

GT 4.2.1: GSI C Admin Guide

GT 4.2.1: GSI C Admin Guide

Introduction

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

Important

The security tools are installed as part of the Globus Toolkit installation process. For instructions on basic installation of the Globus Toolkit, see the [Installing GT 4.2.1](#).

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.
-

Table of Contents

1. Configuring Certificates	1
1. Configuring Globus to Trust a Particular Certificate Authority	1
2. Configuring Globus to Create Appropriate Certificate Requests	2
3. Requesting Service Certificates	4
4. Configuring Credential Mappings	4
5. GSI File Permissions Requirements	6
2. Testing	8
3. Security Considerations	9
1. Security considerations for GSI C	9
4. Debugging	10
1. Logging	10
5. Troubleshooting	11
1. Credential Troubleshooting	11
2. Grid map Troubleshooting	14
Glossary	15

List of Tables

1.1. CA files	1
1.2. Certificate request configuration files	2
1.3. Certificate request files	4
1.4. Gridmap File Location Algorithm	5
1.5. Authorization Configuration File Locations	6
1.6. Authorization Configuration File Locations	6
5.1. Credential Errors	12
5.2. Gridmap Errors	14

Chapter 1. Configuring Certificates

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 system-wide location if no user-specific file was found. The configuration commands described here may be run by administrators to create system-wide defaults and by individuals to override those defaults.

1. Configuring Globus to Trust a Particular Certificate Authority

1.1. Trusted certificates directory

The Globus tools will trust certificates issued by a CA if (and only if) it can find information about the CA in the trusted certificates directory.

The trusted certificates directory is located as described below and exists either on a per-machine or on a per-installation basis.

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

1. value of the X509_CERT_DIR environment variable
2. \$HOME/.globus/certificates
3. /etc/grid-security/certificates exists
4. \$GLOBUS_LOCATION/share/certificates

1.2. Trusted certificates files

The following two files must exist in the directory for each trusted CA:

Table 1.1. CA files

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

Non-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.

1.3. Hash of the CA certificate

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
```

1.4. Creating a signing policy by hand

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.

1.5. Repository of CAs

A repository of CA certificates that are widely used in academic and research settings can be found [here](#)¹.

2. Configuring Globus to Create Appropriate Certificate Requests

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

Table 1.2. Certificate 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 ² .
<code>globus-host-ssl.conf</code>	Defines the distinguished name for a host (or service) certificate request. The format is described here ³ .
<code>grid-security.conf</code>	A base configuration file that contains the name and email address for the CA.
<code>directions</code>	An optional file that may contain directions on using the CA.

¹ <https://www.tacar.org/certs.html>

² http://www.openssl.org/docs/apps/req.html#CONFIGURATION_FILE_FORMAT

³ http://www.openssl.org/docs/apps/req.html#CONFIGURATION_FILE_FORMAT

Many CAs provide tools to install configuration files with the following names in the Trusted Certificates directory:

- `globus-user-ssl.conf.cert_hash`
- `globus-host-ssl.conf.cert_hash`
- `grid_security.conf.cert_hash`
- `directions.cert_hash`

2.1. Creating a certificate request for a specific CA

The command:

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

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

2.2. Listing available CAs

The command:

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

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

2.3. Specifying a default CA for certificate requests

The default CA is the CA that will be used for certificate requests if **`grid-cert-request`** is invoked without the `-ca` flag.

You can specify a default CA by invoking the **`grid-default-ca`** command (follow the link for examples of using the command).

2.4. directions file

The `directions` file may contain specific directions on how to use the CA. There are three types of printed messages:

- *REQUEST HEADER*, printed to a certificate request file,
- *USER INSTRUCTIONS*, printed on the screen when one requests a *user certificate*,
- *NONUSER INSTRUCTIONS*, printed on the screen when one requests a certificate for a service.

Each message is delimited from others with lines `----- BEGIN message type TEXT -----` and `----- END message type TEXT -----`. For example, the `directions` file would contain the following lines:

```
----- BEGIN REQUEST HEADER TEXT -----
```

```
This is a Certificate Request file
```

```
It should be mailed to ${GSI_CA_EMAIL_ADDR}
```

```
----- END REQUEST HEADER TEXT -----
```

If this file does not exist, the default messages are printed.

3. 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 1.3. Certificate request files

<code>GRID_SECURITY/service_name/service_namecert.pem</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_namecert_request.pem</code>	The certificate request, which you should send to your CA.
<code>GRID_SECURITY/service_name/service_namekey.pem</code>	The <i>private key</i> associated with your certificate request, encrypted with the pass phrase that you entered when prompted by grid-cert-request .

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

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

4. Configuring Credential Mappings

Several Globus services map certificates to local unix usernames to be used with unix services. The default implementation uses a *gridmap* file to map the distinguished name of the identity of the client's certificate to a local login name. Administrators can modify the contents of the gridmap file to control what certificate identities are allowed to access Globus services, as well as configure, via an environment variable, what gridmap file a particular service uses.

In addition to the identity-based mapping done via the gridmap file, administrators can configure Globus services to use arbitrary mapping functions. These may use other criteria, such as SAML assertions, to map a certificate to a local account, or may map certificates to temporary accounts. Administrators can install different mapping implementations and configure services to use them by creating appropriate configuration files and setting environment variables.

4.1. Configuring Identity Mappings Using *gridmap* Files

Gridmap files contain a database of entries mapping distinguished names to local user names. These may be manipulated by using the following tools.

4.1.1. Adding an entry to a gridmap file

To add an entry to the gridmap file, run:

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

4.1.2. Deleting an entry from a gridmap file

To delete an entry from the gridmap file, run:

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

4.1.3. Checking consistency of a gridmap file

To check the consistency of the gridmap file, run

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

4.1.4. Configuring per-service gridmap files

To configure a service to use a particular gridmap file, set the GRIDMAP variable in the service's environment to the path of the gridmap file. In this way, you can grant different access rights to different certificate identities on a per-service basis by setting the GRIDMAP variable in different service environments.

You can use tools described above to operate on different gridmap files by either setting the GRIDMAP environment variable prior to invoking them, or by using the `-mapfile` command-line option.

For reference, the GSI C code looks for the gridmap in these locations:

Table 1.4. Gridmap File Location Algorithm

Location	notes
GRIDMAP environment variable	
<code>/etc/grid-security/grid-mapfile</code>	Only for services running as root.
<code>HOME.gridmap</code>	Only for services not running as root.

4.1.5. Gridmap formats

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.

For more detailed information about the gridmap file see the [file description and grammars](https://dev.globus.org/wiki/Gridmap)⁴ on dev.globus.org.

⁴ <https://dev.globus.org/wiki/Gridmap>

4.2. Configuring Alternate Credential Mappings

To use an alternative credential mapping, you create a `gsi-authz.conf` file containing information about how the mapping functions are called from the authorization library.

To configure a per-service authorization configuration file, set the `GSI_AUTHZ_CONF` variable to the path to the configuration file in the environment of the service.

For reference, the GSI C code looks for the authorization configuration file in these locations (in the given order):

Table 1.5. Authorization Configuration File Locations

Location
<code>GSI_AUTHZ_CONF</code> environment variable
<code>/etc/grid-security/gsi-authz.conf</code>
<code>GLOBUS_LOCATION/etc/gsi-authz.conf</code>
<code>HOME/.gsi-authz.conf</code>

4.2.1. Callout File Format

The authorization file defines a set of callouts, one per line. Each callout is defined by an *abstract type*, *library*, and *symbol* separated by whitespace. Comments begin with the `#` character and continue to the end of line.

Table 1.6. Authorization Configuration File Locations

Field	Meaning
<i>abstract type</i>	Type of the callout: <i>globus_mapping</i> is used for credential mapping callouts
<i>library</i>	Path to the shared object containing the callout implementation. The library name may be a literal filename, or a partial filename to which the compilation flavor of the service is appended to the filename before its extension.
<i>symbol</i>	The exported symbol containing the entry point to the callout implementation.

Here is a sample `gsi-authz.conf` file that configures a *globus_mapping* callout to use the *globus_gridmap_callout* function in the `/usr/local/globus/lib/libglobus_gridmap_callout_gcc32dbg` shared object:

```
# abstract-type      library                                     symbol
globus_mapping      /opt/globus/lib/libglobus_gridmap_callout_gcc32dbg globus_gridmap_call
```

5. GSI File Permissions Requirements

- End Entity Certificate (User, Host and Service) Certificates and the GSI Authorization Callout Configuration File:
 - May not be executable
 - May not be writable by group and other
 - Must be either regular files or soft links
- Private Keys and Proxy Credentials:

- 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, CA Signing Policy Files, the Grid Map File and the GAA Configuration File:
 - Must be either regular files or soft links
- GSI Authorization callout configuration files
 - Must exist
 - Should be world readable
 - Should not be writable by group and other
 - Should be either a regular file or a soft link
- GSI GAA configuration files
 - Must exist
 - Should be world readable
 - Should not be writable by group and other
 - Should be either a regular file or a soft link

Chapter 2. Testing

There is no content available at this time.

Chapter 3. Security Considerations

1. Security considerations for GSI C

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

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



Note

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



Note

If a host was set up to accept "hostname-1.edu", it will not accept any of "hostname-*ANYTHING*.edu" but will accept "hostname.edu". That is, only one of the names being compared may contain the hyphen character in the host name.

A [bug¹](#) has been opened to see if this feature needs to be modified.

In GT 4.2.1, it is possible to disable this behavior, by setting the environment variable `GLOBUS_GSS-API_NAME_COMPATIBILITY` to `STRICT_RFC2818`.

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

Chapter 4. Debugging

1. Logging

N/A

Chapter 5. Troubleshooting

For a list of common errors in GT, see [Error Codes](#).

1. Credential Troubleshooting

1.1. Credential Errors

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

For a list of common errors in GT, see [Error Codes](#).

Table 5.1. Credential Errors

Error Code	Definition	Possible Solutions
Your proxy credential may have expired	Your proxy credential may have expired.	Use <code>grid-proxy-info</code> to check whether the proxy credential has actually expired. If it has, generate a new proxy with <code>grid-proxy-init</code> .
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.	Check the system clocks on the local and remote system.
Your end-user certificate may have expired	Your end-user certificate may have expired	Use <code>grid-cert-info</code> 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 <code>grid-proxy-destroy</code>) and creating a new one (with <code>grid-proxy-init</code>). Important: 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), <code>grid-proxy-init</code> will refuse to create a proxy certificate.	You can "fix" this by changing the permissions on the private key file. Important: However, you will still have a much more serious problem: it is 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	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 Installing GT 4.2.1 for details.
You may not trust the remote system's CA	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 Installing GT 4.2.1 for details.
There may be something wrong with the remote service's credentials	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 cannot find anything wrong with your credentials, check for the same conditions on the remote system (or ask a remote administrator to do so).

1.2. Some tools to validate certificate setup

1.2.1. grid-cert-diagnostics

The **grid-cert-diagnostics** program checks prints diagnostics about the user's certificates, and host security environment.

```
% grid-cert-diagnostics -p
```

1.2.2. Check that the user certificate is valid

```
openssl verify -CApath /etc/grid-security/certificates
-purpose sslclient ~/.globus/usercert.pem
```

1.2.3. Connect to the server using s_client

```
openssl s_client -ssl3 -cert ~/.globus/usercert.pem -key
~/.globus/userkey.pem -CApath /etc/grid-security/certificates
-connect <host:port>
```

Here `<host:port>` denotes the server and port you connect to.

If it prints an error and puts you back at the command prompt, then it typically means that the *server* has closed the connection, i.e. that the server was not happy with the client's certificate and verification. Check the SSL log on the server.

If the command "hangs" then it has actually opened a telnet style (but secure) socket, and you can "talk" to the server.

You should be able to scroll up and see the subject names of the server's verification chain:

```
depth=2 /DC=net/DC=ES/O=ESnet/OU=Certificate Authorities/CN=ESnet Root CA 1
verify return:1
depth=1 /DC=org/DC=DOEGrids/OU=Certificate Authorities/CN=DOEGrids CA 1
verify return:1
depth=0 /DC=org/DC=doegrids/OU=Services/CN=wiggum.mcs.anl.gov
verify return:1
```

In this case, there were no errors. Errors would give you an extra line next to the subject name of the certificate that caused the error.

1.2.4. Check that the server certificate is valid

Requires root login on server:

```
openssl verify -CApath /etc/grid-security/certificates -purpose sslserver
/etc/grid-security/hostcert.pem
```

2. Grid map Troubleshooting

2.1. Grid map errors

The following are some common problems that may cause clients or servers to report that user are not authorized:

For a list of common errors in GT, see [Error Codes](#).

Table 5.2. Gridmap Errors

Error Code	Definition	Possible Solutions
The content of the grid map file does not conform to the expected format	The content of the grid map file does not conform to the expected format	Run grid-mapfile-check-consistency to make sure that your gridmap file conforms to the expected format.
The grid map file does not contain a entry for your DN	The grid map file does not contain a entry for your DN	Use grid-mapfile-add-entry to add the relevant entry.

Glossary

some terms not in the docs but wanted in glossary: [scheduler](#)

C

Certificate Authority (CA)	An entity that issues certificates. [fixme - flesh out]
CA Certificate	The CA's certificate. This certificate is used to verify signature on certificates issued by the CA. GSI typically stores a given CA certificate in <code>/etc/grid-security/certificates/<hash>.0</code> , where <code><hash></code> is the hash code of the CA identity.
CA Signing Policy	The CA signing policy is used to place constraints on the information you trust a given CA to bind to public keys. Specifically it constrains the identities a CA is trusted to assert in a certificate. In GSI the signing policy for a given CA can typically be found in <code>/etc/grid-security/certificates/<hash>.signing_policy</code> , where <code><hash></code> is the hash code of the CA identity.

E

End Entity Certificate (EEC)	A certificate belonging to a non-CA entity, e.g. you, me or the computer on your desk.
------------------------------	--

G

GAA configuration file	A file that configures the Generic Authorization and Access control GAA libraries. When using GSI, this file is typically found in <code>/etc/grid-security/gsi-gaa.conf</code> .
grid map file	A file containing entries mapping certificate subjects to local user names. This file can also serve as a access control list for GSI enabled services and is typically found in <code>/etc/grid-security/grid-mapfile</code> . For more information see the Gridmap section here .
GSI authorization callout configuration file	A file that configures authorization callouts to be used for mapping and authorization in GSI enabled services. When using GSI this file is typically found in <code>/etc/grid-security/gsi-authz.conf</code> .

H

host certificate	An EEC belonging to a host. When using GSI this certificate is typically stored in <code>/etc/grid-security/hostcert.pem</code> . For more information on possible host certificate locations see the GSI C Developer's Guide .
------------------	---

P

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

or `/etc/grid-security/<service>/<service>key.pem` (for service certificates).

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

proxy credentials

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

S

scheduler

Term used to describe a job scheduler mechanism to which GRAM interfaces. It is a networked system for submitting, controlling, and monitoring the workload of batch jobs in one or more computers. The jobs or tasks are scheduled for execution at a time chosen by the subsystem according to an available policy and availability of resources. Popular job schedulers include Portable Batch System (PBS), Platform LSF, and IBM LoadLeveler.

service certificate

A EEC for a specific service (e.g. FTP or LDAP). When using GSI this certificate is typically stored in `/etc/grid-security/<service>/<service>cert.pem`. For more information on possible service certificate locations, see [this](#).

U

user certificate

A EEC belonging to a user. When using GSI, this certificate is typically stored in `$HOME/.globus/usercert.pem`. For more information on possible user certificate locations, see [this](#).