

globus rls client Reference Manual

5.1

Generated by Doxygen 1.3.5

Fri Jun 26 23:57:15 2009

Contents

1 globus rls client Main Page	1
2 globus rls client Module Index	1
3 globus rls client Data Structure Index	2
4 globus rls client Module Documentation	2
5 globus rls client Data Structure Documentation	33

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	2
Miscellaneous	6
Query Results	11
Activation	12

Connection Management	13
LRC Operations	15
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:

globus_rls_attribute_object_t (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
globus_rls_attribute_t (Object (LFN or PFN) attribute type)	33
globus_rls_handle_t (RLS Client Handle)	34
globus_rls_rli_info_t (Information about RLI server, returned by globus_rls_client_lrc_rli_info() and globus_rls_client_lrc_rli_list())	35
globus_rls_sender_info_t (Information about server sending updates to an rli, returned by globus_rls_client_rli_sender_list())	36
globus_rls_stats_t (Various configuration options and statistics about an RLS server returned in the following structures by globus_rls_client_stats())	36
globus_rls_string2_bulk_t	36
globus_rls_string2_t	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`
- `#define GLOBUS_RLS_GLOBUSER1`
- `#define GLOBUS_RLS_INVHANDLE2`
- `#define GLOBUS_RLS_BADURL3`
- `#define GLOBUS_RLS_NOMEMORY4`
- `#define GLOBUS_RLS_OVERFLOW5`
- `#define GLOBUS_RLS_BADARG6`
- `#define GLOBUS_RLS_PERM7`
- `#define GLOBUS_RLS_BADMETHOD8`
- `#define GLOBUS_RLS_INVSERVER9`

- #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_CONNECTION 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_qls_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_qls_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 #de ne GLOBUS_RLS_RLI_EXIST 18

RLI already exists in LRC database.

4.1.2.20 #de ne GLOBUS_RLS_RLI_NEXIST 19

RLI doesn't exist in LRC.

4.1.2.21 #de ne GLOBUS_RLS_MAPPING_EXIST 20

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

4.1.2.22 #de ne GLOBUS_RLS_INV_ATTR_TYPE 21

Invalid attribute type, see globus_rls_attr_type_t.

4.1.2.23 #de ne GLOBUS_RLS_ATTR_EXIST 22

Attribute already exists.

4.1.2.24 #de ne GLOBUS_RLS_ATTR_NEXIST 23

Attribute doesn't exist.

4.1.2.25 #de ne GLOBUS_RLS_INV_OBJ_TYPE 24

Invalid object type, see globus_rls_obj_type_t.

4.1.2.26 #de ne GLOBUS_RLS_INV_ATTR_OP 25

Invalid attribute search operator, see globus_rls_attr_op_t.

4.1.2.27 #de ne GLOBUS_RLS_UNSUPPORTED 26

Operation is unsupported.

4.1.2.28 #de ne GLOBUS_RLS_TIMEOUT 27

IO timeout.

4.1.2.29 #de ne GLOBUS_RLS_TOO_MANY_CONNECTIONS 28

Too many connections.

4.1.2.30 #de ne GLOBUS_RLS_ATTR_VALUE_NEXIST 29

Attribute with speci ed value not found.

4.1.2.31 #de ne 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_RL_SERVER](#) 0x2
- #define [RLS_RCV_LFN_LIST](#) 0x4
- #define [RLS_RCV_BLOOM_FILTER](#) 0x8
- #define [RLS_SND_LFN_LIST](#) 0x10
- #define [RLS_SND_BLOOM_FILTER](#) 0x20

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_attr(globus_uls_attr_t attr, char buf, int buflen)
- globus_result_t globus_uls_client_s2attr(globus_uls_attr_type_t type, char sval, globus_uls_attr_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.

4.2.3 Enumeration Type Documentation

4.2.3.1 enumglobus_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 enumglobus_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 enumglobus_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 enumglobus_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_rls_attr_op_ge Values greater than or equal to op1.

globus_rls_attr_op_lt Values less than operand 1.

globus_rls_attr_op_le Values less than or equal to op1.

globus_rls_attr_op_btw Values between operand1 and 2.

globus_rls_attr_op_like Strings "like" operand1 (SQL like).

4.2.3.5 enum `globus_rls_admin_cmd_t`

`globus_rls_client_admin_cmd_t` Commands.

Enumeration values:

- `globus_rls_admin_cmd_ping` Verify RLS server responding.
- `globus_rls_admin_cmd_quit` Tell RLS server to exit.
- `globus_rls_admin_cmd_ssu` Tell LRC server to do softstate update.

4.2.4 Function Documentation

4.2.4.1 `globus_result_t globus_rls_client_admin_cmd(globus_rls_handle_t h, globus_rls_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_rls_client_get_configuration(globus_rls_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_rls_string2_t`. `conf_list` should be freed with `globus_rls_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 DNSs), a colon, and space separated list of permissions the matching users are granted.

4.2.4.3 `globus_result_t globus_rls_client_set_configuration(globus_rls_handle_t h, char option, char value)`

Set server configuration option.

Client needs admin privilege.

Parameters:

- `h` Handle connected to RLS 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 RLS server.

Requires stats privilege.

Parameters:

h Handle connected to RLS 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 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_s2attr(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_qls_attr_type_data](#) sval should be in the form YYYY-MM-DD HH:MM:SS.

attr Attribute whose type and value fields are to be set.

Return values:

GLOBUS_SUCCESS attr->type and attr->value successfully set.

4.2.4.7 `globus_result_t globus_qls_client_error_info (globus_result_t int 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. It 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.
- `rc` 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 len` Size of `buf`.

Return values:

- `globus_result_t` If `preserve` is set a new `globus_result_t` identical to `rc` 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_qls_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 len` Length of `buf`. Message will be truncated to fit 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 (e.g. `globus_qls_string2_t`, `globus_qls_attribute_t` etc).

Data Structures

- `struct globus_qls_attribute_object_t`
`globus_qls_client_attr_search_t` returns a list of these structures which include the object name (LFN or PFN) and attribute value found by the query.
- `struct globus_qls_string2_t`
- `struct globus_qls_string2_bulk_t`

Functions

- `globus_result_t globus_rls_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_rls_string2_t`, `globus_rls_attribute_t`, etc).

A list result should be freed with `globus_rls_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_rls_client_lrc_get_lfn()`.

```
globus_list_t *str2_list;
globus_list_t *p;
globus_rls_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_rls_client_lrc_get_lfn(h, "somepfn", NULL, 0, &str2_list);
for (p = str2_list; p; p = globus_list_rest(p)) {
    str2 = (globus_rls_string2_t *) globus_list_first(p);
    printf("lfn: %s pfn:%s\n", str2->s1, str2->s2);
}
globus_rls_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_rls_client_lrc_get_lfn(h, "somepfn", &offset, 5, &str2_list) == GLOBUS_SUCCESS) {
    for (p = str2_list; p; p = globus_list_rest(p)) {
        str2 = (globus_rls_string2_t *) globus_list_first(p);
        printf("lfn: %s pfn:%s\n", str2->s1, str2->s2);
    }
    globus_rls_client_free_list(str2_list);
    if (offset == -1)
        break;
}
```

4.3.2 Function Documentation

4.3.2.1 `globus_result_t globus_rls_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.

De nes

- #de ne GLOBUS_RLS_URL_SCHEME "rls"
- #de ne GLOBUS_RLS_URL_SCHEME_NOAUTH "rlsn"
- #de ne GLOBUS_RLS_SERVER_DEFPORT 39281
- #de ne MAXERRMSG 1024

Functions

- void globus_rls_client_certificate(char cert le, char key le)
- void globus_rls_client_proxy_certificate(char proxy)
- globus_result globus_rls_client_connect(char url, globus_rls_handle_t h)
- globus_result 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 De ne Documentation

4.5.2.1 #de ne GLOBUS_RLS_URL_SCHEME "rls"

URL scheme to use when connecting to RLS server.

4.5.2.2 #de ne GLOBUS_RLS_URL_SCHEME_NOAUTH "rlsn"

URL scheme when connecting to RLS server without authentication.

4.5.2.3 #de ne GLOBUS_RLS_SERVER_DEFPORT 39281

Default port number that RLS server listens on.

4.5.2.4 #de ne 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 cert le, char key le)

Set certificate used in authentication.

Sets environment variables X509_USER_CERT, X509_USER_KEY, and clears X509_USER_PROXY.

Parameters:

- cert le Name of X509 certificate le.
- key le Name of X509 key le.

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 Handle 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](#) h)
- globus_result_t [globus_rls_client_lrc_create\(\)](#) ([globus_rls_handle_t](#) h, char lfn, char pfn)
- globus_result_t [globus_rls_client_lrc_create_bulk\(\)](#) ([globus_rls_handle_t](#) h, [globus_list_t](#) str2_list, [globus_list_t](#) str2bulk_list)
- globus_result_t [globus_rls_client_lrc_delete\(\)](#) ([globus_rls_handle_t](#) h, char lfn, char pfn)
- globus_result_t [globus_rls_client_lrc_delete_bulk\(\)](#) ([globus_rls_handle_t](#) h, [globus_list_t](#) str2_list, [globus_list_t](#) str2bulk_list)
- globus_result_t [globus_rls_client_lrc_exists\(\)](#) ([globus_rls_handle_t](#) h, char key, [globus_rls_obj_type_t](#) objtype)
- globus_result_t [globus_rls_client_lrc_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_lrc_get_lfn\(\)](#) ([globus_rls_handle_t](#) h, 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 h, globus_list_t pfnlist, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_get_lfn_wc(globus_rls_handle_t h, 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 h, 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 h, globus_list_t lfnlist, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_get_pfn_wc(globus_rls_handle_t h, 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 h, char lfn, char pfn)`
- `globus_result_t globus_rls_client_lrc_rename(globus_rls_handle_t h, char oldname, char newname)`
- `globus_result_t globus_rls_client_lrc_rename_lfn_bulk(globus_rls_handle_t h, globus_list_t str2_list, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_rename_pfn(globus_rls_handle_t h, char oldname, char newname)`
- `globus_result_t globus_rls_client_lrc_rename_pfn_bulk(globus_rls_handle_t h, globus_list_t str2_list, globus_list_t str2bulk_list)`
- `globus_result_t globus_rls_client_lrc_rli_add(globus_rls_handle_t h, char rli_url, int ags, char pattern)`
- `globus_result_t globus_rls_client_lrc_rli_delete(globus_rls_handle_t h, char rli_url, char pattern)`
- `globus_result_t globus_rls_client_lrc_rli_get_part(globus_rls_handle_t h, char rli_url, char pattern, globus_list_t str2_list)`
- `globus_result_t globus_rls_client_lrc_rli_info(globus_rls_handle_t h, char rli_url, globus_rls_rli_info_t info)`
- `globus_result_t globus_rls_client_lrc_rli_list(globus_rls_handle_t h, 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 FRLI_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_rls_client_lrc_attr_add(globus_rls_handle_t h, char key, globus_rls_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_uls_client_lrc_attr_add_bulk(globus_uls_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_uls_attribute_object_t`

str2bulk_list List of failed updates. Each list datum is `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.

4.6.3.3 `globus_result_t globus_uls_client_lrc_attr_create(globus_uls_handle_t h, char name, globus_uls_obj_type_t objtype, globus_uls_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_uls_client_lrc_attr_delete(globus_uls_handle_t h, char name, globus_uls_obj_type_t objtype, globus_bool_t clearvalues)`

Undelete attribute in LRC database, previously created with `globus_uls_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 first 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 type 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 `rc` 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` `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 of `globus_uls_attribute_t` structures.

Return values:

GLOBUS_SUCCESS Attributes successfully retrieved. `attr_list` should be freed with `globus_uls_client_free_list()` when it is no longer needed.

4.6.3.11 `globus_result_t globus_uls_client_lrc_attr_value_get_bulk(globus_uls_handle_t h, globus_list_t keylist, char name, globus_uls_obj_type_t objtype, globus_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 `globus_uls_attribute_object` structures.

Return values:

GLOBUS_SUCCESS Attributes successfully retrieved. attr_obj_list should be freed with `globus_uls_client_free_list()` when it is no longer needed.

4.6.3.12 `globus_result_t globus_uls_client_lrc_add(globus_uls_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 `globus_result_t globus_uls_client_lrc_add_bulk(globus_uls_handle_t h, globus_list_t str2_list, globus_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 `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.

4.6.3.14 `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 [globus_uls_string2_bulk_t](#) 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 **pf**. 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 **pf** 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 pflist, globus_list_t str2_bulk_list)`

Bulk return LFNs mapped to PFN in the LRC database.

Parameters:

h Handle connected to an RLS server.

pflist PFNs to search for.

str2_bulk_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 pf_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.

pf_pattern PFN pattern to search for.

type Identifies wildcard characters used in **pf_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 **pf_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 **pf_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_wcd](#)([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_rename_lfn(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_rename_lfn_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_rename_pfn(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_SUCCESS`RLI (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_SUCCESS`RLI 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_lrc(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 c 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_uls client Data Structure Documentation

5.1 globus_uls_attribute_object_t Struct Reference

[globus_uls_client_uls_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_uls_attribute_attr](#)
- char [key](#)
- int [rc](#)

5.1.1 Detailed Description

[globus_uls_client_uls_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_uls_attribute_t](#)[globus_uls_attribute_object_t::attr](#)

Attribute value.

5.1.2.2 char [globus_uls_attribute_object_t::key](#)

LFN or PFN.

5.1.2.3 int [globus_uls_attribute_object_t::rc](#)

Result code, only used in bulk query.

5.2 globus_uls_attribute_t Struct Reference

Object (LFN or PFN) attribute type.

Data Fields

- char [name](#)
- [globus_uls_obj_type_t](#) [objtype](#)
- [globus_uls_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_tglobus_rls_handle_t::url

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

5.3.2.2 globus_io_handle_tglobus_rls_handle_t::handle

Globus IO handle.

5.3.2.3 intglobus_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 charglobus_rls_rli_info_t::url [256]

URL of server.

5.4.2.2 intglobus_rls_rli_info_t::updateinterval

Interval between softstate updates.

5.4.2.3 intglobus_rls_rli_info_t::ags

RLI ags (see [FRLI_BLOOMFILTER](#)).

5.4.2.4 time_tglobus_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, [12](#)

attr

 globus_rls_attribute_object [33](#)

Connection Management [13](#)

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 [14](#)

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 [14](#)

 globus_rls_client_close [15](#)

 globus_rls_client_connect [15](#)

 globus_rls_client_get_timeout [15](#)

globus_rls_client_proxy_certificate [14](#)
 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 [11](#)
 globus_rls_client_lrc_add_bulk
 globus_rls_client_lrc_operation [11](#)
 globus_rls_client_lrc_attr_add
 globus_rls_client_lrc_operation [17](#)
 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 [18](#)
 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 [20](#)
 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_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 [17](#)
 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 [18](#)
 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 [20](#)
 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_rls_client_lrc_renamelfn
 - globus_rls_client_lrc_operatio[a6](#)
- globus_rls_client_lrc_renamelfn_bulk
 - globus_rls_client_lrc_operatio[a6](#)
- globus_rls_client_lrc_renamepfm
 - globus_rls_client_lrc_operatio[a6](#)
- globus_rls_client_lrc_renamepfm_bulk
 - globus_rls_client_lrc_operatio[a6](#)
- globus_rls_client_lrc_rli_add
 - globus_rls_client_lrc_operatio[a7](#)
- globus_rls_client_lrc_rli_delete
 - globus_rls_client_lrc_operatio[a7](#)
- globus_rls_client_lrc_rli_get_part
 - globus_rls_client_lrc_operatio[a7](#)
- globus_rls_client_lrc_rli_info
 - globus_rls_client_lrc_operatio[a8](#)
- globus_rls_client_lrc_rli_list
 - globus_rls_client_lrc_operatio[a8](#)
- globus_rls_client_miscellaneous
 - globus_rls_admin_cmd_ping,[9](#)
 - globus_rls_admin_cmd_qu[9](#),
 - globus_rls_admin_cmd_ss[9](#),
 - globus_rls_attr_op_al[8](#)
 - globus_rls_attr_op_bt[v8](#)
 - globus_rls_attr_op_e[8](#)
 - globus_rls_attr_op_g[8](#)
 - globus_rls_attr_op_g[8](#)
 - globus_rls_attr_op_l[8](#)
 - globus_rls_attr_op_like[8](#)
 - globus_rls_attr_op_l[8](#)
 - globus_rls_attr_op_n[8](#)
 - globus_rls_attr_type_data,[8](#)
 - globus_rls_attr_type_t[8](#)
 - globus_rls_attr_type_in[8](#)
 - globus_rls_attr_type_st[8](#)
 - globus_rls_obj_lrc_lfn[8](#)
 - globus_rls_obj_lrc_pfr[8](#)
 - globus_rls_obj_rli_lfn[8](#)
 - globus_rls_obj_rli_lrc[8](#)
 - rls_pattern_sq[8](#)
 - rls_pattern_unix[8](#)
- globus_rls_client_miscellaneous
 - globus_list_len[11](#)
 - globus_rls_admin_cmd_8,
 - globus_rls_attr_op_8
 - globus_rls_attr_type_8
 - globus_rls_client_admin[9](#)
 - globus_rls_client_attr2[10](#)
 - globus_rls_client_error_inf[10](#)
 - globus_rls_client_get_con_guratio[8](#),
 - globus_rls_client_s2att[10](#)
 - globus_rls_client_set_con_guratio[8](#),
- globus_rls_client_stat[10](#)
- globus_rls_errmsg[11](#)
- globus_rls_obj_type_8
- globus_rls_pattern_8
- RLS_LRC_SERVER[7](#)
- RLS_RCVBLOOMFILTER,[7](#)
- RLS_RCVLFNLIST,[7](#)
- RLS_RLISERVER,[7](#)
- RLS_SNDBLOOMFILTER,[7](#)
- RLS_SNDLFNLIST,[7](#)
- GLOBUS_RLS_CLIENT_MODULE
 - globus_rls_client_activation[13](#)
- globus_rls_client_module
 - globus_rls_client_activation[13](#)
- globus_rls_client_proxy_certificate
 - globus_rls_client_connection[14](#)
- globus_rls_client_queryresult
 - globus_rls_client_free_list[12](#)
- globus_rls_client_rli_exists
 - globus_rls_client_rli_operation[29](#)
- globus_rls_client_rli_exists_bulk
 - globus_rls_client_rli_operation[29](#)
- globus_rls_client_rli_get_lrc
 - globus_rls_client_rli_operation[29](#)
- globus_rls_client_rli_get_lrc_bulk
 - globus_rls_client_rli_operation[30](#)
- globus_rls_client_rli_get_lrc_wc
 - globus_rls_client_rli_operation[30](#)
- globus_rls_client_rli_lrc_list
 - globus_rls_client_rli_operation[31](#)
- globus_rls_client_rli_mapping_exists
 - globus_rls_client_rli_operation[31](#)
- globus_rls_client_rli_operation
 - globus_rls_client_rli_exists[29](#)
 - globus_rls_client_rli_exists_bulk[29](#)
 - globus_rls_client_rli_get_lrc[29](#)
 - globus_rls_client_rli_get_lrc_bulk[30](#)
 - globus_rls_client_rli_get_lrc_wc[30](#)
 - globus_rls_client_rli_lrc_list[31](#)
 - globus_rls_client_rli_mapping_exists[31](#)
 - globus_rls_client_rli_rli_add[31](#)
 - globus_rls_client_rli_rli_delete[32](#)
 - globus_rls_client_rli_rli_get_part[32](#)
 - globus_rls_client_rli_rli_list[32](#)
 - globus_rls_client_rli_sender_list[30](#)
- globus_rls_client_rli_rli_add
 - globus_rls_client_rli_operation[31](#)
- globus_rls_client_rli_rli_delete
 - globus_rls_client_rli_operation[32](#)
- globus_rls_client_rli_rli_get_part
 - globus_rls_client_rli_operation[32](#)
- globus_rls_client_rli_rli_list
 - globus_rls_client_rli_operation[32](#)

globus_rls_client_rli_sender_list
 globus_rls_client_rli_operation [3,0](#)
 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 [11](#)
 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 [5](#)
- 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 [15](#)
- 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_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](#)