

Basic ELFI usage

A walkthrough of an ELFI session.

Basic ELFI usage

Author: Riccardo Murri <riccardo.murri@ictp.it>

This document is a walk-through of a sample session with ELFI on the command-line of a gLite UI; it presents all steps: mounting the filesystem, doing some basic data management operations on the Grid, and unmounting the filesystem at the end.

Table of contents

- [Before you start with ELFI](#)
- [Mounting an ELFI filesystem](#)
 - ◆ [Structure of the ELFI filesystem](#)
- [Uploading files to the Grid](#)
- [Accessing Grid file contents](#)
- [Downloading files from the Grid](#)
- [Renaming Grid files](#)
- [Erasing Grid files](#)
- [Unmounting](#)
- [Recommended reading](#)

Before you start with ELFI

ELFI program code needs support from the some function libraries (that are distributed and installed as part of a standard gLite UI) and from the FUSE generic filesystem interface (which is not part of the gLite software bundle). So, be sure to have installed ELFI following all the steps in the [installation instructions](#).

A bit of nomenclature used in this document:

LFN, or "logical file name"

This is the string by which you refer to a file in the LFC namespace: it's a UNIX-like path starting with `/grid/``*yourvo*`. In other words, it's the string you would feed to ```lfc-*` commands; for instance, `/grid/egrid/myfile`.

replica

A "replica" is a copy of a Grid file data, residing on a particular SE. The LFC catalog maps a single LFN to at least one (and possibly many) replica.

Mounting an ELFI filesystem

"Mounting" a filesystem makes the file tree under a certain directory available for system use; with ELFI, this makes the files that are registered in your LFC host available as if they were files on your local machine.

You must mount an ELFI filesystem before you can do anything with ELFI.

To mount an ELFI filesystem:

```
elfi /path/to/elfi/mountpoint
```

The mountpoint must be an existing directory, and you must have write permission to it; otherwise ELFI will refuse to start. It's a good idea to use a subdirectory of your home directory, for instance `~/elfi` (this is what will be used in the rest of this walkthrough).

To give it a try, type these commands into your shell:

```
# go to home directory
cd ~

# ensure mountpoint exists
mkdir ~/elfi

# mount ELFI on ~/elfi
elfi ~/elfi
```

After a while, you will be returned to the shell prompt: ELFI is now operational!

Warning

Depending on the number of sites present in your BDII, ELFI can take up to some minutes to start. Please be patient.

ELFI requires a valid user proxy certificate to run; it will use the current proxy to access the Grid services. If no proxy is available when you mount an ELFI filesystem, `elfi` will print a warning message:

```
$ ./elfi ~/elfi
Warning: Unable to read proxy file '/tmp/x509up_ul000': No such file or directory
```

If you see such a message, you should create a proxy with `grid-proxy-init` or `voms-proxy-init`, as soon as you're returned to the shell prompt, or you will not be able to read any file or directory in ELFI.

Note

Any user may mount an ELFI filesystem, and *only that user can access* the filesystem she has mounted: the FUSE kernel module will deny access to other users (unless the mount option `-o allow_other` is explicitly passed).

Structure of the ELFI filesystem

Under the ELFI mountpoint (in the example, the `~/elfi` directory), you will see:

- ◆ a directory named LFC, which mirrors the tree of logical file names registered in the LFC catalog namespace.
- ◆ one directory for each SE that is accessible by your VO; the tree under an SE-directory will display a subset of the LFC namespace, namely, *all directories* that are in the LFC namespace but *only those files that actually have a replica on that SE*.

You can access a Grid-stored file by appending its *logical file name* (that is, its path in the LFC namespace) to the (local filesystem) path to the `LFC/` or one of the SE-named directories. For instance, to access the LFN `/grid/egrid/mydata.txt` you would use `~/elfi/LFC/grid/egrid/mydata.txt`.

The only difference between the `LFC/` and the SE-named trees is this: when you read or write a file *contents* in the `LFC/` tree, ELFI will choose a destination SE, [1] if you read or write a file contents under an SE-named directory tree, ELFI will read from or write to the file replica residing on that SE, if it already

exists. [2]

So, you will normally use the LFC/ tree, and operate on the SE-named trees only when data storage location is important, or for replica management.

Note

It is important to stress that you *always* refer to files by their *logical file name*; no matter if it's under the LFC/ or the SE-named trees: always use logical file names.

How ELFI actually chooses the destination SE is detailed in the [elfi\(1\)](#) man page. The algorithm roughly is as follows:

- [1]
- ◆ if a preferred SE has been set by the `--preferred-se` command-line option, use that SE;
 - ◆ if the environment variable `VO_<yourvo>_DEFAULT_SE` has non-empty value, use the SE named by that variable;
 - ◆ otherwise, use the first SE returned by a query to the information system.

ELFI will only show a file under an SE-named tree if and only if a replica of that file is already present on that SE; in other words, you can only operate on existing replicas. (With one exception to this rule: when creating a new file, you can do that in any SE-named directory: the first replica of that LFN will be put on that SE.)

[2]

Uploading files to the Grid

When you create a new file with ELFI, it will be registered in the [LFC](#) catalog, and its contents will be written on some SE (chosen according to the rules described in the [ELFI](#) man page).

To copy & register a local file on the Grid:

```
# copy & register
cp /some/local/file ~/elfi/LFC/logical/file/name
```

Accessing Grid file contents

You can directly access Grid file contents, as you normally do with local files. When you try to access a file contents, [ELFI](#) will try to open that file from an SE via the GSI-RFIO protocol; only SEs supporting that protocol will be actually accessible through ELFI.

Note

Unfortunately, the GSI-RFIO is not (yet) standard on the EGEE Grid, and most SEs run the insecure RFIO protocol; instructions on how to upgrade an SE to run the GSI-RFIO protocol are given at <http://www.egrid.it/sw/elfi/wiki/InstallDPMRfioServer>

For instance:

```
# view/edit a file stored on Grid
vi ~/elfi/LFC/logical/file/name
```

You can also choose to access a particular replica of a LFN:

```
# check that two replicas are consistent by printing their checksum
md5sum ~/elfi/SE1/logical/file/name
```

```
md5sum ~/elfi/SE2/logical/file/name
```

Note

ELFI will fetch the contents as required: it will not download the whole file in one go, rather, it will fetch portions of the file as requested by the application.

Downloading files from the Grid

As a particular case, you can download files from the Grid to your local disk storage:

```
# copy to local file
cp ~/elfi/LFC/logical/file/name /some/local/file
```

This is by no means different than any other file access; see [Accessing Grid file contents](#)

Renaming Grid files

You can rename Grid files with the `mv` command as usual:

```
# rename LFN
mv ~/elfi/LFC/logical/file/name ~/elfi/LFC/different/logical/file/name
```

Warning

ELFI can change the logical file name of any Grid file registered in the LFC catalog; however, it will *never* change the SURLs of its replicas. (Neither will `lcf-utl`s; at present, SURLs have to be considered immutable.)

Erasing Grid files

You can remove a *file* from Grid storage with the `rm` command as usual. You can remove directories with the usual `rmdir` command.

Warning

Removing a file by its LFN will *remove all replicas* of that LFN, so beware!

For instance:

```
# remove a file and all of its replicas
rm ~/elfi/LFC/logical/file/name

# remove an empty directory
rmdir ~/elfi/LFC/logical/file/name
```

Unmounting

Unmounting an ELFI filesystem will terminate the use of the ELFI system: the mountpoint will revert to its local, non-Grid contents and you can no longer access Grid-stored files through ELFI.

To unmount an ELFI filesystem:

```
elfi --unmount /path/to/elfi/mountpoint
```

To end the sample session:

```
# terminate ELFI session
elfi --unmount ~/elfi
```

Upon unmounting, the `elfi` program instance that's serving the filesystem at mountpoint will exit. The BDI cache daemon `elfi-bdii-cache` will only exit when it's no longer in use, that is, the last `elfi` instance has exited, and no new one has connected to the cache within 15 seconds.

Warning

When the `elfi` instance crashes, the filesystem remains mounted, but all operations will fail with an error "Transport endpoint is not connected".

You *must* unmount the "orphaned" mountpoint before you can mount a new ELFI filesystem on it again.

You can unmount an ELFI filesystem even if it's currently in use: the kernel will actually unmount the filesystem when all files (still open at the time of unmounting) are closed.[#] Note that *you can no longer operate on a filesystem that has been scheduled for unmounting*, even if it has not actually been unmounted.

[3] This is called "lazy" unmounting.

Recommended reading

UNIX filesystem

<http://www.uwsg.iu.edu/UAU/filesystem/>

http://en.wikipedia.org/wiki/Filesystem#File_systems_under_Unix_and_Unix-like_systems

<http://tldp.org/LDP/sag/html/filesystems.html> (esp. section 5.10.7)

We value your feed back!

Whether you would like to test ELFI in your VO applications, deploy it for production use, hack the sources, or you just like the idea, we would like to hear from you! And, we're willing to offer installation and usage support and implement your ideas in ELFI.

So, please tell us your suggestions, critiques, bug reports and experiences with ELFI! Write to the mailing list elfi@egrid.it (if you are not subscribed, your mail will get through after moderator approval; you can subscribe at <http://www.egrid.it/cgi-bin/mailman/listinfo/elfi>)

Docutils System Messages

System Message: INFO/1 (<string>, line 337)

External hyperlink target "egrid" is not referenced.