

GT 4.2.1 GSI C: Developer's Guide

GT 4.2.1 GSI C: Developer's Guide

Introduction

This component provides an API for authentication and two APIs for authorization.

The authentication API is an implementation of the GSS-API (RFC 2743 and RFC 2744) extended with the functions described in the GSS-API Extensions document.

On the authorization front there is a coarse-grained API, which in addition to authorizing also provides a mapping function, and an API that allows finer grained authorization decisions to be made. The finer grained API follows the subject, object, action paradigm.

Both of the authorization APIs allow different back end implementations through the use of dynamic library loading.

Table of Contents

1. Before you begin	1
1. Feature summary	1
2. Tested platforms	1
3. Backward compatibility summary	1
4. Technology dependencies	1
5. Security considerations for GSI C	2
2. Usage scenarios	3
3. Tutorials	4
4. Architecture and design overview	5
1. Authentication	5
2. Authorization	6
5. APIs	7
6. Protocol Specifications	9
1. GSI Message Specification	9
I. GSI Commands	10
grid-cert-diagnostics	11
grid-cert-info	13
grid-cert-request	15
grid-default-ca	18
grid-change-pass-phrase	20
grid-proxy-init	21
grid-proxy-destroy	24
grid-proxy-info	25
grid-mapfile-add-entry	27
grid-mapfile-check-consistency	28
grid-mapfile-delete-entry	29
7. Configuring Certificates	30
1. Configuring Globus to Trust a Particular Certificate Authority	30
2. Configuring Globus to Create Appropriate Certificate Requests	31
3. Requesting Service Certificates	33
4. Configuring Credential Mappings	33
5. GSI File Permissions Requirements	35
8. Environment variable interface	37
1. Environmental Variables for GSI C	37
9. Debugging	41
10. Troubleshooting	42
1. Credential Troubleshooting	42
2. Grid map Troubleshooting	45
11. Related Documentation	46
Glossary	47

List of Tables

1. Command line options	20
2. Command line options	22
3. Command line options	24
4. Command line options	25
5. Print options	25
6. Validity options	26
7. Command line options	27
8. Command line options	28
9. Command line options	29
7.1. CA files	30
7.2. Certificate request configuration files	31
7.3. Certificate request files	33
7.4. Gridmap File Location Algorithm	34
7.5. Authorization Configuration File Locations	35
7.6. Authorization Configuration File Locations	35
10.1. Credential Errors	43
10.2. Gridmap Errors	45

Chapter 1. Before you begin

1. Feature summary

Features new in GT 4.2.1

- Support for processing host certificates containing X.509 subjectAltName extensions with dNSName or iPAddress values.

Other Supported Features

- Authentication of user using standard X.509 End Entity and *Proxy Certificates*.
- Delegation using X.509 Proxy Certificates.
- Pluggable authorization based on the client's certificate chain for GridFTP and GRAM2.
- Pluggable authorization for GRAM2 based on the RSL of the job.

Deprecated Features

- None

2. Tested platforms

Tested platforms for GSI C:

- i386 Linux

3. Backward compatibility summary

Protocol changes in GSI C since GT 4.0.x

- None

API changes since GT 4.0.x

- None

Exception changes since GT 4.0.x

- Not applicable

Schema changes since GT 4.0.x

- Not applicable

4. Technology dependencies

The GSI C component depends on the following GT components:

- C Common Libraries

The GSI C component depends on the following 3rd party software:

- OpenSSL

5. 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 DN's "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 2. Usage scenarios

There is no content available at this time.

Chapter 3. Tutorials

There are no tutorials available at this time

Chapter 4. Architecture and design overview

1. Authentication

As mentioned in the introduction, the GSI C security framework uses the GSSAPI API and extensions to it to abstract security mechanism specific details. Below the GSSAPI layer there exist multiple APIs for dealing with credential management, X.509 certificates in general and *proxy certificates* in particular as well as security configuration. Each of these APIs is described in more detail below.

The general design principle guiding these APIs is data encapsulation. Data structures (handles and attributes) capture and encapsulate the state of the system. These data structures are then acted upon by various getters and setters, as well as other functions.

1.1. The GSS Assist API

The GSS Assist API provides helper functions wrapping the process of security (GSS) context establishment, support for gridmap authorization and various other helper functions that wrap GSSAPI functions and capture common usage.

1.2. GSSAPI

The GSSAPI implementation provided by the toolkit is based upon SSL/TLS with extensions to the standard path validation mechanism to handle proxy certificates. It relies upon the credential and certificate utility APIs for general certificate acquisition and inspection functionality.

1.3. The Callback API

This API provides a callback that can be plugged into the OpenSSL path validation framework. This callback provides the additions to path validation required for dealing with proxy certificates and X.509 extensions. Furthermore, it allows applications to inspect data, e.g. the validated certificate chain, after the validation is done.

1.4. The Certificate Utilities API

The Certificate Utilities API provides helper functions for dealing with X.509 certificates. This API does not use the "handle" concept mentioned in the introduction. Rather, it operates on datatypes provided by the OpenSSL APIs.

1.5. The Credential API

The Credential API deals with reading and writing certificates from and to the file system and the OpenSSL I/O abstraction layer. It also provides functions for inspecting and validating the read credentials.

1.6. The Proxy APIs

The Proxy APIs provide an implementation of the X.509 Proxy Certificate Extension ASN.1 structure as well as functions for creating new proxies.

1.7. The System Configuration API

This API serves as a abstraction layer for OS specific information needed by the security infrastructure. It provides OS specific functions for discovering certificates from a set of predefined standard locations as well as functions for doing the same for various configuration files.

2. Authorization

As described in the introduction the GSI C security framework essentially provides two authorization APIs, the generic Authorization API and the Gridmap API. These APIs differ in various ways:

The Authorization API provides a framework that allows callouts to 3rd party authorization solutions, does not provide a default authorization mechanism and is geared to authorizing the subject-action-object tuple.

The Gridmap API on the other hand, while allowing for custom callouts to be plugged in and override the default behavior, provides a default authorization and mapping mechanism based on the *grid map file*. Also, it only furnishes the callouts with information about the entity to be authorized, i.e. it does not provide information on the action and the object, so it is somewhat simpler in its approach. Finally, it provides the ability to map authorized entities to local system entities, e.g. UNIX user names. More information on the interface used for Gridmap callouts can be found [here](#)¹.

¹ ../GSIAuthorizationCalloutSpecification-04.pdf

Chapter 5. APIs

Documentation for the APIs in this component can be found here:

- [gaa_core](#)¹ [no frames²]
- [gaa_gss_generic](#)³ [no frames⁴]
- [gaa_plugin](#)⁵ [no frames⁶]
- [globus_authz](#)⁷ [no frames⁸]
- [globus_authz_callout_error](#)⁹ [no frames¹⁰]
- [globus_gridmap_callout_error](#)¹¹ [no frames¹²]
- [globus_gsi_callback](#)¹³ [no frames¹⁴]
- [globus_gsi_cert_utils](#)¹⁵ [no frames¹⁶]
- [globus_gsi_credential](#)¹⁷ [no frames¹⁸]
- [globus_gsi_openssl_error](#)¹⁹ [no frames²⁰]
- [globus_gsi_proxy_core](#)²¹ [no frames²²]
- [globus_gsi_proxy_ssl](#)²³ [no frames²⁴]
- [globus_gsi_sysconfig](#)²⁵ [no frames²⁶]
- [globus_gss_assist](#)²⁷ [no frames²⁸]

¹ http://www.globus.org/api/c-globus-4.0/gaa_core/html/index.html#_top

² http://www.globus.org/api/c-globus-4.0/gaa_core/html/main.html#_top

³ http://www.globus.org/api/c-globus-4.0/gaa_gss_generic/html/index.html#_top

⁴ http://www.globus.org/api/c-globus-4.0/gaa_gss_generic/html/main.html#_top

⁵ http://www.globus.org/api/c-globus-4.0/gaa_plugin/html/index.html#_top

⁶ http://www.globus.org/api/c-globus-4.0/gaa_plugin/html/main.html#_top

⁷ http://www.globus.org/api/c-globus-4.0/globus_authz/html/index.html#_top

⁸ http://www.globus.org/api/c-globus-4.0/globus_authz/html/main.html

⁹ http://www.globus.org/api/c-globus-4.0/globus_authz_callout_error/html/index.html#_top

¹⁰ http://www.globus.org/api/c-globus-4.0/globus_authz_callout_error/html/main.html

¹¹ http://www.globus.org/api/c-globus-4.0/globus_gridmap_callout_error/html/index.html#_top

¹² http://www.globus.org/api/c-globus-4.0/globus_gridmap_callout_error/html/main.html

¹³ http://www.globus.org/api/c-globus-4.0/globus_gsi_callback/html/index.html#_top

¹⁴ http://www.globus.org/api/c-globus-4.0/globus_gsi_callback/html/main.html

¹⁵ http://www.globus.org/api/c-globus-4.0/globus_gsi_cert_utils/html/index.html#_top

¹⁶ http://www.globus.org/api/c-globus-4.0/globus_gsi_cert_utils/html/main.html

¹⁷ http://www.globus.org/api/c-globus-4.0/globus_gsi_credential/html/index.html#_top

¹⁸ http://www.globus.org/api/c-globus-4.0/globus_gsi_credential/html/main.html

¹⁹ http://www.globus.org/api/c-globus-4.0/globus_gsi_openssl_error/html/index.html#_top

²⁰ http://www.globus.org/api/c-globus-4.0/globus_gsi_openssl_error/html/main.html

²¹ http://www.globus.org/api/c-globus-4.0/globus_gsi_proxy_core/html/index.html#_top

²² http://www.globus.org/api/c-globus-4.0/globus_gsi_proxy_core/html/main.html

²³ http://www.globus.org/api/c-globus-4.0/globus_gsi_proxy_ssl/html/index.html#_top

²⁴ http://www.globus.org/api/c-globus-4.0/globus_gsi_proxy_ssl/html/main.html

²⁵ http://www.globus.org/api/c-globus-4.0/globus_gsi_sysconfig/html/index.html#_top

²⁶ http://www.globus.org/api/c-globus-4.0/globus_gsi_sysconfig/html/main.html

²⁷ http://www.globus.org/api/c-globus-4.0/globus_gss_assist/html/index.html#_top

²⁸ http://www.globus.org/api/c-globus-4.0/globus_gss_assist/html/main.html

- [globus_gssapi_gsi](#)²⁹ [[no frames](#)³⁰]
- [globus_openssl_module](#)³¹ [[no frames](#)³²]
- [gssapi_error](#)³³ [[no frames](#)³⁴]

For information on the internationalization API, see the [CCommon Libraries Public Interface](#).

²⁹ http://www.globus.org/api/c-globus-4.0/globus_gssapi_gsi/html/index.html#_top

³⁰ http://www.globus.org/api/c-globus-4.0/globus_gssapi_gsi/html/main.html

³¹ http://www.globus.org/api/c-globus-4.0/globus_openssl_module/html/index.html#_top

³² http://www.globus.org/api/c-globus-4.0/globus_openssl_module/html/main.html

³³ http://www.globus.org/api/c-globus-4.0/gssapi_error/html/index.html#_top

³⁴ http://www.globus.org/api/c-globus-4.0/gssapi_error/html/main.html

Chapter 6. Protocol Specifications

1. GSI Message Specification

The GSSAPI implementation contained in this component produces security tokens that follow an extended version of the SSL/TLS protocol. More information about the protocol can be found [here](#)¹.

¹ ../GSI-message-specification-02.doc

GSI Commands

Name

grid-cert-diagnostics -- Print diagnostic information about certificates and keys

grid-cert-diagnostics [-h] [-p]

Description

The **grid-cert-diagnostics** command displays information about the current user's security environment, including information about security-related environment variables, security directory search path, personal key and certificates, and trusted certificates. It is intended to provide information to help diagnose problems using GSI security.

The full set of command-line options to **grid-cert-diagnostics** consists of:

-h	Display a help message and exit
-p	Display information about the personal certificate and key that is the current user's default credential.

Examples

In this example, we see the default mode of checking the default security environment for the system, without processing the user's key and certificate. Note the user receives a warning about a `cog.properties` and about an expired CA certificate.

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

```
Checking Environment Variables
```

```
=====
```

```
Checking if X509_CERT_DIR is set... no  
Checking if X509_USER_CERT is set... no  
Checking if X509_USER_KEY is set... no  
Checking if X509_USER_PROXY is set... no
```

```
Checking Security Directories
```

```
=====
```

```
Determining trusted cert path... /etc/grid-security/certificates  
Checking for cog.properties... found  
    WARNING: If the cog.properties file contains security properties,  
             Java apps will ignore the security paths described in the GSI  
             documentation
```

```
Checking trusted certificates...
```

```
=====
```

```
Getting trusted certificate list...  
Checking CA file /etc/grid-security/certificates/1c4f4c48.0... ok  
Verifying certificate chain for "/etc/grid-security/certificates/1c3f2ca8.0"... ok  
Checking CA file /etc/grid-security/certificates/9d8788eb.0... ok  
Verifying certificate chain for "/etc/grid-security/certificates/9d8753eb.0"... failed  
    globus_credential: Error verifying credential: Failed to verify credential  
    globus_gsi_callback_module: Could not verify credential  
    globus_gsi_callback_module: The certificate has expired:
```

Credential with subject: /DC=org/DC=example/OU=grid/CN=CA has expired.

In this example, we show a user with a mismatched private key and certificate:

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

```
Checking Environment Variables
```

```
=====
```

```
Checking if X509_CERT_DIR is set... no  
Checking if X509_USER_CERT is set... no  
Checking if X509_USER_KEY is set... no  
Checking if X509_USER_PROXY is set... no
```

```
Checking Security Directories
```

```
=====
```

```
Determining trusted cert path... /etc/grid-security/certificates  
Checking for cog.properties... not found
```

```
Checking Default Credentials
```

```
=====
```

```
Determining certificate and key file names... ok  
Certificate Path: "/home/juser/.globus/usercert.pem"  
Key Path: "/home/juser/.globus/userkey.pem"  
Reading certificate... ok  
Reading private key...  
ok
```

```
Checking Certificate Subject...
```

```
"/O=Grid/OU=Example/OU=User/CN=Joe User"
```

```
Checking cert... ok
```

```
Checking key... ok
```

```
Checking that certificate contains an RSA key... ok
```

```
Checking that private key is an RSA key... ok
```

```
Checking that public and private keys have the same modulus... failed
```

```
Private key modulus: D294849E37F048C3B5ACEEF2CCDF97D88B679C361E29D5CB5  
219C3E948F3E530CFC609489759E1D751F0ACFF0515A614276A0F4C11A57D92D7165B8  
FA64E3140155DE448D45C182F4657DA13EDA288423F5B9D169DFF3822EFD81EB2E6403  
CE3CB4CCF96B65284D92592BB1673A18354DA241B9AFD7F494E54F63A93E15DCAE2  
Public key modulus : C002C7B329B13BFA87BAF214EACE3DC3D490165ACEB791790  
600708C544175D9193C9BAC5AED03B7CB49BB6AE6D29B7E635FAC751E9A6D1CEA98022  
6F1B63002902D6623A319E4682E7BFB0968DCE962CF218AAD95FAAD6A0BA5C42AA9AAF  
7FDD32B37C6E2B2FF0E311310AA55FFB9EAFDF5B995C7D9EEAD8D5D81F3531E0AE5
```

```
Certificate and private key don't match
```

Name

grid-cert-info -- Display certificate information

```
grid-cert-info [-help] [-version]
[-file CERTIFICATE-FILENAME]
[-all] [-subject] [-issuer] [-issuerhash] [-startdate] [-enddate]
```

Description

The **grid-cert-info** displays information from a user's credential, or from any X.509 certificate if the `-file CERTIFICATE-FILENAME` is used. By default, a text representation of the entire certificate is displayed. If more than one display option is present on the command line, the output is generated in the order the options occur on the command line.

The following search order is used to locate the default certificate:

- `$X509_USER_CERT`
- `$HOME/.globus/usercert.pem`
- `$HOME/.globus/usercred.p12`

If the certificate is encoded in pkcs12, **grid-cert-info** will prompt for the password used to protect the `.p12` file.

The full set of command-line options to **grid-cert-info** is:

<code>-help</code>	Print help information and exit
<code>-version</code>	Print version information and exit
<code>-file CERTIFICATE-FILENAME</code>	Read credential from <code>CERTIFICATE-FILENAME</code> instead of the default location. The file must have a <code>.pem</code> or <code>.p12</code> extension.
<code>-all</code>	Print all information from the certificate. This is the default unless any of the following options are given.
<code>-subject</code>	Print the subject name of the certificate.
<code>-issuer</code>	Print the subject name of the issuer of the certificate. This is the subject name of the <i>Certificate Authority</i> which signed the certificate.
<code>-issuerhash</code>	Print the hash of the name of the issuer of the certificate. This is the hash of the Certificate Authority which signed the certificate.
<code>-startdate</code>	Print the date and time from which the certificate is valid
<code>-enddate</code>	Print the date and time when the certificate expires.

Examples

Print out the date range when a certificate is valid:

```
% grid-cert-info -startdate -enddate
```

```
Oct 29 13:09:42 2007 GMT
Oct 28 13:09:42 2008 GMT
```

Note that in this example, the start date is printed first, based on the order of the command-line options.

Limitations

The `-issuerhash` fails with some versions of OpenSSL.

Name

grid-cert-request -- Create a certificate request

```
grid-cert-request [-help] [-version] [-verbose] [-force]
[-commonname NAME] [-service SERVICE] [-host FQDN] [-interactive]
[-dir DIRECTORY] [-prefix PREFIX] [-ca [HASH]] [-nopw]
```

Description

grid-cert-request generates a public/private key pair and an X.509 certificate request containing the public key and a subject name. By default, it generates a request for a user certificate for the invoking user. **grid-cert-request** can also be used to create host or service certificates based on command-line options. At least one *Certificate Authority* must be configured to use with the Globus Toolkit in order for this command to succeed.

Complete set of options to **grid-cert-request** is:

-help	Print help information and exit
-version	Print version information and exit
-verbose	Don't clear screen after running OpenSSL
-force	Overwrite an existing certificate request if present.
-commonname <i>NAME</i>	Construct a subject name with <i>NAME</i> as the final name component. By default, the subject name is inferred from the output of the finger program. If that fails, grid-cert-request will prompt for a name.
-service <i>SERVICE</i>	Construct a subject name with the common name constructed from the <i>SERVICE</i> name and the hostname joined by the / character. The <i>-service</i> requires that the <i>-host</i> option also be used. The private key created for a service certificate request is not encrypted.
-host <i>FQDN</i>	Construct a subject name with <i>FQDN</i> as the name of the host. This must be a fully-qualified name in dotted string notation (e.g. <i>grid.example.org</i>). If no service is specified by the <i>-service</i> option, the subject name will be <i>host/FQDN</i> . The private key created for a host certificate request is not encrypted. By default the host certificate request and key are created in <i>/etc/grid-security</i> .
-interactive	Interactively prompt for the components of the certificate subject name.
-dir <i>DIRECTORY</i>	Write the certificate request and key to <i>DIRECTORY</i> , creating it if the directory does not exist. By default, the certificate request and key are placed in <i>\$HOME/.globus</i>
-prefix <i>PREFIX</i>	Prepend the string <i>PREFIX</i> to the certificate, key, and request filenames. The default prefix is <i>user</i> for user certificates and <i>host</i> for host certificates.
-ca <i>HASH</i>	Choose a non-default Certificate Authority configuration to construct the certificate request. If <i>HASH</i> is present on the command line, then grid-cert-request will use that certificate authority's configuration. Otherwise, it will prompt the user for a CA to choose from the list of configured CAs.
-nopw	Create a private key without a password. This may be a security risk if the file permissions of the private key are not carefully maintained.

Examples

Request a user certificate:

```
% grid-cert-request
```

```
A certificate request and private key is being created.
You will be asked to enter a PEM pass phrase.
This pass phrase is akin to your account password,
and is used to protect your key file.
If you forget your pass phrase, you will need to
obtain a new certificate.
```

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

A private key and a certificate request has been generated with the subject:

```
/O=Grid/OU=Example/OU=User/CN=Joe User
```

If the CN=Joe User is not appropriate, rerun this script with the `-force -cn "Common Name"` options.

```
Your private key is stored in /home/juser/.globus/userkey.pem
Your request is stored in /home/juser/.globus/usercert_request.pem
```

Please e-mail the request to the Globus Certificate Service `ca@grid.example.org`
You may use a command similar to the following:

```
cat /home/juser/.globus/usercert_request.pem | mail ca@grid.example.org
```

Only use the above if this machine can send AND receive e-mail. if not, please mail using some other method.

Your certificate will be mailed to you within two working days.
If you receive no response, contact Globus Certificate Service at `ca@grid.example.org`

Request a host certificate, putting the request and key files in the `$HOME/.globus/host` directory.

```
% grid-cert-request -host grid.example.org -dir $HOME/.globus/host
```

A private host key and a certificate request has been generated with the subject:

```
/O=Grid/OU=Example/OU=User/CN=host/grid.example.org
```

The private key is stored in /tmp/examplegrid/hostkey.pem
The request is stored in /tmp/examplegrid/hostcert_request.pem

Please e-mail the request to the Globus Certificate Service ca@grid.example.org
You may use a command similar to the following:

```
cat /tmp/examplegrid/hostcert_request.pem | mail ca@grid.example.org
```

Only use the above if this machine can send AND receive e-mail. if not, please mail using some other method.

Your certificate will be mailed to you within two working days.
If you receive no response, contact Globus Certificate Service at ca@grid.example.org

Limitations

Only supports PEM-encoded keys, certificates and certificate requests.

Name

grid-default-ca -- Set the default CA to use for certificate requests

```
grid-default-ca [-help] [-list] [-ca CA-HASH] [-dir SECURITY-DIRECTORY]
```

Description

The **grid-default-ca** program sets the default CA used by **grid-cert-request**. Based on the default CA choice, **grid-cert-request** will create a certificate request that matches the CA's naming policies.

If the `-ca` option is not provided on the command-line, **grid-default-ca** will display a list of available Certificate Authorities and prompt the user to choose one.

The full set of command-line options to **grid-default-ca** are:

<code>-help</code>	Display a help message and exit
<code>-list</code>	List the available CAs but do not alter the default
<code>-ca CA-HASH</code>	Select the default CA whose subject name hash matches <code>CA-HASH</code> .
<code>-dir SECURITY-DIRECTORY</code>	Search <code>SECURITY-DIRECTORY</code> for additional CA certificates.

Examples

Show what certificate authorities are in the trusted cert directory:

```
% grid-default-ca -list
```

The available CA configurations installed on this host are:

Directory: /etc/grid-security/certificates

- 1) 1c3f2ca8 - /DC=org/DC=DOEGrids/OU=Certificate Authorities/CN=DOEGrids CA 1
- 2) 3d8e6ce8 - /O=Grid/CN=Example CA
- 3) 6349a761 - /O=DOE Science Grid/OU=Certificate Authorities/CN=Certificate Manager
- 4) b38b4d8c - /C=US/O=Globus Alliance/CN=Globus Certificate Service

The default CA is: /C=US/O=Globus Alliance/CN=Globus Certificate Service
Location: /etc/grid-security/certificates/b38b4d8c.0

Change the default CA to be *DOEGrids CA 1*:

```
% grid-default-ca
```

The available CA configurations installed on this host are:

Directory: /etc/grid-security/certificates

- 1) 1c3f2ca8 - /DC=org/DC=DOEGrids/OU=Certificate Authorities/CN=DOEGrids CA 1
- 2) 3d8e6ce8 - /O=Grid/CN=Example CA
- 3) 6349a761 - /O=DOE Science Grid/OU=Certificate Authorities/CN=Certificate Manager
- 4) b38b4d8c - /C=US/O=Globus Alliance/CN=Globus Certificate Service

The default CA is: /C=US/O=Globus Alliance/CN=Globus Certificate Service
Location: /etc/grid-security/certificates/b38b4d8c.0

Enter the index number of the CA to set as the default [q to quit]: **1**

setting the default CA to: /DC=org/DC=DOEGrids/OU=Certificate Authorities/CN=DOEGrids CA 1

linking /etc/grid-security/certificates/grid-security.conf.1c3f2ca8 to
/etc/grid-security/grid-security.conf

linking /etc/grid-security/certificates/globus-host-ssl.conf.1c3f2ca8 to
/etc/grid-security/globus-host-ssl.conf

linking /etc/grid-security/certificates/globus-user-ssl.conf.1c3f2ca8 to
/etc/grid-security/globus-user-ssl.conf

...done.

Limitations

Displays all CAs in the output, even those where the globus-user-ssl.conf and globus-host-ssl.conf files are not installed in the trusted certificate directory. If one of those is chosen, **grid-default-ca** displays an error and exits.

Name

grid-change-pass-phrase -- Change the pass phrase on a private key

grid-change-pass-phrase

Tool description

grid-change-pass-phrase allows one to change the passphrase that protects the private key.

Command syntax

```
grid-change-pass-phrase [-help] [-version] [-file private_key_file]
```

Changes the passphrase that protects the private key. Note that this command will work even if the original key is not password protected. If the `-file` argument is not given, the default location of the file containing the private key is assumed:

- The location pointed to by `X509_USER_KEY`
- If `X509_USER_KEY` not set, `$HOME/.globus/userkey.pem`

Options

Table 1. Command line options

help, -usage	Displays usage.
-version	Displays version.
-file location	Changes the passphrase on the key stored in the file at the non-standard location 'location'.

Limitations

Nothing applicable

Name

grid-proxy-init -- Generate a new *proxy certificate*

grid-proxy-init

Tool description

grid-proxy-init generates X.509 proxy certificates.

By default, this command generates [RFC 3820](#)¹ Proxy Certificates.

There are also options available for generating other types of proxy certificates, including limited, independent and legacy. For more information about proxy certificate types and their compatibility in GT, see <http://dev.globus.org/wiki/Security/ProxyCertTypes>.

Command syntax

```
grid-proxy-init [-help][-pwstdin][-limited][-valid H:M] ...
```

¹ <http://www.ietf.org/rfc/rfc3820.txt>

Options

Table 2. Command line options

-help, -usage	Displays usage.
-version	Displays version.
-debug	Enables extra debug output.
-q	Quiet mode, minimal output.
-verify	Verifies the certificate to make the proxy for.
-pwstdin	Allows passphrase from stdin.
-limited	Creates a limited globus proxy.
-independent	Creates an independent globus proxy.
-draft	Creates a draft (GSI-3) proxy.
-old	Creates a legacy globus proxy.
-valid <h:m>	Proxy is valid for <i>h</i> hours and <i>m</i> minutes (default:12:00).
-hours <hours>	Deprecated support of hours option.
-bits <bits>	Number of bits in key {512 1024 2048 4096}.
-policy <policyfile>	File containing the policy to store in the ProxyCertInfo extension.
-pl <oid>, -policy-language <oid>	OID string for the policy language used in the policy file.
-path-length <l>	Allows a chain of at most 1 proxies to be generated from this one.
-cert <certfile>	Non-standard location of user certificate.
-key <keyfile>	Non-standard location of user key.
-certdir <certdir>	Non-standard location of trusted cert directory.
-out <proxyfile>	Non-standard location of new proxy cert.

Creating a Proxy Certificate

Proxies are certificates signed by the user, or by another proxy, that do not require a password to submit a job. They are intended for short-term use, when the user is submitting many jobs and cannot be troubled to repeat his password for every job.

The subject of a proxy certificate is the same as the subject of the certificate that signed it, with /CN=proxy added to the name. The gatekeeper will accept any job requests submitted by the user, as well as any proxies he has created.

Proxies provide a convenient alternative to constantly entering passwords, but are also less secure than the user's normal security credential. Therefore, they should always be user-readable only, and should be deleted after they are no longer needed (or after they expire).

To create a proxy with the default expiration (12 hours), run the grid-proxy-init program. For example:

```
% grid-proxy-init
```

The grid-proxy-init program can also take arguments to specify the expiration and proxy key length. For example:

```
% grid-proxy-init -hours 8 -bits 512
```

Limitations

Nothing applicable

Name

grid-proxy-destroy -- Destroy the current proxy certificate (previously created with grid-proxy-init)

grid-proxy-destroy

Tool description

grid-proxy-destroy removes X.509 proxy certificates.

Command syntax

```
grid-proxy-destroy [-help][--dryrun][-default][-all][--] [file1...]
```

Options

Table 3. Command line options

-help, -usage	Displays usage.
-version	Displays version.
-debug	Displays debugging information.
-dryrun	Prints what files would have been destroyed.
-default	Destroys file at default proxy location.
-all	Destroys any user (default) and delegated proxies that are found.
--	Ends processing of options.
file1 file2 ...	Destroys the files listed.

Limitations

Nothing applicable

Name

grid-proxy-info -- Display information obtained from a proxy certificate

grid-proxy-info

Tool description

grid-proxy-info extracts information from X.509 proxy certificates.

Command syntax

```
grid-proxy-info [-help][-f proxyfile][-subject][...][-e [-h H][-b B]]
```

Options

Table 4. Command line options

-help, -usage	Displays usage.
-version	Displays version.
-debug	Displays debugging output.
-file <proxyfile> (-f)	Non-standard location of proxy.
[printoptions]	See Table 5, “Print options”.
-exists [options] (-e)	Determine whether a valid proxy exists. <code>options</code> may contain any <u>validation options</u> described below. If a proxy exists, and meets any criteria defined by the validity options, then grid-proxy-info will terminate with the exit code 0. Otherwise, grid-proxy-info will terminate with the exit code 1. If no validity options are specified, the program will terminate with 0 if a currently-valid proxy file exists.

Table 5. Print options

-subject (-s)	Distinguished name (DN) of the subject.
-issuer (-i)	DN of the issuer (certificate signer).
-identity	DN of the identity represented by the proxy.
-type	Type of proxy (full or limited).
-timeleft	Time (in seconds) until proxy expires.
-strength	Key size (in bits).
-all	All above options in a human readable format.
-text	All of the certificate.
-path	Pathname of the proxy file.

Table 6. Validity options

-valid H:M (-v)	Time requirement for the proxy to be valid.
-hours H (-h)	Time requirement for the proxy to be valid (deprecated, use -valid instead).
-bits B (-b)	Strength requirement for the proxy to be valid.

Limitations

Nothing applicable

Name

grid-mapfile-add-entry -- Add an entry to a *grid map file*

grid-mapfile-add-entry

Tool description

grid-mapfile-add-entry adds entries to grid map files.

Command syntax

```
grid-mapfile-add-entry -dn DN -ln LN [-help] [-d] [-f mapfile FILE]
```

Options:

Table 7. Command line options

-help, -usage	Displays help.
-version	Displays version.
-dn DN	Distinguished Name (DN) to add. Remember to quote the DN if it contains spaces.
-ln LN1 [LN2...]	Local login name(s) to which the DN is mapped.
-dryrun, -d	Shows what would be done but will not add the entry.
-mapfile FILE, -f FILE	Path of the grid map file to be used.

Limitations

Nothing applicable.

Name

grid-mapfile-check-consistency -- Check the internal consistency of a grid map file

grid-mapfile-check-consistency

Tool description

grid-mapfile-check-consistency checks that the given grid mapfile conforms to the expected format as well as checking for common subject name problems.

Command syntax

grid-mapfile-check-consistency [-help] [-mapfile FILE]

Options:

Table 8. Command line options

-help, -usage	Displays help.
-version	Displays version.
-mapfile FILE, -f FILE	Path of the grid map file to be used.

Limitations

Nothing applicable

Name

grid-mapfile-delete-entry -- Delete an entry from a grid map file

grid-mapfile-delete-entry

Tool description

grid-mapfile-delete entry deletes a grid map file entry from the given file.

Command syntax

grid-mapfile-delete-entry [-help] [-dn <DN>] [-ln <local name>] [-d] [-f file]

Options:

Table 9. Command line options

-help, -usage	Displays help.
-version	Displays version.
-dn <DN>	Distinguished Name (DN) to delete.
-ln <local name>	Local Login Name (LN) to delete.
-dryrun, -d	Shows what would be done but will not delete the entry.
-mapfile file, -f file	Path of the grid map file to be used.

Limitations

Nothing applicable.

Chapter 7. 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 7.1. 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 7.2. 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 7.3. 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 7.4. 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 7.5. Authorization Configuration File Locations

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

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 7.6. 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 8. Environment variable interface

1. Environmental Variables for GSI C

1.1. Credentials

Credentials are looked for in the following order:

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

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

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

For *service credentials*:

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

For *host credentials*:

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

For *user credentials*:

1. If X509_USER_CERT and X509_USER_KEY exist and contain a valid certificate and key, those files are used.
2. Otherwise, if the files `usercert.pem` and `userkey.pem` exist in the user's `.globus` directory, those files are used.
3. Otherwise, if a PKCS-12 file called `usercred.p12` exists in the user's `.globus` directory, the certificate and key are read from that file.

1.2. Gridmap file

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

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

1.3. Trusted CAs directory

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

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

1.4. GSI authorization callout configuration file

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

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

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

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

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

1.6. Grid security directory

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

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

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

1.7. Using TLS

GLOBUS_GSSAPI_FORCE_TLS specifies whether to use TLS by default when establishing a security context. The default behavior if this is not set is to use SSLv3.

1.8. Name Comparisons

GLOBUS_GSSAPI_NAME_COMPATIBILITY specifies what name matching algorithms are supported by GSSAPI for mutual authentication and `gss_compare_name`. This variable may be set to any of the following values:

STRICT_GT2	Strictly backward-compatible with GT 2.0 name matching. X.509 subjectAltName values are ignored. Names with hyphens are treated as wildcarded as described in the security considerations documentation. Name matching will rely on canonical host name associated with connection IP addresses.
STRICT_RFC2818	Support RFC 2818 ¹ server identity processing. Hyphen characters are treated as normal part of a host name. DNSName and IPAddress subjectAltName extensions are matched against the host and port passed to GSSAPI. If subjectAltName is present, X.509 SubjectName is ignored.

¹ <http://www.ietf.org/rfc/rfc2818.txt>

HYBRID	Support a hybrid of the two previous name matching algorithms, liberally matching both hyphen wildcards, canonical names associated with IP addresses, and subjectAltName extensions.
--------	---

If this variable is not set, the HYBRID behavior is used.

Chapter 9. Debugging

For information about system administrator logs, see [Chapter 4, Debugging](#) in the GSI C Admin Guide.

Chapter 10. 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 10.1. 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 10.2. 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.

Chapter 11. Related Documentation

- [RFC 3820](#)¹ Proxy Certificates
- [RFC 2744](#)² GSSAPI: C-bindings
- [RFC 2743](#)³ GSSAPI
- [GSSAPI Extensions](#)⁴
- [RFC 2246](#)⁵ TLS
- [Grid Security Infrastructure Message Specification](#)⁶

¹ <http://www.faqs.org/rfcs/rfc3820.html>

² <http://www.faqs.org/rfcs/rfc2744.html>

³ <http://www.faqs.org/rfcs/rfc2743.html>

⁴ <http://www.ggf.org/documents/GWD-I-E/GFD-E.024.pdf>

⁵ <http://www.faqs.org/rfcs/rfc2246.html>

⁶ <http://www.globus.org/toolkit/docs/3.0/gsi/GSI-message-specification-02.doc>

Glossary

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 .
grid security directory	The directory containing GSI configuration files such as the GSI authorization callout configuration and GAA configuration files. Typically this directory is <code>/etc/grid-security</code> . For more information see this .
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 .
host credentials	The combination of a host certificate and its corresponding private key.

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 `$HOME/.globus/userkey.pem` (for user certificates), `/etc/grid-security/hostkey.pem` (for host certificates) or `/etc/grid-security/<service>/<service>key.pem` (for service certificates).

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

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

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

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

S

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](#).

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

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](#).

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