Migration: Surfing on the Wave of Technological Evolution – An ENSTORE Story

C.-H. Huang, W. Baisley, J. Bakken, D. Berg, E. Berman, T. Jones, D. Litvintsev, A. Moibenko, G. Oleynik, D. Petravick, G. Szmuksta, M. Zalokar FNAL, Batavia, IL 60510, USA

Abstract

ENSTORE is a very successful petabyte-scale mass storage system developed at Fermilab. Since its inception in the late 1990s, ENSTORE has been serving the Fermilab community, as well as its collaborators, and now holds more than 3.5 petabytes of data on tape. New data is arriving at an ever increasing rate.

One practical issue that we are confronted with is: storage technologies have been evolving at an ever faster pace. New drives and media have been brought to the market constantly with larger capacity, better performance, and lower price. It is not cost effective for a forward looking system to stick with older technologies. In order to keep up with this technological evolution, ENSTORE was in need of a mechanism to migrate data onto newer media.

Migrating large quantities of data in a highly available mass storage system does present a technical challenge. An auto-migration scheme was developed in ENSTORE that carries out this task seamlessly, behind the scenes, and without interrupting service nor requiring much operational attention. After two years in service, auto-migration has lived up to its expectations and ENSTORE has gone through several generations of drives and media. In addition, migration can be used in media copying, media consolidation, and data compaction.

In this paper, we are going to present the conceptual design of ENSTORE, the issues in data migration in a highly available mass storage system, the implementation of auto-migration in ENSTORE, our experience and its extended applications.

INTRODUCTION

ENSTORE is a petabyte scale, highly available tape based permanent storage system developed by and installed at Fermilab. When data size is huge, tape based systems provide the best price/capacity ratio with reasonable performance.

Being a tape based system, its total capacity depends on the number of media, its online capacity further depends on the size of the media library, and, its maximal performance depends on the performance of the library, the number of drives and the aggregation of their individual performance within the capability of the library and communication channels.

In theory, a tape based storage system is easily scalable without a pre-set limit. In reality, it is limited by the physical space that houses the system and the media, the infrastructure (media libraries, drives, ... etc.) to support its operation and the operational cost to run the infrastructure. Every thing is further constrained by capital. In fact, the total capacity has a lower bond because that is the amount of the space where the data are to be stored and such space is needed and is the whole purpose of having a storage system in the first place.

It is obvious to see, if the density of the media is increased, the capacity of the system is proportionally increased without increasing the infrastructure, the physical space, nor the operational cost. Similarly, if the performance of individual drives are increased, the performance of the whole system is increased without increasing the physical space nor the operational cost. The increase of the cost of infrastructure can be offset by the benefit from the performance and, very likely, the capacity increase. To replace a drive by another kind, it almost always means media change, either physically, such as the form factor, or logically, such as the format.

The reality is, technologies have not stopped reinventing themselves. The lower price/capacity media and better price/performance drives are brought to the market at an increasing pace. Experience has taught us that it costs more to stick with an aging technology than to take price/capacity and price/performance advantages of the newer ones. To a storage system, it means migrating data from existing media to the new kind.

A typical example is 9940A to 9940B migration. The media are physically identical. Due to different recording format, 9940B has three times capacity, 200 GB vs. 60GB. By migrating data from 9940A media to 9940B and reformat 9940A media to become 9940B, we

increased the capacity of the system by a factor of three within the same physical and fiscal constraints.

Migrating data is not intellectually difficult. What is difficult is to migrate huge amounts of data in a highly available and fully utilized system, such as ENSTORE, without interrupting services. About three years ago, ENSTORE started the data migration project. Now on its second generation, migration has become part of routine operation, and ENSTORE has retired several older generations of drives and media.

ENSTORE

To illustrate how migration works in ENSTORE, let's briefly present one conceptual view of the system. For this purpose, ENSTORE can be viewed as two parts, the back end storage system and the front end name space. See Figure 1. The back end storage system maintains complete metadata regarding the files and volumes. That is, with a single universally unique file id, the system is able to locate the file on tape and transfer its content. The front end name space provides a file system like interface for the users. Users interact with the files in the storage system as if they are dealing with a normal file system. The name space is loosely coupled with the back end storage system. In each file entry in the front end name space, it remembers the unique file id in the storage system. Currently, ENSTORE is using pnfs for the front end name space.

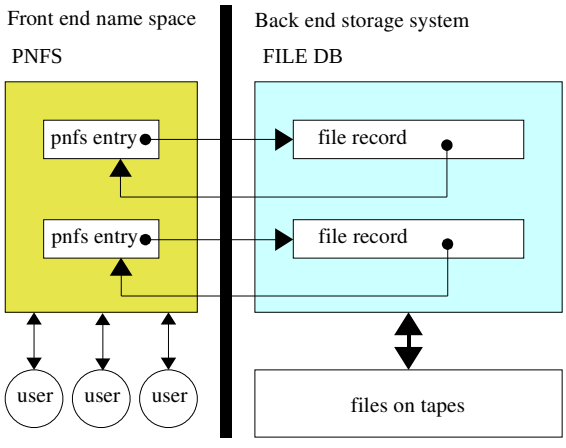

Figure 1: Conceptual View of ENSTORE

In ENSTORE, for data integrity reasons, the metadata of the files are kept in two places, the file database, in the back end storage system, and the layers in pnfs. If one is lost, it can be recovered from the other. Keeping metadata in front end name space speeds up gathering information about the file without actually accessing the back end storage system. Keeping the metadata in the back end storage system, does keep the information in case where the file entry is removed from the front end name space, allowing deleted files to be restored.

Due to the inherited nature of sequentially accessed media, it is unrealistic to reuse the tape space occupied by deleted files. If the file layout requires contiguous allocation, as of most formats, internal fragmentation will prevent such space from being reused efficiently. Even if a format allows non-contiguous allocation, the fragmented file will prevent the data transfer from progressing at its top rate. As long as non-contiguous allocation is allowed, eventually the files will be fragmented. In ENSTORE, a deleted file is only marked deleted in its metadata without being removed from tape unless the tape is recycled. Another advantage is that, if a user deletes a file by mistake, ENSTORE is able to restore the file to its original state at any time before the tape is recycled, reused and the first byte is written. The space occupied by deleted files that are confirmed to be no longer needed can be reclaimed by a process called compaction. Compaction can be naturally done through migration.

MIGRATION

The requirement and design considerations for migration process are as follows:

- File based migration – the minimal migration unit is a file. The other aggregations can be implemented on top of it. The most requested migration is batched migration based on volumes, which is implemented in terms of file migration.
- All files are always available during migration. Since ENSTORE is a highly available and heavily used system, all files have to be available even during the time when migration is taking place.
- Migration is transparent to the users. The users should not notice any difference before, during, and after the migration while accessing the files through the standard interface. No change is required to the user programs that are reading the files. To achieve this, it means reuse front end name space entry.
- No special resources are reserved for migration – all data movement for migration purposes compete with other data transfers in the system. Reserving resources for special purpose may prevent them from being used for general services hence reducing their utilization.
- Minimal impact to the system performance. Migration is not the most important task of ENSTORE. While competing with other transfers for resources, it should not affect the system. Since the files are always available before, during and after the

migration, how long it takes to migrate a file is not very critical. Migration in ENSTORE takes the lowest priority and takes advantage of system resources when they are idle.

- Progress history is kept in a persistent store. Since migration is a long running process taking lower priority in the system, anything may happen to it in its life time. Keeping progress history allows the migration process to be restarted with exactly the same conditions and it will skip what has already been done. In its implementation, a RDBM is used to keep all process history in every verifiable stage.
- Minimal administrator attention. It would be nice if the whole system is fully automatic without human interaction. However, in reality, in a storage system, any task that changes data is critical and is potentially a threat of data loss. Therefore, in operation, we still keep a person in the loop. Every thing can be prepared for that person to decide whether to kick off a migration process. Once a migration process starts, it does everything, including verification, to the end. To accomplish this, the key is a smarter error recognition and handling scheme.
- Concurrent migrations. Should there be a need to increase migration throughput, multiple migration processes can be launched at the same time withoutt interfering with each other.
- Migration is reversible – in case there is a need, every file can revert to its original state before the migration. Since every stage of the migration is logged and nothing is actually deleted in the migration process, everything can be restored.

The operational models of migration are as follows:

- User requested migration. Users request migration by specifying a list of files or a list of volumes and the administrator launches migration process(es) according to the requests.
- Routine migration by system – system recognizes conditions of certain volumes and puts them through migration. Examples of such conditions are, volumes with excessive mounts, deteriorated media, media conversion and so on. Currently, this is done through monitoring cron jobs that constantly analyse the volumes in the system and make recommendations.

IMPLEMENTATION

A file migration is done in the following 3 steps: copying the file, swapping the meta data, and verification. To illustrate how migration works, let's assume a file has a pnfs entry p1 which points to file record f1 which contains all metadata that is necessary to access the content of the file on tape t1. p1 is the only information visible to users. See Figure 2. f1 also has a pointer back to p1 which is not shown.

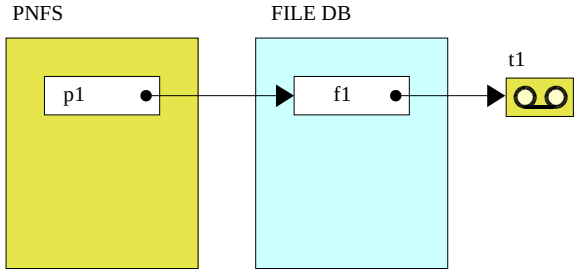

Figure 2: A file

Step 1: copying the file.

The file, having pnfs entry p1 pointing to file record f1 accessing data on tape t1, is copied to disk then to a destination tape t2, creating pnfs entry p2 and file record f2. See Figure 3.

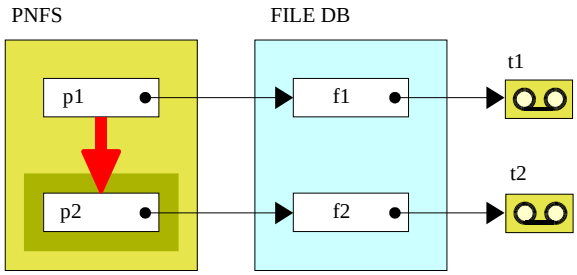

Figure 3: Copying file

f1 and f2 are two distinct files that have the same content. During copying, the read and write streams are handled by two parallel threads to increase the throughput. In each stage, the file's integrity is checked and the progress is recorded. If integrity check fails, an error is logged and reported, and migration for this file aborts.

Step 2: swapping metadata.

Let p1 point to f2. See Figure 4.

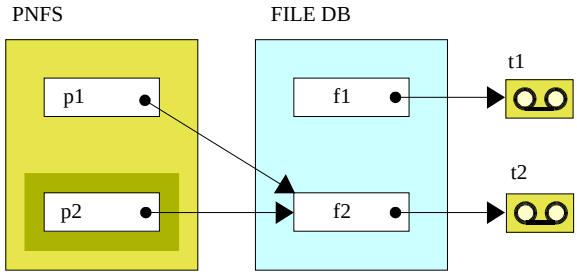

Figure 4: swapping metadata

User still sees the file as p1. Even though it is called “swapping”, the metadata are not actually “swapped”. It is only a pair of one way copies. The pointer to f2 in p2 is

copied to p1 and the back pointer to p1 in f1 is copied to f2. In case of retrying, if it is repeatedly applied any number of times, the metadata always end up in a known and correct state. The status is recorded, too.

Step 3: verification.

The file is read back through p1, in exactly the same way as the users would do, to verify its content. See Figure 5. Once the file is successfully read back, migration of this file is completed and the status is recorded and f1 is treated as a deleted file in ENSTORE. During the entire migration process, f1 is never changed and still points to the content on t1. If the verification fails, access can always be restored to f1.

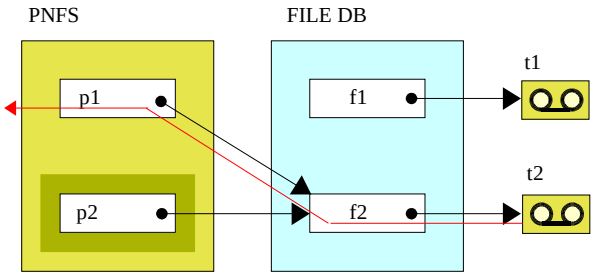

Figure 5: verification

In volume migration mode, each file goes through the same steps except the verification is deferred until the destination tape is full or until human intervention. On one hand, it is much more efficient to read through a tape sequentially than read after each write. On the other hand, every write is a potential threat to destroy the content on tape. Therefore, the final scan of the files after the tape is no longer writeable is an added assurance to the migration process. Once the verification, for single file migration, or the final scan, for volume migration, succeeds, the migration is done. When all files on the original tape are marked deleted, the tape can be recycled or removed from the ENSTORE system. In volume migration mode, any single file migration error will be logged and reported but the migration process will continue to work on the next file. A batch migration with any error will be left in an incomplete state until all the problems are resolved.

In the current implementation, deleted files may be optionally migrated. All deleted files will be migrated to a set “zombie” media which may later be restored or removed all together.

ADDITIONAL APPLICATIONS OF MIGRATION

In addition to media/format change, migration can also accomplish the following:

- media cloning. When a tape is exceeding its life time, its content needs to be copied to another tape. Traditionally, tape cloning involves reserving two drives dedicated for copying service. Media cloning can be done easily through migration without dedicated resources or human attention.
- media compaction. As mentioned, deleted files are only marked deleted in the system and the space is not reclaimed. To reclaim the space and render it in a meaningful (contiguous) way, the files need to be “repacked”. This can be naturally done through migration.
- media consolidation. Similar to media compaction, partially filled tapes may be combined on fewer tapes.

EXPERIENCE AND CONCLUDING REMARKS

ENSTORE has been running migration for more than two years. In 2004, 4557 60GB 9940A tapes have been migrated to 200GB 9940B tapes. The migration increased the capacity of the tape library by a factor of three and the media were reusable. In 2005, 1240 Eagle tapes were migrated to 9940B tapes, freeing up 1000 needed slots in the tape library. Now, migration has become a routine process in ENSTORE operations. Through migration, ENSTORE keeps surfing on the wave of technological evolution.

REFERENCES

[1] G. Oleynik and others, “Fermilab Multi-Petabyte Scalable Mass Storage System”, Proceedings of the 22nd IEEE/13th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST 2005).

[2] A. Moibenko and others, “Status of Fermilab ENSTORE Data Storage System”, CHEP 04.

[3] E. Berman and others, “Monitoring a Petabyte Scale Storage System”, CHEP 04.

[4] J. Bakken and others, “The Fermilab Data Storage Infrastructure”, IEEE Symposium on Mass Storage System, 2003.

[5] D. Petravick and others, “ENSTORE - an Alternate Data Storage System”, CHEP 98