

# globus gsi proxy core Reference Manual

## 4.3

Generated by Doxygen 1.3.5

Sat Feb 6 11:06:32 2010

## Contents

<a href="#">1 Globus GSI Proxy API</a>	1
<a href="#">2 globus gsi proxy core Module Index</a>	1
<a href="#">3 globus gsi proxy core Module Documentation</a>	1

## 1 Globus GSI Proxy API

The `globus_gsi_proxy` library is motivated by the desire to provide a abstraction layer for the proxy creation and delegation process. For background on this process please refer to the proxy certificate profile draft.

Any program that uses Globus GSI Proxy functions must include "`globus_gsi_proxy.h`".

We envision the API being used in the following manner:

Delegator:	Delegatee:
	set desired cert info extension in the handle by using the handle set functions.
	<code>globus_gsi_proxy_create_req</code>
<code>globus_gsi_proxy_inquire_req</code>	
modify cert info extension by using handle set/get/clear functions.	
<code>globus_gsi_proxy_sign_req</code>	
	<code>globus_gsi_proxy_assemble_cred</code>

## 2 globus gsi proxy core Module Index

### 2.1 globus gsi proxy core Modules

Here is a list of all modules:

Activation	1
Handle Management	2
Handle Attributes	13
Proxy Operations	18
Proxy Constants	21

## 3 globus gsi proxy core Module Documentation

### 3.1 Activation

Globus GSI Proxy uses standard Globus module activation and deactivation.

De nes

- #define [GLOBUS\\_GSI\\_PROXY\\_MODULE](#)

### 3.1.1 Detailed Description

Globus GSI Proxy uses standard Globus module activation and deactivation.

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

```
globus_module_activate(GLOBUS_GSI_PROXY_MODULE)
```

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

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

```
globus_module_deactivate(GLOBUS_GSI_PROXY_MODULE)
```

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

### 3.1.2 De ne Documentation

#### 3.1.2.1 #define GLOBUS\_GSI\_PROXY\_MODULE

Module descriptor.

## 3.2 Handle Management

Create/Destroy/Modify a GSI Proxy Handle.

Initialize and Destroy

- `globus_result_t` [globus\\_gsi\\_proxy\\_handle\\_init](#)(`globus_gsi_proxy_handle_t` handle, `globus_gsi_proxy_handle_attrs_t` handle\_attrs)
- `globus_result_t` [globus\\_gsi\\_proxy\\_handle\\_destroy](#)(`globus_gsi_proxy_handle_t` handle)

Get/Set Request

- `globus_result_t` [globus\\_gsi\\_proxy\\_handle\\_get\\_req](#)(`globus_gsi_proxy_handle_t` handle, `X509_REQ` req)
- `globus_result_t` [globus\\_gsi\\_proxy\\_handle\\_set\\_req](#)(`globus_gsi_proxy_handle_t` handle, `X509_REQ` req)

Get/Set Private Key

- `globus_result_t` [globus\\_gsi\\_proxy\\_handle\\_get\\_private\\_key](#)(`globus_gsi_proxy_handle_t` handle, `EVP_PKEY` proxy\_key)
- `globus_result_t` [globus\\_gsi\\_proxy\\_handle\\_set\\_private\\_key](#)(`globus_gsi_proxy_handle_t` handle, `EVP_PKEY` proxy\_key)

## Get/Set Proxy Type

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_get\\_type](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, globus\_gsi\_cert\_utils\_cert\_type\_t type)
- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_set\\_type](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, globus\_gsi\_cert\_utils\_cert\_type\_t type)

## Get/Set Policy

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_set\\_policy](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, unsigned char policy\_data, int policy\_length, int policy\_language\_NID)
- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_get\\_policy](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, unsigned char policy\_data, int policy\_length, int policy\_NID)

## Get/Set X509 Extensions

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_add\\_extension](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, X509\_EXTENSION ext)
- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_set\\_extensions](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, STACK\_OF(X509\_EXTENSION) exts)
- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_get\\_extensions](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, STACK\_OF(X509\_EXTENSION) exts)

## Get/Set Path Length

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_set\\_path\\_length](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, long pathlen)
- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_get\\_path\\_length](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, int pathlen)

## Get/Set Time Valid

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_get\\_time\\_valid](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, int time\_valid)
- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_set\\_time\\_valid](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, int time\_valid)

## Clear Cert Info

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_clear\\_cert\\_info](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle)

## Get/Set Cert Info

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_get\\_proxy\\_cert\\_info](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, PROXYCERTINFO pci)
- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_set\\_proxy\\_cert\\_info](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, PROXYCERTINFO pci)

## Get Signing Algorithm

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_get\\_signing\\_algorithm](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, EVP\_MD signing\_algorithm)

### Get Key Bits

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_get\\_key\\_bits](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, int key\_bits)

### Get Init Prime

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_get\\_init\\_prime](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, int init\_prime)

### Get Clock Skew

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_get\\_clock\\_skew\\_allowance](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, int skew)

### Get Callback for Creating Keys

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_get\\_key\\_gen\\_callback](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, void( callback)(int, int, void ))

### Get/Set Proxy Common Name

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_get\\_common\\_name](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, char common\_name)
- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_set\\_common\\_name](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, char common\_name)

### Set/Check Proxy Is Limited

- globus\_result\_t [globus\\_gsi\\_proxy\\_handle\\_set\\_is\\_limited](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, globus\_bool\_t is\_limited)
- globus\_result\_t [globus\\_gsi\\_proxy\\_is\\_limited](#)([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, globus\_bool\_t is\_limited)

### Typedefs

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

#### 3.2.1 Detailed Description

Create/Destroy/Modify a GSI Proxy Handle.

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

This section defines operations to create, modify and destroy GSI Proxy handles.

#### 3.2.2 Typedef Documentation

##### 3.2.2.1 typedef struct globus\_l\_gsi\_proxy\_handle\_t [globus\\_gsi\\_proxy\\_handle\\_t](#)

GSI Proxy Handle.

An GSI Proxy handle is used to associate state with a group of operations. Handles can have immutable attributes associated with them. All proxy operations take a handle pointer as a parameter.

See also:

[globus\\_gsi\\_proxy\\_handle\\_init\(\)](#), [globus\\_gsi\\_proxy\\_handle\\_destroy\(\)](#), [Handle Attributes](#)

### 3.2.3 Function Documentation

3.2.3.1 `globus_result_t globus_gsi_proxy_handle_init(globus_gsi_proxy_handle_t handle, globus_gsi_proxy_handle_attr_t handle_attr)`

Initialize a GSI Proxy handle.

Initialize a proxy handle which can be used in subsequent operations. The handle may only be used in one sequence of operations at a time.

Parameters:

handle A pointer to the handle to be initialized. If the handle is originally NULL, space is allocated for it. Otherwise, the current values of the handle are overwritten.

handle\_attr Initial attributes to be used to create this handle.

Returns:

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

See also:

[globus\\_gsi\\_proxy\\_handle\\_destroy\(\)](#)

3.2.3.2 `globus_result_t globus_gsi_proxy_handle_get_req(globus_gsi_proxy_handle_t handle, X509_REQ req)`

Get the certificate request from a GSI Proxy handle.

Parameters:

handle The handle from which to get the certificate request

req Parameter used to return the request. It is the users responsibility to free the returned request.

Returns:

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

See also:

[globus\\_gsi\\_proxy\\_handle\\_set\\_req\(\)](#)

3.2.3.3 `globus_result_t globus_gsi_proxy_handle_get_private_key(globus_gsi_proxy_handle_t handle, EVP_PKEY proxy_key)`

Get the private key from a GSI Proxy handle.

Parameters:

handle The handle from which to get the private key

proxy\_key Parameter used to return the key. It is the users responsibility to free the returned key.

**Returns:**

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

**See also:**

[globus\\_gsi\\_proxy\\_handle\\_set\\_private\\_key\(\)](#)

3.2.3.4 `globus_result_t globus_gsi_proxy_handle_get_type(globus\_gsi\_proxy\_handle\_t handle, globus_gsi_cert_utils_cert_type_t type)`

Determine the type of proxy that will be generated when using this handle.

**Parameters:**

handle The handle from which to get the type

type Parameter used to return the type.

**Returns:**

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

**See also:**

[globus\\_gsi\\_proxy\\_handle\\_set\\_type\(\)](#)

3.2.3.5 `globus_result_t globus_gsi_proxy_handle_set_policy(globus\_gsi\_proxy\_handle\_t handle, unsigned char policy_data, int policy_length, int policy_language_NID)`

Set the policy to be used in the GSI Proxy handle.

This function sets the policy to be used in the proxy cert info extension.

**Parameters:**

handle The handle to be modified.

policy\_data The policy data.

policy\_length The length of the policy data

policy\_language\_NID The NID of the policy language.

**Returns:**

GLOBUS\_SUCCESS if the handle and its associated fields are valid otherwise an error is returned

**See also:**

[globus\\_gsi\\_proxy\\_handle\\_get\\_policy\(\)](#)

3.2.3.6 `globus_result_t globus_gsi_proxy_handle_add_extension(globus\_gsi\_proxy\_handle\_t handle, X509_EXTENSION ext)`

Add an X509 extension to the GSI Proxy handle to be added to certificate.

This function adds a X509 extension to the proxy certificate.

**Parameters:**

handle The handle for the proxy to which the extension should be added.

extension The extension to be added.

**Returns:**

GLOBUS\_SUCCESS if the addition was successful, otherwise an error is returned.

**See also:**

globus\_gsi\_proxy\_handle\_get\_extensions()  
globus\_gsi\_proxy\_handle\_set\_extensions()

3.2.3.7 globus\_result\_t globus\_gsi\_proxy\_handle\_set\_pathlen([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, long pathlen)

Set the path length to be used in the GSI Proxy handle.

This function sets the path length to be used in the proxy cert info extension.

**Parameters:**

handle The handle to be modified.  
pathlen The maximum allowable path length

**Returns:**

GLOBUS\_SUCCESS if the handle is valid, otherwise an error is returned

**See also:**

[globus\\_gsi\\_proxy\\_handle\\_get\\_pathlen\(\)](#)

3.2.3.8 globus\_result\_t globus\_gsi\_proxy\_handle\_get\_time\_valid([globus\\_gsi\\_proxy\\_handle\\_t](#) handle, int time\_valid)

Get the validity time of the proxy

**Parameters:**

handle The proxy handle to get the expiration date of  
time\_valid expiration date of the proxy handle

**Returns:**

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

3.2.3.9 globus\_result\_t globus\_gsi\_proxy\_handle\_clear\_cert\_info([globus\\_gsi\\_proxy\\_handle\\_t](#) handle)

Clear the proxy cert info extension stored in the GSI Proxy handle.

This function clears proxy cert info extension related setting in the GSI Proxy handle.

**Parameters:**

handle The handle for which to clear the proxy cert info extension.

**Returns:**

GLOBUS\_SUCCESS if the handle is valid, otherwise an error is returned



3.2.3.10 `globus_result_t globus_gsi_proxy_handle_get_proxy_cert_info(globus_gsi_proxy_handle_t handle, PROXYCERTINFO pci)`

Get the proxy cert info extension stored in the GSI Proxy handle.

This function retrieves the proxy cert info extension from the GSI Proxy handle.

Parameters:

handle The handle from which to get the proxy cert info extension.

pci Contains the proxy cert info extension upon successful return. If the handle does not contain a pci extension, this parameter will be NULL upon return.

Returns:

GLOBUS\_SUCCESS upon success GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_HANDLE if handle is invalid

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_PROXYCERTINFO if the pci pointer is invalid or if the get failed.

3.2.3.11 `globus_result_t globus_gsi_proxy_handle_get_signing_algorithm(globus_gsi_proxy_handle_t handle, EVP_MD signing_algorithm)`

Get the signing algorithm used to sign the proxy cert request

Parameters:

handle The proxy handle containing the type of signing algorithm used

signing\_algorithm signing algorithm of the proxy handle

Returns:

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

3.2.3.12 `globus_result_t globus_gsi_proxy_handle_get_key_bits(globus_gsi_proxy_handle_t handle, int key_bits)`

Get the key bits used for the pub/private key pair of the proxy

Parameters:

handle The proxy handle to get the key bits of

key\_bits key bits of the proxy handle

Returns:

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

3.2.3.13 `globus_result_t globus_gsi_proxy_handle_get_init_prime(globus_gsi_proxy_handle_t handle, int init_prime)`

Get the init prime of the proxy handle

Parameters:

handle The handle to get the init prime used in generating the key pair

init\_prime The resulting init prime

Returns:

GLOBUS\_SUCCESS unless an error occurred, in which case an error object identifier (in the form of a globus\_result\_t) is returned

3.2.3.14 `globus_result_t globus_gsi_proxy_handle_get_clock_skew_allowable(globus_gsi_proxy_handle_t handle, int skew)`

Get the clock skew of the proxy handle

Parameters:

handle The handle to get the clock skew of  
skew The resulting clock skew

Returns:

GLOBUS\_SUCCESS unless an error occurred, in which case an error object identifier (in the form of a `globus_result_t`) is returned

3.2.3.15 `globus_result_t globus_gsi_proxy_handle_get_key_gen_callback(globus_gsi_proxy_handle_t handle, void( callback)(int, int, void ))`

Get the callback for creating the public/private key pair

Parameters:

handle The proxy handle to get the callback from  
callback Parameter used for returning the callback

Returns:

GLOBUS\_SUCCESS or an error object identifier

3.2.3.16 `globus_result_t globus_gsi_proxy_handle_get_common_name(globus_gsi_proxy_handle_t handle, char common_name)`

Get the proxy common name stored in the GSI Proxy handle.

This function retrieves the proxy common name from the GSI Proxy handle. The common name only impacts draft compliant proxies.

Parameters:

handle The handle from which to get the proxy common name.  
common\_name Contains the proxy common name upon successful return. If the handle does not contain a common name, this parameter will be NULL upon return.

Returns:

GLOBUS\_SUCCESS upon success GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_HANDLE if handle is invalid

3.2.3.17 `globus_result_t globus_gsi_proxy_handle_set_is_limited(globus_gsi_proxy_handle_t handle, globus_bool_t is_limited)`

Set the limited proxy flag on the proxy handle

Parameters:

handle the proxy handle  
is\_limited boolean value to set on the proxy handle

Returns:

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

3.2.3.18 `globus_result_t globus_gsi_proxy_handle_destroy(globus\_gsi\_proxy\_handle\_t handle)`

Destroy a GSI Proxy handle.

Parameters:

`handle` The handle to be destroyed.

Returns:

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

See also:

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

3.2.3.19 `globus_result_t globus_gsi_proxy_handle_set_req(globus\_gsi\_proxy\_handle\_t handle, X509_REQ req)`

Set the certificate request in a GSI Proxy handle.

Parameters:

`handle` The handle for which to set the certificate request

`req` Request to be copied to handle.

Returns:

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

See also:

[globus\\_gsi\\_proxy\\_handle\\_get\\_req\(\)](#)

3.2.3.20 `globus_result_t globus_gsi_proxy_handle_set_private_key(globus\_gsi\_proxy\_handle\_t handle, EVP_PKEY proxy_key)`

Set the private key in a GSI Proxy handle.

Parameters:

`handle` The handle for which to set the private key

`proxy_key` Parameter used to pass the key

Returns:

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

See also:

[globus\\_gsi\\_proxy\\_handle\\_get\\_private\\_key\(\)](#)

3.2.3.21 `globus_result_t globus_gsi_proxy_handle_set_type(globus\_gsi\_proxy\_handle\_t handle, globus_gsi_cert_utils_cert_type_t type)`

Set the type of proxy that will be generated when using this handle.

Note that this will have no effect when generating a proxy from a proxy. In that case the generated proxy will inherit the type of the parent.

## Parameters:

handle The handle for which to set the type  
 type Parameter used to pass the type.

## Returns:

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

## See also:

[globus\\_gsi\\_proxy\\_handle\\_set\\_type\(\)](#)

3.2.3.22 `globus_result_t globus_gsi_proxy_handle_get_policy(globus\_gsi\_proxy\_handle\_t handle, unsigned char policy_data, int policy_length, int policy_NID)`

Get the policy from the GSI Proxy handle.

This function gets the policy that is being used in the proxy cert info extension.

## Parameters:

handle The handle to be interrogated.  
 policy\_data The policy data.  
 policy\_length The length of the returned policy  
 policy\_NID The NID of the policy language.

## Returns:

GLOBUS\_SUCCESS if the handle is valid, otherwise an error is returned

## See also:

[globus\\_gsi\\_proxy\\_handle\\_set\\_policy\(\)](#)

3.2.3.23 `globus_result_t globus_gsi_proxy_handle_set_extensions(globus\_gsi\_proxy\_handle\_t handle, STACK_OF(X509_EXTENSION) exts)`

Set the X509 extensions from a GSI Proxy handle.

This function sets the X509 extensions for a proxy certificate.

## Parameters:

handle The handle for the proxy from which the extension should be set.  
 extensions The extensions to be set. Can be NULL to clear extensions.

## Returns:

GLOBUS\_SUCCESS if the addition was successful, otherwise an error is returned.

## See also:

[globus\\_gsi\\_proxy\\_handle\\_add\\_extension\(\)](#)  
[globus\\_gsi\\_proxy\\_handle\\_get\\_extensions\(\)](#)

3.2.3.24 `globus_result_t globus_gsi_proxy_handle_get_extensions(globus\_gsi\_proxy\_handle\_t handle, STACK_OF(X509_EXTENSION) *exts)`

Get the X509 extensions from a GSI Proxy handle.

This function returns the X509 extensions from the proxy certificate.

Parameters:

`handle` The handle for the proxy from which the extensions should be retrieved.

`exts` The variable to hold the extensions. The caller is responsible for freeing the extensions with `sk_X509_EXTENSION_free()` when they are done with them.

Returns:

GLOBUS\_SUCCESS if the retrieval was successful, otherwise an error is returned.

See also:

[globus\\_gsi\\_proxy\\_handle\\_add\\_extension\(\)](#)

[globus\\_gsi\\_proxy\\_handle\\_set\\_extensions\(\)](#)

3.2.3.25 `globus_result_t globus_gsi_proxy_handle_get_pathlen(globus\_gsi\_proxy\_handle\_t handle, int *pathlen)`

Get the path length from the GSI Proxy handle.

This function gets the path length that is being used in the proxy cert info extension.

Parameters:

`handle` The handle to be interrogated.

`pathlen` The maximum allowable path length

Returns:

GLOBUS\_SUCCESS if the handle is valid, otherwise an error is returned

See also:

[globus\\_gsi\\_proxy\\_handle\\_set\\_pathlen\(\)](#)

3.2.3.26 `globus_result_t globus_gsi_proxy_handle_set_time_valid(globus\_gsi\_proxy\_handle\_t handle, int time_valid)`

Set the validity time of the proxy.

Parameters:

`handle` The proxy handle to set the expiration date for

`time_valid` desired expiration date of the proxy

Returns:

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

3.2.3.27 `globus_result_t globus_gsi_proxy_handle_set_proxy_cert_info(globus_gsi_proxy_handle_t handle, PROXYCERTINFO pci)`

Set the proxy cert info extension stored in the GSI Proxy handle.

This function sets the proxy cert info extension in the GSI Proxy handle.

Parameters:

handle The handle for which to set the proxy cert info extension.

pci The proxy cert info extension to set.

Returns:

GLOBUS\_SUCCESS upon success GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_HANDLE if handle is invalid

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_PROXYCERTINFO if the pci pointer is invalid or if the set failed.

3.2.3.28 `globus_result_t globus_gsi_proxy_handle_set_common_name(globus_gsi_proxy_handle_t handle, char common_name)`

Set the proxy common name stored in the GSI Proxy handle.

This function sets the proxy common name in the GSI Proxy handle. Note that the common name is only used for draft compliant proxies.

Parameters:

handle The handle for which to set the proxy common name.

common\_name The proxy common name to set.

Returns:

GLOBUS\_SUCCESS upon success GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_HANDLE if handle is invalid

3.2.3.29 `globus_result_t globus_gsi_proxy_is_limited(globus_gsi_proxy_handle_t handle, globus_bool_t is_limited)`

Check to see if the proxy is a limited proxy.

Parameters:

handle the proxy handle to check

is\_limited boolean value to set depending on the type of proxy

Returns:

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

## 3.3 Handle Attributes

Handle attributes are used to control additional features of the GSI Proxy handle.

Initialize & Destroy

- `globus_result_t globus_gsi_proxy_handle_attrs_init(globus_gsi_proxy_handle_attrs_t handle_attrs)`
- `globus_result_t globus_gsi_proxy_handle_attrs_destroy(globus_gsi_proxy_handle_attrs_t handle_attrs)`

## Get/Set Key Bits

- globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_set\_key\_bits(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, int bits)
- globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_get\_key\_bits(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, int bits)

## Get/Set Initial Prime Number

- globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_set\_init\_prime(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, int prime)
- globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_get\_init\_prime(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, int prime)

## Get/Set Signing Algorithm

- globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_set\_signing\_algorithm(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, EVP\_MD algorithm)
- globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_get\_signing\_algorithm(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, EVP\_MD algorithm)

## Get/Set Clock Skew Allowable

- globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_set\_clock\_skew\_allowable(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, int skew)
- globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_get\_clock\_skew\_allowable(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, int skew)

## Get/Set Key Gen Callback

- globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_get\_key\_gen\_callback(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, void( callback)(int, int, void ))
- globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_set\_key\_gen\_callback(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, void( callback)(int, int, void ))

## Copy Attributes

- globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_copy(globus\_gsi\_proxy\_handle\_attrs\_t a, globus\_gsi\_proxy\_handle\_attrs\_t b)

## Typedefs

- typedef globus\_l\_gsi\_proxy\_handle\_attrs\_t globus\_gsi\_proxy\_handle\_attrs\_t

## 3.3.1 Detailed Description

Handle attributes are used to control additional features of the GSI Proxy handle.

These features are operation independent.

Currently there are no attributes.

See also:

[globus\\_gsi\\_proxy\\_handle\\_t](#)

### 3.3.2 Typedef Documentation

#### 3.3.2.1 typedef struct globus\_l\_gsi\_proxy\_handle\_attrs\_t [globus\\_gsi\\_proxy\\_handle\\_attrs\\_t](#)

Handle Attributes.

A GSI Proxy handle attributes type is used to associate immutable parameter values [Handle Management](#) handle. A handle attributes object should be created with immutable parameters and then passed to the proxy handle init function [globus\\_gsi\\_proxy\\_handle\\_init\(\)](#)

See also:

[Handle Management](#)

### 3.3.3 Function Documentation

#### 3.3.3.1 globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_init([globus\\_gsi\\_proxy\\_handle\\_attrs\\_t](#) handle\_attrs)

Initialize GSI Proxy Handle Attributes.

Initialize proxy handle attributes, which can (and should) be associated with a proxy handle. For most purposes, these attributes should primarily be used by the proxy handle.

Currently, no attribute values are initialized.

Parameters:

handle\_attrs The handle attributes structure to be initialized

Returns:

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

See also:

[globus\\_gsi\\_proxy\\_handle\\_attrs\\_destroy\(\)](#)

#### 3.3.3.2 globus\_result\_t globus\_gsi\_proxy\_handle\_attrs\_set\_keybits([globus\\_gsi\\_proxy\\_handle\\_attrs\\_t](#) handle\_attrs, int bits)

Set the length of the public key pair used by the proxy certificate

Parameters:

handle\_attrs the attributes to set

bits the length to set it to (usually 1024)

Returns:

GLOBUS\_SUCCESS



3.3.3.3 `globus_result_t globus_gsi_proxy_handle_attrs_set_init_prime(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, int prime)`

Set the initial prime number used for generating public key pairs in the RSA algorithm

Parameters:

`handle_attrs` The attributes to set

`prime` The prime number to set it to This value needs to be a prime number

Returns:

GLOBUS\_SUCCESS

3.3.3.4 `globus_result_t globus_gsi_proxy_handle_attrs_set_signing_algorithm(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, EVP_MD algorithm)`

Sets the Signing Algorithm to be used to sign the certificate request. In most cases, the signing party will ignore this value, and sign with an algorithm of its choice.

Parameters:

`handle_attrs` The proxy handle to set the signing algorithm of

`algorithm` The signing algorithm to set

Returns:

Returns GLOBUS\_SUCCESS if the handle is valid, otherwise an error object is returned.

3.3.3.5 `globus_result_t globus_gsi_proxy_handle_attrs_set_clock_skew_allowable(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, int skew)`

Sets the clock skew in minutes of the proxy cert request so that time differences between hosts won't cause problems. This value defaults to 5 minutes.

Parameters:

`handle_attrs` the `handle_attrs` containing the clock skew to be set

`skew` the amount to skew by (in seconds)

Returns:

GLOBUS\_SUCCESS if the `handle_attrs` is valid - otherwise an error is returned.

3.3.3.6 `globus_result_t globus_gsi_proxy_handle_attrs_get_key_gen_callback(globus\_gsi\_proxy\_handle\_attrs\_t handle\_attrs, void( callback)(int, int, void ))`

Get the public/private key generation callback that provides status during the generation of the keys

Parameters:

`handle_attrs` The `handle_attrs` to get the callback from

`callback` The callback from the handle attributes

Returns:

GLOBUS\_SUCCESS if the `handle_attrs` is valid, otherwise an error is returned

3.3.3.7 `globus_result_t globus_gsi_proxy_handle_attrs_copy(globus\_gsi\_proxy\_handle\_attrs\_t a, globus\_gsi\_proxy\_handle\_attrs\_t b)`

Make a copy of GSI Proxy handle attributes

Parameters:

- a The handle attributes to copy
- b The copy

Returns:

GLOBUS\_SUCCESS

3.3.3.8 `globus_result_t globus_gsi_proxy_handle_attrs_destroy(globus\_gsi\_proxy\_handle\_attrs\_t handle_attrs)`

Destroy the GSI Proxy handle attributes.

Parameters:

- handle\_attrs The handle attributes to be destroyed.

Returns:

GLOBUS\_SUCCESS

See also:

[globus\\_gsi\\_proxy\\_handle\\_attrs\\_init\(\)](#)

3.3.3.9 `globus_result_t globus_gsi_proxy_handle_attrs_get_keybits(globus\_gsi\_proxy\_handle\_attrs\_t handle_attrs, int *bits)`

Gets the length of the public key pair used by the proxy certificate.

Parameters:

- handle\_attrs the attributes to get the key length from
- bits the length of the key pair in bits

Returns:

GLOBUS\_SUCCESS

3.3.3.10 `globus_result_t globus_gsi_proxy_handle_attrs_get_init_prime(globus\_gsi\_proxy\_handle\_attrs\_t handle_attrs, int *prime)`

Get the initial prime number used for generating the public key pair in the RSA algorithm.

Parameters:

- handle\_attrs The attributes to get the initial prime number from
- prime The initial prime number taken from the attributes

Returns:

GLOBUS\_SUCCESS

3.3.3.11 `globus_result_t globus_gsi_proxy_handle_attrs_get_signing_algorithm(globus\_gsi\_proxy\_handle\_attrs\_t handle_attrs, EVP_MD *algorithm)`

Gets the Signing Algorithm to used to sign the certi cate request.

In most cases, the signing party will ignore this value, and sign with an algorithm of its choice.

Parameters:

`handle_attrs` The proxy handle\_attrs to get the signing algorithm of  
`algorithm` Parameter used to return the signing algorithm used

Returns:

Returns GLOBUS\_SUCCESS if the handle is valid, otherwise an error object is returned.

3.3.3.12 `globus_result_t globus_gsi_proxy_handle_attrs_get_clock_skew_allowable(globus\_gsi\_proxy\_handle\_attrs\_t handle_attrs, int *skew)`

Get the allowable clock skew for the proxy certi cate.

Parameters:

`handle_attrs` The handle\_attrs to get the clock skew from  
`skew` The allowable clock skew (in seconds) to get from the proxy certi cate request. This value gets set by the function, so it needs to be a pointer.

Returns:

GLOBUS\_SUCCESS if the handle\_attrs is valid, otherwise an error is returned

3.3.3.13 `globus_result_t globus_gsi_proxy_handle_attrs_set_key_gen_callback(globus\_gsi\_proxy\_handle\_attrs\_t handle_attrs, void (*callback)(int, int, void *)))`

Set the public/private key generation callback that provides status during the generation of the keys.

Parameters:

`handle_attrs` The handle\_attrs to get the callback from  
`callback` The callback from the handle attributes

Returns:

GLOBUS\_SUCCESS if the handle\_attrs is valid, otherwise an error is returned

## 3.4 Proxy Operations

Initiate a proxy operation.

Create Request

- `globus_result_t globus\_gsi\_proxy\_create\_req(globus\_gsi\_proxy\_handle\_t handle, BIO output_bio)`

Inquire Request

- `globus_result_t globus\_gsi\_proxy\_inquire\_req(globus\_gsi\_proxy\_handle\_t handle, BIO input_bio)`

## Resign Certificate

- `globus_result_t globus_gsi_proxy_resign_certificate(globus_gsi_proxy_handle_t handle, globus_gsi_cred_handle_t issuer_credential, globus_gsi_cred_handle_t peer_credential, globus_gsi_cred_handle_t assigned_credential)`

## Sign Request

- `globus_result_t globus_gsi_proxy_sign_request(globus_gsi_proxy_handle_t handle, globus_gsi_cred_handle_t issuer_credential, BIO output_bio)`

## Create Signed

- `globus_result_t globus_gsi_proxy_create_signed(globus_gsi_proxy_handle_t handle, globus_gsi_cred_handle_t issuer, globus_gsi_cred_handle_t proxy_credential)`

## Assemble credential

- `globus_result_t globus_gsi_proxy_assemble_credentials(globus_gsi_proxy_handle_t handle, globus_gsi_cred_handle_t proxy_credential, BIO input_bio)`

## 3.4.1 Detailed Description

Initiate a proxy operation.

This module contains the API functions for a user to request proxy request generation, proxy request inspection and proxy request signature.

## 3.4.2 Function Documentation

3.4.2.1 `globus_result_t globus_gsi_proxy_create_request(globus_gsi_proxy_handle_t handle, BIO output_bio)`

## Create a proxy credential request

This function creates a proxy credential request, ie. a unsigned certificate and the corresponding private key, based on the handle that is passed in. The public part of the request is written to the BIO supplied in the `output_bio` parameter. After the request is written, the PROXYCERTINFO extension contained in the handle is written to the BIO. The proxy handle is updated with the private key.

## Parameters:

`handle` A GSI Proxy handle to use for the request operation.

`output_bio` A BIO to write the resulting request structure to.

## Returns:

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

3.4.2.2 `globus_result_t globus_gsi_proxy_inquire_req(globus\_gsi\_proxy\_handle\_t handle, BIO input_bio)`

Inquire a proxy credential request

This function reads the public part of a proxy credential request from `input_bio` and if the request contains a ProxyCertInfo extension, updates the handle with the information contained in the extension.

Parameters:

- `handle` A GSI Proxy handle to use for the inquire operation.
- `input_bio` A BIO to read a request structure from.

Returns:

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

3.4.2.3 `globus_result_t globus_gsi_proxy_resign_cert(globus\_gsi\_proxy\_handle\_t handle, globus_gsi_cred_handle_t issuer_credential, globus_gsi_cred_handle_t peer_credential, globus_gsi_cred_handle_t resigned_credential)`

Resign a existing certi cate into a proxy

This function use the public key in a existing certi cate to create a new proxy certi cate chained to the issuers credentials. This operation will add a ProxyCertInfo extension to the proxy certi cate if values contained in the extension are speci ed in the handle.

Parameters:

- `handle` A GSI Proxy handle to use for the signing operation.
- `issuer_credential` The credential structure to be used for signing the proxy certi cate.
- `peer_credential` The credential structure that contains the certi cate to be resigned.
- `resigned_credential` A credential structure that upon return will contain the resigned certi cate and associated certi cate chain.

Returns:

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

3.4.2.4 `globus_result_t globus_gsi_proxy_sign_req(globus\_gsi\_proxy\_handle\_t handle, globus_gsi_cred_handle_t issuer_credential, BIO output_bio)`

Sign a proxy certi cate request

This function signs the public part of a proxy credential request, i.e. the unsigned certi cate, previously read by `inquire req` using the supplied `issuer_credential`. This operation will add a ProxyCertInfo extension to the proxy certi cate if values contained in the extension are speci ed in the handle. The resulting signed certi cate is written to the `output_bio`.

Parameters:

- `handle` A GSI Proxy handle to use for the signing operation.
- `issuer_credential` The credential structure to be used for signing the proxy certi cate.
- `output_bio` A BIO to write the resulting certi cate to.

Returns:

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

3.4.2.5 `globus_result_t globus_gsi_proxy_create_signed(globus\_gsi\_proxy\_handle\_t handle, globus\_gsi\_cred\_handle\_t issuer, globus\_gsi\_cred\_handle\_t proxy_credential)`

Create Signed Proxy Certificate

Parameters:

`handle` The proxy handle used to create and sign the proxy certificate

`issuer` The issuing credential, used for signing the proxy certificate

`proxy_credential` The new proxy credential, containing the signed cert, private key, etc.

Returns:

GLOBUS\_SUCCESS if no error occurred, an error object ID otherwise

3.4.2.6 `globus_result_t globus_gsi_proxy_assemble_cred(globus\_gsi\_proxy\_handle\_t handle, globus\_gsi\_cred\_handle\_t proxy_credential, BIO input_bio)`

Assemble a proxy credential

This function assembles a proxy credential. It reads a signed proxy certificate and a associated certificate chain from the `input_bio` and combines them with a private key previously generated by a call to `globus_gsi_proxy_create_req`. The resulting credential is then returned through the `proxy_credential` parameter.

Parameters:

`handle` A GSI Proxy handle to use for the assemble operation.

`proxy_credential` This parameter will contain the assembled credential upon successful return.

`input_bio` A BIO to read a signed certificate and corresponding certificate chain from.

Returns:

GLOBUS\_SUCCESS if no error occurred, an error object ID otherwise

## 3.5 Proxy Constants

Enumerations

- `enum globus\_gsi\_proxy\_error\_t`
  - `GLOBUS_GSI_PROXY_ERROR_SUCCESS` 0,
  - `GLOBUS_GSI_PROXY_ERROR_WITH_HANDLE` 1,
  - `GLOBUS_GSI_PROXY_ERROR_WITH_HANDLE_ATTRS` 2,
  - `GLOBUS_GSI_PROXY_ERROR_WITH_PROXYCERTINFO` 3,
  - `GLOBUS_GSI_PROXY_ERROR_WITH_PROXYPOLICY` 4,
  - `GLOBUS_GSI_PROXY_ERROR_WITH_PATHLENGTH` 5,
  - `GLOBUS_GSI_PROXY_ERROR_WITH_X509_REQ` 6,
  - `GLOBUS_GSI_PROXY_ERROR_WITH_X509` 7,
  - `GLOBUS_GSI_PROXY_ERROR_WITH_X509_EXTENSIONS` 8,
  - `GLOBUS_GSI_PROXY_ERROR_WITH_PRIVATE_KEY` 9,
  - `GLOBUS_GSI_PROXY_ERROR_WITH_BIO` 10,
  - `GLOBUS_GSI_PROXY_ERROR_WITH_CREDENTIAL` 11,

```

GLOBUS_GSI_PROXY_ERROR_WITH_CRED_HANDLE=12,
GLOBUS_GSI_PROXY_ERROR_WITH_CRED_HANDLE_ATTRS=13,
GLOBUS_GSI_PROXY_ERROR_ERRNO=14,
GLOBUS_GSI_PROXY_ERROR_SETTING_HANDLE_TYPE=15,
GLOBUS_GSI_PROXY_INVALID_PARAMETER=16,
GLOBUS_GSI_PROXY_ERROR_SIGNING=17,
GLOBUS_GSI_PROXY_ERROR_LAST=18 }

```

### 3.5.1 Enumeration Type Documentation

#### 3.5.1.1 enumglobus\_gsi\_proxy\_error\_t

Proxy Error codes.

Enumeration values:

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

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_HANDLE Invalid proxy handle state.

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_HANDLE\_ATTRS Invalid proxy handle attributes state.

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_PROXYCERTINFO Error with ASN.1 proxycertinfo structure.

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_PROXYPOLICY Error with ASN.1 proxypolicy structure.

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_PATHLENGTH Error with proxy path length.

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_X509\_REQUEST Error with the X.509 request structure.

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_X509 Error with X.509 structure.

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_X509\_EXTENSIONS Error with X.509 extensions.

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_PRIVATE\_KEY Error with private key.

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_BIO Error with OpenSSL's BIO handle.

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_CREDENTIAL Error with credential.

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_CRED\_HANDLE Error with credential handle.

GLOBUS\_GSI\_PROXY\_ERROR\_WITH\_CRED\_HANDLE\_ATTRS Error with credential handle attributes.

GLOBUS\_GSI\_PROXY\_ERROR\_ERRNO System error.

GLOBUS\_GSI\_PROXY\_ERROR\_SETTING\_HANDLE\_TYPE Unable to set proxy type.

GLOBUS\_GSI\_PROXY\_INVALID\_PARAMETER Invalid function parameter.

GLOBUS\_GSI\_PROXY\_ERROR\_SIGNING A error occurred while signing the proxy certificate.

GLOBUS\_GSI\_PROXY\_ERROR\_LAST Last marker - never used.

## Index

Activation, [1](#)

globus\_gsi\_proxy\_activation

[GLOBUS\\_GSI\\_PROXY\\_MODULE2](#)

globus\_gsi\_proxy\_assemble\_cred

[globus\\_gsi\\_proxy\\_operation31](#)

globus\_gsi\_proxy\_constants

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_ERRNO22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_LAST22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_SETTING\\_-  
HANDLE\\_TYPE,22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_SIGNING,  
22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_SUCCESS,  
22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_BIO,  
22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
CRED\\_HANDLE,22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
CRED\\_HANDLE\\_ATTRS,22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
CREDENTIAL,22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
HANDLE,22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
HANDLE\\_ATTRS,22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
PATHLENGTH,22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
PRIVATE\\_KEY,22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
PROXYCERTINFO,22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
PROXYPOLICY,22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
X509,22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
X509\\_EXTENSIONS22](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
X509\\_REQ,22](#)

[GLOBUS\\_GSI\\_PROXY\\_INVALID\\_-  
PARAMETER,22](#)

globus\_gsi\_proxy\_constants

[globus\\_gsi\\_proxy\\_error\\_22](#)

globus\_gsi\_proxy\_create\_req

[globus\\_gsi\\_proxy\\_operation19](#)

globus\_gsi\_proxy\_create\_signed

[globus\\_gsi\\_proxy\\_operation30](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_ERRNO](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_LAST](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_SETTING\\_-  
HANDLE\\_TYPE](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_SIGNING](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_SUCCESS](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[globus\\_gsi\\_proxy\\_error\\_t](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_BIO](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_CRED\\_-  
HANDLE](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_CRED\\_-  
HANDLE\\_ATTRS](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
CREDENTIAL](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_HANDLE](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
HANDLE\\_ATTRS](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
PATHLENGTH](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
PRIVATE\\_KEY](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
PROXYCERTINFO](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_-  
PROXYPOLICY](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_X509](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_X509\\_-  
EXTENSIONS](#)

[globus\\_gsi\\_proxy\\_constant32](#)

[GLOBUS\\_GSI\\_PROXY\\_ERROR\\_WITH\\_X509\\_-  
REQ](#)

[globus\\_gsi\\_proxy\\_constant32](#)

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

[globus\\_gsi\\_proxy\\_handle\\_add\\_extensions,](#)

[globus\\_gsi\\_proxy\\_handle\\_clear\\_cert\\_info,](#)



```

globus_gsi_proxy_handle_destroy,
globus_gsi_proxy_handle_get_clock_skew_
allowable,8
globus_gsi_proxy_handle_get_common_name,
globus_gsi_proxy_handle_get_extensions,11
globus_gsi_proxy_handle_get_init_prime,
globus_gsi_proxy_handle_get_key_gen_
callback,9
globus_gsi_proxy_handle_get_keybits,8
globus_gsi_proxy_handle_get_pathlen,
globus_gsi_proxy_handle_get_policy,11
globus_gsi_proxy_handle_get_private_key,
globus_gsi_proxy_handle_get_proxy_cert_info,
7
globus_gsi_proxy_handle_get_req,
globus_gsi_proxy_handle_get_signing_
algorithm,8
globus_gsi_proxy_handle_get_time_val,
globus_gsi_proxy_handle_get_type,
globus_gsi_proxy_handle_init,
globus_gsi_proxy_handle_set_common_name,
13
globus_gsi_proxy_handle_set_extensions,
globus_gsi_proxy_handle_set_is_limited,
globus_gsi_proxy_handle_set_pathlen,
globus_gsi_proxy_handle_set_policy,
globus_gsi_proxy_handle_set_private_key,
globus_gsi_proxy_handle_set_proxy_cert_info,
12
globus_gsi_proxy_handle_set_req,
globus_gsi_proxy_handle_set_time_val,
globus_gsi_proxy_handle_set_type,
globus_gsi_proxy_handle,4,
globus_gsi_proxy_is_limited,3
globus_gsi_proxy_handle_add_extension
globus_gsi_proxy_handle,6,
globus_gsi_proxy_handle_attrs
globus_gsi_proxy_handle_attrs_copy,
globus_gsi_proxy_handle_attrs_destroy,
globus_gsi_proxy_handle_attrs_get_clock_
skew_allowable,18
globus_gsi_proxy_handle_attrs_get_init_prime,
17
globus_gsi_proxy_handle_attrs_get_key_gen_
callback,16
globus_gsi_proxy_handle_attrs_get_keybits,
globus_gsi_proxy_handle_attrs_get_signing_
algorithm,17
globus_gsi_proxy_handle_attrs_init,
globus_gsi_proxy_handle_attrs_set_clock_
skew_allowable,16
globus_gsi_proxy_handle_attrs_set_init_prime,
15
globus_gsi_proxy_handle_attrs_set_key_gen_
callback,18
globus_gsi_proxy_handle_attrs_set_keybits,
globus_gsi_proxy_handle_attrs_set_signing_
algorithm,16
globus_gsi_proxy_handle_attrs,15
globus_gsi_proxy_handle_attrs_copy,
globus_gsi_proxy_handle_attrs_destroy,
globus_gsi_proxy_handle_attrs,
globus_gsi_proxy_handle_attrs_get_clock_skew_
allowable,
globus_gsi_proxy_handle_attrs,
globus_gsi_proxy_handle_attrs_get_init_prime,
globus_gsi_proxy_handle_attrs,
globus_gsi_proxy_handle_attrs_get_key_gen_
callback,
globus_gsi_proxy_handle_attrs_get_keybits,
globus_gsi_proxy_handle_attrs_get_signing_
algorithm,
globus_gsi_proxy_handle_attrs,
globus_gsi_proxy_handle_attrs_init,
globus_gsi_proxy_handle_attrs,
globus_gsi_proxy_handle_attrs_set_clock_skew_
allowable,
globus_gsi_proxy_handle_attrs_set_init_prime,
globus_gsi_proxy_handle_attrs_set_init_prime,
globus_gsi_proxy_handle_attrs_set_key_gen_
callback,
globus_gsi_proxy_handle_attrs_set_keybits,
globus_gsi_proxy_handle_attrs_set_signing_
algorithm,
globus_gsi_proxy_handle_attrs,
globus_gsi_proxy_handle_attrs_t,
globus_gsi_proxy_handle_attrs,
globus_gsi_proxy_handle_clear_cert_info,
globus_gsi_proxy_handle,
globus_gsi_proxy_handle_destroy,
globus_gsi_proxy_handle,
globus_gsi_proxy_handle_get_clock_skew_allowable,
globus_gsi_proxy_handle,
globus_gsi_proxy_handle_get_common_name,
globus_gsi_proxy_handle,
globus_gsi_proxy_handle_get_extensions,
globus_gsi_proxy_handle,
globus_gsi_proxy_handle_get_init_prime,
globus_gsi_proxy_handle,
globus_gsi_proxy_handle_get_key_gen_callback,

```

[globus\\_gsi\\_proxy\\_handle](#)[16](#),  
[globus\\_gsi\\_proxy\\_handle\\_get\\_keybits](#)  
[globus\\_gsi\\_proxy\\_handle](#)[16](#),  
[globus\\_gsi\\_proxy\\_handle\\_get\\_pathlen](#)  
[globus\\_gsi\\_proxy\\_handle](#)[42](#)  
[globus\\_gsi\\_proxy\\_handle\\_get\\_policy](#)  
[globus\\_gsi\\_proxy\\_handle](#)[41](#)  
[globus\\_gsi\\_proxy\\_handle\\_get\\_private\\_key](#)  
[globus\\_gsi\\_proxy\\_handle](#)[16](#),  
[globus\\_gsi\\_proxy\\_handle\\_get\\_proxy\\_cert\\_info](#)  
[globus\\_gsi\\_proxy\\_handle](#)[7](#),  
[globus\\_gsi\\_proxy\\_handle\\_get\\_req](#)  
[globus\\_gsi\\_proxy\\_handle](#)[16](#),  
[globus\\_gsi\\_proxy\\_handle\\_get\\_signing\\_algorithm](#)  
[globus\\_gsi\\_proxy\\_handle](#)[16](#),  
[globus\\_gsi\\_proxy\\_handle\\_get\\_time\\_valid](#)  
[globus\\_gsi\\_proxy\\_handle](#)[7](#),  
[globus\\_gsi\\_proxy\\_handle\\_get\\_type](#)  
[globus\\_gsi\\_proxy\\_handle](#)[16](#),  
[globus\\_gsi\\_proxy\\_handle\\_init](#)  
[globus\\_gsi\\_proxy\\_handle](#)[16](#),  
[globus\\_gsi\\_proxy\\_handle\\_set\\_common\\_name](#)  
[globus\\_gsi\\_proxy\\_handle](#)[43](#)  
[globus\\_gsi\\_proxy\\_handle\\_set\\_extensions](#)  
[globus\\_gsi\\_proxy\\_handle](#)[41](#)  
[globus\\_gsi\\_proxy\\_handle\\_set\\_is\\_limited](#)  
[globus\\_gsi\\_proxy\\_handle](#)[16](#),  
[globus\\_gsi\\_proxy\\_handle\\_set\\_pathlen](#)  
[globus\\_gsi\\_proxy\\_handle](#)[7](#),  
[globus\\_gsi\\_proxy\\_handle\\_set\\_policy](#)  
[globus\\_gsi\\_proxy\\_handle](#)[16](#),  
[globus\\_gsi\\_proxy\\_handle\\_set\\_private\\_key](#)  
[globus\\_gsi\\_proxy\\_handle](#)[40](#)  
[globus\\_gsi\\_proxy\\_handle\\_set\\_proxy\\_cert\\_info](#)  
[globus\\_gsi\\_proxy\\_handle](#)[42](#)  
[globus\\_gsi\\_proxy\\_handle\\_set\\_req](#)  
[globus\\_gsi\\_proxy\\_handle](#)[40](#)  
[globus\\_gsi\\_proxy\\_handle\\_set\\_time\\_valid](#)  
[globus\\_gsi\\_proxy\\_handle](#)[42](#)  
[globus\\_gsi\\_proxy\\_handle\\_set\\_type](#)  
[globus\\_gsi\\_proxy\\_handle](#)[40](#)  
[globus\\_gsi\\_proxy\\_handle\\_t](#)  
[globus\\_gsi\\_proxy\\_handle](#)[4](#),  
[globus\\_gsi\\_proxy\\_inquire\\_req](#)  
[globus\\_gsi\\_proxy\\_operations](#)[19](#)  
[GLOBUS\\_GSI\\_PROXY\\_INVALID\\_PARAMETER](#)  
[globus\\_gsi\\_proxy\\_constant](#)[82](#)  
[globus\\_gsi\\_proxy\\_is\\_limited](#)  
[globus\\_gsi\\_proxy\\_handle](#)[43](#)  
[GLOBUS\\_GSI\\_PROXY\\_MODULE](#)  
[globus\\_gsi\\_proxy\\_activation](#)[2](#),  
[globus\\_gsi\\_proxy\\_operations](#)  
[globus\\_gsi\\_proxy\\_assemble\\_cr](#)[21](#),  
[globus\\_gsi\\_proxy\\_create\\_req](#)[19](#)  
[globus\\_gsi\\_proxy\\_create\\_signed](#)[20](#),  
[globus\\_gsi\\_proxy\\_inquire\\_req](#)[19](#)  
[globus\\_gsi\\_proxy\\_resign\\_cert](#)[20](#)  
[globus\\_gsi\\_proxy\\_sign\\_req](#)[20](#)  
[globus\\_gsi\\_proxy\\_resign\\_cert](#)  
[globus\\_gsi\\_proxy\\_operations](#)[80](#)  
[globus\\_gsi\\_proxy\\_sign\\_req](#)  
[globus\\_gsi\\_proxy\\_operations](#)[80](#)  
  
[Handle Attributes](#)[13](#)  
[Handle Management](#)[2](#),  
  
[Proxy Constants](#)[21](#)  
[Proxy Operations](#)[18](#)