

# globus gsi credential Reference Manual

## 3.2

Generated by Doxygen 1.3.5

Sat Feb 6 15:29:06 2010

## Contents

<a href="#">1</a>	<a href="#">Globus GSI Credential</a>	<a href="#">1</a>
<a href="#">2</a>	<a href="#">globus gsi credential Module Index</a>	<a href="#">1</a>
<a href="#">3</a>	<a href="#">globus gsi credential Module Documentation</a>	<a href="#">1</a>

## 1 Globus GSI Credential

The Globus GSI Credential library. This library contains functions that provide support for handling X.509 based PKI credentials

- [Activation](#)
- [Credential Handle Management](#)
- [Credential Handle Attributes](#)
- [Credential Operations](#)
- [Credential Constants](#)

## 2 globus gsi credential Module Index

### 2.1 globus gsi credential Modules

Here is a list of all modules:

<a href="#">Credential Constants</a>	<a href="#">1</a>
<a href="#">Activation</a>	<a href="#">3</a>
<a href="#">Credential Handle Management</a>	<a href="#">4</a>
<a href="#">Credential Handle Attributes</a>	<a href="#">12</a>
<a href="#">Credential Operations</a>	<a href="#">15</a>

## 3 globus gsi credential Module Documentation

### 3.1 Credential Constants

Enumerations

- [enumglobus\\_gsi\\_cred\\_error{t](#)  
[GLOBUS\\_GSI\\_CRED\\_ERROR\\_SUCCESS](#),  
[GLOBUS\\_GSI\\_CRED\\_ERROR\\_READING\\_PROXY\\_CRED](#),  
[GLOBUS\\_GSI\\_CRED\\_ERROR\\_INVALID\\_CRED](#)

```

GLOBUS_GSI_CRED_ERROR_READING_HOST_CRED,
GLOBUS_GSI_CRED_ERROR_READING_SERVICE_CRED,
GLOBUS_GSI_CRED_ERROR_READING_CRED,
GLOBUS_GSI_CRED_ERROR_WRITING_CRED,
GLOBUS_GSI_CRED_ERROR_WRITING_PROXY_CRED,
GLOBUS_GSI_CRED_ERROR_CHECKING_PROXY,
GLOBUS_GSI_CRED_ERROR_VERIFYING_CRED,
GLOBUS_GSI_CRED_ERROR_WITH_CRED,
GLOBUS_GSI_CRED_ERROR_WITH_CRED_CERT,
GLOBUS_GSI_CRED_ERROR_WITH_CRED_PRIVATE_KEY,
GLOBUS_GSI_CRED_ERROR_WITH_CRED_CERT_CHAIN,
GLOBUS_GSI_CRED_ERROR_ERRNO,
GLOBUS_GSI_CRED_ERROR_SYSTEM_CONFIG,
GLOBUS_GSI_CRED_ERROR_WITH_CRED_HANDLE_ATTRS,
GLOBUS_GSI_CRED_ERROR_WITH_SSL_CTX,
GLOBUS_GSI_CRED_ERROR_WITH_CALLBACK_DATA,
GLOBUS_GSI_CRED_ERROR_CREATING_ERROR_OBJECT,
GLOBUS_GSI_CRED_ERROR_KEY_IS_PASS_PROTECTED,
GLOBUS_GSI_CRED_ERROR_NO_CRED_FOUND,
GLOBUS_GSI_CRED_ERROR_SUBJECT_CM,
GLOBUS_GSI_CRED_ERROR_GETTING_SERVICE_NAME,
GLOBUS_GSI_CRED_ERROR_BAD_PARAMETER,
GLOBUS_GSI_CRED_ERROR_WITH_CRED_CERT_NAME,
GLOBUS_GSI_CRED_ERROR_LAST }
• enum globus_gsi_cred_type_t

```

### 3.1.1 Enumeration Type Documentation

#### 3.1.1.1 enum globus\_gsi\_cred\_error\_t

Credential Error codes.

Enumeration values:

GLOBUS\_GSI\_CRED\_ERROR\_SUCCESS Success - never used.

GLOBUS\_GSI\_CRED\_ERROR\_READING\_PROXY\_CRED Failed to read proxy credential.

GLOBUS\_GSI\_CRED\_ERROR\_READING\_HOST\_CRED Failed to read host credential.

GLOBUS\_GSI\_CRED\_ERROR\_READING\_SERVICE\_CRED Failed to read service credential.

GLOBUS\_GSI\_CRED\_ERROR\_READING\_CRED Failed to read user credential.

GLOBUS\_GSI\_CRED\_ERROR\_WRITING\_CRED Failed to write credential.

GLOBUS\_GSI\_CRED\_ERROR\_WRITING\_PROXY\_CRED Failed to write proxy credential.

GLOBUS\_GSI\_CRED\_ERROR\_CHECKING\_PROXY Error checking for proxy credential.

GLOBUS\_GSI\_CRED\_ERROR\_VERIFYING\_CRED Failed to verify credential.

GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED Invalid credential.

GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED\_CERTInvalid certi cate.  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED\_PRIVATE\_KEYInvalid private key.  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED\_CERT\_CHAINInvalid certi cate chain.  
 GLOBUS\_GSI\_CRED\_ERROR\_ERRNOSystem error.  
 GLOBUS\_GSI\_CRED\_ERROR\_SYSTEM\_CONFIGAGlobus GSI System Con guration call failed.  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED\_HANDLE\_ATTRSInvalid credential handle attributes.  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_SSL\_CTXFaulty SSL context.  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CALLBACK\_DATAFaulty callback data.  
 GLOBUS\_GSI\_CRED\_ERROR\_CREATING\_ERROR\_OBFFailed to aggregate errors.  
 GLOBUS\_GSI\_CRED\_ERROR\_KEY\_IS\_PASS\_PROTECTEDError reading private key - the key is pass-  
 word protected.  
 GLOBUS\_GSI\_CRED\_ERROR\_NO\_CRED\_FOUNDCouldn't nd credential to read.  
 GLOBUS\_GSI\_CRED\_ERROR\_SUBJECT\_CMPCredential subjects do not compare.  
 GLOBUS\_GSI\_CRED\_ERROR\_GETTING\_SERVICE\_NAMEUnable to obtain service name from CN en-  
 try.  
 GLOBUS\_GSI\_CRED\_ERROR\_BAD\_PARAMETERInvalid function parameter.  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED\_CERT\_NAMEFailed to process certi cate subject.  
 GLOBUS\_GSI\_CRED\_ERROR\_LASTEnd marker - never used.

### 3.1.1.2 [enumglobus\\_gsi\\_cred\\_type\\_t](#)

Credential Type.

An enum representing a GSI Credential Type which holds info about the type of a particular credential. The three types of credential can be: GLOBUS\_PROXY, GLOBUS\_USER, or GLOBUS\_HOST.

See also:

[globus\\_gsi\\_cred\\_handle](#)

## 3.2 Activation

Globus GSI Credential uses standard Globus module activation and deactivation.

De nes

- #de ne [GLOBUS\\_GSI\\_CREDENTIAL\\_MODULE](#)

### 3.2.1 Detailed Description

Globus GSI Credential uses standard Globus module activation and deactivation.

Before any Globus GSI Credential functions are called, the following function must be called:

```
globus_module_activate(GLOBUS_GSI_CREDENTIAL_MODULE)
```

This function returns GLOBUS\_SUCCESS if Globus GSI Credential was successfully initialized, and you are therefore allowed to subsequently call Globus GSI Credential functions. Otherwise, an error code is returned, and Globus GSI Credential functions should not be subsequently called. This function may be called multiple times.

To deactivate Globus GSI Credential, the following function must be called:

```
globus_module_deactivate(GLOBUS_GSI_CREDENTIAL_MODULE)
```

This function should be called once for each time Globus GSI Credential was activated.

### 3.2.2 De ne Documentation

#### 3.2.2.1 #de ne GLOBUS\_GSI\_CREDENTIAL\_MODULE

Module descriptor.

## 3.3 Credential Handle Management

Create/Destroy/Modify a GSI Credential Handle.

Initializing and Destroying a Handle

- `globus_result_t globus_gsi_cred_handle_init(globus_gsi_cred_handle_t handle, globus_gsi_cred_handle_attrs_t handle_attrs)`
- `globus_result_t globus_gsi_cred_handle_destroy(globus_gsi_cred_handle_t handle)`

Copying a Handle

- `globus_result_t globus_gsi_cred_handle_copy(globus_gsi_cred_handle_t source, globus_gsi_cred_handle_t dest)`

Getting the Handle Attributes

- `globus_result_t globus_gsi_cred_get_handle_attr(globus_gsi_cred_handle_t handle, globus_gsi_cred_handle_attr_t attr)`

Getting the Credential Expiration

- `globus_result_t globus_gsi_cred_get_goodtill(globus_gsi_cred_handle_t cred_handle, time_t goodtill)`

Getting the Credential Lifetime

- `globus_result_t globus_gsi_cred_get_lifetime(globus_gsi_cred_handle_t cred_handle, time_t lifetime)`

Getting the Credential Strength

- `globus_result_t globus_gsi_cred_get_key_bits(globus_gsi_cred_handle_t cred_handle, int key_bits)`

## Setting and Getting the Certificate

- globus\_result\_t [globus\\_gsi\\_cred\\_set\\_cert](#)(globus\_gsi\_cred\_handle\_t handle, X509 cert)
- globus\_result\_t [globus\\_gsi\\_cred\\_get\\_cert](#)(globus\_gsi\_cred\_handle\_t handle, X509 cert)

## Setting and Getting the Credential Key

- globus\_result\_t [globus\\_gsi\\_cred\\_set\\_key](#)(globus\_gsi\_cred\_handle\_t handle, EVP\_PKEY key)
- globus\_result\_t [globus\\_gsi\\_cred\\_get\\_key](#)(globus\_gsi\_cred\_handle\_t handle, EVP\_PKEY key)

## Setting and Getting the Certificate Chain

- globus\_result\_t [globus\\_gsi\\_cred\\_set\\_cert\\_chain](#)(globus\_gsi\_cred\_handle\_t handle, STACK\_OF(X509) cert\_chain)
- globus\_result\_t [globus\\_gsi\\_cred\\_get\\_cert\\_chain](#)(globus\_gsi\_cred\_handle\_t handle, STACK\_OF(X509) cert\_chain)

## Get Cred Cert X509 Subject Name object

- globus\_result\_t [globus\\_gsi\\_cred\\_get\\_X509\\_subject\\_name](#)(globus\_gsi\_cred\_handle\_t handle, X509\_NAME subject\_name)

## Get X509 Identity Name

- globus\_result\_t [globus\\_gsi\\_cred\\_get\\_X509\\_identity\\_name](#)(globus\_gsi\_cred\_handle\_t handle, X509\_NAME identity\_name)

## Get Cred Cert Subject Name

- globus\_result\_t [globus\\_gsi\\_cred\\_get\\_subject\\_name](#)(globus\_gsi\_cred\_handle\_t handle, char subject\_name)

## Get Policies from Cert Chain

- globus\_result\_t [globus\\_gsi\\_cred\\_get\\_policies](#)(globus\_gsi\_cred\_handle\_t handle, STACK policies)

## Get Policy Languages from Cert Chain

- globus\_result\_t [globus\\_gsi\\_cred\\_get\\_policy\\_languages](#)(globus\_gsi\_cred\_handle\_t handle, STACK\_OF(ASN1\_OBJECT) policy\_languages)

## Get Cred Cert X509 Issuer Name object

- globus\_result\_t [globus\\_gsi\\_cred\\_get\\_X509\\_issuer\\_name](#)(globus\_gsi\_cred\_handle\_t handle, X509\_NAME issuer\_name)

## Get Issuer Name

- globus\_result\_t [globus\\_gsi\\_cred\\_get\\_issuer\\_name](#)(globus\_gsi\_cred\_handle\_t handle, char issuer\_name)

### Get Identity Name

- `globus_result_t globus_gsi_cred_get_identity_name(globus_gsi_cred_handle_t handle, char identity_name)`

### Credential validation functions

- `globus_result_t globus_gsi_cred_verify_cert_chain(globus_gsi_cred_handle_t cred_handle, globus_gsi_callback_data_t callback_data)`
- `globus_result_t globus_gsi_cred_verify(globus_gsi_cred_handle_t handle)`

### Typedefs

- `typedef globus_l_gsi_cred_handle_t globus_gsi_cred_handle_t`

#### 3.3.1 Detailed Description

Create/Destroy/Modify a GSI Credential Handle.

Within the Globus GSI Credential Library, all credential operations require a handle parameter. Currently only one operation may be in progress at once per credential handle.

This section describes operations to create, modify and destroy GSI Credential handles.

#### 3.3.2 Typedef Documentation

##### 3.3.2.1 typedef struct globus\_l\_gsi\_cred\_handle\_t globus\_gsi\_cred\_handle\_t

GSI Credential Handle.

A GSI Credential handle keeps track of state relating to a credential. Handles can have immutable attributes associated with them. All credential operations take a credential handle pointer as a parameter.

See also:

`globus_gsi_cred_handle_init()`, `globus_gsi_cred_handle_destroy()`, `globus_gsi_cred_handle_attrs_t`

#### 3.3.3 Function Documentation

##### 3.3.3.1 `globus_result_t globus_gsi_cred_handle_init(globus_gsi_cred_handle_t handle, globus_gsi_cred_handle_attrs_t handle_attrs)`

Initializes a credential handle to be used credential handling functions. Takes a set of handle attributes that are immutable to the handle. The handle attributes are only pointed to by the handle, so the lifetime of the attributes needs to be as long as that of the handle.

Parameters:

- handle The handle to be initialized
- handle\_attrs The immutable attributes of the handle

Returns:

GLOBUS\_SUCCESS or an error captured in a `globus_result_t`

3.3.3.2 `globus_result_t globus_gsi_cred_handle_destroy(globus\_gsi\_cred\_handle\_t handle)`

Destroys the credential handle.

Parameters:

handle The credential handle to be destroyed

Returns:

GLOBUS\_SUCCESS

3.3.3.3 `globus_result_t globus_gsi_cred_handle_copy(globus\_gsi\_cred\_handle\_t source, globus\_gsi\_cred\_handle\_t dest)`

Copies a credential handle.

Parameters:

source The handle to be copied

dest The destination of the copy

Returns:

GLOBUS\_SUCCESS or an error captured in a `globus_result_t`

3.3.3.4 `globus_result_t globus_gsi_cred_get_handle_attrs(globus\_gsi\_cred\_handle\_t handle, globus\_gsi\_cred\_handle\_attrs\_t attrs)`

This function retrieves a copy of the credential handle attributes

Parameters:

handle The credential handle to retrieve the attributes from

attrs Contains the credential attributes on return

Returns:

GLOBUS\_SUCCESS or an error captured in a `globus_result_t`

3.3.3.5 `globus_result_t globus_gsi_cred_get_goodtill(globus\_gsi\_cred\_handle\_t cred_handle, time_t goodtill)`

This function retrieves the expiration time of the credential contained in the handle

Parameters:

cred\_handle The credential handle to retrieve the expiration time from

goodtill Contains the expiration time on return

Returns:

GLOBUS\_SUCCESS or an error captured in a `globus_result_t`



3.3.3.6 `globus_result_t globus_gsi_cred_get_lifetime(globus_gsi_cred_handle_t cred_handle, time_t * lifetime)`

This function retrieves the lifetime of the credential contained in a handle

Parameters:

`cred_handle` The credential handle to retrieve the lifetime from  
`lifetime` Contains the lifetime on return

Returns:

GLOBUS\_SUCCESS or an error captured in a `globus_result_t`

3.3.3.7 `globus_result_t globus_gsi_cred_get_key_bits(globus_gsi_cred_handle_t cred_handle, int * key_bits)`

This function retrieves the key strength of the credential contained in a handle

Parameters:

`cred_handle` The credential handle to retrieve the strength from  
`key_bits` Contains the number of bits in the key on return

Returns:

GLOBUS\_SUCCESS or an error captured in a `globus_result_t`

3.3.3.8 `globus_result_t globus_gsi_cred_set_cert(globus_gsi_cred_handle_t handle, X509 * cert)`

Set the Credential's Certificate. The X509 cert that is passed in should be a valid X509 certificate object

Parameters:

`handle` The credential handle to set the certificate on  
`cert` The X509 cert to set in the cred handle. The cert passed in can be NULL which will set the cert in the handle to NULL, freeing the current cert in the handle.

Returns:

GLOBUS\_SUCCESS or an error object id if an error

3.3.3.9 `globus_result_t globus_gsi_cred_get_cert(globus_gsi_cred_handle_t handle, X509 * cert)`

Get the certificate of a credential

Parameters:

`handle` The credential handle to get the certificate from  
`cert` The resulting X509 certificate, a duplicate of the certificate in the credential handle. This variable should be freed when the user is finished with it using the function `X509_free`.

Returns:

GLOBUS\_SUCCESS if no error, otherwise an error object id is returned

3.3.3.10 `globus_result_t globus_gsi_cred_set_key(globus_gsi_cred_handle_t handle, EVP_PKEY key)`

Set the private key of the credential handle

Parameters:

`handle` The handle on which to set the key.

`key` The private key to set the handle's key to. This value can be NULL, in which case the current handle's key is freed.

3.3.3.11 `globus_result_t globus_gsi_cred_get_key(globus_gsi_cred_handle_t handle, EVP_PKEY key)`

Get the credential handle's private key.

Parameters:

`handle` The credential handle containing the private key to get

`key` The private key which after this function returns is set to a duplicate of the private key of the credential handle. This variable needs to be freed by the user when it is no longer used via the function `EVP_PKEY_free`.

Returns:

GLOBUS\_SUCCESS or an error object identifier

3.3.3.12 `globus_result_t globus_gsi_cred_set_cert_chain(globus_gsi_cred_handle_t handle, STACK_OF(X509) cert_chain)`

Set the certificate chain of the credential handle

Parameters:

`handle` The handle containing the certificate chain field to set

`cert_chain` The certificate chain to set the handle's certificate chain to

Returns:

GLOBUS\_SUCCESS if no error, otherwise an error object id is returned

3.3.3.13 `globus_result_t globus_gsi_cred_get_cert_chain(globus_gsi_cred_handle_t handle, STACK_OF(X509) cert_chain)`

Get the certificate chain of the credential handle.

Parameters:

`handle` The credential handle containing the certificate chain to get

`cert_chain` The certificate chain to set as a duplicate of the cert chain in the credential handle. This variable (or the variable it points to) needs to be freed when the user is finished with it using `sk_X509_free`.

Returns:

GLOBUS\_SUCCESS if no error, otherwise an error object id is returned

3.3.3.14 `globus_result_t globus_gsi_cred_get_X509_subject_name(globus\_gsi\_cred\_handle\_t handle, X509_NAME subject\_name)`

Get the credential handle's certificate subject name

Parameters:

`handle` The credential handle containing the certificate to get the subject name of

`subject_name` The subject name as an X509\_NAME object. This should be freed using X509\_NAME\_free when the user is finished with it

Returns:

GLOBUS\_SUCCESS if no error, a error object id otherwise

3.3.3.15 `globus_result_t globus_gsi_cred_get_X509_identity_name(globus\_gsi\_cred\_handle\_t handle, X509_NAME identity\_name)`

Get the identity's X509 subject name from the credential handle

Parameters:

`handle` The credential handle containing the certificate to get the identity from

`identity_name` The identity certificate's X509 subject name

Returns:

GLOBUS\_SUCCESS if no error, otherwise an error object identifier is returned

3.3.3.16 `globus_result_t globus_gsi_cred_get_subject_name(globus\_gsi\_cred\_handle\_t handle, char subject\_name)`

Get the credential handle's certificate subject name

Parameters:

`handle` The credential handle containing the certificate to get the subject name of

`subject_name` The subject name as a string. This should be freed using free() when the user is finished with it

Returns:

GLOBUS\_SUCCESS if no error, a error object id otherwise

3.3.3.17 `globus_result_t globus_gsi_cred_get_policies(globus\_gsi\_cred\_handle\_t handle, STACK policies)`

Get the Policies from the Cert Chain in the handle. The policies will be null-terminated as they are added to the handle. If a policy for a cert in the chain doesn't exist, the string in the stack will be set to the static string GLOBUS\_NULL\_POLICIES

Parameters:

`handle` the handle to get the cert chain containing the policies

`policies` the stack of policies retrieved from the handle's cert chain

Returns:

GLOBUS\_SUCCESS or an error object if an error occurred

3.3.3.18 `globus_result_t globus_gsi_cred_get_policy_languages(globus_gsi_cred_handle_t handle, STACK_OF(ASN1_OBJECT) *policy_languages)`

Get the policy languages from the cert chain in the handle.

Parameters:

handle The handle to get the cert chain containing the policies

policy\_languages the stack of policies retrieved from the handle's cert chain

Returns:

GLOBUS\_SUCCESS or an error object if an error occurred

3.3.3.19 `globus_result_t globus_gsi_cred_get_X509_issuer_name(globus_gsi_cred_handle_t handle, X509_NAME *issuer_name)`

Get the credential handle's certificate issuer name

Parameters:

handle The credential handle containing the certificate to get the issuer name of

issuer\_name The issuer name as an X509\_NAME object. This should be freed using X509\_NAME\_free when the user is finished with it

Returns:

GLOBUS\_SUCCESS if no error, a error object id otherwise

3.3.3.20 `globus_result_t globus_gsi_cred_get_issuer_name(globus_gsi_cred_handle_t handle, char issuer_name[])`

Get the issuer's subject name from the credential handle

Parameters:

handle The credential handle containing the certificate to get the issuer of

issuer\_name The issuer certificate's subject name

Returns:

GLOBUS\_SUCCESS if no error, otherwise an error object identifier is returned

3.3.3.21 `globus_result_t globus_gsi_cred_get_identity_name(globus_gsi_cred_handle_t handle, char identity_name[])`

Get the identity's subject name from the credential handle

Parameters:

handle The credential handle containing the certificate to get the identity of

identity\_name The identity certificate's subject name

Returns:

GLOBUS\_SUCCESS if no error, otherwise an error object identifier is returned

3.3.3.22 `globus_result_t globus_gsi_cred_verify_cert_chain(globus_gsi_cred_handle_t cred_handle, globus_gsi_callback_data_t callback_data)`

This function performs path validation on the certificate chain contained in the credential handle.

Parameters:

- `cred_handle` The credential handle containing the certificate chain to be validated
- `callback_data` A initialized callback data structure

Returns:

- GLOBUS\_SUCCESS if no error, otherwise an error object identifier is returned

3.3.3.23 `globus_result_t globus_gsi_cred_verify(globus_gsi_cred_handle_t handle)`

This function ensures that the certificate and private key in the credential handle match.

Parameters:

- `handle` The credential handle containing the certificate and key to be validated

Returns:

- GLOBUS\_SUCCESS if no error, otherwise an error object identifier is returned

## 3.4 Credential Handle Attributes

Create/Destroy/Modify GSI Credential Handle Attributes.

Credential Handle Attributes Initialization and Destruction

- `globus_result_t globus_gsi_cred_handle_attrs_init(globus_gsi_cred_handle_attrs_t handle_attrs)`
- `globus_result_t globus_gsi_cred_handle_attrs_destroy(globus_gsi_cred_handle_attrs_t handle_attrs)`

Copy Credential Handle Attributes

- `globus_result_t globus_gsi_cred_handle_attrs_copy(globus_gsi_cred_handle_attrs_t source, globus_gsi_cred_handle_attrs_t dest)`

Setting and Getting the CA Cert Dir

- `globus_result_t globus_gsi_cred_handle_attrs_set_ca_cert(globus_gsi_cred_handle_attrs_t handle_attrs, char ca_cert_dir)`
- `globus_result_t globus_gsi_cred_handle_attrs_get_ca_cert(globus_gsi_cred_handle_attrs_t handle_attrs, char ca_cert_dir)`

Setting and Getting the Search Order

- `globus_result_t globus_gsi_cred_handle_attrs_set_search_order(globus_gsi_cred_handle_attrs_t handle_attrs, globus_gsi_cred_type_t search_order[])`
- `globus_result_t globus_gsi_cred_handle_attrs_get_search_order(globus_gsi_cred_handle_attrs_t handle_attrs, globus_gsi_cred_type_t search_order)`

## Typedefs

- typedef globus\_l\_gsi\_cred\_handle\_attrs\_t [globus\\_gsi\\_cred\\_handle\\_attrs\\_t](#)

### 3.4.1 Detailed Description

Create/Destroy/Modify GSI Credential Handle Attributes.

Within the Globus GSI Credential Library, all credential handles contain a attribute structure, which in turn contains handle instance independent attributes.

This section describes operations to create, modify and destroy GSI Credential handle attributes.

### 3.4.2 Typedef Documentation

#### 3.4.2.1 typedef struct globus\_l\_gsi\_cred\_handle\_attrs\_t [globus\\_gsi\\_cred\\_handle\\_attrs\\_t](#)

Credential Handle Attributes.

Credential handle attributes provide a set of immutable parameters for a credential handle

See also:

[globus\\_gsi\\_cred\\_handle\\_init](#)

### 3.4.3 Function Documentation

#### 3.4.3.1 globus\_result\_t globus\_gsi\_cred\_handle\_attrs\_init([globus\\_gsi\\_cred\\_handle\\_attrs\\_t](#) handle\_attrs)

Initializes the immutable Credential Handle Attributes The handle attributes are initialized as follows:

- The search order is set to SERVICE, HOST, PROXY, USER
- All other attributes are set to 0/NULL

Parameters:

handle\_attrs the attributes to be initialized

Returns:

GLOBUS\_SUCCESS if initialization was successful, otherwise an error is returned

#### 3.4.3.2 globus\_result\_t globus\_gsi\_cred\_handle\_attrs\_destroy([globus\\_gsi\\_cred\\_handle\\_attrs\\_t](#) handle\_attrs)

Destroy the Credential Handle Attributes. This function does some cleanup and deallocation of the handle attributes.

Parameters:

handle\_attrs The handle attributes to destroy

Returns:

GLOBUS\_SUCCESS

3.4.3.3 `globus_result_t globus_gsi_cred_handle_attrs_copy(globus_gsi_cred_handle_attrs_t source, globus_gsi_cred_handle_attrs_t dest)`

Copy the Credential Handle Attributes.

Parameters:

source The handle attribute to be copied

dest The copy

Returns:

GLOBUS\_SUCCESS unless there was an error, in which case an error object is returned.

3.4.3.4 `globus_result_t globus_gsi_cred_handle_attrs_set_ca_cert_dir(globus_gsi_cred_handle_attrs_t handle_attrs, char *ca_cert_dir)`

Set the Trusted CA Certificate Directory Location

Parameters:

handle\_attrs the credential handle attributes to set

ca\_cert\_dir the trusted ca certificates directory

Returns:

GLOBUS\_SUCCESS if no errors occurred. In case of a null handle\_attrs, an error object id is returned

3.4.3.5 `globus_result_t globus_gsi_cred_handle_attrs_get_ca_cert_dir(globus_gsi_cred_handle_attrs_t handle_attrs, char **ca_cert_dir)`

Get the trusted ca cert directory.

Parameters:

handle\_attrs the credential handle attributes to get the trusted ca cert directory from

ca\_cert\_dir the trusted ca certificates directory

Returns:

GLOBUS\_SUCCESS if no errors occurred. In case of a null handle\_attrs or pointer to ca\_cert\_dir, an error object id is returned

3.4.3.6 `globus_result_t globus_gsi_cred_handle_attrs_set_search_order(globus_gsi_cred_handle_attrs_t handle_attrs, globus_gsi_cred_type_search_order_t search_order)`

Set the search order for finding a user certificate. The default value is {SERVICE, HOST, PROXY, USER}

Parameters:

handle\_attrs The handle attributes to set the search order of

search\_order The search order. Should be a three element array containing in some order PROXY, USER, HOST, SERVICE. The array should be terminated by the value GLOBUS\_SO\_END.

Returns:

GLOBUS\_SUCCESS unless handle\_attrs is null

3.4.3.7 `globus_result_t globus_gsi_cred_handle_attrs_get_search_order(globus_gsi_cred_handle_attrs_t handle_attrs, globus_gsi_cred_type_t search_order)`

Get the search order of the handle attributes

Parameters:

`handle_attrs` The handle attributes to get the search order from

`search_order` The search\_order of the handle attributes

Returns:

GLOBUS\_SUCCESS unless `handle_attrs` is null

## 3.5 Credential Operations

Read/Write a GSI Credential Handle.

Read Credential

- `globus_result_t globus_gsi_cred_read(globus_gsi_cred_handle_t handle, X509_NAME desired_subject)`

Reading Proxy Credentials

- `globus_result_t globus_gsi_cred_read_proxy(globus_gsi_cred_handle_t handle, const char proxy_lename)`
- `globus_result_t globus_gsi_cred_read_proxy_bio(globus_gsi_cred_handle_t handle, BIO bio)`

Read Key

- `globus_result_t globus_gsi_cred_read_key(globus_gsi_cred_handle_t handle, char key_lename, int( pw_-cb)())`

Read Cert

- `globus_result_t globus_gsi_cred_read_cert(globus_gsi_cred_handle_t handle, char cert_lename)`

Read Cert & Key in PKCS12 Format

- `globus_result_t globus_gsi_cred_read_pkcs12(globus_gsi_cred_handle_t handle, char pkcs12_lename)`

Write Credential

- `globus_result_t globus_gsi_cred_write(globus_gsi_cred_handle_t handle, BIO bio)`
- `globus_result_t globus_gsi_cred_write_proxy(globus_gsi_cred_handle_t handle, char proxy_lename)`

Get the X509 certificate type (EEC, CA, proxy type, etc.)

- `globus_result_t globus_gsi_cred_get_cert_type(globus_gsi_cred_handle_t handle, globus_gsi_cert_utils_cert_type_t type)`



### 3.5.1 Detailed Description

Read/Write a GSI Credential Handle.

This section defines operations to read and write GSI Credential handles.

### 3.5.2 Function Documentation

3.5.2.1 `globus_result_t globus_gsi_cred_read(globus\_gsi\_cred\_handle\_t handle, X509_NAME desired_subject)`

Read a Credential from a `lesystem` location. The credential to read will be determined by the search order specified in the handle attributes.

Parameters:

`handle` The credential handle to set. This credential handle should already be initialized using `globus_gsi_cred_handle_init`.

`desired_subject` The subject to check for when reading in a credential. The `desired_subject` should be either an exact match of the read cert's subject or should just contain the `/CN` entry. If null, the credential read in is the first match based on the system configuration (paths and environment variables)

Returns:

`GLOBUS_SUCCESS` if no errors occurred, otherwise, an error object identifier is returned.

See also:

[globus\\_gsi\\_cred\\_read\\_proxy\(\)](#)

[globus\\_gsi\\_cred\\_read\\_cert\\_and\\_key\(\)](#)

Note:

This function always searches for the desired credential. If you don't want to perform a search, then don't use this function. The search goes in the order of the handle attributes' search order.

3.5.2.2 `globus_result_t globus_gsi_cred_read_proxy(globus\_gsi\_cred\_handle\_t handle, const char proxy_lease)`

Read a proxy from a PEM file.

Parameters:

`handle` The credential handle to set based on the proxy credential read from the file

`proxy_lease` The file containing the proxy credential

Returns:

`GLOBUS_SUCCESS` or an error object identifier

3.5.2.3 `globus_result_t globus_gsi_cred_read_proxy_bio(globus\_gsi\_cred\_handle\_t handle, BIO bio)`

Read a Proxy Credential from a BIO stream and set the credential handle to represent the read credential. The values read from the stream, in order, will be the signed certificate, the private key, and the certificate chain

Parameters:

`handle` The credential handle to set. The credential should handle be initialized (i.e. not NULL).

bio The stream to read the credential from

Returns:

GLOBUS\_SUCCESS unless an error occurred, in which case an error object is returned

3.5.2.4 globus\_result\_t globus\_gsi\_cred\_read\_key(globus\_gsi\_cred\_handle\_t handle, char key\_ lename, int( pw\_cb)())

Read a key from a PEM le.

Parameters:

handle the handle to set based on the key that is read

key\_ lename the lename of the key to read

pw\_cb the callback for obtaining a password for decrypting the key.

Returns:

GLOBUS\_SUCCESS or an error object identifier

3.5.2.5 globus\_result\_t globus\_gsi\_cred\_read\_cert(globus\_gsi\_cred\_handle\_t handle, char cert\_ lename)

Read a cert from a le. Cert should be in PEM format.

Parameters:

handle the handle to set based on the certificate that is read

cert\_ lename the lename of the certificate to read

Returns:

GLOBUS\_SUCCESS or an error object identifier

3.5.2.6 globus\_result\_t globus\_gsi\_cred\_read\_pkcs12(globus\_gsi\_cred\_handle\_t handle, char pkcs12\_ lename)

Read a cert & key from a le. The le should be in PKCS12 format.

Parameters:

handle the handle to populate with the read credential

pkcs12\_ lename the lename containing the credential to read

Returns:

GLOBUS\_SUCCESS or an error object identifier

3.5.2.7 globus\_result\_t globus\_gsi\_cred\_write(globus\_gsi\_cred\_handle\_t handle, BIO bio)

Write out a credential to a BIO. The credential parameters written, in order, are the signed certificate, the RSA private key, and the certificate chain (a set of X509 certificates). the credential is written out in PEM format.

Parameters:

handle The credential to write out

bio The BIO stream to write out to

Returns:

GLOBUS\_SUCCESS unless an error occurred, in which case an error object ID is returned.

3.5.2.8 `globus_result_t globus_gsi_cred_write_proxy(globus\_gsi\_cred\_handle\_t handle, char proxy_lename)`

Write out a credential to a file. The credential parameters written, in order, are the signed certificate, the RSA private key, and the certificate chain (a set of X509 certificates). the credential is written out in PEM format.

Parameters:

handle The credential to write out  
proxy\_lename The file to write out to

Returns:

GLOBUS\_SUCCESS unless an error occurred, in which case an error object ID is returned.

3.5.2.9 `globus_result_t globus_gsi_cred_get_cert_type(globus\_gsi\_cred\_handle\_t handle, globus_gsi_cert_utils_cert_type_t type)`

Determine the type of the given X509 certificate For the list of possible values returned, see `globus_gsi_cert_utils_cert_type_t`.

Parameters:

handle The credential handle containing the certificate  
type The returned X509 certificate type

Returns:

GLOBUS\_SUCCESS or an error captured in a `globus_result_t`

## Index

Activation, [3](#)

Credential Constants, [4](#)

Credential Handle Attributes, [4,2](#)

Credential Handle Management, [4t](#)

Credential Operations, [4,5](#)

GLOBUS\_GSI\_CRED\_ERROR\_BAD\_-  
PARAMETER

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_CHECKING\_-  
PROXY

globus\_gsi\_credential\_constants, [2s](#)

GLOBUS\_GSI\_CRED\_ERROR\_CREATING\_-  
ERROR\_OBJ

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_ERRNO

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_GETTING\_-  
SERVICE\_NAME

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_KEY\_IS\_PASS\_-  
PROTECTED

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_LAST

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_NO\_CRED\_-  
FOUND

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_READING\_CRED

globus\_gsi\_credential\_constants, [2s](#)

GLOBUS\_GSI\_CRED\_ERROR\_READING\_-  
HOST\_CRED

globus\_gsi\_credential\_constants, [2s](#)

GLOBUS\_GSI\_CRED\_ERROR\_READING\_-  
PROXY\_CRED

globus\_gsi\_credential\_constants, [2s](#)

GLOBUS\_GSI\_CRED\_ERROR\_READING\_-  
SERVICE\_CRED

globus\_gsi\_credential\_constants, [2s](#)

GLOBUS\_GSI\_CRED\_ERROR\_SUBJECT\_CMP

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_SUCCESS

globus\_gsi\_credential\_constants, [2s](#)

GLOBUS\_GSI\_CRED\_ERROR\_SYSTEM\_CONFIG

globus\_gsi\_credential\_constants, [3s](#)

globus\_gsi\_cred\_error\_t

globus\_gsi\_credential\_constants, [2s](#)

GLOBUS\_GSI\_CRED\_ERROR\_VERIFYING\_-  
CRED

globus\_gsi\_credential\_constants, [2s](#)

GLOBUS\_GSI\_CRED\_ERROR\_WITH\_-  
CALLBACK\_DATA

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED

globus\_gsi\_credential\_constants, [2s](#)

GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED\_-  
CERT

globus\_gsi\_credential\_constants, [2s](#)

GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED\_-  
CERT\_CHAIN

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED\_-  
CERT\_NAME

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED\_-  
HANDLE\_ATTRS

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED\_-  
PRIVATE\_KEY

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_WITH\_SSL\_CTX

globus\_gsi\_credential\_constants, [3s](#)

GLOBUS\_GSI\_CRED\_ERROR\_WRITING\_CRED

globus\_gsi\_credential\_constants, [2s](#)

GLOBUS\_GSI\_CRED\_ERROR\_WRITING\_-  
PROXY\_CRED

globus\_gsi\_credential\_constants, [2s](#)

globus\_gsi\_cred\_get\_cert

globus\_gsi\_cred\_handle, [6](#)

globus\_gsi\_cred\_get\_cert\_chain

globus\_gsi\_cred\_handle, [6](#)

globus\_gsi\_cred\_get\_cert\_type

globus\_gsi\_cred\_operations, [1,8](#)

globus\_gsi\_cred\_get\_goodtill

globus\_gsi\_cred\_handle, [7](#)

globus\_gsi\_cred\_get\_handle\_attrs

globus\_gsi\_cred\_handle, [7](#)

globus\_gsi\_cred\_get\_identity\_name

globus\_gsi\_cred\_handle, [6,1](#)

globus\_gsi\_cred\_get\_issuer\_name

globus\_gsi\_cred\_handle, [6,1](#)

globus\_gsi\_cred\_get\_key

globus\_gsi\_cred\_handle, [6](#)

globus\_gsi\_cred\_get\_key\_bits

globus\_gsi\_cred\_handle, [6](#)

globus\_gsi\_cred\_get\_lifetime

globus\_gsi\_cred\_handle, [7](#)

globus\_gsi\_cred\_get\_policies

globus\_gsi\_cred\_handle, [6,0](#)

globus\_gsi\_cred\_get\_policy\_languages

globus\_gsi\_cred\_handle [10](#)  
 globus\_gsi\_cred\_get\_subject\_name  
     globus\_gsi\_cred\_handle [10](#)  
 globus\_gsi\_cred\_get\_X509\_identity\_name  
     globus\_gsi\_cred\_handle [10](#)  
 globus\_gsi\_cred\_get\_X509\_issuer\_name  
     globus\_gsi\_cred\_handle [11](#)  
 globus\_gsi\_cred\_get\_X509\_subject\_name  
     globus\_gsi\_cred\_handle [10](#),  
 globus\_gsi\_cred\_handle  
     globus\_gsi\_cred\_get\_cert [8](#),  
     globus\_gsi\_cred\_get\_cert\_chain [9](#),  
     globus\_gsi\_cred\_get\_goodti [11](#),  
     globus\_gsi\_cred\_get\_handle\_attrs [13](#),  
     globus\_gsi\_cred\_get\_identity\_name [10](#),  
     globus\_gsi\_cred\_get\_issuer\_name [11](#),  
     globus\_gsi\_cred\_get\_key [9](#),  
     globus\_gsi\_cred\_get\_key\_bits [18](#),  
     globus\_gsi\_cred\_get\_lifetime [7](#),  
     globus\_gsi\_cred\_get\_policies [10](#)  
     globus\_gsi\_cred\_get\_policy\_language [13](#),  
     globus\_gsi\_cred\_get\_subject\_name [10](#),  
     globus\_gsi\_cred\_get\_X509\_identity\_name [10](#),  
     globus\_gsi\_cred\_get\_X509\_issuer\_name [11](#),  
     globus\_gsi\_cred\_get\_X509\_subject\_name [10](#),  
     globus\_gsi\_cred\_handle\_copy [7](#),  
     globus\_gsi\_cred\_handle\_destroy [6](#),  
     globus\_gsi\_cred\_handle\_init [6](#),  
     globus\_gsi\_cred\_handle [6](#),  
     globus\_gsi\_cred\_set\_cert [8](#),  
     globus\_gsi\_cred\_set\_cert\_chain [9](#),  
     globus\_gsi\_cred\_set\_key [9](#),  
     globus\_gsi\_cred\_verify [12](#)  
     globus\_gsi\_cred\_verify\_cert\_chain [11](#)  
 globus\_gsi\_cred\_handle\_attrs  
     globus\_gsi\_cred\_handle\_attrs\_copy [13](#),  
     globus\_gsi\_cred\_handle\_attrs\_destroy [13](#),  
     globus\_gsi\_cred\_handle\_attrs\_get\_ca\_cert\_dir,  
         [14](#)  
     globus\_gsi\_cred\_handle\_attrs\_get\_search\_order,  
         [14](#)  
     globus\_gsi\_cred\_handle\_attrs\_init [13](#)  
     globus\_gsi\_cred\_handle\_attrs\_set\_ca\_cert\_dir,  
         [14](#)  
     globus\_gsi\_cred\_handle\_attrs\_set\_search\_order,  
         [14](#)  
     globus\_gsi\_cred\_handle\_attrs [13](#),  
 globus\_gsi\_cred\_handle\_attrs\_copy  
     globus\_gsi\_cred\_handle\_attrs [13](#),  
 globus\_gsi\_cred\_handle\_attrs\_destroy  
     globus\_gsi\_cred\_handle\_attrs [13](#),  
 globus\_gsi\_cred\_handle\_attrs\_get\_ca\_cert\_dir  
     globus\_gsi\_cred\_handle\_attrs [13](#),  
 globus\_gsi\_cred\_handle\_attrs\_get\_search\_order  
     globus\_gsi\_cred\_handle\_attrs [13](#),  
 globus\_gsi\_cred\_handle\_attrs\_init  
     globus\_gsi\_cred\_handle\_attrs [13](#),  
 globus\_gsi\_cred\_handle\_attrs\_set\_ca\_cert\_dir  
     globus\_gsi\_cred\_handle\_attrs [13](#),  
 globus\_gsi\_cred\_handle\_attrs\_set\_search\_order  
     globus\_gsi\_cred\_handle\_attrs [13](#),  
 globus\_gsi\_cred\_handle\_copy  
     globus\_gsi\_cred\_handle [7](#),  
 globus\_gsi\_cred\_handle\_destroy  
     globus\_gsi\_cred\_handle [6](#),  
 globus\_gsi\_cred\_handle\_init  
     globus\_gsi\_cred\_handle [6](#),  
 globus\_gsi\_cred\_handle\_t  
     globus\_gsi\_cred\_handle [6](#),  
 globus\_gsi\_cred\_operations  
     globus\_gsi\_cred\_get\_cert\_type [18](#),  
     globus\_gsi\_cred\_read [16](#)  
     globus\_gsi\_cred\_read\_cert [17](#),  
     globus\_gsi\_cred\_read\_key [17](#),  
     globus\_gsi\_cred\_read\_pkcs [112](#),  
     globus\_gsi\_cred\_read\_proxy [16](#),  
     globus\_gsi\_cred\_read\_proxy\_bio [16](#),  
     globus\_gsi\_cred\_write [17](#),  
     globus\_gsi\_cred\_write\_proxy [17](#)  
 globus\_gsi\_cred\_read  
     globus\_gsi\_cred\_operations [16](#),  
 globus\_gsi\_cred\_read\_cert  
     globus\_gsi\_cred\_operations [17](#),  
 globus\_gsi\_cred\_read\_key  
     globus\_gsi\_cred\_operations [17](#),  
 globus\_gsi\_cred\_read\_pkcs12  
     globus\_gsi\_cred\_operations [17](#),  
 globus\_gsi\_cred\_read\_proxy  
     globus\_gsi\_cred\_operations [16](#),  
 globus\_gsi\_cred\_read\_proxy\_bio  
     globus\_gsi\_cred\_operations [16](#),  
 globus\_gsi\_cred\_set\_cert  
     globus\_gsi\_cred\_handle [8](#),  
 globus\_gsi\_cred\_set\_cert\_chain  
     globus\_gsi\_cred\_handle [8](#),  
 globus\_gsi\_cred\_set\_key  
     globus\_gsi\_cred\_handle [8](#),  
 globus\_gsi\_cred\_type\_t  
     globus\_gsi\_credential\_constants [18](#),  
 globus\_gsi\_cred\_verify  
     globus\_gsi\_cred\_handle [12](#)  
 globus\_gsi\_cred\_verify\_cert\_chain  
     globus\_gsi\_cred\_handle [11](#)  
 globus\_gsi\_cred\_write

globus\_gsi\_cred\_operations [17](#),  
 globus\_gsi\_cred\_write\_proxy  
 globus\_gsi\_cred\_operations [17](#),  
 globus\_gsi\_credential\_activation  
 GLOBUS\_GSI\_CREDENTIAL\_MODULE [4](#)  
 globus\_gsi\_credential\_constants  
 GLOBUS\_GSI\_CRED\_ERROR\_BAD\_-  
 PARAMETER, [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_CHECKING\_-  
 PROXY, [2](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_CREATING\_-  
 ERROR\_OBJ [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_ERRNO [3](#),  
 GLOBUS\_GSI\_CRED\_ERROR\_GETTING\_-  
 SERVICE\_NAME, [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_KEY\_IS\_-  
 PASS\_PROTECTED [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_LAST [3](#),  
 GLOBUS\_GSI\_CRED\_ERROR\_NO\_CRED\_-  
 FOUND, [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_READING\_-  
 CRED, [2](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_READING\_-  
 HOST\_CRED [2](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_READING\_-  
 PROXY\_CRED [2](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_READING\_-  
 SERVICE\_CRED [2](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_SUBJECT\_-  
 CMP, [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_SUCCESS [3](#),  
 GLOBUS\_GSI\_CRED\_ERROR\_SYSTEM\_-  
 CONFIG, [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_-  
 VERIFYING\_CRED, [2](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_-  
 CALLBACK\_DATA, [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_CRED,  
[2](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_-  
 CRED\_CERT [2](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_-  
 CRED\_CERT\_CHAIN, [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_-  
 CRED\_CERT\_NAME, [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_-  
 CRED\_HANDLE\_ATTRS, [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_-  
 CRED\_PRIVATE\_KEY, [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_WITH\_SSL\_-  
 CTX, [3](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_WRITING\_-  
 CRED, [2](#)  
 GLOBUS\_GSI\_CRED\_ERROR\_WRITING\_-  
 PROXY\_CRED, [2](#)  
 globus\_gsi\_credential\_constants  
 globus\_gsi\_cred\_error [2](#),  
 globus\_gsi\_cred\_type [3](#),  
 GLOBUS\_GSI\_CREDENTIAL\_MODULE  
 globus\_gsi\_credential\_activation [4](#),