

# Chapter 1: Introduction to the Enstore Mass Storage System

## 1.1 About Enstore

---

Enstore is the mass storage system implemented at Fermilab as the primary data store for large data sets. Its design was inspired by the mass storage architecture at DESY, and it originates from discussions with the DESY designers. Enstore provides access to data on tape or other storage media both local to a user's machine and over networks. Enstore is designed to provide high fault tolerance and availability sufficient for the RunII data acquisition needs, as well as easy administration and monitoring. It uses a client-server architecture which provides a generic interface for users and allows for hardware and software components that can be replaced and/or expanded.

Enstore has four major kinds of software components:

- namespace (implemented by PNFS), which presents the storage media library contents as though the data files existed in a hierarchical UNIX file system
- dCache, a front-end data file caching system through which users typically access Enstore
- the Enstore servers, which are software modules that have specific functions, e.g., maintain database of data files, maintain database of storage volumes, maintain configuration, look for error conditions and sound alarms, communicate user requests down the chain to the tape robots, and so on
- **encp**, a program for copying files directly to and from the mass storage system

Enstore can be used from machines both on and off-site. Off-site users are restricted to accessing Enstore via dCache, and on-site users are encouraged to go through dCache as well.

Enstore supports both automated and manual storage media libraries. It allows for a larger number of storage volumes than slots. It also allows for simultaneous access to multiple volumes through automated media libraries. There is no preset upper limit to the size of a data file in the enstore system; the actual size is limited by the physical resources. The lower limit on the file size is zero. The upper limit on the number of files that can be stored on a single volume is about 5000.

Enstore allows users to search and list contents of media volumes as easily as they search native file systems. The stored files appear to the user as though they exist in a mounted UNIX directory. The mounted directory is actually a distributed virtual file system in PNFS namespace containing metadata for each stored file. Enstore eliminates the need to know volume names or other details about the actual file storage.

Users typically access Enstore via the dCache caching system. The protocols supported by dCache include dcpp, gridftp (globus-url-copy), kerberized ftp and weakly-authenticated ftp (these are described in Chapter 4: *Using the dCache to Copy Files to/from Enstore*). On-site users may bypass dCache and use the **encp** program, the Enstore copy command roughly modeled on UNIX's **cp**, to copy files directly to and from storage media.

There are several installed Enstore systems at Fermilab. Currently these include CDFEN for CDF RunII, D0EN for D0 RunII, and STKEN for all other Fermilab users. Web-based monitoring for the Enstore systems is available at <http://hppc.fnal.gov/enstore/>. Currently, all storage libraries are tape libraries. The Computing Division operates and maintains the tape robots, slots, and other tape equipment, but for the present, experiments provide and manage their own volumes.

## 1.2 PNFS Namespace

---

PNFS is a virtual file system package that implements the Enstore namespace. It was written at DESY. PNFS is mounted like NFS, but it is a virtual file system only. It maintains file grouping and structure information via a set of tags in each directory. The **encp** program communicates this information between PNFS and the Enstore servers when it uploads or downloads a data file.

PNFS can only be mounted on machines that are physically at the lab. When a user copies a data file from disk to the Enstore system, he or she specifies its destination in terms of a PNFS directory. The data file gets copied to a storage volume (selected according to the tags of the specified PNFS directory) and a corresponding metadata entry is created in the PNFS directory. This entry takes the name given in the **encp** command line or in the protocol-specific dCache command. It contains metadata about the data file, including information about the file transfer, the data storage volume on which the data file resides, the file's location on the volume, and so on.

To browse file entries in the Enstore system, on-site users can mount their experiment's PNFS storage area on their own computers, and interact with it using standard non-I/O UNIX operating system utilities (see section 3.1 *UNIX*

*Commands You can Use in PNFS Space*). Normal UNIX permissions and administered export points are used for preventing unauthorized access to the name space.

## 1.3 Storage Groups

---

Each experiment or research project is assigned a unique *storage group* identifier by the Enstore administrators. Enstore uses the storage group names to control and balance assignment of resources, such as tape drives and media, among the experiments. Each storage group is assigned an area in PNFS, e.g., an experiment XYZ might be assigned the storage area `/pnfs/xyz`.

## 1.4 File Organization on Storage Media

---

### 1.4.1 File Family

Files are grouped on data storage volumes according to a *file family*<sup>1</sup> attribute. A file family is a name that defines a category, or family, of data files. Each experiment (i.e., each storage group) must carefully plan its set of file families. There may be many file families configured; by design there is no pre-set upper limit on the number. A given storage volume may only contain files belonging to one file family.

Every directory in `/pnfs` namespace has a file family associated with it. Every data file added to the Enstore system (i.e., every file for which an entry appears in the `/pnfs` namespace) is thus associated with the file family of the directory under `/pnfs` into which it was initially copied.

Associated with a file family are a *file family width* (an integer value), and a *file family wrapper* (a format specification). The file family, file family width, and file family wrapper for a PNFS directory are initially inherited from the parent directory. They may be reset as permissions allow, generally only by a small group of designated people in each experiment.

---

1. The grouping is really based on the triplet of quantities: storage group + file family + wrapper. However, most users have access to only one storage group, and use the default wrapper, so from the user's perspective, the only relevant attribute for file grouping is file family, typically.

## 1.4.2 File Family Width

File family width is an integer value associated with a file family that is used to limit write-accessibility on data storage volumes. There is currently no width associated with reading. For a given media type and for a given file family, Enstore limits the number of volumes available for writing at any given time to the value of the file family width (except when unfilled volumes are already mounted for previous reads). Correspondingly, the number of media drives on which the volumes are loaded is also limited to the width.

## 1.4.3 File Family Wrapper

A file family wrapper specifies the format of files on the storage volume. It defines information that gets added before and after data files as they're written to media. In this way the data written to tape is self-contained and independent of metadata stored externally.

There are two wrapper types implemented, `cpio_odc` and `cern`. Currently (November 2003) all tapes are written using the `cpio_odc` wrapper. (For NULL volumes there is a null wrapper.)

- All files with the `cpio_odc` wrapper are dumpable via `cpio`. This wrapper has a file length limit of  $(8G - 1)$  bytes. It is sufficient for the vast majority of data files, as most files are still under 2GB.
- The `cern` wrapper accommodates data files up to  $(2^{64} - 1)$  bytes, which in effect limits the filesize to the tape length, since spanning and striping are not supported as explained in section 1.6 *Data Storage Volumes in Enstore*. It matches an extension to the ANSI standard, as proposed by CERN, and allows data files written at Fermilab to be readable by CERN, and vice-versa<sup>1</sup>.

## 1.5 File Size Limitations

---

Enstore limits the size of a data file to the tape capacity, i.e., a single file cannot span more than one volume. The `cpio_odc` wrapper further limits the file size to  $(8G - 1)$  bytes, as mentioned in section 1.4.3 *File Family Wrapper*. Your OS may restrict your files to yet a smaller size.



Note: PNFS can only represent a data file's size accurately up to  $(2G-1)B$ ; beyond that, the file size is shown as 1. Enstore knows, stores and uses the real file size, so this PNFS display limitation does not pose a functionality problem.

---

1. Since CERN will allow files to span volumes, and Fermilab doesn't, users will not be able to use Enstore to read volumes from the CERN system that contain partial files.

A 1 byte file size indicated by PNFS indicates to the the user that the file is likely quite large. (You can use the `enstore pnfs` command with the `--filesize` option as described under section 8.4 *enstore pnfs* to find the actual file size.)

## 1.6 Data Storage Volumes in Enstore

---

Enstore is designed to support a variety of data storage media. Currently, only tapes have been implemented, and the information in this section has been written with tapes in mind. In principle, the information should apply equally to other media types; we will update this section as necessary when other media types are implemented.

### 1.6.1 Tape Features

Tapes are self-describing and exportable so that in the unlikely event of lost metadata in Enstore, a volume can be dumped, and the information retrieved.

Tapes are required to have ANSI Vol1 header labels. Labelling helps to easily identify each volume and/or test for a blank one, thereby inhibiting the inadvertent overwriting of used tapes. The Enstore admin needs to know whether tapes are labelled or completely blank when they are inserted into the library; the Enstore software can label tapes if necessary.

### 1.6.2 File Organization, Storage and Access

Data files are physically clustered on volumes according to each experiment's file family classification scheme. A given volume may only contain files belonging to one file family, one wrapper type, and one storage group.

A single file cannot span more than one volume<sup>1</sup>. When writing files to media, Enstore compares the size of the file it's ready to copy against the volume's remaining empty space in order to determine whether the file will fit. If the file is too large to fit, Enstore marks the volume as full, and writes the file to a different volume. Thus volumes are filled, modulo some fraction of a data file. (Volumes can be reopened if an administrator decides that too much space is left unused.)

Enstore supports random access of files on storage volumes, and also *streaming*, the sequential access of adjacent files.

---

1. Since files cannot span multiple volumes, striping is not supported either. Striping refers to files (usually large ones) being split onto two or more volumes, each writing simultaneously, in order to expedite the writing process.

### 1.6.3 Quantity of Volumes

Enstore allows data storage volumes to be “faulted out to shelf”, i.e., removed from a robot. This feature makes it possible for each experiment to have a larger number of volumes than it has slots in the robot, and in fact there is no limit on the number of volumes used by an experiment. To accommodate the unmatched numbers of volumes and slots, Enstore provides separate quotas in volumes and in slots.

Note that moving volumes in and out of the robot requires operator intervention, and should be minimized.

### 1.6.4 Import/Export of Volumes

Tapes can be generated outside the Enstore system and imported into it. Conversely, Enstore tapes can be dumped via standard UNIX utilities, thereby allowing them to be readable with simple tools outside the Enstore framework. The tools to do this are wrapper-dependent (e.g., tapes whose files have the cpio wrapper can be dumped via the `cpio` utility). Currently there is no utility (except `dd`) to dump a tape whose file family wrapper is cern.

## 1.7 dCache: The Enstore Front-End

---

### 1.7.1 Overview

The dCache has been designed as a front-end for a set of Hierarchical Storage Managers (HSMs), namely Enstore, EuroGate and DESY’s OSM. It can be viewed as an intermediate “relay station” between client applications and the HSM (Enstore, in our case). Client systems communicate with dCache via any of a number of protocols, listed in 1.7.3 *Protocols for Communicating with dCache*. dCache communicates with Enstore (in a manner transparent to the user) via a high-speed ethernet connection. The dCache decouples the potentially slow network transfer (to and from client machines) from the fast storage media I/O in order to keep Enstore from bogging down.

Data files uploaded to the dCache from a user’s machine are stored on highly reliable RAID disks pending transfer to Enstore. Files already written to storage media that get downloaded to the dCache from Enstore are stored on ordinary disks.

The dCache is installed at Fermilab on a server machine on which the `/pnfs` root area is mounted. Since PNFS namespace can only be mounted on machines in the `fnal.gov` domain,<sup>1</sup> off-site users may only access Enstore via

the dCache. On-site users are strongly encouraged to go through the dCache as well. We discuss dCache in more depth in Chapter 4: *Using the dCache to Copy Files to/from Enstore*.

Read more general information about the dCache at the DESY site:  
<http://www-dcache.desy.de>.

## 1.7.2 Advantages

The principal advantages of using the dCache are:

- Optimized usage of existing tape drives due to transfer rate adaption.
- Possible usage of slower and cheaper drive technology without overall performance reduction.
- Optimized usage of the robot systems by coordinated read and write requests.
- Better usage of network bandwidth by exploring the best location for the data.
- No explicit staging required to access the data.
- Ability to do posix-like IO reads and writes to data files instead of transferring entire files.
- Working ROOT interfaces.
- Tapeless data methods, raw data to reconstruction to analysis to users.
- Written to tape as 'by-product'; no tape delays.
- Pnfs does not have to be mounted for access to the data.
- Same access to storage system, on and off site. Strong authentication, both gss and gsi to the data. Native and ftp access to the data.
- The access methods for data would be uniform, independent of data's media location.
- Even without the back-end HSM (e.g., Enstore), the dCache system could be seen as a huge data store with a unique namespace and standardized access methods. Care will be taken that valuable data resides on safe disks as long as no HSM copy exists. Back-end storage to the HSM can be done regularly (policy based) or by manual intervention only.
- A joint DESY-FNAL effort makes the use of manpower more efficient and guarantees continued support and maintenance of the developed software.

---

1. There are some exceptions; arrangements for PNFS mounting have been made for some experiments whose systems are managed by the Computing Division, e.g., soudan.org for Minos.

### 1.7.3 Protocols for Communicating with dCache

Whenever an application needs to talk to the dCache, it has to choose an appropriate *door* into the system. There are a number of different dCache doors through which users/applications can send requests to Enstore. Doors are protocol converters from the dCache point of view, and they are responsible for strong authentication, as necessary. One door may be for Kerberized ftp read/write access, another for dcap (dCache native C API), gridftp, weakly authenticated ftp read-only access, and so on. Each experiment determines which door(s) its experimenters may use, and communicates this information to the Enstore administrators who manage the doors' configurations. Most doors are for native transfers, and are local. See Chapter 4: *Using the dCache to Copy Files to/from Enstore* for more information.