Enabling Grids for E-science in Europe

www.eu-egee.org

HANDS ON

Grid Data Management Hands-on

Simone Campana
LCG Experiment Integration and Support
CERN IT

EGEE is a project funded by the European Union under contract IST-2003-508833

Agenda

- Data Management command line tools
- lcg_utils API

DM Command Line Tools

1. Check the syntax of the described commands in the manpages and LCG-2 User Guide

   https://edms.cern.ch/file/454439//LCG-2-UserGuide.html

2. Create a dummy file and copy it to the grid

3. Check the LFN, GUID and SURL of the file

4. Copy the file to two SEs with lcg-rep and globus-url-copy. Check that both files exist but only one is registered

5. Register the file that was not already in the catalog

6. Copy the file back to the UI and check it

7. Delete and unregister all the files and replicas created

SOLUTION: check the *sol-DM-CLI.txt* file in the agenda

lcg_utils API

1. Check the syntax of the lcg_util API in the manpages and in $LCG_LOCATION/include/lcg_util.h

2. Create an application that copies and registers "/etc/services" ("file1") to the Grid. Submit it in a job and use the default SE.

3. Create an application that gets the SURLs and TURLs of "file1", given the LFN

4. Submit a job that retrieves "file1" from the SE and ships it out with the Sandbox

DOWNLOAD and unpack *lcg-utils.tar.gz* file in the agenda

See /SOLUTION folder and following slides

lcg_utils API

General Remarks:

- Look into the “lcg_util_api” folder. Complete the programs of the “skeletons” folder to create yours. In the “solutions” folder you can find some working solutions.

- You may use the provided “Makefile” changing only the PROGRAM variable if necessary. It is important that you:
  - **Use the** g++-3.2.2 **compiler (and linker)**
  - **Link against** -lgfal -llcg_util **from** -L$(LCG_LOCATION)/lib
  - **Link against** -lglobus_<name>_<flavor> **from -**L$(GLOBUS_LOCATION)/lib

- Globus flavors:

gcc32	→	**Globus for gcc 3.2**
gcc32pthr	→	**Support for multi-threaded applications**
gcc32dbg	→	**Debuggable**
gcc32dbgpthr	→	**Multi-threaded debuggable**

lcg_utils API

Remarks:

- You need to include the “lcg_util.h” library as an extern C.
- First create an application that just copies and registers your local “/etc/services”. Compare your solution with “copy_and_register_1.cpp”
- Try again, and realize that if the LFN exists already, you get an error, but the file is copied and registered in the LRC anyway. Improve the previous application in order to deal with this. Compare with “copy_and_register_2.cpp”
- Now, create a JDL file and send your application with a job, so that it gets executed in a WN.
- Modify the application so that it reads the environmental variable “VO_GILDA_DEFAULT_SE” and uses its value as the destination SE. The variable should be defined everywhere, but to be sure, you may use “grid010.ct.infn.it”. Compare with “copy_and_reg_3.cpp”

lcg_utils API

Remarks:

- You should use “lcg_lr” and “lcg_gt”.
- Hint: consider that if the function expects char ***pfns you probably should define char **pfns and pass &pfns as the argument.
- Take into account that you may get more than one SURL, and that you should get one TURL for each of these SURLs.
- As protocol for the TURLs, we will request “gsiftp”, since we are sure that it is defined for every SE.
- Compare your results with *get_names.cpp*

lcg_utils API

Remarks:

- The idea is to write an application which copies the file to the WN.
- Then, the file can be retrieved using WMS
  - Someone said the OutputSandbox can be used only for std.err and std.out ?????????
- In your application, try to copy the file to the current directory.
  - The advantage: we know that in $PWD we have write access
  - We most likely have write access also to /tmp: why we do not copy it over there?
- Compare your result with *only_copy.cpp*