

# globus gss assist Reference Manual

## 4.0

Generated by Doxygen 1.3.9.1

Tue Aug 11 14:46:40 2009

# Contents

<a href="#">1 Globus GSI GSS Assist</a>	<a href="#">1</a>
<a href="#">2 globus gss assist Module Index</a>	<a href="#">1</a>
<a href="#">3 globus gss assist Module Documentation</a>	<a href="#">1</a>

## 1 Globus GSI GSS Assist

The GSS Assist code provides convenience functions for using the Globus GSS-API.

## 2 globus gss assist Module Index

### 2.1 globus gss assist Modules

Here is a list of all modules:

<b>Activation</b>	<a href="#">1</a>
<b>Utility Functions</b>	<a href="#">2</a>
<b>GSI GSS Assist Constants</b>	<a href="#">10</a>
<b>Security Token Transport</b>	<a href="#">12</a>

## 3 globus gss assist Module Documentation

### 3.1 Activation

Globus GSI GSS Assist uses standard Globus module activation and deactivation.

#### Defines

- `#define` [GLOBUS\\_GSI\\_GSS\\_ASSIST\\_MODULE](#)

#### 3.1.1 Detailed Description

Globus GSI GSS Assist uses standard Globus module activation and deactivation.

Before any Globus GSS Assist functions are called, the following function must be called:

```
globus_module_activate(GLOBUS_GSI_GSS_ASSIST_MODULE);
```

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

To deactivate Globus GSS Assist, the following function must be called:

```
globus_module_deactivate(GLOBUS_GSI_GSS_ASSIST_MODULE)
```

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

### 3.1.2 Define Documentation

#### 3.1.2.1 #define GLOBUS\_GSI\_GSS\_ASSIST\_MODULE

Module descriptor.

## 3.2 Utility Functions

Utility functions for GSSAPI.

### Accept Security Context

- OM\_uint32 [globus\\_gss\\_assist\\_accept\\_sec\\_context](#) (OM\_uint32 \*minor\_status, gss\_ctx\_id\_t \*context\_handle, const gss\_cred\_id\_t cred\_handle, char \*\*src\_name\_char, OM\_uint32 \*ret\_flags, int \*user\_to\_user\_flag, int \*token\_status, gss\_cred\_id\_t \*delegated\_cred\_handle, int(\*gss\_assist\_get\_token)(void \*, void \*\*, size\_t \*), void \*gss\_assist\_get\_context, int(\*gss\_assist\_send\_token)(void \*, void \*, size\_t), void \*gss\_assist\_send\_context)

### Accept Security Context Asynchronous

- OM\_uint32 [globus\\_gss\\_assist\\_accept\\_sec\\_context\\_async](#) (OM\_uint32 \*minor\_status, gss\_ctx\_id\_t \*context\_handle, const gss\_cred\_id\_t cred\_handle, char \*\*src\_name\_char, OM\_uint32 \*ret\_flags, int \*user\_to\_user\_flag, void \*input\_buffer, size\_t input\_buffer\_len, void \*\*output\_bufferp, size\_t \*output\_buffer\_lenp, gss\_cred\_id\_t \*delegated\_cred\_handle)

### Acquire Credential

- OM\_uint32 [globus\\_gss\\_assist\\_acquire\\_cred](#) (OM\_uint32 \*minor\_status, gss\_cred\_usage\_t cred\_usage, gss\_cred\_id\_t \*output\_cred\_handle)

### Acquire Credential Extension

- OM\_uint32 [globus\\_gss\\_assist\\_acquire\\_cred\\_ext](#) (OM\_uint32 \*minor\_status, char \*desired\_name\_char, OM\_uint32 time\_req, const gss\_OID\_set desired\_mechs, gss\_cred\_usage\_t cred\_usage, gss\_cred\_id\_t \*output\_cred\_handle, gss\_OID\_set \*actual\_mechs, OM\_uint32 \*time\_rec)

### Display Status

- OM\_uint32 [globus\\_gss\\_assist\\_display\\_status](#) (FILE \*fp, char \*comment, OM\_uint32 major\_status, OM\_uint32 minor\_status, int token\_status)

### Display Status String

- OM\_uint32 [globus\\_gss\\_assist\\_display\\_status\\_str](#) (char \*\*str, char \*comment, OM\_uint32 major\_status, OM\_uint32 minor\_status, int token\_status)

## Gridmap

- int [globus\\_gss\\_assist\\_gridmap](#) (char \*globusidp, char \*\*useridp)

## User OK

- int [globus\\_gss\\_assist\\_userok](#) (char \*globusid, char \*userid)

## Map Local User

- int [globus\\_gss\\_assist\\_map\\_local\\_user](#) (char \*local\_user, char \*\*globusidp)

## [NOHEADER]

- OM\_uint32 [globus\\_gss\\_assist\\_import\\_sec\\_context](#) (OM\_uint32 \*minor\_status, gss\_ctx\_id\_t \*context\_handle, int \*token\_status, int fdp, FILE \*fperr)

## Init Security Context

- OM\_uint32 [globus\\_gss\\_assist\\_init\\_sec\\_context](#) (OM\_uint32 \*minor\_status, const gss\_cred\_id\_t cred\_handle, gss\_ctx\_id\_t \*context\_handle, char \*target\_name\_char, OM\_uint32 req\_flags, OM\_uint32 \*ret\_flags, int \*token\_status, int(\*gss\_assist\_get\_token)(void \*, void \*\*, size\_t \*), void \*gss\_assist\_get\_context, int(\*gss\_assist\_send\_token)(void \*, void \*, size\_t), void \*gss\_assist\_send\_context)

## Init Security Context Async

- OM\_uint32 [globus\\_gss\\_assist\\_init\\_sec\\_context\\_async](#) (OM\_uint32 \*minor\_status, const gss\_cred\_id\_t cred\_handle, gss\_ctx\_id\_t \*context\_handle, char \*target\_name\_char, OM\_uint32 req\_flags, OM\_uint32 \*ret\_flags, void \*input\_buffer, size\_t input\_buffer\_len, void \*\*output\_bufferp, size\_t \*output\_buffer\_lenp)

## Will Handle Restrictions

- OM\_uint32 [globus\\_gss\\_assist\\_will\\_handle\\_restrictions](#) (OM\_uint32 \*minor\_status, gss\_ctx\_id\_t \*context\_handle)

## Get Unwrap

- OM\_uint32 [globus\\_gss\\_assist\\_get\\_unwrap](#) (OM\_uint32 \*minor\_status, const gss\_ctx\_id\_t context\_handle, char \*\*data, size\_t \*length, int \*token\_status, int(\*gss\_assist\_get\_token)(void \*, void \*\*, size\_t \*), void \*gss\_assist\_get\_context, FILE \*fperr)

## Wrap

- OM\_uint32 [globus\\_gss\\_assist\\_wrap\\_send](#) (OM\_uint32 \*minor\_status, const gss\_ctx\_id\_t context\_handle, char \*data, size\_t length, int \*token\_status, int(\*gss\_assist\_send\_token)(void \*, void \*, size\_t), void \*gss\_assist\_send\_context, FILE \*fperr)

## Defines

- #define [NI\\_MAXHOST](#) 255

### 3.2.1 Detailed Description

Utility functions for GSSAPI.

### 3.2.2 Define Documentation

#### 3.2.2.1 #define NI\_MAXHOST 255

Create a GSS Name structure from the given hostname. This function tries to resolve the given host name string to the canonical DNS name for the host.

#### Parameters:

*hostname* The host name or numerical address to be resolved and transform into a GSS Name  
*authorization\_hostname* The resulting GSS Name

#### Returns:

GLOBUS\_SUCCESS on successful completion, a error object otherwise

### 3.2.3 Function Documentation

**3.2.3.1 OM\_uint32 globus\_gss\_assist\_accept\_sec\_context (OM\_uint32 \* *minor\_status*, gss\_ctx\_id\_t \* *context\_handle*, const gss\_cred\_id\_t *cred\_handle*, char \*\* *src\_name\_char*, OM\_uint32 \* *ret\_flags*, int \* *user\_to\_user\_flag*, int \* *token\_status*, gss\_cred\_id\_t \* *delegated\_cred\_handle*, int(\*) (void \*, void \*\*, size\_t \*) *gss\_assist\_get\_token*, void \* *gss\_assist\_get\_context*, int(\*) (void \*, void \*, size\_t) *gss\_assist\_send\_token*, void \* *gss\_assist\_send\_context*)**

This routine accepts a GSSAPI security context and is called by the gram\_gatekeeper. It isolates the GSSAPI from the rest of the gram code.

Initialize a gssapi security connection. Used by the server. The context\_handle is returned, and there is one for each connection. This routine will take care of the looping and token processing, using the supplied get\_token and send\_token routines.

#### Parameters:

*minor\_status* gssapi return code  
*context\_handle* pointer to returned context.  
*cred\_handle* the cred handle obtained by acquire\_cred.  
*src\_name\_char* Pointer to char string representation of the client which contacted the server. Maybe NULL if not wanted. Should be freed when done.  
*ret\_flags* Pointer to which services are available after the connection is established. Maybe NULL if not wanted. We will also use this to pass in flags to the globus version of gssapi\_ssleay  
*user\_to\_user\_flag* Pointer to flag to be set if the src\_name is the same as our name. (Following are particular to this assist routine)  
*token\_status* assist routine get/send token status  
*delegated\_cred\_handle* pointer to be set to the credential delegated by the client if delegation occurs during the security handshake  
*gss\_assist\_get\_token* a get token routine  
*gss\_assist\_get\_context* first arg for the get token routine  
*gss\_assist\_send\_token* a send token routine  
*gss\_assist\_send\_context* first arg for the send token routine

#### Returns:

GSS\_S\_COMPLETE on success Other gss errors on failure.

**3.2.3.2 OM\_uint32 globus\_gss\_assist\_accept\_sec\_context\_async (OM\_uint32 \* *minor\_status*, gss\_ctx\_id\_t \* *context\_handle*, const gss\_cred\_id\_t *cred\_handle*, char \*\* *src\_name\_char*, OM\_uint32 \* *ret\_flags*, int \* *user\_to\_user\_flag*, void \* *input\_buffer*, size\_t *input\_buffer\_len*, void \*\* *output\_bufferp*, size\_t \* *output\_buffer\_lenp*, gss\_cred\_id\_t \* *delegated\_cred\_handle*)**

This is an asynchronous version of the `globus_gss_assist_accept_sec_context()` function. Instead of looping itself it passes in and out the read and written buffers and the calling application is responsible for doing the I/O directly.

**Parameters:**

*minor\_status* gssapi return code

*context\_handle* pointer to returned context.

*cred\_handle* the cred handle obtained by `acquire_cred`.

*src\_name\_char* Pointer to char string representation of the client which contacted the server. Maybe NULL if not wanted. Should be freed when done.

*ret\_flags* Pointer to which services are available after the connection is established. Maybe NULL if not wanted. We will also use this to pass in flags to the globus version of `gssapi_ssleay`

*user\_to\_user\_flag* Pointer to flag to be set if the *src\_name* is the same as our name.

*input\_buffer* pointer to a buffer received from peer.

*input\_buffer\_len* length of the buffer *input\_buffer*.

*output\_bufferp* pointer to a pointer which will be filled in with a pointer to a allocated block of memory. If non-NULL the contents of this block should be written to the peer where they will be fed into the `gss_assist_init_sec_context_async()` function.

*output\_buffer\_lenp* pointer to an integer which will be filled in with the length of the allocated output buffer pointed to by *\*output\_bufferp*.

*delegated\_cred\_handle* pointer to be set to the credential delegated by the client if delegation occurs during the security handshake

**Returns:**

GSS\_S\_COMPLETE on successful completion when this function does not need to be called again.

GSS\_S\_CONTINUE\_NEEDED when *\*output\_bufferp* should be sent to the peer and a new *input\_buffer* read and this function called again.

Other gss errors on failure.

**3.2.3.3 OM\_uint32 globus\_gss\_assist\_acquire\_cred (OM\_uint32 \* *minor\_status*, gss\_cred\_usage\_t *cred\_usage*, gss\_cred\_id\_t \* *output\_cred\_handle*)**

Called once at the start of the process, to obtain the credentials the process is running under. The

**Parameters:**

*minor\_status* pointer for return code

*cred\_usage* GSS\_C\_INITIATE, GSS\_C\_ACCEPT, or GSS\_C\_BOTH

*output\_cred\_handle* Pointer to the returned handle. This needs to be passed to many gss routines.

**Returns:**

GSS\_S\_COMPLETE on success Other GSS return codes

**3.2.3.4 OM\_uint32 globus\_gss\_assist\_acquire\_cred\_ext (OM\_uint32 \* *minor\_status*, char \* *desired\_name\_char*, OM\_uint32 *time\_req*, const gss\_OID\_set *desired\_mechs*, gss\_cred\_usage\_t *cred\_usage*, gss\_cred\_id\_t \* *output\_cred\_handle*, gss\_OID\_set \* *actual\_mechs*, OM\_uint32 \* *time\_rec*)**

Called once at the start of the process, to obtain the credentials the process is running under. All the parameters of the gss\_acquire\_cred, except the desired\_name is a string of the form: [type:]name. This will be imported with the type.

**Returns:**

GSS\_S\_COMPLETE on success Other GSS return codes

**See also:**

globus\_gsi\_gss\_acquire\_cred

**3.2.3.5 OM\_uint32 globus\_gss\_assist\_display\_status (FILE \* *fp*, char \* *comment*, OM\_uint32 *major\_status*, OM\_uint32 *minor\_status*, int *token\_status*)**

Display the messages for the major and minor status on the file pointed at by fp. Takes care of the overloaded major\_status if there was a problem with the get\_token or send\_token routines.

**Parameters:**

*fp* a file pointer

*comment* String to print out before other error messages.

*major\_status* The major status to display

*minor\_status* The minor status to display

*token\_status* token status to display

**Returns:**

0

**3.2.3.6 OM\_uint32 globus\_gss\_assist\_display\_status\_str (char \*\* *str*, char \* *comment*, OM\_uint32 *major\_status*, OM\_uint32 *minor\_status*, int *token\_status*)**

Display the messages for the major and minor status and return a string with the messages. Takes care of the overloaded major\_status if there was a problem with the get\_token or send\_token routines.

**Parameters:**

*str* pointer to char \* for returned string. Must be freed

*comment* String to print out before other error messages.

*major\_status* The major status to display

*minor\_status* The minor status to display

*token\_status* token status to display

**Returns:**

0

### 3.2.3.7 int globus\_gss\_assist\_gridmap (char \* *globusidp*, char \*\* *useridp*)

Routines callable from globus based code to map a globusID to a local unix user

GRIDMAP environment variable pointing at the map file. Defaults to ~/.gridmap

A gridmap file is required if being run as root. if being run as a user, it is not required, and defaults to the current user who is running the command.

This is the same file used by the gssapi\_cleartext but will be used with other gssapi implementations which do not use the gridmap file.

#### Parameters:

*globusidp* the GSSAPI name from the client who requested authentication

*useridp* the resulting user ID name for the local system

#### Returns:

0 on success -1 if bad arguments 1 on error

### 3.2.3.8 int globus\_gss\_assist\_userok (char \* *globusid*, char \* *userid*)

Check to see if a particular globusid is authorized to access the given local user account.

#### Parameters:

*globusid* the globus id in string form - this should be the user's subject

*userid* the local account that access is sought for

#### Returns:

0 on success (authorization allowed) -1 if bad arguments 1 on error

### 3.2.3.9 int globus\_gss\_assist\_map\_local\_user (char \* *local\_user*, char \*\* *globusidp*)

Routine for returning the default globus ID associated with a local user name. This is somewhat of a hack since there is not a guaranteed one-to-one mapping. What we do is look for the first entry in the gridmap file that has the local user as the default login. If the user is not a default on any entry, we find the first entry in which the user exists as a secondary mapping.

#### Parameters:

*local\_user* the local username to find the DN for

*globusidp* the first DN found that reverse maps from the local\_user

#### Returns:

0 on success, otherwise an error object identifier is returned. use globus\_error\_get to get the error object from the id. The resulting error object must be freed using globus\_object\_free when it is no longer needed.

#### See also:

globus\_error\_get

globus\_object\_free

### 3.2.3.10 OM\_uint32 globus\_gss\_assist\_import\_sec\_context (OM\_uint32 \* *minor\_status*, gss\_ctx\_id\_t \* *context\_handle*, int \* *token\_status*, int *fdp*, FILE \* *fperr*)

Import the security context from a file



**Parameters:**

*minor\_status* GSSAPI return code. This is a Globus Error code (or GLOBUS\_SUCCESS) cast to a OM\_uint32 pointer. If an error has occurred, the resulting error (from calling globus\_error\_get on this variable) needs to be freed by the caller

*context\_handle* The imported context

*token\_status* Errors that occurred while reading from the file

*fdp* the file descriptor pointing to a file containing the security context

*fperr* FILE \* to write error messages

**Returns:**

the major status

**3.2.3.11 OM\_uint32 globus\_gss\_assist\_init\_sec\_context (OM\_uint32 \* minor\_status, const gss\_cred\_id\_t cred\_handle, gss\_ctx\_id\_t \* context\_handle, char \* target\_name\_char, OM\_uint32 req\_flags, OM\_uint32 \* ret\_flags, int \* token\_status, int(\*) (void \*, void \*\*, size\_t \*) gss\_assist\_get\_token, void \* gss\_assist\_get\_context, int(\*) (void \*, void \*, size\_t) gss\_assist\_send\_token, void \* gss\_assist\_send\_context)**

Initialize a gssapi security connection. Used by the client. The context\_handle is returned, and there is one for each connection. This routine will take care of the looping and token processing, using the supplied get\_token and send\_token routines.

**Parameters:**

*minor\_status* GSSAPI return code. The new minor\_status is a globus\_result\_t cast to an OM\_uint32. If the call was successful, the minor status is equivalent to GLOBUS\_SUCCESS. Otherwise, it is a globus error object ID that can be passed to globus\_error\_get to get the error object. The error object needs to be freed with globus\_object\_free.

*cred\_handle* the cred handle obtained by acquire\_cred.

*context\_handle* pointer to returned context.

*target\_name\_char* char string representation of the server to be contacted.

*req\_flags* request flags, such as GSS\_C\_DELEG\_FLAG for delegation and the GSS\_C\_MUTUAL\_FLAG for mutual authentication.

*ret\_flags* Pointer to which services are available after the connection is established. Maybe NULL if not wanted.

The Following are particular to this assist routine:

**Parameters:**

*token\_status* the assist routine's get/send token status

*gss\_assist\_get\_token* function pointer for getting the token

*gss\_assist\_get\_context* first argument passed to the gss\_assist\_get\_token function

*gss\_assist\_send\_token* function pointer for setting the token

*gss\_assist\_send\_context* first argument passed to the gss\_assist\_set\_token function pointer

**Returns:**

The major status

**3.2.3.12 OM\_uint32 globus\_gss\_assist\_init\_sec\_context\_async** (OM\_uint32 \* *minor\_status*, const gss\_cred\_id\_t *cred\_handle*, gss\_ctx\_id\_t \* *context\_handle*, char \* *target\_name\_char*, OM\_uint32 *req\_flags*, OM\_uint32 \* *ret\_flags*, void \* *input\_buffer*, size\_t *input\_buffer\_len*, void \*\* *output\_bufferp*, size\_t \* *output\_buffer\_lenp*)

This is an asynchronous version of the [globus\\_gss\\_assist\\_init\\_sec\\_context\(\)](#) function. Instead of looping itself it passes in and out the read and written buffers and the calling application is responsible for doing the I/O directly.

**Parameters:**

*minor\_status* GSSAPI return code. The new minor status is a globus\_result\_t cast to a OM\_uint32. If an error occurred (GSS\_ERROR(major\_status)) the minor\_status is a globus error object id. The error object can be obtained via globus\_error\_get and should be destroyed with globus\_object\_free when no longer needed. If no error occurred, the minor status is equal to GLOBUS\_SUCCESS.

*cred\_handle* the cred handle obtained by acquire\_cred.

*context\_handle* pointer to returned context.

*target\_name\_char* char string representation of the server to be contacted.

*req\_flags* request flags, such as GSS\_C\_DELEG\_FLAG for delegation and the GSS\_C\_MUTUAL\_FLAG for mutual authentication.

*ret\_flags* Pointer to which services are available after the connection is established. Maybe NULL if not wanted.

*input\_buffer* pointer to a buffer received from peer. Should be NULL on first call.

*input\_buffer\_len* length of the buffer input\_buffer. Should be zero on first call.

*output\_bufferp* pointer to a pointer which will be filled in with a pointer to a allocated block of memory. If non-NULL the contents of this block should be written to the peer where they will be fed into the gss\_assist\_init\_sec\_context\_async() function.

*output\_buffer\_lenp* pointer to an integer which will be filled in with the length of the allocated output buffer pointed to by \*output\_bufferp.

**Returns:**

GSS\_S\_COMPLETE on successful completion when this function does not need to be called again.

GSS\_S\_CONTINUE\_NEEDED when \*output\_bufferp should be sent to the peer and a new input\_buffer read and this function called again.

Other gss errors on failure.

**3.2.3.13 OM\_uint32 globus\_gss\_assist\_will\_handle\_restrictions** (OM\_uint32 \* *minor\_status*, gss\_ctx\_id\_t \* *context\_handle*)

Sets the context to handle restrictions

**Parameters:**

*minor\_status* the resulting minor status from setting the context handle

*context\_handle* the context handle to set the minor status of

**Returns:**

the major status from setting the context

**3.2.3.14 OM\_uint32 globus\_gss\_assist\_get\_unwrap** (OM\_uint32 \* *minor\_status*, const gss\_ctx\_id\_t *context\_handle*, char \*\* *data*, size\_t \* *length*, int \* *token\_status*, int(\*) (void \*, void \*\*, size\_t \*) *gss\_assist\_get\_token*, void \* *gss\_assist\_get\_context*, FILE \* *fperr*)

Gets a token using the specific tokenizing functions, and performs the GSS unwrap of that token

**See also:**

`gss_unwrap`

**Parameters:**

*minor\_status* GSSAPI return code,

**See also:**

`gss_unwrap`

**Parameters:**

*context\_handle* the context

*data* pointer to be set to the unwrapped application data. This must be freed by the caller.

*length* pointer to be set to the length of the *data* byte array.

*token\_status* assist routine get/send token status

*gss\_assist\_get\_token* a detokenizing routine

*gss\_assist\_get\_context* first arg for above routine

*fperr* error stream to print to

**Returns:**

GSS\_S\_COMPLETE on success Other gss errors on failure.

**3.2.3.15** `OM_uint32 globus_gss_assist_wrap_send (OM_uint32 * minor_status, const gss_ctx_id_t context_handle, char * data, size_t length, int * token_status, int(*) (void *, void *, size_t) gss_assist_send_token, void * gss_assist_send_context, FILE * fperr)`

**Parameters:**

*minor\_status* GSSAPI return code. If the call was successful, the minor status is equal to GLOBUS\_SUCCESS. Otherwise, it is an error object ID for which `globus_error_get()` and `globus_object_free()` can be used to get and destroy it.

*context\_handle* the context.

*data* pointer to application data to wrap and send

*length* length of the *data* array

*token\_status* assist routine get/send token status

*gss\_assist\_send\_token* a send\_token routine

*gss\_assist\_send\_context* first arg for the send\_token

*fperr* file handle to write error message to.

**Returns:**

GSS\_S\_COMPLETE on success Other gss errors on failure.

**See also:**

`gss_wrap()`

### 3.3 GSI GSS Assist Constants

**Enumerations**

- enum `globus_gsi_gss_assist_error_t` {  
    GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_SUCCESS = 0,

```

GLOBUS_GSI_GSS_ASSIST_ERROR_WITH_ARGUMENTS = 1,
GLOBUS_GSI_GSS_ASSIST_ERROR_USER_ID_DOESNT_MATCH = 2,
GLOBUS_GSI_GSS_ASSIST_ERROR_IN_GRIDMAP_NO_USER_ENTRY = 3,
GLOBUS_GSI_GSS_ASSIST_ERROR_WITH_GRIDMAP = 4,
GLOBUS_GSI_GSS_ASSIST_ERROR_INVALID_GRIDMAP_FORMAT = 5,
GLOBUS_GSI_GSS_ASSIST_ERROR_ERRNO = 6,
GLOBUS_GSI_GSS_ASSIST_ERROR_WITH_INIT = 7,
GLOBUS_GSI_GSS_ASSIST_ERROR_WITH_WRAP = 8,
GLOBUS_GSI_GSS_ASSIST_ERROR_WITH_TOKEN = 9,
GLOBUS_GSI_GSS_ASSIST_ERROR_EXPORTING_CONTEXT = 10,
GLOBUS_GSI_GSS_ASSIST_ERROR_IMPORTING_CONTEXT = 11,
GLOBUS_GSI_GSS_ASSIST_ERROR_INITIALIZING_CALLOUT_HANDLE = 12,
GLOBUS_GSI_GSS_ASSIST_ERROR_WITH_CALLOUT_CONFIG = 13,
GLOBUS_GSI_GSS_ASSIST_CALLOUT_ERROR = 14,
GLOBUS_GSI_GSS_ASSIST_GSSAPI_ERROR = 15,
GLOBUS_GSI_GSS_ASSIST_GRIDMAP_LOOKUP_FAILED = 16,
GLOBUS_GSI_GSS_ASSIST_BUFFER_TOO_SMALL = 17,
GLOBUS_GSI_GSS_ASSIST_ERROR_CANONICALIZING_HOSTNAME = 18 }

```

### 3.3.1 Enumeration Type Documentation

#### 3.3.1.1 enum `globus_gsi_gss_assist_error_t`

GSI GSS Assist Error codes.

Enumeration values:

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_SUCCESS*** Success.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_ARGUMENTS*** No user entry in gridmap file.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_USER\_ID\_DOESNT\_MATCH*** Error user ID doesn't match.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_IN\_GRIDMAP\_NO\_USER\_ENTRY*** Error with arguments passed to function.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_GRIDMAP*** Error querying gridmap file.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_INVALID\_GRIDMAP\_FORMAT*** Invalid gridmap file format.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_ERRNO*** System Error.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_INIT*** Error during context initialization.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_WRAP*** Error during message wrap.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_TOKEN*** Error with token.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_EXPORTING\_CONTEXT*** Error exporting context.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_IMPORTING\_CONTEXT*** Error importing context.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_INITIALIZING\_CALLOUT\_HANDLE*** Error initializing callout handle.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_CALLOUT\_CONFIG*** Error reading callout configuration.

***GLOBUS\_GSI\_GSS\_ASSIST\_CALLOUT\_ERROR*** Error invoking callout.

***GLOBUS\_GSI\_GSS\_ASSIST\_GSSAPI\_ERROR*** A GSSAPI returned an error.

***GLOBUS\_GSI\_GSS\_ASSIST\_GRIDMAP\_LOOKUP\_FAILED*** Gridmap lookup failure.

***GLOBUS\_GSI\_GSS\_ASSIST\_BUFFER\_TOO\_SMALL*** Caller provided insufficient buffer space for local identity.

***GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_CANONICALIZING\_HOSTNAME*** Failed to obtain canonical host name.

## 3.4 Security Token Transport

Token routines using fread and fwrite.

### Token Get File Descriptor

- int [globus\\_gss\\_assist\\_token\\_get\\_fd](#) (void \*arg, void \*\*bufp, size\_t \*sizep)

### Token Send File Descriptor

- int [globus\\_gss\\_assist\\_token\\_send\\_fd](#) (void \*arg, void \*buf, size\_t size)

### Token Send File Descriptor Without Length

- int [globus\\_gss\\_assist\\_token\\_send\\_fd\\_without\\_length](#) (void \*arg, void \*buf, size\_t size)

### Token Send File Descriptor Flag EX

- int [globus\\_gss\\_assist\\_token\\_send\\_fd\\_ex](#) (void \*exp, void \*buf, size\_t size)

### Token Get Nexus

- int [globus\\_gss\\_assist\\_token\\_get\\_nexus](#) (void \*arg, void \*\*bufp, size\_t \*sizep)

### Token Send Nexus

- int [globus\\_gss\\_assist\\_token\\_send\\_nexus](#) (void \*arg, void \*buf, size\_t size)

### Token Send Nexus Without Length

- int [globus\\_gss\\_assist\\_token\\_send\\_nexus\\_without\\_length](#) (void \*arg, void \*buf, size\_t size)

### Token Send Nexus EX

- int [globus\\_gss\\_assist\\_token\\_send\\_nexus\\_ex](#) (void \*exp, void \*buf, size\_t size)

#### 3.4.1 Detailed Description

Token routines using fread and fwrite.

Additional code has been added to detect tokens which are sent without a length field. These can currently be only SSL tokens. This does require some knowledge of the underlying GSSAPI, by the application, but is within the guidelines of the GSSAPI specifications.

The get routine will automatically attempt this test, while a new send routine will check a flag. The old send routine will work as before, sending a 4-byte length.

### 3.4.2 Function Documentation

#### 3.4.2.1 `int globus_gss_assist_token_get_fd (void * arg, void ** bufp, size_t * sizep)`

Use a open file discriptor to get a token. This function provides parameter types that allow it to be passed to [globus\\_gss\\_assist\\_init\\_sec\\_context](#) and [globus\\_gss\\_assist\\_accept\\_sec\\_context](#)

##### Parameters:

- arg* the FILE \* stream cast to a void pointer
- bufp* the resulting token
- sizep* the size (number of bytes) read into bufp

##### Returns:

- 0 on success > 0 is internal return < 0 is the -errno

#### 3.4.2.2 `int globus_gss_assist_token_send_fd (void * arg, void * buf, size_t size)`

Write a token to the open file descriptor. Will write it with a 4 byte length. This function provides parameter types that allow it to be passed to [globus\\_gss\\_assist\\_init\\_sec\\_context](#) and [globus\\_gss\\_assist\\_accept\\_sec\\_context](#)

##### Parameters:

- arg* the FILE \* stream to send the token on
- buf* the token
- size* the size of the token in bytes

##### Returns:

- 0 on success >0 on error <0 on errno error

#### 3.4.2.3 `int globus_gss_assist_token_send_fd_without_length (void * arg, void * buf, size_t size)`

Write a token to the open file descriptor. Will write it without a length. so as to

#### 3.4.2.4 `int globus_gss_assist_token_send_fd_ex (void * exp, void * buf, size_t size)`

Write a token to the open file descriptor. will look at the flag to determine if the length field need to be written.

##### Parameters:

- exp* the globus\_gss\_assist\_ex variable that holds the FILE \* stream and flags to bet set
- buf* the token buffer to send
- size* size of the token buffer

##### Returns:

- 0 on success >0 on error <0 on errno error (-errno)

### 3.4.2.5 `int globus_gss_assist_token_get_nexus (void * arg, void ** bufp, size_t * sizep)`

Use a nexus socket to get the tokens.

Additional code has been added to detect tokens which are sent without a length field. These can currently be only SSL tokens. This does require some knowledge of the underlying GSSAPI, by the application, but is within the guidelines of the GSSAPI specifications.

The get routine will automatically attempt this test, while a new send routine will check a flag. The old send routine will work as before, sending a 4-byte length.

#### Parameters:

*arg* the `globus_io_handle_t` to get the token from

*bufp* the buffer to read the token into

*sizep* the size of what gets read

#### Returns:

0 on success > 0 is internal return < 0 is the -errno returned from nexus

### 3.4.2.6 `int globus_gss_assist_token_send_nexus (void * arg, void * buf, size_t size)`

Write a token to the nexus io handle. This function provides parameter types that allow it to be passed to [globus\\_gss\\_assist\\_init\\_sec\\_context](#) and [globus\\_gss\\_assist\\_accept\\_sec\\_context](#)

#### Parameters:

*arg* nexus io handle to send the token on

*buf* the token as a buffer

*size* the size of the buffer

#### Returns:

0 on success >0 on error <0 on errno error (-errno)

### 3.4.2.7 `int globus_gss_assist_token_send_nexus_without_length (void * arg, void * buf, size_t size)`

Send a token on a nexus IO handle. Using this function the length is not sent.

#### See also:

[globus\\_gss\\_assist\\_token\\_get\\_nexus\(\)](#) for further info.

### 3.4.2.8 `int globus_gss_assist_token_send_nexus_ex (void * exp, void * buf, size_t size)`

Write a token to the open file descriptor. will look at the flag to determine if the length field need to be written.

#### Parameters:

*exp* The `globus_gss_assist_ex` that the wraps the nexus IO handle to send the token on

*buf* the buffer holding the token

*size* the size of the buffer

#### Returns:

0 on success >0 on error <0 on errno error (-errno)

## Index

Activation, [1](#)

globus\_gsi\_gss\_assist

- globus\_gss\_assist\_accept\_sec\_context, [4](#)
- globus\_gss\_assist\_accept\_sec\_context\_async, [4](#)
- globus\_gss\_assist\_acquire\_cred, [5](#)
- globus\_gss\_assist\_acquire\_cred\_ext, [5](#)
- globus\_gss\_assist\_display\_status, [6](#)
- globus\_gss\_assist\_display\_status\_str, [6](#)
- globus\_gss\_assist\_get\_unwrap, [9](#)
- globus\_gss\_assist\_gridmap, [6](#)
- globus\_gss\_assist\_import\_sec\_context, [7](#)
- globus\_gss\_assist\_init\_sec\_context, [8](#)
- globus\_gss\_assist\_init\_sec\_context\_async, [8](#)
- globus\_gss\_assist\_map\_local\_user, [7](#)
- globus\_gss\_assist\_userok, [7](#)
- globus\_gss\_assist\_will\_handle\_restrictions, [9](#)
- globus\_gss\_assist\_wrap\_send, [10](#)
- NI\_MAXHOST, [4](#)

globus\_gsi\_gss\_assist\_activation

- GLOBAL\_GSI\_GSS\_ASSIST\_MODULE, [2](#)

GLOBAL\_GSI\_GSS\_ASSIST\_BUFFER\_TOO\_SMALL

- globus\_gsi\_gss\_assist\_constants, [12](#)

GLOBAL\_GSI\_GSS\_ASSIST\_CALLOUT\_ERROR

- globus\_gsi\_gss\_assist\_constants, [11](#)

globus\_gsi\_gss\_assist\_constants

- GLOBAL\_GSI\_GSS\_ASSIST\_BUFFER\_TOO\_SMALL, [12](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_CALLOUT\_ERROR, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_CANONICALIZING\_HOSTNAME, [12](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_ERRNO, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_EXPORTING\_CONTEXT, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_IMPORTING\_CONTEXT, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_IN\_GRIDMAP\_NO\_USER\_ENTRY, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_INITIALIZING\_CALLOUT\_HANDLE, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_INVALID\_GRIDMAP\_FORMAT, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_SUCCESS, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_USER\_ID\_DOESNT\_MATCH, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_ARGUMENTS, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_CALLOUT\_CONFIG, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_GRIDMAP, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_INIT, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_TOKEN, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_WRAP, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_GRIDMAP\_LOOKUP\_FAILED, [11](#)

- GLOBAL\_GSI\_GSS\_ASSIST\_GSSAPI\_ERROR, [11](#)

globus\_gsi\_gss\_assist\_constants

- globus\_gsi\_gss\_assist\_error\_t, [11](#)

GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_CANONICALIZING\_HOSTNAME

- globus\_gsi\_gss\_assist\_constants, [12](#)

GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_ERRNO

- globus\_gsi\_gss\_assist\_constants, [11](#)

GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_EXPORTING\_CONTEXT

- globus\_gsi\_gss\_assist\_constants, [11](#)

GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_IMPORTING\_CONTEXT

- globus\_gsi\_gss\_assist\_constants, [11](#)

GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_IN\_GRIDMAP\_NO\_USER\_ENTRY

- globus\_gsi\_gss\_assist\_constants, [11](#)

GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_INITIALIZING\_CALLOUT\_HANDLE

- globus\_gsi\_gss\_assist\_constants, [11](#)

GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_INVALID\_GRIDMAP\_FORMAT

- globus\_gsi\_gss\_assist\_constants, [11](#)

GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_SUCCESS

- globus\_gsi\_gss\_assist\_constants, [11](#)

globus\_gsi\_gss\_assist\_error\_t

- globus\_gsi\_gss\_assist\_constants, [11](#)

GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_USER\_ID\_DOESNT\_MATCH

- globus\_gsi\_gss\_assist\_constants, [11](#)

GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_ARGUMENTS

- globus\_gsi\_gss\_assist\_constants, [11](#)

GLOBAL\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_CALLOUT\_CONFIG

- globus\_gsi\_gss\_assist\_constants, [11](#)



GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_GRIDMAP  
     globus\_gsi\_gss\_assist\_constants, 11  
 GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_INIT  
     globus\_gsi\_gss\_assist\_constants, 11  
 GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_TOKEN  
     globus\_gsi\_gss\_assist\_constants, 11  
 GLOBUS\_GSI\_GSS\_ASSIST\_ERROR\_WITH\_WRAP  
     globus\_gsi\_gss\_assist\_constants, 11  
 GLOBUS\_GSI\_GSS\_ASSIST\_GRIDMAP\_LOOKUP\_FAILED  
     globus\_gsi\_gss\_assist\_constants, 11  
 GLOBUS\_GSI\_GSS\_ASSIST\_GSSAPI\_ERROR  
     globus\_gsi\_gss\_assist\_constants, 11  
 GLOBUS\_GSI\_GSS\_ASSIST\_MODULE  
     globus\_gsi\_gss\_assist\_activation, 2  
 globus\_gsi\_gss\_assist\_tokens  
     globus\_gss\_assist\_token\_get\_fd, 13  
     globus\_gss\_assist\_token\_get\_nexus, 13  
     globus\_gss\_assist\_token\_send\_fd, 13  
     globus\_gss\_assist\_token\_send\_fd\_ex, 13  
     globus\_gss\_assist\_token\_send\_fd\_without\_length, 13  
     globus\_gss\_assist\_token\_send\_nexus, 14  
     globus\_gss\_assist\_token\_send\_nexus\_ex, 14  
     globus\_gss\_assist\_token\_send\_nexus\_without\_length, 14  
 globus\_gss\_assist\_accept\_sec\_context  
     globus\_gsi\_gss\_assist, 4  
 globus\_gss\_assist\_accept\_sec\_context\_async  
     globus\_gsi\_gss\_assist, 4  
 globus\_gss\_assist\_acquire\_cred  
     globus\_gsi\_gss\_assist, 5  
 globus\_gss\_assist\_acquire\_cred\_ext  
     globus\_gsi\_gss\_assist, 5  
 globus\_gss\_assist\_display\_status  
     globus\_gsi\_gss\_assist, 6  
 globus\_gss\_assist\_display\_status\_str  
     globus\_gsi\_gss\_assist, 6  
 globus\_gss\_assist\_get\_unwrap  
     globus\_gsi\_gss\_assist, 9  
 globus\_gss\_assist\_gridmap  
     globus\_gsi\_gss\_assist, 6  
 globus\_gss\_assist\_import\_sec\_context  
     globus\_gsi\_gss\_assist, 7  
 globus\_gss\_assist\_init\_sec\_context  
     globus\_gsi\_gss\_assist, 8  
 globus\_gss\_assist\_init\_sec\_context\_async  
     globus\_gsi\_gss\_assist, 8  
 globus\_gss\_assist\_map\_local\_user  
     globus\_gsi\_gss\_assist, 7  
 globus\_gss\_assist\_token\_get\_fd  
     globus\_gsi\_gss\_assist\_tokens, 13

globus\_gss\_assist\_token\_get\_nexus  
     globus\_gsi\_gss\_assist\_tokens, 13  
 globus\_gss\_assist\_token\_send\_fd  
     globus\_gsi\_gss\_assist\_tokens, 13  
 globus\_gss\_assist\_token\_send\_fd\_ex  
     globus\_gsi\_gss\_assist\_tokens, 13  
 globus\_gss\_assist\_token\_send\_fd\_without\_length  
     globus\_gsi\_gss\_assist\_tokens, 13  
 globus\_gss\_assist\_token\_send\_nexus  
     globus\_gsi\_gss\_assist\_tokens, 14  
 globus\_gss\_assist\_token\_send\_nexus\_ex  
     globus\_gsi\_gss\_assist\_tokens, 14  
 globus\_gss\_assist\_token\_send\_nexus\_without\_length  
     globus\_gsi\_gss\_assist\_tokens, 14  
 globus\_gss\_assist\_userok  
     globus\_gsi\_gss\_assist, 7  
 globus\_gss\_assist\_will\_handle\_restrictions  
     globus\_gsi\_gss\_assist, 9  
 globus\_gss\_assist\_wrap\_send  
     globus\_gsi\_gss\_assist, 10  
 GSI GSS Assist Constants, 10  
  
 NI\_MAXHOST  
     globus\_gsi\_gss\_assist, 4

Security Token Transport, 12

Utility Functions, 2