StoRM-GPFS-TSM: a new approach to Hierarchical Storage Management for the LHC experiments

A. Cavalli[1], L. dell'Agnello[1], A. Ghiselli[1], D. Gregori[1], L. Magnoni[1], B. Martelli[1], M. Mazzucato[1], A. Prosperini[1], P. P. Ricci[1], E. Ronchieri[1], V. Sapunenko[1], V. Vagnoni[2], D. Vitlacil[1], R. Zappi[1]

[1] INFN-CNAF, viale Berti-Pichat 6/2, 40127 Bologna, Italy
[2] INFN-Bologna, via Irnerio 46, 40126 Bologna, Italy

Abstract. The mass storage challenge for the experiments at the Large Hadron Collider (LHC) is still nowadays a critical issue for the various Tier-1 computing centres and the Tier-0 centre involved in the custodial and analysis of the data produced by the experiments. In particular, the requirements for the tape mass storage systems are quite strong, amounting to about 15 PB of data produced annually that should be available for near-line access at any time. Besides the solutions already widely employed by the High Energy Physics community so far, INFN-CNAF has approched, in the last year, a solution based on the collaboration between the General Parallel File System (GPFS) and the Tivoli Storage Manager (TSM) by IBM and StoRM as Storage Resource Management (SRM) interface, developed at INFN. The new features available in GPFS version 3.2 allow in general to interface GPFS with any tape storage manager. We implemented such an interface for TSM, and we performed various performance studies on a prototype testbed system. The first StoRM/GPFS/TSM based system is already in production at CNAF for T1D1 Storage Class; it is used by the LHCb experiment. Currently we are performing new tests to exploit features implemented in the new versione of TSM 6.1. These features are focused in high performance and optimized access to the tape backend. We will describe the implementation of the interface and details of the prototype testbed for T1D0 SC and we will discuss the results of the LHCb production system.

1. Introduction

Data management and file access are at the heart of Grid computing, in particular for the High-Energy Physics (HEP) community involved in the activity of the LHC[1] experiments. The LHC experiments have very stringent requirements i.e. several PB/year capacity for storing data produced and the need of steady rate of several Gb/s for data transfers between the computing centers around the world. For these reasons the use of a common set of tools and protocols has been agreed by the involved centers and the experiments forming the LHC Computing Grid (WLCG) [2]. Access to storage resources in the framework of WLCG is based on the so-called Storage Resource Management (SRM) [3] protocol. The main goal of this middleware layer is to hide the details of the underlying storage system, exposing a common interface to applications.

Three different types of so-called Storage Classes [4], i.e. classes of service for storage, have been defined in WLCG:

- Replica-online (also known as Tape0Disk1 or T0D1) - Pure disk space without tape copy. This Storage Class is intended for temporary files or files which have a master copy in one or more other computing centres on WLCG so that in case of file-system loss they can be replicated by copying the files from other sites.

- Custodial-online (also known as Tape1Disk1 or T1D1) - For this storage class we have biunivoc correspondence between files on disk and on tape. It is intended for data files frequently accessed, like the D1T0 case, but in addition the data files are master copies, hence they must have a copy on tape. In case of file-system loss they must be restored from tape. It does not need a complicated active recall policy, since recalls can be just ordered by system administrators when a disaster recovery is performed.
- Custodial-nearline (also known as Tape1Disk0 or T1D0) - In this case the disk is just a temporary staging area. When files are written to disk, they must be immediately migrated to tape for safety. Files can be deleted from the disk buffer when the it is becoming full. In order to access a file previously written, an active recall triggered by the user application is required if the file is no longer on disk.

In the present implementation of Storage Classes at CNAF, besides the CASTOR for the part with tape back-end (T1D0), also GPFS [5] with StoRM for the disk-only system (T0D1) and tape-disk system (T1D1) is used. The choice to split the CNAF storage system into CASTOR and GPFS was driven by the problematic T0D1 access in CASTOR (at least up to 2.1.3-x) and by the upstanding performance of GPFS [6]. Moreover the high performance of a parallel file-systems is becoming increasingly important to fulfill the large I/O throughput required by High-Energy Physics applications.

In the long term a unique solution to implement all the Storage Classes is strongly preferable: in this perspective we have started testing the new HSM features provided by GPFS 3.2 enabling its use as a front-end to a tape system, such as TSM.

Since GPFS does not offer natively a Storage Resource Manager (SRM) interface, which is necessary to access and manage the data volumes in a Grid environment, a specific project called StoRM [7] has been developed for providing it with the necessary SRM capabilities[8]. StoRM is in production at CNAF for ATLAS experiment T0D1 Storage Class since the end of 2007, while to support T1D1 for LHCb some extensions were needed.

In this paper we describe the setup of StoRM/GPFS/TSM and the modifications for StoRM to support T1D1 Storage Class, the layout of the test-bed for a future T1D0 SC implementation.

2. Storage Layout

In this paragraph we describe the different structural parts of our storage system: StoRM as SRM interface, GPFS as disk file-system, TSM for space management (movement from disk to tape and viceversa) and the test-bed layout adopted for performce study.

2.1. StoRM

The StoRM service is a storage resource manager for generic posix disk based storage systems decoupling the data management layer from the underlying storage systems characteristics. It implements the SRM interface version 2.2. StoRM provides a flexible, configurable, scalable and high performance SRM solution. It supports standard Grid access protocols as well as direct access (native POSIX I/O call) on data, fostering the integration of non Grid aware application which provides local access on shared storage.

A modular architecture decouples StoRM logic from the different file-system supported, and plug-in mechanisms allow an easy integration of new file-systems. With this approach, data centres are able to choose the preferred underlying storage system using the same SRM service. StoRM takes advantage from high performance parallel file systems like GPFS (from IBM) and Lustre (from Sun Microsystems) but it is able to manage data on any other standard POSIX file-system. Clustered file-systems allow large numbers of disks attached to multiple storage servers to be configured as a single file-system, providing transparent parallel access to storage devices while maintaining standard UNIX file-system semantics. StoRM is able to leverage the advantages resulting from cluster approach in a Grid environment, enabling data intensive applications to directly access data into the storage resource without interact with any other transfer services.

In the current T1D1 implementation StoRM uses hidden file to indicate file must be copied to tape (pre-migration in TSM jargon). Since T1D0 adds other operations such as recall, pinning, garbage collections, it's clear that a more sophisticate mechanism to manipulate file state in GPFS/TSM is needed. This mechanism should apply for both T1D1 and T1D0 storage classes. A feature already used by GPFS/TSM is the Extended Attributes (EA)[1]: TSM keeps track of file state with a couple of EA. Our choice is to use EAs also to register file attributes (i.e. pinning, space token, storage class destination and so on) and the actions to be taken (e.g. if the file must be copied on tape).

2.2. GPFS

The IBM General Parallel File System (GPFS) is the file-system adopted at INFN-CNAF to store data on disk. It provides a parallel file-system implementation. In general a parallel file-system gives the possibility to have a single file hierarchy (or directory) seen by a set of clients, by aggregating the resources (in particular disks, but also the other resources like computing power and network bandwidth) of several disk-servers, where the data files are striped. The clients have multiple paths to the data providing a redundant and load-balanced system and in the same way this implementation reduces every potential bottleneck as well as increasing the total bandwidth. The clients can also simultaneously access the same files in a concurrent way since the global coherence is somewhat granted by the parallel file-system cluster itself.

Our GPFS installation is based on a Storage Area Network (SAN) infrastructure interconnecting storage systems and disk-servers; while the Farm worker nodes access the storage through standard Ethernet network using the posix file protocol.

Since GPFS is a cluster file-system, with an opportune SAN hardware a true full no single point of failures is possible (disk-servers failures just decrease the theorical bandwidth but the file-system is still avaliable to the clients) and a single "big file-system" for each experiment could be possible, which is strongly preferred by users. Previous tests [6] also showed that the usage of parallel I/O drastically increases and optimizes the disk performance compared to other systems (like CASTOR disk pools). We have been using GPFS in production for more that 4 years at CNAF with all the Farm worker nodes (roughly 1000 clients) accessing the GPFS file-system through the Network Shared Devices (NSD) over the LAN.

2.3. TSM

IBM Tivoli Storage Manager (TSM) [9] is an enterprise-wide storage management application. It provides automated storage management services to workstations, personal computers, and file servers, with a variety of operating systems. Tivoli Storage Manager includes the following components:

- **Server.** The server program provides backup, archive, and space management services to the clients. The Tivoli Storage Manager server uses a database to track information about server storage, client nodes, client data, policy and schedules.
- **Client node.** A client node can be a workstation, a personal computer, a file server, a network-attached storage (NAS) file server, or even another Tivoli Storage Manager server. The client node has IBM Tivoli Storage Manager client software installed.

[1] Extended file Attributes is a file-system feature that enables users to associate computer files with metadata not interpreted by the file-system, whereas regular attributes have a purpose strictly defined by the file-system.

- **Tivoli Storage Manager for Space Management (HSM).** Tivoli Storage Manager for Space Management provides space management services for workstations on some platforms. Tivoli Storage Manager for Space Management automatically migrates files that are less frequently used to server storage, freeing space on the workstation. The migrated files are also called space-managed files. Tivoli Storage Manager for Space Management is also known as the space manager client, or the hierarchical storage management (HSM) client.
- **Storage Agent.** The storage agent is an optional component that may also be installed on a system that is a client node. The storage agent enables LAN-free data movement for client operations and is supported on a number of operating systems.

With TSM-HSM files can be archived automatically to different storage pools. In our use case, the files to be archived on tape are first written to the disk buffer on GPFS. TSM defines the following states for files:

- **Resident.** A file is resident when there is only the copy on disk.
- **Pre-migrated.** When a file is pre-migrated a copy is archived on tape while the original file still resides on disk (GPFS) area.
- **Migrated.** Migration operation performs garbage collecting of pre-migrated files: files are removed from disk and replaced by stub files.

A **stub file** is a pointer to the "real" data file resident on tape. The block size of a stub file is zero under GPFS but it appears (i.e. with the ls command) to have the effective dimension of the original file. Metadata about stub files are stored in the TSM database.

The key component of the system for the GPFS-TSM integration is the GPFS Information Life-cycle Management (ILM)[2] policy engine: according to a set of configurable policies it performs a scan of matadata file-system and migrates data from GPFS to TSM (from disk to the tape library). An example of a policy is given in figure 1.

The policy is applied to the file-system and define RULES and thresholds for GPFS filesets (a GPFS fileset is a set of directories, e.g. /dir1/A or /dir1/A/B). In the current version of GPFS, rules natively allow selecting files according to their name, dimension, access time and creation time, while using an external script is also possible to discriminate by Extended Attributes. In a future verion of GPFS the scan of EA will be natively supported by the policy engine enhancing the performance.

File-system scanning can be parallelized doing it from several servers with a nominal maximum speed, from next released of 100k file per second.

2.4. T1D1 system in production

The first implementation of T1D1 with StoRM/GPFS/TSM [10] dates back to 2008: it has been successfully used in production for LHCb since the May round of CCR08 (the deployement schema is showed in figure 2). For the T1D1 storage class, only pre-migration is performed: the file is copied to tape but not deleted from disk (in accordance with the definition, tape and disk areas have the same size). Pre-migration starts every time ILM file system scan is completed and new files are found, while in a backup system data transfer is performed only at fixed time.

A general configuration for data transfer at INFN Tier1 center consists of some gridftp servers to actually transfer the files, a StoRM end-point as SRM interface, and GPFS servers to manage the disk area. Moreover, in our setup for T1D1 we have two TSM clients (version 5.5) with the

[2] Information life cycle management (ILM) is a comprehensive approach to managing the flow of an information system's data and associated metadata from creation and initial storage to the time when it becomes obsolete and is deleted. Unlike earlier approaches to data storage management, ILM involves all aspects of dealing with data, starting with user practices, rather than just automating storage procedures, as for example, hierarchical storage management (HSM) does.

```
/* Policy implementing T1D1 for LHCb:
-) 1 GPFS storage pool
-) 1 SRM space token: LHCb_M-DST
-) 1 TSM management class
-) 1 TSM storage pool */

/* Placement policy rules */
RULE 'DATA1' SET POOL 'data1' LIMIT (99)
RULE 'DATA2' SET POOL 'data2' LIMIT (99)
RULE 'DEFAULT' SET POOL 'system'

/* We have 1 space token: LHCb_M-DST. Define 1 external pool accordingly. */
RULE EXTERNAL POOL 'TAPE MIGRATION LHCb_M-DST'
EXEC '/var/mmfs/etc/hsmControl' OPTS 'LHCb_M-DST'
/* Exclude from migration hidden directories (e.g. .SpaceMan),
baby files, hidden and weird files. */
RULE 'exclude hidden directories' EXCLUDE WHERE PATH_NAME LIKE '%/.%'
RULE 'exclude hidden file' EXCLUDE WHERE NAME LIKE '.%'
RULE 'exclude empty files' EXCLUDE WHERE FILE_SIZE=0
RULE 'exclude baby files' EXCLUDE
WHERE (CURRENT_TIMESTAMP-MODIFICATION_TIME)<INTERVAL '3' MINUTE
/* Migrate to the external pool according to
space token (i.e. fileset). */

RULE 'migrate from system to tape LHCb_M-DST'
MIGRATE FROM POOL 'system' THRESHOLD(0,100,0)
WEIGHT(CURRENT_TIMESTAMP-ACCESS_TIME)
TO POOL 'TAPE MIGRATION LHCb_M-DST'
FOR FILESET('LHCb_M-DST')

RULE 'migrate from data1 to tape LHCb_M-DST'
MIGRATE FROM POOL 'data1' THRESHOLD(0,100,0)
WEIGHT(CURRENT_TIMESTAMP-ACCESS_TIME)
TO POOL 'TAPE MIGRATION LHCb_M-DST'
FOR FILESET('LHCb_M-DST')

RULE 'migrate from data2 to tape LHCb_M-DST'
MIGRATE FROM POOL 'data2' THRESHOLD(0,100,0)
WEIGHT(CURRENT_TIMESTAMP-ACCESS_TIME)
TO POOL 'TAPE MIGRATION LHCb_M-DST'
FOR FILESET('LHCb_M-DST')
```

Figure 1. GPFS ILM Policy example for LHCb experiment T1D1 Storage Class.

Storage Agent package installed to be able to archive files from disk to tape. The tape drives are accessed over a dedicated tape area network (TAN) via fibre channel. These TSM clients are part of the LHCb GPFS cluster, accessing the disk via LAN. Each TSM client has also direct access to three LTO2 drives in a SUN L5500 tape library. After a first test phase during May 2008 we have put in production this archivial system for the LHCb experiment: 40 TB of data were successfully stored with an average 70 MB/s sustained throughput. Moreover a good result of zerotape migration failures was achieved proving the high reliability of the TSM backend and the related GPFS migration system.

2.5. The Test-Bed for T1D0 study

The test-bed is composed by three TSM client nodes (with a dedicated TSM client v.6.1 beta version installation), named TSM-HSM-1,2,3 in figure 3, four tape drives (9940b technology), one TSM server V.5.5, named TSM-TEST-1 and a disk buffer, under GPFS, of 4.5 TB served by one GPFS server, with two Gbit Ethernet cards in bonding. The GPFS file system, located in a dedicated cluster, is built over a single LUN from an EMC CX3-80 storage subsystem with the standard block size (1MB) used in all our production systems. The GPFS version used is 3.2.1-9.

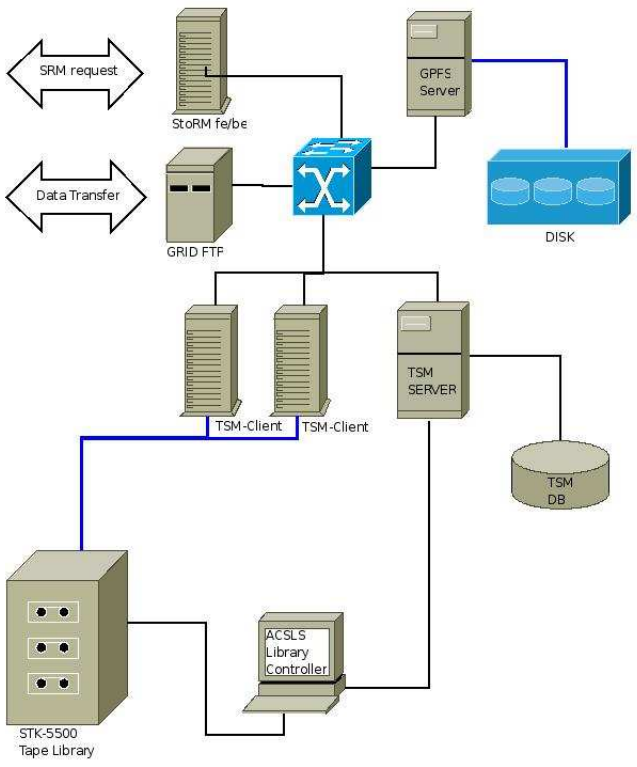

Figure 2. Schematic representation of the T1D1 production system. In black the LAN connections, in blue the Fibre Channel connections via Storage Area Network.

The three TSM clients access GPFS via LAN. Besides these server performing the throughput, there is a TSM server on which the database resides. The TSM database is the heart of TSM server having all metadata stored: for this reason we have also tested the database replication into an external disk. The migration is driven through the ILM policies, used for selecting migrate candidates. The TSM beta version (v. 6.1) used for this test has a new fundamental feature, the so-called "intelligent" recall, i.e. files to be recalled are first reordered by file tape allocation hence optimizing the access time to the tape system. With this feature, all tapes are mounted just once to download all the files requested in the list. In our study we have validated the system for migrations, optimized tape recalls and DB replica and further test of performance and availability are under way.

3. Conclusions

By using the HSM features introduced in GPFS version 3.2 we were able to realize an interface for GPFS-driven automatic migration of files from a GPFS filesystem to a TSM tape pool, hence implementing the so-called T1D1 Storage Class required by WLCG.

The first implementation of this SC is in production since May 2008 (CCRC'08) for the LHCb experiment.

We have also successfully tested the basic functionalities needed to implement the T1D0 Storage Class, namely the recall and the garbage collection, verifying that recalls from TSM to GPFS are transparent to the users and can be reordered to optimize the access to the tape library.

In order to put in production TSM also for the T1D0 Storage Class we are going to perform a large scalability test with concurrent migrations and recalls at nominal rate. Appropriate

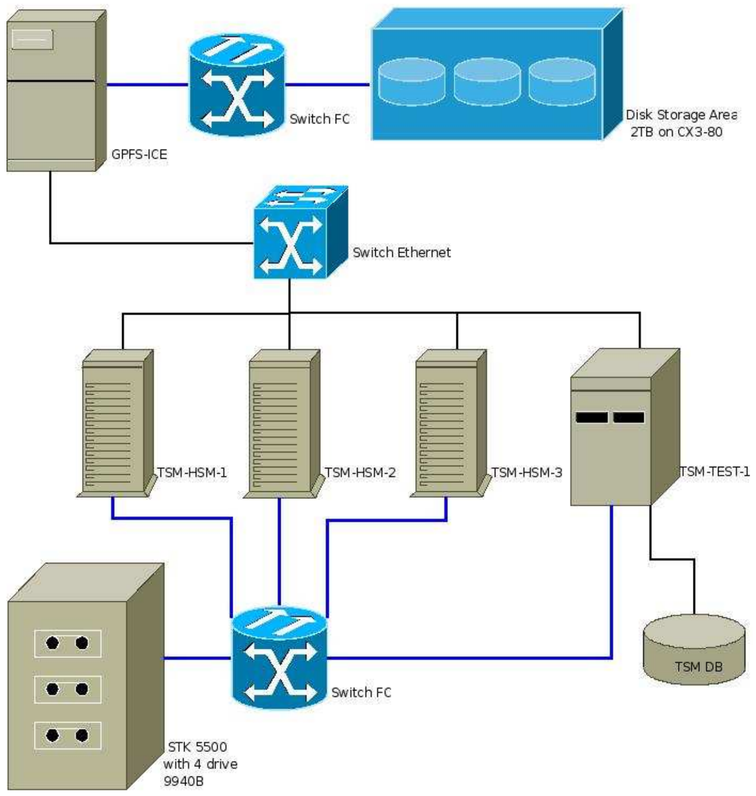

Figure 3. Schematic rapresentation of the Test-Bed adopted for Migration and Recall performance test.

modifications to the SRM layer StoRM are also on going in order to include (or extend) all necessary methods for the T1D0 Storage Class.

References

[1] LHC. Available Online at `http://lhc.web.cern.ch/lhc/`
[2] WLCG. Available Online at `http://lcg.web.cern.ch/LCG/`
[3] F. Donno et al. *The Storage Resource Manager Interface Specification Version 2.2.* Available Online at `http://www.ogf.org/documents/GFD.129.pdf`
[4] F. Donno et al. *Addendum to the SRM v2.2 WLCG Usage Agreement.* Available Online at `https://twiki.cern.ch/twiki/pub/LCG/WLCGCommonComputingReadinessChallenges/WLCG_SRMv22_Memo-14.pdf`
[5] GPFS. Available Online at `http://www-03.ibm.com/systems/clusters/software/gpfs/index.html`
[6] M. Bencivenni et al., *A comparison of Data-Access Platforms for the Computing of Large Hadron Collider Experiments.* IEEE Transactions on Nuclear Science (June 2008) Volume: 55, Issue: 3, Part 3, pp. 1621-1630 ISSN: 0018-9499
[7] E. Corso, S. Cozzini, A. Forti, A. Ghiselli, L. Magnoni, A.Messina, A. Nobile, A. Terpin, V. Vagnoni, and R. Zappi. *StoRM: A SRM Solution on Disk Based Storage System.* Proceedings of the Cracow Grid Workshop 2006 (CGW2006), Cracow, Poland, October 15-18, 2006.
[8] A. Carbone et al., *Performance studies of the StoRM Storage Resource Manager.* Proceedings of Third IEEE International Conference on e-Science and Grid Computing (10-13 Dec. 2007), pp. 423-430 ISBN: 978-0-7695-3064-2 INSPEC Accession Number: 9903980 Digital Object Identifier: 10.1109/E-SCIENCE.2007.59 Current Version Published: 2008-01-07
[9] TSM. Available Online at `http://www-01.ibm.com/software/tivoli/products/storage-mgr/`
[10] A. Carbone et al. *A Novel Approach for Mass Storage Data Custodial.* 2008 IEEE Nuclear Science Symposium Conference Record - N70-4, pp. 3553-3557. Nuclear Science Symposium Conference Record, 2008. NSS apos;08. IEEE Volume, Issue , 19-25 Oct. 2008 Page(s): 3553 - 3557