Storage Resource Managers at Brookhaven

O. Rind[*], Z. Liu, R. Popescu, BNL, Upton, NY 11973, USA
S. O'Hare, Stony Brook University, Stony Brook, NY 11794, USA

Abstract

The introduction of Storage Resource Managers (SRM) was prompted by the need to provide grid applications with effective access to large volumes of data residing on a multitude of disparate storage systems. Their purpose is to provide consistent and efficient wide-area access to storage resources unconstrained by their particular implementation (tape, large disk arrays, dispersed small disks). To assess their viability in the context of the RHIC/US Atlas Tier 1 computing facility at Brookhaven, two implementations of SRM were tested: dCache (FNAL/DESY joint project) and HRM/DRM (LBNL). Both systems include a connection to the local HPSS mass data store providing Grid access to the main tape repository. In addition, dCache offers storage aggregation of dispersed small disks (local drives on computing farm nodes). An overview of our experience with both systems is presented, including details about configuration, performance, interoperability and limitations.

SRM AND ITS ROLE AT THE RCF/ACF

The RHIC Computing Facility (RCF) is a multipurpose center located at the Brookhaven National Laboratory. It was created to provide the computing infrastructure for the experiments at the Relativistic Heavy Ion Collider. In the late 90's, it took on the additional role of providing the Tier 1 infrastructure for the US contingent of the ATLAS experiment at the CERN Large Hadron Collider.

The facility provides general computing services (backup, email, web-hosting, etc.) along with its primary function of support for scientific computing. The main components of the facility include:

- **Mass Storage:** Consisting of 4 StorageTex tape silos managed by HPSS, this subsystem presently hosts over 1500 TB of data. It is currently fronted by a relatively small 10 TB disk cache. Both PFTP and Hierarchical Storage Interface (HSI) [5] transfer protocols are provided for local and remote access.

- **Centralized Disk:** The central disk farm consists of a 220 TB Storage Area Network which is served via NFS by 39 Sun Servers.

- **Linux Farm:** The main computing resource at the facility, the Linux farm comprises 1350 rackmounted servers which are allocated among the experiments. In recent years, the compute servers have been purchased with increasing amounts of local disk, such that the facility currently houses approximately 230 TB in aggregate storage.

Notably, the usable storage on the computing farm itself has now reached a level comparable to the centralized disk servers. This increasing availability of low-cost local disk space has driven a growing interest in distributed storage solutions. At the same time, the advent of grid computing has pushed the need for unified, global access to data located on a diverse set of storage elements, including distributed local storage. Storage Resource Managers (SRM) have been developed to address these storage access needs.

STORAGE RESOURCE MANAGERS

Storage Resource Managers are middleware components that manage shared storage resources on the grid. They provide standardized, uniform access to heterogeneous, distributed storage elements and complement Computing Resource Managers by facilitating the data movement necessary for the scheduling and execution of grid jobs. This is done by providing a number of services including storage reservation, information on file availability, dynamic space allocation, and file management.

Some key features of SRM include:

- Smooth synchronization between storage resources: allocating space on ”as needed” basis, pinning and releasing files;

- Facilitation of file sharing by eliminating unnecessary file transfers: caching, read-ahead;

- Insulation of clients from storage and network system failures;

- Control of the number of concurrent file transfers: throttling to avoid flooding the network or thrashing the Mass Storage System;

- Efficient quota-based storage management allowing long running tasks to process large numbers of files (”streaming model”).

The SRM protocol is being developed and refined by an international working group. More details can be found at [1]. Note that the protocol defines a uniform interface to storage elements without specifying the underlying implementation and, in particular, it does not perform file transfer (though it can invoke other middleware components to do

[*] rind@bnl.gov

so). These specifications are left up to the middleware developers. In the following sections, we discuss our ongoing experience deploying two implementations of SRM at the Brookhaven RCF/ACF.

THE BERKELEY HRM

The Berkeley HRM [2] has been developed by the Scientific Data Management Group at the Lawrence Berkeley National Laboratory. The current implementation provides a Hierarchical Resource Manager (HRM) and client software, plus a recently developed Web Services Gateway (WSG). The WSG, along with a Gateway to Web Services (GWS), provides a translation layer interfacing to the developing SRM protocol. Internal functions use CORBA.

A limited implementation of this software has been in use at the RCF by the STAR experiment for some time. In that configuration, the HRM is used to migrate data between the BNL HPSS system and the NERSC HPSS at LBNL, without employing a Web Services Gateway or the SRM protocol.

BNL Deployment Experience

At the time of this writing, the RCF/ACF has deployed the Berkeley HRM on a single public server with a 350 GB disk cache (recently upgraded) [3]. The interface accepts GSI authorization only and is now available throughout BNL as well as offsite. The client software has been deployed throughout the Linux farm along with the File Monitoring Tool provided by LBNL. The web services gateway is operational and will be made available to the user community once documentation, currently in preparation, is completed.

The Berkeley HRM is compact and easy to deploy, making it suitable for small sites. The installation procedure has been simplified greatly with the advent of a binary release and recent improvements in the documentation. Technical support from LBNL has been strong. Software bugs related to GSI-enabled access to HPSS were resolved at BNL and fed back into the codebase.

Interoperability of the HRM with the dCache SRM has been investigated at BNL. The test used the dCache *srmcp* client to effect a third party transfer from the Berkeley HRM to the dCache SRM. The dCache SRM uses a version of the GLUE platform from The Mind Electric which required access to a WSDL (Web Service Definition Language) file in a specified location on the Berkeley HRM side. This created an incompatibility which was resolved by LBNL support. With this small change, the two SRMs were able to interoperate fully.

Some limitations of the Berkeley HRM still remain:

- Multiple HRMs cannot be used to optimize performance by sharing a single disk cache;
- A single HRM cannot manage multiple file systems on the back-end;
- There is no proxy expiration handling, which is a potential issue while transferring long lists of files.

Some of these limitations should be resolved in future development and, in particular, the proxy issue is being investigated at BNL.

THE DCACHE SRM

The dCache [4] software provides a full-featured distributed storage management package which can serve as a flexible front-end to a mass storage system. It has been developed as a joint venture between DESY and FNAL. Some features of particular interest to the RCF/ACF include:

- A distributed caching front-end to HPSS;
- Availability of multiple file transfer protocols, including SRM, plus POSIX-like I/O and integration with ROOT through the tDcache class;
- Dynamic distributed cache management with load balancing, hotspot handling, and garbage collection;
- A global namespace (PNFS) covering distributed pool elements;
- Java-based portability.

The fact that dCache is already in production use within the community, having demonstrated a workable level of scalability and robustness, has also been an important factor driving the interest at BNL.

In brief, dCache provides a layered, modular architecture in which entry points to the file system are provided by door nodes. The doors are defined by the transfer protocols they provide, e.g. gridftp, SRM, dCap. File transfers are setup by a middleware layer which includes a file catalog (PNFS) and a pool manager. For each transfer, a cost calculation is used to optimize performance by distributing the load among the pools. Transfers occur directly between the disk pool and the client, with automatic file retrieval from mass storage as needed. A structural schematic of this process is shown in figure 1.

BNL Deployment Experience & HPSS Interface Development

BNL has been receiving helpful dCache support from DESY and FNAL since a time prior to the recent official production releases. The installation and configuration process has improved greatly following the advent of newly packaged RPMs over the summer. In particular, the SRM component is well-integrated and installation is straightforward. Most of the deployment issues which arose involved GSI configuration, especially in setting up third party SRM transfers. Currently, the security architecture requires that individual pool nodes participating in such transfers maintain a valid host certificate on at least one end of the transfer.

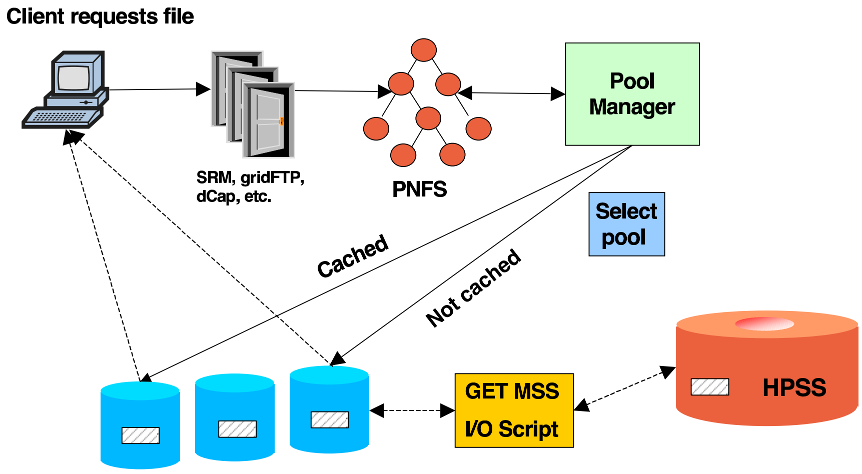

Figure 1: A schematic representation of a file request showing the modular dCache architecture. An interfacing script for access to the HPSS back-end has been added at BNL.

Good system performance was measured during single file, multiple transfer rate tests, up to saturation of the network bandwidth. Pool loads and memory usage present potential issues which are currently being investigated.

One of the main efforts within the context of this deployment is the development of an interface to the HPSS back-end. Within dCache, file destinations are automatically determined by a pool attraction mechanism that makes use of the user-defined PNFS database tag structure. For transfer to mass storage, the dCache software provides a hook for a drop-in script that is called with generic commands (GET, PUT, etc.) and returns a functional set of exit values. Subsequent to a successful PUT — in which the file is backed up to the tape archive — the drop-in script is also used to update the metadata in PNFS. This latter functionality was also developed into a standalone registration utility (hp-register.pl) that can be used to map a previously existing HPSS directory tree into PNFS.

The initial design effort has piggybacked on the available OSM interface using HSI as the transfer mechanism. This system, however, does not provide for throttling or aggregation of GET requests to HPSS. Since this is a potential problem, a plan is being implemented to replace HSI with a queueing system acting as a tape access optimizer.

In contrast to the FNAL deployment using an Enstore back-end [6], the internal database used by HPSS is independent of the PNFS database used by dCache. Depending on the designed implementation, the maintenance of file catalog consistency between dCache and HPSS is an issue that must be addressed. The idea of file addressing using unique HPSS bitfile IDs was considered, but there is no feasible API available for this. Thus, two scenarios have been considered:

1. Allow dCache files *within HPSS* to be owned by various users, but make them accessible to a unique and privileged dCache user;
2. Require dCache files *within HPSS* to be owned by a special dCache user while maintaining normal user ownership through PNFS.

The first scenario is the least restrictive in the sense that users can retain the ability to access their files through alternate methods. The drawback is that this requires intervention to maintain the file locations in the PNFS database by relying on a responsible user (such as a production manager) and/or automated, periodic consistency checks (which have the potential of adversely loading the mass storage system). In an effort to avoid this potentially cumbersome infrastructure, the current plan is to move toward the second scenario as the adoption of dCache increases. That scenario offers automatic maintenance of file catalog consistency, although it may be less flexible for the user and involves a greater initial setup effort as ownership of the existing data store must be transferred to dCache.

CONCLUSIONS

The Berkeley HRM and dCache/SRM are both in the process of being deployed at the RCF/ACF in BNL. Future plans for the HRM include efforts to extend the deployment to other US ATLAS sites, to integrate with Replica Location Services (RLS) and Grid Monitoring service, and to encourage further user adoption. Interoperability testing will also continue. For dCache/SRM, performance testing will continue on an increasing scale as the system is brought online for limited use by the experimenters. A major goal will be to evaluate the feasibility of using dCache as a distributed storage solution on dual use (pool/analysis) farm nodes.

ACKNOWLEDGEMENTS

We would like to thank the HRM support group from LBNL, the dCache support teams from DESY and FNAL, and Jerome Lauret from the STAR experiment for their invaluable assistance.

REFERENCES

[1] http://sdm.lbl.gov/srm-wg/

[2] http://sdm.lbl.gov/projectindividual.php?ProjectID=SRM

[3] http://www.atlasgrid.bnl.gov/srm/manuals/

[4] http://www.dcache.org

[5] http://www.sdsc.edu/Storage/hsi/

[6] http://www-isd.fnal.gov/enstore/