

GT 4.2.1 RLS : User's Guide

GT 4.2.1 RLS : User's Guide

Introduction

The Replica Location Service (RLS) maintains and provides access to mapping information from *logical names* for data items to *target names*.

RLS was co-developed by the Globus team and Work Package 2 of the DataGrid project. The distributed RLS is intended to replace the centralized Globus replica catalog available in earlier releases of GT2.x. The distributed RLS provides higher performance, reliability and scalability.

Replication of data items can reduce access latency, improve data locality, and increase robustness, scalability and performance for distributed applications. An RLS typically does not operate in isolation but functions as one component of a data grid architecture (other components include services that provide reliable file transfers, metadata management, reliable replication and workflow management).

The RLS implementation is based on the following mechanisms:

- *Consistent local state maintained in Local Replica Catalogs (LRCs)*. Local catalogs maintain mappings between arbitrary *logical file names (LFNs)* and the *physical file names (PFNs)* associated with those LFNs on its storage system(s).
 - *Collective state with relaxed consistency maintained in Replica Location Indices (RLIs)*. Each *RLI* contains a set of mappings from LFNs to LRCs. A variety of index structures can be defined with different performance characteristics simply by varying the number of RLIs and the amount of redundancy and partitioning among the RLIs.
 - *Soft state maintenance of RLI state*. LRCs send information about their state to RLIs using soft state protocols. State information in RLIs times out and must be periodically refreshed by soft state updates.
 - *Compression of state updates*. This optional compression uses *Bloom filters* to summarize the content of a Local Replica Catalog before sending a soft state update to a Replica Location Index Node.
 - *Membership and partitioning information maintenance*. The current RLS implementation maintains static information about the LRCs and RLIs participating in the distributed system. As new implementations of the RLS are developed, they will use OGSA mechanisms for registration of services and for service lifetime management.
-

Table of Contents

1. Mapping Replica Locations	1
1. Generate a valid proxy	1
2. Ping the server	1
3. Creating replica location mappings	1
4. Adding replica location mappings	1
5. Querying replica location mappings	2
6. Deleting replica location mappings	2
7. Using bulk operations	2
8. Using interactive mode	3
2. Command line tools	4
globus-rls-admin	5
globus-rls-cli	7
globus-rls-server	11
3. Troubleshooting	16
1. Verbose error messages	16
2. Errors	17
4. Usage statistics collection by the Globus Alliance	18
1. RLS-specific usage statistics	18
Glossary	20
Index	21

List of Tables

2.1. Options for globus-rls-admin	6
2.2. Options for globus-rls-cli	7
2.3. Commands for globus-rls-cli	9
2.4. Options for globus-rls-server	14
3.1. Replica Locator Service (RLS) Errors	17

Chapter 1. Mapping Replica Locations

This section describes a few key usage scenarios and provides examples of using the RLS command-line tools.

1. Generate a valid proxy

Before using any of the tools, a user must generate a valid user proxy. Use [grid-proxy-init](#).

```
% $GLOBUS_LOCATION/bin/grid-proxy-init
Your identity: /O=Grid/OU=GlobusTest/OU=simpleCA.mymachine/OU=mymachine/CN=John Smith
Enter GRID pass phrase for this identity:
```

2. Ping the server

To check whether your server is active you may use the [globus-rls-admin\(1\) ping](#) command.

```
% $GLOBUS_LOCATION/bin/globus-rls-admin -p rls://localhost
ping rls://localhost: 0 seconds
```

3. Creating replica location mappings

When the RLS server is first installed its database of replica location information will be empty, as expected. To create a replica location mapping, use the [globus-rls-cli\(1\) create](#) command. Replica information in RLS is represented as mappings from logical names to target names. Typically, the logical name will be a unique identifier for a given replicated data set and the target name will be a URL identifying a particular replica of the data set.

```
% $GLOBUS_LOCATION/bin/globus-rls-cli create my-logical-name-1 url-for-target-name-1 rls://lo
```



Note

The create command is intended for creating the initial replica mapping entry for a given logical name. If the user attempts to create another entry using an existing logical name, RLS will report a user error. To map additional target names to an existing logical name, see [Section 4, “Adding replica location mappings”](#).

4. Adding replica location mappings

To map additional target names to a logical name created by the previously described create command, use the [globus-rls-cli\(1\) add](#) command.

```
% $GLOBUS_LOCATION/bin/globus-rls-cli add my-logical-name-1 url-for-target-name-2 rls://lo
```

5. Querying replica location mappings

Once your RLS server is populated with replica location mappings, you can query the server for useful information using the `globus-rls-cli(1) query` command.

```
% $GLOBUS_LOCATION/bin/globus-rls-cli query lrc lfn my-logical-name-1 rls://localhost
my-logical-name-1: url-for-target-name-1
my-logical-name-1: url-for-target-name-2
```

6. Deleting replica location mappings

To remove unwanted replica location mappings from your RLS server, use the `globus-rls-cli(1) delete` command. The delete operation works *directly* on the mapping and *indirectly* on the logical and target names. When the delete operation is performed by the RLS server the association between the specified logical name and the specified target name is eliminated. However, there may still be other target names associated with the logical name, and there could still be other logical names associated with the target name, though the latter scenario is less likely. Only when all mapping associations for a given logical name (or a given target name) are eliminated (i.e., the specified logical name has no target names associated with it) will the logical (or target) name be deleted from the RLS server.

```
% $GLOBUS_LOCATION/bin/globus-rls-cli delete my-logical-name-1 url-for-target-name-1 rls://
% $GLOBUS_LOCATION/bin/globus-rls-cli query lrc lfn my-logical-name-1 rls://localhost
my-logical-name-1: url-for-target-name-2
% $GLOBUS_LOCATION/bin/globus-rls-cli delete my-logical-name-1 url-for-target-name-2 rls://
% $GLOBUS_LOCATION/bin/globus-rls-cli query lrc lfn my-logical-name-1 rls://localhost
globus_rls_client: LFN doesn't exist: my-logical-name-1
```

7. Using bulk operations

The `globus-rls-cli(1)` supports a variety of bulk operations that enhance productivity for users and reduce network connection overhead from making multiple, separate invocations of the client. The general pattern for bulk operation support as implemented by the client is a parameter list consisting of `bulk command-name [command-modifiers] param-1 param-2 param-N`, such as `bulk query lrc lfn my-logical-name-1 my-logical-name-2 my-logical-name-3`.

```
% $GLOBUS_LOCATION/bin/globus-rls-cli bulk create my-logical-name-1 url-for-target-name-1-1
% $GLOBUS_LOCATION/bin/globus-rls-cli bulk add my-logical-name-1 url-for-target-name-1-2 m
% $GLOBUS_LOCATION/bin/globus-rls-cli bulk query lrc lfn my-logical-name-1 my-logical-name
my-logical-name-3: LFN doesn't exist
my-logical-name-2: url-for-target-name-2-1
my-logical-name-2: url-for-target-name-2-2
my-logical-name-1: url-for-target-name-1-1
my-logical-name-1: url-for-target-name-1-2
```

8. Using interactive mode

The `globus-rls-cli(1)` supports an interactive mode in addition to the general command-line mode. To enter the interactive mode, simply invoke the client without any command.

```
% $GLOBUS_LOCATION/bin/globus-rls-cli rls://localhost
rls> query lrc lfn my-logical-name-2
my-logical-name-2: url-for-target-name-2-1
my-logical-name-2: url-for-target-name-2-2
rls> query lrc lfn my-logical-name-1
my-logical-name-1: url-for-target-name-1-1
my-logical-name-1: url-for-target-name-1-2
rls> bulk delete my-logical-name-1 url-for-target-name-1-1 my-logical-name-1
url-for-target-name-1-2 my-logical-name-2 url-for-target-name-2-1
my-logical-name-2 url-for-target-name-2-2
rls> bulk query lrc lfn my-logical-name-2 my-logical-name-1
my-logical-name-1: LFN doesn't exist
my-logical-name-2: LFN doesn't exist
rls> exit
```

Chapter 2. Command line tools

Name

globus-rls-admin -- RLS administration tool

globus-rls-admin

Tool description

Performs administrative operations on an RLS server.

Synopsis

-A/-a/-C option value/-c option/-d/-e/-p/-q/-s/-t timeout/-u/-v [rli] [pattern] [server]

Options

Table 2.1. Options for globus-rls-admin

-A	<p>Adds rli to the list of <i>RLI</i> servers updated by an <i>LRC</i> server using <i>Bloom filters</i>.</p> <p><i>Note:</i> Partitions are not supported with Bloom filters. The LRC server maintains one Bloom filter for all <i>LFNs</i> in its database, which is sent to all RLI servers configured to receive Bloom filter updates with this option.</p>
-a	<p>Adds rli and optionally pattern to the list of RLI servers that the LRC server sends updates to (using a list of LFNs).</p> <p>If pattern is specified, then only LFNs matching it will be sent to rli.</p> <p>If rli is added with no patterns, then it is sent all updates. Pattern matching is done using standard Unix file globbing.</p>
-C option value	<p>Sets server option to value.</p> <p><i>Important:</i> This does <i>not</i> update the configuration file. The next time the server is restarted, the configuration change will be <i>lost</i>.</p>
-c option	<p>Retrieves the configuration value for the specified option from the server.</p> <p>If option is set to all, then all options are retrieved.</p>
-d	<p>Removes rli and pattern from the list of RLI servers that the LRC server sends updates to.</p> <p>If pattern is not specified, then all entries for rli are removed.</p> <p><i>Note:</i> If all patterns are removed separately, then rli is sent all updates. To stop any updates from being sent to rli, do <i>not</i> specify pattern.</p>
-e	<p>Clears the LRC database. Removes all lfn, <i>pfn</i> mappings.</p>
-p	<p>Verifies that the server is responding.</p>
-q	<p>Causes the RLS server to exit.</p>
-S	<p>Shows statistics and other information gathered by the RLS server.</p> <p>This is intended to be input into GRIS.</p>
-s	<p>Shows the list of RLI servers and patterns being sent updates by the LRC server.</p> <p>If rli or pattern are not specified, they are considered wildcards.</p>
-t timeout	<p>Sets timeout (in seconds) for RLS server requests.</p> <p>The default value is 30.</p>
-u	<p>Causes the LRC server to immediately start full soft state updates to any RLI servers previously added with the -a option.</p>
-v	<p>Shows the version and exits.</p>

Name

globus-rls-cli -- RLS client tool

globus-rls-cli

Tool description

Provides a command line interface to some of the functions supported by RLS. It also supports an interactive interface (if *command* is not specified). In interactive mode, double quotes may be used to encode an argument that contains white space.

Synopsis

command [*-c*] [*-h*] [*-l reslimit*] [*-s*] [*-t timeout*] [*-u*] [*command*] *rls-server*

Options

The client command tool uses **getopt** for command line parsing.

Note: Some versions will continue scanning for options (works that begin with a hyphen) for the entire command line, which makes it impossible to specify negative integer or floating point value for an *attribute*. The workaround for this problem is to tell **getopt()** that there are no more options by including 2 hyphens. For example, to specify the value **-2** you must enter **-- -2**.

Table 2.2. Options for globus-rls-cli

-c	Sets "clearvalues" flag when deleting an attribute (will remove any attribute value records when an attribute is deleted).
-h	Shows usage.
-l reslimit	Sets an incremental limit on the number of results returned by a wildcard query at a time. Note that <i>all results will be returned</i> by the client. This parameter only limits the number of results <i>incrementally retrieved</i> by the client during a single internal communication call. For instance, if the wildcard query produces 1000 results and the reslimit is set to 100, the client will internally make 10 calls to the server. From the user's perspective the client will simply return all 1000 results. Zero means no limit.
-s	Uses SQL style wildcards (% and _).
-t timeout	Sets timeout (in seconds) for RLS server requests. The default is 30 seconds.
-u	Uses Unix style wildcards (* and ?).
-v	Shows version.

Commands

Table 2.3. Commands for globus-rls-cli

add <lfn> <pfn>	Adds <i>pfn</i> to mappings of <i>lfn</i> in an <i>LRC</i> catalog.
attribute add <object> <attr> <obj-type> <attr-type>	Adds an attribute to an object, where <i>object</i> should be the lfn or pfn name. <i>obj-type</i> should be one of lfn or pfn. <i>attr-type</i> should be one of date, float, int, or string. If <value> is of type date then it should be in the form "YYYY-MM-DD HH:MM:DD".
attribute bulk add <object> <attr> <obj-type>	Bulk adds attribute values.
attribute bulk delete <object> <attr> <obj-type>	Bulk deletes attributes.
attribute bulk query <attr> <obj-type> <object>	Bulk queries attributes.
attribute define <attr> <obj-type> <attr-type>	Defines a new attribute.
attribute delete <object> <attr> <obj-type>	Removes <i>attribute</i> from <i>object</i> .
attribute modify <object> <attr> <obj-type> <attr-type>	Modifies the value of an attribute.
attribute query <object> <attr> <obj-type>	Retrieves the value of the specified attribute for object .
attribute search <attr> <obj-type> <operator> <attr-type>	Searches for objects which have the specified attribute matching <i>operator</i> and <i>value</i> . <i>operator</i> should be one of =, !=, >, >=, <, or <=.
attribute show <attr> <obj-type>	Shows an attribute definition. If <i>attr</i> is a hyphen (-) then all attributes are shown.
attribute undefine <attr> <obj-type>	Deletes an attribute definition. Will return an error if any objects possess this attribute.
bulk add <lfn> <pfn> [<lfn> <pfn>]	Bulk adds lfn, pfn mappings.
bulk create <lfn> <pfn> [<lfn> <pfn>]	Bulk creates lfn, pfn mappings.
bulk delete <lfn> <pfn> [<lfn> <pfn>]	Bulk deletes lfn, pfn mappings.
bulk query lrc lfn [<lfn> ...]	Bulk queries the LRC for lfns.
bulk query lrc pfn [<pfn> ...]	Bulk queries the LRC for pfns.
bulk query rli lfn [<lfn> ...]	Bulk queries the <i>RLI</i> for lfns.
create <lfn> <pfn>	Creates a new <i>lfn</i> , <i>pfn</i> mapping in an LRC catalog.
delete <lfn> <pfn>	Deletes a <i>lfn</i> , <i>pfn</i> mapping from an LRC catalog.
exit	Exits the interactive session.
help	Prints a help message.
query lrc lfn <lfn>	Queries an LRC server for mappings of <i>lfn</i> .
query lrc pfn <pfn>	Queries an LRC server for mappings to <i>pfn</i> .
query rli lfn <lfn>	Queries an RLI server for mappings of <i>lfn</i> .

query wildcard lrc lfn <lfn-pattern>	Performs a wildcarded query of an LRC server for mappings of <i>lfn-pattern</i> . Patterns use the standard Unix wildcard characters: an asterisk (*) matches 0 or more characters, and a question mark (?) matches any single character.
query wildcard lrc pfn <pfn-pattern>	Queries an LRC server for mappings to <i>pfn-pattern</i> . Patterns use the standard Unix wildcard characters: an asterisk (*) matches 0 or more characters, and a question mark (?) matches any single character.
query wildcard rli lfn <lfn-pattern>	Queries an RLI server for mappings of <i>lfn-pattern</i> . Patterns use the standard Unix wildcard characters: an asterisk (*) matches 0 or more characters, and a question mark (?) matches any single character.
set reslimit <limit>	<p>Sets an incremental limit on the number of results returned by a wildcard query at a time.</p> <p>Note that <i>all results will be returned</i> by the client. This parameter only limits the number of results <i>incrementally retrieved</i> by the client during a single internal communication call. For instance, if the wildcard query produces 1000 results and the reslimit is set to 100, the client will internally make 10 calls to the server. From the user's perspective the client will simply return all 1000 results.</p>
set timeout <timeout>	<p>Sets the timeout (in seconds) on calls to the RLS server.</p> <p>The default value is 30.</p>
version	Shows the version and exits.

Name

globus-rls-server -- RLS server tool

globus-rls-server

Tool description

The RLS server (**globus-rls-server**) can be configured as either one or both of the following:

- *Location Replica Catalog (LRC)* server, which manages *Logical FileName (LFN)* to *Physical FileName (PFN)* mappings in a database. *Note*: If **globus-rls-server** is configured as an LRC server, the *RLI* servers that it sends updates to should be added to the database using **globus-rls-admin**.
- *Replica Location Index (RLI)* server, which manages mappings of LFNs to LRC servers.

Clients wishing to locate one or more physical filenames associated with a logical filename should first contact an RLI server, which will return a list of LRCs that may know about the LFN. The LRC servers are then contacted in turn to find the physical filenames.

Note: RLI information may be out of date, so clients should be prepared to get a negative response when contacting an LRC (or no response at all if the LRC server is unavailable).

Synopsis

```
[ -B lrc_update_bf ] [ -b maxbackoff ] [ -C rlsconf ] [ -c conffile ] [ -d ] [ -e rli_expire_int ] [ -F lrc_update_factor ] [ -f maxfreethreads ] [ -I true/false ] [ -i idletimeout ] [ -K rlskeyfile ] [ -L loglevel ] [ -l true/false ] [ -M maxconnections ] [ -m maxthreads ] [ -N ] [ -o lrc_buffer_time ] [ -p pidfiledir ] [ -r true/false ] [ -S rli_expire_stale ] [ -s starthreads ] [ -t timeout ] [ -U myurl ] [ -u lrc_update_ll ] [ -v ]
```

LRC to RLI Updates

Two methods exist for LRC servers to inform RLI servers of their LFNs.

- By default, the LFNs are sent from the LRC to the RLI. This can be time consuming if the number of LFNs is large, but it does give the RLI an exact list of the LFNs known to the LRC, and it allows wildcard searching of the RLI.
- Alternatively, *Bloom filters* may be sent, which are highly compressed summaries of the LFNs. However, they do not allow wildcard searching and will generate more "false positives" when querying an RLI.

Please see below for more on Bloom filters.

globus-rls-admin can be used to manage the list of RLIs that an LRC server updates. This includes partitioning LFNs among multiple RLI servers.

A soft state algorithm is used in both update modes: periodically the LRC server sends its state (LFN information) to the RLI servers it updates. The RLI servers add these LFNs to their indexes or update timestamps if the LFNs were already known. RLI servers expire information about LFN, LRC mappings if they haven't been updated for a period longer than the soft state update interval.

The following options in the configuration file control the soft state algorithm when an LRC updates an RLI by sending LFNs:

- **rli_expire_int** (seconds)
- **rli_expire_stale** (seconds)
- **lrc_update_ll** (seconds)
- **lrc_update_bf** (seconds)

Updates to an LRC (new LFNs or deleted LFNs) normally don't propagate to RLI servers until the next soft state update (controlled by options **lrc_update_ll** and **lrc_update_bf**).

However, by enabling "immediate update" mode (set **lrc_update_immediate** to **true**), an LRC will send updates to an RLI within **lrc_buffer_time** seconds.

If updates are done with LFN lists then only the LFNs that have been added or deleted to the LRC are sent. If Bloom filters are used, then the entire Bloom filter is sent.

When immediate updates are enabled, the interval between soft state updates is multiplied by **lrc_update_factor** as long as no updates have failed (LRC and RLI are considered to be in sync). This can greatly reduce the number of soft state updates an LRC needs to send to an RLI.

Incremental updates are buffered by the LRC server until either 200 updates have accumulated (when LFN lists are used), or **lrc_buffer_time** seconds have passed since the last update.

Bloom filter updates

A Bloom filter is an array of bits. Each LFN is hashed multiple times and the corresponding bits in the Bloom filter are set.

Querying an RLI to verify if an LFN exists is done by performing the same hashes and checking if the bits in the filter are on. If not, then the LFN is known not to exist. If they're all on, then all that's known is that the LFN probably exists.

The size of the Bloom filter (as a multiple of the number of LFNs) and the number of hash functions control the false positive rate. The default values of 10 and 3 give a false positive rate of approximately 1%.

The advantage of Bloom filters is their efficiency. For example, if the LRC has 1,000,000 LFNs in its database, with an average length of 20 bytes, then 20,000,000 bytes must be sent to an RLI during a soft state update (assuming no partitioning). The RLI server must perform 1,000,000 updates to its database to create new LFN, LRC mappings or update timestamps on existing entries. With Bloom filters only 1,250,000 bytes are sent (10 x 1,000,000 bits / 8), and there are no database operations on the RLI (Bloom filters are maintained entirely in memory). A comparison of the time to perform a 1,000,000 LFN update: it took 20 minutes sending all the LFNs and less than 1 second using a Bloom filter. However as noted before, Bloom filters do *not* support wild card searches of an RLI.

Note: An LRC server can update some RLIs with Bloom filters and others with LFNs. However, an RLI server can only be updated using one method.

The following options in the [Configuration](#) file control Bloom filter updates:

- **rli_bloomfilter** true/false
- **rli_bloomfilter_dir** none/default/pathname
- **lrc_bloomfilter_numhash** N
- **lrc_bloomfilter_ratio** N

- `irc_update_bf` seconds

Log Messages

globus-rls-server uses syslog to log errors and other information (facility **LOG_DAEMON**) when it's running in normal (daemon) mode.

If the **-d** option (debug) is specified, then log messages are written to stdout.

Signals

The server will reread its configuration file if it receives a **HUP** signal. It will wait for all current requests to complete and shut down cleanly if sent any of the following signals: **INT**, **QUIT** or **TERM**.

Options (globus-rls-server)

The following table describes the command line options available for `globus-rls-server`:

Table 2.4. Options for globus-rls-server

-b maxbackoff	Maximum time (in seconds) that globus-rls-server will attempt to reopen the socket it listens on after an I/O error.
-C rls-certfile	Name of the X.509 certificate file that identifies the server; sets environment variable X509_USER_CERT .
-c conffile	Name of the configuration file for the server. The default is \$GLOBUS_LOCATION/etc/globus-rls-server.conf if the environment variable GLOBUS_LOCATION is set; else, /usr/local/etc/globus-rls-server.conf .
-d	Enables debugging. The server will not detach from the controlling terminal, and log messages will be written to stdout rather than syslog. For additional logging verbosity set the loglevel (see the -L option) to higher values.
-e rli_expire_int	Interval (seconds) at which an RLI server should expire stale entries.
-F lrc_update_factor	If lrc_update_immediate mode is on, and the LRC server is in sync with an RLI server (an LRC and RLI are synced if there have been no failed updates since the last full soft state update), then the interval between RLI updates for this server (lrc_update_ll) is multiplied by lrc_update_factor .
-f maxfreethreads	Maximum number of idle threads the server will leave running. Excess threads are terminated.
-I true false	Turns LRC to RLI immediate update mode on (true) or off (false). The default value is false .
-i idletimeout	Seconds after which idle client connections are timed out.
-K rls-keyfile	Name of the X.509 key file. Sets environment variable X509_USER_KEY .
-L loglevel	Sets the log level. By default this is 0 , which means only errors will be logged. Higher values mean more verbose logging.
-l true false	Configures whether the server is an LRC server. The default is false .
-M maxconnections	Maximum number of active connections. It should be small enough to prevent the server from running out of open file descriptors. The default value is 100 .
-m maxthreads	Maximum number of threads server will start up to support simultaneous requests.
-N	Disables authentication checking. This option is intended for debugging. Clients should use the URL RLSN://host to disable authentication on the client side.
-o lrc_buffer_time	LRC to RLI updates are buffered until either the buffer is full or this much time (in seconds) has elapsed since the last update. The default value is 30 .
-p pidfiledir	Directory where PID files should be written.

-r	Configures whether the server is an RLI server. The default value is false .
-S rli_expire_stale	Interval (in seconds) after which entries in the RLI database are considered stale (presumably because they were deleted in the LRC). Stale entries are not returned in queries.
-s startthreads	Number of threads to start up initially.
-t timeout	Timeout (in seconds) for calls to other RLS servers (in other words, for LRC calls to send an update to an RLI). A value of 0 disables timeouts. The default value is 30 .
-U myurl	URL for this server.
-u lrc_update_ll	Interval (in seconds) between lfn-list LRC to RLI updates.
-v	Shows version and exits.

Chapter 3. Troubleshooting

The following section described a few troubleshooting tips. Additional information on troubleshooting can be found in the [FAQ](#)¹.

For a list of common errors in GT, see [Error Codes](#).

1. Verbose error messages

When troubleshooting problems encountered during usage of the RLS client or server, verbose error messages may be enabled by setting the `GLOBUS_ERROR_VERBOSE` environment variable. Verbose error messages are particularly helpful when communicating on the *rls-user@globus.org* or *gt-user@globus.org* list or when reporting problems on the *bugzilla.globus.org* site.

¹ http://www.globus.org/toolkit/data/rls/rls_faq.html

2. Errors

Table 3.1. Replica Locator Service (RLS) Errors

Error Code	Definition	Possible Solutions
<p>Error with credential: The proxy credential: <credential> with subject: <subject> expired <minutes> minutes ago</p>	<p>Expired proxy credential</p>	<p>Create a new proxy with <code>grid-proxy-init</code>.</p>
<p>Unable to connect to local-host:xxxx</p>	<p>Unable to connect to the local host. This can be due to a variety of reasons, including a wrong address or port number in the RLS connection URL or an issue with a firewall configuration.</p>	<ul style="list-style-type: none"> • Double-check the address and port number in the RLS connection URL. parameters are correct. • If a firewall configuration is preventing connections to the target host for a particular port, you may need to consult the system administrator.
<p>"connection timeout"</p>	<p>At times, a client may experience a connection timeout when interacting with the RLS server due to a variety of reasons:</p> <ul style="list-style-type: none"> • One reason could simply be due to wide-area network latency or congestion. • Another situation that users eventually encounter is due to scaling of the system. As the RLS server's database of replica location mappings grows in size, some query operations, such as bulk queries involving large quantities of mappings or wildcard queries that result in a large subset of mappings, will begin to take more time both to process the query and to return the large results set to the client over the network. 	<p>If timeouts are experienced with increasing frequency, increase the RLS server's timeout configuration parameter found in the <code>\$GLOBUS_LOCATION/var/globus-rls-server.conf</code> file. You may also use the <code>-t</code> timeout option of the <code>globus-rls-cli</code> tool.</p>

Chapter 4. Usage statistics collection by the Globus Alliance

1. RLS-specific usage statistics

The following usage statistics are sent by RLS Server by default in a UDP packet:

- Component identifier
- Usage data format identifier
- Time stamp
- Source IP address
- Source hostname (to differentiate between hosts with identical private IP addresses)
- Version number
- Uptime
- *LRC* service indicator
- *RLI* service indicator
- Number of *LFNs*
- Number of *PFNs*
- Number of Mappings
- Number of RLI LFNs
- Number of RLI LRCs
- Number of RLI Senders
- Number of RLI Mappings
- Number of threads
- Number of connections

The RLS sends the usage statistics at server startup, server shutdown, and once every 24 hours when the service is running.

If you wish to disable this feature, you can set the following environment variable before running the RLS:

```
export GLOBUS_USAGE_OPTOUT=1
```

By default, these usage statistics UDP packets are sent to `usage-stats.globus.org:4180` but can be redirected to another host/port or multiple host/ports with the following environment variable:

```
export GLOBUS_USAGE_TARGETS="myhost.mydomain:12345 myhost2.mydomain:54321"
```

You can also dump the usage stats packets to stderr as they are sent (although most of the content is non-ascii). Use the following environment variable for that:

```
export GLOBUS_USAGE_DEBUG=MESSAGES
```

Also, please see our [policy statement](#)¹ on the collection of usage statistics.

¹ ../../Usage_Stats.html

Glossary

B

Bloom filter Compression scheme used by the Replica Location Service (RLS) that is intended to reduce the size of soft state updates between Local Replica Catalogs (LRCs) and Replica Location Index (RLI) servers. A Bloom filter is a bit map that summarizes the contents of a Local Replica Catalog (LRC). An LRC constructs the bit map by applying a series of hash functions to each logical name registered in the LRC and setting the corresponding bits.

L

Local Replica Catalog (LRC) Stores mappings between logical names for data items and the target names (often the physical locations) of replicas of those items. Clients query the LRC to discover replicas associated with a logical name. Also may associate attributes with logical or target names. Each LRC periodically sends information about its logical name mappings to one or more RLIs.

See also [RLI](#)⁶.

logical file name A unique identifier for the contents of a file.

logical name A unique identifier for the contents of a data item.

P

physical file name The address or the location of a copy of a file on a storage system.

R

Replica Location Index (RLI) Collects information about the logical name mappings stored in one or more Local Replica Catalogs (LRCs) and answers queries about those mappings. Each RLI periodically receives updates from one or more LRCs that summarize their contents.

RLS attribute Descriptive information that may be associated with a logical or target name mapping registered in a Local Replica Catalog (LRC). Clients can query the LRC to discover logical names or target names that have specified RLS attributes.

T

target name The address or location of a copy of a data item on a storage system.

⁶ #rli

Index

A

- administrative operations
 - configuring LRC server to stop updating RLI, 5
 - configuring LRC-to-RLI updates
 - compressed (Bloom filters), 5
 - uncompressed, 5
 - configuring settings (runtime only), 5
 - pinging the server, 1, 5
 - stopping RLS server, 5

C

- client operations
 - attributes
 - adding an attribute to an lfn or pfn, 7
 - bulk adding attribute values, 7
 - bulk deleting attribute values, 7
 - bulk querying attributes, 7
 - defining a new attribute, 7
 - deleting an attribute definition, 7
 - modifying the value of an attribute, 7
 - removing an attribute, 7
 - retrieving the value of the specified attribute for an lfn or pfn, 7
 - searching for lfns or pfns which have the specified matching operator and value, 7
 - showing an attribute definition, 7
 - basic
 - adding physical filenames to mappings of logical filenames in a LocalReplicaCatalog, 1, 7
 - creating a new lfn, pfn mapping in an LRC catalog, 1, 7
 - deleting a lfn, pfn mapping from an LRC catalog, 2, 7
 - exiting the interactive session, 7
 - bulk, 2
 - bulk adding lfn, pfn mappings, 7
 - bulk creating lfn, pfn mappings, 7
 - bulk deleting lfn, pfn mappings, 7
 - bulk querying the LRC for lfns, 7
 - bulk querying the LRC for pfns, 7
 - bulk querying the RLI for lfns, 7
 - querying, 2
 - performing a wildcarded query of an LRC server for mappings of lfn-pattern, 7
 - querying an LRC server for mappings of lfn, 7
 - querying an LRC server for mappings of pfn, 7
 - querying an LRC server for mappings to pfn-pattern, 7
 - querying an RLI server for mappings of lfn, 7

- querying an RLI server for mappings to lfn-pattern, 7

- using interactive mode, 3, 7

E

- errors, 17

G

- generating a valid proxy, 1

S

- server operations, 11
 - configuring RLS server as Location Replica Catalog (LRC), 11
 - configuring RLS server as Replica Location Index (RLI), 11

T

- troubleshooting, 16
 - (for end-users), 16
 - verbose error messages, 16

U

- usage statistics, 18