

Fermilab's Multi-Petabyte Scalable Mass Storage System

Gene Oleynik, Bonnie Alcorn, Wayne Baisley, Jon Bakken, David Berg, Eileen Berman, Chih-Hao Huang, Terry Jones, Robert D. Kennedy, Alexander Kulyavtsev, Alexander Moibenko, Timur Perelmutov, Don Petravick, Vladimir Podstavkov, George Szmuksta, Michael Zalokar

Fermi National Accelerator Laboratory

{oleynik, alcorn, baisley, bakken, berg, berman, huangch, jonest, kennedy, aik, moibenko, timur, petravick, podstvkv, georges, zalokar}@fnal.gov

Abstract

Fermilab provides a multi-Petabyte scale mass storage system for High Energy Physics (HEP) Experiments and other scientific endeavors. We describe the scalability aspects of the hardware and software architecture that were designed into the Mass Storage System to permit us to scale to multiple petabytes of storage capacity, manage tens of terabytes per day in data transfers, support hundreds of users, and maintain data integrity. We discuss in detail how we scale the system over time to meet the ever-increasing needs of the scientific community, and relate our experiences with many of the technical and economic issues related to scaling the system. Since the 2003 MSST conference, the experiments at Fermilab have generated more than 1.9 PB of additional data. We present results on how this system has scaled and performed for the Fermilab CDF and D0 Run II experiments as well as other HEP experiments and scientific endeavors.

1. Introduction

Fermilab has developed a mass storage architecture and software suitable for the demands of high-capacity and high-throughput scientific endeavors, specifically for Run II of the CDF and D0 experiments that are currently in progress at the Fermilab Tevatron collider. These experiments record their data 24x7 to the Fermilab mass storage system, perform analysis of this data, and write the results back into the system all in real-time. In this paper, we will focus on the scalability aspects of the mass storage system and its performance.

The Fermilab mass storage architecture [1] consists of a data storage system, called "Enstore" that provides access to any number of automated tape libraries, and a disk caching front end to this, called "dCache", which permits transparent pre-staging of files from the libraries and efficient access to files by multiple users.

Enstore [2] is the mass storage system implemented at Fermilab as the primary data store for experiments' large data sets. Enstore provides distributed access to data on tape to both local and remote users. Enstore is designed to provide a high degree of fault tolerance and availability as well as easy administration and monitoring. It uses a client-server architecture that provides a generic interface for users and allows for hardware and software components that can be replaced and/or expanded dynamically.

File read/write requests to the mass storage system can also go through dCache [3], a disk caching system that uses Enstore as a permanent store. DCache decouples the (potentially slow) network transfer from the (fast) storage media I/O in order to keep the Enstore system from bogging down. Data exchanges between the dCache and Enstore are performed automatically and are transparent to the users.

The dCache project is a joint DESY¹-Fermilab effort to overcome the accessibility limitations posed by the types of mass storage software and devices found at HEP labs. The dCache optimizes the location of staged copies and makes more efficient use of expensive tape drives and automated tape libraries. In addition, the dCache provides a uniform view of the storage repository, hiding the physical location of the file data (disk-cached or tape-only). The dCache provides several interfaces for off-site grid-based access to data in the Fermilab mass storage system: ftp, GridFTP, SRM and the dCache "dcep" interface for on-site access.

Mass storage is provided through a number of automated tape libraries and a disk cache. Currently, Fermilab has six Storage Tek 9310 libraries and one ADIC AML-2 library with a total potential capacity of around 8PB. These libraries are configured with over 100

¹ DESY – Deutsches Elektronen Synchrotron HEP research facility in Hamburg Germany

tape drives consisting of 9940, 9940B, LTO, LTO2, and DLT devices. At the moment 180 TB of disk cache is provided, mainly for the CDF experiment. These libraries and drives are divided into three logical mass storage systems: CDF, D0, and CMS/General Use. The CDF system has two 9310 libraries, the D0 system two 9310 and the AML-2, and the General/CMS system two 9310 and one of the AML-2 quadratowers.

2. Architecture and scalability

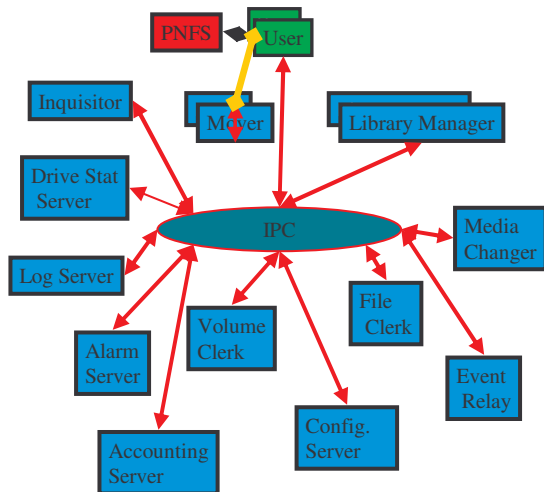


Figure 1. Enstore Architecture

The number of user computers is not restricted by Enstore or dCache, and enstore's components can be distributed over an unlimited number of computers, tape libraries and tape drives. Enstore and dCache are scaled by building the equipment out. Two things are required to make this possible: the decoupling of the user interface from the underlying technologies and a scalable design of the software architecture. Software has not been a factor in scaling the system to higher capacity, data rates, and users, with one exception in dCache which is discussed in the section titled 'DCache Scalability'.

Hardware consists of vendor supplied tape libraries and drives. The enstore and dCache systems run on commodity PCs using the Linux operating system. Critical disks (servers with databases or dcache nodes) are hardware RAID-5 or RAID-10. Enstore and dCache PCs are connected by Gigabit Ethernet to each other and typically to end-user machines. The systems span across three floors at Fermilab's Feynman Computing Center.

Enstore and dCache provide file based interfaces to end-users. Enstore users access their data through the encp interface which deals only with files and is independent of

the underlying storage technologies. The underlying storage system can therefore be expanded and new storage technologies can be introduced without affecting the end user. Similarly, dCache users access their data through the dccp interface or the ftp, GridFTP, or SRM interfaces, all of which only expose the file interface and thus are independent of the storage system.

The Enstore software architecture is presented in Figure 1. Enstore is designed using a client-server architecture and provides a generic interface for users. Enstore components are:

- A configuration server keeps the system configuration information and provides it to the rest of the system. Configuration is described in an easily maintainable configuration file.
- A volume clerk maintains the volume database and is responsible for declaration of new volumes, assignments of volumes, user quotas, and volume bookkeeping.
- A file clerk maintains the file database, assigns unique bit file IDs and keeps all necessary information about files written into Enstore.
- Multiple distributed library managers provide queuing, optimization, and distribution of user requests to assigned movers.
- Movers write / read user data to tapes. A mover can be assigned to more than one library manager.
- A media changer mounts / dismounts tapes in the tape drives at the request of a mover.
- Alarm and log servers generate alarms and log messages from Enstore components correspondingly.
- An accounting server maintains an accounting database containing information about completed and failed transfers and mounts.
- A drivestat server maintains a database with information about tape drives and their usage.
- An inquisitor monitors the state of the Enstore components.
- PNFS namespace server implements a name space that externally looks like a set of Network File Systems.
- Events are used in the Enstore system to inform its components about changes in the configuration, completed and ongoing transfers, states of the servers, etc. An event relay transmits these events to its subscribers.

All Enstore components communicate using inter-process communications based on UDP. Great care has

been taken to provide reliable communications under extreme load conditions. The user command, `enpc`, retries in case of an internal Enstore error.

Enstore supports automated and manual storage libraries, which allows for a larger number of tapes than slots in the robotic storage. The user command interface program `enpc` has a syntax similar to the Unix `cp` command with some additional options, allowing users to specify request processing parameters such as priority, and number of retries, whether to calculate a CRC, and to set a tape dismount time, etc.

Data stored in Enstore are grouped based on the storage group unique to each experiment, and file families inside of the storage group. The storage group is assigned by the storage system administrator while the file family is assigned by the user. Files in the same file family are written to the same set of tapes moderated by a file family width. The file family width controls the amount of simultaneous write transfers for a certain file family.

Enstore allows users to specify the priorities for data transfer. There are 2 kinds of priorities: regular and administrative or Data Acquisition (DAQ) Priority. The library manager will dispatch DAQ priority requests ahead of any regular priority requests. In this case the mover may dismount the currently mounted tape and mount the DAQ one. Priorities can be assigned to requests according to configuration parameters in the Enstore configuration file.

Another important feature of Enstore is its capability to specify the number of tape drives dedicated to an experiment (storage group). Experiments have separate budgets. Some of them may have their own tape drives installed in the general purpose robotic library. These drives are in the common pool but the experiment will preferentially be given access to the number of drives equivalent to its contribution. This amount is normally statically specified in the configuration file by the system administrators, and is used while processing the request queue.

DCache consists of

- Administrative Nodes which setup user requests to the read and write pool nodes
- Monitoring Node which monitors the other dCache nodes
- Namespace (PNFS) Server
- Read pool nodes from which files can be spooled to the user from Enstore

- Write pool nodes from which data can be written to Enstore

Of particular concern for scalability are the Movers, Library Managers, Media Changers, Namespace servers, and dCache Administrative and pool nodes. In general, all of these servers scale by replication. Their scalability is discussed below.

2.1. Movers and Media Changers

The knowledge of the underlying storage technologies is isolated to the media changers and the movers. The movers move data between the user and libraries and communicate with the media changers which instruct the libraries to fetch and store tapes as needed.

Movers transfer data between tape drives and the end-user over high speed network connections. Movers isolate the knowledge of the underlying tape technology and generally communicate with drives through a SCSI interface. In Enstore, 1 or 2 tape drives are managed by a single mover node which is a PC running Linux. System throughput is increased by adding tape drives and Mover nodes to perform the transfer function.

Media changers need to accommodate the number of user mount and dismount requests. Since automated tape libraries typically are limited to hundreds of exchanges per hour, this component of the system does not need to scale, but does provide the important function of isolating knowledge of the interfaces to the automated tape libraries.

2.2. Namespace and file/volume clerks

Enstore and dCache use a file system namespace called PNFS [4]. The users access files through PNFS names. A PNFS database maintains meta-data about these files names, and this information, in conjunction with file and volume databases, can be used to identify the location and size of a file in the mass storage system given its name. PNFS does not maintain the actual data, but rather the meta-data for Enstore and dCache to find the actual data.

The PNFS database is accessible using the NFS protocol. This means that users access PNFS via NFS file systems mounted on their nodes. A separate node is usually used to run a PNFS server. PostgreSQL [5] is used as the database management system for the name space. The size of the database can be up to 2^{64} bytes, therefore PNFS scalability is only limited by the number and the rate of the PNFS requests. To cope with the scaling of name space request load, we split the namespace and set

up additional PNFS server machines as needed. This scales just as an NFS file system would scale.

Scalability of the file and volume database is limited only by the scalability of the PostgreSQL DBMS in Enstore. The goal of the scalability in this case is to avoid overloading the system with requests. Scalability is achieved by adding databases and corresponding Enstore servers along with machines on which these servers run.

2.3. Library Managers

Library managers are responsible for processing user data transfer requests and therefore need to scale with the number of user requests for data. They can process thousands of queued requests and so far have not reached a saturation point, but new library managers can be added as needed.

2.4. DCache scalability

From the file access perspective, dCache is scaled by adding read or write pool nodes. DCache scales in simultaneous access by automatically duplicating frequently accessed files across pool nodes to spread the load.

The Administrative dCache nodes setup user requests to the pools. DCache can be scaled up to handle high rates of user requests by adding Administrative nodes.

The only software limitation to scalability encountered so far has been with PNFS on very large and highly utilized dCache systems due to the overhead of accessing the namespace server over the network for every file access. A new version of dCache removes this bottleneck by allowing local namespace resolution.

3. File and Metadata Integrity

At the current time, the Fermilab mass storage system contains on the order of 25000 tapes and monitors and maintains meta-data on these volumes and the files they contain. On this and ever growing scales, maintaining and monitoring the integrity of the files and meta-data is a great concern.

3.1. Maintaining File Integrity

The Enstore system takes the following steps to monitor and maintain the integrity of the files on the tape and in the data cache:

1. Extensive CRC checking at all stages of getting the files on and off tape and in and out of dCache.
2. A flexible policy for read after write CRC checking for tapes
3. Automated read checking of the CRC of randomly selected files on tapes
4. Automated CRC checking of dCache files
5. Automatically disabling access to tapes or tape drives that exhibit read or write problems
6. Cloning overused or problematic tapes

CRC checks are performed on the file transfers at all stages. A CRC is generated when the data is sent over the network to Enstore to be written on tape, and recalculated and compared when it is received by enstore. The original CRC is maintained in the metadata and is recalculated and compared at every stage of file movement, including in and out of dCache pool nodes. When a file is read from tape or a dCache pool, its CRC is calculated and checked against the one kept in the metadata, and the CRC is again checked when transferred to the user. If a dcache file with a bad CRC is alarmed, it is usually removed by an administrator. – future references will then get a fresh copy from tape.

The movers can be configured to read the first file written on a volume after it was mounted or not, and/or to check randomly selected files at a configurable frequency. For example, one experiment is using both approaches.

The Enstore system also automatically randomly selects files, reads them, calculates their CRC, and compares them to the files' metadata CRC. dCache also periodically checks the CRC of files that it has stored. This is the only part of data integrity checking that may be affected by the system scale. As the system grows, the frequency of these random checks can be increased.

CRC failures are alarmed and monitored by the administrative staff.

Enstore identifies two classes of errors: recoverable and unrecoverable. If enstore encounters an error executing some action, and if the error is identified as recoverable, it will retry the action until it succeeds or a configurable limit in retries is reached, in which case it reports the error as unrecoverable. Unrecoverable errors related to tape operations result in the tape being placed into the NOACCESS state. In addition, any time a write error occurs when writing to a tape, the tape is placed in the "READ-ONLY" state. If a configurable number of write errors or read errors occur in a row on the same tape

