

globus gsi callback

4.1

Generated by Doxygen 1.7.6.1

Tue Feb 21 2012 14:55:46

## Contents

<b>1</b>	<b>Globus GSI Callback</b>	<b>1</b>
<b>2</b>	<b>Deprecated List</b>	<b>1</b>
<b>3</b>	<b>Module Index</b>	<b>2</b>
3.1	Modules . . . . .	2
<b>4</b>	<b>Module Documentation</b>	<b>2</b>
4.1	Activation . . . . .	2
4.1.1	Detailed Description . . . . .	2
4.1.2	Define Documentation . . . . .	2
4.2	Callback Functions . . . . .	3
4.2.1	Detailed Description . . . . .	3
4.2.2	Typedef Documentation . . . . .	3
4.2.3	Function Documentation . . . . .	3
4.3	Callback Data Functions . . . . .	6
4.3.1	Detailed Description . . . . .	7
4.3.2	Typedef Documentation . . . . .	7
4.3.3	Function Documentation . . . . .	8
4.4	GSI Callback Constants . . . . .	15
4.4.1	Enumeration Type Documentation . . . . .	15

## 1 Globus GSI Callback

The Globus GSI Callback library. This library contains functions that extend OpenSSL path validation.

- **Activation** (p. 2)
- **Callback Functions** (p. 3)
- **Callback Data Functions** (p. 6)

## 2 Deprecated List

**Global `globus_gsi_callback_get_multiple_limited_proxy_ok` (p. 10) (`globus_gsi_callback_data_t` `callback_data`, `int *multiple_limited_proxy_ok`)**

This function always returns true now. It will be removed in the next release.

**Global `globus_gsi_callback_set_multiple_limited_proxy_ok` (p. 11) (`globus_gsi_callback_data_t` `callback_data`, `int multiple_limited_proxy_ok`)**

This function has been turned into a no-op. It will be removed in the next release.

## 3 Module Index

### 3.1 Modules

Here is a list of all modules:

<b>Activation</b>	<b>2</b>
<b>Callback Functions</b>	<b>3</b>
<b>Callback Data Functions</b>	<b>6</b>
<b>GSI Callback Constants</b>	<b>15</b>

## 4 Module Documentation

### 4.1 Activation

Defines

- `#define GLOBUS_GSI_CALLBACK_MODULE`

#### 4.1.1 Detailed Description

Globus GSI Callback uses standard Globus module activation and deactivation. Before any Globus GSI Callback functions are called, the following function must be called:

```
globus_module_activate(GLOBUS_GSI_CALLBACK_MODULE)
```

This function returns `GLOBUS_SUCCESS` if Globus GSI Callback was successfully initialized, and you are therefore allowed to subsequently call Globus GSI Callback 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 Callback, the following function must be called:

```
globus_module_deactivate(GLOBUS_GSI_CALLBACK_MODULE)
```

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

#### 4.1.2 Define Documentation

##### 4.1.2.1 `#define GLOBUS_GSI_CALLBACK_MODULE`

Module descriptor.

## 4.2 Callback Functions

### Typedefs

- typedef int(\* **globus\_gsi\_extension\_callback\_t**)(**globus\_gsi\_callback\_data\_t** callback\_data, X509\_EXTENSION \*extension)

### Get callback data index from X509\_STORE

- **globus\_result\_t globus\_gsi\_callback\_get\_X509\_STORE\_callback\_data\_index** (int \*index)

### Get callback data index from SSL structure

- **globus\_result\_t globus\_gsi\_callback\_get\_SSL\_callback\_data\_index** (int \*index)

### Certificate verify wrapper

- int **globus\_gsi\_callback\_X509\_verify\_cert** (X509\_STORE\_CTX \*context, void \*arg)

### Independent path validation callback.

- int **globus\_gsi\_callback\_create\_proxy\_callback** (int preverify\_ok, X509\_STORE\_CTX \*x509\_context)

### SSL path validation callback.

- int **globus\_gsi\_callback\_handshake\_callback** (int preverify\_ok, X509\_STORE\_CTX \*x509\_context)

### OpenSSL X509\_check\_issued() wrapper

- int **globus\_gsi\_callback\_check\_issued** (X509\_STORE\_CTX \*context, X509 \*cert, X509 \*issuer)

### 4.2.1 Detailed Description

Functions that plug into various plug points in the OpenSSL path validation mechanism. These functions add CRL checking, X509 Extension handling and proxy validation.

### 4.2.2 Typedef Documentation

#### 4.2.2.1 typedef int(\* **globus\_gsi\_extension\_callback\_t**)(**globus\_gsi\_callback\_data\_t** callback\_data, X509\_EXTENSION \*extension)

Typedef for a callback that may be registered for dealing with unhandled X.509 extension.

### 4.2.3 Function Documentation

#### 4.2.3.1 **globus\_result\_t globus\_gsi\_callback\_get\_X509\_STORE\_callback\_data\_index** ( int \* *index* )

Retrieve or create the index for our callback data structure in the X509\_STORE.

#### Parameters

<i>index</i>	Will contain the index upon return
--------------	------------------------------------

#### Returns

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

#### 4.2.3.2 globus\_result\_t globus\_gsi\_callback\_get\_SSL\_callback\_data\_index ( int \* *index* )

Retrieve or create the index for our callback data structure in the SSL structure.

#### Parameters

<i>index</i>	Will contain the index upon return
--------------	------------------------------------

#### Returns

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

#### 4.2.3.3 int globus\_gsi\_callback\_X509\_verify\_cert ( X509\_STORE\_CTX \* *context*, void \* *arg* )

This function wraps the OpenSSL certificate verification callback for the purpose of replacing the standard issuer check with one that deals with proxy certificates.

Should be used with SSL\_CTX\_set\_cert\_verify\_callback()

#### Parameters

<i>context</i>	The X509_STORE_CTX for which to register the callback.
<i>arg</i>	Arguments to the callback. Currently ignored.

#### Returns

1 on success 0 on failure

#### 4.2.3.4 int globus\_gsi\_callback\_create\_proxy\_callback ( int *preverify\_ok*, X509\_STORE\_CTX \* *x509\_context* )

This function provides a path validation callback for validation outside of a SSL session.

It should be used in X509\_STORE\_set\_verify\_cb\_func().

#### Parameters

<i>preverify_ok</i>	Communicates the result of default validation steps performed by OpenSSL
<i>x509_context</i>	The validation state object

#### Returns

1 on success 0 on failure

#### 4.2.3.5 int globus\_gsi\_callback\_handshake\_callback ( int *preverify\_ok*, X509\_STORE\_CTX \* *x509\_context* )

This function provides a path validation callback for the validation part of establishing a SSL session.

It handles proxy certificates, X509 Extensions and CRL checking. It should be used in SSL\_CTX\_set\_verify().

#### Parameters

<i>preverify_ok</i>	Communicates the result of default validation steps performed by OpenSSL
<i>x509_context</i>	The validation state object.

#### Returns

1 on success 0 on failure

#### 4.2.3.6 int globus\_gsi\_callback\_check\_issued ( X509\_STORE\_CTX \* *context*, X509 \* *cert*, X509 \* *issuer* )

This function wraps the OpenSSL X509\_check\_issued() call and catches the error caused by the fact that a proxy certificate issuer may not have to have the correct KeyUsage fields set.

#### Parameters

<i>context</i>	The validation state object.
<i>cert</i>	The certificate to check
<i>issuer</i>	The issuer certificate to check

#### Returns

1 on success 0 on failure

## 4.3 Callback Data Functions

### Typedefs

- typedef struct globus\_l\_gsi\_callback\_data\_s \* **globus\_gsi\_callback\_data\_t**

### Initializing and destroying a callback data structure

- globus\_result\_t **globus\_gsi\_callback\_data\_init** (globus\_gsi\_callback\_data\_t \*callback\_data)
- globus\_result\_t **globus\_gsi\_callback\_data\_destroy** (globus\_gsi\_callback\_data\_t callback\_data)

### Copying a callback data structure

- globus\_result\_t **globus\_gsi\_callback\_data\_copy** (globus\_gsi\_callback\_data\_t source, **globus\_gsi\_callback\_data\_t** \*dest)

### Getting and setting the certificate chain depth

- globus\_result\_t **globus\_gsi\_callback\_get\_cert\_depth** (globus\_gsi\_callback\_data\_t callback\_data, int \*cert\_depth)
- globus\_result\_t **globus\_gsi\_callback\_set\_cert\_depth** (globus\_gsi\_callback\_data\_t callback\_data, int cert\_depth)

### Getting and setting the "proxy chain" depth

- globus\_result\_t **globus\_gsi\_callback\_get\_proxy\_depth** (globus\_gsi\_callback\_data\_t callback\_data, int \*proxy\_depth)
- globus\_result\_t **globus\_gsi\_callback\_set\_proxy\_depth** (globus\_gsi\_callback\_data\_t callback\_data, int proxy\_depth)

### Getting and setting the certificate type

- globus\_result\_t **globus\_gsi\_callback\_get\_cert\_type** (globus\_gsi\_callback\_data\_t callback\_data, globus\_gsi\_cert\_utils\_cert\_type\_t \*cert\_type)
- globus\_result\_t **globus\_gsi\_callback\_set\_cert\_type** (globus\_gsi\_callback\_data\_t callback\_data, globus\_gsi\_cert\_utils\_cert\_type\_t cert\_type)

### Getting and setting the certificate chain

- globus\_result\_t **globus\_gsi\_callback\_get\_cert\_chain** (globus\_gsi\_callback\_data\_t callback\_data, STACK\_OF(X509)\*\*cert\_chain)
- globus\_result\_t **globus\_gsi\_callback\_set\_cert\_chain** (globus\_gsi\_callback\_data\_t callback\_data, STACK\_OF(X509)\*cert\_chain)

### Getting and setting the limited proxy handling setting

- globus\_result\_t **globus\_gsi\_callback\_get\_multiple\_limited\_proxy\_ok** (globus\_gsi\_callback\_data\_t callback\_data, int \*multiple\_limited\_proxy\_ok)
- globus\_result\_t **globus\_gsi\_callback\_set\_multiple\_limited\_proxy\_ok** (globus\_gsi\_callback\_data\_t callback\_data, int multiple\_limited\_proxy\_ok)

Getting and setting a set of X.509 extension OIDs.

- globus\_result\_t **globus\_gsi\_callback\_get\_extension\_oids** (globus\_gsi\_callback\_data\_t callback\_data, void \*\*extension\_oids)
- globus\_result\_t **globus\_gsi\_callback\_set\_extension\_oids** (globus\_gsi\_callback\_data\_t callback\_data, void \*extension\_oids)

Getting and setting the trusted certificate directory

- globus\_result\_t **globus\_gsi\_callback\_get\_cert\_dir** (globus\_gsi\_callback\_data\_t callback\_data, char \*\*cert\_dir)
- globus\_result\_t **globus\_gsi\_callback\_set\_cert\_dir** (globus\_gsi\_callback\_data\_t callback\_data, char \*cert\_dir)

Getting and setting the callback to be called for unknown X.509 extensions

- globus\_result\_t **globus\_gsi\_callback\_get\_extension\_cb** (globus\_gsi\_callback\_data\_t callback\_data, globus\_gsi\_extension\_callback\_t \*extension\_cb)
- globus\_result\_t **globus\_gsi\_callback\_set\_extension\_cb** (globus\_gsi\_callback\_data\_t callback\_data, globus\_gsi\_extension\_callback\_t extension\_cb)

Getting and setting the error status

- globus\_result\_t **globus\_gsi\_callback\_get\_error** (globus\_gsi\_callback\_data\_t callback\_data, globus\_result\_t \*error)
- globus\_result\_t **globus\_gsi\_callback\_set\_error** (globus\_gsi\_callback\_data\_t callback\_data, globus\_result\_t error)

Getting and setting the check self-signed policy flag

- globus\_result\_t **globus\_gsi\_callback\_get\_check\_policy\_for\_self\_signed\_certs** (globus\_gsi\_callback\_data\_t callback\_data, globus\_bool\_t \*check)
- globus\_result\_t **globus\_gsi\_callback\_set\_check\_policy\_for\_self\_signed\_certs** (globus\_gsi\_callback\_data\_t callback\_data, globus\_bool\_t check)

Getting and setting the allow missing signing policy flag

- globus\_result\_t **globus\_gsi\_callback\_get\_allow\_missing\_signing\_policy** (globus\_gsi\_callback\_data\_t callback\_data, globus\_bool\_t \*allow)
- globus\_result\_t **globus\_gsi\_callback\_set\_allow\_missing\_signing\_policy** (globus\_gsi\_callback\_data\_t callback\_data, globus\_bool\_t allow)

#### 4.3.1 Detailed Description

Functions that deal with the data structure that contains state associated with the path validation callback.

#### 4.3.2 Typedef Documentation

##### 4.3.2.1 typedef struct globus\_l\_gsi\_callback\_data\_s\* globus\_gsi\_callback\_data\_t

Callback data typedef.



### 4.3.3 Function Documentation

#### 4.3.3.1 `globus_result_t globus_gsi_callback_data_init ( globus_gsi_callback_data_t * callback_data )`

This function initializes a `globus_gsi_callback_data_t`.

##### Parameters

<i>callback_data</i>	Reference to the structure to be initialized
----------------------	--

##### Returns

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

#### 4.3.3.2 `globus_result_t globus_gsi_callback_data_destroy ( globus_gsi_callback_data_t callback_data )`

This function destroys a `globus_gsi_callback_data_t`.

##### Parameters

<i>callback_data</i>	The structure to be destroyed
----------------------	-------------------------------

##### Returns

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

#### 4.3.3.3 `globus_result_t globus_gsi_callback_data_copy ( globus_gsi_callback_data_t source, globus_gsi_callback_data_t * dest )`

This function copies a `globus_gsi_callback_data_t`.

##### Parameters

<i>source</i>	The structure to be copied
<i>dest</i>	The destination of the copy

##### Returns

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

#### 4.3.3.4 `globus_result_t globus_gsi_callback_get_cert_depth ( globus_gsi_callback_data_t callback_data, int * cert_depth )`

This function returns the certificate chain depth.

##### Parameters

<i>callback_data</i>	The <code>globus_gsi_callback_data_t</code> to retrieve the depth from
<i>cert_depth</i>	The returned certificate chain depth

##### Returns

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

**4.3.3.5 globus\_result\_t globus\_gsi\_callback\_set\_cert\_depth ( globus\_gsi\_callback\_data\_t callback\_data, int cert\_depth )**

This function sets the certificate chain depth.

**Parameters**

<i>callback_data</i>	The globus_gsi_callback_data_t to retrieve the depth from
<i>cert_depth</i>	The certificate chain depth

**Returns**

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

**4.3.3.6 globus\_result\_t globus\_gsi\_callback\_get\_proxy\_depth ( globus\_gsi\_callback\_data\_t callback\_data, int \* proxy\_depth )**

This function returns the number of proxies in the certificate chain.

**Parameters**

<i>callback_data</i>	The globus_gsi_callback_data_t to retrieve the depth from
<i>proxy_depth</i>	The returned "proxy chain" depth

**Returns**

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

**4.3.3.7 globus\_result\_t globus\_gsi\_callback\_set\_proxy\_depth ( globus\_gsi\_callback\_data\_t callback\_data, int proxy\_depth )**

This function sets the number of proxies in the certificate chain.

**Parameters**

<i>callback_data</i>	The globus_gsi_callback_data_t to retrieve the depth from
<i>proxy_depth</i>	The "proxy chain" depth

**Returns**

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

**4.3.3.8 globus\_result\_t globus\_gsi\_callback\_get\_cert\_type ( globus\_gsi\_callback\_data\_t callback\_data, globus\_gsi\_cert\_utils.cert\_type\_t \* cert\_type )**

This function returns the certificate type of the certificate currently being processed.

**Parameters**

<i>callback_data</i>	The globus_gsi_callback_data_t to retrieve the certificate type from
<i>cert_type</i>	Variable containing the certificate type on return

#### Returns

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

**4.3.3.9 globus\_result\_t globus\_gsi\_callback\_set\_cert\_type ( globus\_gsi\_callback\_data\_t callback\_data, globus\_gsi\_cert\_utils\_cert\_type\_t cert\_type )**

This function sets the certificate type of the certificate currently being processed.

#### Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to set the certificate type on
<i>cert_type</i>	The certificate type

#### Returns

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

**4.3.3.10 globus\_result\_t globus\_gsi\_callback\_get\_cert\_chain ( globus\_gsi\_callback\_data\_t callback\_data, STACK\_OF(X509)\*\* cert\_chain )**

This function returns the certificate chain associated with the callback data.

#### Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to retrieve the certificate chain from.
<i>cert_chain</i>	Contains the certificate chain upon successful return

#### Returns

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

**4.3.3.11 globus\_result\_t globus\_gsi\_callback\_set\_cert\_chain ( globus\_gsi\_callback\_data\_t callback\_data, STACK\_OF(X509)\* cert\_chain )**

This function sets the certificate chain associated with the callback data.

#### Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to set the certificate chain on
<i>cert_chain</i>	The certificate chain

#### Returns

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

**4.3.3.12 globus\_result\_t globus\_gsi\_callback\_get\_multiple\_limited\_proxy\_ok ( globus\_gsi\_callback\_data\_t callback\_data, int \* multiple\_limited\_proxy\_ok )**

This function gets the value of the limited proxy handling setting.

This setting determines whether path validation will accept limited proxies that have been further delegated, ie certificate chains with a limited proxy followed by further proxies.

#### Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to get the limited proxy setting from
<i>multiple_limited-proxy_ok</i>	Contains the value of the setting upon successful return.

#### Returns

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

**Deprecated** This function always returns true now. It will be removed in the next release.

4.3.3.13 `globus_result_t globus_gsi_callback_set_multiple_limited_proxy_ok ( globus_gsi_callback_data_t callback_data, int multiple_limited_proxy_ok )`

This function sets the value of the limited proxy handling setting.

This setting determines whether path validation will accept limited proxies that have been further delegated, ie certificate chains with a limited proxy followed by further proxies.

#### Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to set the limited proxy setting on
<i>multiple_limited-proxy_ok</i>	The value of the setting

#### Returns

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

**Deprecated** This function has been turned into a no-op. It will be removed in the next release.

4.3.3.14 `globus_result_t globus_gsi_callback_get_extension_oids ( globus_gsi_callback_data_t callback_data, void ** extension_oids )`

This function gets a list of X.509 extension OIDs that may be used by the extensions callback to allow or disallow certain extensions.

#### Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to get the array of extension OIDs from.
<i>extension_oids</i>	Contains the list of extension OIDs upon successful return.

#### Returns

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

4.3.3.15 `globus_result_t globus_gsi_callback_set_extension_oids ( globus_gsi_callback_data_t callback_data, void * extension_oids )`

This function sets a list of X.509 extension OIDs that may be used by the extensions callback to allow or disallow certain extensions.

#### Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to get the array of extension OIDs from.
<i>extension_oids</i>	The list of extension OIDs

## Returns

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

**4.3.3.16** `globus_result_t globus_gsi_callback_get_cert_dir ( globus_gsi_callback_data_t callback_data, char ** cert_dir )`

This function gets the trusted certificate directory from the callback data.

## Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to get the trusted certificates directory from.
<i>cert_dir</i>	Contains the path to the trusted certificate directory upon successful return.

## Returns

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

**4.3.3.17** `globus_result_t globus_gsi_callback_set_cert_dir ( globus_gsi_callback_data_t callback_data, char * cert_dir )`

This function sets the trusted certificate directory on the callback data.

## Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to set the trusted certificates directory on.
<i>cert_dir</i>	The path to the trusted certificate directory

## Returns

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

**4.3.3.18** `globus_result_t globus_gsi_callback_get_extension_cb ( globus_gsi_callback_data_t callback_data, globus_gsi_extension_callback_t * extension_cb )`

This function gets the callback that is called for unknown X.509 extensions.

## Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to get the callback information from
<i>extension_cb</i>	Contains the extension callback upon successful return.

## Returns

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

**4.3.3.19** `globus_result_t globus_gsi_callback_set_extension_cb ( globus_gsi_callback_data_t callback_data, globus_gsi_extension_callback_t extension_cb )`

This function sets the callback that is called for unknown X.509 extensions.

## Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to set the callback information on
<i>extension_cb</i>	The extension callback

## Returns

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

**4.3.3.20** `globus_result_t globus_gsi_callback_get_error ( globus_gsi_callback_data_t callback_data, globus_result_t * error )`

This function gets the error status stored in the callback data.

## Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to get the error from
<i>error</i>	Contains the error upon successful return.

## Returns

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

**4.3.3.21** `globus_result_t globus_gsi_callback_set_error ( globus_gsi_callback_data_t callback_data, globus_result_t error )`

This function sets the error status stored in the callback data.

## Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to set the error on
<i>error</i>	The error

## Returns

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

**4.3.3.22** `globus_result_t globus_gsi_callback_get_check_policy_for_self_signed_certs ( globus_gsi_callback_data_t callback_data, globus_bool_t * check )`

This function gets the value of the "check policy for self-signed certificates flag" in the callback data.

If this is set then the CA signing policy for a self-signed certificate must include a policy line that allows it to sign itself.

## Parameters

<i>callback_data</i>	The globus_gsi_callback_data_t to get the error from
<i>check</i>	Contains the value of the flag upon successful return.

## Returns

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

## Since

Globus Toolkit 4.2.1

**4.3.3.23** `globus_result_t globus_gsi_callback_set_check_policy_for_self_signed_certs ( globus_gsi_callback_data_t callback_data, globus_bool_t check )`

This function sets the value of the "check policy for self-signed certificates flag" in the callback data.

If this is set then the CA signing policy for a self-signed certificate must include a policy line that allows it to sign itself.

#### Parameters

<i>callback_data</i>	The <code>globus_gsi_callback_data_t</code> to set the error on
<i>check</i>	New value of the flag

#### Returns

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

#### Since

Globus Toolkit 4.2.1

**4.3.3.24** `globus_result_t globus_gsi_callback_get_allow_missing_signing_policy ( globus_gsi_callback_data_t callback_data, globus_bool_t * allow )`

This function gets the value of the "allow missing signing policy" flag in the callback data.

If this is TRUE then the CA signing policy need not be present.

#### Parameters

<i>callback_data</i>	The <code>globus_gsi_callback_data_t</code> to get the error from
<i>allow</i>	Contains the value of the flag upon successful return.

#### Returns

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

#### Since

Globus Toolkit 5.2.0

**4.3.3.25** `globus_result_t globus_gsi_callback_set_allow_missing_signing_policy ( globus_gsi_callback_data_t callback_data, globus_bool_t allow )`

This function sets the value of the "allow missing signing policy" flag in the callback data.

If this is TRUE then the CA signing policy need not be present.

#### Parameters

<i>callback_data</i>	The <code>globus_gsi_callback_data_t</code> to set the error on
<i>allow</i>	New value of the flag

#### Returns

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

#### Since

Globus Toolkit 5.2.0

## 4.4 GSI Callback Constants

### Enumerations

- enum **globus\_gsi\_callback\_error\_t** { **GLOBUS\_GSI\_CALLBACK\_ERROR\_SUCCESS** = 0, **GLOBUS\_GSI\_CALLBACK\_ERROR\_VERIFY\_CRED** = 1, **GLOBUS\_GSI\_CALLBACK\_ERROR\_CERT\_NOT\_YET\_VALID** = 2, **GLOBUS\_GSI\_CALLBACK\_ERROR\_CANT\_GET\_LOCAL\_CA\_CERT** = 3, **GLOBUS\_GSI\_CALLBACK\_ERROR\_CERT\_HAS\_EXPIRED** = 4, **GLOBUS\_GSI\_CALLBACK\_ERROR\_INVALID\_PROXY** = 5, **GLOBUS\_GSI\_CALLBACK\_ERROR\_LIMITED\_PROXY** = 6, **GLOBUS\_GSI\_CALLBACK\_ERROR\_INVALID\_CRL** = 7, **GLOBUS\_GSI\_CALLBACK\_ERROR\_REVOKED\_CERT** = 8, **GLOBUS\_GSI\_CALLBACK\_ERROR\_MIXING\_DIFFERENT\_PROXY\_TYPES** = 9, **GLOBUS\_GSI\_CALLBACK\_ERROR\_WITH\_SIGNING\_POLICY** = 10, **GLOBUS\_GSI\_CALLBACK\_ERROR\_OLD\_GAA** = 11, **GLOBUS\_GSI\_CALLBACK\_ERROR\_CALLBACK\_DATA** = 12, **GLOBUS\_GSI\_CALLBACK\_ERROR\_ERRNO** = 13, **GLOBUS\_GSI\_CALLBACK\_ERROR\_CERT\_CHAIN** = 14, **GLOBUS\_GSI\_CALLBACK\_ERROR\_WITH\_CALLBACK\_DATA\_INDEX** = 15, **GLOBUS\_GSI\_CALLBACK\_ERROR\_PROXY\_PATH\_LENGTH\_EXCEEDED** = 16, **GLOBUS\_GSI\_CALLBACK\_ERROR\_LAST** = 18 }

### 4.4.1 Enumeration Type Documentation

#### 4.4.1.1 enum **globus\_gsi\_callback\_error\_t**

GSI Callback Error codes.

Enumerator:

**GLOBUS\_GSI\_CALLBACK\_ERROR\_SUCCESS** Success - never used.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_VERIFY\_CRED** Error verifying credential.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_CERT\_NOT\_YET\_VALID** The certificate is not yet valid.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_CANT\_GET\_LOCAL\_CA\_CERT** Unable to discover a local trusted CA for a given certificate.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_CERT\_HAS\_EXPIRED** The certificate has expired.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_INVALID\_PROXY** The proxy format is invalid.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_LIMITED\_PROXY** Limited proxies are not accepted.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_INVALID\_CRL** Invalid CRL.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_REVOKED\_CERT** The certificate has been revoked.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_MIXING\_DIFFERENT\_PROXY\_TYPES** The cert chain contains both legacy on rfc compliant proxies.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_WITH\_SIGNING\_POLICY** Could not verify certificate chain against the signing policy for the issuing CA.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_OLD\_GAA** OldGAA error.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_CALLBACK\_DATA** Error with the callback data structure.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_ERRNO** System error.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_CERT\_CHAIN** Error setting or getting the cert chain from callback data.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_WITH\_CALLBACK\_DATA\_INDEX** Error getting callback data index.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_PROXY\_PATH\_LENGTH\_EXCEEDED** Exceeded the path length specified in the proxy cert info extension.

**GLOBUS\_GSI\_CALLBACK\_ERROR\_LAST** Last marker - never used.