

The Enstore User's Guide

10/7/14

Stuart Fuess
Katherine Lato
Dmitry O. Litvintsev
Alexander N. Moibenko
Gene Oleynek

Abstract

Enstore is the mass storage system developed and implemented at Fermilab as the primary data store for scientific data sets. This document covers the basics of data storage including how users get ready, and how they can organize their data storage and copy files. It includes information about Enstore Servers, commands, monitoring and job priority and queue management.

Table of Contents

Chapter 1: Introduction to the Mass Storage System	4
1.1 About Enstore	4
1.2 Chimera Namespace	4
1.3 dCache	5
Chapter 2: Data Storage in Enstore	5
2.1 Storage Groups	5
2.2 File Organization on Storage Media	5
2.2.1 File Family	5
2.2.2 File Family Width	6
2.2.3 File Family Wrapper	6
2.3 File Size Limitations	7
2.4 Data Storage Volumes in Enstore	7
2.4.1 Tape Features	7
2.4.2 Tape and Library (Robot) Lifetimes	8
2.4.3 File Organization, Storage and Access	8
2.4.4 Quantity of Volumes	8
2.4.5 Import/Export of Volumes	9
Chapter 3: Getting Ready to Use Enstore and dCache	9
3.1 Initial Steps for All Users	9
3.2 Further Steps for Direct Access Enstore Users Only	10
3.3 Installing Encp	11
Chapter 4: Copying Files with Encp	11
4.1 Setup encp	11
4.2 Encp Command Syntax and Usage	12
4.3 Copy Files to and from Enstore Media	12
4.3.1 Run encp	12
4.3.2 Examples	13
4.3.3 Preventing Unwanted Overwriting	14
4.3.4 Killing an encp Process	14
4.3.5 Encp Transfer Rates Defined	14
4.3.6 Isolating Source of Bottlenecks	15
4.3.7 Encp Error Handling	16
4.3.8 Finding files in different Enstore Systems	16
4.3.9 Order of Processing Queued Requests	17
4.3.10 NULL File Directories	17
4.3.11 Important Environment Variables	17
4.3.12 Encp Limitations	18
4.4 Encp Command Options	18
Chapter 5: Copying Directory Structures with Ensync	21
5.1 About Ensync	21
5.2 Ensync Command Syntax	21
Chapter 6: Overview of the Enstore Servers	23
6.1 File Clerk	23

6.2 Volume Clerk.....	24
6.3 Library Manager.....	24
6.4 Mover.....	24
6.5 Media Changer.....	25
6.6 Configuration Server.....	25
6.7 Inquisitor.....	25
6.8 Alarm Server.....	25
6.9 Log Server.....	25
6.10 Event Relay.....	25
6.11 Monitor Server.....	26
6.12 Accounting Server.....	26
6.13 Drivestat Server.....	26
6.14 Info Server.....	27
6.15 Ratekeeper.....	27
Chapter 7: Enstore Commands	27
Chapter 8: Monitoring Enstore on the Web	27
Chapter 9: Job Priority and Queue Management	29
9.1 Job Priority Categories.....	29
9.2 Numerical Priority Values.....	29
9.3 Fair Share Resource Allotment.....	29
9.4 Resource Ownership.....	30

Chapter 1: Introduction to the Mass Storage System

The mass storage system at Fermilab has three major components:

- Enstore, the principal distributed scalable multi-petabyte mass storage component; Enstore provides storage and access to data on tape or other storage media both local to a user's machine and over networks.
- A namespace (implemented by Chimera), which presents the storage media library contents as though the data files existed in a hierarchical UNIX file system.
- dCache, a data file caching system; dCache is a distributed, multi-petabyte scalable disk storage system with a single rooted filesystem providing location independent file access. dCache can be used stand-alone or configured as a disk cache front-end to a tertiary hierarchical storage management (HSM) systems (e.g., hard disk and tape) for I/O optimization.

1.1 About Enstore

Enstore is the mass storage system developed and implemented at Fermilab as the primary data store for scientific data sets. It provides access to data on tape to/from a user's machine on-site over the local area network, or over the wide area network through the dCache disk caching system. Enstore was designed to provide high fault tolerance and availability, originally for the Run II data acquisition and analysis needs. It was extended to the CMS Tier One facility, Intensity Frontier experiments, and an assortment of other scientific endeavors. It uses a client-server architecture, provides a simple interface for users and allows for hardware and software components to be replaced and/or expanded to meet needs.

Enstore can be used independently or in combination with caching systems such as dCache or SAM. When used with caching/buffering system, files get written to disks and then migrated to Enstore tapes. For file read requests, if the files do not reside in the disk cache, they first get retrieved from Enstore. Direct access to Enstore is limited to on-site machines - dCache is required for off-site access.

Enstore systems at Fermilab include:

- [CDFEN](#) for CDF RunII
- [D0EN](#) for D0 RunII
- [STKEN](#) for all other Fermilab users, including US-CMS T1

1.2 Chimera Namespace

The dCache namespace is shared by both dCache and Enstore. The implementation of the namespace is called Chimera. It presents data stored in the system in a directory tree structure. The namespace can be exposed to clients via a NFS mount. dCache supports NFS v2, v3 and v4.1 (pNFS) versions. NFS v4.1, which can be mounted only on the hosts running Scientific Linux 6 (SLF6) and higher, provides POSIX file I/O whereas v2 and v3 variants allow only access to file metadata and filesystem directory functions (unless the dcap preload library is used).

In addition to regular file metadata, the namespace stores storage-specific and other steering metadata in special files, called directory tags and file layers.

To browse file entries in the dCache system, on-site users can mount their experiment's namespace storage area on their own computers, and interact with it using standard UNIX operating system utilities. Normal UNIX permissions and administered export points are used to prevent unauthorized access to the name space.

Additionally the namespace can be browsed using the following clients. (This is described in the dCache document linked below.)

- WebDAV - Web Distributed Authoring and Versioning
- FTP – File Transfer Protocol
- SRM – Storage Resource Management
- xrd - xrootd file and directory meta-data utility

1.3 dCache

Information about dCache can be found at:

<http://www.fnal.gov/docs/products/enstore/PublicdCacheHowTo.html>

Chapter 2: Data Storage in Enstore

2.1 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.

2.2 File Organization on Storage Media

2.2.1 File Family

Files are grouped on data storage volumes according to a *file family* attribute. The grouping is really based on the triplet of quantities: storage group + file family + wrapper, collectively called a "volume family". 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 typically the file family.

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 (e.g. tape) may only contain files belonging to one file family.

Every directory in 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 namespace) is thus associated with the file family of the directory in the namespace to 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.

2.2.2 File Family Width

The file family width is an integer value associated with a file family that is used to limit write-accessibility on data storage volumes. For a given media type and for a given file family, Enstore limits the number of volumes (e.g. tapes) 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 (e.g. tapes) on which the volumes are loaded is also limited to the width.

2.2.3 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 three wrapper types implemented, `cpio_odc`, `cern` and `null`. The `cpio_odc` wrapper is the default wrapper set up by the Enstore admin when a new namespace (PNFS) area is created.

- 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 $(10^{21} - 1)$ bytes, which in effect limits the file size to the tape length, since spanning and striping are not supported. 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. (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.)
- For NULL volumes there is a `null` wrapper. This wrapper is also used for disk media.
- Tape affinity (which tape set files will go to) depends on the volume family which includes the file-family and the wrapper. Tapes with `cern`-wrapper files will only get `cern`-wrapped files written to them. Tapes with `cpio`-files will only get files that are `cpio`-wrapped to them. So you can have a split of a file family with files larger than 8GB going to a `cern`-wrapped tape set and files smaller than

8 GB to another tape set. This can waste space if a file family has only a few files greater than 8GB as they can take up a whole tape by themselves.

2.3 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. Your OS may restrict your files to yet a smaller size.

Tape sizes are as follows (all sizes shown are for non-compressed tapes):

T10000D	8000 GB
T10000C	5000 GB
LTO-4	800 GB
LTO-3	400 GB

Wrapper size limitations are as follows:

<code>cpio_odc</code>	8 GB -1B; (about $8 \cdot 10^9$ Bytes)
<code>cern</code>	About 10^{12} GB; or 86.7 Exabytes; (10^{21} -1 Bytes)
<code>null</code>	N/A; not available for writing to tapes; used for special situations

2.4 Data Storage Volumes in Enstore

Enstore is designed to support a variety of data storage media. Currently, only tapes and disks have been implemented. 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.

2.4.1 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. Labeling helps to easily identify each volume and/or test for a blank one, thereby inhibiting the inadvertent overwriting of used tapes. The Enstore administrators need to know whether tapes are labeled or completely blank when they are inserted into the library; the Enstore software can label tapes if necessary.

2.4.2 Tape and Library (Robot) Lifetimes

Over time wear and tear can damage the magnetic tape. This translates into an expected number of tape mounts before problems may start to occur. Eventually, these files are migrated to another tape, hopefully before any problems appear. The old tapes are then taken out of service. There are two limits. A soft limit where the files on a tape with more than that many mounts should be migrated, but it is not urgent to do so. A tape with more than the hard limit of mounts should be migrated as soon as possible.

Tape Type	Soft Limit	Hard Limit
T10000	10,000	15,000
LTO-4	10,000	15,000

Over time tape libraries (also known as tape robots) become obsolete. Files on tapes inside soon-to-be-decommissioned tape libraries will also need to be migrated. By default Enstore administrators will preserve deleted files on tape while migrating although groups may be asked if deleted files can be skipped during the migration process.

2.4.3 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. 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.

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.

2.4.4 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.

2.4.5 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`.

Chapter 3: Getting Ready to Use Enstore and dCache

3.1 Initial Steps for All Users

- 1) Find out what your volume quota is from your experiment’s Enstore administrator, and make sure you reserve what you need, according to your experiment’s procedures. The experiment’s Enstore liaison should request quota for the experiment via Services Now at <https://fermi.service-now.com/fsc/> and make a *STKEN Mass Storage Request*.
- 2) Find out what area in the namespace your experiment uses.
- 3) Read about file families, and find out from the people in your experiment responsible for implementing Enstore how file families have been configured for your experiment. Determine what file family(ies), and hence which subdirectories in the namespace, you want to write to and/or read from.
- 4) `Encp` and Enstore commands use whatever routing the client system or network administrator sets between the client system and the Enstore system for data transfers. If you (as the `sysadmin` or network admin of the large client machine) want to restrict the set of interfaces that `encp/Enstore` uses, you need to create the file `enstore.conf`. This file controls the interface-router mapping for the network connections used by `encp/Enstore`.
- 5) Navigate to the Enstore monitoring system web page, titled *Fermilab Mass Storage System*, at <http://www-ccf.fnal.gov/enstore/>. Select the Enstore system that your experiment uses, and browse the system information for it. You might want to bookmark this page.

6) Subscribe to the *stk-users@fnal.gov* listserv mailing list for announcements about Enstore and the STKEN Enstore system. D0 users, subscribe to *d0en-announce@fnal.gov*. CDF users, subscribe to *cdfdh_oper@fnal.gov*.

Data volumes are moved with the default TCP window size on the machine. There is a potential for an extreme performance degradation if the default window is set too large. A value of about 32K bytes works well at most locations at Fermilab.

3.2 Further Steps for Direct Access Enstore Users Only

If you access Enstore through dCache, this section doesn't pertain to you.

1) Make sure your node and network can provide adequate throughput. To determine the optimal data transfer rate, consult the Enstore administrators.

2) See if your experiment's `/pnfs` area is mounted on your machine, by using standard UNIX utilities like `df`, `cd` and `ls`. If it's already mounted, skip to step (6). If not, continue.

3) Check to see if authorization has been granted to mount the `/pnfs` area on the machine you plan to use. To do so:

a) Go to the *PNFS Exports Page*, at `http://www-<xyz>en.fnal.gov:/enstore/pnfsExports.html`, where `<xyz>` is one of `stk`, `d0` or `cdf`, depending on the Enstore system used by your experiment.

b) Scroll down to the *PNFS ExportList Fetch Begin: <date/time>* area, and look for your node and `/pnfs/storage-group` area. If they're listed, authorization has been granted; skip to (5). If not, continue.

4) Notify your experiment's Enstore liaison that you need authorization to mount the `/pnfs` area on the machine you plan to use. He or she will need to send your request on to enstore-admin@fnal.gov.

5) Once authorization has been granted, mount the `/pnfs` area on your machine if you have root permission, or send a request to the machine's system administrator to mount it. To mount the area yourself, edit the `/etc/fstab` file and add a line with the following strings (they should appear all on the same line in the file; we separate them into six lines here for clarity):

```
remote_enstore_server_node:enstore_server_directory
/pnfs/local_mount_point
mount type
comma_separated_attributes
0
0
```

where the 0 in the 2nd-to-last line means no dump of filesystem, and the 0 in the last line means no fsck checks at boot time. For example:

```
stkensrv1:/E872 /pnfs/E872 nfs user,intr,bg,hard,rw,noac 0 0
```



Usually, `local_mount_point` is the same as `enstore_server_directory`. Make sure that `local_mount_point` exists! (A typical error message is "backgrounding".)

6) Install UPS/UPD on your system. See Part III of the UPS/UPD manual at <http://www.fnal.gov/docs/products/ups/ReferenceManual/parts.html#partIII>.

7) Install the **encp** product on your machine (see below).

3.3 Installing Encp

This section pertains to you only if you access Enstore directly (not through dCache). To install the **encp** product from KITS using UPD, run:

```
$ setup upd
```

```
$ upd install -G "-c -q <xyz>" encp
```

where `<xyz>` stands for one of the Enstore systems. Currently, these include:

stken for general Fermilab users

d0en for D0 users

cdfen for CDF users

For example, a CDF experimenter would type:

```
$ upd install -G "-c -q cdfen" encp
```

Chapter 4: Copying Files with Encp

Encp is an end-user command used to copy data files from disk to storage media and vice-versa. Its use is being discouraged in favor of the dCache, however we document it here for completeness.

Please note that **Encp** can fail. Users must check the return codes to ensure that the command executed successfully.

Encp is maintained in KITS and in AFS product space as a separate product from Enstore, and is designed to be used in conjunction with it. **Encp** does not support recursive copies of data to and from Enstore; `ensync` is provided as a wrapper to **encp** for that purpose when writing to Enstore. **Encp** can copy multiple files to a single directory only. **Encp** can be used only from on-site machines in the `fnal.gov` domain.

In this chapter, we assume you have UPS/UPD running on your local machine.

4.1 Setup encp

To setup **encp**, run the command:

```
% setup -q <qualifier> encp
```

where **<qualifier>** stands for one of the Enstore system hosts. Currently, these include:

stken for general Fermilab (and CMS) users

d0en for D0 users

cdfen for CDF users

For example, a CDF experimenter would type:

```
$ setup -q cdfen encp
```

If you don't specify the qualifier, the environment variable `ENSTORE_CONFIG_HOST` may get set to the wrong value.

The `ENSTORE_USER_DEFINED_CONFIG_HOST` and `ENSTORE_USER_DEFINED_CONFIG_PORT` environmental variables can be specified before executing `setup` without the `-q` option. The `setup` will use these user defined values to set `ENSTORE_CONFIG_HOST` and `ENSTORE_CONFIG_PORT`.

4.2 Encp Command Syntax and Usage

Encp plays the same role in the Enstore system that **cp** plays in UNIX. The syntax is:

```
% encp [<options>] <source_file> <destination_file>
```

Remember to check the return code to ensure that the command did not fail.

Use the `--help` option to request the option listing for **encp**, or the `--usage` option for syntax information:

```
% encp --usage
```

```
encp [OPTIONS]... <source file> <destination file>  
encp [OPTIONS]... <source file> [source file [...]]\  
    <destination>
```

4.3 Copy Files to and from Enstore Media

4.3.1 Run encp

First, setup **encp** (using the `-q` flag). You can use filename expansion (wildcard characters to specify a group of files). We recommend, however, that you copy one file at a time. `Encp` returns 0 upon success and non 0 value upon error (1); the error is accompanied with explanatory output into `STDERR`.

Run the command as follows to copy a file to Enstore:

```
% encp [<options>] /<path-to>/.../<localfilename> \  
/pnfs/<storage-group>/.../<targetdir>/<remotefilename>  
ret=$?  
% if [ $ret -ne 0 ]; then  
    echo "error in encp transfer $ret"  
# retry logic  
fi
```

Note it is very important to check the status of `encp` to make sure it succeeded. `Encp` can fail for many reasons, including during routine maintenance work on the Enstore storage systems.

The presence of `/pnfs/` in the destination path indicates to the Enstore system that this is a copy to Enstore. To copy from Enstore, change the source and destination file specifications, e.g.,:

```
% encp [<options>] \  
  
/pnfs/<storage-  
group>/.../<targetdir>/<remotefilename> \  
  
/<path-to>/.../<localfilename>
```

4.3.2 Examples

1) Standard copy to Enstore; no options. Copy `myfile` to the directory `/pnfs/expt1/subdir/`:

```
% encp /path/to/myfile /pnfs/expt1/subdir/
```

2) Standard copy; no options. Download `/pnfs/expt1/subdir/myfile` to a different local directory from the `cwd`, and change the filename:

```
% encp /pnfs/expt1/subdir/myfile \  
/other/local/dir/newfilename
```

3) Request the process to output some information to screen (`--verbose`). Again, copy `myfile` to the directory `/pnfs/expt1/subdir/`:

```
% encp --verbose 3 /path/to/myfile \ /pnfs/expt1/subdir/
```

4) Copy all the files in the cwd starting with the string `trigger1` to the directory `/pnfs/expt1/subdir/`:

```
% encp ./trigger1* /pnfs/expt1/subdir/
```

5) Copy all the files in `/pnfs/expt1/subdir/` starting with the string `trigger1` to the cwd:

```
% encp /pnfs/expt1/subdir/trigger1* .
```

4.3.3 Preventing Unwanted Overwriting

When an **encp** job starts, it first creates a zero length output file for every input file. In this way it reserves the necessary file names and thus prevents another party from starting a competing **encp** process which would clobber the first.

4.3.4 Killing an encp Process

There are four traditional ways to abort a process:

- Ctrl-C (SIGINT)
- Ctrl-\ (SIGQUIT)
- kill (SIGTERM)
- kill -9 (SIGKILL)

The first three result in **encp** removing any remaining zero length files (as discussed directly above). With a “kill -9”, no cleanup occurs. For multi-file transfers, files successfully transferred before the signal is caught will be left alone.

4.3.5 Encp Transfer Rates Defined

You can find out the network rates by running the command **encp --verbose 1** or from the **Encp History** page.

-) Network transfer rate - The rate at which the file was transferred over the network between the **encp** node and the mover node, in megabytes per second.
-) Transfer rate - The rate measured when moving the file between the disk local to **encp** and the tape, in megabytes per second (includes reading file, network transfer of file and writing file; does not include tape mount or seek times).
-) Drive rate - The rate measured when reading/writing from/to the tape drive, in megabytes per second.
-) Disk rate - The rate of reading/writing the file from/to the disk local to **encp**, in megabytes per second.

-) Overall rate - The rate for the overall process (from when the mover connects to **encp** until the mover sends its final “success or failure” message to **encp**), including all **encp** and media overhead, in megabytes per second (this does not include time spent in the library manager queue).

Without the `--threaded` switch the network and disk rates will be reported as the same.

4.3.6 Isolating Source of Bottlenecks

Encp supports isolating the rate transfers in the tape, disk and network via the option `--threaded` used in conjunction with the option `--verbose` with a value of 1 or higher. If `--threaded` is not specified, then the network and disk rates are calculated the same way as before, and display the same value as one another. The various rates are defined in the online monitoring pages, under **Encp** History Help.

Here is an example without `--threaded` (with off-topic output removed for brevity):

```
% encp --verbose 1 /pnfs/xyz/10MB_002 /tmp/myfile
```

with output:

```
...
Transfer /pnfs/xyz/10MB_002 -> /tmp/myfile: 10485760 bytes
copied from 'TEST01' at 1.57 MB/S
(1.67 MB/S network) (2.87 MB/S drive) (1.67 MB/S disk)
...
Completed transferring 10485760 bytes in 1 files in
14.2875500917 sec.
Overall rate = 0.7 MB/sec. Drive rate = 2.87 MB/sec.
Network rate = 1.67 MB/sec. Exit status = 0.
Note in the above output, the network and disk rates are the same.
```

Here is an example with `--threaded` and `verbose 1` (abbreviated output); note that the rates are separated, so that you can see where the bottleneck is (the disk, in this case):

```
% encp --verbose 1 --threaded /pnfs/xyz/10MB_002
/tmp/myfile
```

It produces output:

```
...
Transfer /pnfs/xyz/10MB_002 -> /tmp/myfile:
10485760 bytes copied from 'TEST01' at 2.41 MB/S
(8.09 MB/S network) (9.36 MB/S drive) (2.71 MB/S disk)
```

...
Completed transferring 10485760 bytes in 1 files in
14.9129179716 sec.
Overall rate = 0.671 MB/sec. Drive rate = 9.36 MB/sec.
Network rate = 8.09 MB/sec. Exit status = 0.
The network and drive each have rates above 8 MB/s, and the disk rate is only 2.71
MB/s.

4.3.7 Encp Error Handling

Encp has functionality to retry and resubmit requests, where we distinguish between these two terms. **Encp** will *retry* (i.e., resend) a request after an error occurs. **Encp** will *resubmit* a request if it has been waiting for a mover for over 15 minutes; this is not due to an error condition but rather to keep queues current regardless of the server condition. The **encp** exit statuses are zero (0) for success, one (1) for error.

You must check the status of **encp** to make sure the file got copied successfully. **Encp** can fail for numerous reasons, including during Enstore maintenance work. Usually, you should retry failed transfers (see exceptions below). Check to see if there is maintenance work scheduled and retry after the maintenance work is complete. If the problem persists, create a service desk incident.

There are two general classifications of errors in **encp**: those that can be retried and those that can't. Three "retriable" errors can occur before the error "TOO_MANY_RETRIES" occurs.

The most common nonretriable errors include:

NOACCESS - the system has marked the volume as "potentially" bad

NOTALLOWED - an enstore administrator has marked a tape as unavailable for user access

USERERROR - usually is a file accessibility problem (doesn't exist, has wrong permissions, etc.)

Among the less common ones, there are:

VERSION_MISMATCH - the **encp** version is no longer compatible with the running Enstore system

CRC_MISMATCH - indicates a corruption error

FILE_WAS_MODIFIED - **encp** determined that another process modified the file during the transfer

Ask your Enstore administrator or submit a service desk request if you see others.

4.3.8 Finding files in different Enstore Systems

File reads: When reading from Enstore, **encp** can determine whether the current value of `$ENSTORE_CONFIG_HOST` is pointing to the Enstore system that contains the requested file. If it points to the wrong one, **encp** will try the other Enstore installations to find the requested file. If the file is found, **encp** will retrieve the file; if the file is not found on any Enstore system, an error is returned to the user.

File writes: When writing to Enstore; the value of `$ENSTORE_CONFIG_HOST` is always used.

4.3.9 Order of Processing Queued Requests

For reads, files are sorted out by volume. When all files from a single volume are complete, the next volume's files are requested.

For writes, one file at a time is submitted to the library manager. The order is that in which the files are specified on the command line. The tape is kept mounted during file writes on a best-effort basis.

4.3.10 NULL File Directories

When **encp** accesses files via a null mover, a directory in the file path must contain the name NULL, e.g., `/pnfs/expt1/NULL/myfile`.

4.3.11 Important Environment Variables

There are two important environment variables that are generally set in the Enstore setup script. Users who work on more than one Enstore system (e.g., `stken` and `cdfen`) at a time in different windows may need to know about these in case they use the wrong window for a particular Enstore system!

The variables are:

`$ENSTORE_CONFIG_HOST` - points to the Enstore server that is running the configuration server.

`$ENSTORE_CONFIG_PORT`

`$ENSTORE_RANDOM_UB`

`$ENSTORE_RANDOM_LB`

When reading from `/dev/zero`, `/dev/random` or `/dev/urandom` these represent the upper and lower bounds the random amount of data to read in bytes.

`$ENCPCANONICAL_DOMAINNAME`

When reading files; **encp** attempts to resolve the given name with names like `/pnfs/xyz`, `/pnfs/fs/usr/xyz` and `/pnfs/<network_name>/usr/xyz`. If the hostname name of the current

host resolves, for example to pool.storage.fnal.gov, then **encp** would look for /pnfs/storage.fnal.gov/usr/xyz. If the real path is /pnfs/fnal.gov/usr/xyz, then we need to set ENCP_CANONICAL_DOMAINNAME to fnal.gov for **encp** to automatically try the correct canonical PNFS path.

4.3.12 Encp Limitations

Encp does not work from behind a firewall. It uses an “active” protocol for data transmission.

New versions of Linux come with an internal firewall enabled by default. To use **encp** on these nodes the firewall must be disabled first.

4.4 Encp Command Options

In this section, we have placed a bomb in front of any option that should be used with utmost care; these options, if misused, can adversely affect not only your jobs, but those of others, as well. We have placed a pointing finger in front of options that, if misused, may adversely affect your own job, but not others’ jobs.



--age-time <AGE_TIME> Specifies the time period, in minutes, after which the priority is eligible to change from the current job priority. We recommend that you do not set this, just use the default (which is “never”).



--array-size <ARRAY_SIZE> Sets the number of buffers in the array. If **--threaded** is specified but this option is not, array-size defaults to 3. If this is used without **--threaded**, this value becomes 1 and is ignored. Changing this value for multi-threaded transfers may increase transfer speed.



--buffer-size <BUFFER_SIZE> Sets the number of bytes of data to transfer at one time (default is 256k). Increasing this value may increase transfer speed. This value must remain lower than the available memory.



--bypass-filesystem-max-filesize-check

Disables the check to protect against the user reading from Enstore a file larger than the maximum size file the local filesystem supports. Use this switch with care.

--check Checks that enstore is running, that the metadata is ok, and that **encp** would thus start. Running the **encp** command successfully using this switch is not sufficient to guarantee that a transfer would succeed.

Result of 0 is success, 1 is failure, 2 means indeterminable at this time.

--copies <COPIES> Directs **encp** to write <COPIES> number of additional multiple

copies on tape. A value of zero is the default; indicating that only the original should be written.

It is possible to enable this feature by setting the library tag to contain a comma separated list of libraries without any white space.

`--copy <COPY>` Directs **encp** to read the <COPY> number multiple copy on tape. A value of zero is the default; indicating that the original should be read.

`--data-access-layer` Turns on special status printing; output has standardized format whether error occurred or not.

 `--delayed-dismout <DELAY>` Specifies time period in minutes to delay dismount of volume. Use this to tell Enstore: “More work is coming for the volume, don’t dismount the volume too quickly once the current transfer is completed.”

 `--delpri <DELPRI>` Changes the initial job priority by specified value after a period given by the age-time switch. We recommend that you don’t set this, just use the default (1).

 `--direct-io` Uses direct I/O for disk access on supporting file systems. (Direct I/O is not universally supported; some filesystems, versions of filesystems, kernels, etc. do not support it. If this doesn’t work for you, contact an enstore admin, and communicate your kernel, library versions, filesystem and filesystem version.) Generally, direct I/O makes disk access slower. But when the size of the read/write buffer is made large enough, say, 64Mb or larger, direct I/O is faster because of the skipped memory-to-memory copy.

`--ecrc` (stands for Enstore crc) This can be used when reading from Enstore. After a file is written to disk, this causes Enstore to reread the disk copy of the file and recalculate the checksum on it.

 `--ephemeral` (The options `--ephemeral` and `--file-family` require care when used so that tapes do not get mounted in a way that causes improper and/or inefficient tape usage. Beware of runaway scripts.) This option creates a temporary file family of name “ephemeral”, and copies files to this ephemeral file family on storage media in the order specified. Overrides file family tag in /pnfs destination directory.

 `--file-family <FILE_FAMILY>`

This is used to write data on volumes assigned to specified file family. Overrides file family tag in /pnfs destination directory.

`--help` Displays the list of options for **encp**.

 `--mmap-io` Uses memory-mapped I/O for disk access on supporting file systems (see the *Enstore Glossary* for an explanation). Make sure you have read and write permissions on the file.

 `--mmap-size <MMAP_SIZE>` The amount of data to map from the file to local memory at one time in bytes (default is 96Mb); use with `--mmap-io`.

`--no-crc` Tells **encp** to bypass the crc on the local file. (For the minor performance enhancement that this affords, we discourage use of this option.)

`--pnfs-is-automounted` Typically, users should not automount pnfs. If you do, you can specify this option. It alerts **encp** to retry errors due to known OS automounting problems.

Do not use this in non-automounted cases; it can slow the setup of the transfer.

 `--priority <PRIORITY>` Sets the initial job priority to the specified integer value. We recommend that you don't set this, just use the default.

`--threaded` Multithreads the actual data transfer.

`--usage` Displays information about the **encp** options.

`--verbose <LEVEL>` Changes the amount of information printed about the transfer; provide an integer value. Default is 0. Larger integer numbers provide more “verbosity”. Largest meaningful number may change as development continues.

`--version` Displays **encp** version information.

Chapter 5: Copying Directory Structures with Ensync

5.1 About Ensync

Ensync is a wrapper for **encp** that allows you to copy the contents of an entire directory structure to Enstore via a single command. It is intended for uploading files to Enstore only, not downloading from it. Ensync makes a call to **encp** to handle each file transfer, and the transfers are done serially. The ensync program is intended for smaller experiments lacking the resources to create their own scripts to do this sort of thing, and without the high data volume of the large experiments.

Ensync works similarly to **rsync -R** in that it recursively copies files down a directory hierarchy, which **encp** cannot do. It behaves in much the same way that **rcp** does, but has more features and uses the rsync remote-update protocol to greatly speed up file transfers when the destination file already exists. It creates an area in PNFS namespace structurally similar to that from which the user is copying. It copies files/directories from a user's local disk to this space. If a file already exists in Enstore with the same relative path/filename as one on the user's disk, the user's file does not get copied; the preexisting Enstore file stays as is.

5.2 Ensync Command Syntax

The command takes two arguments, the “from” and the “to” directories or files. The syntax is:

```
% ensync /<path_to>/.../<local_dir> /pnfs/.../<dest_dir>
```

There are no options to specify. A few notes:

- Symbolic links work if the target is under the same mount point as the link.
- Hard links are not kept. A new copy for each link would get two separate, identical files.
- Ensync transfers one file at a time, it does not transfer them in parallel.

Chapter 6: Overview of the Enstore Servers

In this chapter we describe the software modules that act as Enstore servers and the libraries with which they interact.

The servers include:

- File Clerk (FC)
- Volume Clerk (VC)
- Library Manager (LM)
- Mover (MV)
- Media Changer (MC)
- Configuration Server (CS)

All of the above-listed servers must be running in order for data reads and writes to succeed.

The Enstore monitoring framework includes:

- Inquisitor
- Alarm Server (AS)
- Log Server (LS)
- Event Relay (ER)
- Monitor Server
- Accounting Server
- Drivestat Server
- Information Server
- Ratekeeper

Typically, data transfer can still take place even if any of these monitoring systems is down.

6.1 File Clerk

The File Clerk (FC) is a server that tracks files in the system. It manages a database of metadata for each data file in the Enstore system. The metadata includes the file's name, its unique identifier (the bit file ID, or bfid, that the FC itself assigns to each new file), the volume on which it resides, and so on. You can get information on specific files using the **enstore info** command described in section two of: http://www-ccf.fnal.gov/enstore/Enstore_Administrator_Guide.pdf

6.2 Volume Clerk

The Volume Clerk (VC) is a server that stores and administers storage volume (tape) information. You can get information on specific volumes using the **enstore info** command.

6.3 Library Manager

A Library in Enstore is comprised of both the physical media and a robot arm used to mount the media in attached drives. An Enstore library is typically called a robot. A library/robot interfaces to software that controls the robot arm. Each library can contain a variety of media types and employ different types of media drives.

A Virtual Library (VL) is a subset of an Enstore library. It can contain one and only one type of media. A Library Manager (LM) is a server which is bound to a single Virtual Library (VL), and controls what happens within that VL. We speak of bound “LM-VL pairs”. An LM receives requests for file reads and writes from the user, stores these unassigned requests in a queue, prioritizes them, and dispatches the requests to a Mover for actual data transfer to and from its VL.

There may be many LM-VL pairs in an Enstore system. There may be more than one LM-VL pair for each media type, but not vice-versa. For example, given an STK Powderhorn library holding 20, 60 and 200 GB media, Enstore would need to divide it into at least three LM-VL pairs.

You can get information on specific library managers using the **enstore library** command.

6.4 Mover

A Mover (MV) is a process responsible for efficient data transfer between the **encp** process and a single, assigned media drive in a library (robot). The Mover receives instructions from a Library Manager (LM) on how to satisfy the users’ requests. The Mover sends instructions to the Media Changer (MC) that services the Mover’s assigned drive in order to get the proper volume mounted.

A mover can be configured to serve multiple LMs. (The media types governed by the LMs must be supported by the Mover’s assigned drive.) Allowing flexible LM assignment has two benefits:

- First, since a virtual library (an LM-VL pair) handles only one type of media, a drive which can handle multiple types of media (e.g., different capacity media) can be shared by multiple LM-VL pairs without a static partitioning of the system.
- Secondly, suppose user groups A and B want to share the capacity of a VL, in which half the tapes belong to group A and the other half to group B. You want to guarantee that groups A and B each get one third of the tape drives, and that the last third is shared. To do this, your administrator can configure the Movers to partition resources in the VL,

and assign an LM to each type of use.

6.5 Media Changer

The Media Changer (MC) mounts and dismounts the media into and out of drives according to requests from the Movers. One MC can serve multiple drives and thus multiple VLs. When the drives are in the robot, the MC is the interface to the robotic software.

6.6 Configuration Server

The Configuration Server (CS) maintains and distributes the information about Enstore system configuration, such as the location and parameters of each Enstore component and/or server. At startup, each server asks the CS for the information it needs (e.g., the location of any other server with which it must communicate). New configurations can be loaded into the CS without disturbing the current running system.

6.7 Inquisitor

The Inquisitor monitors the Enstore servers, obtains information from them, and creates reports at regular intervals that can be viewed on the web under http://www-stken.fnal.gov/enstore/enstore_saag.html

If the Inquisitor goes down, the **System-At-A-Glance** web page indicates this by a red ball next to *Inquisitor*. In this case, data can still be transferred via **encp**, however, the information on the reports mentioned above doesn't continue to update!

6.8 Alarm Server

The Alarm Server (AS) maintains a record of alarms raised by other servers. Since Enstore attempts error recovery whenever possible, it is expected that raised alarms will need human intervention to correct the problem. The AS compares each newly raised alarm with the previously raised ones (it raises a counter) in order to prevent raising the same alarm more than once. Alarm output can be configured to be sent in email messages, to a web page, and so on for notification.

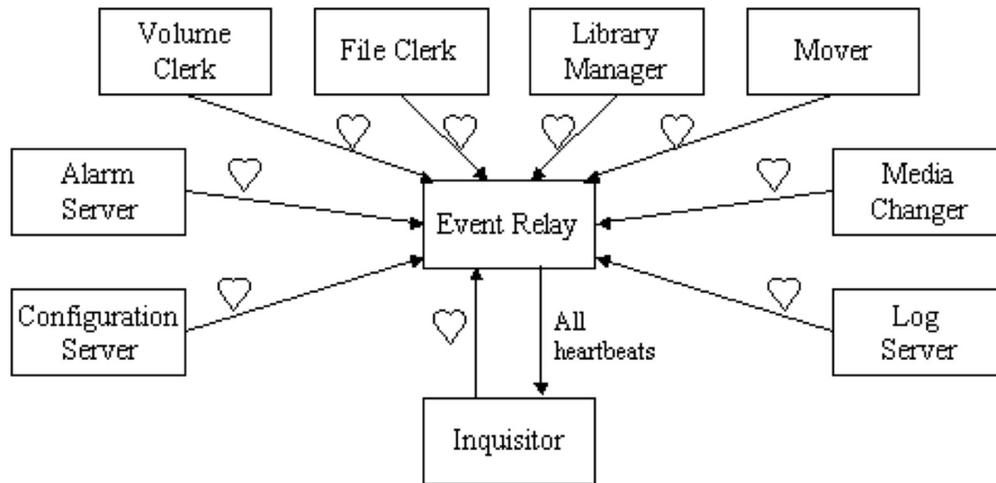
6.9 Log Server

The Log Server (LS) receives messages from other processes and logs them into formatted log files. These messages are transactional records. Log files are labeled by date. Every night at midnight, the currently opened log file gets closed and another one is opened. Logs are backed up to tape.

6.10 Event Relay

The Event Relay (ER) is a server that forwards messages based on subscription. All the Enstore servers send messages to the ER. Any server may "subscribe" to the ER in order to have

messages of particular types forwarded to it.



For example, the ER periodically receives “I’m alive” messages (called *heartbeats*) from the other servers in the system. The Inquisitor subscribes to the heartbeat messages, so the ER forwards these messages to it. This is illustrated in the image below:

 If the ER goes down (indicated by a red ball next to *Event Relay* on the **System-At-A-Glance** web page), the information on **Enstore Server Status** may not be accurate.

6.11 Monitor Server

The Monitor Server (MS) is available for investigating network-related problems. It attempts to mimic the communication between **encp**, the corresponding library manager, and the mover. To initiate a test of this kind, you must use the **enstore monitor** command.

6.12 Accounting Server

The accounting server is a front-end to a Postgres SQL database, and at this time is not intended for use by end users. It maintains statistical information on a running system. Such information is not essential to operations, however it can be used by administrators to analyze the performance and utilization of the system for purposes of troubleshooting and future planning.

6.13 Drivestat Server

Drivestat server maintains statistical information of the drives. This is used to update the “ingest rate” plots. As with the accounting server, such information is not essential to operations, however it can be used to analyze the performance and utilization of the system for purposes of troubleshooting and future planning.

6.14 Info Server

The Information Server is a read-only server that maintains detailed file and volume information. You can access this information for particular files and volumes using the `enstore info` command and its various options.

6.15 Ratekeeper

The ratekeeper performs a number of monitoring tasks that update the accounting DB at regular intervals with information about instantaneous rate, drives busy and slot usage information.

Chapter 7: Enstore Commands

Enstore provides commands that allow you to communicate with various components of the system. The basic syntax of all Enstore commands is

```
% enstore <command> [--option [argument] ...]
```

All options start with a double dash (--). The return codes are 0 (zero) for success, non-zero for failure (currently all failures return number 1).

For specifics on Enstore commands, please see Chapter 8 of the Enstore Administrator's Guide available at:

http://www-ccf.fnal.gov/enstore/Enstore_Administrator_Guide.pdf

Chapter 8: Monitoring Enstore on the Web

There are several installed Enstore systems at Fermilab. For each Enstore system, a separate but structurally identical series of web pages is available for monitoring the system and any jobs you've submitted to it. This can be accessed from:

<http://www-ccf.fnal.gov/enstore/index.html>

The Enstore pages present snapshots of the status of various components of the Enstore system, and the pages are updated and refreshed periodically. The auto-refresh time interval varies from page to page, and does not correspond with the information update interval, which also varies from page to page. See the online help screens for more detailed information.

Chapter 9: Job Priority and Queue Management

Users submit read and write jobs to **encp** (often through an interface, e.g., dCache). **Encp** sends a request for each job to an Enstore library manager for processing. The library manager receives these requests, stores them in a queue, assigns a priority to each one, and passes them to a mover for actual data transfer. A request's priority determines when it will get processed relative to others in the queue, and its priority may change as circumstances change while it waits in the queue. There are four items that factor into determining priority: category, numerical value within the category, "Fair Share", and ownership of resources by a group/experiment.

9.1 Job Priority Categories

There are two categories of job priority: normal and DAQ/Admin. The default priority is "normal". DAQ/Admin priority, as its name implies, is reserved for high-priority jobs.

DAQ/Admin priority is granted only to job requests that satisfy certain preconfigured conditions. Conditions, if set, filter on the request's originating user name, group, node, and so on. For example, conditions could be set to grant DAQ/Admin priority to all jobs submitted by user *joe* from node *mynode1*, by users *george* or *kim* from any node, and by any user from node *myspecialnode*; all other jobs get normal priority.



Setting conditions for DAQ/Admin priority is an administrative function.

Requests that get submitted and receive DAQ/Admin priority get moved to the head of the LM's request queue. If there is more than one at a time, the other priority-related factors determine the order in which these requests get processed. Any normal priority transfer that is in progress is allowed to complete, but the system does not then process other normal priority requests in the queue that are waiting for the same volume. At the completion of the current transfer, the tape is replaced in the drive if necessary, then the (first) DAQ/Admin request gets processed.

9.2 Numerical Priority Values

Within its priority category, each request is also assigned an initial numeric priority value. The numeric priority is set by default according to preconfigured conditions. It can be overridden on the command line using the **encp** options described in section 6.6 *Encp Command Options*, although we strongly recommend against doing this. The numeric priority may change while the request waits in the queue depending on (a) the **encp** options used, (b) the elapsed time in the queue.

9.3 Fair Share Resource Allotment

Enstore queue management has an algorithm called "Fair Share", that helps to keep any one storage group (experiment or group) from monopolizing tape drives.

When a mover which does not have a mounted tape asks the Library Manager for the next request, the Fair Share determines which storage groups currently have requests in progress (at a mover) and which ones don't, then gives preference to requests associated with those that don't. This helps ensure that there is a fair distribution of resources available to all groups currently using the system.

9.4 Resource Ownership

When an experiment or group purchases one or more tape drives, the drives go into the pool of Enstore resources, accessible by all users, with the recognition that the purchasing group has preferential access to this number of drives. The Fair Share configuration gets modified to guarantee this access. The priorities of other requests in the queue may be perturbed when Enstore needs to free up one or more tape drives for jobs submitted by the purchasing group.

In contrast to DAQ/Admin requests, these (normal priority) requests, identified by their storage group, must wait until a tape has dismounted in the normal way before being processed. A tape normally dismounts after all requests in the queue requiring that tape have completed.

Enstore Glossary

accounting server

This server maintains statistical information on a running system. It is Enstore's interface to a database of transfer-related data.

active file

Any file in Enstore that is not deleted.

alarm server (AS)

The Alarm Server maintains a record of alarms raised by other servers, and creates a report that's available online.

bfid

Bit file id; an Enstore-assigned, unique identifier for a data file.

cern wrapper

A file family wrapper that accommodates data files up to $(10^{21} - 1)$ bytes. 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. See **file family wrapper**.

configuration server (CS)

The Configuration Server maintains and distributes the information about Enstore system configuration, such as the location and parameters of each Enstore component and/or server.

cpio_odc wrapper

A file family wrapper which allows the file to be dumpable via cpio. This wrapper has a file length limit of $(8G - 1)$ bytes. See **file family wrapper**.

crc (Cyclic Redundancy Check)

Used to verify that data has been stored properly; it's used like a checksum, but is less prone to multiple-bit errors. During a transfer, both sides calculate the crc and compare the values, unless the **--no-crc** option is specified. Enstore uses a one seeded Adler 32 crc by default. Enstore at Fermilab uses a zero seeded Adler32 crc.

cwd

current working directory

dCache

DCache is a data file caching system which acts as an intelligent manager between the user and the data storage facilities. It optimizes the location of staged copies according to an access profile. It decouples the (potentially slow) network transfer rate from the (fast) storage media I/O rate in order to keep the mass storage system from bogging down.

dCap

DCap is a dCache-native C-API access protocol. The **dCap** package comprises the commands `dcap`, `dc_check` and `dc_stage`.

dc_check

A **dCap** command which checks if a file is on disk in the dCache.

dccp

A **dCap** command which provides a **cp**-like functionality on the PNFS file system.

dc_stage

A **dCap** command which prestages a read request.

ddencp

A command packaged with **encp** which copies a local file to another local file.

DESY

Deutsches Elektronen-SYNchrotron; a laboratory in Hamburg, Germany that conducts particle physics research.

direct I/O

Direct I/O differs from normal disk read/writes in that it by-passes the file system's buffer cache. This is achieved by skipping the (normally done) copy that goes from the application memory space to the kernel buffer's memory space. Direct I/O is an SGI/Linux extension. Compare to **memory-mapped I/O** and **POSIX I/O**.

door (for dCache)

A door is a protocol converter (e.g., for FTP, dCap) between clients and internal dCache systems. Each door is associated with a particular port on the dCache server, and has its own access profile.

drivestat server

The drivestat server maintains statistical information of the drives.

ecrc

Stands for Enstore crc (cyclic redundancy check); see **erc**. **ecrc** is a command packaged with **encp** which calculates the crc of a local file.

en_check

A command packaged with **encp** which determines if a file is on tape.

encp

Encp as an end-user command is considered to be deprecated. It was designed to be used with Enstore, and used to copy data files from disk to storage media and vice-versa. This command is distributed as part of the **encp** product, available from kits under <ftp://fnkits.fnal.gov:8021/products/encp/> or <ftp://fnkits.fnal.gov:8021/KITS/<OS>/encp/>, e.g., <ftp://fnkits.fnal.gov:8021/KITS/Linux/encp/>

The **encp** product also includes three diagnostic tools: **ecrc**, **ddencp**, **en_check**.

Enstore

Enstore is the mass storage system implemented at Fermilab as the primary data store for experiments' data sets. It provides distributed access to data on tape or other storage media both locally and over networks.

enstore.conf

A configuration file to allow for multiple network interface cards dedicated to Enstore, and to map the interfaces to routers. The default location for the file is `/etc/enstore.conf`. The location of the file can be overridden with the environment variable `ENSTORE_CONF`. See `ENSTORE_CONF`.

ENSTORE_CONF

Environment variable that can be used to override the location of the `enstore.conf` file.

See **enstore.conf**.

ENSTORE_CONFIG_HOST

An environment variable which points to the Enstore server that is running the configuration server

ENSTORE_CONFIG_PORT

An environment variable which sets the port number; the value is (by convention) 7500 for all installations at Fermilab.

ensync

A wrapper for **encp** that allows you to copy the contents of an entire directory structure to Enstore via a single command.

event relay (ER)

The Event Relay is a server that forwards messages based on subscription. All the Enstore servers send messages to the ER. Any server may “subscribe” to the ER in order to have messages of particular types forwarded to it.

fairshare

A mechanism used in Enstore’s queue management that helps to keep any one storage group (experiment or group) from monopolizing tape drives. Fairshare determines which storage groups currently have jobs in progress (at a mover) and which ones don’t, then gives preference to requests associated with those that don’t.

file clerk (FC)

The File Clerk is a server that tracks files in the system. It manages a database of metadata for each data file in the Enstore system.

file family

A file family is a grouping of data; it defines a category, or family, of data files. Each experiment defines a set of file families for its data. A given storage volume may only contain files belonging to one file family.

file family width

File family width is a value used to limit write-accessibility on data storage volumes. At any given time, Enstore limits the number of volumes associated with a given file family that are open for writing to the value of the file family width.

file family wrapper

A file family wrapper consists of information that gets added to the front and back of data files as they’re written to media, and defines the files’ format on the storage volume. The format of the wrapper depends on the type of wrapper used. (See **cern wrapper** and **cpio_odc wrapper**.)

filemark

A filemark is a physical mark on tape indicating end of file. Tape drives recognize it and can do high speed searches over it.

ftp

File transfer protocol.

gridftp

See **GSI ftp**.

GSI ftp

An implementation of ftp that uses Grid Proxies for authentication and authorization and is compatible with popular tools such as globus-url-copy (from the globus toolkit).

information server

A read-only server that maintains detailed file and volume information.

inquisitor

The Inquisitor monitors the Enstore servers, obtains information from them, and creates reports at regular intervals that can be viewed on the web.

job

In Enstore terminology, a job is what a user submits to **encp**. See **request** for comparison.

Kerberized ftp client

A Kerberized ftp client is an ftp client that implements Kerberos v5 authentication.

layer

Pnfs stores metadata about each file in “layers”, each layer containing a specific type of metadata. Each stored data file has its own set of these layers. Currently, only layers 1 and 4 are used.

library

A library in Enstore is comprised of both the physical data storage media, robotic devices and drives. An Enstore library is typically called a robot.

library manager (LM)

A Library Manager is a server which controls a virtual library. LMs receive requests for file copies from users via **encp** and they distribute the requests to the Movers.

log server (LS)

The Log Server (LS) receives messages from other processes and logs them into formatted log files that are available online.

media changer (MC)

The Media Changer mounts and dismounts the media into and out of drives according to requests from the Movers.

media drive

The physical device servicing a storage volume, e.g. a tape drive.

memory-mapped I/O (mmapped I/O)

With this type of I/O, part of the CPU’s address space is interpreted not as accesses to memory, but as accesses to a device; once you map a file to memory, changes made to the memory map are propagated back to the file. Mmapped I/O strives to avoid memory copies of the data between the application memory space and the kernel memory space (also see **direct I/O** and **POSIX I/O**). Mmapped I/O is in the POSIX standard.

monitor server (MS)

The Monitor Server is available for investigating network-related problems. It attempts to mimic the communication between an **encp** request, the corresponding library manager, and the mover.

mover (MV)

A Mover is a process responsible for efficient data transfer between the **encp** process and a single, assigned media drive in a library (robot). The Mover receives instructions from a

Library Manager (LM) on how to satisfy the users' requests. The Mover sends instructions to the Media Changer (MC) that services the Mover's assigned drive in order to get the proper volume mounted.

null wrapper

A file family wrapper for NULL volumes. See **file family wrapper**.

pnfs layer

See "layer".

pnfs namespace

Pnfs is an independent namespace package, written at DESY. It presents a collection of library database entries as a UNIX-like file system, and thus allows users to browse stored files as though they reside in this file system. 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.

pnfs tags

See tags.

POSIX I/O

POSIX is a name applied to a widely used family of open system standards based on UNIX. POSIX I/O refers to the POSIX standards for I/O.

request

In Enstore terminology, after a user submits a job to **encp**, **encp** sends a request to Enstore to process the job. See **job** for comparison.

resubmit

Encp has functionality to retry and resubmit requests, where we distinguish between these two terms. **Encp** will *resubmit* a request if it has been waiting for a mover for over 15 minutes, but not due to an error condition. See **retry**.

retry

Encp has functionality to retry and resubmit requests, where we distinguish between these two terms. **Encp** will *retry* (i.e., resend) a request after an error occurs. See **resubmit**.

SRM

SRM (Storage Resource Management) is the middleware for managing storage resources for the grid.

storage group

A storage group is an identifier corresponding to an experiment that Enstore uses as it controls and balances assignment of resources such as tape drives and media. Each storage group (i.e., each experiment) is assigned an area in PNFS.

storage volume

A unit of mass storage, e.g., a tape.

streaming (of files on tape)

Streaming refers to the sequential access of adjacent files on tape at the maximum tape read/write speeds.

striping (of files on tape)

Striping refers to single files (usually large ones) being split onto two or more volumes, each writing simultaneously, in order to expedite the writing process. (Striping is not supported

under enstore.)

suspect volume

A volume becomes suspect when a mover communicates to the appropriate library manager that it had a problem with the volume. It is not yet established that the volume is faulty.

tags

Pnfs uses tag files (usually just called tags) in the `/pnfs` namespace to specify file-specific configuration information, and **encp** transfers this information to Enstore. Tags are associated with directories in the `/pnfs` namespace, not with any specific file, and thus apply to all files within a given directory.

virtual library

A Virtual Library (VL) is a subset of an Enstore data storage library. It can contain one and only one type of media. It is paired with its own library manager which controls it.

volume

See storage volume.

volume assert

A job in which a volume (tape) gets mounted and certain attributes are read in order to check, and thus assert, that the volume is “ok” (without actually checking the entire contents).

Requesting volume asserts is an administrative task, and these requests are assigned the lowest priority.

volume clerk

The Volume Clerk (VC) is a server that stores and administers storage volume (tape) information.

volume family

The triplet “storage group + file family + file family wrapper” is called a volume family. In order for different data files to be stored on the same volume, all three of these pnfs tags for the files must match.

wrapper

See file family wrapper.