The LCG File Catalog (LFC)

Jean-Philippe Baud – Sophie Lemaitre
IT-GD, CERN

May 2005

Overview

- **LFC Architecture**
- LFC/Fireman Tests
- LFC Deployment

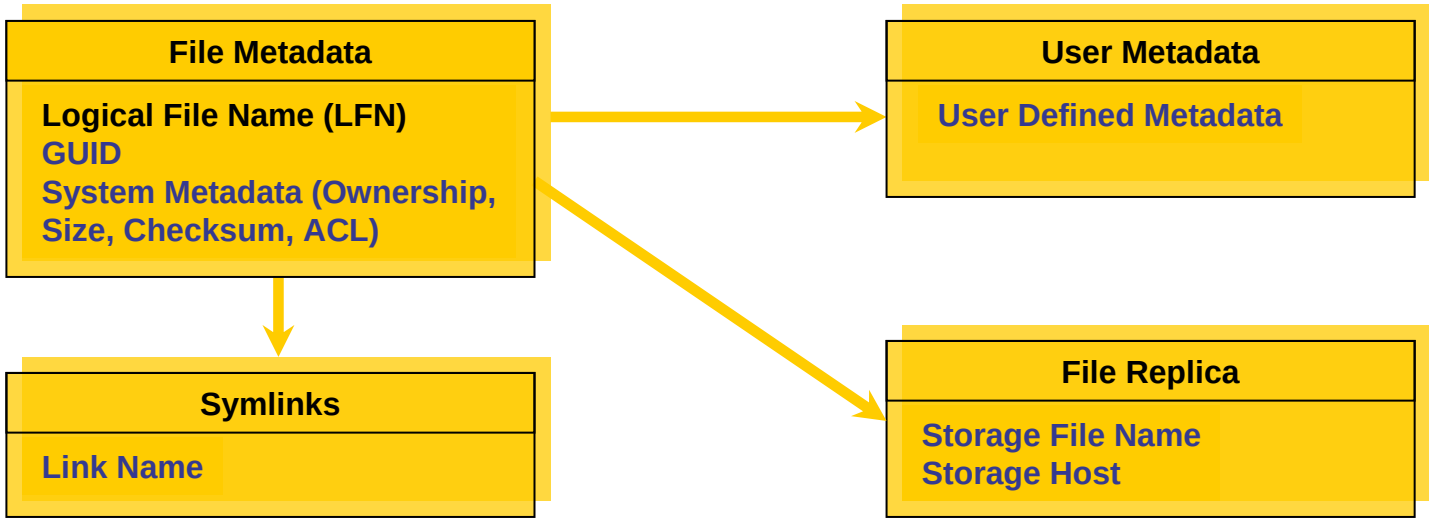
LCG File Catalog

- Based on lessons learned in DC's (2004)
  - Fixes performance and scalability problems seen in EDG Catalogs
    - Cursors for large queries
    - Timeouts and retries from the client
  - Provides more features than the EDG Catalogs
    - User exposed transaction API
    - Hierarchical namespace and namespace operations
    - Integrated GSI Authentication + Authorization
    - Access Control Lists (Unix Permissions and POSIX ACLs)
    - Checksums

LCG File Catalog

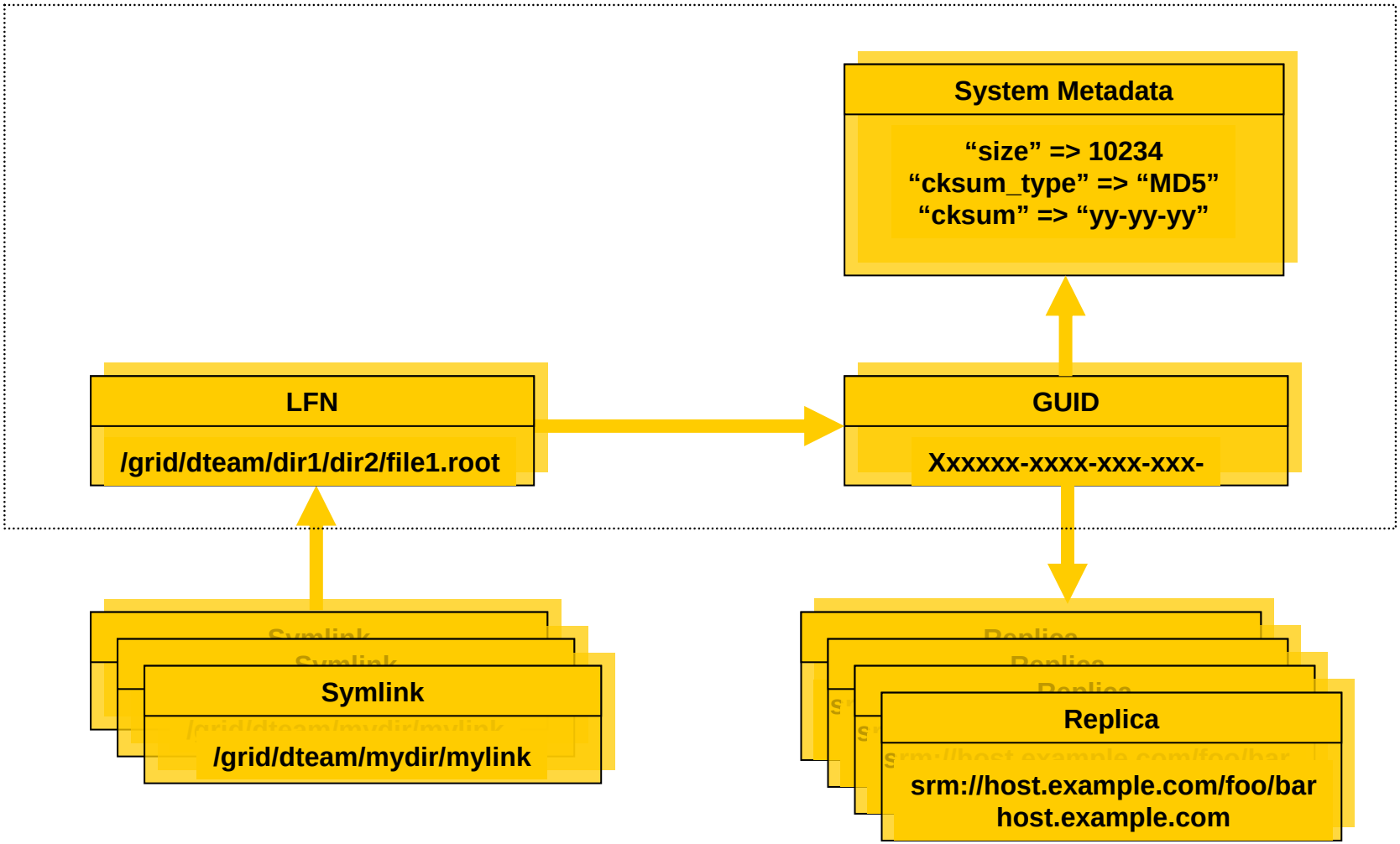
- Based on existing code base
  - Supports Oracle and MySQL database backends
- Aim is to enable rapid development and deployment
  - Integration with GFAL and lcg_util complete
  - Performance and Scalability testing done (up to 40 millions entries)
  - First version deployed for Certification – October 2004
  - Pre-production service running at CERN – December 2004
  - Migration of entries from EDG catalog to LFC done - January 2005
  - In pre-production at DESY/HG-01-GRNET/LIP/Bari/Pisa
  - POOL Integration will be provided – May 2005

LCG File Catalog Schema



- LFN acts as main key in Database. Has:
  - Unique Identifier (GUID)
  - Information on Physical Replicas
  - Symbolic Links to it
  - A small amount (one field) of user attached metadata

Relationships in the Catalog



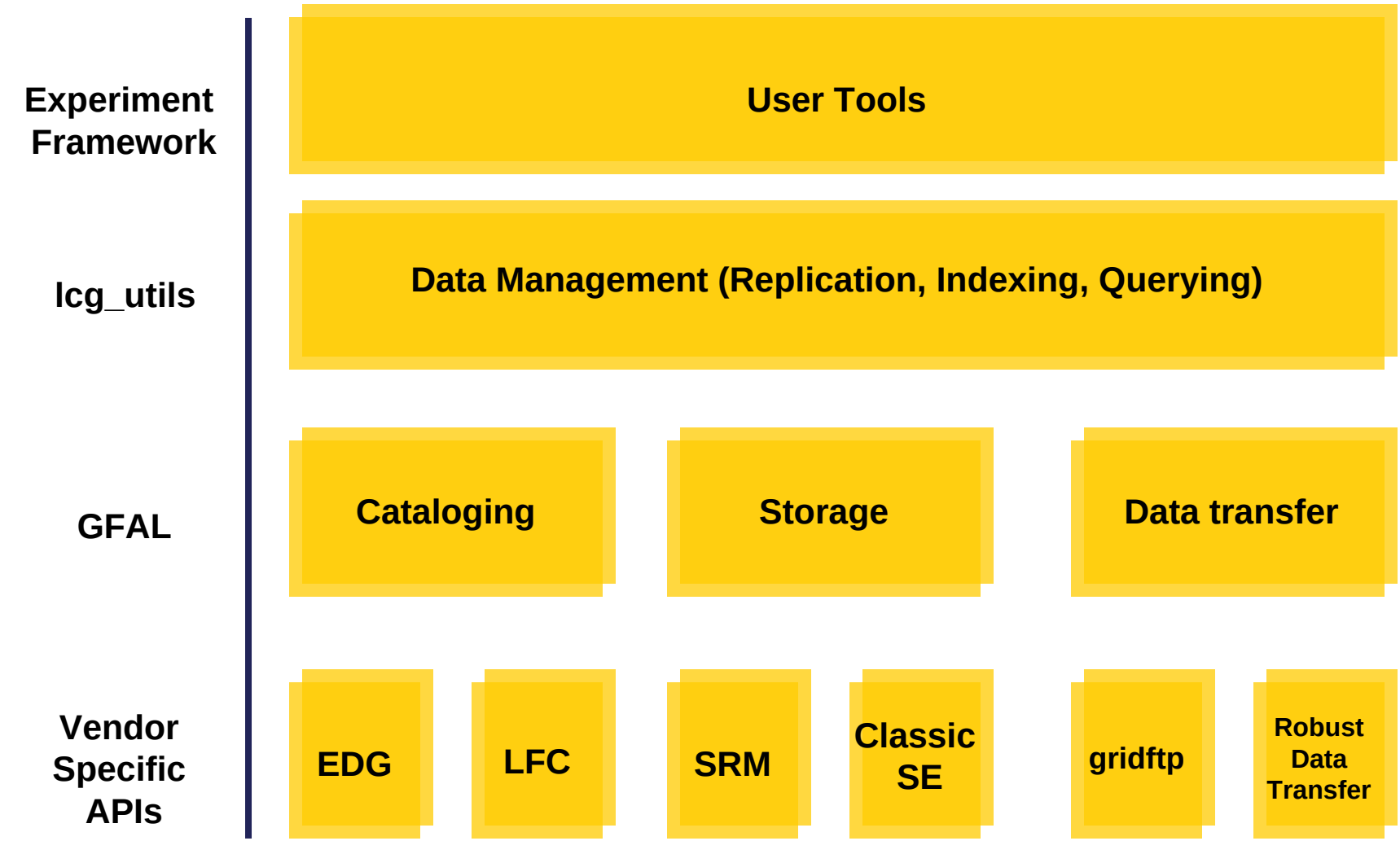
Features (1/2)

- Namespace operations
  - All names are in a hierarchical namespace
  - mkdir(), opendir(), etc…
  - Also chdir()
  - GUID attached to every directory and file
- Security – GSI Authentication and Authorization
  - Mapping done from Client DN to uid/gid pair
  - Authorization done in terms of uid/gid
  - VOMS will be integrated (collaboration with INFN/NIKHEF)
    - VOMS roles appear as a list of gids
  - Ownership of files is stored in catalog
  - Permissions implemented
    - Unix (user, group, all) permissions
    - POSIX ACLs (group and users)

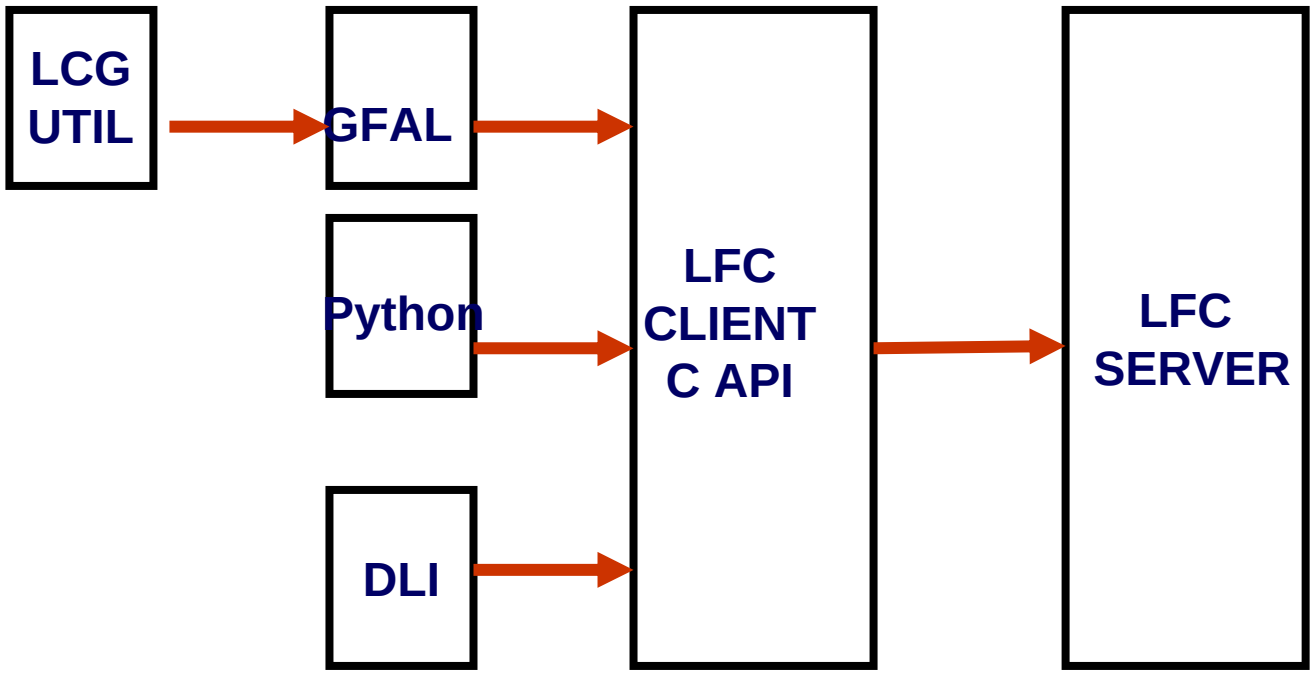
Features (2/2)

- Transactions
  - Exposed to user
    - starttrans(), endtrans(), aborttrans() methods
  - Auto-rollback on failure of mutating method call
- Cursors for queries
  - Modelled on opendir()/readdir()/closedir()
- Retries and timeouts
  - Make client resilient to temporary outage of server

Layered Data Management APIs



LFC interfaces



Interfaces

- GFAL is the common catalog interface used by the replication tools (lcg_util)
- DLI is the common catalog interface used by the Workload Management System
  - It is a web service interface
  - It has been used by CMS for PubDb/RefDb
  - A DLI interface to LFC is being tested

Implementation

- Server and clients are implemented in C
- Database backend: Oracle or MySQL
- Oracle interface uses ProC
- No catalogued procedure
  - Easier to port to a different DB backend
  - DB queries are very simple (key access)

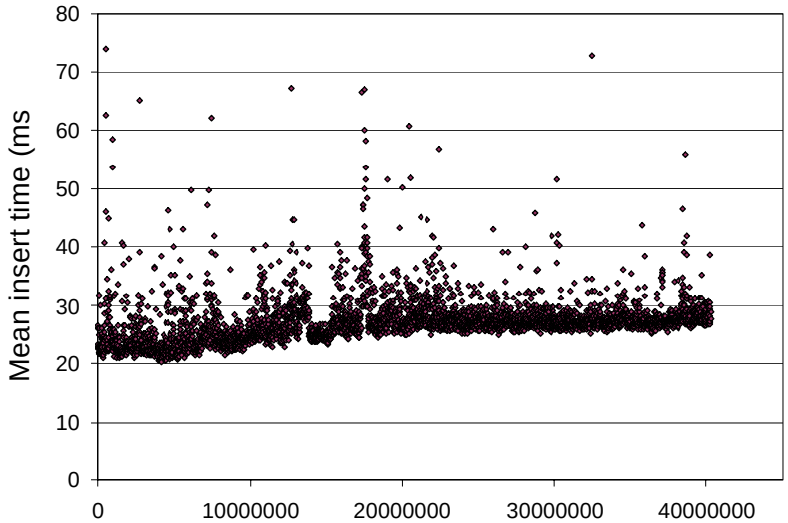
Overview

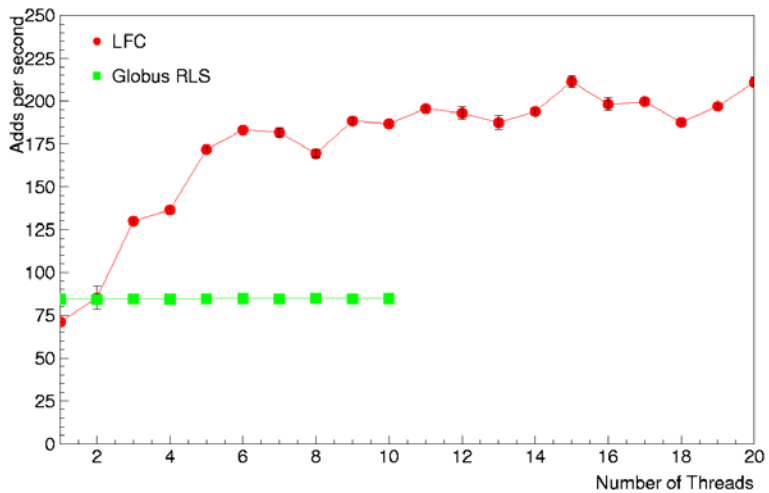
- LFC Architecture
- **LFC/Fireman Tests**
- LFC Deployment

LFC Performance (i) - Inserts

- Mean insert time as number of entries increased up to 40M remains below 30 ms
- EDG mean insert time was ~40 ms with 500,000 entries

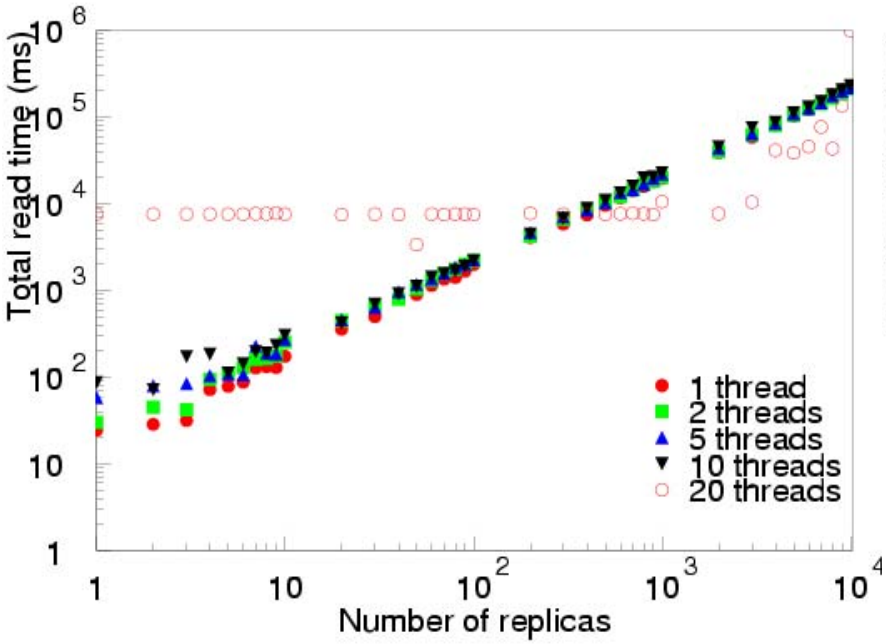
- Insert rate, with increasing number of client threads, for ~1M entries
- Increases up to ~200 adds/sec up to server thread limit
- Globus RLS gave ~84 adds/sec when run with consistency

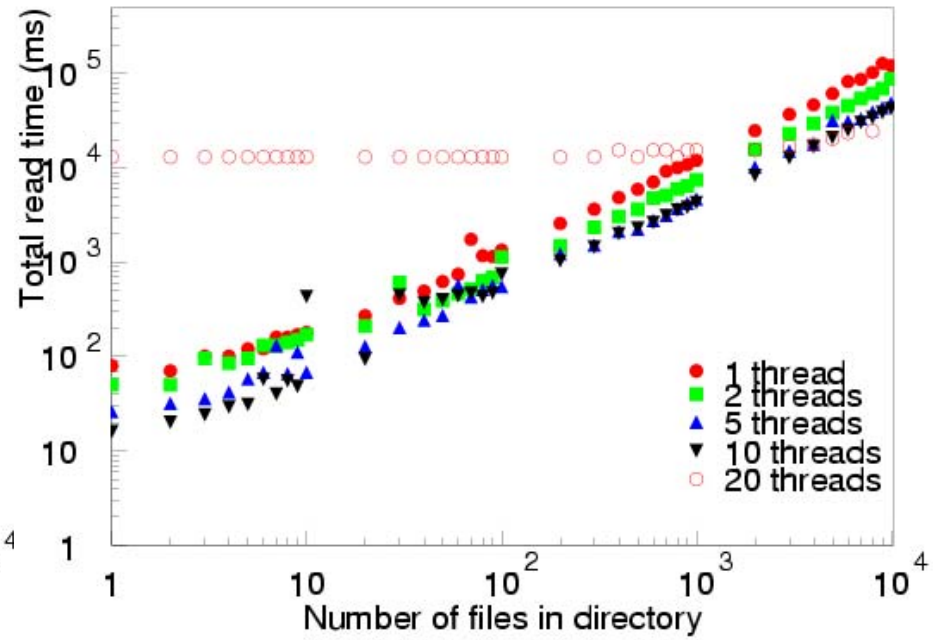




LFC Performance (ii) - Queries

- Time to list and stat all replicas of a file proportional to number of replicas
- Time to read a directory is directly proportional to



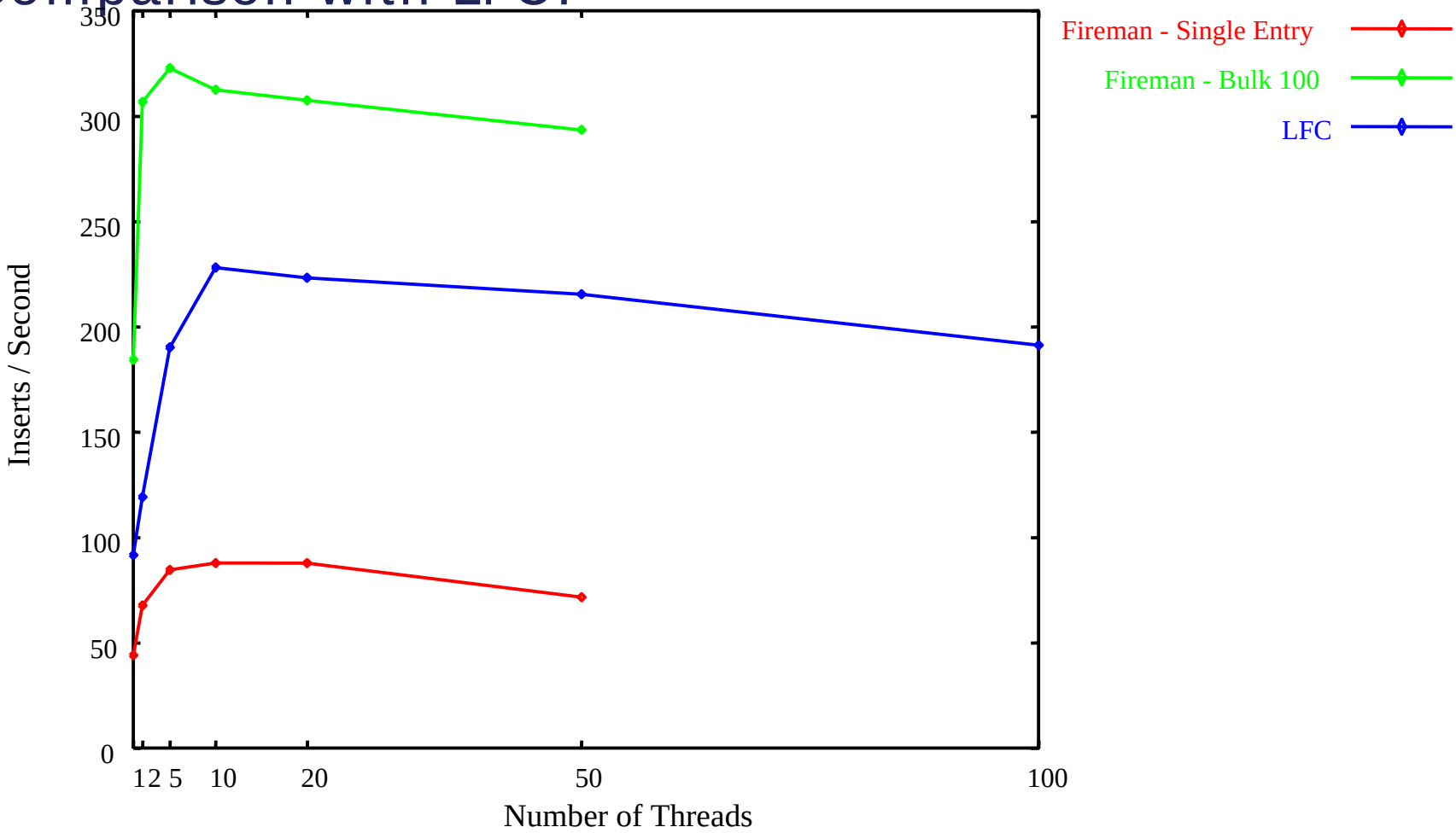


LFC Tests Summary

- LFC has been tested and shown to be scalable to at least:
  - 40 million entries
  - 100 client threads
- Performance improved with comparison to RLSs
- Stable :
  - Continuous running at high load for extended periods of time with no crashes
  - Based on code which has been in production for > 4 years
- Tuning required to improve bulk performance

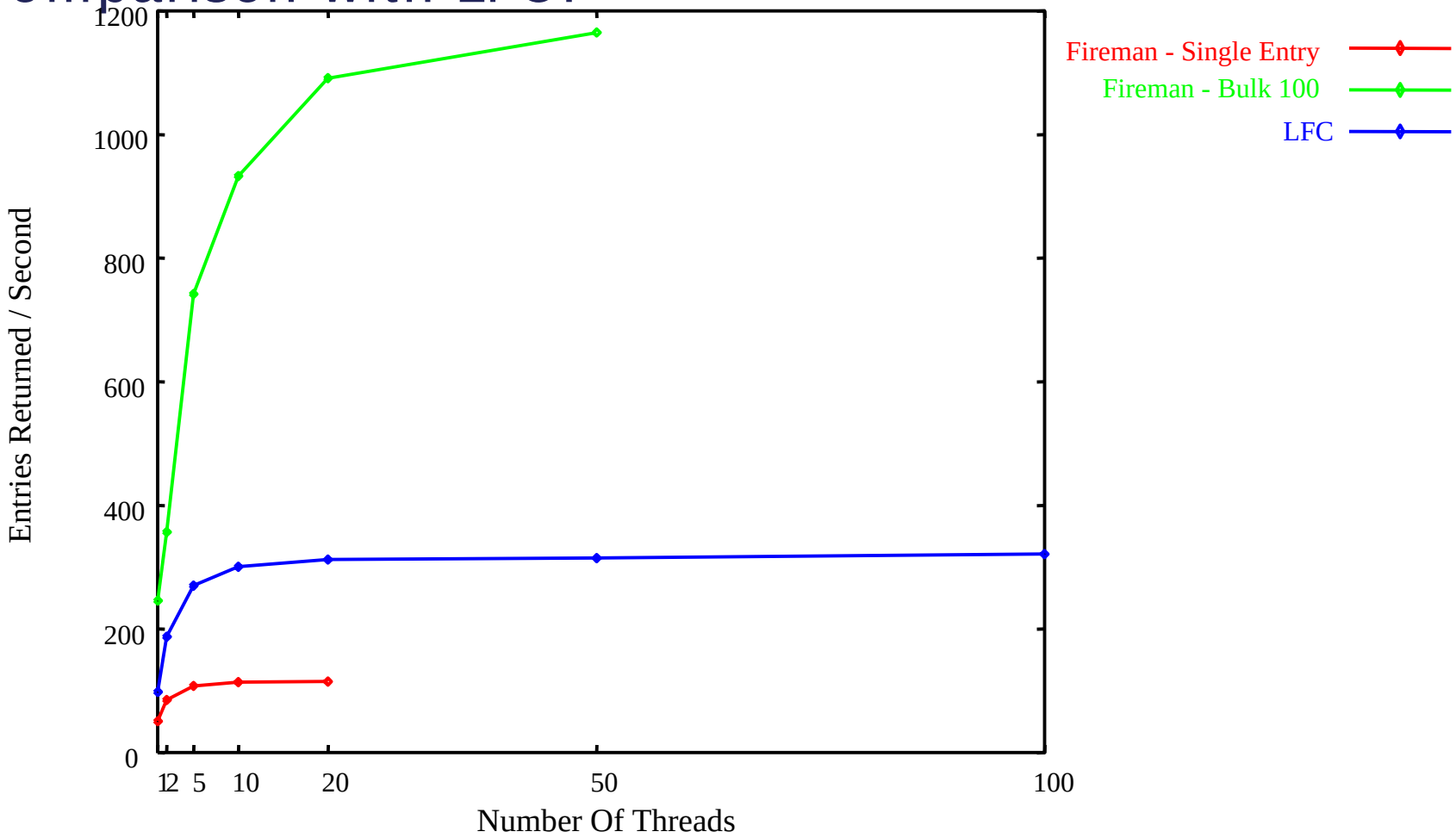
FiReMan Performance - Insert

- Comparison with LFC:



FiReMan Performance - Queries

- Comparison with LFC:



Tests Conclusion

- Both LFC and FiReMan offer large improvements over RLS
- Still some issues remaining:
  - Scalability of FiReMan
  - Bulk Entry for LFC
- More work needed to understand performance and bottlenecks
- Need to test some real Use Cases

Overview

- LFC Architecture
- LFC/Fireman Tests
- **LFC Deployment**

From the RLS to the LFC

- EDG-RLS
  - Central catalog
  - Performance and Scalability problems during experiment DCs in 2004
- LFC developed as a possible solution for LCG
  - Central catalog or local catalog
- Catalog called “LCG File Catalog”, but not HEP specific !
- Secure LFC is part of LCG-2_4_0
  - Easy installation/configuration : RPMs or tarballs
  - YAIM components provided (for MySQL only at the moment)
  - LFC Administrator’s Guide
    http://goc.grid.sinica.edu.tw/gocwiki/How_to_set_up_an_LFC_service
  - Several sites are currently testing the LFC

RLS to LFC Migration

- Simple script provided by the CERN IT-GD group
  - Queries the LRC and RMC databases directly (for efficiency reasons)
  - No user-defined entries migrated by default
    - But, script already exists for file size and checksum
    - Can be changed on demand if user-defined attributes fit in the LFC model

- Example :

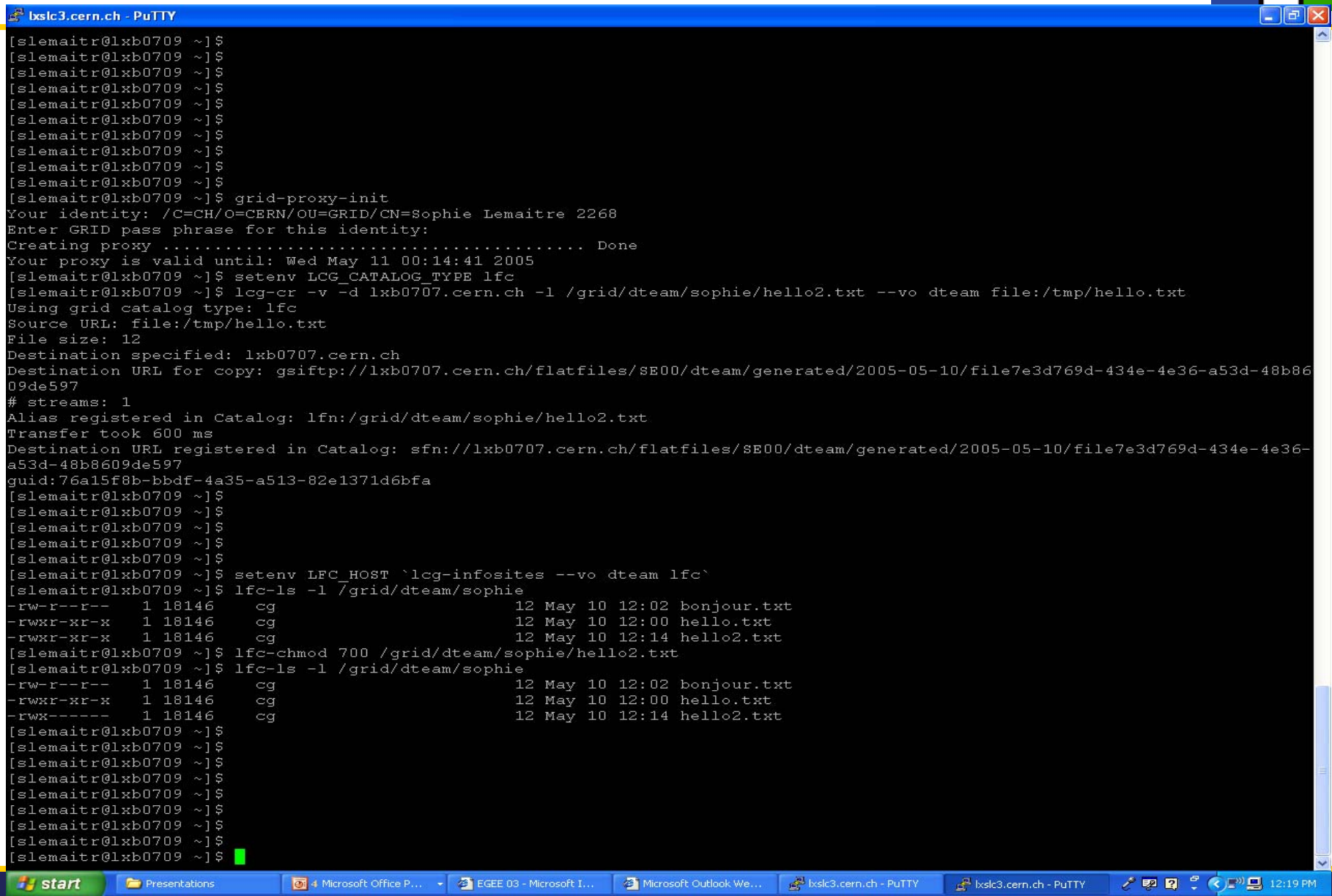
```
./migrate_RLS_entries --db-vendor MySQL --host localhost
--lrc-user lrc_zeus --lrc-passwd lrc_password
--rmc-user rmc_zeus --rmc-passwd rmc_password
--path /grid/zeus
```

- More details at :
  http://goc.grid.sinica.edu.tw/gocwiki/How_to_migrate_the_RLS_entries_into_the_LCG_File_Catalog_%28LFC%29

LFC Integration

- LFC already accessible through :
  - LFC command line interface
    - lfc-ls, lfc-mkdir, lfc-ln, lfc-rm, lfc-rename, lfc-getacl, etc.
  - GFAL
  - lcg_util
    - export LCG_GFAL_INFOSYS=<BDII_hostname>
    - export LCG_CATALOG_TYPE=lfc
    - export LFC_HOST=`lcg-infosites --vo dteam lfc`
  - Python interface
  - POOL (on going)

LFC usage example



LFC future developments

- Bulk operations
- Sessions
- Integration with the ARDA Metadata Catalog
- Integration with VOMS/LCAS/LCMAPS
- Integration with AUTHZ

Questions?

Jean-Philippe.Baud@cern.ch

Sophie.Lemaitre@cern.ch