.gov/workspace/glite-security-util-java.jar>

Chapter 3. Platform Notes

In this section, the word "flavor" refers to a combination of compiler type (gcc or other), 32 or 64 bit libraries, and debugging enabled or not.

1. Apple MacOS X

Binaries not available due to [GPT bug 258](#)¹.

2. Debian

Some kernel/libc combinations trigger a threading problem. See bug [#2194](#)². The workaround is to set LD_ASSUME_KERNEL=2.2.5 in your environment.

3. Fedora Core

Fedora Core 2 and later ship with a broken ant. Install your own ant from <http://ant.apache.org>³ and either remove the ant RPM or edit /etc/ant.conf, setting ANT_HOME to your own ant installation.

4. FreeBSD

No known issues.

5. HP/UX

Specify `--with-flavor=vendorcc32` on the configure line. GNU tar, GNU sed, and GNU make are required on the PATH.

Tested on a PA-RISC box with HP-UX 11.11 with IPv6 patches.

- HP Ansi-C compiler, version B.11.11.12
- Java 1.4.2_06
- Apache Ant 1.6.2

For HP-UX/IA64 and for additional details about GT4 on HP-UX/PA-RISC, please consult the [HP GT4 support](#)⁴ page.

6. IBM AIX

Supported flavors are `vendorcc32dbg/vendorcc32` and `vendorcc64dbg/vendorcc64` using the Visual Age compilers (xlc). No gcc flavors are supported. Specify a flavor using `--with-flavor=flavor`.

GNU sed, tar, and make are required before the IBM ones in the PATH.

¹ http://bugzilla.ncsa.uiuc.edu/show_bug.cgi?id=258

² http://bugzilla.globus.org/globus/show_bug.cgi?id=2194

³ <http://ant.apache.org/>

⁴ <http://h30097.www3.hp.com/globus/gt4/index.html>

The toolkit has been tested on AIX 5.2 with:

- Visual Age C/C++ 6.0
- 32 bit version of IBM Java 1.4
- Apache Ant 1.5.4

7. Red Hat

When building from source on a Red Hat Enterprise line version 3 or 4 based OS, GPT might have a problem retrieving exit codes from subshells. You might see errors which says they were both successful and failed:

```
BUILD SUCCESSFUL Total time: 11 seconds ERROR: Build has failed make: ***  
[globus_wsrf_servicegroup] Error 10
```

The workaround is to configure with `--with-buildopts="-verbose"`

8. SGI Altix (IA64 running Red Hat)

Some extra environment variables are required for building MPI flavors. For the Intel compiler:

```
export CC=icc export CFLAGS=-no-gcc export CXX=icpc export CXXFLAGS=-no-gcc export  
LDLFLAGS=-lmpi
```

For the GNU compiler:

```
export CC=gcc export CXX=g++ export LDLFLAGS=-lmpi
```

In both cases, configure with `--with-flavor=mpicc64`

9. Sun Solaris

Supported flavors are gcc32, gcc64, vendorcc32 and vendorcc64. The dbg flavors should work as well. For gcc64, a gcc built to target 64 bit object files is required. The gcc32dbg flavor will be used by default. Specify other flavors using `--with-flavor=flavor`.

For Solaris 10, you may need to use an updated GNU binutils, or the provided Sun `/usr/ccs/bin/ld` to link. See [binutils bug 1031](http://bugzilla.redhat.com/show_bug.cgi?id=1031)⁵ for details on Solaris 10 symbol versioning errors.

GPT has problems with the Sun provided perl and tar: <http://www.gridpackagingtools.org/book/latest-stable/ch01s07.html>

The toolkit has been tested on Solaris 9 with:

- Sun Workshop 6 update 2 C 5.3
- gcc 3.4.3
- Sun Java 1.4.2_02
- Apache Ant 1.5.4

⁵ http://sources.redhat.com/bugzilla/show_bug.cgi?id=1031

10. SuSE Linux

No known issues.

11. Tru64 Unix

Specify `--with-flavor=vendorcc64` on the configure line. GNU tar, GNU sed, and GNU make are required on the PATH.

The toolkit has been tested on Tru64 UNIX (V5.1A and V5.1B) with:

- HP C V6.4-009 and V6.5-003 compilers
- Java 1.4.2_04
- Apache Ant 1.6.2

For additional details about GT4 on Tru64 Unix, please consult the [HP GT4 support](#)⁶ page.

12. Windows

Only Java-only components will build. Please choose the Java WS Core-only download and follow the instructions in the [Java WS Core Admin Guide](#).

⁶ <http://h30097.www3.hp.com/globus/gt4/index.html>

Chapter 4. Installing GT 4.2.0

1. Basic Installation

1. Create a user named "globus". This non-privileged user will be used to perform administrative tasks such as starting and stopping the container, deploying services, etc. Pick an installation directory, and make sure this account has read and write permissions in the installation directory.

Tip

You might need to create the target directory as root, then chown it to the globus user:

```
# mkdir /usr/local/globus-4.2.0.1
# chown globus:globus /usr/local/globus-4.2.0.1
```

Important

If for some reason you do *not* create a user named "globus", be sure to run the installation as a *non-root* user. In that case, make sure to pick an install directory that your user account has write access to.

2. Download the required software noted in [Software Prerequisites for Installing GT](#).

Tip

Be aware that Apache Ant will use the Java referred to by JAVA_HOME, *not* necessarily the first Java executable on your PATH. Be sure to set JAVA_HOME to the top-level directory of your Java installation before installing.

Also, check the [Platform Notes for GT](#) if your OS includes ant already. Your `/etc/ant.conf` is probably configured to use `gej`, which will fail to compile the Toolkit.

3. In this guide we will assume that you are installing to `/usr/local/globus-4.2.0.1`, but you may replace `/usr/local/globus-4.2.0.1` with whatever directory you wish to install to.

As the globus user, run:

```
globus$ export GLOBUS_LOCATION=/usr/local/globus-4.2.0.1
globus$ ./configure --prefix=$GLOBUS_LOCATION
```

You can use command line arguments to `./configure` for a more custom install. Here are the lines to enable features which are disabled by default:

Optional Features:

<code>--enable-prewsmds</code>	Build pre-webservices mds. Default is disabled.
<code>--enable-wsgram-condor</code>	Build GRAM Condor scheduler interface. Default is disabled.
<code>--enable-wsgram-lsf</code>	Build GRAM LSF scheduler interface. Default is disabled.
<code>--enable-wsgram-pbs</code>	Build GRAM PBS scheduler interface. Default is disabled.
<code>--enable-il8n</code>	Enable internationalization. Default is disabled.

```
--enable-drs          Enable Data Replication Service. Default is disabled.
[...]
Optional Packages:
[...]
--with-iodbc=dir      Use the iodbc library in dir/lib/libiodbc.so.
Required for RLS builds.
--with-gsiopensshargs="args"
Arguments to pass to the build of GSI-OpenSSH, like
--with-tcp-wrappers
```

For a full list of options, see **`./configure --help`**. For a list of GSI-OpenSSH options, see [Optional Build-Time Configuration for GSI-OpenSSH](#). For more information about our packaging or about choosing a flavor, see [Packing Details for Installing GT](#).

4. Run:

```
globus$ make
```

Note that this command can take several hours to complete. If you wish to have a log file of the build, use **tee**:

```
globus$ make 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**.



Note

Using make in parallel mode (-j) is not entirely safe, and is not recommended.

5. Finally, run:

```
globus$ make install
```

This completes your installation. Now you may move on to the configuration sections of the following chapters.

We recommend that you install any security advisories available for your installation, which are available from the [Advisories page](#)¹. You may also be interested in subscribing to some [mailing lists](#)² for general discussion and security-related announcements.

Your next step is to setup security, which includes picking a CA to trust, getting host certificates, user certificates, and creating a grid-mapfile. The next three chapters cover these topics.

With security setup, you may start a GridFTP server, configure a database for RFT, and configure WS-GRAM. You may also start a GSI-OpenSSH daemon, setup a MyProxy server, run RLS, and use CAS. The following chapters will explain how to configure these technologies. If you follow the chapters in order, you will make sure of performing tasks in dependency order.

¹ <http://www.globus.org/toolkit/advisories.html>

² http://dev.globus.org/wiki/Mailing_Lists

2. Advanced Installation

2.1. Building from CVS

[For advanced developers: general instructions for building from CVS and then list of branches? just olink to existing page...]

2.2. Building a specific package from source

[For advanced developers: general instructions for building from source then table with list of packages?]

2.3. Detailed installation instructions for these components

The following is a list of links to more detailed installation information available for the following components:

- [Building and installing GridFTP](#)
- [Building and installing RLS](#)
- [Building and installing WS RLS](#)
- [Optional Build-Time Configuration for GSI-OpenSSH](#)

Chapter 5. Basic Security Configuration

1. Set environment variables

In order for the system to know the location of the Globus Toolkit commands you just installed, you must set an environment variable and source the `globus-user-env.sh` script.

1. As globus, set `GLOBUS_LOCATION` to where you installed the Globus Toolkit. This will be one of the following:

- Using Bourne shells:

```
globus$ export GLOBUS_LOCATION=/path/to/install
```

- Using csh:

```
globus$ setenv GLOBUS_LOCATION /path/to/install
```

2. Source `$GLOBUS_LOCATION/etc/globus-user-env.{sh,csh}` depending on your shell.

- Use `.sh` for Bourne shell:

```
globus$ . $GLOBUS_LOCATION/etc/globus-user-env.sh
```

- Use `.csh` for C shell.

```
globus$ source $GLOBUS_LOCATION/etc/globus-user-env.csh
```

2. Obtain host certificates

You must have X509 certificates to use the GT 4.2.0 software securely (referred to in this documentation as *host certificates*). For an overview of certificates for [GSI](#) (security) see [GSI Configuration Information](#) and [GSI Environmental Variables](#).

Host certificates must:

- consist of the following two files: `hostcert.pem` and `hostkey.pem`
- be in the appropriate directory for secure services: `/etc/grid-security/`
- be for a machine which has a consistent name in DNS; you should *not* run it on a computer using DHCP where a different name could be assigned to your computer.

You have the following options:

2.1. Request a certificate from an existing CA

Your best option is to use an already existing CA. You may have access to one from the company you work for or an organization you are affiliated with. Some universities provide certificates for their members and affiliates. Contact

your support organization for details about how to acquire a certificate. You may find your CA listed in the [TERENA Repository](#)¹.

If you already have a CA, you will need to follow their configuration directions. If they include a CA setup package, follow the CAs instruction on how to install the setup package. If they do not, you will need to create an `/etc/grid-security/certificates` directory and include the CA cert and signing policy in that directory. See [Configuring a Trusted CA](#) for more details.

This type of certificate is best for service deployment and Grid inter-operation.

2.2. SimpleCA

SimpleCA provides a wrapper around the OpenSSL CA functionality and is sufficient for simple Grid services. Alternatively, you can use OpenSSL's `CA.sh` command on its own. Instructions on how to use the SimpleCA can be found in [Installing SimpleCA](#).

SimpleCA is suitable for testing or when a certificate authority is not available.

2.3. Low-trust certificate

Globus offers a low-trust certificate available at <http://gcs.globus.org:8080/gcs>. This option should only be used as a last resort because it does not fulfill some of the duties of a real Certificate Authority.

This type of certificate is best suited for short term testing.

3. Make the host credentials accessible by the container

The host key (`/etc/grid-security/hostkey.pem`) is only readable to root. The *container* (hosting environment provided by Java WS Core) will be running as a non-root user (probably the `globus` user) and in order to have a set of host credentials which are readable by the container, we need to copy the host certificate and key and change the ownership to the container user.

Note

This step assumes you have obtained a signed host certificate from your CA.

As root, run:

```
root# cd /etc/grid-security
root# cp hostkey.pem containerkey.pem
root# cp hostcert.pem containercert.pem
root# chown globus.globus containerkey.pem containercert.pem
```

At this point the certificates in `/etc/grid-security` should look something like:

```
root# ls -l *.pem
-rw-r--r-- 1 globus globus 1785 Oct 14 14:47 containercert.pem
-r----- 1 globus globus  887 Oct 14 14:47 containerkey.pem
```

¹ <http://www.tacar.org/>

```
-rw-r--r-- 1 root  root  1785 Oct 14 14:42 hostcert.pem
-r----- 1 root  root    887 Sep 29 09:59 hostkey.pem
```

4. Add authorization

Add authorizations for users:

Create `/etc/grid-security/grid-mapfile` as root.

You need two pieces of information:

- the subject name of a user
- the account name it should map to.

The syntax is one line per user, with the certificate subject followed by the user account name.

Run **grid-cert-info** to get your subject name, and **whoami** to get the account name:

```
bacon$ grid-cert-info -subject
/O=Grid/OU=GlobusTest/OU=simpleCA-mayed.mcs.anl.gov/OU=mcs.anl.gov/CN=Charles Bacon
bacon$ whoami
bacon
```

You may add the line by running the following as root:

```
root# $GLOBUS_LOCATION/sbin/grid-mapfile-add-entry -dn \
"/O=Grid/OU=GlobusTest/OU=simpleCA-mayed.mcs.anl.gov/OU=mcs.anl.gov/CN=Charles Bacon" \
-ln bacon
```

The corresponding line in the `grid-mapfile` should look like:

```
"/O=Grid/OU=GlobusTest/OU=simpleCA-mayed.mcs.anl.gov/OU=mcs.anl.gov/CN=Charles Bacon" bacon
```

Important

The quotes around the subject name are *important*, because it contains spaces.

5. Verify Basic Security

Now that you have installed a trusted CA, acquired a `hostcert` and acquired a `usercert`, you may verify that your security setup is complete. As your user account, run the following command:

```
bacon$ grid-proxy-init -verify -debug
```

```
User Cert File: /home/bacon/.globus/usercert.pem
User Key File: /home/bacon/.globus/userkey.pem
```

```
Trusted CA Cert Dir: /etc/grid-security/certificates
```

```
Output File: /tmp/x509up_u506
Your identity: /DC=org/DC=doegrids/OU=People/CN=Charles Bacon 332900
Enter GRID pass phrase for this identity:
Creating proxy ...+++++
```

```
.....+++++
Done
Proxy Verify OK
Your proxy is valid until: Fri Jan 28 23:13:22 2005
```

There are a few things you can notice from this command. Your usercert and key are located in `$HOME/.globus/`. The proxy certificate is created in `/tmp/`. The "up" stands for "user proxy", and the `_u506` will be your UNIX userid. It also prints out your distinguished name (DN), and the proxy is valid for 12 hours.

If this command succeeds, your single node is correctly configured.

6. Firewall configuration

For information on configuring services in the presence of a firewall, see [the firewall PDF](#)².

² <http://www.globus.org/toolkit/security/firewalls/>

Chapter 6. Basic Setup for GT 4.2.0

The [Quickstart Guide](#) walks you through setting up basic services on multiple machines.

DRAFT

Appendix A. Packaging details

1. The makefile

You do not have to build every subcomponent of this release. The makefile specifies subtargets for different functional subpieces. See the component map at [GT4 Facts](#)¹ for more details.

Makefile targets

- i18n: Internationalization libraries
- prewsgram: Pre-webservices GRAM
- gridftp: GridFTP
- prewsmds: OpenLDAP-based MDS2
- prews: GRAM2, MDS2, and GridFTP
- wsjava: Java WS Core
- wsc: C WS core
- wsmds: WS MDS
- wsdel: Delegation Service
- wsrft: Reliable File Transfer service
- wsgram: GRAM4
- wscas: Community Authorization Service
- wstests: Tests for java webservices
- wsctests: Tests for C webservices
- prews-test: Tests for pre-webservices components
- rls: Replica Location Service

Note that all of these targets require the "install" target also. So, for instance, to build GridFTP alone, you would run:

```
$ ./configure --prefix=/path/to/install
$ make gridftp install
```

2. The Grid Packaging Toolkit

The Globus Toolkit is packaged using the Grid Packaging Toolkit (GPT). The GPT provides a way for us to version packages and express dependencies between packages. The Makefile for the installer is automatically generated based

¹ [../GT4figure.jpg](#)

on the GPT dependencies expressed in CVS. GPT versions also allow us to release update packages for small subsets of our code. For more information on the GPT, you may see its [website](#)².

3. Picking a flavor for a source installation

If you're building on a platform that is not auto-detected by the configure script, you will be prompted to specify a flavor for the `--with-flavor=` option. Typically "gcc32dbg" will work as a flavor to build 32-bit binaries using gcc. If you want to force a 64bit build, "gcc64dbg" should work.

Some platforms have better support from their native compilers, so you can use "vendorcc32dbg" to build using the native cc. Similarly, "vendorcc64dbg" will force a 64bit build instead.

² <http://gridpackagingtools.com/book/latest-stable/index.html>

Appendix B. Environmental Variables in GT 4.2.0

1. Common Runtime Environmental Variables

1.1. Environmental variables for Java WS Core

Table B.1. Globus standard environment variables

<i>Name</i>	<i>Value</i>	<i>Description</i>	<i>Comments</i>
GLOBUS_LOCATION	<path>	The <path> is the root location of the Java WS Core installation. Must be an absolute path.	Required
GLOBUS_TCP_PORT_RANGE	<min,max>	The <min,max> is the minimum and maximum port range for TCP server sockets (useful for systems behind firewalls). For example, if set, the notification sink on the client will be started within that port range.	Optional
GLOBUS_TCP_SOURCE_PORT_RANGE	<min,max>	The <min,max> is the minimum and maximum port range for TCP outgoing sockets (useful for systems behind firewalls).	Optional
GLOBUS_UDP_SOURCE_PORT_RANGE	<min,max>	The <min,max> is the minimum and maximum port range for UDP outgoing sockets (useful for systems behind firewalls).	Optional
GLOBUS_HOSTNAME	<host>	The <host> is either a hostname or ip address. The host ip address under which the container and services will be exposed.	Optional

Table B.2. Launch script specific environment variables

<i>Name</i>	<i>Value</i>	<i>Description</i>	<i>Comments</i>
GLOBUS_OPTIONS	<arguments>	The <arguments> are arbitrary arguments that can be passed to the JVM. See below for a detailed list of supported options.	Optional
JAVA_HOME	<path>	The <path> is the root location of the JVM installation. If set, the JVM from that installation will be used. Otherwise, the first one found in path will be used.	Optional
CLASSPATH	<classpath>	This environment property is ignored by launch scripts.	Ignored

Table B.3. Options supported by the GLOBUS_OPTIONS environment property

<i>Name</i>	<i>Value</i>	<i>Description</i>
-Dorg.globus.wsrfr.proxy.port	int	This property specifies the port number of the proxy server. The proxy server must run on the same machine as the container. This setting will cause the service address to have the port of the proxy instead of the container (only applies to code that uses the ServiceHost or AddressingUtils API).
-Dorg.globus.wsrfr.container.server.id	string	This property specifies the server id. The server id is used to uniquely identify each container instance. For example, each container gets its own persistent directory based on the server id. By default the standalone container will store the persistent resources under the <code>~/ .globus/persisted/<ip>-<containerPort></code> directory. While in Tomcat the <code>~/ .globus/persisted/<ip>-<webApplicationName></code> directory will be used instead. This property overwrites the default server id and therefore indirectly controls which storage directory is used by the container. If set, the container will store the persisted resources under <code>~/ .globus/persisted/<server.id>/</code> instead. Note, that if somehow multiple containers running as the same user on the same machine end up with the same server id / persistent directory they might overwrite each other's persistent data.
-Dorg.globus.wsrfr.container.persistence.dir	directory	This property specifies the base directory that will be used for storing the persistent resources. This property overwrites the default (<code>~/ .globus/persisted/</code>) base directory assumed by the container.

Any JVM options can also be passed using the GLOBUS_OPTIONS environment property.

1.2. Environmental variables for XIO

The vast majority of the environment variables that effect the Globus XIO framework are defined by the driver in use. The following are links to descriptions of the more common driver environment variables:

- http://www.globus.org/api/c-globus-4.2.0/globus_xio/html/group_tcp_driver_envs.html
- http://www.globus.org/api/c-globus-4.2.0/globus_xio/html/group_file_driver_envs.html
- http://www.globus.org/api/c-globus-4.2.0/globus_xio/html/group_gsi_driver_envs.html
- http://www.globus.org/api/c-globus-4.2.0/globus_xio/html/group_udp_driver_envs.html

1.3. Environmental variables for C Common Libraries

- GLOBUS_ERROR_VERBOSE=1 can be set to enable verbose error messages.
- GLOBUS_ERROR_OUTPUT=1 can be set to enable output of all errors (including some that should be ignored).

2. Security Environmental Variables

2.1. Environmental Variables for GSI C

2.1.1. Credentials

Credentials are looked for in the following order:

1. service credential
2. host credential
3. proxy credential
4. user credential

`X509_USER_PROXY` specifies the path to the *proxy credential*. If `X509_USER_PROXY` is not set, the proxy credential is created (by **grid-proxy-init**) and searched for (by client programs) in an operating-system-dependent local temporary file.

`X509_USER_CERT` and `X509_USER_KEY` specify the path to the end entity (user, service, or host) certificate and corresponding *private key*. The paths to the certificate and key files are determined as follows:

For *service credentials*:

1. If `X509_USER_CERT` and `X509_USER_KEY` exist and contain a valid certificate and key, those files are used.
2. Otherwise, if the files `/etc/grid-security/service/servicecert` and `/etc/grid-security/service/servicekey` exist and contain a valid certificate and key, those files are used.
3. Otherwise, if the files `$GLOBUS_LOCATION/etc/grid-security/service/servicecert` and `$GLOBUS_LOCATION/etc/grid-security/service/servicekey` exist and contain a valid certificate and key, those files are used.
4. Otherwise, if the files `service/servicecert` and `service/servicekey` in the user's `.globus` directory exist and contain a valid certificate and key, those files are used.

For *host credentials*:

1. If `X509_USER_CERT` and `X509_USER_KEY` exist and contain a valid certificate and key, those files are used.
2. Otherwise, if the files `/etc/grid-security/hostcert.pem` and `/etc/grid-security/hostkey.pem` exist and contain a valid certificate and key, those files are used.
3. Otherwise, if the files `$GLOBUS_LOCATION/etc/grid-security/hostcert.pem` and `$GLOBUS_LOCATION/etc/grid-security/hostkey.pem` exist and contain a valid certificate and key, those files are used.
4. Otherwise, if the files `hostcert.pem` and `hostkey.pem` in the user's `.globus` directory, exist and contain a valid certificate and key, those files are used.

For *user credentials*:

1. If `X509_USER_CERT` and `X509_USER_KEY` exist and contain a valid certificate and key, those files are used.

2. Otherwise, if the files `usercert.pem` and `userkey.pem` exist in the user's `.globus` directory, those files are used.
3. Otherwise, if a PKCS-12 file called `usercred.p12` exists in the user's `.globus` directory, the certificate and key are read from that file.

2.1.2. Gridmap file

GRIDMAP specifies the path to the *grid map file*, which is used to map distinguished names (found in certificates) to local names (such as login accounts). The location of the grid map file is determined as follows:

1. If the GRIDMAP environment variable is set, the grid map file location is the value of that environment variable.
2. Otherwise:
 - If the user is root (uid 0), then the grid map file is `/etc/grid-security/grid-mapfile`.
 - Otherwise, the grid map file is `$HOME/.gridmap`.

2.1.3. Trusted CAs directory

X509_CERT_DIR is used to specify the path to the trusted certificates directory. This directory contains information about which CAs are trusted (including the *CA certificates* themselves) and, in some cases, configuration information used by **grid-cert-request** to formulate certificate requests. The location of the trusted certificates directory is determined as follows:

1. If the X509_CERT_DIR environment variable is set, the trusted certificates directory is the value of that environment variable.
2. Otherwise, if `$HOME/.globus/certificates` exists, that directory is the trusted certificates directory.
3. Otherwise, if `/etc/grid-security/certificates` exists, that directory is the trusted certificates directory.
4. Finally, if `$GLOBUS_LOCATION/share/certificates` exists, then it is the trusted certificates directory.

2.1.4. GSI authorization callout configuration file

GSI_AUTHZ_CONF is used to specify the path to the *GSI authorization callout configuration file*. This file is used to configure authorization callouts used by both the gridmap and the authorization API. The location of the GSI authorization callout configuration file is determined as follows:

1. If the GSI_AUTHZ_CONF environment variable is set, the authorization callout configuration file location is the value of this environment variable.
2. Otherwise, if `/etc/grid-security/gsi-authz.conf` exists, then this file is used.
3. Otherwise, if `$GLOBUS_LOCATION/etc/gsi-authz.conf` exists, then this file is used.
4. Finally, if `$HOME/.gsi-authz.conf` exists, then this file is used.

2.1.5. GAA (Generic Authorization and Access control) configuration file

GSI_GAA_CONF is used to specify the path to the GSI *GAA (Generic Authorization and Access control) configuration file*. This file is used to configure policy language specific plugins to the GAA-API. The location of the GSI GAA configuration file is determined as follows:

1. If the GSI_GAA_CONF environment variable is set, the GAA configuration file location is the value of this environment variable.
2. Otherwise, if `/etc/grid-security/gsi-gaa.conf` exists, then this file is used.
3. Otherwise, if `$GLOBUS_LOCATION/etc/gsi-gaa.conf` exists, then this file is used.
4. Finally, if `$HOME/.gsi-gaa.conf` exists, then this file is used.

2.1.6. Grid security directory

GRID_SECURITY_DIR specifies a path to a directory containing configuration files that specify default values to be placed in certificate requests. This environment variable is used only by the **grid-cert-request** and **grid-default-ca** commands.

The location of the *grid security directory* is determined as follows:

1. If the GRID_SECURITY_DIR environment variable is set, the grid security directory is the value of that environment variable.
2. If the configuration files exist in `/etc/grid-security`, the grid security directory is that directory.
3. If the configuration files exist in `$GLOBUS_LOCATION/etc`, the grid security directory is that directory.

2.2. Environmental variables for WS Authentication & Authorization (Java)

Refer to [Configuring](#) for environment variables. Note that the above environment variables do *not* supersede any settings provided in security descriptors.

2.3. Environmental variables for the Delegation Service

Refer to the [environment variable interface](#) for details.

The environment variables described above only affect the selection of credentials if no credentials are specified in any of the applicable security descriptors.

2.4. Environmental variables for CAS

All CAS client programs use the following environment variables to determine the appropriate URL to connect to and server identity to expect. In all cases, the command line options takes precedence over the environment variables.

- The URL is determined using this algorithm:
 - If the `-c` command line option was specified, the URL specified with that option is used.

- Otherwise, the `CAS_SERVER_URL` environment variable must be set, and its value is used.
- The server identity (i.e. the expected subject name of the CAS server certificate) is determined as follows:
 - If the `-s` command line option was specified, the value specified with that option is used as the identity
 - Otherwise, if the `CAS_SERVER_IDENTITY` environment variable is set, the value of that variable is used as the expected server identity. Ensure that the value is enclosed within double quotes if there are spaces in the DN. *The double quotes are required by the CAS scripts when they are run from a Windows shell, although the shell does not require it even if the value has spaces.*
 - If neither is set, host authorization is done and the expected server credential is `cas/<fqdn>`, where `<fqdn>` is the fully qualified domain name of the host on which the CAS service is up.

DRAFT

2.5. Environmental variables for MyProxy

Table B.4. Environment variables

MYPROXY_SERVER	Specifies the hostname where the myproxy-server is running. This environment variable can be used in place of the <code>-s</code> option.
MYPROXY_SERVER_PORT	Specifies the port where the myproxy-server is running. This environment variable can be used in place of the <code>-p</code> option.
MYPROXY_SERVER_DN	Specifies the distinguished name (DN) of the myproxy-server . All MyProxy client programs authenticate the server's identity. By default, MyProxy servers run with host credentials, so the MyProxy client programs expect the server to have a distinguished name of the form "host/<fqhn>" or "myproxy/<fqhn>" (where <fqhn> is the fully-qualified hostname of the server). If the server is running with some other DN, you can set this environment variable to tell the MyProxy clients to accept the alternative DN.
X509_USER_CERT	Specifies a non-standard location for the certificate from which the <i>proxy credential</i> is created by myproxy-init . It also specifies an alternative location for the server's certificate. By default, the server uses <code>/etc/grid-security/hostcert.pem</code> when running as root or <code>~/ .globus/usercert.pem</code> when running as non-root.
X509_USER_KEY	Specifies a non-standard location for the <i>private key</i> from which the proxy credential is created by myproxy-init . It also specifies an alternative location for the server's private key. By default the server uses <code>/etc/grid-security/hostkey.pem</code> when running as root or <code>~/ .globus/userkey.pem</code> when running as non-root.
X509_USER_PROXY	Specifies an alternative location for the server's certificate and private key (in the same file). Use when running the server with a proxy credential. Note that the proxy will need to be periodically renewed before expiration to allow the myproxy-server to keep functioning. When the myproxy-server runs with a non-host credential, clients must have the MYPROXY_SERVER_DN environment variable set to the distinguished name of the certificate being used by the server.
GLOBUS_LOCATION	Specifies the root of the MyProxy installation, used to find the default location of the <code>myproxy-server.config</code> file and the credential storage directory.
LD_LIBRARY_PATH	The MyProxy server is typically linked dynamically with Globus security libraries, which must be present in the dynamic linker's search path. This typically requires <code>\$GLOBUS_LOCATION/lib</code> to be included in the list in the LD_LIBRARY_PATH environment variable, which is set by the <code>\$GLOBUS_LOCATION/libexec/globus-script-initializer</code> script, which should be called from any myproxy-server startup script. Alternatively, to set LD_LIBRARY_PATH appropriately for the Globus libraries in an interactive shell, source <code>\$GLOBUS_LOCATION/etc/globus-user-env.sh</code> (for sh shells) or <code>\$GLOBUS_LOCATION/etc/globus-user.env.csh</code> (for csh shells).
GT_PROXY_MODE	Set to "old" to use the "legacy globus proxy" format. By default, MyProxy uses the RFC 3820 compliant proxy (also known as "proxy draft compliant") format. If GT_PROXY_MODE is set to "old", then <code>myproxy-init</code> will store a legacy proxy and <code>myproxy-logon</code> will retrieve a legacy proxy (if possible). Note that if the repository contains a proxy certificate, rather than an end-entity certificate, the retrieved proxy will be of the same type as the stored proxy, regardless of the setting of this environment variable.

2.6. Environmental variables for GSI-OpenSSH

The GSI-enabled OpenSSH needs to be able to find certain files and directories in order to properly function.

The items that OpenSSH needs to be able to locate, their default location and the environment variable to override the default location are:

- *Host key*
 - Default location: `/etc/grid-security/hostkey.pem`
 - Override with `X509_USER_KEY` environment variable
- *Host certificate*
 - Default location: `/etc/grid-security/hostcert.pem`
 - Override with `X509_USER_CERT` environment variable
- *Grid map file*
 - Default location: `/etc/grid-security/grid-mapfile`
 - Override with `GRIDMAP` environment variable
- *Certificate directory*
 - Default location: `/etc/grid-security/certificates`
 - Override with `X509_CERT_DIR` environment variable

3. Data Management Environmental Variables

3.1. 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.

3.2. Environmental variables for RFT

The only environment variable that needs to be set for RFT is `GLOBUS_LOCATION`, in order to run the command line clients, which should be set to the location of the globus installation.

3.3. Environmental variables for DRS

- `GLOBUS_LOCATION=/path/to/globus/install`

4. Execution Management Environmental Variables

4.1. Environment variables for GridWay

Important

You should include the following environment variables in your shell configuration file. (example `$HOME/.bashrc`)

In order to set the user environment, follow these steps:

1. Set up Globus user environment:

```
$ source $GLOBUS_LOCATION/etc/globus-user-env.sh
```

or

```
$ . $GLOBUS_LOCATION/etc/globus-user-env.csh
```

depending on the shell you are using.

2. Set up the GridWay user environment:

```
$ export GW_LOCATION=<path_to_GridWay_installation>
$ export PATH=$PATH:$GW_LOCATION/bin
```

or

```
$ setenv GW_LOCATION <path_to_GW_location>
$ setenv PATH $PATH:$GW_LOCATION/bin
```

depending on the shell you are using.

3. Optionally, you can set up your environment to use the GridWay DRMAA library:

```
$ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$GW_LOCATION/lib
```

or:

```
$ setenv LD_LIBRARY_PATH $LD_LIBRARY_PATH:$GW_LOCATION/lib
```

4. If GridWay has been compiled with accounting support, you may need to set up the DB library. For example, if DB library has been installed in `/usr/local/BerkeleyDB.4.4`:

```
$ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/BerkeleyDB.4.4/lib
```



Note

This step is only needed if your environment has not been configured, ask your administrator.

5. DRMAA extensions for all the languages use the dynamic drmaa libraries provided by GridWay. To use this libraries it is needed to tell the operating system where to look for them. Here are described the steps needed to do this in Linux and MacOS X.

- 1. In linux we have two ways to do this, one is using environment variables and the other one is modifying systemwide library path configuration. You only need to use one of this methods. If you do not have root access to the machine you are using or you do not want to setup it for every user in your system you have to use the environment variable method.
- 1.1 The environment variable you have to set so the extensions find the required DRMAA library is `LD_LIBRARY_PATH` with a line similar to:

```
export LD_LIBRARY_PATH=$GW_LOCATION/lib
```

If you want to setup this systemwide you can put this line alongside `GW_LOCATION` setup into `/etc/profile`. If you do not have root access or you want to do it per user the best place to do it is in the user's `.bashrc`.

You can also do this steps in the console before launching your scripts as it will have the same effect.

- Systems that use GNU/libc (GNU/Linux is one of them) do have a systemwide configuration file with the paths where to look for dynamic libraries. You have to add this line to `/etc/ld.so.conf`:

```
<path_to_gridway_installation>/lib
```

After doing this you have to rebuild the library cache issuing this command:

```
# ldconfig
```

- In MacOS X you have to use the environment variable method described for Linux but this time the name of the variable is `DYLD_LIBRARY_PATH`.

Appendix C. Installing SimpleCA

The following are instructions for how to use SimpleCA to set up certificates for a GT 4.2.0 installation.

SimpleCA provides a wrapper around the OpenSSL CA functionality and is sufficient for simple Grid services. Alternatively, you can use OpenSSL's `CA.sh` command on its own. SimpleCA is suitable for testing or when a certificate authority (CA) is not available. You can find other CA options in [Obtaining host certificates](#).

1. Create users

Make sure you have the following users on your machine:

- Your *user* account, which will be used to run the client programs.
- A generic *globus* account, which will be used to perform administrative tasks such as starting and stopping the container, deploying services, etc. This user will also be in charge of managing the SimpleCA. To do this, make sure this account has read and write permissions in the `$GLOBUS_LOCATION` directory.

2. Run the setup script

A script was installed to set up a new SimpleCA. You only need to run this script *once* per Grid.

Run the setup script:

```
$GLOBUS_LOCATION/setup/globus/setup-simple-ca
```

2.1. 2.1 Configure the subject name

This script prompts you for information about the CA you wish to create:

The unique subject name for this CA is:

```
cn=Globus Simple CA, ou=simpleCA-mayed.mcs.anl.gov, ou=GlobusTest, o=Grid
```

Do you want to keep this as the CA subject (y/n) [y]:

where:

Table C.1. CA Name components

cn	Represents "common name". Identifies this particular certificate as the CA certificate within the "GlobusTest/simpleCA-hostname" domain, which in this case is Globus Simple CA.
ou	Represents "organizational unit". Identifies this CA from other CAs created by SimpleCA by other people. The second "ou" is specific to your hostname (in this cases GlobusTest).
o	Represents "organization". Identifies the Grid.

Press `y` to keep the default subject name (recommended).

2.2. Configure the CA's email

The next prompt looks like:

```
Enter the email of the CA (this is the email where certificate
requests will be sent to be signed by the CA):
```

Enter the email address where you intend to receive certificate requests. It should be your real email address that you check, not the address of the globus user.

2.3. Configure the expiration date

Then you'll see:

```
The CA certificate has an expiration date. Keep in mind that
once the CA certificate has expired, all the certificates
signed by that CA become invalid. A CA should regenerate
the CA certificate and start re-issuing ca-setup packages
before the actual CA certificate expires. This can be done
by re-running this setup script. Enter the number of DAYS
the CA certificate should last before it expires.
[default: 5 years (1825 days)]:
```

This is the number of days for which the CA certificate is valid. Once this time expires, the CA certificate will have to be recreated, and all of its certificates regranted.

Accept the default (recommended).

2.4. Enter a passphrase

Next you'll see:

```
Generating a 1024 bit RSA private key
.....++++++
.....++++++
writing new private key to '/home/globus/.globus/simpleCA//private/cakey.pem'
Enter PEM pass phrase:
```

The passphrase of the CA certificate will be used only when signing certificates (with **grid-cert-sign**). It should be hard to guess, as its compromise may compromise all the certificates signed by the CA.

Enter your passphrase.

Important:

Your passphrase must *not* contain any spaces.

2.5. Confirm generated certificate

Finally you'll see the following:

A self-signed certificate has been generated for the Certificate Authority with the subject:

```
/O=Grid/OU=GlobusTest/OU=simpleCA-mayed.mcs.anl.gov/CN=Globus Simple CA
```

If this is invalid, rerun this script

```
setup/globus/setup-simple-ca
```

and enter the appropriate fields.

The private key of the CA is stored in /home/globus/.globus/simpleCA//private/cakey.pem
The public CA certificate is stored in /home/globus/.globus/simpleCA//cacert.pem

The distribution package built for this CA is stored in

```
/home/globus/.globus/simpleCA//globus_simple_ca_68ea3306_setup-0.17.tar.gz
```

This information will be important for setting up other machines in your grid. The number *68ea3306* in the last line is known as your *CA hash*. It will be an 8 hexadecimal digit string.

Press any key to acknowledge this screen.

Your CA setup package finishes installing and ends the procedure with the following reminder:

Note: To complete setup of the GSI software you need to run the following script as root to configure your security configuration directory:

```
/opt/gt4/setup/globus_simple_ca_68ea3306_setup/setup-gsi
```

For further information on using the setup-gsi script, use the -help option. The -default option sets this security configuration to be the default, and -nonroot can be used on systems where root access is not available.

```
setup-ssl-utils: Complete
```

We'll run the setup-gsi script in the next section. For now, just notice that it refers to your \$GLOBUS_LOCATION and the *CA Hash* from the last message.

2.6. Complete setup of GSI

To finish the setup of GSI, we'll run the script noted in the previous step.

Run the following as root (or, if no root privileges are available, add the **-nonroot** option to the command line):

```
$GLOBUS_LOCATION/setup/globus_simple_ca_CA_Hash_setup/setup-gsi -default
```

The output should look like:

```
setup-gsi: Configuring GSI security
Installing /etc/grid-security/certificates//grid-security.conf.CA_Hash...
Running grid-security-config...
Installing Globus CA certificate into trusted CA certificate directory...
Installing Globus CA signing policy into trusted CA certificate directory...
setup-gsi: Complete
```

3. Host certificates

You must request and sign a host certificate and then copy it into the appropriate directory for secure services. The certificate must be for a machine which has a consistent name in DNS; you should not run it on a computer using DHCP where a different name could be assigned to your computer.

3.1. 3.1 Request a host certificate

As root, run:

```
grid-cert-request -host 'hostname'
```

This creates the following files:

- /etc/grid-security/hostkey.pem
- /etc/grid-security/hostcert_request.pem
- (an empty) /etc/grid-security/hostcert.pem

Note: If you are using your own CA, follow their instructions about creating a hostcert (one which has a commonName (CN) of your hostname), then place the cert and key in the /etc/grid-security/ location. You may then proceed to [User certificates](#).

3.2. Sign the host certificate

1. As globus, run:

```
grid-ca-sign -in hostcert_request.pem -out hostsigned.pem
```

2. A signed host certificate, named `hostsigned.pem` is written to the current directory.
3. When prompted for a passphrase, enter the one you specified in [Enter a passphrase](#) (for the private key of the CA certificate.)
4. As root, move the signed host certificate to `/etc/grid-security/hostcert.pem`.

The certificate should be owned by root, and read-only for other users.

The key should be read-only by root.

4. User certificates

Users also must request user certificates, which you will sign using the *globus* user.

4.1. Request a user certificate

As your normal user account (*not globus*), run:

```
grid-cert-request
```

After you enter a passphrase, this creates

- `~$USER/.globus/usercert.pem` (empty)
- `~$USER/.globus/userkey.pem`
- `~$USER/.globus/usercert_request.pem`

Email the `usercert_request.pem` file to the SimpleCA maintainer.

4.2. Sign the user certificate

1. As the SimpleCA owner *globus*, run:

```
grid-ca-sign -in usercert_request.pem -out signed.pem
```

2. When prompted for a password, enter the one you specified in [Enter a passphrase](#) (for the private key of the CA certificate).
3. Now send the signed copy (`signed.pem`) back to the user who requested the certificate.
4. As your normal user account (*not globus*), copy the signed user certificate into `~/ .globus/` and rename it as `usercert.pem`, thus replacing the empty file.

The certificate should be owned by the user, and read-only for other users.

The key should be read-only by the owner.

5. Verify the SimpleCA certificate installation

To verify that the SimpleCA certificate is installed in `/etc/grid-security/certificates` and that your certificate is in place with the correct permissions, run:

```
user$ grid-proxy-init -debug -verify
```

After entering your passphrase, successful output will look like:

```
[bacon@mayed schedulers]$ grid-proxy-init -debug -verify
```

```
User Cert File: /home/user/.globus/usercert.pem
```

```
User Key File: /home/user/.globus/userkey.pem
```

```
Trusted CA Cert Dir: /etc/grid-security/certificates
```

```
Output File: /tmp/x509up_u1817
```

```
Your identity: /O=Grid/OU=GlobusTest/OU=simpleCA-mayed.mcs.anl.gov/OU=mcs.anl.gov/CN=User
```

```
Enter GRID pass phrase for this identity:
```

```
Creating proxy .....+++++
```

```
.....+++++
```

```
Done
```

```
Proxy Verify OK
```

```
Your proxy is valid until: Sat Mar 20 03:01:46 2004
```

6. Configure SimpleCA for multiple machines

So far, you have a single machine configured with SimpleCA certificates. Recall that in [Complete setup of GSI](#) a CA setup package was created in `.globus/simpleCA/globus_simple_ca_HASH_setup-0.17.tar.gz`. If you want to use your certificates on another machine, you must install that CA setup package on that machine.

To install it, copy that package to the second machine and run:

```
$GLOBUS_LOCATION/sbin/gpt-build globus_simple_ca_HASH_setup-0.17.tar.gz gcc32dbg
```

Then you will have to perform **setup-gsi -default** from [Sign the host certificate](#).

If you are going to run services on the second host, it will need its own [Host certificates for SimpleCA](#) and grid-mapfile (as described in the basic configuration instructions in [Section 4, "Add authorization"](#)).

You may re-use your user certificates on the new host. You will need to copy the requests to the host where the SimpleCA was first installed in order to sign them.

Appendix D. Deploying in GT 4.2.0

The Globus services can be run either in the standalone Java WS Core container that is installed with GT, or deployed into Tomcat.

1. Deploying into the Java WS Core container

The standalone Java WS Core container can be started and stopped with the provided `globus-start-container` and `globus-stop-container` programs. There are also helper programs (available only with the full GT installation) to start and stop the container detached from the controlling terminal (`globus-start-container-detached` and `globus-stop-container-detached`).

1.1. Deploying and undeploying services

To deploy a service into Java WS Core container use the `globus-deploy-gar` tool. To undeploy a service use `globus-undeploy-gar`.

1.2. Recommended JVM settings for the Java WS Core container

It is recommended to increase the maximum heap size of the JVM when running the container. By default on Sun JVMs a 64MB maximum heap size is used. The maximum heap size can be set using the `-Xmx` JVM option. Example:

```
$ setenv GLOBUS_OPTIONS -Xmx512M $ $GLOBUS_LOCATION/bin/globus-start-container
```

The above example will make the container start with maximum heap size set to 512MB.

It is also recommended to experiment with other JVM settings to improve performance. For example, the `-server` option on Sun JVMs enables a server VM which can deliver better performance for server applications.

2. Deploying into Tomcat

To deploy a Java WS Core installation into Tomcat run:

```
$ cd $GLOBUS_LOCATION $ ant -f share/globus_wsrf_common/tomcat/tomcat.xml
  deploySecureTomcat \ -Dtomcat.dir=<tomcat.dir>
```

Where `<tomcat.dir>` is an *absolute* path to the Tomcat installation directory. Also, `-Dwebapp.name=<name>` can be specified to set the name of the web application under which the installation will be deployed. By default "wsrf" web application name is used. To enable local invocation in Tomcat you must specify `-Dlocal.invocations=true`

The `deploySecureTomcat` task will update an existing Tomcat deployment if Java WS Core was already deployed under the specified web application name. The `redeploySecureTomcat` task can be used instead to overwrite the existing deployment.



Note

Please note that during deployment a subset of the files from Java WS Core installation is copied into Tomcat. Also, the copied files in Tomcat might have different permissions than the originals.

In addition to the above deployment step you will also need to modify the Tomcat `<tomcat_root>/conf/server.xml` configuration file. In particular you will need to add the following configuration entries:

- Tomcat 4.1.x

1. Add a HTTPS Connector in the `<Service name="Tomcat-Standalone">` section and update the parameters appropriately with your local configuration:

```
<Connector className="org.apache.catalina.connector.http.HttpConnector"
  port="8443" minProcessors="5"
  maxProcessors="75" authentic="true"
  secure="true" scheme="https"
  enableLookups="true" acceptCount="10"
  debug="0">
  <Factory className="org.globus.tomcat.catalina.net.HTTPSServerSocketFactory"
    proxy="/path/to/proxy/file"
    cert="/path/to/certificate/file"
    key="/path/to/private/key/file"
    cacertdir="/path/to/ca/certificates/directory"
    encryption="true"/>
</Connector>
```

In the above the `proxy`, `cert`, `key` and `cacertdir` attributes are optional. Furthermore, the `proxy` and the combination of `cert` and `key` attributes are mutually exclusive. The `encryption` attribute is also optional (defaults to `true` if not set).

Important

The credentials and certificate configuration is used only by the connector and is not used by the rest of the web services stack in Globus Toolkit. To configure credentials for use in the toolkit, refer [Security Descriptors Introduction](#).

The `mode` attribute can also be set to specify the connection mode. There are two supported connection modes: `ssl` and `gsi`. The `ssl` mode indicates a regular SSL connection mode. The `gsi` mode indicates a SSL connection mode with transport-level delegation support. The `ssl` mode is the default mode if the `mode` attribute is not specified. Please note that the `gsi` mode is intended for advanced users only.

2. Add a HTTPS Valve in the `<Engine name="Standalone" ... >` section:

```
<Valve
  className="org.globus.tomcat.catalina.valves.HTTPSValve"/>
```

- Tomcat 5.0.x

1. Add a HTTPS Connector in the `<Service name="Catalina">` section and update the parameters appropriately with your local configuration:

```
<Connector
  className="org.globus.tomcat.coyote.net.HTTPSConnector"
  port="8443" maxThreads="150"
  minSpareThreads="25" maxSpareThreads="75"
  autoFlush="true" disableUploadTimeout="true"
  scheme="https" enableLookups="true"
  acceptCount="10" debug="0"
```

```

proxy="/path/to/proxy/file"
cert="/path/to/certificate/file"
key="/path/to/private/key/file"
cacertdir="/path/to/ca/certificates/directory"
encryption="true"/>

```

In the above the `proxy`, `cert`, `key` and `cacertdir` attributes are optional. Furthermore, the `proxy` and the combination of `cert` and `key` attributes are mutually exclusive. The `encryption` attribute is also optional (defaults to `true` if not set).

Important

The credentials and certificate configuration is used only by the connector and is not used by the rest of the web services stack in Globus Toolkit. To configure credentials for use in the toolkit, refer [Security Descriptors Introduction](#).

The `mode` attribute can also be set to specify the connection mode. There are two supported connection modes: `ssl` and `gsi`. The `ssl` mode indicates a regular SSL connection mode. The `gsi` mode indicates a SSL connection mode with transport-level delegation support. The `ssl` mode is the default mode if the `mode` attribute is not specified. Please note that the `gsi` mode is intended for advanced users only.

2. Add a HTTPS Valve in the `<Engine name="Catalina" ... >` section:

```

<Valve
    className="org.globus.tomcat.coyote.valves.HTTPSValve"/>

```

- Tomcat 5.5.x:

1. Add a HTTPS Connector in the `<Service name="Catalina">` section of the Tomcat config file and update the parameters appropriately with your local configuration:

```

<Connector
    className="org.globus.tomcat.coyote.net.HTTPSConnector"
    port="8443" maxThreads="150"
    minSpareThreads="25" maxSpareThreads="75"
    autoFlush="true" disableUploadTimeout="true"
    scheme="https" enableLookups="true"
    acceptCount="10" debug="0"
    protocolHandlerClassName="org.apache.coyote.http11.Http11Protocol"
    socketFactory="org.globus.tomcat.catalina.net.BaseHTTPSServerSocketFactory"
    proxy="/path/to/proxy/file" cert="/path/to/certificate/file"
    key="/path/to/private/key/file"
    cacertdir="/path/to/ca/certificates/directory"
    encryption="true"/>

```

In the above the `proxy`, `cert`, `key` and `cacertdir` attributes are optional. Furthermore, the `proxy` and the combination of `cert` and `key` attributes are mutually exclusive. The `encryption` attribute is also optional (defaults to `true` if not set).

Important

The credentials and certificate configuration is used only by the connector and is not used by the rest of the web services stack in Globus Toolkit. To configure credentials for use in the toolkit, refer [Security Descriptors Introduction](#).

The mode attribute can also be set to specify the connection mode. There are two supported connection modes: `ssl` and `gsi`. The `ssl` mode indicates a regular SSL connection mode. The `gsi` mode indicates a SSL connection mode with transport-level delegation support. The `ssl` mode is the default mode if the mode attribute is not specified. Please note that the `gsi` mode is intended for advanced users only.

2. Add a HTTPS Valve in the `<Engine name="Catalina" ... >` section of the Tomcat config file:

```
<Valve
  className="org.globus.tomcat.coyote.valves.HTTPSValve55"/>
```

Note

It is recommend to run Tomcat with Java 1.4.2+.

2.1. web.xml configuration

You may have to edit `<tomcat.dir>/webapps/wsrif/WEB-INF/web.xml` if you are running Tomcat on a non-default port, that is if not using port 8443 (HTTPS). For example, if you run Tomcat on port 443 using HTTPS then the WSRF servlet entry should be modified to have the following `defaultProtocol` and `defaultPort` parameters:

```
<web-app> ... <servlet>
  <servlet-name>WSRFServlet</servlet-name>
  <display-name>WSRF Container Servlet</display-name>
  <servlet-class> org.globus.wsrif.container.AxisServlet
</servlet-class> <init-param>
  <param-name>defaultProtocol</param-name>
  <param-value>https</param-value> </init-param>
  <init-param> <param-name>defaultPort</param-name>
  <param-value>443</param-value> </init-param>
  <load-on-startup>true</load-on-startup> </servlet>
... </web-app>
```

Alternatively, you can use the `setDefault` Ant task to set the default protocol/port in the `web.xml` file:

```
$ cd $GLOBUS_LOCATION $ ant -f share/globus_wsrif_common/tomcat/tomcat.xml setDefaults \
  -Dtomcat.dir=<tomcat.dir> \
  -DdefaultPort=<port> \
  -DdefaultProtocol=<protocol>
```

Also, by default the `webContext` property is set to the directory name of the web application on the file system. However, sometimes the context under which the web application is published might be different from the directory name of the application. In such cases it is necessary to explicitly configure the published context name in the `web.xml` file. To configure the web application context name set the `webContext` parameter in `web.xml` file. For example (assuming services are published under `http://localhost:8080/foo/services`) the `webContext` should be set to:

```
<web-app> ... <servlet>
  <servlet-name>WSRFServlet</servlet-name> ...
  <init-param> <param-name>webContext</param-name>
  <param-value>foo</param-value> </init-param> ...
  <load-on-startup>true</load-on-startup> </servlet>
... </web-app>
```

2.2. Debugging

2.2.1. Tomcat log files

Please always check the Tomcat log files under the `<tomcat.dir>/logs` directory for any errors or exceptions.

2.2.2. Enabling Log4J debugging

- Tomcat 4.1.x

Copy `$GLOBUS_LOCATION/lib/common/commons-logging-*.jar` files to `<tomcat.dir>/common/lib` directory. Also, copy `<tomcat.dir>/webapps/wsrf/WEB-INF/classes/log4j.properties` file to `<tomcat.dir>/common/classes/` directory. Then configure the Log4j configuration file in `<tomcat.dir>/common/classes/` directory appropriately. The debugging settings will affect all the code in *all* web applications.

- Tomcat 5.0.x, 5.5.x

Copy `$GLOBUS_LOCATION/lib/common/log4j-*.jar` and `$GLOBUS_LOCATION/lib/common/commons-logging-*.jar` files to `<tomcat.dir>/webapps/wsrf/WEB-INF/lib/` directory. Then configure the Log4j configuration file in `<tomcat.dir>/webapps/wsrf/WEB-INF/classes/` directory appropriately. The debugging settings will only affect the web application code.

2.3. Creating WAR file

To create a `.war` of a Java WS Core installation do:

```
$ cd $GLOBUS_LOCATION $ ant -f share/globus_wsrf_common/tomcat/tomcat.xml war
-Dwar.file=<war.file>
```

Where `<war.file>` specifies the *absolute* path of the war file.

Please note that deploying a war file might not be enough to have a working Java WS Core deployment. For example, in some cases the `xalan.jar` must be placed in the endorsed directory of the container.

2.4. Deploying and undeploying services

Assuming Java WS Core is already deployed into Apache Tomcat (as described in [Deploying Java WS Core](#)), use the `globus-deploy-gar` tool with the `-tomcat <tomcat.dir>` option to deploy your GT service directly into Tomcat. Similarly, to undeploy a service, use the `globus-undeploy-gar` tool with the `-tomcat <tomcat.dir>` option to undeploy the service from Tomcat.

Alternatively, to indirectly deploy a service into Tomcat, first deploy the service into a regular GT installation using the `globus-deploy-gar` tool and then redeploy the GT installation into Tomcat (as described in [Deploying Java WS](#)

Core). Similarly, to undeploy a service, first undeploy the service from a regular GT installation using `globus-undeploy-gar` tool and then *redeploy* the GT installation into Tomcat.



Note

Some GT services may not work properly in Tomcat.

3. Deploying into JBoss

To deploy a Java WS Core installation into JBoss (version 4.0.x+) do the following:

1. Run:

```
$ cd $GLOBUS_LOCATION $ ant -f share/globus_wsrf_common/tomcat/jboss.xml
deployJBoss \ -Djboss.dir=<jboss.dir>
```

Where `<jboss.dir>` is an *absolute* path to the JBoss installation directory. Also, `-Dwebapp.name=<name>` can be specified to set the name of the web application under which the installation will be deployed. By default "wsrf" web application name is used.

2. Add a HTTPS Connector and HTTPS Valve:

- a. Add a HTTPS Connector in the `<Service name="Catalina">` section of the Tomcat config file and update the parameters appropriately with your local configuration:

```
<Connector
className="org.globus.tomcat.coyote.net.HTTPSConnector"
port="8443" maxThreads="150"
minSpareThreads="25" maxSpareThreads="75"
autoFlush="true" disableUploadTimeout="true"
scheme="https" enableLookups="true"
acceptCount="10" debug="0"
protocolHandlerClassName="org.apache.coyote.http11.Http11Protocol"
socketFactory="org.globus.tomcat.catalina.net.BaseHTTPSServerSocketFactory"
proxy="/path/to/proxy/file" cert="/path/to/certificate/file"
key="/path/to/private/key/file"
cacertdir="/path/to/ca/certificates/directory"
encryption="true"/>
```

In the above the `proxy`, `cert`, `key` and `cacertdir` attributes are optional. Furthermore, the `proxy` and the combination of `cert` and `key` attributes are mutually exclusive. The `encryption` attribute is also optional (defaults to `true` if not set).



Important

The credentials and certificate configuration is used only by the connector and is not used by the rest of the web services stack in Globus Toolkit. To configure credentials for use in the toolkit, refer [Security Descriptors Introduction](#).

The `mode` attribute can also be set to specify the connection mode. There are two supported connection modes: `ssl` and `gsi`. The `ssl` mode indicates a regular SSL connection mode. The `gsi` mode indicates a

SSL connection mode with transport-level delegation support. The `ssl` mode is the default mode if the mode attribute is not specified. Please note that the `gsi` mode is intended for advanced users only.

- b. Add a HTTPS Valve in the `<Engine name="Catalina" ... >` section of the Tomcat config file:

```
<Valve  
  className="org.globus.tomcat.coyote.valves.HTTPSValve55" />
```



Note

JBoss 4.0.x+ installation with embedded Tomcat is required.

The Tomcat configuration file should be under `<jboss.dir>/default/deploy/jbossweb-tomcat55.sar/server.xml`.

DRAFT

Appendix E. Troubleshooting your installation

The following is a list of links that take you to information about troubleshooting your installation by component

- Common Runtime components
 - [Java WS Core](#)
 - [C WS Core](#)
 - [Python WS Core](#)
 - [XIO](#)
 - [C Common Libraries](#)
 - [CoG jglobus \(Java non-WS Core\)](#)
 - [pyGlobus \(Python non-WS Core\)](#)
- Security components
 - [GSIC](#)
 - [Java WS Authorization and Authentication \(Java WS A&A\)](#)
 - [Community Authorization Service \(CAS\)](#)
 - [Delegation Service](#)
 - [MyProxy](#)
 - [GSI-OpenSSH](#)
- Data Management components
 - [Reliable File Transfer \(RFT\)](#)
 - [GridFTP](#)
 - [Replica Location Service \(RLS\)](#)
 - [WS RLS](#)
 - [Data Replication Service \(DRS\)](#)
- Information Services components
 - [Index Service](#)
 - [Trigger Service](#)
 - [Aggregator Framework](#)

- [WebMDS](#)
- [UsefulRP](#)
- Execution Management components
 - [GRAM4](#)
 - [GridWay](#)

DRAFT

Appendix F. Detailed Configuration by Component

The following is a list of links that take you to information about detailed configuration for each component.

- Common Runtime components
 - [Java WS Core](#)
 - [Python WS Core](#)
 - [XIO](#)
 - [CoG jglobus \(Java non-WS Core\)](#)
 - [pyGlobus \(Python non-WS Core\)](#)
- Security components
 - [GSI C](#)
 - [GSI Java](#)
 - [Java WS Authentication and Authorization \(Java WS A&A\)](#)
 - [Community Authorization Service \(CAS\)](#)
 - [Delegation Service](#)
 - [MyProxy](#)
 - [GSI-OpenSSH](#)
- Data Management components
 - [Reliable File Transfer \(RFT\)](#)
 - [GridFTP](#)
 - [Replica Location Service \(RLS\)](#)
 - [WS RLS](#)
 - [Data Replication Service \(DRS\)](#)
- Information Services components
 - [Configuring WS MDS](#)
 - [Index Service](#)
 - [Trigger Service](#)
 - [Aggregator Framework](#)

- WebMDS
- UsefulRP
- Execution Management components
 - GRAM4
 - GRAM4

DRAFT

Appendix G. Security Considerations in GT 4.2.0

1. Common Runtime

1.1. Java WS Core

1.1.1. Permissions of service configuration files

The service configuration files such as *jndi-config.xml* or *server-config.wsdd* (located under `$GLOBUS_LOCA-TION/etc/<gar>/` directory) may contain private information such as database passwords, etc. Ensure that these configuration files are only readable by the user that is running the container. The deployment process automatically sets the permissions of the *jndi-config.xml* and *server-config.wsdd* files as user readable only. However, this might not work correctly on all platforms and this does not apply to any other configuration files.

1.1.2. Permissions of persistent data

The services using subscription persistence API or other basic persistence helper API will store all or part of its persistent data under the `~/ .globus/persisted` directory. Ensure that the entire `~/ .globus/persisted` directory is only readable by the user running the container.

1.1.3. Invocation of non-public service functions

A client can potentially invoke a service function that is not formally defined in the *WSDL* but it is defined in the service implementation class. There are two ways to prevent this from happening:

1. Define all service methods in your service class as either `private` or `protected`.
2. Configure appropriate *allowedMethods* or *allowedMethodsClass* parameter in the service deployment descriptor (please see [Configuring Java WS Core](#) for details).

1.2. C WS Core

1.2.1. Supported Protocols

C WS-Core supports secure transport (https) and secure message (just X509 signing, not encryption).

1.2.2. Secure Transport

With secure transport, the entire container must be run over an https transport. This is done by default for the C container. If the user does not want security in the container, or wants to use secure message instead of secure transport, they should use the `-nosec` argument to `globus-wsc-container`.

For clients, the secure transport is enabled if the contact URI contains the 'https' scheme instead of 'http', so the client doesn't have to enable or disable it explicitly.

1.3. Security Considerations for Python WS Core

Individual services can be configured with or without message security, but transport security is a characteristic of the entire container (either using ssl or plain tcp). Authentication and authorization of clients is performed using a callback mechanism.

1.3.1. Transport Security

Simply edit the file `config.txt` where the executable is being run and turn on ssl.

By default, pyGridWare will look in the user's home directory for the `.globus/usercert.pem` and `.globus/user-key.pem` files.

To use the grid proxy generated by `grid-proxy-init`, just specify the `/tmp/x509***` as the certfile and keyfile.

Example pyGridWare/bin/config.txt

```
[security]
ssl = 1
certfile =
keyfile =
```

1.4. Security considerations for XIO

Globus XIO is a framework for creating network protocols. Several existing protocols, such as TCP, come built into the framework. XIO itself introduces no known security risks. However, all network applications expose systems to the risks inherent when outsiders can connect to them. Also included in the XIO distribution is the GSI driver, which provides a driver that allows for secure connections.

1.5. Security considerations for CoG jGlobus

1.5.1. Functions that execute an external program

Under some circumstances, the `org.globus.util.Util.setFilePermissions()` and the `org.globus.util.ConfigUtil.getUID()` functions execute an external program; thus, its behavior is influenced by environment variables such as the caller's PATH and the environment variables that control dynamic loading. Care should be used if calling these functions from a program that will be run as a Unix setuid program, or in any other manner in which the owner of the Unix process does not completely control its runtime environment.

1.5.2. Permissions of proxy files

Since Java does not provide an API for setting the permissions of a file, the Java CoG Kit will attempt to execute the `/bin/chmod` program in the background to set the permissions of the given file. If that program cannot be executed for any reason or fails to execute correctly, a proxy file might end up with incorrect file permissions (depending on umask setting). Usually a warning will be displayed if that occurs (especially on Windows since `/bin/chmod` is not supported on that platform).

1.6. Security Considerations for PyGlobus

pyGlobus has a security module which allows for proxy creation, signing, encryption, and the creation and inquiry of security contexts. Care must be taken when developing applications which use GSI to ensure that authentication information will not be compromised. When creating a security context, one must ensure that the context will have the

properties that they desire. For example, should the context use confidentiality or integrity? These concerns are not specific to pyGlobus but rather to any application developer who is using low level security APIs.

2. Security

2.1. Security considerations for GSI C

- During host authorization, the toolkit treats DNs "hostname-*.edu" as equivalent to "hostname.edu". This means that if a service was setup to do host authorization and hence accept the certificate "hostname.edu", it would also accept certificates with DNs "hostname-*.edu".

The feature is in place to allow a multi-homed host following a "hostname-interface" naming convention, to have a single host certificate. For example, host "grid.test.edu" would also accept likes of "grid-1.test.edu" or "grid-foo.test.edu".



Note

The wildcard character "*" matches only name of the host and not domain components. This means that "hostname.edu" will not match "hostname-foo.sub.edu", but will match "host-foo.edu".



Note

If a host was set up to accept "hostname-1.edu", it will not accept any of "hostname-*.edu".

A [bug](#)¹ has been opened to see if this feature needs to be modified.

2.2. Security considerations for Java WS A&A

2.2.1. Security considerations for authorization

2.2.1.1. Client side authorization

Client authorization of the server is done after the completion of the operation when GSI Secure Message authentication is used. If you require client authorization to be done prior, use GSI Secure Conversation or GSI *Transport security*.

2.2.1.2. Host authorization

During host authorization, the toolkit treats DNs "hostname-*.edu" as equivalent to "hostname.edu". This means that if a service was setup to do host authorization and hence accept the certificate "hostname.edu", it would also accept certificates with DNs "hostname-*.edu".

The feature is in place to allow a multi-homed host following a "hostname-interface" naming convention, to have a single host certificate. For example, host "grid.test.edu" would also accept likes of "grid-1.test.edu" or "grid-foo.test.edu".



Note

The wildcard character "*" matches only name of the host and not domain components. This means that "hostname.edu" will not match "hostname-foo.sub.edu", but will match "host-foo.edu".

¹ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2969

 **Note**

If a host was set up to accept "hostname-1.edu", it will not accept any of "hostname-*.edu".

A [bug](#)² has been opened to see if this feature needs to be modified.

2.2.2. Security considerations for Message/Transport-level Security

2.2.2.1. File permissions

The Java security code currently does not enforce secure permissions and, implicitly, file ownership requirements on any of the security related files, e.g. configuration and credential files. It is thus important that administrators ensure that the relevant files have correct permissions and ownership. Permissions should generally be as restrictive as possible, i.e. *private keys* should be readable only by the file owner and other files should be writable by owner only, and the files should generally be owned by the globus user (the requirements that the C code enforces are documented in [Configuring GSI](#)).

Also refer to [Section 5, "Known Problems"](#) for details on any other open issues.

2.3. Security Considerations for CAS

- The database username/password is stored in the service configuration file and the test properties file. Ensure correct permissions to protect the information.

2.4. Delegation Service Security Considerations

2.4.1. Key Pair Reuse

The current design re-uses the keys associated with the Delegation Service for each of the *proxy certificates* delegated to it. During a security review, it was pointed out that while this was fine from a cryptographic standpoint, compromising this single long-lived key pair may significantly extend the time for which a single intrusion (presuming an exploitable security flaw making the intrusion possible) is effective.

This can be remedied by either frequently regenerating the key pair used by the Delegation Service, which can be accomplished with a simple cron job, or by generating a new key pair for each new delegation. The latter of these approaches requires changes to the design and may be adopted in future versions of the toolkit. For the time being, we recommend the former approach should this issue concern you.

2.4.2. Authorizing Server prior to delegation

The delegation client that is distributed with the toolkit allows for delegation of credentials even when no authorization of the server is done. Also, when using secure message authentication, the authorization of the server is done after the completion of the operation. These two scenarios could lead to the delegation of credentials to a malicious server.

To prevent this, users should use secure *transport* (HTTPS) or GSI Secure Conversation and appropriate client-side authorization.

² http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2969

2.5. MyProxy Security Considerations

You should choose a well-protected host to run the myproxy-server on. Consult with security-aware personnel at your site. You want a host that is secured to the level of a Kerberos KDC, that has limited user access, runs limited services, and is well monitored and maintained in terms of security patches.

For a typical myproxy-server installation, the host on which the myproxy-server is running must have `/etc/grid-security` created and a *host certificate* installed. In this case, the myproxy-server will run as root so it can access the host certificate and key.

2.6. GSI-OpenSSH Security Considerations

GSI-OpenSSH is a modified version of [OpenSSH](http://www.openssh.org/)³ and includes full OpenSSH functionality. For more information on OpenSSH security, see the [OpenSSH Security](http://www.openssh.org/security.html)⁴ page.

3. Data Management

3.1. Reliable Transfer Service (RFT) Security Considerations

3.1.1. Permissions of service configuration files

The service configuration files such as `jndi-config.xml` and `server-config.wsdd` (located under `etc/<gar>/` directory) contain private information such as database passwords and usernames. Ensure that these configuration files are only readable by the user that is running the container.

The deployment process automatically sets the permissions of `jndi-config.xml` and `server-config.wsdd` as user readable only. However, this might not work correctly on all platforms and this does not apply to any other configuration files.

3.1.2. Access of information stored in the database

RFT stores the transfer requests in a database. Proper security measures need to be taken to protect the access of the data by granting/revoking appropriate permissions on tables that are created for RFT use and other steps that are appropriate and consistent with site specific security measures.

3.1.3. Permissions of persistent data

RFT uses the subscription persistence API from the GT4 core to store all of its subscription data under the `~/ .globus/persisted` directory. Ensure that the entire `~/ .globus/persisted` directory is only readable by the user running the container.

³ <http://www.openssh.org/>

⁴ <http://www.openssh.org/security.html>

3.2. Security Considerations

3.2.1. Ways to configure your server

As discussed in [Section 2, “Types of configurations”](#), there are three ways to configure your GridFTP server: the default configuration (like any normal FTP server), separate (split) process configuration and striped configuration. The latter two provide greater levels of security as described [here](#).

3.2.2. New authentication option

There is a new authentication option available for GridFTP in GT 4.2.0:

- **SSH Authentication** Globus GridFTP now supports SSH based authentication for the control channel. In order for this to work:
 - Configure server to support SSH authentication,
 - Configure client(globus-url-copy) to support SSH authentication,
 - Use sshftp:// urls in globus-url-copy

For more information, see [Section 4, “SSHFTP \(GridFTP-over-SSH\)”](#).

3.2.3. 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_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](#)⁵.

3.3. Replica Location Service (RLS) Security Considerations

Security recommendations include:

- *Dedicated User Account:* It is recommended that users create a dedicated user account for installing and running the RLS service (e.g., `globus` as recommended in the general GT installation instructions). This account may be used to install and run other services from the Globus Toolkit.
- *Key and Certificate:* It is recommended that users do not use their `hostkey` and `hostcert` for use by the RLS service. Create a `containerkey` and `containercert` with permissions 400 and 644 respectively and owned by the `globus` user. Change the `rlskeyfile` and `rlscertfile` settings in the RLS configuration file (`$GLOBUS_LOCATION/etc/globus-rls-server.conf`) to reflect the appropriate filenames.
- *LRC and RLI Databases:* Users must ensure security of the RLS data as maintained by their chosen database management system. Appropriate precautions should be made to protect the data and access to the database. Such precautions may include creating a user account specifically for RLS usage, encrypting database users' passwords, etc.
- *RLS Configuration:* It is recommended that the RLS configuration file (`$GLOBUS_LOCATION/etc/globus-rls-server.conf`) be owned by and accessible only by the dedicated user account for RLS (e.g., `globus` account per above recommendations). The file contains the database user account and password used to access the LRC and RLI databases along with important settings which, if tampered with, could adversely affect the RLS service.

⁵ <http://www.globus.org/toolkit/security/firewalls/>

3.4. WS Replica Location Service (WS RLS) Security Considerations

Security recommendations include:

- The following paragraph describes our INITIAL IMPLEMENTATION in this first look of the WS RLS interface -- THIS IS NOT INTENDED TO BE A FINAL SOLUTION.

Users of the WS RLS authenticate themselves to the WS RLS in the usual manner based on their credential. The WS RLS uses the WSAA and Authorization Framework to make authorization decisions. Then, the WS RLS uses a single certificate and key file to identify itself to the RLS irregardless of the user accessing the WS RLS. Thus users that can access the WS RLS are given the fixed WS RLS identity to access the RLS. Users accustomed to using RLS may not feel comfortable with this approach -- if it may be an issue for your environment, we suggest that you not use it with your production RLS. If you think of WS RLS as the gating interface to the RLS, as you should, then you should apply the appropriate authorization restrictions at the WS RLS level, which can be done using WSAA. This will in effect achieve a level of authorization similar to that of the RLS.

3.5. Data Replication Service (DRS) Security Considerations

3.5.1. Service configuration files

The service configuration files such as the JNDI configuration file, `jndi-config.xml`, and the Web service deployment descriptor, `server-config.wsdd`, located in the `$GLOBUS_LOCATION/etc/globus_wsrf_replicator` directory, contain sensitive information such as database username and password. It is important to ensure that these files are readable only by the system administrator that is responsible for the container. During deployment, the permissions on these files are adjusted automatically, however, you should verify the permissions to ensure that they have been correctly set for your specific platform.

3.5.2. Delegated proxy credential files

Creating a Replicator requires that the user supply a delegated credential to the DRS during the initial creation request. The service retrieves the delegated credential from the Delegation Service and stores it on the file system. As part of the DRS configuration (see installation and configuration instructions), the user selects a directory to use for storage of delegated credentials. The default setting is for the DRS to store the file in the system's designated temporary directory (e.g., `/tmp` on many platforms). The service sets the permissions on the temporary file such that it can only be accessed by the user account used to run the container.

4. Information Services

4.1. WS MDS Aggregator Services (Index Service and Trigger Service) Security Considerations

By default, the *aggregator sources* do not use authentication credentials -- they retrieve information using anonymous SSL authentication or no authentication at all, and thus retrieve only publicly-available information. If a user or administrator changes that configuration so that a service's aggregator source uses credentials to acquire non-privileged data, then that user or administrator must configure the service's aggregator sink to limit access to authorized users.

4.2. WebMDS Security Considerations

By default, the WebMDS plugins distributed as part of the Toolkit do not use authentication credentials -- they retrieve information using anonymous SSL authentication or no authentication at all, and thus retrieve only publicly-available information.

The `ResourcePropertyNodeSource` and `ResourcePropertyQueryNodeSource` plugins can be configured either to allow users to specify what resources they want to query or to only allow users to query resources pre-configured by the web administrator. The standard WebMDS deployment allows users to specify the resources they want to query; to disallow this (for example, to ensure that people don't use your site's bandwidth to view information about some other site's services), remove the files `$GLOBUS_LOCATION/lib/webmads/conf/openEndedRP` and `$GLOBUS_LOCATION/lib/webmads/conf/openEndedQuery`.

4.3. UsefulRP Security Considerations

Security recommendations for this component include:

- FIXME list

5. Execution Management

5.1. Security Considerations

No special security considerations exist at this time.

Appendix H. Usage Statistics

The following components collect usage statistics as outlined here (along with information about how to opt-out): [Usage Statistics in GT](#)¹

1. Common Runtime Usage Statistics

1.1. Usage statistics sent by Java WS Core

The following usage statistics are sent by Java WS Core by default in a UDP packet (in addition to the Java WS Core component code, packet version, timestamp, and the source IP address):

- On container startup:
 - container id - random number
 - container type - standalone, servlet, or unknown
 - event type - container startup
 - list of deployed services - service names only
- On container shutdown:
 - container id - random number
 - container type - standalone, servlet, or unknown
 - event type - container shutdown
 - list of activated services - service names only
 - container uptime

If you wish to disable this feature, please see the "Usage Statistics Configuration" section of [Section 2.4, "Usage Statistics Configuration"](#) for instructions.

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

1.2. Usage statistics sent by C WS Core

The following usage statistics are sent by C WS Core by default in a UDP packet :

- On container start
 - ip address of container
 - container id - random number
 - event type - container startup

¹ ../Usage_Stats.html

² http://www.globus.org/toolkit/docs/4.2/4.2.0/Usage_Stats.html

- list of deployed service names
- On container shut down
 - ip address of container
 - container id - random number
 - event type - container shutdown
 - list of activated services

It sends it at container startup (globus-wsc-container) and receipt of that packet tells us that the container started.

If you wish to disable this feature, you can set the following environment variable before running the C container:

```
export GLOBUS_USAGE_OPTOUT=1
```

By default, these usage statistics UDP packets are sent to `usage-stats.globus.org:4180` but can be redirected to another host/port or multiple host/ports with the following environment variable:

```
export GLOBUS_USAGE_TARGETS="myhost.mydomain:12345 myhost2.mydomain:54321"
```

You can also dump the usage stats packets to stderr as they are sent (although most of the content is non-ascii). Use the following environment variable for that:

```
export GLOBUS_USAGE_DEBUG=MESSAGES
```

Also, please see our [policy statement](#)³ on the collection of usage statistics.

2. Data Management Usage Statistics

2.1. Usage statistics sent by RFT

The following usage statistics are sent by default in a UDP packet at the end of life time of each RFT Resource (or when a RFT resource is destroyed).

- Total number of files transferred by RFT since RFT was installed
- Total number of bytes transferred by RFT since RFT was installed
- Total number of files transferred in this RFT Resource
- Total number of bytes transferred in this RFT Resource
- Creation time of this RFT Resource
- Factory Start Time

³ ../../Usage_Stats.html

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 the RFT component. Nevertheless, if you wish to disable this feature, please see the "Usage Statistics Configuration" section of [Configuring Java WS Core](#) for instructions.

Also, please see our [policy statement](#)⁴ on the collection of usage statistics.

2.2. 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](#)⁵ 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](#)⁶ on the collection of usage statistics.

2.3. RLS-specific usage statistics

The following usage statistics are sent by RLS Server by default in a UDP packet:

- Component identifier
- Usage data format identifier
- Time stamp

⁴ ../../Usage_Stats.html

⁵ ../../Usage_Stats.html

⁶ ../../Usage_Stats.html

- Source IP address
- Source hostname (to differentiate between hosts with identical private IP addresses)
- Version number
- Uptime
- *LRC* service indicator
- *RLI* service indicator
- Number of *LFNs*
- Number of *PFNs*
- Number of Mappings
- Number of RLI LFNs
- Number of RLI LRCs
- Number of RLI Senders
- Number of RLI Mappings
- Number of threads
- Number of connections

The RLS sends the usage statistics at server startup, server shutdown, and once every 24 hours when the service is running.

If you wish to disable this feature, you can set the following environment variable before running the RLS:

```
export GLOBUS_USAGE_OPTOUT=1
```

By default, these usage statistics UDP packets are sent to `usage-stats.globus.org:4180` but can be redirected to another host/port or multiple host/ports with the following environment variable:

```
export GLOBUS_USAGE_TARGETS="myhost.mydomain:12345 myhost2.mydomain:54321"
```

You can also dump the usage stats packets to `stderr` as they are sent (although most of the content is non-ascii). Use the following environment variable for that:

```
export GLOBUS_USAGE_DEBUG=MESSAGES
```

Also, please see our [policy statement](#)⁷ on the collection of usage statistics.

⁷ ../../Usage_Stats.html

2.4. WS RLS-specific usage statistics

The WS RLS does not collect usage statistics in addition to what the RLS collects. Please consult the RLS documentation to familiarize yourself with usage statistics collected by it.

Also, please see our [policy statement](#)⁸ on the collection of usage statistics.

3. Execution Management Usage Statistics

3.1. GRAM4-specific usage statistics

The following usage statistics are sent by default in a UDP packet (in addition to the GRAM component code, packet version, timestamp, and source IP address) at the end of each job (i.e. when Done, Failed, UserTerminateDone or UserTerminateFailed state is entered).

- job creation timestamp (helps determine the rate at which jobs are submitted)
- *scheduler* type (Fork, *PBS*, *LSF*, *Condor*, etc...)
- jobCredentialEndpoint present in *RSL* flag (to determine if server-side user proxies are being used)
- fileStageIn present in RSL flag (to determine if the staging in of files is used)
- fileStageOut present in RSL flag (to determine if the staging out of files is used)
- fileCleanUp present in RSL flag (to determine if the cleaning up of files is used)
- CleanUp-Hold requested flag (to determine if streaming is being used)
- job type (Single, Multiple, MPI, or Condor)
- gt2 error code if job failed (to determine common scheduler script errors users experience)
- fault class name if job failed (to determine general classes of common faults users experience)

If you wish to disable this feature, please see the "Usage Statistics Configuration" section of [Configuring Java WS Core](#) for instructions.

Also, please see our [policy statement](#)⁹ on the collection of usage statistics.

⁸ ../../Usage_Stats.html

⁹ ../../Usage_Stats.html

Glossary

A

aggregator source A Java class that implements an interface (defined as part of the Aggregator Framework) to collect XML-formatted data. WS MDS contains three aggregator sources: the query aggregator source, the subscription aggregator source, and the execution aggregator source.

C

CA Certificate The CA's certificate. This certificate is used to verify signature on certificates issued by the CA. GSI typically stores a given CA certificate in `/etc/grid-security/certificates/<hash>.0`, where `<hash>` is the hash code of the CA identity.

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.

Condor A job scheduler mechanism supported by GRAM. See <http://www.cs.wisc.edu/condor/> for more information.

container Also referred to as the "hosting environment." Provides a common runtime environment for web services. It manages the execution of services and resources, and manages their lifecycles. Provides security and data persistence infrastructure, and other functionality such as managed threading and registry.

A default "standalone" container is provided with a default GT installation.

G

GAA configuration file A file that configures the Generic Authorization and Access control GAA libraries. When using GSI, this file is typically found in `/etc/grid-security/gsi-gaa.conf`.

grid map file A file containing entries mapping certificate subjects to local user names. This file can also serve as a access control list for GSI enabled services and is typically found in `/etc/grid-security/grid-mapfile`. For more information see the Gridmap section [here](#).

grid security directory The directory containing GSI configuration files such as the GSI authorization callout configuration and GAA configuration files. Typically this directory is `/etc/grid-security`. For more information see [this](#).

Grid Security Infrastructure (GSI) GSI stands for Grid Security Infrastructure and is used to describe the original infrastructure of GT security, which is comprised of SSL, PKI and proxy certificates.

GSI authorization callout configuration file A file that configures authorization callouts to be used for mapping and authorization in GSI enabled services. When using GSI this file is typically found in `/etc/grid-security/gsi-authz.conf`.

H

- host certificate** An EEC belonging to a host. When using GSI this certificate is typically stored in `/etc/grid-security/hostcert.pem`. For more information on possible host certificate locations see the [GSI C Developer's Guide](#).
- host credentials** The combination of a host certificate and its corresponding private key.

J

- jndi-config.xml** It is an XML-based configuration file used to populate the container registry accessible via the JNDI API. See in the [Java WS Core Developer's Guide](#) for details.

L

- Local Replica Catalog (LRC)** Stores mappings between logical names for data items and the target names (often the physical locations) of replicas of those items. Clients query the LRC to discover replicas associated with a logical name. Also may associate attributes with logical or target names. Each LRC periodically sends information about its logical name mappings to one or more RLIs.

See also [RLI](#)⁶.

- logical file name** A unique identifier for the contents of a file.

- LSF** A job scheduler mechanism supported by GRAM.

For more information, see <http://www.platform.com/Products/Platform.LSF.Family/Platform.LSF/>⁷.

P

- Portable Batch System (PBS)** A job scheduler mechanism supported by GRAM. For more information, see <http://www.openpbs.org>.

- physical file name** The address or the location of a copy of a file on a storage system.

- private key** The private part of a key pair. Depending on the type of certificate the key corresponds to it may typically be found in `$HOME/.globus/userkey.pem` (for user certificates), `/etc/grid-security/hostkey.pem` (for host certificates) or `/etc/grid-security/<service>/<service>key.pem` (for service certificates).

For more information on possible private key locations see [this](#).

- proxy certificate** A short lived certificate issued using a EEC. A proxy certificate typically has the same effective subject as the EEC that issued it and can thus be used in its place. GSI uses proxy certificates for single sign on and delegation of rights to other entities.

⁶ #rli

⁷ <http://www.platform.com/Products/Platform.LSF.Family/Platform.LSF/>

For more information about types of proxy certificates and their compatibility in different versions of GT, see <http://dev.globus.org/wiki/Security/ProxyCertTypes>.

proxy credentials

The combination of a proxy certificate and its corresponding private key. GSI typically stores proxy credentials in `/tmp/x509up_u<uid>`, where `<uid>` is the user id of the proxy owner.

R

Replica Location Index (RLI)

Collects information about the logical name mappings stored in one or more Local Replica Catalogs (LRCs) and answers queries about those mappings. Each RLI periodically receives updates from one or more LRCs that summarize their contents.

Resource Specification Language (RSL)

Term used to describe a GRAM job for GT2 and GT3. (Note: This is not the same as RLS - the Replica Location Service)

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.

server-config.wsdd

Axis server-side WSDD configuration file. It contains information about the services, the type mappings and various handlers.

service credentials

The combination of a service certificate and its corresponding private key.

T

transport-level security

Uses transport-level security (TLS) mechanisms.

U

user credentials

The combination of a user certificate and its corresponding private key.

W

Web Services Description Language (WSDL)

WSDL is an XML document for describing Web services. Standardized binding conventions define how to use WSDL in conjunction with SOAP and other messaging substrates. WSDL interfaces can be compiled to generate proxy code that

constructs messages and manages communications on behalf of the client application. The proxy automatically maps the XML message structures into native language objects that can be directly manipulated by the application. The proxy frees the developer from having to understand and manipulate XML. See the WSDL 1.1 specification¹⁵ for details.

DRAFT

¹⁵ <http://www.w3.org/TR/wsdl>