

# globus rls client Reference Manual

## 5.2

Generated by Doxygen 1.3.5

Sat Feb 6 11:37:00 2010

## Contents

<a href="#">1 globus rls client Main Page</a>	<a href="#">1</a>
<a href="#">2 globus rls client Module Index</a>	<a href="#">1</a>
<a href="#">3 globus rls client Data Structure Index</a>	<a href="#">2</a>
<a href="#">4 globus rls client Module Documentation</a>	<a href="#">2</a>
<a href="#">5 globus rls client Data Structure Documentation</a>	<a href="#">33</a>

## 1 globus rls client Main Page

The Globus Replica Location Service (RLS) C API provides functions to view and update data in a RLS catalog. There are 2 types of RLS servers, Local Replica Catalog (LRC) servers, which maintain Logical to Physical File Name mappings (LFN to PFN), and Replica Location Index (RLI) servers, which maintain LFN to LRC mappings. Note an RLS server can act as both an LRC and RLI server.

Functions are divided into the following groups:

- [Activation](#)
- [Connection Management](#)
- [Operations on an LRC server](#)
- [Operations on an RLI server](#)
- [Miscellaneous Types and Functions](#)
- [Query Results](#)
- [Status Codes](#)

Applications using this API should include `globus_rls_client.h`, and be linked with the library `globus_rls_client_FLAVOR`.

## 2 globus rls client Module Index

### 2.1 globus rls client Modules

Here is a list of all modules:

Status Codes	<a href="#">2</a>
Miscellaneous	<a href="#">6</a>
Query Results	<a href="#">11</a>
Activation	<a href="#">13</a>

Connection Management	14
LRC Operations	16
RLI Operations	28

## 3 globus\_rls client Data Structure Index

### 3.1 globus\_rls client Data Structures

Here are the data structures with brief descriptions:

<a href="#">globus_rls_attribute_object_t</a> (Globus_rls_client_lrc_attr_search() returns a list of these structures which include the object name (LFN or PFN) and attribute value found by the query )	33
<a href="#">globus_rls_attribute_t</a> (Object (LFN or PFN) attribute type )	33
<a href="#">globus_rls_handle_t</a> (RLS Client Handle )	34
<a href="#">globus_rls_rli_info_t</a> (Information about RLI server, returned by <a href="#">globus_rls_client_lrc_rli_info()</a> and <a href="#">globus_rls_client_lrc_rli_list()</a> )	35
<a href="#">globus_rls_sender_info_t</a> (Information about server sending updates to an rli, returned by <a href="#">globus_rls_client_rli_sender_list()</a> )	36
<a href="#">globus_rls_stats_t</a> (Various configuration options and statistics about an RLS server returned in the following structures by <a href="#">globus_rls_client_stats()</a> )	36
<a href="#">globus_rls_string2_bulk_t</a>	36
<a href="#">globus_rls_string2_t</a>	37

## 4 globus\_rls client Module Documentation

### 4.1 Status Codes

All of the functions in the API that return status return it in a `globus_result_t` structure.

Defines

- #define [GLOBUS\\_RLS\\_SUCCESS](#)0
- #define [GLOBUS\\_RLS\\_GLOBUSERF](#)1
- #define [GLOBUS\\_RLS\\_INVHANDLE](#)2
- #define [GLOBUS\\_RLS\\_BADURL](#)3
- #define [GLOBUS\\_RLS\\_NOMEMORY](#)4
- #define [GLOBUS\\_RLS\\_OVERFLOW](#)5
- #define [GLOBUS\\_RLS\\_BADARG](#)6
- #define [GLOBUS\\_RLS\\_PERM](#)7
- #define [GLOBUS\\_RLS\\_BADMETHOD](#)8
- #define [GLOBUS\\_RLS\\_INVSERVER](#)9

- #define GLOBUS\_RLS\_MAPPING\_NEXIST 10
- #define GLOBUS\_RLS\_LFN\_EXIST 11
- #define GLOBUS\_RLS\_LFN\_NEXIST 12
- #define GLOBUS\_RLS\_PFN\_EXIST 13
- #define GLOBUS\_RLS\_PFN\_NEXIST 14
- #define GLOBUS\_RLS\_LRC\_EXIST 15
- #define GLOBUS\_RLS\_LRC\_NEXIST 16
- #define GLOBUS\_RLS\_DBERROR 17
- #define GLOBUS\_RLS\_RLI\_EXIST 18
- #define GLOBUS\_RLS\_RLI\_NEXIST 19
- #define GLOBUS\_RLS\_MAPPING\_EXIST 20
- #define GLOBUS\_RLS\_INV\_ATTR\_TYPE 21
- #define GLOBUS\_RLS\_ATTR\_EXIST 22
- #define GLOBUS\_RLS\_ATTR\_NEXIST 23
- #define GLOBUS\_RLS\_INV\_OBJ\_TYPE 24
- #define GLOBUS\_RLS\_INV\_ATTR\_OP 25
- #define GLOBUS\_RLS\_UNSUPPORTED 26
- #define GLOBUS\_RLS\_TIMEOUT 27
- #define GLOBUS\_RLS\_TOO\_MANY\_CONNECTIONS 28
- #define GLOBUS\_RLS\_ATTR\_VALUE\_NEXIST 29
- #define GLOBUS\_RLS\_ATTR\_INUSE 30

#### 4.1.1 Detailed Description

All of the functions in the API that return status return it in a `globus_result_t` structure.

Prior to version 2.0.0 an integer status was returned. The `globus_result_t` structure includes an integer "type" which is set to one of the status codes defined below (the same values that were returned by earlier versions of the API). The function `globus_rls_client_error_info` may be used to extract the status code and/or error message from a `globus_result_t`. GLOBUS\_SUCCESS is returned when the operation was successful.

#### 4.1.2 Define Documentation

##### 4.1.2.1 #define GLOBUS\_RLS\_SUCCESS 0

Operation succeeded.

##### 4.1.2.2 #define GLOBUS\_RLS\_GLOBUSERR 1

An error was returned by the Globus I/O module.

##### 4.1.2.3 #define GLOBUS\_RLS\_INVHANDLE 2

The `globus_rls_handle` handle is invalid.

##### 4.1.2.4 #define GLOBUS\_RLS\_BADURL 3

The URL could not be parsed.

##### 4.1.2.5 #define GLOBUS\_RLS\_NOMEMORY 4

Out of memory.

#### 4.1.2.6 #define GLOBUS\_RLS\_OVERFLOW 5

A result was too large to fit in buffer.

#### 4.1.2.7 #define GLOBUS\_RLS\_BADARG 6

Bad argument (eg NULL where string pointer expected).

#### 4.1.2.8 #define GLOBUS\_RLS\_PERM 7

Client does not have permission for requested action.

#### 4.1.2.9 #define GLOBUS\_RLS\_BADMETHOD 8

RPC error, invalid method name sent to server.

#### 4.1.2.10 #define GLOBUS\_RLS\_INVSERVER 9

LRC request made to RLI server or vice versa.

#### 4.1.2.11 #define GLOBUS\_RLS\_MAPPING\_NEXIST 10

LFN,PFN (LRC) or LFN,LRC (RLI) mapping doesn't exist.

#### 4.1.2.12 #define GLOBUS\_RLS\_LFN\_EXIST 11

LFN already exists in LRC or RLI database.

#### 4.1.2.13 #define GLOBUS\_RLS\_LFN\_NEXIST 12

LFN doesn't exist in LRC or RLI database.

#### 4.1.2.14 #define GLOBUS\_RLS\_PFN\_EXIST 13

PFN already exists in LRC database.

#### 4.1.2.15 #define GLOBUS\_RLS\_PFN\_NEXIST 14

PFN doesn't exist in LRC database.

#### 4.1.2.16 #define GLOBUS\_RLS\_LRC\_EXIST 15

LRC already exists in LRC or RLI database.

#### 4.1.2.17 #define GLOBUS\_RLS\_LRC\_NEXIST 16

LRC doesn't exist in RLI database.

#### 4.1.2.18 #define GLOBUS\_RLS\_DBERROR 17

Database error.

## 4.1.2.19 #define GLOBUS\_RLS\_RLI\_EXIST 18

RLI already exists in LRC database.

## 4.1.2.20 #define GLOBUS\_RLS\_RLI\_NEXIST 19

RLI doesn't exist in LRC.

## 4.1.2.21 #define GLOBUS\_RLS\_MAPPING\_EXIST 20

LFN,PFN (LRC) or LFN,LRC (RLI) mapping already exists.

## 4.1.2.22 #define GLOBUS\_RLS\_INV\_ATTR\_TYPE 21

Invalid attribute type, see globus\_uls\_attr\_type\_t.

## 4.1.2.23 #define GLOBUS\_RLS\_ATTR\_EXIST 22

Attribute already exists.

## 4.1.2.24 #define GLOBUS\_RLS\_ATTR\_NEXIST 23

Attribute doesn't exist.

## 4.1.2.25 #define GLOBUS\_RLS\_INV\_OBJ\_TYPE 24

Invalid object type, see globus\_uls\_obj\_type\_t.

## 4.1.2.26 #define GLOBUS\_RLS\_INV\_ATTR\_OP 25

Invalid attribute search operator, see globus\_uls\_attr\_op\_t.

## 4.1.2.27 #define GLOBUS\_RLS\_UNSUPPORTED 26

Operation is unsupported.

## 4.1.2.28 #define GLOBUS\_RLS\_TIMEOUT 27

IO timeout.

## 4.1.2.29 #define GLOBUS\_RLS\_TOO\_MANY\_CONNECTIONS 28

Too many connections.

## 4.1.2.30 #define GLOBUS\_RLS\_ATTR\_VALUE\_NEXIST 29

Attribute with specified value not found.

## 4.1.2.31 #define GLOBUS\_RLS\_ATTR\_INUSE 30

Attribute in use by some object, can't be deleted.

## 4.2 Miscellaneous

Miscellaneous functions and types.

### Data Structures

- struct [globus\\_uls\\_attribute\\_t](#)  
Object (LFN or PFN) attribute type.
- struct [globus\\_uls\\_stats\\_t](#)  
Various configuration options and statistics about an RLS server returned in the following structure [globus\\_uls\\_client\\_stats\(\)](#)

### Defines

- #define [RLS\\_LRC\\_SERVER](#) 0x1
- #define [RLS\\_RLISERVER](#) 0x2
- #define [RLS\\_RCV\\_LFNLIST](#) 0x4
- #define [RLS\\_RCV\\_BLOOMFILTER](#) 0x8
- #define [RLS\\_SND\\_LFNLIST](#) 0x10
- #define [RLS\\_SND\\_BLOOMFILTER](#) 0x20
- #define [RLS\\_INITIALIZED](#) 0x40

### Enumerations

- enum [globus\\_uls\\_pattern\\_t](#)  
[uls\\_pattern\\_unix](#)  
[uls\\_pattern\\_sq](#)
- enum [globus\\_uls\\_attr\\_type\\_t](#)  
[globus\\_uls\\_attr\\_type\\_date](#)  
[globus\\_uls\\_attr\\_type\\_t](#)  
[globus\\_uls\\_attr\\_type\\_int](#)  
[globus\\_uls\\_attr\\_type\\_str](#)
- enum [globus\\_uls\\_obj\\_type\\_t](#)  
[globus\\_uls\\_obj\\_lrc\\_lfn](#)  
[globus\\_uls\\_obj\\_lrc\\_pfn](#)  
[globus\\_uls\\_obj\\_rli\\_lfn](#)  
[globus\\_uls\\_obj\\_rli\\_lrc](#)
- enum [globus\\_uls\\_attr\\_op\\_t](#)  
[globus\\_uls\\_attr\\_op\\_all](#)  
[globus\\_uls\\_attr\\_op\\_eq](#)  
[globus\\_uls\\_attr\\_op\\_ne](#)  
[globus\\_uls\\_attr\\_op\\_gt](#)  
[globus\\_uls\\_attr\\_op\\_ge](#)  
[globus\\_uls\\_attr\\_op\\_lt](#)

```

globus_uls_attr_op_le
globus_uls_attr_op_bt
globus_uls_attr_op_like
• enum globus_uls_admin_cmd {
    globus_uls_admin_cmd_ping
    globus_uls_admin_cmd_quit
    globus_uls_admin_cmd_ssu
}

```

## Functions

- `globus_result_t globus_uls_client_admin(globus_uls_handle_t h, globus_uls_admin_cmd cmd)`
- `globus_result_t globus_uls_client_get_configuration(globus_uls_handle_t h, char option, globus_list_t conf_list)`
- `globus_result_t globus_uls_client_set_configuration(globus_uls_handle_t h, char option, char value)`
- `globus_result_t globus_uls_client_stats(globus_uls_handle_t h, globus_uls_stats_t stats)`
- `char globus_uls_client_attr2globus_uls_attribute_tattr, char buf, int buflen)`
- `globus_result_t globus_uls_client_setattr(globus_uls_attr_type_t type, char sval, globus_uls_attribute_t attr)`
- `globus_result_t globus_uls_client_error_info(globus_result_t r, int rc, char buf, int buflen, globus_bool_t preserve)`
- `int globus_list_len(globus_list_t len)`
- `char globus_uls_errmsg(int rc, char specimsg, char buf, int buflen)`

### 4.2.1 Detailed Description

Miscellaneous functions and types.

### 4.2.2 Define Documentation

#### 4.2.2.1 #define RLS\_LRCSERVER 0x1

Server is LRC server.

#### 4.2.2.2 #define RLS\_RLISERVER 0x2

Server is RLI server.

#### 4.2.2.3 #define RLS\_RCVLFNLIST 0x4

RLI accepts LFN list updates.

#### 4.2.2.4 #define RLS\_RCVBLOOMFILTER 0x8

RLI accepts Bloom filter updates.

#### 4.2.2.5 #define RLS\_SNDLFNLIST 0x10

LRC sends LFN list updates.



#### 4.2.2.6 #define RLS\_SNDBLOOMFILTER 0x20

LRC sends Bloom filter updates.

#### 4.2.2.7 #define RLS\_INITIALIZED 0x40

RLC is fully initialized.

### 4.2.3 Enumeration Type Documentation

#### 4.2.3.1 enum [globus\\_rls\\_pattern\\_t](#)

Wildcard character style.

Enumeration values:

rls\_pattern\_unix Unix like globbing chars ( , ?).

rls\_pattern\_sql SQL "like" wildcards ( , \_).

#### 4.2.3.2 enum [globus\\_rls\\_attr\\_type\\_t](#)

Attribute Value Types.

Enumeration values:

globus\_rls\_attr\_type\_date Date (time\_t).

globus\_rls\_attr\_type\_t Floating point (double).

globus\_rls\_attr\_type\_int Integer (int).

globus\_rls\_attr\_type\_str String (char ).

#### 4.2.3.3 enum [globus\\_rls\\_obj\\_type\\_t](#)

Object types in LRC and RLI databases.

Enumeration values:

globus\_rls\_obj\_lrc\_lfn LRC Logical File Name.

globus\_rls\_obj\_lrc\_pfn LRC Physical File Name.

globus\_rls\_obj\_rli\_lfn RLI Logical File Name.

globus\_rls\_obj\_rli\_lrc RLI LRC URL.

#### 4.2.3.4 enum [globus\\_rls\\_attr\\_op\\_t](#)

Attribute Value Query Operators.

Enumeration values:

globus\_rls\_attr\_op\_all All values returned.

globus\_rls\_attr\_op\_eq Values matching operand 1 returned.

globus\_rls\_attr\_op\_ne Values not matching operand 1.

globus\_rls\_attr\_op\_gt Values greater than operand 1.

`globus_uls_attr_op_ge` Values greater than or equal to `op1`.  
`globus_uls_attr_op_lt` Values less than operand 1.  
`globus_uls_attr_op_le` Values less than or equal to `op1`.  
`globus_uls_attr_op_btw` Values between operand1 and 2.  
`globus_uls_attr_op_like` Strings "like" operand1 (SQL like).

#### 4.2.3.5 enum `globus_uls_admin_cmd_t` `globus_uls_client_admin_t` Commands.

Enumeration values:

`globus_uls_admin_cmd_ping` Verify RLS server responding.  
`globus_uls_admin_cmd_quit` Tell RLS server to exit.  
`globus_uls_admin_cmd_ssu` Tell LRC server to do softstate update.

#### 4.2.4 Function Documentation

##### 4.2.4.1 `globus_result_t globus_uls_client_admin(globus_uls_handle_t h, globus_uls_admin_cmd_t cmd)`

Miscellaneous administrative operations.

Most operations require the admin privilege.

Parameters:

`h` Handle connected to RLS server.  
`cmd` Command to be sent to RLS server.

Return values:

`GLOBUS_SUCCESS` Command succeeded.

##### 4.2.4.2 `globus_result_t globus_uls_client_get_configuration(globus_uls_handle_t h, char option, globus_list_t conf_list)`

Get server configuration.

Client needs admin privilege.

Parameters:

`h` Handle connected to RLS server.  
`option` Configuration option to get. If NULL all options are retrieved.

Return values:

`conf_list` List of configuration options.

`GLOBUS_SUCCESS` List of retrieved configuration options returned in `conf_list`, each datum is of type `globus_string2_t`. `conf_list` should be freed with `globus_uls_client_free_list()`. There may be multiple "acl" entries in the list, since the access control list can include more than one entry. Each acl configuration value consists of a regular expression (matched against grid-maple users or DNs), a colon, and space separated list of permissions the matching users are granted.

4.2.4.3 `globus_result_t globus_qls_client_set_configuration(globus_qls_handle_t h, char option, char value)`

Set server configuration option.

Client needs admin privilege.

Parameters:

h Handle connected to QLS server.

option Configuration option to set.

value New value for option.

Return values:

GLOBUS\_SUCCESS Option set on server.

4.2.4.4 `globus_result_t globus_qls_client_stats(globus_qls_handle_t h, globus_qls_stats_t qlsstats)`

Retrieve various statistics from QLS server.

Requires stats privilege.

Parameters:

h Handle connected to QLS server.

qlsstats Stats returned here.

Return values:

GLOBUS\_SUCCESS Stats returned in qlsstats

4.2.4.5 `char globus_qls_client_attr2s(globus_qls_attribute_t attr, char buf, int buflen)`

Map attribute value to string.

Parameters:

attr Attribute to convert. If attr->type is `globus_qls_attr_type_data` then the resulting string will be in the format MySQL uses by default, which is YYYYMMDDHHMMSS.

buf Buffer to write string value to. Note if attr->type is `globus_qls_attr_type_string` then attr->value is returned, and buf is unused.

bu len Size of buf in bytes.

Return values:

String Value Attribute value converted to a string.

4.2.4.6 `globus_result_t globus_qls_client_set_attr(globus_qls_attr_type_t type, char sval, globus_qls_attribute_t attr)`

Set `globus_qls_attribute_t` type and value fields from a type and string value.

Parameters:

type Attribute value type.

sval String value to convert to binary. If type is `globus_rls_attr_type_data` sval should be in the form YYYY-MM-DD HH:MM:SS.

attr Attribute whose type and val elds are to be set.

Return values:

GLOBUS\_SUCCESSattr-> typeandattr-> val successfully set.

4.2.4.7 `globus_result_t globus_rls_client_error_info (globus_result_t rc, char buf, int buflen, globus_bool_t preserve)`

Get error code and message from `globus_result_t` returned by this API.

Parameters:

rc Result returned by RLS API function. is freed by this call and should not be referenced again. If preserve is set then a new `globus_result_t` is constructed with the same values and returned as the function value.

buf Address to store error code at. If NULL error code is not returned.

buf Address to store error message at. If NULL error message is not returned.

preserve If GLOBUS\_TRUE then a new `globus_result_t` is constructed with the same values as the old and returned as the function value.

bu en Size of buf.

Return values:

`globus_result_t` If preserve is set a new `globus_result_t` identical to is returned, otherwise GLOBUS\_SUCCESS.

4.2.4.8 `int globus_list_len (globus_list_t len)`

Compute length of list.

`globus_list_size()` is implemented using recursion, besides being inefficient it can run out of stack space when the list is large.

4.2.4.9 `char globus_rls_errmsg (int rc, char specimsg char buf, int buflen)`

Map RLS status code to error string.

Parameters:

rc Status code.

specimsg If not NULL prepended (with a colon) to error string.

buf Buffer to write error message to.

bu en Length of buf. Message will be truncated to t if too long.

Return values:

char Returns buf, error message written to buf.

## 4.3 Query Results

List results are returned as `globus_list_t`'s, list datums depend on the type of query (`globus_rls_string2`, `globus_rls_attribute_t` etc).

## Data Structures

- struct `globus_uls_attribute_object_t`  
`globus_uls_client_lrc_attr_search()` returns a list of these structures which include the object name (LFN or PFN) and attribute value found by the query.
- struct `globus_uls_string2_t`
- struct `globus_uls_string2_bulk_t`

## Functions

- `globus_result_t globus_uls_client_free_list(globus_list_t list)`

### 4.3.1 Detailed Description

List results are returned as `globus_list_t`'s, list datums depend on the type of query (`globus_uls_string2_t`, `globus_uls_attribute_t` etc).

A list result should be freed with `globus_uls_client_free_list()` when it's no longer needed. RLS supports limiting the number of results returned by a single query using an offset and reslimit. The offset specifies which result to begin with, reslimit specifies how many results to return. Offset should begin at 0 to retrieve all records. If reslimit is 0 then all results are returned at once, unless the server has a limit on results configured. If NULL is passed as the offset argument then the API will repeatedly call the query function until all results are retrieved. The following are equivalent examples of how to print the lfn,pfn pairs returned by `globus_uls_client_lrc_get_lfn()`

```
globus_list_t *str2_list;
globus_list_t *p;
globus_uls_string2_t *str2;

// Retrieve all results, API will handle looping through partial results
// if the server has a limit configured. Error handling has been omitted.
globus_uls_client_lrc_get_lfn(h, "somepfn", NULL, 0, &str2_list);
for (p = str2_list; p; p = globus_list_rest(p)) {
    str2 = (globus_uls_string2_t *) globus_list_first(p);
    printf("lfn: %s pfn:%s\n", str2->s1, str2->s2);
}
globus_uls_client_free_list(str2_list);

// This code fragment retrieves results 5 at a time. Note offset is set
// to -1 when the server has no more results to return.
int offset = 0;

while (globus_uls_client_lrc_get_lfn(h, "somepfn", &offset, 5, &str2_list) == GLOBUS_SUCCESS) {
    for (p = str2_list; p; p = globus_list_rest(p)) {
        str2 = (globus_uls_string2_t *) globus_list_first(p);
        printf("lfn: %s pfn:%s\n", str2->s1, str2->s2);
    }
    globus_uls_client_free_list(str2_list);
    if (offset == -1)
        break;
}
```

### 4.3.2 Function Documentation

#### 4.3.2.1 `globus_result_t globus_uls_client_free_list(globus_list_t list)`

Free result list returned by one of the query functions.

## Parameters:

list List returned by one of the query functions.

## Return values:

GLOBUS\_SUCCESS List and contents successfully freed.

## 4.4 Activation

This module must be activated before any functions in this API may be used.

## Defines

- `#define GLOBUS_RLS_CLIENT_MODULE (&globus_rls_client_module)`

## Variables

- `globus_module_descriptor globus_rls_client_module`
- `globus_module_descriptor globus_rls_client_module`

### 4.4.1 Detailed Description

This module must be activated before any functions in this API may be used.

This module depends on other Globus modules GLOBUS\_COMMON\_MODULE and GLOBUS\_IO\_MODULE, which should be activated first:

```
globus_module_activate(GLOBUS_COMMON_MODULE);
globus_module_activate(GLOBUS_IO_MODULE);
globus_module_activate(GLOBUS_RLS_CLIENT_MODULE);
```

When finished modules should be deactivated in reverse order.

### 4.4.2 Define Documentation

#### 4.4.2.1 `#define GLOBUS_RLS_CLIENT_MODULE (&globus_rls_client_module)`

RLS Module Name.

### 4.4.3 Variable Documentation

#### 4.4.3.1 `globus_module_descriptor globus_rls_client_module`

Initial value:

```
{
  "globus_rls_client",
  globus_rls_client_activate,
  globus_rls_client_deactivate,
  GLOBUS_NULL
}
```

RLS module.

4.4.3.2 `globus_module_descriptor` `globus_rls_client_module`

RLS module.

## 4.5 Connection Management

Functions to open and close connections to an RLS server.

Defines

- `#define GLOBUS_RLS_URL_SCHEME "rls"`
- `#define GLOBUS_RLS_URL_SCHEME_NOAUTH "rlsn"`
- `#define GLOBUS_RLS_SERVER_DEFPORT 39281`
- `#define MAXERRMSG 1024`

Functions

- `void globus_rls_client_certificate(char certificate, char keyfile)`
- `void globus_rls_client_proxy_certificate(char proxy)`
- `globus_result_t globus_rls_client_connect(char url, globus_rls_handle_t h)`
- `globus_result_t globus_rls_client_close(globus_rls_handle_t h)`
- `int globus_rls_client_get_timeout()`
- `void globus_rls_client_set_timeout(int seconds)`

### 4.5.1 Detailed Description

Functions to open and close connections to an RLS server.

### 4.5.2 Define Documentation

#### 4.5.2.1 `#define GLOBUS_RLS_URL_SCHEME "rls"`

URL scheme to use when connecting to RLS server.

#### 4.5.2.2 `#define GLOBUS_RLS_URL_SCHEME_NOAUTH "rlsn"`

URL scheme when connecting to RLS server without authentication.

#### 4.5.2.3 `#define GLOBUS_RLS_SERVER_DEFPORT 39281`

Default port number that RLS server listens on.

#### 4.5.2.4 `#define MAXERRMSG 1024`

Maximum length of error messages returned by server.

## 4.5.3 Function Documentation

## 4.5.3.1 void globus\_rls\_client\_certificate (char certificate, char keyfile)

Set certificate used in authentication.

Sets environment variables X509\_USER\_CERT, X509\_USER\_KEY, and clears X509\_USER\_PROXY.

Parameters:

certificate Name of X509 certificate file.

keyfile Name of X509 key file.

## 4.5.3.2 void globus\_rls\_client\_proxy\_certificate (char proxy)

Set X509\_USER\_PROXY environment variable to specified file.

Parameters:

proxy Name of X509 proxy certificate file. If NULL clears X509\_USER\_PROXY.

## 4.5.3.3 globus\_result\_t globus\_rls\_client\_connect (char url, globus\_rls\_handle\_t \*h)

Open connection to RLS server.

Parameters:

url URL of server to connect to. URL scheme should be RLS or RLSN, eg RLS://my.host. If the URL scheme is RLSN then no authentication is performed (the RLS server must be started with authentication disabled as well, this option is primarily intended for testing).

h If the connection is successful h will be set to the connection handle. This handle is required by all other functions in the API.

Return values:

GLOBUS\_SUCCESS h now connected to RLS server identified by url.

## 4.5.3.4 globus\_result\_t globus\_rls\_client\_close (globus\_rls\_handle\_t h)

Close connection to RLS server.

Parameters:

h Connection handle to be closed, previously allocated by globus\_rls\_client\_connect()

Return values:

GLOBUS\_SUCCESS Connection closed, h is no longer valid.

## 4.5.3.5 int globus\_rls\_client\_get\_timeout ()

Get timeout for IO calls to RLS server.

If 0 IO calls do not timeout. The default is 30 seconds.

Return values:

timeout Seconds to wait before timing out an IO operation.



#### 4.5.3.6 void globus\_rls\_client\_set\_timeout(int seconds)

Set timeout for IO calls to RLS server.

Parameters:

seconds Seconds to wait before timing out an IO operation. If 0 IO calls do not timeout. The default is 30 seconds.

## 4.6 LRC Operations

Functions to view and update data managed by a LRC server.

### Data Structures

- struct `globus_rls_rli_info_t`  
Information about RLI server, returned by `globus_rls_client_lrc_rli_info()` and `globus_rls_client_lrc_rli_list()`

### Defines

- #define `FRLI_BLOOMFILTER` 0x1
- #define `MAXURL` 256

### Functions

- `globus_result_t globus_rls_client_lrc_attr_add(globus_rls_handle_t h, char key, globus_rls_attribute_t attr)`
- `globus_result_t globus_rls_client_lrc_attr_add_bulk(globus_rls_handle_t h, globus_list_t attr_obj_list, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_attr_create(globus_rls_handle_t h, char name, globus_rls_obj_type_t objtype, globus_rls_attr_type_t type)`
- `globus_result_t globus_rls_client_lrc_attr_delete(globus_rls_handle_t h, char name, globus_rls_obj_type_t objtype, globus_bool_t clearvalues)`
- `globus_result_t globus_rls_client_lrc_attr_get(globus_rls_handle_t h, char name, globus_rls_obj_type_t objtype, globus_list_t attr_list)`
- `globus_result_t globus_rls_client_lrc_attr_modify(globus_rls_handle_t h, char key, globus_rls_attribute_t attr)`
- `globus_result_t globus_rls_client_lrc_attr_remove(globus_rls_handle_t h, char key, globus_rls_attribute_t attr)`
- `globus_result_t globus_rls_client_lrc_attr_remove_bulk(globus_rls_handle_t h, globus_list_t attr_obj_list, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_attr_search(globus_rls_handle_t h, char name, globus_rls_obj_type_t objtype, globus_rls_attr_op_t op, globus_rls_attribute_t operand1, globus_rls_attribute_t operand2, int offset, int reslimit, globus_list_t attr_obj_list)`
- `globus_result_t globus_rls_client_lrc_attr_value_get(globus_rls_handle_t h, char key, char name, globus_rls_obj_type_t objtype, globus_list_t attr_list)`
- `globus_result_t globus_rls_client_lrc_attr_value_get_bulk(globus_rls_handle_t h, globus_list_t keylist, char name, globus_rls_obj_type_t objtype, globus_list_t attr_obj_list)`
- `globus_result_t globus_rls_client_lrc_add(globus_rls_handle_t h, char lfn, char pfn)`
- `globus_result_t globus_rls_client_lrc_add_bulk(globus_rls_handle_t h, globus_list_t str2_list, globus_list_t str2bulk_list)`

- `globus_result_t globus_rls_client_lrc_clear(globus_rls_handle_t)`
- `globus_result_t globus_rls_client_lrc_create(globus_rls_handle_t, char lfn, char pfn)`
- `globus_result_t globus_rls_client_lrc_create_bulk(globus_rls_handle_t, globus_list_t str2_list, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_delete(globus_rls_handle_t, char lfn, char pfn)`
- `globus_result_t globus_rls_client_lrc_delete_bulk(globus_rls_handle_t, globus_list_t str2_list, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_exists(globus_rls_handle_t, char key, globus_rls_obj_type objtype)`
- `globus_result_t globus_rls_client_lrc_exists_bulk(globus_rls_handle_t, globus_list_t keylist, globus_rls_obj_type_t objtype, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_get_lfn(globus_rls_handle_t, char pfn, int offset, int reslimit, globus_list_t str2_list)`
- `globus_result_t globus_rls_client_lrc_get_lfn_bulk(globus_rls_handle_t, globus_list_t pfnlist, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_get_lfn_w(globus_rls_handle_t, char pfn_pattern, globus_rls_pattern_type, int offset, int reslimit, globus_list_t str2_list)`
- `globus_result_t globus_rls_client_lrc_get_pfn(globus_rls_handle_t, char lfn, int offset, int reslimit, globus_list_t str2_list)`
- `globus_result_t globus_rls_client_lrc_get_pfn_bulk(globus_rls_handle_t, globus_list_t lfnlist, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_get_pfn_w(globus_rls_handle_t, char lfn_pattern, globus_rls_pattern_type, int offset, int reslimit, globus_list_t str2_list)`
- `globus_result_t globus_rls_client_lrc_mapping_exists(globus_rls_handle_t, char lfn, char pfn)`
- `globus_result_t globus_rls_client_lrc_rename(globus_rls_handle_t, char oldname, char newname)`
- `globus_result_t globus_rls_client_lrc_rename_lfn_bulk(globus_rls_handle_t, globus_list_t str2_list, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_rename_pfn(globus_rls_handle_t, char oldname, char newname)`
- `globus_result_t globus_rls_client_lrc_rename_pfn_bulk(globus_rls_handle_t, globus_list_t str2_list, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_rli_add(globus_rls_handle_t, char rli_url, int ags, char pattern)`
- `globus_result_t globus_rls_client_lrc_rli_delete(globus_rls_handle_t, char rli_url, char pattern)`
- `globus_result_t globus_rls_client_lrc_rli_get_part(globus_rls_handle_t, char rli_url, char pattern, globus_list_t str2_list)`
- `globus_result_t globus_rls_client_lrc_rli_info(globus_rls_handle_t, char rli_url, globus_rls_rli_info_t info)`
- `globus_result_t globus_rls_client_lrc_rli_list(globus_rls_handle_t, globus_list_t rliinfo_list)`

#### 4.6.1 Detailed Description

Functions to view and update data managed by a LRC server.

#### 4.6.2 De ne Documentation

##### 4.6.2.1 #de ne FRILI\_BLOOMFILTER 0x1

Update RLI using bloom filters (see `globus_rls_client_lrc_rli_add()`)

##### 4.6.2.2 #de ne MAXURL 256

Maximum length of URL string.

## 4.6.3 Function Documentation

4.6.3.1 `globus_result_t globus_qls_client_lrc_attr_add(globus_qls_handle_t h, char key, globus_qls_attribute_t attr)`

Add an attribute to an object in the LRC database.

Parameters:

h Handle connected to an RLS server.

key Logical or Physical File Name (LFN or PFN) that identifies object attribute should be added to.

attr Attribute to be added to object. name, objtype, type and val should be set in attr.

Return values:

GLOBUS\_SUCCESS Attribute successfully associated with object.

4.6.3.2 `globus_result_t globus_qls_client_lrc_attr_add_bulk(globus_qls_handle_t h, globus_list_t attr_obj_list, globus_list_t str2bulk_list)`

Bulk add attributes to objects in the LRC database.

Parameters:

h Handle connected to an RLS server.

attr\_obj\_list List of object names (LFN or PFN) and attributes to be added. Each list datum should be of type `globus_qls_attribute_object_t`

str2bulk\_list List of failed updates. Each list datum is `globus_qls_string2_bulk_t` structure. str2.s1 will be the object name, str2.s2 the attribute name, and str2.s3 will be the result code from the failed update. Only failed updates will be returned.

4.6.3.3 `globus_result_t globus_qls_client_lrc_attr_create(globus_qls_handle_t h, char name, globus_qls_obj_type_t objtype, globus_qls_attr_type_t type)`

Define new attribute in LRC database.

Parameters:

h Handle connected to an LRC server.

name Name of attribute.

objtype Object (LFN or PFN) type that attribute applies to.

type Type of attribute value.

Return values:

GLOBUS\_SUCCESS Attribute successfully created.

4.6.3.4 `globus_result_t globus_qls_client_lrc_attr_delete(globus_qls_handle_t h, char name, globus_qls_obj_type_t objtype, globus_bool_t clearvalue)`

Delete attribute in LRC database, previously created with `globus_qls_client_lrc_attr_create()`

Parameters:

h Handle connected to an LRC server.

name Name of attribute.

objtype Object (LFN or PFN) type that attribute applies to.

clearvalues If GLOBUS\_TRUE then any any values for this attribute are rst removed from the objects they're associated with. If GLOBUS\_FALSE and any values exist then GLOBUS\_RLS\_ATTR\_EXISTS is returned.

Return values:

GLOBUS\_SUCCESS Attribute successfully removed.

4.6.3.5 globus\_result\_t globus\_uls\_client\_lrc\_attr\_get(globus\_uls\_handle\_t h, char name, globus\_uls\_obj\_type\_t objtype, globus\_list\_t attr\_list)

Return definitions of attributes in LRC database.

Parameters:

h Handle connected to an RLS server.

name Name of attribute. If name is NULL all attributes of the specified objtype are returned.

objtype Object (LFN or PFN) type that attribute applies to.

attr\_list Any attribute definitions found will be returned as a list of globus\_uls\_attribute\_t structures.

Return values:

GLOBUS\_SUCCESS Attribute definitions successfully retrieved attr\_list should be freed with globus\_uls\_client\_free\_list() when it is no longer needed.

4.6.3.6 globus\_result\_t globus\_uls\_client\_lrc\_attr\_modify(globus\_uls\_handle\_t h, char key, globus\_uls\_attribute\_t attr)

Modify an attribute value.

Parameters:

h Handle connected to an RLS server.

key Name of object (LFN or PFN).

attr Attribute to be modified. The objtype, name and type fields should be set in attr to identify the attribute, the val field should be the new value.

Return values:

GLOBUS\_SUCCESS Attribute successfully modified.

4.6.3.7 globus\_result\_t globus\_uls\_client\_lrc\_attr\_remove(globus\_uls\_handle\_t h, char key, globus\_uls\_attribute\_t attr)

Remove an attribute from an object (LFN or PFN) in the LRC database.

Parameters:

h Handle connected to an RLS server.

key Name of object (LFN or PFN).

attr Attribute to be removed. The objtype and name fields should be set in attr to identify the attribute.

Return values:

GLOBUS\_SUCCESS Attribute successfully removed.

4.6.3.8 `globus_result_t globus_uls_client_lrc_attr_remove_bulk(globus_uls_handle_t h, globus_list_t attr_obj_list, globus_list_t str2bulk_list)`

Bulk remove attributes from objects in the LRC database.

Parameters:

h Handle connected to an RLS server.

attr\_obj\_list List of object names (LFN or PFN) and attributes to be removed. It is not necessary to set the attribute type or value. Each list datum should be of type `globus_uls_attribute_object_t`

str2bulk\_list List of failed updates. Each list datum is a `globus_uls_string2_bulk_t` structure. str2.s1 will be the object name, str2.s2 the attribute name, and str2.s3 will be the result code from the failed update. Only failed updates will be returned.

4.6.3.9 `globus_result_t globus_uls_client_lrc_attr_search(globus_uls_handle_t h, char name, globus_uls_obj_type_t objtype, globus_uls_attr_op_t op, globus_uls_attribute_t operand1, globus_uls_attribute_t operand2, int offset, int reslimit, globus_list_t attr_obj_list)`

Search for objects (LFNs or PFNs) in a LRC database that have the specified attribute whose value matches a boolean expression.

Parameters:

h Handle connected to an RLS server.

name Name of attribute.

objtype Object (LFN or PFN) type that attribute applies to.

op Operator to be used in searching for values.

operand1 First operand in boolean expression, type and val should be set in `globus_uls_attribute_t`

operand2 Second operand in boolean expression, only used when `globus_uls_client_attr_op_bt` is `type` and `val` should be set in `globus_uls_attribute_t`

offset Offset into result list. Used in conjunction with reslimit to retrieve results a few at a time. Use 0 to begin with first result. If NULL then API will handle accumulating partial results transparently.

reslimit Maximum number of results to return. Used in conjunction with offset to retrieve results a few at a time. Use 0 to retrieve all results.

attr\_obj\_list Any objects with the specified attribute will be returned, with the attribute value, in a list of `globus_uls_attribute_object_t` structures.

Return values:

GLOBUS\_SUCCESS Objects with specified attribute returned in attr\_obj\_list. attr\_obj\_list should be freed with `globus_uls_client_free_list()` when it is no longer needed. See [Query Results](#)

4.6.3.10 `globus_result_t globus_uls_client_lrc_attr_value_get(globus_uls_handle_t h, char key, char name, globus_uls_obj_type_t objtype, globus_list_t attr_list)`

Return attributes in LRC database for specified object (LFN or PFN).

Parameters:

h Handle connected to an RLS server.

key Logical or Physical File Name (LFN or PFN) that identifies object attributes should be retrieved for.

name Name of attribute to retrieve. If NULL all attributes for key, objtype are returned.

objtype Object (LFN or PFN) type that attribute applies to.

attr\_list Any attributes found will be returned in this list [gobus\\_rls\\_attribute](#) structures.

Return values:

GLOBUS\_SUCCESS Attributes successfully retrieved attr\_list should be freed with [gobus\\_rls\\_client\\_free\\_list\(\)](#) when it is no longer needed.

4.6.3.11 `gobus_result_t gobus_rls_client_lrc_attr_value_get_bulk(gobus\_rls\_handle\_t h, gobus_list_t keylist, char name, gobus\_rls\_obj\_type\_t objtype, gobus_list_t attr_obj_list)`

Return attributes in LRC database for specified objects (LFN or PFN).

Parameters:

h Handle connected to an RLS server.

keylist Logical or Physical File Names (LFNs or PFNs) that identify object attributes should be retrieved for.

Each list datum should be a string containing the LFN or PFN.

name Name of attribute to retrieve. If NULL all attributes for key, objtype are returned.

objtype Object (LFN or PFN) type that attribute applies to.

attr\_obj\_list Any attributes found will be returned in this list [gobus\\_rls\\_attribute\\_object](#) structures.

Return values:

GLOBUS\_SUCCESS Attributes successfully retrieved attr\_obj\_list should be freed with [gobus\\_rls\\_client\\_free\\_list\(\)](#) when it is no longer needed.

4.6.3.12 `gobus_result_t gobus_rls_client_lrc_add(gobus\_rls\_handle\_t h, char lfn, char pfn)`

Add mapping to PFN to an existing LFN.

Parameters:

h Handle connected to an RLS server.

lfn LFN to add pfn mapping to, should already exist.

pfn PFN that lfn should map to.

Return values:

GLOBUS\_SUCCESS New mapping created.

4.6.3.13 `gobus_result_t gobus_rls_client_lrc_add_bulk(gobus\_rls\_handle\_t h, gobus_list_t str2_list, gobus_list_t str2bulk_list)`

Bulk add LFN, PFN mappings in LRC database.

LFNs must already exist.

Parameters:

h Handle connected to an RLS server.

str2\_list LFN, PFN pairs to add mappings.

str2bulk\_list List of failed updates. Each list datum is [gobus\\_rls\\_string2\\_bulk](#) structure. str2.s1 will be the LFN, and str2.s2 the PFN it maps to, and str2.c will be the result code from the failed update. Only failed updates will be returned.

#### 4.6.3.14 globus\_result\_t globus\_uls\_client\_lrc\_clear(globus\_uls\_handle\_t h)

Clear all mappings from LRC database.

User needs both ADMIN and LRCUPDATE privileges to perform this operation. Note that if the LRC is cleared this will not be reflected in any RLI servers updated by the LRC until the next softstate update, even if immediate updates are enabled.

Parameters:

h Handle connected to an RLS server.

Return values:

GLOBUS\_SUCCESS Mappings cleared.

#### 4.6.3.15 globus\_result\_t globus\_uls\_client\_lrc\_create(globus\_uls\_handle\_t h, char lfn, char pfn)

Create mapping between a LFN and PFN.

LFN should not exist yet.

Parameters:

h Handle connected to an RLS server.

lfn LFN to add pfn mapping to, should not already exist.

pfn PFN that lfn should map to.

Return values:

GLOBUS\_SUCCESS New mapping created.

#### 4.6.3.16 globus\_result\_t globus\_uls\_client\_lrc\_create\_bulk(globus\_uls\_handle\_t h, globus\_list\_t str2\_list, globus\_list\_t str2bulk\_list)

Bulk create LFN,PFN mappings in LRC database.

Parameters:

h Handle connected to an RLS server.

str2\_list LFN,PFN pairs to create mappings for.

str2bulk\_list List of failed updates. Each list datum is a globus\_uls\_string2\_bulk structure. str2.s1 will be the LFN, and str2.s2 the PFN it maps to, and str2.c will be the result code from the failed update. Only failed updates will be returned.

#### 4.6.3.17 globus\_result\_t globus\_uls\_client\_lrc\_delete(globus\_uls\_handle\_t h, char lfn, char pfn)

Delete mapping between LFN and PFN.

Parameters:

h Handle connected to an RLS server.

lfn LFN to remove mapping from.

pfn PFN that lfn maps to that is being removed.

Return values:

GLOBUS\_SUCCESS Mapping removed.

4.6.3.18 `globus_result_t globus_uls_client_lrc_delete_bulk(globus_uls_handle_t h, globus_list_t str2_list, globus_list_t str2bulk_list)`

Bulk delete LFN,PFN mappings in LRC database.

Parameters:

h Handle connected to an RLS server.

str2\_list LFN,PFN pairs to add mappings.

str2bulk\_list List of failed updates. Each list datum is `globus_uls_string2_bulk_t` structure. str2.s1 will be the LFN, and str2.s2 the PFN it maps to, and c will be the result code from the failed update. Only failed updates will be returned.

4.6.3.19 `globus_result_t globus_uls_client_lrc_exists(globus_uls_handle_t h, char key, globus_uls_obj_type_t objtype)`

Check if an object exists in the LRC database.

Parameters:

h Handle connected to an RLS server.

key LFN or PFN that identifies object.

objtype Type of object key refers to `globus_uls_obj_lrc_lfn` or `globus_uls_obj_lrc_pfn`

Return values:

GLOBUS\_SUCCESS Object exists.

4.6.3.20 `globus_result_t globus_uls_client_lrc_exists_bulk(globus_uls_handle_t h, globus_list_t keylist, globus_uls_obj_type_t objtype, globus_list_t str2bulk_list)`

Bulk check if objects exist in the LRC database.

Parameters:

h Handle connected to an RLS server.

keylist LFNs or PFNs that identify objects.

objtype Type of object key refers to `globus_uls_obj_lrc_lfn` or `globus_uls_obj_lrc_pfn`

str2bulk\_list Results of existence check. Each list datum will be `globus_uls_string2_bulk_t` structure. str2.s1 will be the LFN or PFN, and str2.s2 empty, and c will be the result code indicating existence.

4.6.3.21 `globus_result_t globus_uls_client_lrc_get_lfn(globus_uls_handle_t h, char pfn, int offset, int reslimit, globus_list_t str2_list)`

Return LFNs mapped to PFN in the LRC database.

Parameters:

h Handle connected to an RLS server.

pfn PFN to search for.

offset Offset into result list. Used in conjunction with reslimit to retrieve results a few at a time. Use 0 to begin with first result. If NULL then API will handle accumulating partial results transparently.



**reslimit** Maximum number of results to return. Used in conjunction with **offset** to retrieve results a few at a time. Use 0 to retrieve all results.

**str2\_list** List of LFNs that map to **pfns**. Each list datum will be a `globus_uls_string2_t` structure. **s1** will be the LFN, and **s2** the PFN it maps to.

Return values:

**GLOBUS\_SUCCESS** List of LFNs that map to **pfns** in **str2\_list**. See [Query Results](#)

4.6.3.22 `globus_result_t globus_uls_client_lrc_get_lfn_bulk(globus_uls_handle_t h, globus_list_t pfnslist, globus_list_t str2bulk_list)`

Bulk return LFNs mapped to PFN in the LRC database.

Parameters:

**h** Handle connected to an RLS server.

**pfnslist** PFNs to search for.

**str2bulk\_list** Results of queries. Each list datum will be a `globus_uls_string2_bulk_t` structure. **str2.s1** will be the LFN, and **str2.s2** the PFN it maps to, and **rc** will be the result code from the query.

4.6.3.23 `globus_result_t globus_uls_client_lrc_get_lfn_wcd(globus_uls_handle_t h, char pfns_pattern, globus_uls_pattern_t type, int offset, int reslimit, globus_list_t str2_list)`

Return LFNs mapped to wildcarded PFN in the LRC database.

Parameters:

**h** Handle connected to an RLS server.

**pfns\_pattern** PFN pattern to search for.

**type** Identifies wildcard characters used in **pfns\_pattern**. Wildcard chars can be Unix like globbing chars ( **\* matches 0 or more characters, ? matches any single character**) or SQL "like" wildcard characters ( **% matches 0 or more characters, \_ matches any single character**).

**offset** Offset into result list. Used in conjunction with **reslimit** to retrieve results a few at a time. Use 0 to begin with first result. If NULL then the API will handle accumulating partial results transparently.

**reslimit** Maximum number of results to return. Used in conjunction with **offset** to retrieve results a few at a time. Use 0 to retrieve all results.

**str2\_list** List of LFNs that map to **pfns\_pattern**. Each list datum will be a `globus_uls_string2_t` structure. **s1** will be the LFN, and **s2** the PFN it maps to.

Return values:

**GLOBUS\_SUCCESS** List of LFNs that map to **pfns\_pattern** in **str2\_list**. See [Query Results](#)

4.6.3.24 `globus_result_t globus_uls_client_lrc_get_pfn(globus_uls_handle_t h, char lfn, int offset, int reslimit, globus_list_t str2_list)`

Return PFNs mapped to LFN in the LRC database.

Parameters:

**h** Handle connected to an RLS server.

lfn LFN to search for.

offset Offset into result list. Used in conjunction with reslimit to retrieve results a few at a time. Use 0 to begin with rst result.

reslimit Maximum number of results to return. Used in conjunction with offset to retrieve results a few at a time. Use 0 to retrieve all results.

str2\_list List of PFNs that map to lfn. Each list datum will be a [globus\\_rls\\_string2\\_t](#) structure. s1 will be the LFN, and s2 the PFN it maps to.

Return values:

GLOBUS\_SUCCESS List of PFNs that map to lfn in str2\_list

4.6.3.25 [globus\\_result\\_t](#) [globus\\_rls\\_client\\_lrc\\_get\\_pfn\\_bulk](#)([globus\\_rls\\_handle\\_t](#) h, [globus\\_list\\_t](#) lfnlist, [globus\\_list\\_t](#) str2bulk\_list)

Bulk return PFNs mapped to LFN in the LRC database.

Parameters:

h Handle connected to an RLS server.

lfnlist LFNs to search for.

str2bulk\_list Results of queries. Each list datum will be a [globus\\_rls\\_string2\\_bulk\\_t](#) structure. str2.s1 will be the LFN, and str2.s2 the PFN it maps to, and rc will be the result code from the query.

4.6.3.26 [globus\\_result\\_t](#) [globus\\_rls\\_client\\_lrc\\_get\\_pfn\\_wc](#)([globus\\_rls\\_handle\\_t](#) h, char lfn\_pattern, [globus\\_rls\\_pattern\\_t](#) type, int offset, int reslimit, [globus\\_list\\_t](#) str2\_list)

Return PFNs mapped to wildcarded LFN in the LRC database.

Parameters:

h Handle connected to an RLS server.

lfn\_pattern LFN pattern to search for.

type Identifies wildcard characters used in lfn\_pattern Wildcard chars can be Unix like globbing chars ( \* matches 0 or more characters, ? matches any single character) or SQL "like" wildcard characters ( % matches 0 or more characters, \_ matches any single character).

offset Offset into result list. Used in conjunction with reslimit to retrieve results a few at a time. Use 0 to begin with rst result.

reslimit Maximum number of results to return. Used in conjunction with offset to retrieve results a few at a time. Use 0 to retrieve all results.

str2\_list List of PFNs that map to lfn\_pattern Each list datum will be a [globus\\_rls\\_string2\\_t](#) structure. s1 will be the LFN, and s2 the PFN it maps to.

Return values:

GLOBUS\_SUCCESS List of PFNs that map to lfn\_pattern in str2\_list See [Query Results](#)

4.6.3.27 `globus_result_t globus_uls_client_lrc_mapping_exists(globus_uls_handle_t h, char lfn, char pfn)`

Check if a mapping exists in the LRC database.

Parameters:

- h Handle connected to an RLS server.
- lfn LFN of mapping.
- pfn PFN of mapping.

Return values:

GLOBUS\_SUCCESS Object exists.

4.6.3.28 `globus_result_t globus_uls_client_lrc_renamefn(globus_uls_handle_t h, char oldname, char newname)`

Rename LFN.

If immediate mode is ON, the LRC will send update messages to associated RLIs.

Parameters:

- h Handle connected to an RLS server.
- oldname Existing LFN name, to be renamed.
- newname New LFN name, to replace existing name.

Return values:

GLOBUS\_SUCCESS LFN renamed.

4.6.3.29 `globus_result_t globus_uls_client_lrc_renamefn_bulk(globus_uls_handle_t h, globus_list_t str2_list, globus_list_t str2bulk_list)`

Bulk rename LFN names in LRC database.

LFNs must already exist. If immediate mode is ON, the LRC will send update messages to associated RLIs.

Parameters:

- h Handle connected to an RLS server.
- str2\_list oldname,newname pairs such that newname replaces oldname for LFNs.
- str2bulk\_list List of failed updates. Each list datum is `globus_uls_string2_bulk_t` structure. str2.s1 will be the old LFN name, and str2.s2 the new LFN name, and str2.c will be the result code from the failed update. Only failed updates will be returned.

4.6.3.30 `globus_result_t globus_uls_client_lrc_renamepfn(globus_uls_handle_t h, char oldname, char newname)`

Rename PFN.

Parameters:

- h Handle connected to an RLS server.
- oldname Existing PFN name, to be renamed.
- newname New PFN name, to replace existing name.

Return values:

GLOBUS\_SUCCESS PFN renamed.

4.6.3.31 `globus_result_t globus_rls_client_lrc_renamepfns_bulk(globus_rls_handle_t h, globus_list_t str2_list, globus_list_t str2bulk_list)`

Bulk rename PFN names in LRC database.

PFNs must already exist.

Parameters:

`h` Handle connected to an RLS server.

`str2_list` oldname,newname pairs such that newname replaces oldname for PFNs.

`str2bulk_list` List of failed updates. Each list datum is a `globus_rls_string2_bulk_t` structure. `str2.s1` will be the old PFN name, and `str2.s2` the new PFN name, and `str2.s3` will be the result code from the failed update. Only failed updates will be returned.

4.6.3.32 `globus_result_t globus_rls_client_lrc_rli_add(globus_rls_handle_t h, char rli_url, int ags, char pattern)`

LRC servers send information about LFNs in their database to the the list of RLI servers in the database, added with the following function.

Updates may be partitioned amongst multiple RLIs by specifying one or more patterns for an RLI.

Parameters:

`h` Handle connected to an RLS server.

`rli_url` URL of RLI server that LRC should send updates to.

`ags` Should be zero or `RLI_BLOOMFILTER`.

`pattern` If not NULL used to filter which LFNs are sent to `rli_url`. Standard Unix wildcard characters (?) may be used to do wildcard matches.

Return values:

`GLOBUS_SUCCESSRLI` (with pattern if not NULL) added to LRC database.

4.6.3.33 `globus_result_t globus_rls_client_lrc_rli_delete(globus_rls_handle_t h, char rli_url, char pattern)`

Delete an entry from the LRC rli/partition tables.

Parameters:

`h` Handle connected to an RLS server.

`rli_url` URL of RLI server to remove from LRC partition table.

`pattern` If not NULL then only the specified `rli_url/pattern` is removed, else all partition information for `rli_url` is removed.

Return values:

`GLOBUS_SUCCESSRLI` and pattern (if specified) removed from LRC partition table.

4.6.3.34 `globus_result_t globus_rls_client_lrc_rli_get_partitions(globus_rls_handle_t h, char rli_url, char pattern, globus_list_t str2_list)`

Get RLI update partitions from LRC server.

Parameters:

h Handle connected to an RLS server.

rli\_url If not NULL identifies RLI that partition data will be retrieved for. If NULL then all RLIs are retrieved.

pattern If not NULL returns only partitions with matching patterns, otherwise all patterns are retrieved.

str2\_list Results added to list. Datums in str2\_list are of type `globus_rls_string2_t` structure. s1 will be the rli url, s2 an empty string or the pattern used to partition updates. [See Results](#)

Return values:

GLOBUS\_SUCCESS Partition data retrieved from server, written to str2\_list

4.6.3.35 `globus_result_t globus_rls_client_lrc_rli_info(globus_rls_handle_t h, char rli_url, globus_rls_rli_info_t info)`

Get info about RLI server updated by an LRC server.

Parameters:

h Handle connected to an RLS server.

rli\_url URL of RLI server to retrieve info for.

info Data about RLI server will be written here.

Return values:

GLOBUS\_SUCCESS Info about RLI server successfully retrieved.

4.6.3.36 `globus_result_t globus_rls_client_lrc_rli_list(globus_rls_handle_t h, globus_list_t rliinfo_list)`

Return URLs of RLIs that LRC sends updates to.

Parameters:

h Handle connected to an RLS server.

rliinfo\_list List of RLIs updated by this LRC returned in this list. Each list datum is of type `globus_rls_rli_info_t`. rliinfo\_list should be freed with `globus_rls_client_free_list()` when no longer needed.

Return values:

GLOBUS\_SUCCESS List of RLIs updated by this LRC returned in rliinfo\_list.

## 4.7 RLI Operations

Functions to view and update data managed by a RLI server.

Data Structures

- struct `globus_rls_sender_info_t`

Information about server sending updates to an rli, returned by `globus_rls_client_rli_sender_list()`

## Functions

- `globus_result_t globus_rls_client_rli_exists(globus_rls_handle_t h, char key, globus_rls_obj_type_t objtype)`
- `globus_result_t globus_rls_client_rli_exists_bulk(globus_rls_handle_t h, globus_list_t keylist, globus_rls_obj_type_t objtype, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_rli_get_lfn(globus_rls_handle_t h, char lfn, int offset, int reslimit, globus_list_t str2_list)`
- `globus_result_t globus_rls_client_rli_get_lrc_bulk(globus_rls_handle_t h, globus_list_t lfnlist, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_rli_get_lrc_wc(globus_rls_handle_t h, char lfn_pattern, globus_rls_pattern_type_t type, int offset, int reslimit, globus_list_t str2_list)`
- `globus_result_t globus_rls_client_rli_sender_list(globus_rls_handle_t h, globus_list_t senderinfo_list)`
- `globus_result_t globus_rls_client_rli_lrc_list(globus_rls_handle_t h, globus_list_t lrcinfo_list)`
- `globus_result_t globus_rls_client_rli_mapping_exists(globus_rls_handle_t h, char lfn, char lrc)`
- `globus_result_t globus_rls_client_rli_rli_add(globus_rls_handle_t h, char rli_url, char pattern)`
- `globus_result_t globus_rls_client_rli_rli_delete(globus_rls_handle_t h, char rli_url, char pattern)`
- `globus_result_t globus_rls_client_rli_rli_get_params(globus_rls_handle_t h, char rli_url, char pattern, globus_list_t str2_list)`
- `globus_result_t globus_rls_client_rli_rli_list(globus_rls_handle_t h, globus_list_t rliinfo_list)`

## 4.7.1 Detailed Description

Functions to view and update data managed by a RLI server.

## 4.7.2 Function Documentation

4.7.2.1 `globus_result_t globus_rls_client_rli_exists(globus_rls_handle_t h, char key, globus_rls_obj_type_t objtype)`

Check if an object exists in the RLI database.

Parameters:

h Handle connected to an RLS server.

key LFN or LRC that identifies object.

objtype Type of objectkey refers to `globus_rls_obj_rli_lfn` or `globus_rls_obj_rli_lrc`.

Return values:

GLOBUS\_SUCCESS Object exists.

4.7.2.2 `globus_result_t globus_rls_client_rli_exists_bulk(globus_rls_handle_t h, globus_list_t keylist, globus_rls_obj_type_t objtype, globus_list_t str2bulk_list)`

Bulk check if objects exist in the RLI database.

Parameters:

h Handle connected to an RLS server.

keylist LFNs or LRCs that identify objects.

objtype Type of objectkey refers to `globus_rls_obj_rli_lfn` or `globus_rls_obj_rli_lrc`.

str2bulk\_list Results of existence check. Each list datum will be a `globus_rls_string2_bulk` structure. str2.s1 will be the LFN or LRC, and str2.s2 empty, and c will be the result code indicating existence.

4.7.2.3 `globus_result_t globus_rls_client_rli_get_lrc(globus\_rls\_handle\_t h, char lfn, int offset, int reslimit, globus\_list\_t str2_list)`

Return LRCs mapped to LFN in the RLI database.

Parameters:

h Handle connected to an RLS server.

lfn LFN whose list of LRCs is desired.

offset Offset into result list. Used in conjunction with `reslimit` to retrieve results a few at a time. Use 0 to begin with first result.

reslimit Maximum number of results to return. Used in conjunction with `offset` to retrieve results a few at a time. Use 0 to retrieve all results.

str2\_list List of LRCs that lfn maps to. Each list datum will be `globus_rls_string2_t` structure. s1 will be the LFN, and s2 the LRC it maps to. str2\_list should be freed with `globus_rls_client_free_list()`

Return values:

GLOBUS\_SUCCESS List of LRCs that map to lfn in str2\_list. See [Query Results](#)

4.7.2.4 `globus_result_t globus_rls_client_rli_get_lrc_bulk(globus\_rls\_handle\_t h, globus\_list\_t lfnlist, globus\_list\_t str2bulk_list)`

Bulk return LRCs mapped to LFN in the RLI database.

Parameters:

h Handle connected to an RLS server.

lfnlist LFNs to search for.

str2bulk\_list Results of queries. Each list datum will be `globus_rls_string2_bulk_t` structure. str2.s1 will be the LFN, and str2.s2 the LRC it maps to, and str2.s3 will be the result code from the query.

4.7.2.5 `globus_result_t globus_rls_client_rli_get_lrc_wild(globus\_rls\_handle\_t h, char lfn_pattern, globus\_rls\_pattern\_t type, int offset, int reslimit, globus\_list\_t str2_list)`

Return LRCs mapped to wildcarded LFN in the RLI database.

Parameters:

h Handle connected to an RLS server.

lfn\_pattern LFN pattern to search for.

type Identifies wildcard characters used in lfn\_pattern. Wildcard chars can be Unix file globbing chars (matches 0 or more characters, ? matches any single character) or SQL "like" wildcard characters (matches 0 or more characters, \_ matches any single character).

offset Offset into result list. Used in conjunction with `reslimit` to retrieve results a few at a time. Use 0 to begin with first result.

reslimit Maximum number of results to return. Used in conjunction with `offset` to retrieve results a few at a time. Use 0 to retrieve all results.

str2\_list List of LRCs that map to lfn\_pattern. Each list datum will be `globus_rls_string2_t`. s1 will be the LFN, and s2 the LRC it maps to. str2\_list should be freed with `globus_rls_client_free_list()`

Return values:

GLOBUS\_SUCCESS List of LRCs that map to lfn\_pattern in str2\_list. See [Query Results](#)

4.7.2.6 `globus_result_t globus_rls_client_rli_sender_list(globus_rls_handle_t h, globus_list_t senderinfo_list)`

Get list of servers updating this RLI server.

Similar to `globus_rls_client_rli_get_part()` except no partition information is returned.

Parameters:

h Handle connected to an RLS server.

senderinfo\_list Datums in senderinfo\_list will be of type `globus_rls_sender_info_t`. senderinfo\_list should be freed with `globus_rls_client_free_list()`

Return values:

GLOBUS\_SUCCESS List of LRCs updating RLI added to senderinfo\_list

4.7.2.7 `globus_result_t globus_rls_client_rli_lrc_list(globus_rls_handle_t h, globus_list_t lrcinfo_list)`

Deprecated, use `globus_rls_client_rli_sender_list()`

Parameters:

h Handle connected to an RLS server.

lrcinfo\_list Datums in lrcinfo\_list will be of type `globus_rls_lrc_info_t`. lrcinfo\_list should be freed with `globus_rls_client_free_list()`

Return values:

GLOBUS\_SUCCESS List of LRCs updating RLI added to lrcinfo\_list.

4.7.2.8 `globus_result_t globus_rls_client_rli_mapping_exists(globus_rls_handle_t h, char lfn, char lrc)`

Check if a mapping exists in the RLI database.

Parameters:

h Handle connected to an RLS server.

lfn LFN of mapping.

lrc LRC of mapping.

Return values:

GLOBUS\_SUCCESS Mapping exists.

4.7.2.9 `globus_result_t globus_rls_client_rli_rli_add(globus_rls_handle_t h, char rli_url, char pattern)`

RLI servers send information about LFNs in their database to the the list of RLI servers in the database, added with the following function.

Updates may be partitioned amongst multiple RLIs by specifying one or more patterns for an RLI.

Parameters:

h Handle connected to an RLS server.



rli\_url URL of RLI server that LRC should send updates to.

pattern If not NULL used to filter which LFNs are sent to rli\_url. Standard Unix wildcard characters (?) may be used to do wildcard matches.

Return values:

GLOBUS\_SUCCESS RLI (with pattern if not NULL) added to RLI database.

4.7.2.10 globus\_result\_t globus\_rls\_client\_rli\_rli\_delete(globus\_rls\_handle\_t h, char rli\_url, char pattern)

Delete an entry from the RLI rli/partition tables.

Parameters:

h Handle connected to an RLS server.

rli\_url URL of RLI server to remove from RLI partition table.

pattern If not NULL then only the specified rli\_url/pattern is removed, else all partition information for rli\_url is removed.

Return values:

GLOBUS\_SUCCESS RLI and pattern (if specified) removed from LRC partition table.

4.7.2.11 globus\_result\_t globus\_rls\_client\_rli\_rli\_get\_partitions(globus\_rls\_handle\_t h, char rli\_url, char pattern, globus\_list\_t str2\_list)

Get RLI update partitions from RLI server.

Parameters:

h Handle connected to an RLS server.

rli\_url If not NULL identifies RLI that partition data will be retrieved for. If NULL then all RLIs are retrieved.

pattern If not NULL returns only partitions with matching patterns, otherwise all patterns are retrieved.

str2\_list Results added to list. Datums in str2\_list are of type globus\_rls\_string2\_t structure. s1 will be the rli url, s2 an empty string or the pattern used to partition updates. See Results

Return values:

GLOBUS\_SUCCESS Partition data retrieved from server, written to str2\_list

4.7.2.12 globus\_result\_t globus\_rls\_client\_rli\_rli\_list\_urls(globus\_rls\_handle\_t h, globus\_list\_t rliinfo\_list)

Return URLs of RLIs that RLI sends updates to.

Parameters:

h Handle connected to an RLS server.

rliinfo\_list List of RLIs updated by this RLI returned in this list. Each list datum is of type globus\_rls\_rli\_info\_t. rliinfo\_list should be freed with globus\_rls\_client\_free\_list() when no longer needed.

Return values:

GLOBUS\_SUCCESS List of RLIs updated by this LRC returned in rliinfo\_list.

## 5 globus\_rls client Data Structure Documentation

### 5.1 globus\_rls\_attribute\_object\_t Struct Reference

[globus\\_rls\\_client\\_lrc\\_attr\\_search\(\)](#) returns a list of these structures which include the object name (LFN or PFN) and attribute value found by the query.

#### Data Fields

- [globus\\_rls\\_attribute\\_attr](#)
- char [key](#)
- int [rc](#)

#### 5.1.1 Detailed Description

[globus\\_rls\\_client\\_lrc\\_attr\\_search\(\)](#) returns a list of these structures which include the object name (LFN or PFN) and attribute value found by the query.

#### 5.1.2 Field Documentation

##### 5.1.2.1 [globus\\_rls\\_attribute\\_t](#) [globus\\_rls\\_attribute\\_object\\_t::attr](#)

Attribute value.

##### 5.1.2.2 char [globus\\_rls\\_attribute\\_object\\_t::key](#)

LFN or PFN.

##### 5.1.2.3 int [globus\\_rls\\_attribute\\_object\\_t::rc](#)

Result code, only used in bulk query.

### 5.2 globus\_rls\_attribute\_t Struct Reference

Object (LFN or PFN) attribute type.

#### Data Fields

- char [name](#)
- [globus\\_rls\\_obj\\_type\\_t](#) [objtype](#)
- [globus\\_rls\\_attr\\_type\\_t](#) [type](#)
- union {
  - [time\\_t](#) [time](#)
  - [double](#) [doubled](#)
  - int [i](#)
  - char [s](#)
- [val](#)

### 5.2.1 Detailed Description

Object (LFN or PFN) attribute type.

### 5.2.2 Field Documentation

#### 5.2.2.1 char globus\_rls\_attribute\_t::name

Attribute name.

#### 5.2.2.2 globus\_rls\_obj\_type\_t globus\_rls\_attribute\_t::objtype

Object type.

#### 5.2.2.3 globus\_rls\_attr\_type\_t globus\_rls\_attribute\_t::type

Attribute value type.

#### 5.2.2.4 time\_t globus\_rls\_attribute\_t::t

Date value (unix time).

#### 5.2.2.5 double globus\_rls\_attribute\_t::d

Floating point value.

#### 5.2.2.6 int globus\_rls\_attribute\_t::i

Integer value.

#### 5.2.2.7 char globus\_rls\_attribute\_t::s

String value.

#### 5.2.2.8 union { ... } globus\_rls\_attribute\_t::val

Value of attribute (depends on type).

## 5.3 globus\_rls\_handle\_t Struct Reference

RLS Client Handle.

### Data Fields

- globus\_url\_t url
- globus\_io\_handle\_t handle
- int ags

### 5.3.1 Detailed Description

RLS Client Handle.

### 5.3.2 Field Documentation

#### 5.3.2.1 globus\_url\_t [globus\\_rls\\_handle\\_t::url](#)

URL of RLS server (RLS://host[:port]).

#### 5.3.2.2 globus\_io\_handle\_t [globus\\_rls\\_handle\\_t::handle](#)

Globus IO handle.

#### 5.3.2.3 int [globus\\_rls\\_handle\\_t::ags](#)

See FH\_xxx ags below.

## 5.4 globus\_rls\_rli\_info\_t Struct Reference

Information about RLI server, returned by [globus\\_rls\\_client\\_lrc\\_rli\\_info\(\)](#) and [globus\\_rls\\_client\\_lrc\\_rli\\_list\(\)](#)

### Data Fields

- char [url](#) [256]
- int [updateinterval](#)
- int [ags](#)
- time\_t [lastupdate](#)

### 5.4.1 Detailed Description

Information about RLI server, returned by [globus\\_rls\\_client\\_lrc\\_rli\\_info\(\)](#) and [globus\\_rls\\_client\\_lrc\\_rli\\_list\(\)](#)

### 5.4.2 Field Documentation

#### 5.4.2.1 char [globus\\_rls\\_rli\\_info\\_t::url](#) [ 256 ]

URL of server.

#### 5.4.2.2 int [globus\\_rls\\_rli\\_info\\_t::updateinterval](#)

Interval between softstate updates.

#### 5.4.2.3 int [globus\\_rls\\_rli\\_info\\_t::ags](#)

RLI ags (see [FRLI\\_BLOOMFILTER](#)).

#### 5.4.2.4 time\_t [globus\\_rls\\_rli\\_info\\_t::lastupdate](#)

Time of last softstate update.

## 5.5 globus\_rls\_sender\_info\_t Struct Reference

Information about server sending updates to an rli, returned by [globus\\_rls\\_client\\_rli\\_sender\\_list\(\)](#)

### Data Fields

- char [url](#) [256]
- time\_t [lastupdate](#)

#### 5.5.1 Detailed Description

Information about server sending updates to an rli, returned by [globus\\_rls\\_client\\_rli\\_sender\\_list\(\)](#)

#### 5.5.2 Field Documentation

##### 5.5.2.1 char [globus\\_rls\\_sender\\_info\\_t::url](#) [ 256 ]

URL of server.

##### 5.5.2.2 time\_t [globus\\_rls\\_sender\\_info\\_t::lastupdate](#)

Time of last softstate update.

## 5.6 globus\_rls\_stats\_t Struct Reference

Various configuration options and statistics about an RLS server returned in the following structure by [globus\\_rls\\_client\\_stats\(\)](#)

### Data Fields

- int [ags](#)

#### 5.6.1 Detailed Description

Various configuration options and statistics about an RLS server returned in the following structure by [globus\\_rls\\_client\\_stats\(\)](#)

See [RLS\\_LRCSERVER](#) for possible `ags` values.

#### 5.6.2 Field Documentation

##### 5.6.2.1 int [globus\\_rls\\_stats\\_t::ags](#)

See [RLS\\_LRCSERVER](#)

## 5.7 globus\_rls\_string2\_bulk\_t Struct Reference

#### 5.7.1 Detailed Description

String pair result with return code, returned by bulk query operations.

## 5.8 globus\_rls\_string2\_t Struct Reference

### Data Fields

- char [s1](#)
- char [s2](#)

### 5.8.1 Detailed Description

String pair result. Many of the query functions use this to return pairs of strings (eg LFN,PFN or LFN,LRC).

### 5.8.2 Field Documentation

#### 5.8.2.1 char [globus\\_rls\\_string2\\_t::s1](#)

First string in pair (eg LFN).

#### 5.8.2.2 char [globus\\_rls\\_string2\\_t::s2](#)

Second string in pair (eg PFN or LRC).

## Index

Activation, [13](#)

attr

    globus\_rls\_attribute\_object [33](#)

Connection Management, [14](#)

d

    globus\_rls\_attribute [84](#)

ags

    globus\_rls\_handle [85](#)

    globus\_rls\_rli\_info\_t [35](#)

    globus\_rls\_stats [86](#)

FRLI\_BLOOMFILTER

    globus\_rls\_client\_lrc\_operation [17](#)

globus\_list\_len

    globus\_rls\_client\_miscellaneous [1](#)

globus\_rls\_admin\_cmd\_ping

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_admin\_cmd\_quit

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_admin\_cmd\_ssu

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_admin\_cmd\_t

    globus\_rls\_client\_miscellaneous [8](#),

GLOBUS\_RLS\_ATTR\_EXIST

    globus\_rls\_client\_status [5](#),

GLOBUS\_RLS\_ATTR\_INUSE

    globus\_rls\_client\_status [5](#),

GLOBUS\_RLS\_ATTR\_NEXIST

    globus\_rls\_client\_status [5](#),

globus\_rls\_attr\_op\_all

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_op\_btw

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_op\_eq

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_op\_ge

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_op\_gt

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_op\_le

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_op\_like

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_op\_lt

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_op\_ne

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_op\_t

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_type\_date

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_type\_t

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_type\_int

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_type\_str

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_attr\_type\_t

    globus\_rls\_client\_miscellaneous [8](#),

GLOBUS\_RLS\_ATTR\_VALUE\_NEXIST

    globus\_rls\_client\_status [5](#),

globus\_rls\_attribute\_object [33](#)

    attr, [33](#)

    key, [33](#)

    rc, [33](#)

globus\_rls\_attribute [83](#)

    d, [34](#)

    i, [34](#)

    name, [34](#)

    objtype, [34](#)

    s, [34](#)

    t, [34](#)

    type, [34](#)

    val, [34](#)

GLOBUS\_RLS\_BADARG

    globus\_rls\_client\_status [4](#),

GLOBUS\_RLS\_BADMETHOD

    globus\_rls\_client\_status [4](#),

GLOBUS\_RLS\_BADURL

    globus\_rls\_client\_status [8](#),

globus\_rls\_client\_activation

    GLOBUS\_RLS\_CLIENT\_MODULE, [13](#)

    globus\_rls\_client\_module [13](#)

globus\_rls\_client\_admin

    globus\_rls\_client\_miscellaneous [8](#),

globus\_rls\_client\_attr2s

    globus\_rls\_client\_miscellaneous [10](#)

globus\_rls\_client\_certificate

    globus\_rls\_client\_connection [15](#)

globus\_rls\_client\_close

    globus\_rls\_client\_connection [15](#)

globus\_rls\_client\_connect

    globus\_rls\_client\_connection [15](#)

globus\_rls\_client\_connection

    globus\_rls\_client\_certificate [15](#)

    globus\_rls\_client\_close [15](#)

    globus\_rls\_client\_connect [15](#)

    globus\_rls\_client\_get\_timeout [15](#)

globus\_rls\_client\_proxy\_certificate [15](#)  
 globus\_rls\_client\_set\_timeout [15](#)  
 GLOBUS\_RLS\_SERVER\_DEFPORT [14](#)  
 GLOBUS\_RLS\_URL\_SCHEME [14](#)  
 GLOBUS\_RLS\_URL\_SCHEME\_NOAUTH [14](#)  
 MAXERRMSG, [14](#)  
 globus\_rls\_client\_error\_info  
   globus\_rls\_client\_miscellaneous [15](#)  
 globus\_rls\_client\_free\_list  
   globus\_rls\_client\_queryresult [12](#)  
 globus\_rls\_client\_get\_configuration  
   globus\_rls\_client\_miscellaneous [15](#)  
 globus\_rls\_client\_get\_timeout  
   globus\_rls\_client\_connection [15](#)  
 globus\_rls\_client\_lrc\_add  
   globus\_rls\_client\_lrc\_operation [15](#)  
 globus\_rls\_client\_lrc\_add\_bulk  
   globus\_rls\_client\_lrc\_operation [15](#)  
 globus\_rls\_client\_lrc\_attr\_add  
   globus\_rls\_client\_lrc\_operation [18](#)  
 globus\_rls\_client\_lrc\_attr\_add\_bulk  
   globus\_rls\_client\_lrc\_operation [18](#)  
 globus\_rls\_client\_lrc\_attr\_create  
   globus\_rls\_client\_lrc\_operation [18](#)  
 globus\_rls\_client\_lrc\_attr\_delete  
   globus\_rls\_client\_lrc\_operation [18](#)  
 globus\_rls\_client\_lrc\_attr\_get  
   globus\_rls\_client\_lrc\_operation [19](#)  
 globus\_rls\_client\_lrc\_attr\_modify  
   globus\_rls\_client\_lrc\_operation [19](#)  
 globus\_rls\_client\_lrc\_attr\_remove  
   globus\_rls\_client\_lrc\_operation [19](#)  
 globus\_rls\_client\_lrc\_attr\_remove\_bulk  
   globus\_rls\_client\_lrc\_operation [19](#)  
 globus\_rls\_client\_lrc\_attr\_search  
   globus\_rls\_client\_lrc\_operation [20](#)  
 globus\_rls\_client\_lrc\_attr\_value\_get  
   globus\_rls\_client\_lrc\_operation [20](#)  
 globus\_rls\_client\_lrc\_attr\_value\_get\_bulk  
   globus\_rls\_client\_lrc\_operation [21](#)  
 globus\_rls\_client\_lrc\_clear  
   globus\_rls\_client\_lrc\_operation [21](#)  
 globus\_rls\_client\_lrc\_create  
   globus\_rls\_client\_lrc\_operation [22](#)  
 globus\_rls\_client\_lrc\_create\_bulk  
   globus\_rls\_client\_lrc\_operation [22](#)  
 globus\_rls\_client\_lrc\_delete  
   globus\_rls\_client\_lrc\_operation [22](#)  
 globus\_rls\_client\_lrc\_delete\_bulk  
   globus\_rls\_client\_lrc\_operation [22](#)  
 globus\_rls\_client\_lrc\_exists  
   globus\_rls\_client\_lrc\_operation [23](#)  
 globus\_rls\_client\_lrc\_exists\_bulk  
   globus\_rls\_client\_lrc\_operation [23](#)  
 globus\_rls\_client\_lrc\_get\_lfn  
   globus\_rls\_client\_lrc\_operation [23](#)  
 globus\_rls\_client\_lrc\_get\_lfn\_bulk  
   globus\_rls\_client\_lrc\_operation [24](#)  
 globus\_rls\_client\_lrc\_get\_lfn\_wc  
   globus\_rls\_client\_lrc\_operation [24](#)  
 globus\_rls\_client\_lrc\_get\_pfn  
   globus\_rls\_client\_lrc\_operation [24](#)  
 globus\_rls\_client\_lrc\_get\_pfn\_bulk  
   globus\_rls\_client\_lrc\_operation [25](#)  
 globus\_rls\_client\_lrc\_get\_pfn\_wc  
   globus\_rls\_client\_lrc\_operation [25](#)  
 globus\_rls\_client\_lrc\_mapping\_exists  
   globus\_rls\_client\_lrc\_operation [25](#)  
 globus\_rls\_client\_lrc\_operation  
   FRLI\_BLOOMFILTER, [17](#)  
   globus\_rls\_client\_lrc\_add [21](#)  
   globus\_rls\_client\_lrc\_add\_bulk [21](#)  
   globus\_rls\_client\_lrc\_attr\_add [18](#)  
   globus\_rls\_client\_lrc\_attr\_add\_bulk [18](#)  
   globus\_rls\_client\_lrc\_attr\_create [18](#)  
   globus\_rls\_client\_lrc\_attr\_delete [18](#)  
   globus\_rls\_client\_lrc\_attr\_get [19](#)  
   globus\_rls\_client\_lrc\_attr\_modify [19](#)  
   globus\_rls\_client\_lrc\_attr\_remove [19](#)  
   globus\_rls\_client\_lrc\_attr\_remove\_bulk [19](#)  
   globus\_rls\_client\_lrc\_attr\_search [20](#)  
   globus\_rls\_client\_lrc\_attr\_value\_get [20](#)  
   globus\_rls\_client\_lrc\_attr\_value\_get\_bulk [21](#)  
   globus\_rls\_client\_lrc\_clear [21](#)  
   globus\_rls\_client\_lrc\_create [22](#)  
   globus\_rls\_client\_lrc\_create\_bulk [22](#)  
   globus\_rls\_client\_lrc\_delete [22](#)  
   globus\_rls\_client\_lrc\_delete\_bulk [22](#)  
   globus\_rls\_client\_lrc\_exists [23](#)  
   globus\_rls\_client\_lrc\_exists\_bulk [23](#)  
   globus\_rls\_client\_lrc\_get\_lfn [23](#)  
   globus\_rls\_client\_lrc\_get\_lfn\_bulk [24](#)  
   globus\_rls\_client\_lrc\_get\_lfn\_wc [24](#)  
   globus\_rls\_client\_lrc\_get\_pfn [24](#)  
   globus\_rls\_client\_lrc\_get\_pfn\_bulk [25](#)  
   globus\_rls\_client\_lrc\_get\_pfn\_wc [25](#)  
   globus\_rls\_client\_lrc\_mapping\_exists [25](#)  
   globus\_rls\_client\_lrc\_rename\_lfn [26](#)  
   globus\_rls\_client\_lrc\_rename\_lfn\_bulk [26](#)  
   globus\_rls\_client\_lrc\_rename\_pfn [26](#)  
   globus\_rls\_client\_lrc\_rename\_pfn\_bulk [26](#)  
   globus\_rls\_client\_lrc\_rli\_add [27](#)  
   globus\_rls\_client\_lrc\_rli\_delete [27](#)  
   globus\_rls\_client\_lrc\_rli\_get\_path [27](#)  
   globus\_rls\_client\_lrc\_rli\_info [28](#)  
   globus\_rls\_client\_lrc\_rli\_list [28](#)



- MAXURL, [17](#)
- globus\_uls\_client\_lrc\_renamelfn
  - globus\_uls\_client\_lrc\_operatio[a6](#)
- globus\_uls\_client\_lrc\_renamelfn\_bulk
  - globus\_uls\_client\_lrc\_operatio[a6](#)
- globus\_uls\_client\_lrc\_renamepfm
  - globus\_uls\_client\_lrc\_operatio[a6](#)
- globus\_uls\_client\_lrc\_renamepfm\_bulk
  - globus\_uls\_client\_lrc\_operatio[a6](#)
- globus\_uls\_client\_lrc\_rli\_add
  - globus\_uls\_client\_lrc\_operatio[a7](#)
- globus\_uls\_client\_lrc\_rli\_delete
  - globus\_uls\_client\_lrc\_operatio[a7](#)
- globus\_uls\_client\_lrc\_rli\_get\_part
  - globus\_uls\_client\_lrc\_operatio[a7](#)
- globus\_uls\_client\_lrc\_rli\_info
  - globus\_uls\_client\_lrc\_operatio[a8](#)
- globus\_uls\_client\_lrc\_rli\_list
  - globus\_uls\_client\_lrc\_operatio[a8](#)
- globus\_uls\_client\_miscellaneous
  - globus\_uls\_admin\_cmd\_ping,[9](#)
  - globus\_uls\_admin\_cmd\_qu[9](#),
  - globus\_uls\_admin\_cmd\_ss[9](#),
  - globus\_uls\_attr\_op\_al[8](#)
  - globus\_uls\_attr\_op\_btvd[9](#)
  - globus\_uls\_attr\_op\_e[6](#),
  - globus\_uls\_attr\_op\_g[6](#),
  - globus\_uls\_attr\_op\_g[8](#)
  - globus\_uls\_attr\_op\_l[6](#),
  - globus\_uls\_attr\_op\_like[9](#)
  - globus\_uls\_attr\_op\_l[9](#)
  - globus\_uls\_attr\_op\_n[6](#),
  - globus\_uls\_attr\_type\_data,[6](#)
  - globus\_uls\_attr\_type\_t[8](#)
  - globus\_uls\_attr\_type\_in[8](#)
  - globus\_uls\_attr\_type\_st[8](#),
  - globus\_uls\_obj\_lrc\_lfn[8](#)
  - globus\_uls\_obj\_lrc\_pfr[8](#)
  - globus\_uls\_obj\_rli\_lfn[8](#)
  - globus\_uls\_obj\_rli\_lrc[8](#)
  - uls\_pattern\_sq[8](#)
  - uls\_pattern\_unix[8](#)
- globus\_uls\_client\_miscellaneous
  - globus\_list\_len,[11](#)
  - globus\_uls\_admin\_cmd\_[9](#),
  - globus\_uls\_attr\_op\_[8](#)
  - globus\_uls\_attr\_type\_[8](#)
  - globus\_uls\_client\_admin[9](#)
  - globus\_uls\_client\_attr2s,[0](#)
  - globus\_uls\_client\_error\_info,[1](#)
  - globus\_uls\_client\_get\_con guratio[6](#),
  - globus\_uls\_client\_s2attf,[0](#)
  - globus\_uls\_client\_set\_con guratio[6](#),
- globus\_uls\_client\_stat[40](#)
- globus\_uls\_errmsg,[1](#)
- globus\_uls\_obj\_type\_[8](#)
- globus\_uls\_pattern\_[8](#)
- RLS\_INITIALIZED, [8](#)
- RLS\_LRCSEVER[7](#)
- RLS\_RCVBLOOMFILTER,[7](#)
- RLS\_RCVLFNLIST,[7](#)
- RLS\_RLISEVER,[7](#)
- RLS\_SNDBLOOMFILTER,[7](#)
- RLS\_SNDLFNLIST,[7](#)
- GLOBUS\_ULS\_CLIENT\_MODULE
  - globus\_uls\_client\_activation,[13](#)
- globus\_uls\_client\_module
  - globus\_uls\_client\_activation,[13](#)
- globus\_uls\_client\_proxy\_certificate
  - globus\_uls\_client\_connection,[15](#)
- globus\_uls\_client\_queryresult
  - globus\_uls\_client\_free\_list,[12](#)
- globus\_uls\_client\_rli\_exists
  - globus\_uls\_client\_rli\_operatio[a9](#)
- globus\_uls\_client\_rli\_exists\_bulk
  - globus\_uls\_client\_rli\_operatio[a9](#)
- globus\_uls\_client\_rli\_get\_lrc
  - globus\_uls\_client\_rli\_operatio[a9](#)
- globus\_uls\_client\_rli\_get\_lrc\_bulk
  - globus\_uls\_client\_rli\_operatio[a0](#)
- globus\_uls\_client\_rli\_get\_lrc\_wc
  - globus\_uls\_client\_rli\_operatio[a0](#)
- globus\_uls\_client\_rli\_lrc\_list
  - globus\_uls\_client\_rli\_operatio[a1](#)
- globus\_uls\_client\_rli\_mapping\_exists
  - globus\_uls\_client\_rli\_operatio[a1](#)
- globus\_uls\_client\_rli\_operation
  - globus\_uls\_client\_rli\_exists[a9](#)
  - globus\_uls\_client\_rli\_exists\_bulk[a9](#)
  - globus\_uls\_client\_rli\_get\_lrc[a9](#)
  - globus\_uls\_client\_rli\_get\_lrc\_bulk[a0](#)
  - globus\_uls\_client\_rli\_get\_lrc\_wc[a0](#)
  - globus\_uls\_client\_rli\_lrc\_list[a1](#)
  - globus\_uls\_client\_rli\_mapping\_exists[a1](#)
  - globus\_uls\_client\_rli\_rli\_add[a1](#)
  - globus\_uls\_client\_rli\_rli\_delete[a2](#)
  - globus\_uls\_client\_rli\_rli\_get\_part[a2](#)
  - globus\_uls\_client\_rli\_rli\_list[a2](#)
  - globus\_uls\_client\_rli\_sender\_list[a0](#)
- globus\_uls\_client\_rli\_rli\_add
  - globus\_uls\_client\_rli\_operatio[a1](#)
- globus\_uls\_client\_rli\_rli\_delete
  - globus\_uls\_client\_rli\_operatio[a2](#)
- globus\_uls\_client\_rli\_rli\_get\_part
  - globus\_uls\_client\_rli\_operatio[a2](#)
- globus\_uls\_client\_rli\_rli\_list

globus\_rls\_client\_rli\_operation [62](#)  
 globus\_rls\_client\_rli\_sender\_list  
   globus\_rls\_client\_rli\_operation [60](#)  
 globus\_rls\_client\_s2attr  
   globus\_rls\_client\_miscellaneous [10](#)  
 globus\_rls\_client\_set\_configuration  
   globus\_rls\_client\_miscellaneous [8](#),  
 globus\_rls\_client\_set\_timeout  
   globus\_rls\_client\_connection [15](#)  
 globus\_rls\_client\_stats  
   globus\_rls\_client\_miscellaneous [10](#)  
 globus\_rls\_client\_status  
   GLOBUS\_RLS\_ATTR\_EXIST [5](#)  
   GLOBUS\_RLS\_ATTR\_INUSE [5](#)  
   GLOBUS\_RLS\_ATTR\_NEXIST [5](#)  
   GLOBUS\_RLS\_ATTR\_VALUE\_NEXIST [5](#)  
   GLOBUS\_RLS\_BADARG [4](#)  
   GLOBUS\_RLS\_BADMETHOD [4](#)  
   GLOBUS\_RLS\_BADURL [3](#)  
   GLOBUS\_RLS\_DBERROR [4](#)  
   GLOBUS\_RLS\_GLOBUSERR [3](#)  
   GLOBUS\_RLS\_INV\_ATTR\_OP [5](#)  
   GLOBUS\_RLS\_INV\_ATTR\_TYPE [5](#)  
   GLOBUS\_RLS\_INV\_OBJ\_TYPE [5](#)  
   GLOBUS\_RLS\_INVHANDLE [3](#)  
   GLOBUS\_RLS\_INVSERVER [4](#)  
   GLOBUS\_RLS\_LFN\_EXIST [4](#)  
   GLOBUS\_RLS\_LFN\_NEXIST [4](#)  
   GLOBUS\_RLS\_LRC\_EXIST [4](#)  
   GLOBUS\_RLS\_LRC\_NEXIST [4](#)  
   GLOBUS\_RLS\_MAPPING\_EXIST [5](#)  
   GLOBUS\_RLS\_MAPPING\_NEXIST [4](#)  
   GLOBUS\_RLS\_NOMEMORY [3](#)  
   GLOBUS\_RLS\_OVERFLOW [3](#)  
   GLOBUS\_RLS\_PERM [4](#)  
   GLOBUS\_RLS\_PFN\_EXIST [4](#)  
   GLOBUS\_RLS\_PFN\_NEXIST [4](#)  
   GLOBUS\_RLS\_RLI\_EXIST [4](#)  
   GLOBUS\_RLS\_RLI\_NEXIST [5](#)  
   GLOBUS\_RLS\_SUCCESS [3](#)  
   GLOBUS\_RLS\_TIMEOUT [5](#)  
   GLOBUS\_RLS\_TOO\_MANY\_ -  
     CONNECTIONS [5](#)  
   GLOBUS\_RLS\_UNSUPPORTED [5](#)  
 GLOBUS\_RLS\_DBERROR  
   globus\_rls\_client\_status [4](#),  
 globus\_rls\_errmsg  
   globus\_rls\_client\_miscellaneous [10](#)  
 GLOBUS\_RLS\_GLOBUSERR  
   globus\_rls\_client\_status [8](#),  
 globus\_rls\_handle [84](#)  
   ags, [35](#)  
   handle, [35](#)  
   url, [35](#)  
 GLOBUS\_RLS\_INV\_ATTR\_OP  
   globus\_rls\_client\_status [5](#),  
 GLOBUS\_RLS\_INV\_ATTR\_TYPE  
   globus\_rls\_client\_status [5](#),  
 GLOBUS\_RLS\_INV\_OBJ\_TYPE  
   globus\_rls\_client\_status [5](#),  
 GLOBUS\_RLS\_INVHANDLE  
   globus\_rls\_client\_status [8](#),  
 GLOBUS\_RLS\_INVSERVER  
   globus\_rls\_client\_status [4](#),  
 GLOBUS\_RLS\_LFN\_EXIST  
   globus\_rls\_client\_status [4](#),  
 GLOBUS\_RLS\_LFN\_NEXIST  
   globus\_rls\_client\_status [4](#),  
 GLOBUS\_RLS\_LRC\_EXIST  
   globus\_rls\_client\_status [4](#),  
 GLOBUS\_RLS\_LRC\_NEXIST  
   globus\_rls\_client\_status [4](#),  
 GLOBUS\_RLS\_MAPPING\_EXIST  
   globus\_rls\_client\_status [5](#),  
 GLOBUS\_RLS\_MAPPING\_NEXIST  
   globus\_rls\_client\_status [4](#),  
 GLOBUS\_RLS\_NOMEMORY  
   globus\_rls\_client\_status [8](#),  
 globus\_rls\_obj\_lrc\_lfn  
   globus\_rls\_client\_miscellaneous [8](#),  
 globus\_rls\_obj\_lrc\_pfn  
   globus\_rls\_client\_miscellaneous [8](#),  
 globus\_rls\_obj\_rli\_lfn  
   globus\_rls\_client\_miscellaneous [8](#),  
 globus\_rls\_obj\_rli\_lrc  
   globus\_rls\_client\_miscellaneous [8](#),  
 globus\_rls\_obj\_type\_t  
   globus\_rls\_client\_miscellaneous [8](#),  
 GLOBUS\_RLS\_OVERFLOW  
   globus\_rls\_client\_status [8](#),  
 globus\_rls\_pattern\_t  
   globus\_rls\_client\_miscellaneous [8](#),  
 GLOBUS\_RLS\_PERM  
   globus\_rls\_client\_status [4](#),  
 GLOBUS\_RLS\_PFN\_EXIST  
   globus\_rls\_client\_status [4](#),  
 GLOBUS\_RLS\_PFN\_NEXIST  
   globus\_rls\_client\_status [4](#),  
 GLOBUS\_RLS\_RLI\_EXIST  
   globus\_rls\_client\_status [4](#),  
 globus\_rls\_rli\_info\_t [35](#)  
   ags, [35](#)  
   lastupdate [35](#)  
   updateinterval [35](#)  
   url, [35](#)  
 GLOBUS\_RLS\_RLI\_NEXIST

- globus\_rls\_client\_status [5](#),
- globus\_rls\_sender\_info [36](#)
  - lastupdate [36](#)
  - url, [36](#)
- GLOBUS\_RLS\_SERVER\_DEFPORT
  - globus\_rls\_client\_connection [14](#)
- globus\_rls\_stats [86](#)
  - ags, [36](#)
- globus\_rls\_string2\_bulk [86](#)
- globus\_rls\_string2 [87](#)
  - s1, [37](#)
  - s2, [37](#)
- GLOBUS\_RLS\_SUCCESS
  - globus\_rls\_client\_status [8](#),
- GLOBUS\_RLS\_TIMEOUT
  - globus\_rls\_client\_status [5](#),
- GLOBUS\_RLS\_TOO\_MANY\_CONNECTIONS
  - globus\_rls\_client\_status [5](#),
- GLOBUS\_RLS\_UNSUPPORTED
  - globus\_rls\_client\_status [5](#),
- GLOBUS\_RLS\_URL\_SCHEME
  - globus\_rls\_client\_connection [14](#)
- GLOBUS\_RLS\_URL\_SCHEME\_NOAUTH
  - globus\_rls\_client\_connection [14](#)
- handle
  - globus\_rls\_handle [85](#)
- i
  - globus\_rls\_attribute [84](#)
- key
  - globus\_rls\_attribute\_object [33](#)
- lastupdate
  - globus\_rls\_rli\_info\_t [35](#)
  - globus\_rls\_sender\_info [36](#)
- LRC Operations [16](#)
- MAXERRMSG
  - globus\_rls\_client\_connection [14](#)
- MAXURL
  - globus\_rls\_client\_lrc\_operation [17](#)
- Miscellaneous [6](#)
- name
  - globus\_rls\_attribute [84](#)
- objtype
  - globus\_rls\_attribute [84](#)
- Query Results [11](#)
- rc
  - globus\_rls\_attribute\_object [33](#)
- RLI Operations [28](#)
- RLS\_INITIALIZED
  - globus\_rls\_client\_miscellaneous [8](#),
- RLS\_LRC\_SERVER
  - globus\_rls\_client\_miscellaneous [8](#),
- rls\_pattern\_sql
  - globus\_rls\_client\_miscellaneous [8](#),
- rls\_pattern\_unix
  - globus\_rls\_client\_miscellaneous [8](#),
- RLS\_RCVBLOOMFILTER
  - globus\_rls\_client\_miscellaneous [8](#),
- RLS\_RCVLFNLIST
  - globus\_rls\_client\_miscellaneous [8](#),
- RLS\_RLISERVER
  - globus\_rls\_client\_miscellaneous [8](#),
- RLS\_SNDBLOOMFILTER
  - globus\_rls\_client\_miscellaneous [8](#),
- RLS\_SNDLFNLIST
  - globus\_rls\_client\_miscellaneous [8](#),
- s
  - globus\_rls\_attribute [84](#)
- s1
  - globus\_rls\_string2 [87](#)
- s2
  - globus\_rls\_string2 [87](#)
- Status Codes [2](#)
- t
  - globus\_rls\_attribute [84](#)
- type
  - globus\_rls\_attribute [84](#)
- updateinterval
  - globus\_rls\_rli\_info\_t [35](#)
- url
  - globus\_rls\_handle [85](#)
  - globus\_rls\_rli\_info\_t [35](#)
  - globus\_rls\_sender\_info [36](#)
- val
  - globus\_rls\_attribute [84](#)