Data Management from a User Perspective

Jan Just Keijser

Nikhef

Grid Tutorial, 13-14 November 2008
Outline

- Introduction
- Low Level Data Management
- LCG File Catalog (LFC)
- Data Management CLIs
- Command-line Examples
- Data Management APIs
- Programming Examples
Storage and Data Management can be (are) overwhelming at first

- **Storage systems:**
  - CASTOR
  - dCache
  - DPM
  - gridftp
  - LFC
  - SRB
  - SRM

- **Command sets:**
  - gridftp: globus-url-copy
  - LFC: lcg-* commands, lfc-* commands, edg-* commands
  - SRB: S* commands
  - SRM: srm* commands
Why so many systems, protocols and commands?

- Historically grown
- Competing systems: there is no clear “winner” yet
- Different protocols serve different needs:
  - management protocols vs transfer protocols
  - low-level vs. high-level access
  - some have metadata management (srb, srm)
  - some provide Hierarchical Storage Management (HSM), others disk-only
- The transport protocol gsiftp:// is supported by nearly all products
- The management protocol SRM has a Web Services interface and is supported by CASTOR, DPM and dCache
<table>
<thead>
<tr>
<th><strong>Product</strong></th>
<th><strong>Management Protocol</strong></th>
<th><strong>Transport Protocol</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>CASTOR</td>
<td>SRM v1</td>
<td>gsiftp, rfio</td>
</tr>
<tr>
<td>dCache</td>
<td>SRM v1, v2.1</td>
<td>gsiftp, dcap, gsi-dcap, xrootd</td>
</tr>
<tr>
<td>DPM</td>
<td>SRM v1, v2.2</td>
<td>gsiftp, gsi-rfio</td>
</tr>
<tr>
<td>SToRM</td>
<td>SRM v2</td>
<td>gsiftp, ?</td>
</tr>
<tr>
<td>GFAL (client)</td>
<td>SRM v1, v2</td>
<td>gsiftp, {rfio</td>
</tr>
</tbody>
</table>

**Notes**

- SRM v1 and v2 are **not** compatible
- gsi-rfio and rfio cannot be used simultaneously
- gsiftp does not provide a full POSIX interface (seek, trunc), rfio and gsi-rfio do
Low Level Data Management

• gsiftp/GridFTP (all SEs)
  – globus-url-copy file:///home/janjust/file \
gsiftp://srm.grid.sara.nl/pnfs/grid.sara.nl/data/dteam/file
  – Third party transfer
    ▪ globus-url-copy gsiftp://hostA/pathA gsiftp://hostB/pathB
  – Also edg-gridftp-ls, edg-gridftp-rm, edg-gridftp-mkdir etc.
  – Uberftp
    ▪ Interactive gridftp client
    ▪ ftp commands
    ▪ Gsi authentication
Low Level Data Management

- **dCache: gsi-dcap**
  - 20000:25000 is derived from GLOBUS_TCP_PORT_RANGE environment variable

- **Secure rfio**
  - `rfcp /path/myfile \ t2se01.physics.ox.ac.uk:/dpm/physics.ox.ac.uk/home/dteam/file`

- **SRM: srmcp**
  - `Srmcp file:///tmp/file \ srm://srm.grid.sara.nl:8443//pnfs/grid.sara.nl/data/dteam/file`
  - Count the slashes!
• **Provides:**
  – Command line tools with administrative functionality
  – Hierarchical unix-like namespace and namespace operations for LFNs
    - `lfn:/grid/<vo name>/mydir/myfile`
    - `lfc-mkdir, lfc-chmod`
  – Integrated GSI Authentication + Authorization
  – Access Control Lists (Unix Permissions and POSIX ACLs)
  – Checksums
  – Sessions (multiple operations inside a single transaction)
  – Bulk operations (inside transactions)
  – User exposed transaction C/C++ API (+ auto rollback on failure)
    - Python wrapper provided (python module lfc)
LFC Interfaces

• **Integration with GFAL and lcg_utils APIs**
  ➔ lcg-utils/GFAL access the catalog in a transparent way

• **Integration with the WMS**
  – The RB can locate Grid files: allows for data based match-making
  – Jdl file:
    ▪ `InputData = "lfn:/grid/tutor/MyFile";`
These two packages provide (nearly) all the functionality needed by most grid users:

- **LFC_client:**
  - lfc-* commands
  - mostly for manipulating *directories*

- **lcg_utils:**
  - lcg-* commands
  - Transparent interaction with file catalogs and storage interfaces when needed
  - Abstraction from technology of specific implementations
  - mostly for manipulating *files*
## Summary of LFC commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lfc-chmod</td>
<td>Change access mode of the LFC file/directory</td>
</tr>
<tr>
<td>lfc-chown</td>
<td>Change owner and group of the LFC file-directory</td>
</tr>
<tr>
<td>lfc-delcomment</td>
<td>Delete the comment associated with the file/directory</td>
</tr>
<tr>
<td>lfc-getacl</td>
<td>Get file/directory access control lists</td>
</tr>
<tr>
<td>lfc-ln</td>
<td>Make a symbolic link to a file/directory</td>
</tr>
<tr>
<td>lfc-ls</td>
<td>List file/directory entries in a directory</td>
</tr>
<tr>
<td>lfc-mkdir</td>
<td>Create a directory</td>
</tr>
<tr>
<td>lfc-rename</td>
<td>Rename a file/directory</td>
</tr>
<tr>
<td>lfc-rm</td>
<td>Remove a file/directory</td>
</tr>
<tr>
<td>lfc-setacl</td>
<td>Set file/directory access control lists</td>
</tr>
<tr>
<td>lfc-setcomment</td>
<td>Add/replace a comment</td>
</tr>
</tbody>
</table>
### CLI: `lcg-*` Commands

#### lcg_utils commands: Replica Management

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lcg-cp</td>
<td>Copies a grid file to a local destination</td>
</tr>
<tr>
<td>lcg-cr</td>
<td>Copies a file to a SE and registers the file in the catalog</td>
</tr>
<tr>
<td>lcg-del</td>
<td>Delete one file</td>
</tr>
<tr>
<td>lcg-rep</td>
<td>Replication between SEs and registration of the replica</td>
</tr>
<tr>
<td>lcg-gt</td>
<td>Gets the TURL for a given SURL and transfer protocol</td>
</tr>
<tr>
<td>lcg-sd</td>
<td>Sets file status to “Done” for a given SURL in a SRM request</td>
</tr>
</tbody>
</table>

#### lcg_utils commands: File Catalog Interaction

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lcg-aa</td>
<td>Add an alias in LFC for a given GUID</td>
</tr>
<tr>
<td>lcg-ra</td>
<td>Remove an alias in LFC for a given GUID</td>
</tr>
<tr>
<td>lcg-rf</td>
<td>Registers in LFC a file placed in a SE</td>
</tr>
<tr>
<td>lcg-uf</td>
<td>Unregisters in LFC a file placed in a SE</td>
</tr>
<tr>
<td>lcg-la</td>
<td>Lists the alias for a given SURL, GUID or LFN</td>
</tr>
<tr>
<td>lcg-lg</td>
<td>Get the GUID for a given LFN or SURL</td>
</tr>
<tr>
<td>lcg-lr</td>
<td>Lists the replicas for a given GUID, SURL or LFN</td>
</tr>
</tbody>
</table>
Preparation for using lfc-* and lcg-* commands

• Define the server hostname
  – The LFC server must be published in the BDII ($LFC_GFAL_INFOSYS)
  – Use environment variable:
    $LFC_HOST=<lfc_server_hostname>

• Use the 'lcg-infosites' command to find out the name of the current LFC and SE:
  > lcg-infosites --vo tutor lfc
    lfc.grid.sara.nl
  > lcg-infosites --vo tutor se
    gb-se-ams.els.sara.nl  se.grid.rug.nl  srm.grid.sara.nl

• Remember
  – lfc-* commands are mostly for manipulating directories
  – lcg-* commands are mostly for manipulating files
Listing the entries of an LFC directory


- Where *path* specifies the LFN pathname (mandatory)
- Remember that LFC has a directory tree structure
- `/grid/<VO_name>/(<you create it>)`

- All members of a VO have read-write permissions under their directory
- You can set LFC_HOME to use relative paths
  - > lfc-ls /grid/tutor/me
  - > export LFC_HOME=/grid/tutor
  - > lfc-ls -l me
  - > lfc-ls -l -R /grid

- *l* : long listing
- *R* : list the contents of directories recursively: Don’t use it!
Creating directories in the LFC

`lfc-mkdir [-m mode] [-p] path...`

- Where `path` specifies the LFC pathname
- Remember that while registering a new file (using lcg-cr, for example) the corresponding destination directory must be created in the catalog beforehand.
- Example:

  > `lfc-mkdir /grid/tutor/me`

You can check the directory with:

  > `lfc-ls -l /grid/tutor/me`

```
drwxr-xr-x  0 19122    1077             0 Jun 14 11:36 demo
```
lcg-cr: copy and register a file

\texttt{lcg-cr [-d dest\_file | dest\_host] [-l lfn] [--vo vo] src\_file}

Where \texttt{lfn} is the Logical File Name that can include an LFC pathname created with \texttt{lfc-mkdir}

- Example:

\texttt{> lcg-cr --vo tutor -l me/test -d}
\texttt{srm://srm.grid.sara.nl:8443/pnfs/grid.sara.nl/data/tutor/test/some\_file\_name file:`pwd`/test}

guid:7b4efaef-bb0f-42a3-bb6f-bbe35080d105

- List our file by looking at the SFN of the replica's:

\texttt{> lcg-lr --vo tutor lfn:me/test}
\texttt{sfn://srm.grid.sara.nl//pnfs/grid.sara.nl/data/tutor/test/some\_file\_name}

- List the files in our directory in the LFC:

\texttt{> lfc-ls -l me}
\texttt{-rw-rw-r-- 1 30010 2024 114 Sep 18 10:33 test}
**lfc-ln**: creating a symbolic link

- `lfc-ln -s file linkname`
- `lfc-ln -s directory linkname`

Create a link to the specified *file* or *directory* with *linkname*.

- Example:
  ```bash
  > lfc-ln -s /grid/tutor/me/test /grid/tutor/aLink
  ```

- Let’s check the link using `lfc-ls` with long listing (`-l`):
  ```bash
  > lfc-ls -l
  lrwxrwxrwx 1 30010 2024 0 Sep 18 10:38 aLink ->
  /grid/tutor/me/test
  ```
Examples

**lfc-*comment**: adding/deleting metadata information

- **lfc-setcomment path comment**
  Add/replace a *comment* associated with a *path* (i.e. file or directory)

- **lfc-delcomment path**
  Delete a comment previously added

- This is the only metadata (one field) supported by the catalog

- Example:
  > `lfc-setcomment me/test “nice file”`

- Let’s see what happened:
  > `lfc-ls --comment /grid/tutor/me/test`
  
  /grid/tutor/me/test   nice file
**Examples**

**lfc-rm, lcg-del:** deleting the file

*lfc-rm*
Remove a file/link/directory only from the catalog

*lcg-del*
Remove a file from the SE(s) and the Ifns/links from the catalog

**Examples**

- Delete all replicas:
  ```
  > lcg-del -a --vo tutor \ 
  guid:8e413879-7cb3-4260-af9f-6964392da7e8
  ```

- Delete only one replica:
  ```
  > lcg-del -a --vo tutor -s srm.grid.sara.nl \ 
  guid:8e413879-7cb3-4260-af9f-6964392da7e8
  ```
srscp: an SRM Example

- **Note**: the srm* tools do not use the File Catalog. Hence they do not know of logical file names nor can they hide the storage hierarchy (SFN) from the user!

```
  srscp source-url destination-url
```

where the source-url and destination-url can be of the form
- `file://some-path/some-filename`
- `srm://server:8443/pnfs/grid.sara.nl/data/<VO-name>/<filename>`

Count the slashes ('/')! This is because *srscp* is a Java application.

Example

- Copy a file to the SRM server

  ```
  > srscp file:///home/janjust/file \ 
  srm://srm.grid.sara.nl:8443/pnfs/grid.sara.nl/data/tutor/file
  ```
Data Management APIs

APIs for most client commands are available:

- **Grid File Access Library (GFAL): API**
  - Adds POSIX-like file I/O and explicit catalog interaction functionality
  - Useful/unavoidable when accessing files that are larger than the available scratch space on a grid worker node
  - Still provides the abstraction and transparency of lcg_utils
  - C/C++ interface
  - Python wrapper interface

- **LFC_client**: lfc_* API calls
  - Interaction with file catalogs

- **lcg_utils**: lcg_* API calls
  - Transparent interaction with file catalogs and storage interfaces when needed
<table>
<thead>
<tr>
<th>Function</th>
<th>Function</th>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>lfc_access</td>
<td>lfc_deleteclass</td>
<td>lfc_listreplica</td>
<td>lfc_setacl</td>
</tr>
<tr>
<td>lfc_aborttrans</td>
<td>lfc_delreplica</td>
<td>lfc_lstat</td>
<td>lfc_setatime</td>
</tr>
<tr>
<td>lfc_addreplica</td>
<td>lfc_endtrans</td>
<td>lfc_mkmdir</td>
<td>lfc_setcomment</td>
</tr>
<tr>
<td>lfc_apiinit</td>
<td>lfc_enterclass</td>
<td>lfc_modifyclass</td>
<td>lfc_seterrbuf</td>
</tr>
<tr>
<td>lfc_chclass</td>
<td>lfc_errmsg</td>
<td>lfc_opendir</td>
<td>lfc_setfsize</td>
</tr>
<tr>
<td>lfc_chdir</td>
<td>lfc_getacl</td>
<td>lfc_queryclass</td>
<td>lfc_starttrans</td>
</tr>
<tr>
<td>lfc_chmod</td>
<td>lfc_getcomment</td>
<td>lfc_readdir</td>
<td>lfc_stat</td>
</tr>
<tr>
<td>lfc_chown</td>
<td>lfc_getcwd</td>
<td>lfc_readlink</td>
<td>lfc_symlink</td>
</tr>
<tr>
<td>lfc_closedir</td>
<td>lfc_getpath</td>
<td>lfc_rename</td>
<td>lfc_umask</td>
</tr>
<tr>
<td>lfc_creat</td>
<td>lfc_lchown</td>
<td>lfc_rewind</td>
<td>lfc_undelete</td>
</tr>
<tr>
<td>lfc_delcomment</td>
<td>lfc_listclass</td>
<td>lfc_rmdir</td>
<td>lfc_unlink</td>
</tr>
<tr>
<td>lfc_delete</td>
<td>lfc_listlinks</td>
<td>lfc_selectsrvr</td>
<td>lfc_utime</td>
</tr>
<tr>
<td></td>
<td></td>
<td>send2lfc</td>
<td></td>
</tr>
<tr>
<td>lcg_cp</td>
<td>lcg_lr</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lcg_cr</td>
<td>lcg_ra</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lcg_del</td>
<td>lcg_rf</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lcg_rep</td>
<td>lcg_uf</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lcg_sd</td>
<td>lcg_la</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lcg_aa</td>
<td>lcg-lg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lcg_gt</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
#!/usr/bin/python

# Copy a file to Storage element and register in the LFC.

import sys
import lcg_util
src='file:/etc/hosts'
dest='srm.grid.sara.nl'
guid="212fa800-9d65-11da-a746-0800200c9b13"
lfn="/grid/tutor/me/testfile"
vo='tutor'
relativepath='me/testfile'
nstreams=1
config="
insecure=0
verbose=1
actual_guid=""
for i in range(0,37):
    actual_guid=actual_guid + " 
output= lcg_util.lcg_cr(src,dest,guid,lfn,vo,relativepath,nstreams,"\n, insecure, verbose, actual_guid)
print "teststatus: ", output
print "actual_guid: ", actual_guid
Still Interested? Questions?