Analisi di un servizio SRM: StoRM

Luca Magnoni INFN-CNAF

Scuola per utenti INFN della GRID

27 November 2007

- General Parallel File System (GPFS)
- The StoRM service
- Deployment configuration
- Authorization and ACLs
- Conclusions.

Definition of terms 1/2

- **Distributed File System** The generic term for a client/server or "network" file system where the data is not locally attached to a host. Network File System (NFS) is the most common distributed file system currently in use.
- **Storage Area Network (SAN) File System** provides a means for hosts to share Fiber Channel storage, which is traditionally separated into private physical areas bound to different hosts. A SAN File system mounts storage natively on only one node and connects all other nodes to that storage by distributing the block address of that storage to all other nodes. Scalability is often an issue due to the metadata managers and the large network transactions required in order to access data.

Definition of terms 2/2

- **Symmetric File Systems** A symmetric file system is one in which the clients also host the metadata manager code, resulting in all nodes understanding the disk structures. A concern with these systems is the burden that metadata management places on the client node, serving both itself and other nodes, which can impact the ability of the client node to perform its intended computational jobs.
- **Asymmetric File Systems** An asymmetric file system is a file system in which there are one or more dedicated metadata managers that maintain the file system and its associated disk structures.

Parallel and cluster file system

- A cluster file system allows large numbers of disks attached to multiple storage servers to be configured as a single file system.
- A cluster file system provides:
  - Transparent parallel access to storage devices while maintaining standard UNIX file system semantics.
  - High-speed file access to applications executing on multiple nodes of a cluster.
  - High availability and fault tolerance.

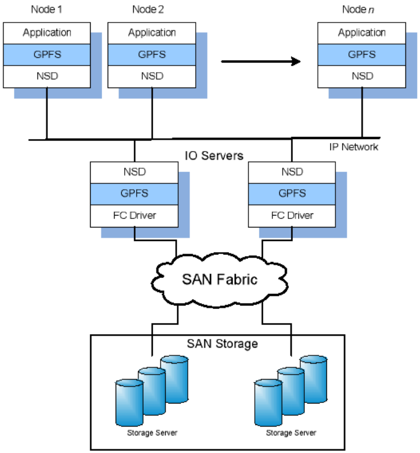
General Parallel File System (GPFS)

The IBM General Parallel File System (GPFS) is a high-performance shared-disk file system that provides fast, reliable access to a common set of file data from two computers to hundreds of systems. It belong to the **symmetric file system** category.

- GPFS integrates into storage environment by bringing together mixed server and storage components to provide a common view to enterprise file data.
- GPFS provides online storage management, scalable access and integrated information lifecycle tools capable of managing petabytes of data and billions of files.

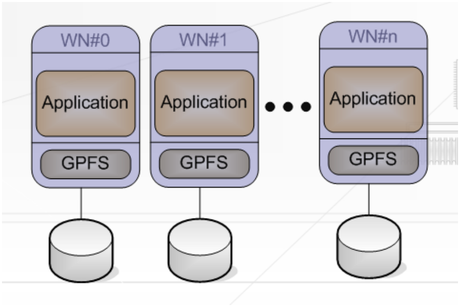
The GPFS file system

- A GPFS file system is built from a **collection of disks** which contain the file system data and metadata.
- A file system can be built from a single disk or contain thousands of disks, storing Petabytes of data.



GPFS Application interface

- Applications can access files through **standard UNIX file system interfaces** or through enhanced interfaces available for parallel programs.
- Parallel and distributed applications can be scheduled on GPFS clusters to take advantage of the shared access architecture.



StoRM: Storage Resource Manager

Result of collaboration between:

- **INFN - CNAF**: where StoRM was thought and the development started (within Grid.IT project context).
- **ICTP - EGRID** Project: to build a pilot national grid facility for research in Economics and Finance

Now, the StoRM team is:

- leaded by Antonia Ghiselli
- **INFN - CNAF**: Alberto Forti, Luca Magnoni, Riccardo Zappi
- **ICTP -EGRID**: Ezio Corso

A Storage Resource Manager service: StoRM

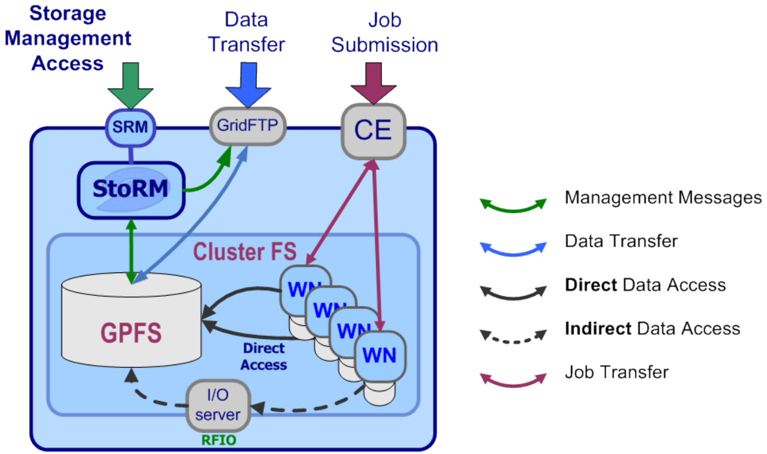
StoRM is a **storage resource manager** for disk based storage systems, implementing the SRM interface v2.2.

- Manages space and files in a generic disk based storage resources.
- Relies on the **aggregation functionalities** provided by file systems.
- Designed to be **indipendent from the different file system** supported.
- **Highly scalable and configurable**, can be used at site with different size and requirements.
- Allow to **expose in Grid via SRM interface files stored in a standard file system**.

StoRM and cluster file system

- It is designed to take advantage from **high performing cluster file system**, as GPFS from IBM and Lustre from ClusterInc., but it supports also every standard POSIX FS.
- It allows **direct access** (through the protocol *file://*) to the storage resource, as well as other standard grid protocol as *gsiftp* and *rfio*.
- Authentication and authorization are based on the **VOMS** credential.
- Permission enforcing are based on setting **physical ACLs** on files and directories.

StoRM role in a site



StoRM features 1/4

- Dynamic file and space management as defined by the SRM v2.2 functionalities.
- Support for different file system provided by a **driver** mechanism. Easy to expands.
- It's able to works on different file system type **at the same time**.
- Support for **file** protocol, natively.
- As well as for other standard protocol as **rfio** and **gridftp**.
- Guaranteed dynamic space reservation (using underlying file system functionalities)

Allow to easily **expose via SRM the file stored in a classic SE**.

StoRM features 2/4

- **Storage Area (SA)**:
  - Storage Area can be defined editing the StoRM namespace configuration.
  - Each SA is addressed by path and by **Storage Area token**.
- **Quota**, relying on the underlying file system capabilities (as GPFS).
- Light and flexible **Namespace mechanism** based on file system structure.
- Space and file garbage collector.

StoRM features 3/4

Layered security mechanism:

- **VOMS** compliant.
- **StoRM retrieves authorization information** from:
  - External services (as the LFC catalogue).
  - Local configuration.
- Enforcing of **file system ACL** on file and directory at user/group level.

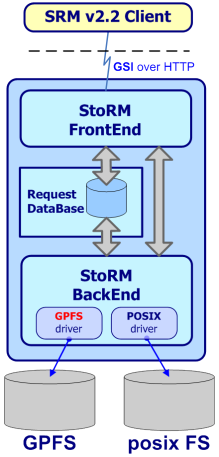
StoRM features 4/4

- Interact with **Data transfer service** (GridFTP) for copy functionalities.
- Interact with **User Mapping service** (LCAS/LCMAPS) for authorization operations.
- Publish storage information for service discovery.

StoRM architecture 1/2

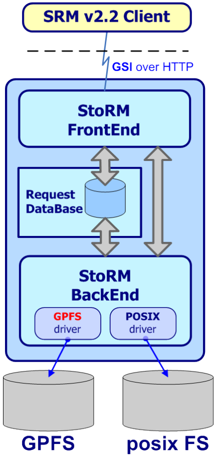
StoRM has a multilayer architecture.
The **Frontend (FE)** component:

- exposes the web service interface.
- manages user authentication.
- manages connection with clients.
- store asynchronous request into the database
- direct communication with Backend for synchronous request.
- retrieve request status.



StoRM architecture 2/2

- **Database** is used to store SRM request data and the internal StoRM metadata.
- The **Backend (BE)** is the core of StoRM.
- it executes all synchronous and asynchronous SRM request
- manages user authorization
- enforces permissions
- interacts with other grid services
- provides support to file systems through a **driver mechanism**

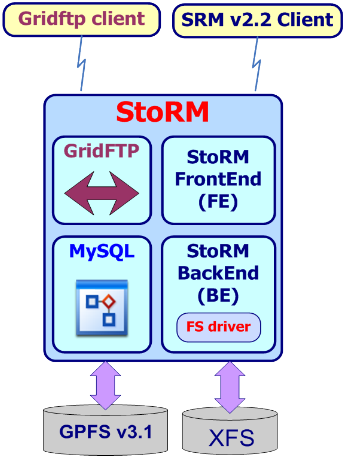


Deployment on single site

All the component:

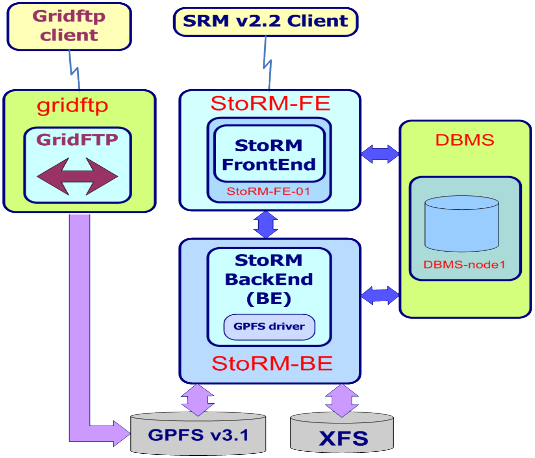
- **StoRM Frontend**
- **StoRM Backend**
- **MySQL**
- **GridFTP**

are **deployed on the same host**



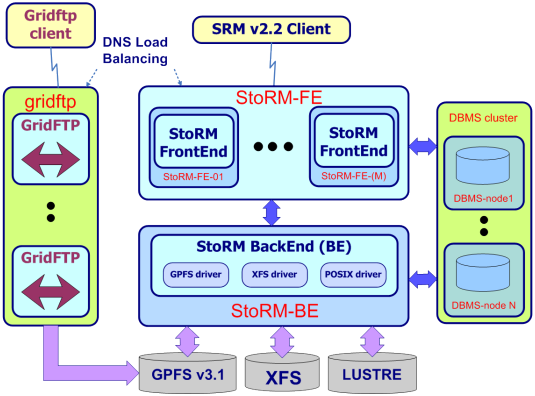
Deployment on different host

- StoRM architecture allow to **exchange information** over network .
- Each component can be deployed on a dedicated host.



Deployment on cluster

- StoRM supports **component replication**
- This allow to satisfy the **high availability and scalability** requirements.

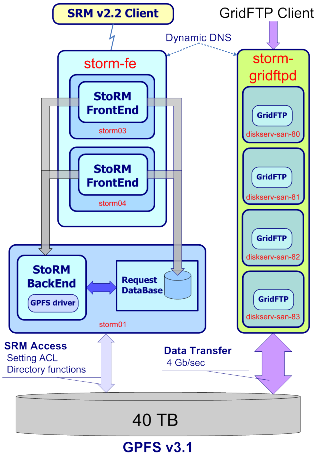


StoRM at CNAF Tier 1

The **CNAF T1** use StoRM as SRM v2.2 endpoint for the **T0D1** storage class.
We have set up a clustered configuration to make StoRM working in a production scale scenario:

- **2 FE** hosts (dual AMD Opteron 2.2 GHz, 4 GB).
- **1 BE** host (dual Intel Xeon 2.4 GHz, 2GB), shared with the MySQL DBMS.
- **40 TB** of disks available for experiment tests.
- **4 GPFS disk server** (dual Intel Xeon 1.6 GHz, 4 GB), with the *gridftpd* server in a dynamic DNS configuration.

StoRM at CNAF Tier 1



StoRM at CNAF Tier 1 - Some number on performances

Tests made by LHCb to validate their analysis scenario with StoRM and GPFS.

- Data transfer from CERN through CNAF via *gridftp*. Before and after each transfer a set of SRM requests are performed to StoRM.
- Average bandwidth: 240 MB/s.
- Bandwidth peak: 370MB/s.
- Stress test on SRM request on StoRM:
  - 100k handled.
  - At least 400k interactions (PtP files, statusPtP, Ls, Rm, etc.).
  - 600 parallel process that submit request to StoRM
  - Low failure rate ¡ 0-2%.
  - Execution rate of 40 request per seconds

This tests help to male optimization on database interactions.

StoRM status

StoRM general status:

- Latest stable release: v**1.3.18**
- Included in the INFN Grid release.
- Available by **YAIM**, **APT** - **rpm** or **Quattor**.
- File system driver currently available:
  - GPFS v2.3
  - GPFS v3.x (Improved ACL management using the new GPFS API)
  - XFS
  - generic PosixFS (as Lustre, Ext3). ACL enforcing made through Linux *setf/getfacl()* functionalities.
- StoRM web site: *http://storm.forge.cnaf.infn.it* for documentation, installation guide and client tutorial.

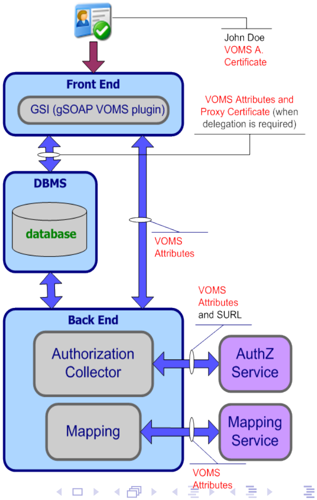
SRM v2.2 command line client

Together with StoRM a command line client for the SRM v 2.2 is available:

- Written in C++ using the GSOAP toolkit.
- Compatible with every SRM v 2.2 implementation.
- Provides the SRM syntax in a classic UNIX style.
- Example and tutorial available on the StoRM site.
- Usage example:
  clientSRM ptp -e httpg://ibm139.cnaf.infn.it:8444 -s srm://ibm139.cnaf.infn.it:8444/dteam/test111 -p

VOMS attributes in StoRM

- The VOMS attributes are retrieved from the user proxy by the StoRM Frontend through the CGSI_GSOAP plugin.
- The attributes are stored into dedicated tables in the StoRM catalog, to represents the user credential and the requestor identities.
- The attributes are evaluated for **authorization** operation by the StoRM Backend.



Approachable rules

- StoRM is able to represents the **authorization rule** for the SA, or in other words which VO can approach (or view) the SA, by a configuration features named **approachable rule**.
    - Approachable rules are defined per Storage Area and they are expressed by **regular expression in terms of FQAN**.
    - The SURLs within a request will be considered well formed if and only if the requestor is compliant with the specific App-rule.
    - The default value for app-rules is ".*", so all FQAN can approach every SA.

Authorization sources

StoRM is able to interact with external authorization service (named **authorization sources**) to perform the authorization decision.

- OGSA AuthZ WG is defining a standardized AuthZ Service Interface.
- Waiting for a standardized interface, we suppose that external service answer question like *"Can USER perform the ACTION on this RESOURCE"*.

We distinguish two kind of authorization sources:

- **Local**: based on configuration file or local information.
- **Global**: external service.

Local authorization sources

Local authorization source holds AuthZ policies for file or directories.

- Basic policy: Permit/Deny All .
- **Regular expression per path and FQANs** (permissions are equals on the same path) A simple XML file which defines AuthZ policies on the bases of directories (i.e. files within a directory will hold the same ACL).

Global authorization source

Global authorization source holds AuthZ policies valid for all storage resources accessible by VO-users.

- Currently StoRM can uses ECAR, a client for the **LFC catalogue**, to retrieve AuthZ information based on the ACLs on LFN.
- In the future version StoRM will have a Policy Enforcement Point (PEP) bound with external tools as the G-PBox. (G-PBox is an highly distributed policy management and evaluation framework).

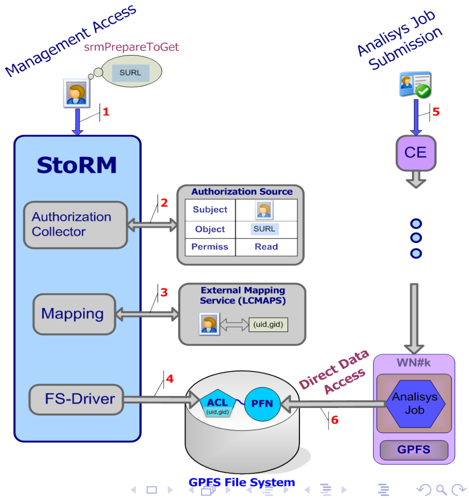
ACL files and directories.

StoRM uses **ACL on file and directories** to enforce authorization decision.
StoRM interacts with the **LCMAPS** service to retrieve the **local uid** and **gid**. Two approach for ACL enforcing:

- **Just in Time (JiT)**: ACL enforced for **local user_id**, removed when the SrmPutDone/SrmReleaseFiles operation take place.
- **Ahead of Time (AoT)**: ACL enforced for **local group_id**. ACL will remain in place until the file or directory exists.

Authorization management operation

1. Verifies if the principal holds a valid proxy certificate.
2. Queries the Authorization Sources.
3. Queries the Mapping Service to obtain the local user account.
4. Enforces permissions by setting a new ACL on the physical file.
5. The application can perform a standard POSIX call to access the file.



Conclusion 1/2

StoRM:

- StoRM is an SRM implementation, other SRM services exist and are used in the HEP context (DPM, dCache, Castor, BeSTMan).
- leverages on cluster and parallel file system advantages in a Grid environment.
- support direct access on data.
- support Storage Area, Token and Description concepts.
- is heavily configurable, to satisfy the different site requirements
- StoRM is used at **CNAF T1** to provide T0D1 storage class.

Conclusion 2/2

Regarding authorization and access permission StoRM provides:

- A layered and pluggable security mechanism.
- Approachable rule, then regular expression in terms of VO and DN to restrict file system access.
- Interaction with external authorization services (as the LFC catalogue).
- Enforcement of physical ACLs on file and directory for the local user identity corresponding to Grid credentials.

The resulting security framework address the strict requirement coming from the financial Grid community and allow the Grid application to perform a secure and efficient direct access to the storage resource.

StoRM

StoRM
Storage Resource Manager

http://storm.forge.cnaf.infn.it

INFN GRID

Antonia Ghiselli
Alberto Forti
Luca Magnoni
Riccardo Zappi

egrid

Ezio Corso