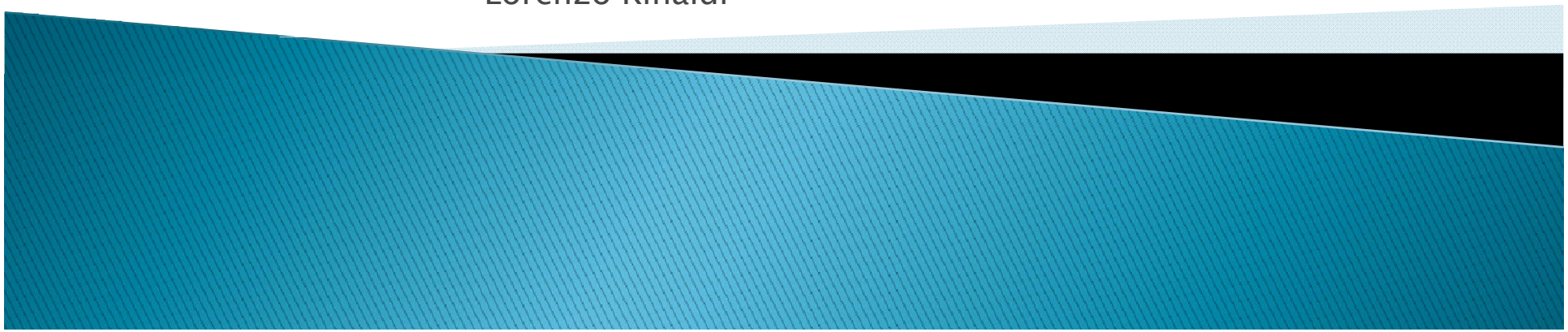
A lightweight high availability strategy for Atlas LCG File Catalogs

Daniela Anzellotti
Alessandro De Salvo
Barbara Martelli
Lorenzo Rinaldi

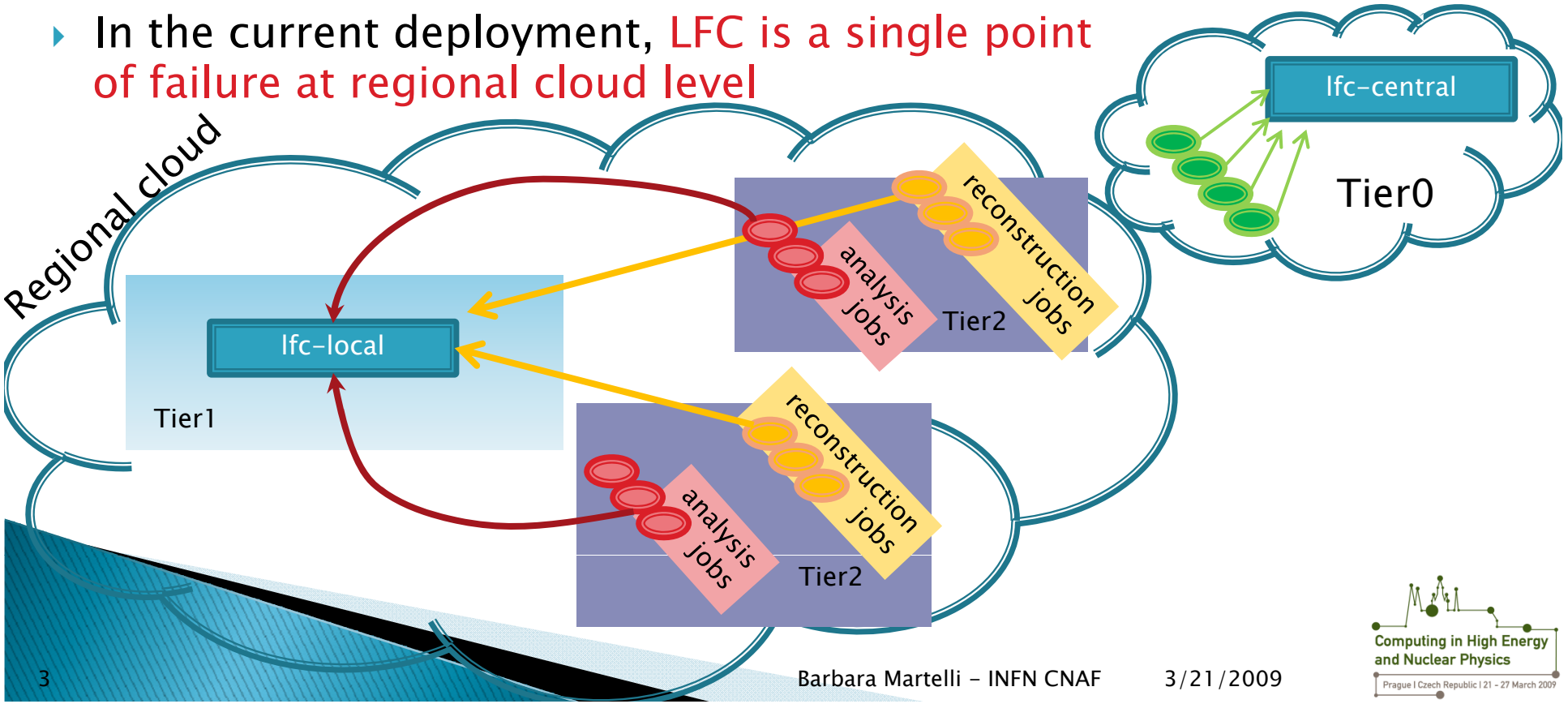


Outline

- Motivations
  - Limits of current ATLAS LFC deployment
- Purposes
- Proposed solution: LFC on Oracle DataGuard
  - Oracle DataGuard technology
  - Applications failover
  - Deployment outlook
- Feasibility tests:
  - Functionality/performance/robustness results
- Conclusions

Motivations: limits of current deployment

- ATLAS uses the LFC as both central and local catalogue:
  - **Central LFC at T0**: stores entries referring to production data
  - **Local LFC at each T1**: store entries referring to raw data, ESD, AOD, TAGs delivered to that *regional cloud*
- Therefore a local LFC down implies a total interruption of all reprocessing and analysis operations in that whole regional cloud
- In the current deployment, LFC is a single point of failure at regional cloud level



Purposes

- Disaster recovery: setup an high availability deployment allowing to quickly face critical events which imply a down of the local LFC
  - Without requiring any change in ATLAS s/w or LCG middleware
    - Not even configuring a try-list of alternate LFC
  - Possibly simple and fast to install and manage with current sites know-how
- Read requests load balancing as secondary requirement
  - LFC load is quite low, but could become and issue for read operations at LHC startup
- Transparency: jobs started after the LFC failure need to be redirected to the new catalog without any human intervention
  - No reconfiguration of LFC_HOST variable
  - No need for DBA or LFC admin to activate the failover process

Proposed solution: LFC on Oracle DataGuard

The idea is to put together various technologies already used and widely tested

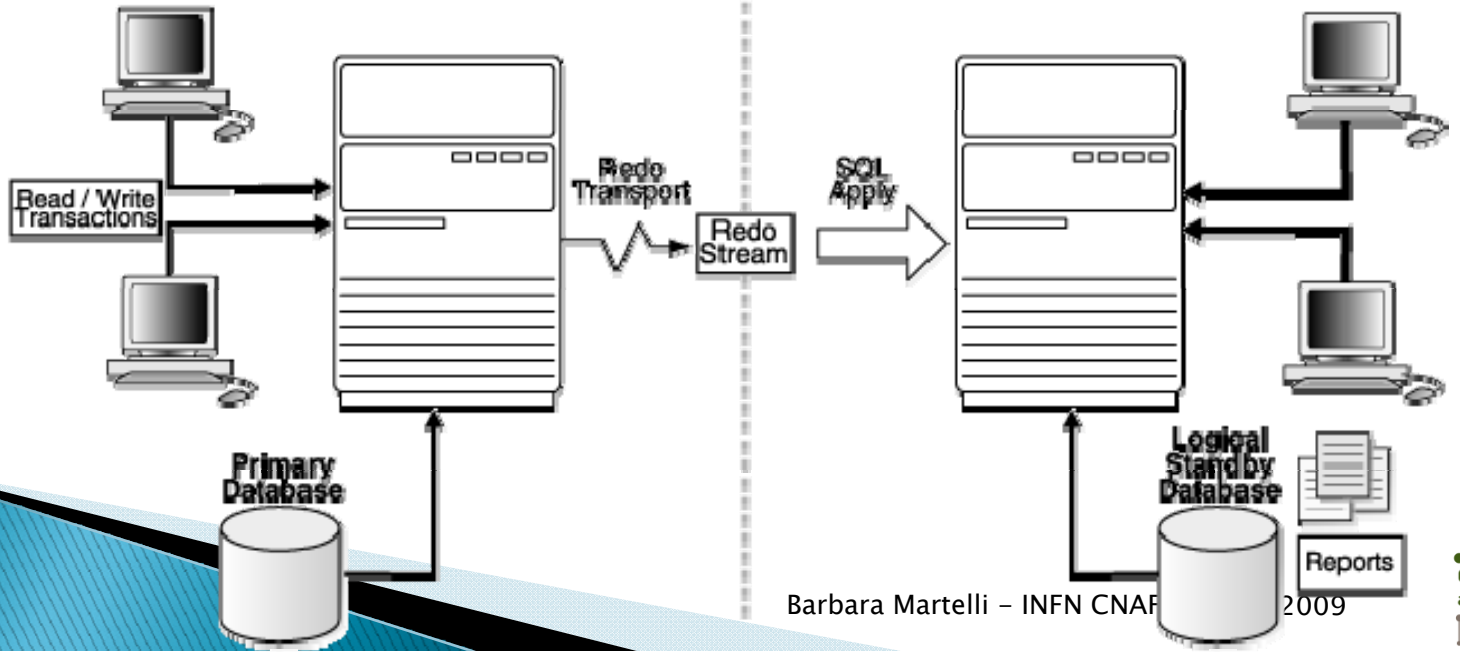
- Exploit Oracle DataGuard technology in order to create a standby instance at a remote site
  - Standby site: INFN-ROMA1 ATLAS Tier2
- Exploit LFC RUN_READONLY configuration variable in order to set up a read-only LFC on top of the standby database
  - Already used and widely tested
- Exploit dynamic DNS updates in order to expose the same logical name to clients in case of failover
  - nsupdate script run via Nagios like other HA services

Oracle DataGuard

- Oracle Dataguard is a high availability technology which consists of one production database (primary) and one or more standby databases
- You can change the role of a database using either a switchover or a failover
  - **Switchover**: role reversal between the primary database and standby
    - Ensures no data loss
    - Useful for maintenance operations
  - **Failover**: in the event of a catastrophic failure of the primary database, the failover results in a transition of a standby database to the primary role
    - Some transactions can be lost, but consistency is preserved
- Enabling **fast-start failover** DataGuard fails over *automatically* when the primary database becomes unavailable, **with no DBA intervention**
- In order to reinstate a failed primary database in a logical standby configuration DBA actions are needed
  - must re-create the database from backups taken from the current primary

DataGuard Configuration

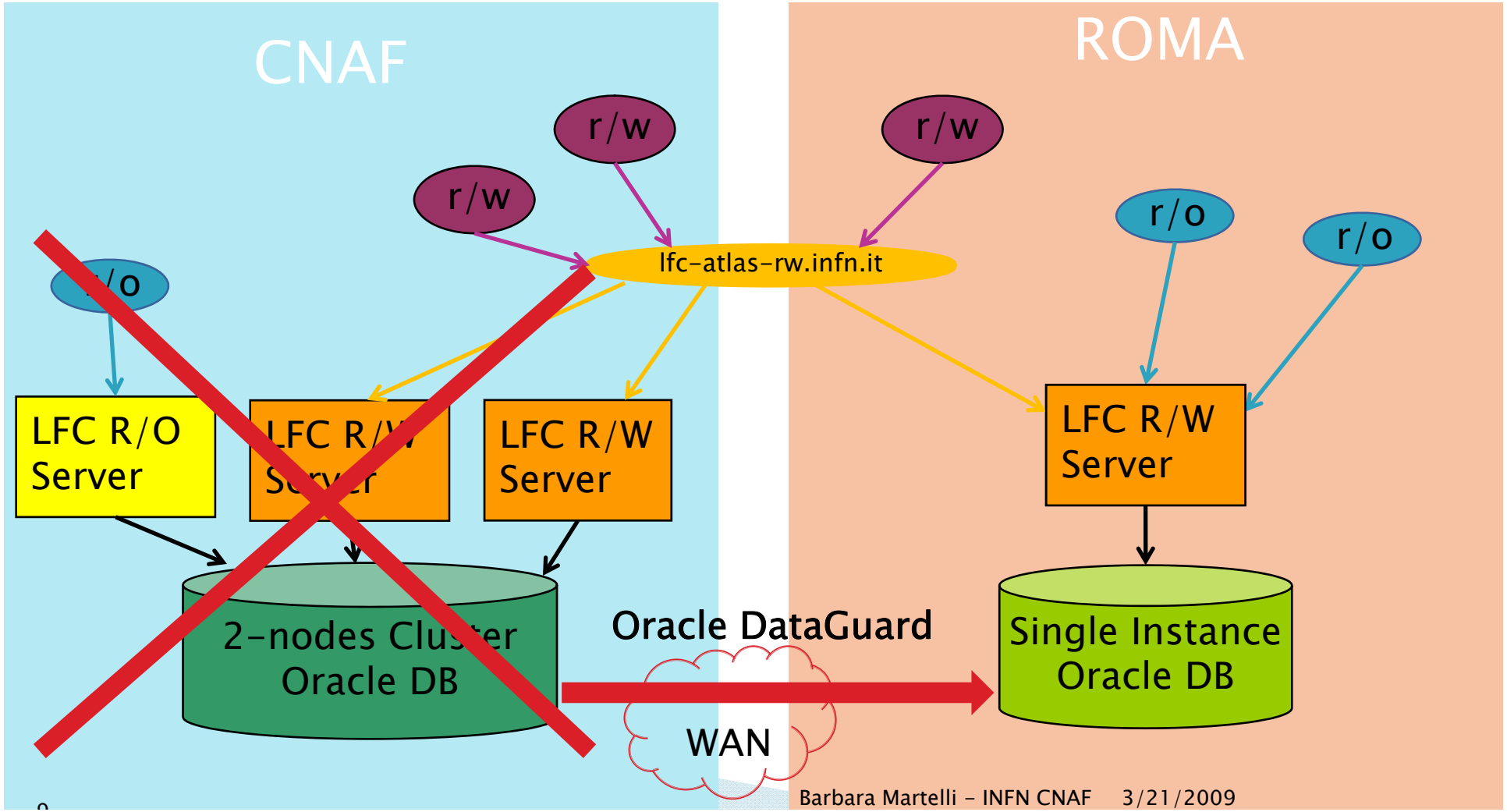
- Configuration: **Logical Standby.** It's kept synchronized with primary database through *SQL Apply* which transforms the data in the redo received from the primary database into SQL statements and then executing the SQL statements on the standby database
  - Supports read-only operations on the standby
- Protection mode: **Maximum Performance.** a transaction commits when the redo data is written locally. The redo data is sent to the standby database and applied **asynchronously**
  - Ensures network latency/delay don't impact on primary performances



Applications Failover

- With DataGuard you have the automatic failover of the LFC Oracle backend, but what about clients?
- A nagios script in Roma monitors its local database back-end
  - Checks column DATABASE_ROLE from V$DATABASE view
  - If the role changes from *standby* to *primary* -> a failover has happened
  - Set RUN_READONLY="no" in Roma LFC
  - Restarts the lfcdaemon service
  - nsupdate in order to reconfigure DNS alias

Deployment Outlook



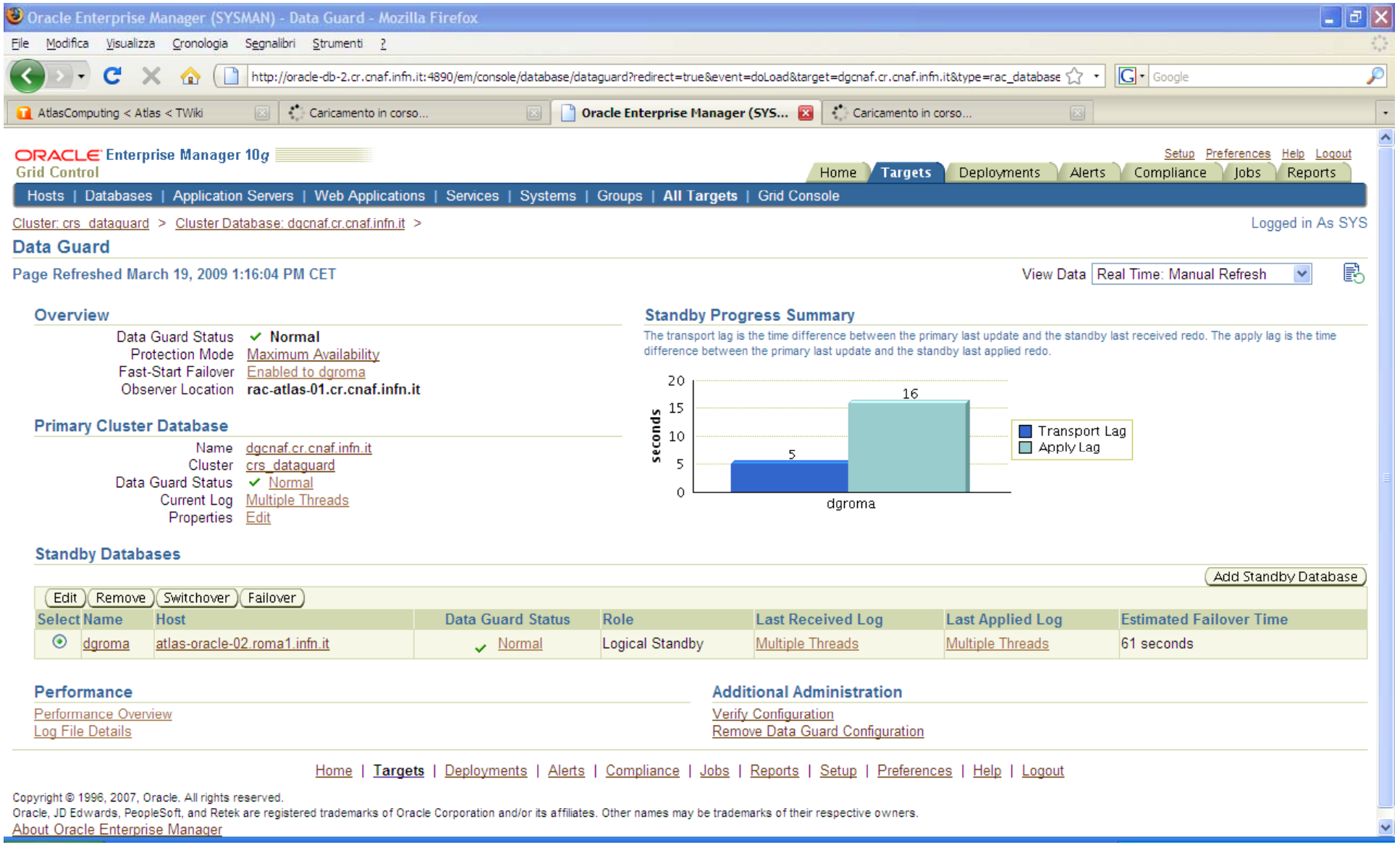
Feasibility Test

- Performed various tests with increasing number of files and threads, the heaviest one:
  - Python test script using LFC API functions. Basically a loop where the following actions are performed:
    - start session
    - create an entry on LFC
    - add a replica
    - close session
    - no delays between actions
  - In parallel, with a delay of 3 minutes, another scripts checks the consistency between CNAF and ROMA LFCs
  - 1M entries per thread
  - 40 simultaneous threads

Results

- Test run for one hour
  - all inserted entries were correctly found in Roma LFC
  - Maximum lag measured: 23 seconds
    - 5 seconds transport lag
    - 18 apply lag
- Test restarted. 5 minutes later, simulated failure in primary database (power-cut of both machines)
  - Database automatic failover succeeded in 10 seconds
    - No manual intervention needed
  - Roma LFC restarted in 5 seconds after failover
    - No manual intervention needed
  - R/W clients: successfully keep on inserting entries without reconfiguration

Oracle Enterprise Manager View



Conclusions

- DataGuard can be used as disaster recovery strategy for Oracle LFC back-ends
  - Automatic failover
  - Apply lag is less than 30 seconds (required <10 minutes)
    - r/o clients can connect to standby DB
  - Loss of few entries during failover is tolerable
    - As primary LFC is fully redundant at Tier1, Failover happens only in particularly severe situations
  - With few simple nagios scripts is possible to implement a transparent failover for applications

Backup slides

Requirements to be a standby Tier2

- Enable the Archivelog mode
  - CNAF LFC database size: 11 GB (10 GB for real data and 1 GB for system data + indexes)
  - ~ 12 M files and ~ 8 M replicas (average 800 B per entry)
  - Redolog size: 512 MB, 4 files, a file is filled up in 3 hours during peak load.
  - ~ 4 GB of archived log per day
  - ~ 4 GB of *standby logs* per day (needed for real time apply)
- Install ASM as database storage manager is the preferred one
- The SYS password must be the same on all systems
  - Having an independent instance may be required
- Both databases must have the same operating system, same platform (32 or 64 bit), OS release can differ
- The same release of Oracle Database **Enterprise** Edition must be installed on both databases.
- Write rate at primary DB ~ 220 kB/s