

# GT4 Admin Guide

**GT4 Admin Guide**  
Published February 2005

# Table of Contents

<b>1. Introduction</b> .....	<b>1</b>
<b>2. Before you begin</b> .....	<b>3</b>
<b>3. Software Prerequisites</b> .....	<b>5</b>
Required software .....	5
Optional software.....	5
Platform Notes.....	5
Apple MacOS X.....	5
Debian .....	5
Fedora Core .....	5
FreeBSD.....	5
HP/UX.....	5
IBM AIX .....	6
Red Hat .....	6
Sun Solaris .....	6
SuSE Linux.....	6
Windows .....	6
<b>4. Installing GT 3.9.5</b> .....	<b>9</b>
<b>5. Basic Security Configuration</b> .....	<b>11</b>
Set environment variables.....	11
Obtain host certificates .....	11
Request a certificate from an existing CA .....	11
SimpleCA.....	12
Low-trust certificate .....	12
Make the host credentials accessible by the container.....	12
Add authorization.....	12
Verify Basic Security .....	13
<b>6. Security Overview</b> .....	<b>15</b>
Configuration.....	15
Configuring Globus to Trust a Particular Certificate Authority.....	15
Configuring Globus to Create Appropriate Certificate Requests .....	16
Requesting Service Certificates .....	17
Host Certificates and Client-side Authorization .....	17
Specifying Identity Mapping Information .....	17
GSI File Permissions Requirements.....	18
Troubleshooting.....	19
<b>7. GridFTP Configuration</b> .....	<b>21</b>
Introduction .....	21
Deploying the GridFTP Server.....	21
Running in daemon mode .....	21
Running under inetd or xinetd .....	21
Remote data-nodes and striped operation .....	21
Testing.....	22
Security Considerations .....	22
Troubleshooting.....	22
<b>8. Webservices container</b> .....	<b>25</b>
Starting the container.....	25
Service configuration overview .....	25
Container configuration .....	26
Configuration Profiles .....	27
<b>9. RFT Configuration</b> .....	<b>29</b>
Introduction .....	29
Configuring .....	29
Required configuration: configuring the PostgreSQL database .....	29
Security Considerations .....	30
Troubleshooting.....	30

<b>10. WS GRAM Configuration .....</b>	<b>31</b>
Introduction .....	31
Local Prerequisites .....	31
Host credentials .....	31
GRAM service account .....	31
Gridmap authorization of user account .....	31
Functioning sudo .....	32
Local scheduler .....	32
RFT Dependency .....	32
Configuring .....	32
Configuration settings.....	32
Setting up service credentials .....	32
Enabling Local Scheduler Adapter .....	33
Configuring sudo.....	33
Extra steps for non-default installation .....	34
Testing .....	36
Security Considerations .....	36
Troubleshooting.....	36
<b>11. Configuring GSI-OpenSSH.....</b>	<b>37</b>
Introduction .....	37
Building and Installing.....	37
Configuring .....	37
Configuration settings .....	37
System clocks .....	37
Deploying .....	38
Testing .....	39
<b>12. Configuring MyProxy .....</b>	<b>41</b>
Introduction .....	41
Configuring .....	41
Configuration settings .....	41
Configuring a MyProxy server installation .....	41
Deploying .....	42
Testing.....	42
Security Considerations .....	42
<b>13. Configuring CAS.....</b>	<b>45</b>
Introduction .....	45
Configuring .....	45
Deploying .....	45
Obtaining credentials for the CAS server .....	45
Database installation and configuration .....	45
Installing the database .....	45
Initializing the CAS database.....	46
Testing .....	46
Testing the backend database module .....	46
Testing CAS service module .....	47
Example of CAS Server Administration .....	48
1. Adding a user group .....	49
2. Adding a trust anchor .....	49
3. <i>Adding users</i> .....	49
4. <i>Adding users to a user group</i> .....	50
4. <i>Adding a new FTP server</i> .....	50
4. Creating an object group .....	50
5. <i>Adding members to an object group</i> .....	51
6. Adding service types.....	51
7. Adding action mappings.....	51
8. <i>Granting permissions</i> .....	51
Security Considerations .....	52

<b>A. Installing SimpleCA .....</b>	<b>53</b>
Create users .....	53
Run the setup script .....	53
2.1 Configure the subject name.....	53
Configure the CA's email .....	54
Configure the expiration date .....	54
Enter a passphrase.....	54
Confirm generated certificate .....	54
Complete setup of GSI .....	55
Host certificates .....	56
3.1 Request a host certificate .....	56
Sign the host certificate .....	56
User certificates.....	56
Request a user certificate .....	56
Sign the user certificate .....	57
Verify the SimpleCA certificate installation .....	57
Configure SimpleCA for multiple machines.....	57



## Chapter 1. Introduction

This guide is the starting point for everyone who wants to install Globus Toolkit 3.9.5. 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 (MDS4).

This admin guide is a first attempt to create a DocBook version of the Admin Guide. It is also available as a PDF<sup>1</sup>. However, the import into DocBook is not yet complete, so some information is only contained in the component-specific admin guides. Any section with the word "import" can be found in the component-specific documentation. The master list of documentation is here<sup>2</sup>.

**Important:** *Important:* We highly recommend that you fully read through this installation guide, then the system administrator's guide of the component of your choice before attempting the installation.

### Notes

1. [admin.pdf](#)
2. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/toc\\_all.html](http://www-unix.globus.org/toolkit/docs/development/3.9.5/toc_all.html)



## Chapter 2. Before you begin

Before you start installing the Globus Toolkit 3.9.5, 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-webservices implementations of Security, GridFTP, Resource Management (GRAM), Replica Location Service, and Information Services (MDS2). These all run on Unix platforms only.

Additionally, there are WSRF implementations of Security, Resource Management (GRAM), Reliable File Transfer (RFT), and Information Services (Index). 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. Before you begin*

## Chapter 3. Software Prerequisites

### Required software

- Globus Toolkit installer, from Globus Toolkit development download page<sup>1</sup>
- JDK 1.4.2+ from Sun<sup>2</sup>, IBM<sup>3</sup> or BEA<sup>4</sup> (do not use GCJ<sup>5</sup>).
- Ant 1.5.1+<sup>6</sup>. Do not use the ant distributed with Fedora Core 2.
- C compiler. If gcc<sup>7</sup>, avoid version 3.2. 3.2.1 and 2.95.x are okay.
- GNU tar<sup>8</sup>
- GNU Make<sup>9</sup>
- JDBC compliant database. For instance, postgres<sup>10</sup> 7.1+
- gpt-3.2autotools2004 (shipped with the installers, but required if building standalone GPT bundles/packages)

### Optional software

- IODBC<sup>11</sup> (required for RLS)
- Tomcat<sup>12</sup> (optional for Java WS Core, required for WebMDS)

### Platform Notes

#### Apple MacOS X

Use gcc32dbg or gcc32 as flavor

#### Debian

Some kernel/libc combinations trigger a threading problem. See bug #2194<sup>13</sup>. The workaround is to set LD\_ASSUME\_KERNEL=2.2.5 in your environment.

#### Fedora Core

Fedora Core 2 ships with a broken ant. Install your own ant from <http://ant.apache.org><sup>14</sup> and either remove the ant RPM or edit /etc/ant.conf, setting ANT\_HOME to your own ant installation.

#### FreeBSD

No known issues.

#### HP/UX

GPT's MD5.xs needs to be patched at line 52 with

```
./gpt-3.2autotools2004/support/Digest-MD5-2.20/MD5.xs:
```

## Chapter 3. Software Prerequisites

```
+ #define na PL_na  
+ #define dowarn PL_dowarn
```

Specify `--with-flavor=vendorcc32` on the configure line. GNU tar 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

### IBM AIX

Supported flavors are `vendorcc32dbg/vendorcc32` and `vendorcc64dbg/vendorcc64` using the Visual Age compilers (xlc). No gcc flavors are supported.

`vendorcc64dbg/vendorcc64` will be supported for 4.0 final but currently there are a few smaller problems with the installer which stops the 64bit install to be easy.

GNU tar and make is required before the IBM ones in the PATH.

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

### Red Hat

No known issues.

### 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.

WS-C does not build correctly. This should be fixed for 4.0 final.

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

### SuSE Linux

No known issues.

## Windows

Only Java-only components will build. Please choose the Java WS Core-only download and follow the instructions in the Java WS Core System Administrator's Guide<sup>16</sup>.

## Notes

1. <http://www-unix.globus.org/toolkit/downloads/development/>
2. <http://java.sun.com/j2se>
3. <http://www.ibm.com/developerworks/java/jdk>
4. <http://www.bea.com/framework.jsp?CNT=index.htm&FP=/content/products/jrockit>
5. <http://gcc.gnu.org/java/>
6. <http://jakarta.apache.org/ant>
7. <http://gcc.gnu.org>
8. <http://www.gnu.org/software/tar/tar.html>
9. <http://www.gnu.org/software/make/>
10. <http://www.postgresql.org>
11. <http://www.iodbc.org/>
12. <http://jakarta.apache.org/tomcat/>
13. [http://bugzilla.globus.org/globus/show\\_bug.cgi?id=2194](http://bugzilla.globus.org/globus/show_bug.cgi?id=2194)
14. <http://ant.apache.org/>
15. <http://www.gridpackagingtools.org/book/latest-stable/ch01s07.html>
16. <http://www-unix.globus.org/toolkit/docs/development/3.9.5/common/javawscore/admin/index.I>



## Chapter 4. Installing GT 3.9.5

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-3.9.5
# chown globus:globus /usr/local/globus-3.9.5
```

**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 Chapter 3.

**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 the Section called *Platform Notes* in Chapter 3 if your OS includes ant already. Your `/etc/ant.conf` is probably configured to use gcj, which will fail to compile the Toolkit.

3. In this guide we will assume that you are installing to `/usr/local/globus-3.9.5`, but you may replace `/usr/local/globus-3.9.5` with whatever directory you wish to install to.

As the globus user, run:

```
globus$ export GLOBUS_LOCATION=/usr/local/globus-3.9.5
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:

```
--enable-prewsmds      Build pre-webservices mds. Default is disabled.
--enable-wsgram-condor Build GRAM Condor scheduler interface. De-
fault is disabled.
--enable-wsgram-lsf    Build GRAM LSF scheduler interface. De-
fault is disabled.
--enable-wsgram-pbs    Build GRAM PBS scheduler interface. De-
fault is disabled.
--enable-il8n          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 Table 11-1

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**.

## Chapter 5. Basic Security Configuration

### 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
```

### Obtain host certificates

You must have X509 certificates to use the GT 3.9.5 software securely (referred to in this documentation as *host certificates*). For an overview of certificates for GSI (security) see GSI Configuration Information<sup>1</sup>.

Host certificates must be:

- consist of the following two files: `hostcert.pem` and `hostkey.pem`
- must be in the appropriate directory for secure services: `/etc/grid-security/`
- 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.

You have the following options:

### 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<sup>2</sup>.

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<sup>3</sup>](#) for more details.

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

## 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 Appendix A.

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

## 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.

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

## 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.

## Verify Basic Security

Now that you have installed a trusted CA, acquired a hostcert and 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. If it does not succeed, or you want to continue to configure multiple nodes, you may want to continue to the full security overview in the next chapter. Otherwise, you may proceed to the services chapter.

## **Notes**

1. <http://www.globus.org/toolkit/docs/3.2/gsi/admin/configuration.html>
2. <http://www.terena.nl/tech/task-forces/tf-aace/tacar/certs.html>
3. <http://www.globus.org/toolkit/docs/3.2/gsi/admin/configuration.html#TrustedCA>
4. <http://gcs.globus.org:8080/gcs>

## Chapter 6. Security Overview

Authentication in the Globus Toolkit is based on X.509 certificates. This document describes the configuration steps required to:

- Determine whether or not to trust certificates issued by a particular Certificate Authority (CA),
- Provide appropriate default values for use by the **grid-cert-request** command, which is used to generate certificates,
- Request *service certificates*, used by services to authenticate themselves to users, and
- Specify identity mapping information.

### Configuration

This section describes the configuration steps required to:

- Determine whether or not to trust certificates issued by a particular Certificate Authority (CA),
- Provide appropriate default values for use by the **grid-cert-request** command, which is used to generate certificates,
- Request *service certificates*, used by services to authenticate themselves to users, and
- Specify identity mapping information.

In general, Globus tools will look for a configuration file in a user-specific location first, and in a systemwide location if no user-specific file was found. The configuration commands described here may be run by administrators to create systemwide defaults, and by individuals to override those defaults. [TODO: Add a reference to an overview document]

### Configuring Globus to Trust a Particular Certificate Authority

The Globus tools will trust certificates issued by a CA if (and only if) it can find information about that CA in the trusted certificate directory. [TODO: link to the environment fragment, which explains where the trusted certificates directory is] The following two files must exist in that directory for each trusted CA:

**Table 6-1. Trusted CA files**

<code>cert_hash.0</code>	The trusted CA certificate.
<code>cert_hash.signing_policy</code>	A configuration file defining the distinguished names of certificates signed by the CA.

Pre-WS Globus components will honor a certificate only if:

- its CA certificate exists (with the appropriate name) in the `TRUSTED_CA` directory, and
- the certificate's distinguished name matches the pattern described in the signing policy file.

WSRF-based components ignore the signing policy file and will honor all valid certificates issued by trusted CAs.

The `cert_hash` that appears in the file names above is the hash of the CA certificate, which can be found by running the command:

```
$GLOBUS_LOCATION/bin/openssl x509 -hash -noout < ca_certificate
```

Some CAs provide tools to install their CA certificates and signing policy files into the trusted certificates directory. You can, however, create a signing policy file by hand; the signing policy file has the following format:

```
access_id_CA X509 'CA Distinguished Name'
pos_rights globus CA:sign
cond_subjects globus '"Distinguished Name Pattern"'
```

In the above, the *CA Distinguished Name* is the subject name of the CA certificate, and the *Distinguished Name Pattern* is a string used to match the distinguished names of certificates granted by the CA. Some very simple wildcard matching is done -- if the *Distinguished Name Pattern* ends with a '\*', then any distinguished name that matches the part of the CA subject name before the '\*' is considered a match. Note: the `cond_subjects` line may contain a space-separated list of distinguished name patterns.

## Configuring Globus to Create Appropriate Certificate Requests

The `grid-cert-request` command, which is used to create certificates, uses the following configuration files:

**Table 6-2. grid-cert-request configuration files**

<code>globus-user-ssl.conf</code>	defines the distinguished name to use for a user's certificate request. The format is described here <sup>1</sup> .
<code>globus-host-ssl.conf</code>	defines the distinguished name for a host (or service) certificate request. The format is described here <sup>1</sup> .
<code>grid-security.conf</code>	a base configuration file that contains the name and email address for the CA.

Many CAs provide tools to install configuration files called `globus-user-ssl.conf.cert_hash`, `globus-host-ssl.conf.cert_hash`, and `grid_security.conf.cert_hash` in the trusted certificates directory. The command:

```
grid-cert-request -ca cert_hash
```

will create a certificate request based on the specified CA's configuration files. The command:

```
grid-cert-request -ca
```

will list the available CAs and let the user choose which one to create a request for.

You can specify a default CA for certificate requests (i.e., a CA that will be used if `grid-cert-request` is invoked without the `-ca` flag) by making the following symbolic links (where `GRID_SECURITY` is the grid security directory [TODO: link to environment fragment] and `TRUSTED_CA` is the trusted CA directory):

```
ln -s GRID_SECURITY/globus-user-ssl.conf \
    TRUSTED_CA/globus-user-ssl.conf.cert_hash
ln -s GRID_SECURITY/globus-host-ssl.conf \
```

```
TRUSTED_CA/globus-host-ssl.conf.cert_hash
ln -s GRID_SECURITY/grid_security.conf \
TRUSTED_CA/grid_security.conf.cert_hash
```

## Requesting Service Certificates

Different CAs use different mechanisms for issuing end-user certificates; some use mechanisms that are entirely web-based, while others require you to generate a certificate request and send it to the CA. If you need to create a certificate request for a service certificate, you can do so by running:

```
grid-cert-request -host hostname -service service_name
```

where *hostname* is the fully-qualified name of the host on which the service will be running, and *service\_name* is the name of the service. This will create the following three files:

**Table 6-3. grid-cert-request output files**

<code>GRID_SECURITY/service_name/service</code>	An empty file. When you receive your actual service certificate from your CA, you should place it in this file.
<code>GRID_SECURITY/service_name/service</code>	The certificate request, which you should send to your CA.
<code>GRID_SECURITY/service_name/service</code>	The private key associated with your certificate request, encrypted with the pass phrase that you entered when prompted by <b>grid-cert-request</b> .

The **grid-cert-request** command recognizes several other useful options; you can list these with:

```
grid-cert-request -help
```

## Host Certificates and Client-side Authorization

Globus clients are generally configured to authorize the the remote service based on the name of the host running the service. In practice this is done by first doing a reverse lookup on the IP address used to connect to the service. This provides the client with the canonical name for the host. Once the service has authenticated to the client, the client compares the content of the first common name (CN) component in the service's identity with the canonical hostname obtained in the first step. For the purpose of comparison the following rules apply:

- The comparison is not case sensitive.
- A common name of *host/FQDN* (Fully Qualified Domain Name) is equivalent to a common name of *FQDN*.
- Since many sites use the convention of naming interfaces by having the FQDN in the form *host-interface.domain* and clients may not be aware of what interface they are talking to, *host-interface.domain* is considered equal to *host.domain*.

## Specifying Identity Mapping Information

Several Globus services map distinguished names (found in certificates) to local identities (e.g., unix logins). These mappings are maintained in the `gridmap` file. [TODO: link to the environment fragment, which tells how to find the gridmap file] A gridmap line of the form:

```
"Distinguished Name" local_name
```

maps the distinguished name *Distinguished Name* to the local name *local\_name*. A gridmap line of the form:

```
"Distinguished Name" local_name1,local_name2
```

maps *Distinguished Name* to both *local\_name1* and *local\_name2*; any number of local user names may occur in the comma-separated local name list.

Several tools exist to manage gridmap files. To add an entry to the gridmap file, run:

```
$GLOBUS_LOCATION/sbin/grid-mapfile-add-entry \  
-dn "Distinguished Name" \  
-ln local_name
```

To delete an entry from the gridmap file, run:

```
$GLOBUS_LOCATION/sbin/grid-mapfile-delete-entry \  
-dn "Distinguished Name" \  
-ln local_name
```

To check the consistency of the gridmap file, run

```
$GLOBUS_LOCATION/sbin/grid-mapfile-check-consistency
```

These commands recognize several useful options, including a **-help** option, which lists detailed usage information.

The location of the `gridmap` file is determined as follows:

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

## GSI File Permissions Requirements

- End Entity<sup>1</sup> (User<sup>2</sup>, Host<sup>3</sup> and Service<sup>4</sup>) Certificates and the GSI Authorization Callout Configuration File<sup>5</sup>:
  - May not be executable
  - May not be writable by group and other
  - Must be either regular files or soft links
- Private Keys<sup>6</sup> and Proxy Credentials<sup>7</sup>:
  - Must be owned by the current (effective) user

- May not be executable
  - May not be readable by group and other
  - May not be writable by group and other
  - Must be either regular files or soft links
- CA Certificates<sup>8</sup>, CA Signing Policy Files<sup>9</sup>, the Grid Map File<sup>10</sup> and the GAA Configuration File<sup>11</sup>:
    - Have to be either regular files or soft links

## Troubleshooting

The following are some common problems that may cause clients or servers to report that credentials are invalid:

- Your proxy credential may have expired
 

Use **grid-proxy-info** to check whether the proxy has actually expired. If it has, generate a new proxy with **grid-proxy-init**.
- The system clock on either the local or remote system is wrong
 

This may cause the server or client to conclude that a credential has expired.
- Your end-user certificate may have expired
 

Use **grid-cert-info** to check your certificate's expiration date. If it has expired, follow your CA's procedures to get a new one.
- The permissions may be wrong on your proxy file
 

If the permissions on your proxy file are too lax (for example, if others can read your proxy file), Globus Toolkit clients will not use that file to authenticate. You can "fix" this problem by changing the permissions on the file or by destroying it (with **grid-proxy-destroy** and creating a new one (with **grid-proxy-init**). However, it is still possible that someone else has made a copy of that file during the time that the permissions were wrong. In that case, they will be able to impersonate you until the proxy file expires or your permissions or end-user certificate are revoked, whichever happens first.
- The permissions may be wrong on your private key file
 

If the permissions on your end user certificate private key file are too lax (for example, if others can read the file), **grid-proxy-init** will refuse to create a proxy certificate. You can "fix" this by changing the permissions on the private key file; however, you will still have a much more serious problem: it's possible that someone has made a copy of your private key file. Although this file is encrypted, it is possible that someone will be able to decrypt the private key, at which point they will be able to impersonate you as long as your end user certificate is valid. You should contact your CA to have your end-user certificate revoked and get a new one.
- The remote system may not trust your CA

Verify that the remote system is configured to trust the CA that issued your end-entity certificate. See the [TODO: add admin guide link] for details.

- You may not trust the remote system's CA  
Verify that your system is configured to trust the remote CA (or that your environment is set up to trust the remote CA). See the [TODO: add admin guide link] for details.
- There may be something wrong with the remote service's credentials  
It is sometimes difficult to distinguish between errors reported by the remote service regarding your credentials and errors reported by the client interface regarding the remote service's credentials. If you can't find anything wrong with your credentials, check for the same conditions (or ask a remote administrator to do so) on the remote system.
- The following are some common problems that may cause clients or servers to report that user are not authorized:
  - The content of the gridmap file does not conform to the expected format  
Use **grid-mapfile-check-consistency** to make sure that your gridmap conforms to the expected format.
  - The gridmap file does not contain a entry for your DN  
Use **grid-mapfile-add-entry** to add the relevant entry.

## Notes

1. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#End\\_Entity\\_C](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#End_Entity_C)
2. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#User\\_Certific](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#User_Certific)
3. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#Host\\_Certific](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#Host_Certific)
4. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#Service\\_Certi](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#Service_Certi)
5. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#GSI\\_Authoriz](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#GSI_Authoriz)
6. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#Private\\_Key](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#Private_Key)
7. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#Proxy\\_Crede](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#Proxy_Crede)
8. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#CA\\_Certifica](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#CA_Certifica)
9. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#CA\\_Signing\\_](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#CA_Signing_)
10. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#Grid\\_Map\\_Fi](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#Grid_Map_Fi)
11. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#GAA\\_Config](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/glossary.html#GAA_Config)

# Chapter 7. GridFTP Configuration

## Introduction

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

This is a partially-complete docbook translation of the GridFTP Admin Guide<sup>1</sup>. Please see that document for additional information.

## Deploying the GridFTP Server

- Running in daemon mode
- Running under inetd/xinetd
- Remote data-nodes and striped operation

## Running in daemon mode

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

Run the server:

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

where:

**Table 7-1. TITLE**

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

The following additional steps may be required when running as a user other than root.

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

## Running under inetd or xinetd

The `-i` command line option enables the server to be run under inetd or xinetd.

See the Configuration and command line options<sup>2</sup> section for example xinetd and inetd configuration entries.

## Remote data-nodes and striped operation

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

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

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

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

To start the front end:

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

To start the data-node:

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

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

For example:

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

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

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

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

## Testing

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

Please also see the [Call for Community Testing](#)<sup>3</sup>.

## Security Considerations

import

## Troubleshooting

If you are having problems using the GridFTP server, you should try these steps:

Verify that the server has started successfully. The easiest way to do this is to `telnet` to the port on which the server is running.

```
% telnet localhost 2811
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 GridFTP Server localhost.localdomain 0.11 (gcc32dbg, 1098910702-1) ready.
```

If you see anything other than a 220 banner such as that, then the server has not started correctly. Verify that you have the options you expect. Try no other options besides `-s`, `-i`, or `-p` (the server defaults should be fine in most cases). Verify that there are no configuration files being unexpectedly loaded from `/etc/grid-security/gridftp.conf` or `$GLOBUS_LOCATION/etc/gridftp.conf`. If all else fails and you still cannot pass this test, seek help on [discuss@globus.org](mailto:discuss@globus.org)

If the server has started correctly, and your problem is with a security failure or gridmap lookup failure, verify that you have security configured properly here<sup>4</sup>.

If the server is running and your client successfully authenticates, but has a problem at some other time during the session, please ask for help on [discuss@globus.org](mailto:discuss@globus.org)

## Notes

1. <http://www-unix.globus.org/toolkit/docs/development/3.9.5/data/gridftp/admin/index.html>
2. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/data/gridftp/GridFTP\\_Public\\_Interface.html](http://www-unix.globus.org/toolkit/docs/development/3.9.5/data/gridftp/GridFTP_Public_Interface.html)
3. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/data/gridftp/GridFTP\\_Call\\_for\\_Testing.html](http://www-unix.globus.org/toolkit/docs/development/3.9.5/data/gridftp/GridFTP_Call_for_Testing.html)
4. <http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/prewsaa/admin/>



## Chapter 8. Webservices container

The Webservices components of the Toolkit run inside of a hosting environment, also known as a container. This chapter covers the startup and configuration of that container.

Prerequisites to this chapter are setting up the `containercert.pem` and `containerkey.pem`, as outlined in the the Section called *Make the host credentials accessible by the container* in Chapter 5.

This is a partially-complete docbook translation of the Java WS Core Admin Guide<sup>1</sup>. Please see that document for additional information.

### Starting the container

As the globus user, run:

```
$ $GLOBUS_LOCATION/bin/globus-start-container
Starting SOAP server at: https://140.221.57.104:8443/wsrf/services/
With the following services:
[1]: https://140.221.57.104:8443/wsrf/services/IndexFactoryService
[2]: https://140.221.57.104:8443/wsrf/services/TriggerFactoryService
[... more services listed ...]
[47]: https://140.221.57.104:8443/wsrf/services/ManagedJobFactoryService
[48]: https://140.221.57.104:8443/wsrf/services/TestServiceRequest
```

**Note:** Please make sure to have the JAAS<sup>2</sup> library installed if running with J2SE 1.3.1.

`globus-start-container` supports the following options:

**Table 8-1. globus-start-container options**

<code>-help</code>	Displays help information about the command.
<code>-p <i>port</i></code>	Sets the port number for the container.
<code>-quiet</code>	Does not show a list of services at startup.
<code>-nosec</code>	Starts a non secure (HTTP) container.
<code>-containerDesc <i>file</i></code>	Specifies a container security descriptor file.
<code>-profile <i>name</i></code>	Specifies a configuration profile name for the container. See the Section called <i>Configuration Profiles</i> for more details.

### Service configuration overview

Java WS Core provides per-service configuration and supports configuration profiles. The configuration information of a service is mainly encapsulated in three separate configuration files:

**Table 8-2. Webservice configuration files**

Filename	Description
server-config.wsdd	(Web Service Deployment Descriptor) - contains information about the web service.
jndi-config.xml	(JNDI configuration file) - contains information about the resource management.
security-config.xml	(security deployment descriptor) - Please see WEBSERVICES AUTHENTICATION AND AUTHZ for details. This file is optional.

All these configuration files are dropped into the `$GLOBUS_LOCATION/etc/service_name /` directory during the deployment process.

That is the general overview of service configuration. Specific service configuration will be left for later chapters.

## Container configuration

The container itself has configuration files in `$GLOBUS_LOCATION/etc/globus_wsrf_core .` The general syntax is

```
<parameter name="name" value="value" />
```

Table 8-3. Container configuration parameters

Name	Value	Description
logicalHost	<i>hostname</i>	This parameter specifies the hostname to use instead of the default local host. It is equivalent to setting the <code>GLOBUS_HOSTNAME</code> environment property. Can be FQDN or just hostname.
disableDNS	<i>boolean</i>	This parameter specifies whether to perform DNS lookup on the logicalHost parameter. By default "false" is assumed, meaning a DNS lookup is performed.
domainName	<i>domanin name</i>	This parameter specifies the domain name to append to the host name if the host name is not qualified by a domain.

Name	Value	Description
publishHostName	<i>boolean</i>	This parameter specifies whether to publish the hostname or the ip address. It is only used when DNS lookups are enabled (disableDNS is false).

Suppose your container usually published the following information:

```
[1]: https://140.221.36.14:8443/wsrf/services/TriggerFactoryService
```

If you want it to publish the hostname for that IP address, you could set the following:

```
<parameter name="logicalHost" value="mayed.mcs.anl.gov" />
<parameter name="publishHostName" value="true" />
```

Starting the container would then show:

```
[1]: https://mayed.mcs.anl.gov:8443/wsrf/services/TriggerFactoryService
```

## Configuration Profiles

Configuration profiles allow for the same Java WS Core installation to have multiple configurations. That is, the same installation can be used to run different containers each with different configuration.

You can create a copy of the configuration files with a profile name prefix (like *profile.name-server-config.wsdd* ). Then you can start the container with the **-profile name** option, and the profile's configuration files will be used.

**Note:** Each configuration profile should duplicate the contents of `$GLOBUS_LOCATION/etc/globus_wsrf_core/server-config.wsdd` and `$GLOBUS_LOCATION/etc/globus_wsrf_core/jndi-config.xml` in order to make the basic functionality to work properly.

## Notes

1. <http://www-unix.globus.org/toolkit/docs/development/3.9.5/common/javawscore/admin/index.1>
2. <http://java.sun.com/products/jaas/index-10.html>



## Chapter 9. RFT Configuration

### Introduction

RFT is used to perform third-party transfers across GridFTP servers. It uses a database to store its state periodically so the transfers can be recovered from any failures. RFT uses standard grid security mechanisms for authorization and authentication of the users. So in order to effectively use RFT you should have installed and configured a database with RFT database schemas and have the necessary security infrastructure in place to perform a 3rd party transfer.

This is a partially-complete docbook translation of the RFT Admin Guide<sup>1</sup>. Please see that document for additional information.

### Configuring

- Required configuration: configuring the PostgreSQL database

#### Required configuration: configuring the PostgreSQL database

PostgreSQL (Version 7.1 or greater ) needs to be installed and configured for RFT to work. You can either use the packages which came with your operating system (RPMs, DEBs, ...) or build from source. We used PostgreSQL version 7.3.2 for our testing and the following instructions are good for the same.

1. Install Postgresql. Instructions on how to install/configure postgresql can be found here<sup>2</sup>.
2. Configure the postmaster daemon so that it accepts TCP connections. This can be done by adding -o "-i" switch to postmaster script.
3. To create the database that is used for RFT, run: **createdb rftDatabase**
4. To populate the RFT database with appropriate schemas, run: **psql -d rft-Database -f \$GLOBUS\_LOCATION/share/globus\_wsrft\_rft\_schema.sql**  
Now that you have created a database to store RFT's state, the following steps configure RFT to find the database:
5. Open `$GLOBUS_LOCATION/etc/globus_wsrft_rft/jndi-config.xml`
6. Find the dbConfiguration section under ReliableFileTransferService <service> section.
7. Change the connectionString to point to the machine on which you installed Postgres and name of the database you used in step 2. If you installed Postgres on the same machine as your Globus install, the default should work fine for you.
8. Change the userName to the name of the user who owns/created the database and do the same for the password. (It also depends on how you configured your database.)
9. Don't worry about the other parameters in that section. The defaults should work fine for now.
10. Edit the configuration section under ReliableFileTransferService. There are two values that can be edited in this section.
  - backOff: Time in seconds you want RFT to backoff before a failed transfer is retried by RFT. Default should work fine for now.

- `maxActiveAllowed`: This is the number of transfers the container can do at given point. Default should be fine for now.

## Security Considerations

import

## Troubleshooting

*Problem:* If RFT is not configured properly to talk to a PostgreSQL database, you will see this message displayed on the console when you start the container :

```
"Error creating RFT Home: Failed to connect to database ...  
Until this is corrected all RFT request will fail and all GRAM jobs that re-  
quire staging will fail".
```

*Solution:* Usual mistake is Postmaster is not accepting TCP connections which means that you must restart Postmaster with `-i` option ( see step 2<sup>3</sup>).

## Notes

1. <http://www-unix.globus.org/toolkit/docs/development/3.9.5/data/rft/admin/index.html>
2. <http://www.postgresql.org/docs/manuals/>
3. #step2

# Chapter 10. WS GRAM Configuration

## Introduction

This guide contains advanced configuration information for system administrators working with WS GRAM. It provides references to information on procedures typically performed by system administrators, including installation, configuring, deploying, and testing the installation. It also describes additional prerequisites and host settings necessary for WS GRAM operation. Readers should be familiar with the Key Concepts<sup>1</sup> and Implementation Approach<sup>2</sup> for WS GRAM to understand the motivation for and interaction between the various deployed components.

This is a partially-complete docbook translation of the WS GRAM Admin Guide<sup>3</sup>. Please see that document for additional information.

## Local Prerequisites

WS GRAM requires the following:

- Host credentials
- GRAM service account
- Gridmap authorization of user account
- Functioning sudo
- Local scheduler

## Host credentials

In order to use WS GRAM, the services running in the WSRF hosting environment require access to an appropriate host certificate.

## GRAM service account

WS GRAM requires a *dedicated local account* within which the WSRF hosting environment and GRAM services will execute. This account will often be a `globus>` account used for all local services, but may also be specialized to only host WS GRAM. User jobs will run in separate accounts as specified in the `grid-mapfile` or associated authorization policy configuration of the host.

## Gridmap authorization of user account

In order to authorize a user to call GRAM services, the security configuration must map the Distinguished Name (DN) of the user to the name of the user in the system where the GRAM services run. Here are the configuration steps:

1. In order to obtain the DN, which is the subject of the user certificate, run the `bin/grid-cert-info` command in `$GLOBUS_LOCATION` on the submission machine:

```
% bin/grid-cert-info -identity
/O=Grid/OU=GlobusTest/OU=simpleCA-foo.bar.com/OU=bar.com/CN=John Doe
```

2. Create a `/etc/grid-security/grid-mapfile`. The syntax is to have one line per user, with the distinguished name followed by a whitespace and then the

user account name on the GRAM machine. Since the distinguished name usually contains whitespace, it is placed between quotation marks, as in:

```
"/O=Grid/OU=GlobusTest/OU=simpleCA-foo.bar.com/OU=bar.com/CN=John Doe" john-doe
```

## Functioning sudo

WS GRAM requires that the **sudo** command is installed and functioning on the service host where WS GRAM software will execute.

Authorization rules will need to be added to the `sudoers` file to allow the WS GRAM service account to execute (without a password) local scheduler adapters in the accounts of authorized GRAM users. See Configuring sudo below

## Local scheduler

WS GRAM depends on a local mechanism for starting and controlling jobs. If the fork-based WS GRAM mode is to be used, no special software is required. For batch scheduling mechanisms, the local scheduler must be installed and configured for local job submission prior to deploying and operating WS GRAM. The supported batch schedulers in the GT 3.9.5 release are: PBS, Condor, LSF

## RFT Dependency

RFT prerequisites include PostgreSQL to be installed and configured. The instructions are here. WS GRAM depends on RFT for file staging and cleanup. File staging from client host to compute host and visa versa.

**Important:** Jobs requesting these functions will fail if RFT is not properly setup.

## Configuring

- Configuration settings
- Setting up service credentials
- Enabling Local Scheduler Adapter
- Configuring sudo
- Extra steps for non-default installation

## Configuration settings

include

## Setting up service credentials

In a default build and install of the Globus Toolkit, the local account is configured to use host credentials at `/etc/grid-service/containercert.pem` and `containerkey.pem`.

If you already have host certs, then you can just copy them to the new name and set ownership.

```
% cd /etc/grid-security
% cp hostcert.pem containercert.pem
% cp hostkey.pem containerkey.pem
% chown globus.globus container*.pem
```

Replace globus.globus with the user and group the container is installed as.

You should now have something like:

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

The result is a copy of the host credentials which are accessible by the container.

If this is not an option, then you can configure an alternate location to point to host credentials -or- configure to use just a user proxy (personal mode).

## Enabling Local Scheduler Adapter

The batch scheduler interface implementations included in the release tarball are: PBS, Condor and LSF. To install one of the batch scheduler adapters, follow these steps (shown for pbs):

```
% cd $GLOBUS_LOCATION\gt3.9.5-all-source-installer
% make gt4-gram-pbs postinstall
% gpt-postinstall
```

Using PBS as the example, make sure the batch scheduler commands are in your path (qsub, qstat, pbsnodes).

For PBS, another setup step is required to configure the remote shell for rsh access:

```
% cd $GLOBUS_LOCATION/setup/globus
% ./setup-globus-job-manager-pbs --remote-shell=rsh
```

The last thing is to define the GRAM and GridFTP file system mapping<sup>4</sup> for PBS.

Done! You have added the PBS scheduler adapters to your GT installation.

## Configuring sudo

When the credentials of the service account and the job submitter are different (multi user mode), then GRAM will prepend a call to sudo to the local adapter callout command.

**Important:** If sudo is not configured properly, the command and thus job will fail.

As *root*, here are the two lines to add to the `/etc/sudoers` file for each GLOBUS\_LOCATION installation, where `/opt/globus/GT3.9.5` should be replaced with the GLOBUS LOCATION for your installation:

```
# Globus GRAM entries
globus ALL=(username1,username2)
NOPASSWD: /opt/globus/GT3.9.5/libexec/globus-gridmap-and-execute
/opt/globus/GT3.9.5/libexec/globus-job-manager-script.pl *
```

```
globus ALL=(username1,username2)
NOPASSWD: /opt/globus/GT3.9.5/libexec/globus-gridmap-and-execute
/opt/globus/GT3.9.5/libexec/globus-gram-local-proxy-tool *
```

## Extra steps for non-default installation

- Non-default service credentials
- Non-default GridFTP server
- Non-default container port
- Non-default gridmap
- Non-default job resource limit

### Non-default service credentials

- Alternative location for host credentials
- User proxy

### Alternate location for host credentials

If setting up host credentials in the default location of `/etc/grid-security/containercert.pem` and `containerkey.pem` is *not* an option for you, then you can configure an alternate location to point to host credentials.

Security descriptor configuration details are here<sup>5</sup>, but the quick change is to edit this file - `$GLOBUS_LOCATION/etc/globus_wsrf_core/global_security_descriptor.xml` - by changing the cert and key paths to point to host credentials that the service account owns.

### User proxy

To run the container using just a user proxy, simply comment out the `ContainerSecDesc` parameter in this file `$GLOBUS_LOCATION/etc/globus_wsrf_core/server-config.wsdd` as follows:

```
<!--
  <parameter
    name="containerSecDesc"
    value="etc/globus_wsrf_core/global_security_descriptor.xml" />
-->
```

Running in personal mode (user proxy), another GRAM configuration setting is required. For GRAM to authorize the RFT service when performing staging functions, it needs to know the subject DN for verification. Here are the steps:

```
% cd $GLOBUS_LOCATION/setup/globus
% ./setup-gram-service-common --staging-subject=
  "/DC=org/DC=doegrids/OU=People/CN=Stuart Martin 564720"
```

You can get your subject DN by running this command:

```
% grid-cert-info -subject
```

### Non-default GridFTP server

By default, the GridFTP server is assumed to run as root on localhost:2811. If this is not true for your site then you must update the configuration by editing the GRAM and GridFTP file system mapping<sup>6</sup> config file: `$GLOBUS_LOCATION/etc/gram-service/globus_gram_fs_map_config.xml`.

### Non-default container port

By default, the globus services will assume 8443 is the port the Globus container is using. However the container can be run under a non-standard port, for example:

```
% globus-start-container -p 4321
```

When doing this, GRAM needs to be told the port to use to contact the RFT service, like so:

```
% cd $GLOBUS_LOCATION/setup/globus
% ./setup-gram-service-common --staging-port="4321"
```

### Non-default gridmap

If you wish to specify a non-standard gridmap file in a multi-user installation, two basic configurations need to be changed:

- `$GLOBUS_LOCATION/etc/globus_wsrf_core/global_security_descriptor.xml`  
As specified in the gridmap config<sup>7</sup> instructions, add a `<gridmap value="..." />` element to the file appropriately.
- `/etc/sudoers`  
Add `"-g /path/to/grid-mapfile"` as the first argument to all instances of the `globus-gridmap-and-exec` command.

Example: `global_security_descriptor.xml`

```
...
<gridmap value="/opt/grid-mapfile"/>
...
sudoers
...
# Globus GRAM entries
globus ALL=(username1,username2)
NOPASSWD: /opt/globus/GT3.9.5/libexec/globus-gridmap-and-execute
-g /opt/grid-mapfile
/opt/globus/GT3.9.5/libexec/globus-job-manager-script.pl *
globus ALL=(username1,username2)
NOPASSWD: /opt/globus/GT3.9.5/libexec/globus-gridmap-and-execute
-g /opt/grid-mapfile
/opt/globus/GT3.9.5/libexec/globus-gram-local-proxy-tool *
```

### Non-default job resource limit

The current limit on the number of job resources (both exec and multi) allowed to exist at any one time is 1000. This limit was chosen from scalability tests as an appropriate precaution to avoid out-of-memory errors. To change this value to, say, 150, use the `setup-gram-service-common` script as follows:

```
% cd $GLOBUS_LOCATION/setup/globus
% ./setup-gram-service-common --max-job-limit="150"
```

## Testing

See the WS GRAM users guide<sup>8</sup> for information about submitting a test job.

## Security Considerations

import

## Troubleshooting

[todo]

## Notes

1. ../../key/
2. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/execution/key/WS\\_GRAM\\_Approach](http://www-unix.globus.org/toolkit/docs/development/3.9.5/execution/key/WS_GRAM_Approach)
3. <http://www-unix.globus.org/toolkit/docs/development/3.9.5/execution/wsgram/admin/index.html>
4. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/execution/wsgram/WS\\_GRAM\\_Publications](http://www-unix.globus.org/toolkit/docs/development/3.9.5/execution/wsgram/WS_GRAM_Publications)
5. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/authzframe/security\\_descriptions](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/authzframe/security_descriptions)
6. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/execution/wsgram/WS\\_GRAM\\_Publications](http://www-unix.globus.org/toolkit/docs/development/3.9.5/execution/wsgram/WS_GRAM_Publications)
7. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/authzframe/security\\_descriptions](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/authzframe/security_descriptions)
8. <http://www-unix.globus.org/toolkit/docs/development/3.9.5/execution/wsgram/user/commands>

# Chapter 11. Configuring GSI-OpenSSH

## Introduction

This guide contains advanced configuration information for system administrators working with GSI-OpenSSH. It provides references to information on procedures typically performed by system administrators, including installation, configuring, deploying, and testing the installation.

This guide is meant solely to cover the GSI aspects of GSI-OpenSSH, and is not meant to be a full manual for OpenSSH itself. Please refer to the OpenSSH Home Page<sup>1</sup> for general documentation for OpenSSH.

This is a partially-complete docbook translation of the GSI-OpenSSH Admin Guide<sup>2</sup>. Please see that document for additional information.

## Building and Installing

You can optionally pass build-time configure options to the GSI-OpenSSH package by setting the `--with-gsiopenssh-opts="args"` option to `configure` during the build phase. No options are typically needed for client-only installations, but options are often needed for full server functionality. The following table lists suggested options for different platforms.

Table 11-1. `gsi-openssh` build options

Platform	Configuration
Linux	<code>--with-pam --with-md5-passwords --with-tcp-wrappers</code>
Solaris	<code>--with-pam --with-md5-passwords --with-tcp-wrappers</code>
Irix	<code>--with-tcp-wrappers</code>
AIX	<code>--with-tcp-wrappers</code>

Note: If you enable PAM support with the `--with-pam` configuration option, be sure to also set "UsePAM yes" in `$GLOBUS_LOCATION/etc/ssh/sshd_config` after installation.

If you have an already configured and installed system-wide SSHD and you would like your build of GSI-OpenSSH to behave similarly, investigate the configure options available in GSI-OpenSSH and select those options that would add the functionality that your current SSHD possesses. Be aware that since GSI-OpenSSH is based on OpenSSH, the standard set of functionality is turned on by default.

## Configuring

- Configuration settings
- System clocks

## Configuration settings

conf settings

## System clocks

GSI authentication is very sensitive to clock skew. You must run a system clock synchronization service of some type on your system to prevent authentication problems caused by incorrect system clocks. We recommend NTP<sup>3</sup>. Please refer to your operating system documentation or the NTP Home Page<sup>4</sup> for installation instructions. Please also ensure your system timezone is set correctly.

## Deploying

1. To install the GSI-Enabled OpenSSH Server on most systems, you must be a privileged user, such as root.

```
sh$ /bin/su - root
```

Note: If your system functions like this and you attempt to run these commands as a user other than root, these commands should fail.

2. (optional) Start a copy of your system's currently running SSH server on an alternate port by running, eg.

```
sh# /usr/sbin/sshd -p 2000 &
```

You may then choose to log in to this server and continue the rest of these steps from that shell. We recommend doing this since some sshd shutdown scripts do particularly nasty things like killing *all* of the running SSH servers on a system, not just the parent server that may be listening on port 22.

Roughly translated, this step is about guaranteeing that an alternate method of access is available should the main SSH server be shutdown and your connection via that server be terminated.

3. Locate your server's startup/shutdown script directory. On some systems this directory may be located at /etc/rc.d/init.d, but since this location is not constant across operating systems, for the purposes of this document we will refer to this directory as INITDIR. Consult your operating system's documentation for your system's location.

4. Run the following command

```
sh# mv $INITDIR/sshd $INITDIR/sshd.bak
```

5. Either copy or link the new sshd script to your system's startup/shutdown script directory.

```
sh# cp $GLOBUS_LOCATION/sbin/SXXsshd $INITDIR/sshd
```

6. Shutdown the currently running main SSH server.

```
sh# $INITDIR/sshd.bak stop
```

7. Provided you still have a connection to the machine, start the new SSH server.

```
sh# $INITDIR/sshd start
```

8. Test the new server by connecting to the standard SSH port (22) and authenticating via multiple methods. Especially test that GSI authentication works correctly.

9. If you are performing a new install, or if the old server was not configured to be started at run-time and shutdown automatically at system halt or reboot, either use a system utility such as RedHat's chkconfig to configure the system for the correct run-levels, or manually link up the correct run-levels.

```
sh# /sbin/chkconfig sshd reset
```

The recommended run-levels are listed in a set of comments within the SXXsshd startup script. For example, on standard Unix systems we recommend running the GSI-Enabled OpenSSH server in run-levels two, three, four, and five.

10. Finally, if, as a precautionary measure, you started a SSH server on an alternate port in order to complete the install process, you can now safely stop all instances of that server.

## Testing

1. Edit the file `$GLOBUS_LOCATION/sbin/SXXsshd` so that the GSI-Enabled OpenSSH server starts up on an alternate port.

2. Run the command

```
sh# $GLOBUS_LOCATION/sbin/SXXsshd start
```

and verify that the server is running by checking that it both shows up in a process listing and creates a file named `$GLOBUS_LOCATION/var/sshd.pid`.

3. From a remote machine attempt to connect to the local server on the modified test port using the standard SSH authentication methods plus authenticating via your GSI credentials. This may require you to authorize these users via an appropriate entry in the `grid-mapfile`.

4. Stop the SSH server by running the command

```
sh# $GLOBUS_LOCATION/sbin/SXXsshd stop
```

and reverse any changes you made that altered the port on which the server resided upon startup. After this step, running `SXXsshd start` should start the server on the default port (22).

## Notes

1. <http://www.openssh.org/>
2. <http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/openssh/admin/index.htm>
3. <http://www.ntp.org/>
4. <http://www.ntp.org/>



## Chapter 12. Configuring MyProxy

### Introduction

This guide contains advanced configuration information for system administrators working with MyProxy. It provides references to information on procedures typically performed by system administrators, including installation, configuring, deploying, and testing the installation.

A typical MyProxy configuration has one dedicated myproxy-server for the site, with MyProxy clients installed on all systems where other Globus Toolkit client software is installed.

This is a partially-complete docbook translation of the MyProxy Admin Guide<sup>1</sup>. Please see that document for additional information.

### Configuring

No additional configuration is required to use MyProxy clients after they are installed, although you may want to set the MYPROXY\_SERVER environment variable to the hostname of your myproxy-server in the default user environment on your systems. Information for configuring the myproxy-server is here.

- Configuration settings
- Configuring a MyProxy Server installation

### Configuration settings

```
import
```

### Configuring a MyProxy server installation

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.

Next, modify \$GLOBUS\_LOCATION/etc/myproxy-server.config to configure the MyProxy server. *If you skip this step, your myproxy-server will not accept any requests.* The default configuration does not enable any myproxy-server features to provide the greatest security until you have configured your server. To enable all myproxy-server features, uncomment to provided sample policy at the top of the myproxy-server.config config file, as follows:

```
#
# Complete Sample Policy
#
# The following lines define a sample policy that enables all
# myproxy-server features. See below for more examples.
accepted_credentials  "*"
authorized_retrievers "*"
default_retrievers   "*"
authorized_renewers  "*"

```

```
default_renewers      "none"
```

If you have root access, you can copy your `myproxy-server.config` file to `/etc/myproxy-server.config` so it is not overwritten by later installations.

## Deploying

A sample SysV-style boot script for MyProxy is installed at `$GLOBUS_LOCATION/share/myproxy/etc.init.d.myproxy`. To install on Linux, copy the file to `/etc/rc.d/init.d/myproxy` and run `'chkconfig --add myproxy'`. You will need to edit the file to set the `GLOBUS_LOCATION` environment variable correctly.

Alternatively, to run the myproxy server out of `inetd` or `xinetd`, you need to do the following as root:

- Add the entries in `$GLOBUS_LOCATION/share/myproxy/etc.services.modifications` to the `/etc/services` or `/etc/inet/services` file.
- Add the entries in `$GLOBUS_LOCATION/share/myproxy/etc.inetd.conf.modifications` to `/etc/inetd.conf` or `/etc/inet/inetd.conf`, or copy `$GLOBUS_LOCATION/share/myproxy/etc.xinetd.myproxy` to `/etc/xinetd.d/myproxy`. You'll need to modify the paths in the file according to your installation.
- Reactivate the `inetd` (or `xinetd`). This is typically accomplished by sending the `SIGHUP` signal to the daemon. Refer to the `inetd` or `xinetd` man page for your system.

## Testing

To verify your `myproxy-server` installation and configuration, you can run the `myproxy-server` directly from your shell. If using a host certificate, you will need to run the `myproxy-server` as root. First, make sure your Globus environment is setup in your shell. Set the `GLOBUS_LOCATION` environment variable to the location of your MyProxy installation. Then, depending on your shell, run one of the following commands.

For `csh` shells:

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

For `sh` shells:

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

Then, run `$GLOBUS_LOCATION/sbin/myproxy-server -d`. The `-d` argument runs the `myproxy-server` in debug mode. It will write debugging messages to the terminal and exit after servicing a single request. You'll need to start it once for each test request. In another shell, you can run the MyProxy client programs to test the server.

If run without the `-d` argument, the `myproxy-server` program will start up and background itself. It accepts connections on TCP port 7512, forking off a separate child to handle each incoming connection. It logs information via the `syslog` service under the daemon facility.

## Security Considerations

```
import
```

## Notes

1. <http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/myproxy/admin/index.htm>



# Chapter 13. Configuring CAS

## Introduction

This guide contains advanced configuration information for system administrators working with the Community Authorization Service (CAS). It provides references to information on procedures typically performed by system administrators, including installation, configuring, deploying, and testing the installation.

*Note:* This document contains information about deploying a CAS server and is not needed for a CAS client installation. [expand on this a little more]

This is a partially-complete docbook translation of the CAS Admin Guide<sup>1</sup>. Please see that document for additional information.

## Configuring

PHP Include

## Deploying

The following steps are needed to deploy the CAS service.

- Obtaining credentials for the CAS server
- Database installation and configuration

## Obtaining credentials for the CAS server

The CAS service could run with its own set of credentials. Instructions to obtain service credentials may be found here<sup>2</sup>.

The standard administrator clients that come with the distribution can be used to perform host authorization and expect that the CAS service have credentials that have service name "cas". The command in the above mentioned web page<sup>3</sup> may be altered as follows:

```
casadmin$ grid-cert-request -service cas -host FQDN
```

The certificate and private key are typically placed in `/etc/grid-security/cas-cert.pem` and `/etc/grid-security/cas-key.pem`, respectively. In this document, the location of certificate and key files is referred to as `CAS_CERT_FILE` and `CAS_KEY_FILE`, respectively.

## Database installation and configuration

CAS uses a backend database to store all user data. Any JDBC compliant database may be used. This section describes briefly the installation of such a database and the creation of the database using schema required for the CAS backend.

## Installing the database

Any JDBC compliant database may be used. PostgreSQL has been used for development and testing. The drivers for the same are included in the distribution.

If a different database is used, the corresponding driver should be added to `GLOBAL_LOCATION/lib`.

Brief instructions to install a JDBC compliant database and specifically PostGres can be found here<sup>4</sup>. For more detailed instructions, please refer to specific database documentation.

## Initializing the CAS database

The schema of the database that needs to be created for CAS can be found at `$GLOBAL_LOCATION/globus_cas_service/casDbSchema/cas_psql_database_schema.sql`

To create a database, for example *casDatabase*, on a PostgreSQL, install on local machine:

```
casadmin$ createdb casDatabase
casadmin$ psql -U casadmin -d casDatabase -f \
    $GLOBAL_LOCATION/globus_cas_service/casDbSchema/cas_psql_database_schema.sql
```

You will see a list of notices on the screen. Unless any of them say "ERROR", these are just output for information.

Information about the database needs to be configured as described here<sup>5</sup> for the CAS service to be able to use it.

## Testing

CAS has two sets of tests, one for the backend database access module and another set to test the service itself. To install both tests, install the CAS test package (*gt4-cas-delegation-test-3.9-src\_bundle.tar.gz*) using *GPT FILLME: instructions* into `GLOBAL_LOCATION`.

It is assumed that:

- a backend database has been set up and configured
- CAS service and tests are installed in `$GLOBAL_LOCATION`
- the service is running on localhost and port 8080 (for the sample commands)
- the database is set up with:
  1. username as *tester*
  2. database name as *casDatabase*
  3. host as *foo.bar.gov* and default port.

## Testing the backend database module

1. Run:

```
cd $GLOBAL_LOCATION
```

2. Populate the file `etc/globus_cas_unit_test/casTestProperties` with the following database configuration information:

Table 13-1. TITLE

dbDriver	The JDBC driver to be used
dbConnectionURL	JDBC connection url to be used to connect to database
dbUsername	user name to connect to database as
dbPassword	database password for the said username

3. The database needs to be empty for these tests to work. To delete all the data in the database, run:

```
psql -d casDatabase -U tester -h foo.bar.gov -f etc/globus_cas_utils/database_delete
```

4. Run:

```
ant -f share/globus_cas_unit_test/cas-test-build.xml testDatabase
```

5. Test reports are placed in \$GLOBUS\_LOCATION/share/globus\_cas\_unit\_test/cas-test-reports.

## Testing CAS service module

These tests can be set up so as to be able to test multiple user scenario or it can be configured to run just as a single identity. The first set of tests are used to test admin functionality and sets up the database for second user. As the second user the permissions and queries are tested to ensure that the set up worked.

All the configuration information for the test, needs to be configured in *etc/globus\_cas\_unit\_test/casTestProperties* file. The database section of the properties file needs to be configured as described here<sup>6</sup>. Other than those, the following properties also need to be configured for these tests:

Table 13-2. TITLE

user1SubjectDN	The DN of the user running the first set of tests.
user2SubjectDN	The DN of the user running the second set of tests. Can be the same as previous property
maxAssertionLifetime	Should match the value set in service configuration as shown in Configuration information <sup>7</sup>

Steps for testing:

1. Run:

```
cd $GLOBUS_LOCATION
```

2. The database needs to be empty for these tests to work. To delete all the data in the database, run:

```
psql -d casDatabase -U tester -h foo.bar.gov -f etc/globus_cas_utils/database_delete
```

3. In the test properties file, set *user2SubjectDN* to be the subject in your regular proxy. The following returns the appropriate string:

```
casadmin$ bin\grid-cert-info -subject -globus
```

4. Generate an independent proxy using the following command:

```
casadmin$ bin\grid-proxy-init -independent
```

5. Set the identity in the proxy generated from the above step as *user1SubjectDN* in the test properties file. The following command will return the relevant string:

```
casadmin$ java org.globus.tools.ProxyInfo -subject -globus
```

6. Start a container on port (for example, testPort) and host (for example, host-Port).

7. The following command runs tests for self permissions and sets up the database for user with subjectDN that is user2SubjectDN:

```
casadmin$ ant -f share/globus_cas_unit_test/cas-test-build.xml -  
Dtest.port=testPort \  
-Dtest.host=testHost user1TestService
```

8. To test as the second user, generate proxy for the subject DN specified for the second user:

```
casadmin$ bin/grid-proxy-init
```

9. To test as second user, run the following. If you are not running container on default host/port, set properties as shown here.<sup>7</sup>

```
casadmin$ ant -f share/globus_cas_unit_test/cas-test-build.xml -  
Dtest.port=testPort \  
-Dtest.host=testHost user2TestService
```

10. Test reports are placed in \$GLOBUS\_LOCATION/share/globus\_cas\_unit\_test/cas-test-reports.

11. After these tests, the CAS database needs to be reset. The following command will delete all entries from the database:

```
casadmin$ psql -U casadmin -d casDatabase -f GLOBUS_LOCATION/share/cas/database_c
```

## Example of CAS Server Administration

The following contains an example of administering the CAS server policies using the CAS administrative clients described here<sup>8</sup>.

Alice, Bob and Carol are three members of a community who have set up a Community Authorization service:

- Alice's role is primarily to administer the CAS server.
- Bob is an analyst who needs read access to much of the community data.
- Carol is a scientist who needs to be able to both read and write community data.

These examples show how:

1. Alice adds the users Bob and Carol to the CAS server
2. Adds a FTP server with some data available to the community
3. Adds permissions for the users using the CAS administration clients.

These examples assume the following:

- Alice has installed the CAS server and bootstrapped the database with herself as super user. Please refer to installation documentation<sup>9</sup> for details.
- Alice's nickname on the CAS server is *alice* and at bootstrap she has created a user group, *suGroup*, which has super user permissions on the database.
- The CAS service URL is `http://localhost:8080/ogsa/services/base/cas/CASService`.
- For all commands listed below, the environment variable, `GLOBUS_LOCATION`, has been set to point to the GT core install and the commands are run from `GLOBUS_LOCATION/bin`.
- An environment variable, `CAS_SERVER_URL`, has been set to point to the CAS server URL, `http://localhost:8080/ogsa/services/base/cas/CASService`.

## 1. Adding a user group

Since at the time of booting up the CAS server, only one user group that has super user permissions on the CAS server is created, Alice might want to create another user group to which new users maybe added and permissions to newly enrolled CAS entities may be given. This also eases the process of giving same rights to many users. Given they are two types of roles in the community she might want to create two groups, *analysts* and *scientists*.

Also, all permissions on the newly created group will be given to users of a particular user group. Say, Alice would like all users of the user group *analysts* to be able to manipulate the group.

To create a new user group Alice uses the *cas-group-admin* client. It requires a name for the new group being created, say *analysts*.

```
alice% cas-group-admin user create analysts analysts
```

This will create a user group *analysts* and give all users in that group the permission to manage the group (i.e add users, remove users and so on). She can similarly create a group called *scientist*

## 2. Adding a trust anchor

Prior to adding Bob and Carol to CAS server, Alice needs to ensure that the trust anchors for both have been added. If they share the same trust anchor with Alice then this step can be skipped, since at bootstrap Alice's trust anchor would have been added to the database.

In our example, Alice and Carol share a trust anchor different from Bob's. Therefore, Alice needs to add Bob's trust anchor by using *cas-enroll* client with the *trustAnchor* option. She needs to provide details about the trust anchor such as the authentication method and authentication data used.

```
alice% cas-enroll trustAnchor analysts AbcTrust X509 "/C=US/O=some/CN=ABC CA"
```

The above will enroll a trust anchor with nickname *AbcTrust*, who uses *X509* as authentication method and has the DN specified in the command. The members of the *analysts* user group are given all rights on this object. This implies that any user who has this trust anchor, would present credentials signed by this trust anchor.

### 3. Adding users

Now, Alice can add Bob and Carol as users using the *cas-enroll* command with the *user* option. She needs to provide the user's subject DN and a reference to the trust anchor used by the user. As with any entity added to the CAS server, the admin needs to choose a user group whose members will have all permissions on that entity. In this example, Alice would like the members of user group *suUser* to be able to manipulate the user entity *Bob*.

```
alice% cas-enroll user suUser bob "/O=Our Community/CN=Bob Foo" AbcTrust
```

Alice uses a similar command to add Carol to the CAS database.

### 4. Adding users to a user group

CAS server allows rights to be assigned only to user groups and not to individual users. Hence before Alice can assign rights to Bob or Carol, she needs to add them to some user group. She does this by using *cas-group-add-entry* client with the *user* option to indicate she is adding to a user group. This client requires the group name and nick name of the user who needs to be added. To add Bob to the *analysts* group, the command would be:

```
alice% cas-group-add-entry user analysts bob
```

If a user group *scientists* was created, Carol could similarly be added as a member.

### 4. Adding a new FTP server

Alice now has the community users in the database and organized. She now wants to add some resources. Because the community currently has a single FTP server, *foo.bar.edu*, available to it, she will add this to the CAS database.

Each resource or object in the CAS server has a namespace associated to it that defines certain features. For example, it can define the comparison algorithm that needs to be used when this object name is used for comparison. It may also define the base URL that should be prefixed to objects that belong to this namespace. In this case, Alice chooses to use the *FTPDirectoryTree* namespace that is added to the CAS server at bootup. She uses the *cas-enroll* client with the *object* option to add the FTP server to the CAS database:

```
alice% cas-enroll object suGroup ftp://foo.bar.edu/* FTPDirectoryTree
```

This command adds the FTP server as an object and gives all members of the *suGroup* rights to manipulate the object.

To be able to grant/revoke access on an individual directory, add an object for the directory. For example, if Alice would like to be able to manipulate the *data* on the server as a separate entity, the following command will add an object for that.

```
alice% cas-enroll object suGroup ftp://foo.bar.edu/data/* FTPDirectoryTree
```

### 4. Creating an object group

Alice suspects the community will end up with more directories containing data on other servers that will have identical policies as the */data* directory on *foo.bar.edu*. So she is going to create an object group called *data* and assign *foo.bar.edu/data* to this group. This will allow her to grant rights on this group and easily add other directories to this group later.

To create a group called *data*, she uses the *cas-group-admin* client with the *group* and *create* options:

```
alice% cas-group-admin object create suGroup data
```

This creates an object group called *data* and the members of *suGroup* get all rights on this group and hence should be able to add/remove members, grant rights to add/delete from this group to others and also delete this group.

## 5. Adding members to an object group

Alice now can add *foo.bar.edu/data* to the *data* group. She can do this by using the *cas-group-add-entry* with the *object* option. To add the above described object, *ftp://foo.bar.edu/data/\** in namespace *FooFTPNamespace*, to object group *data*, Alice uses the following command:

```
alice% cas-group-add-entry object data object FooFTPNamespace ftp://foo.bar.edu/data/*
```

In the above command:

- the first *object* refers to the group type.
- *data* is the name of the object group.
- the second *object* refers to the type of CAS entity that is being added as a member.
- the last two parameters define the namespace and the object that needs to be added.

## 6. Adding service types

Alice now needs to add information about the kinds of rights that can be granted for these objects. These are stored as *service types* and relevant actions are mapped to these service types.

In this scenario, the kind of service types that Alice should add would be *file*, *directory* and so on. To do so, the *cas-enroll* client with *serviceType* option may be used. To add a service type called *file* and give members of *suGroup* all rights on this service type, Alice uses the following command.

```
alice% cas-enroll serviceType suGroup file
```

## 7. Adding action mappings

The relevant action mappings to the above mentioned service types would be *read*, *write* and so on. Alice needs to add these mappings to the database so that she can grant rights that allow a user to have *file/read* or *file/write* permissions on some object.

To add action mappings to a service type, she uses the *cas-action* client with *add* option. The following command should add a mapping of action *read* to service type *file*.

```
alice% cas-action add file add
```

Similarly, she can add other mappings, like *write*, to this service type.

## 8. Granting permissions

Alice now has resources in the object group *data* and users in the user groups *analysts* and *scientists*. She now wants to grant permissions on *data* group to the analysts and scientists, namely read permissions to the analysts and read and write permissions to the scientists.

To grant permissions, Alice needs to use the *cas-rights-admin* with the *grant* option. To give read permissions to the analysts group, Alice runs:

```
alice% cas-rights-admin grant analysts objectGroup data serviceAction file read
```

She similarly grants rights to *scientists* group.

## Security Considerations

import

## Notes

1. <http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/cas/admin/index.html>
2. <http://www.globus.org/toolkit/docs/2.4/admin/guide-verify.html#ldapcert>
3. <http://www.globus.org/toolkit/docs/2.4/admin/guide-verify.html#ldapcert>
4. <http://www-unix.globus.org/toolkit/3.0/ogsa/docs/admin/installation.html>
5. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/cas/WS\\_AA\\_CAS\\_Public\\_I](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/cas/WS_AA_CAS_Public_I)
6. testDbConfig
7. #testParams
8. [http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/cas/WS\\_AA\\_CAS\\_Public\\_I](http://www-unix.globus.org/toolkit/docs/development/3.9.5/security/cas/WS_AA_CAS_Public_I)
9. ./installGuide.html

## Appendix A. Installing SimpleCA

The following are instructions for how to use SimpleCA to set up certificates for a GT 3.9.5 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 the Section called *Obtain host certificates* in Chapter 5.

### 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.

### 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 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 A-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).

### 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.

### 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).

### 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.

## 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.

## 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
```

## 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 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 the Section called *User certificates*.

## 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 the Section called *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.

## User certificates

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

## 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.

## 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 the Section called *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.

## 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 looks 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
```

## Configure SimpleCA for multiple machines

So far , you have a single machine configured with SimpleCA certificates. Recall that in the Section called *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 the Section called *Sign the host certificate*.

If you are going to run services on the second host, it will need its own the Section called *Host certificates* and `grid-mapfile` (as described in the basic configuration instructions in the Section called *Add authorization* in Chapter 5).

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.