Increasing the efficiency of tape-based storage backends

Nicola Bessone, German Cancio, Steve Murray, Giulia Taurelli

CERN – European Organization for Nuclear Research, Geneva, Switzerland

{Nicola.Bessone,German.Cancio,Steven.Murray,Giulia.Taurelli}@cern.ch

Abstract. HSM systems such as the CERN's Advanced STORage manager (CASTOR) [1] are responsible for storing Petabytes of data which is first cached on disk and then persistently stored on tape media. The contents of these tapes are regularly repacked from older, lower-density media to new-generation, higher-density media in order to free up physical space and ensure long term data integrity and availability. With the evolution of price decay and higher capacity of disk (and flash memory) based storage, our future vision for tape usage is to move away from serving on demand, random, per-file access to non-disk cached files, and to move towards loosely coupled, efficient bulk data transfers where large-volume data sets are stored and retrieved in aggregations, fully exploiting the stream-based nature of tape media. Mechanisms for grouped migration policies and priorities have been implemented, and an innovative tape format optimized for data aggregations is being developed. This new tape format will also allow for increasing the performance of repacking data from old to newer generation tape media with substantially reduced hardware costs. In this paper, we will describe the improvements for migration policies and priorities, as well as the proposed tape format and the changes which will be applied to the architecture of the CASTOR mass storage system.

1. Introduction

CASTOR, the CERN Advanced STORage manager [1], is a hierarchical storage management (HSM) system developed at CERN used to store LHC physics data. CASTOR is composed of a disk cache and a tape backend. It is is in production at CERN, ASGC, CNAF and RAL. CERN has the largest CASTOR installation which currently stores approximately 114 million files in over 20'000 tapes, amounting to 17 Petabytes of data; this will grow by some 15 PB/year once LHC starts. The CERN CASTOR installation has five main users: the media repack system and the four LHC experiments ALICE, ATLAS, CMS and LHCB.

2. Media repacking

Media repacking is the copying of data from one set of tapes to another. This activity is done for three reasons: data recovery, media defragmentation and media upgrade. Data recovery is the copying of undamaged files from partially damaged tapes. Media defragmentation is the process of not copying files that have been deleted and then liberating a tape once all of its valid files have been copied off. Media upgrade is either the reformatting of existing tapes to a higher density formats or the replacement of old tapes due to wear or due to new tape generations with higher capacities.

The media repack system is currently the most active user of the CERN CASTOR installation and is estimated to be equivalent in demand to one of the LHC experiments when in full production. These two facts make repack a realistic gauge of the performance of the CASTOR tape backend and a valuable source of information for identifying possible areas for improvement.

3. Prioritization of user requests and migration policies

Past usage of CASTOR at CERN by the four LHC experiments showed an average of 1.5 user files being accessed per tape mount. This is extremely inefficient considering it takes between 1 and 3 minutes to mount a tape. The problem is caused by the tape system trying to service each user request immediately. To increase the number of files read or written per tape mount, solutions are needed which would hold back user requests until there were enough to warrant the overhead of mounting a tape. The solution would also have to take into account the latency introduced between the moment a user made a file request and the moment when it was processed.

The users of CASTOR can be divided into two main categories: production managers and random-access (or lambda) users. Production managers are usually in charge of ensuring large-scale DAQ or reconstruction activities, as well as allocating and managing disk space for lambda user groups. Production managers usually understand how to use CASTOR efficiently because they know its internal setup. They naturally process more files per tape mount than random-access users because they know the contents of the tapes and the cost of mounting a tape. On the other hand, end users are not expected to have this knowledge and rely on CASTOR to transparently handle recalls and migrations for them in an efficient way.

The challenge of increasing the numbers of files read or written per tape mount is addressed by implementing tape request access control and priorities, and recall/migration policies. These are described below.

3.1 Tape request access control

Whenever possible, end users should access data which has already been staged on disk rather than recalling it from tape. This is particularly relevant as their files may be scattered across many tapes and therefore high mount rates can be the consequence of users skimming over data sets not yet staged to disk. End users should be encouraged to work with production managers, which become responsible for deciding which overall data sets are to be analyzed to users. Production managers therefore what files need to be staged to or removed from disk pools. They can anticipate and group requests of multiple lambda users in an efficient “freight train” approach and thus minimize the number of tape mounts.

In order to facilitate this, the CASTOR stager was extended to support access control for defining who is allowed to recall files from tape. This is implemented in the form of black and white lists, defining what users and/or groups are allowed to, or denied from, recalling files to disk pools. A lambda user will typically lack authorization to access tape, while production managers will be granted such rights and will be able to recall files on behalf of their lambda users.

3.2 Priorities

It is not always possible or feasible to have all tape recall operations driven by production managers. In addition to access control, the possibility of defining per-user and/or per-group based priorities for tape mount requests has been implemented. In older CASTOR versions, tape mount requests are first sorted by type (write requests always have higher priority than read requests), and then by submission time. This has the inconvenience that any user requesting to recall large amounts of data scattered over many tapes could effectively block other users. Consequently, the CASTOR tape drive scheduling system was extended to allow defining per-user and per-group priorities so that production managers can be granted higher priority than lambda users. In addition, it is possible for tape operators to define a high priority for a specific tape. This is particularly useful when trying to quickly recover the contents of a tape e.g. after it has been damaged.

3.3 Recall and migration policies

The base concept of both recall and migration policies is holding back the migration and recalls depending on the amount of data and elapsed time. This way, the total count of tape mount operations

should be minimized for both reads and writes. In CASTOR, a framework was developed which allows specifying policies using python-based plug-ins which can be tailored to the needs of each specific stager. As an example, a typical migration policy will not start a new stream to tape until a reasonable amount of data has accumulated and/or a sufficient amount of files is waiting for migration. Streams can also be forced to be started in random intervals, in order to avoid an indefinite postponement of the migration of smaller data chunks.

4. New tape format

The media repack system behaves as the perfect user with respect to tape mounts. It mounts a tape to be repacked and then proceeds to recall all of the user files from that tape. Likewise when a destination tape is mounted, it is filled with files until it is full. Monitoring the media repack system has shown that the current CASTOR tape format is introducing a significant overhead of up to 9 seconds per user file - independent of its size. This is due to the tape data format used in CASTOR.

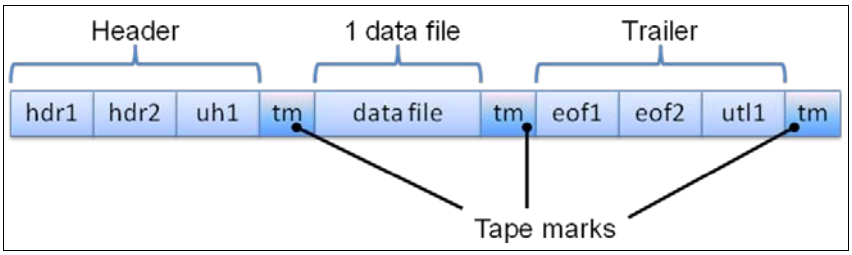

Figure 1: AUL tape format. Each data file is wrapped between a header and a trailer separated by tape marks.

The tape data format currently used by CASTOR is ANSI AUL [2] and was set in place in the 1990s. As figure [1] shows, each CASTOR file consists of three tape files: A header and a trailer file containing metadata information, and the actual data file which is embedded between the header and the trailer. These files are separated by end-of-file markers called tape marks. Writing a tape mark tells the tape drive not only to close the file, but also implies a sync operation: It flushes all buffered data and physically writes it to the tape media, so to avoid data loss. While writing a tape mark after every file ensures high reliability, it is a costly operation, particularly when dealing with small files. In fact, writing of tape marks is the most dominant factor in the writing of the header and trailer metadata. A total of ~5-9 seconds is taken per file to write the three tape marks and the metadata (~2-3 seconds per tape mark).

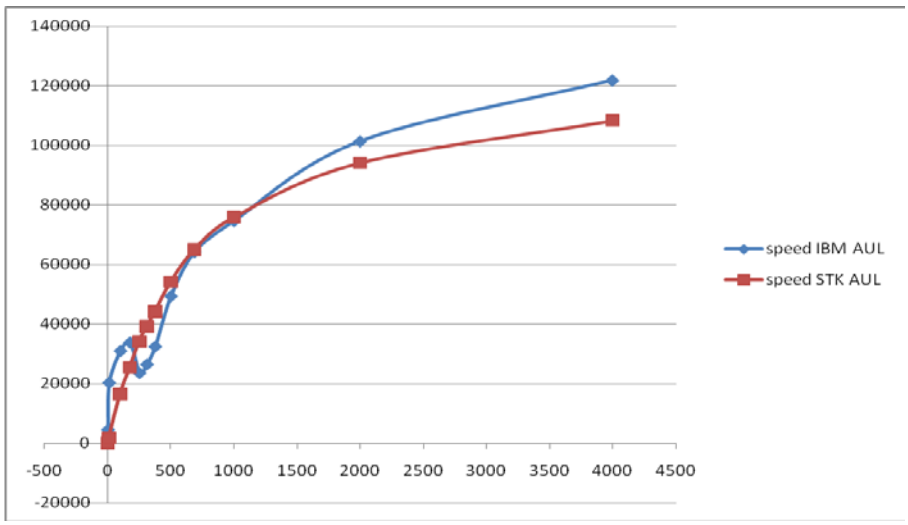

Figure 2: Drive speed (expressed in KB/s) as a function of file size (expressed in MiB) when using the AUL tape format, on IBM and Sun/STK drives used in production at CERN.

Figure [2] clearly shows that the smaller the files written to tape are, the more the tape mark writing overhead acts as a performance penalty. Optimal performances are reached with file sizes above 2GB. While the average capacity of a tape cartridge has significantly increased since the 1990s (from 5-10 GB to nowadays 1TB), the quantity of large physics data files has not increased by the same magnitude and the majority of files are still very small (see figure [3]). Given this observation it was decided to develop a new tape format that had a reduced overhead per user file.

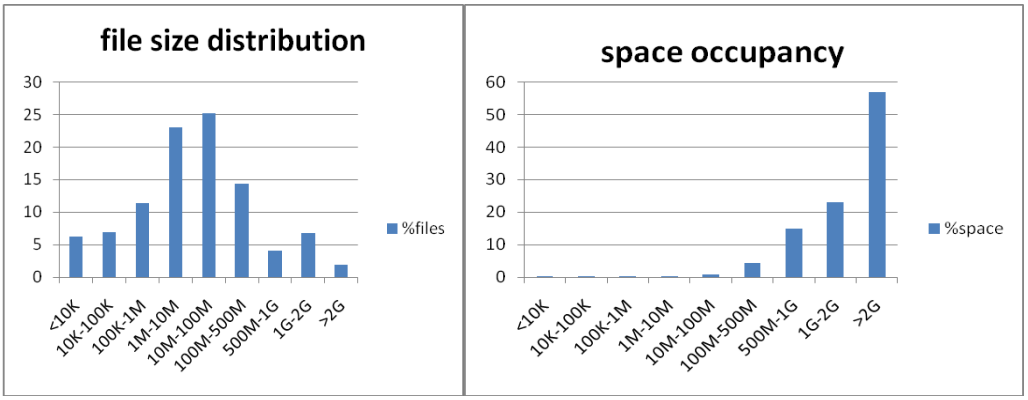

Figure 3: size and space occupancy distribution for CASTOR migrated files at CERN

A new tape format (ALB, ANSI Label with Block format) is being developed for CASTOR, with the aim to increase efficiency and redundancy. The new ALB format is based on the ANSI AUL format. As figure [4] shows, the payload inside each AUL data file will consist of an *aggregation* of multiple CASTOR files, in order to reduce the number of tape marks and therefore achieve high write speed also with smaller files. The incoming stream (list of files) to be migrated will be aggregated on-the-fly to a configurable maximum total size (e.g. 10GB) and/or configurable maximum number of files (e.g. 1000 files). If a file exceeds the maximum total size it will be written in a separate aggregation consisting of that single file. On hardware with efficient tape mark handling, the number of files per aggregation can be decreased. As there is no per-file sync any longer, the CASTOR architecture needs changes to handle all files belonging to an aggregation in an atomic way (individual files can only be considered to be successfully committed to tape once the complete aggregation has been written).

Aggregations are not a static concept but are dynamically (re-)created on every migration; files may be rearranged into different aggregations e.g. after repacking of a tape. For each file within an aggregation, its exact position on tape is recorded in the CASTOR directory service. This allows files within an aggregation to be recalled individually without having to read the whole aggregation.

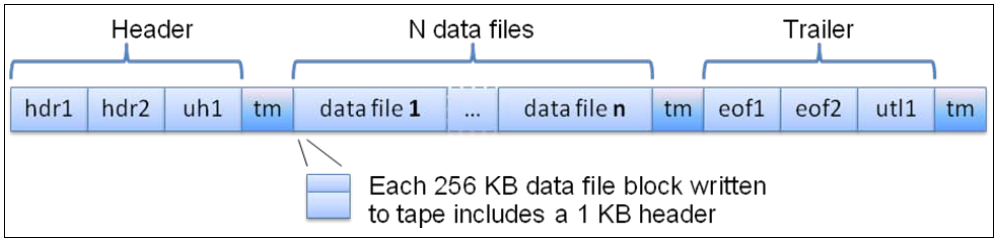

Figure 4: New ALB format - embedding an aggregation of multiple CASTOR files inside a single AUL data file.

Every file within an aggregation is split into fixed-sized blocks (e.g. 256KB). Every block contains a 1KB header for self description. This header provides metadata information such as:

- Information about the tape: identifier, density, serial number
- Information about the drive: name, serial number, firmware version, tape server host name
- Information about the file: name, CASTOR file identifier, size, creation time, checksum type and value
- Information about the block: size, block number within the file, total number of blocks, checksum

4.1 Expected performance improvements

Based on the characteristics of CASTOR data at CERN (figure [3]), we have calculated the time it would take to migrate all existing CASTOR files to new media, using the current CASTOR AUL format and the proposed block-based ALB format, and based on 1GB/s network connectivity between disk and tape servers. We also evaluated the performance of a simplified version of the AUL format with only one tape mark per file (suppressing tape marks between headers and trailers), which we named ALC (for "ANSI Label Compact"). The results are depicted in figure [5]. For Sun/STK drives, there is a 70% gain over AUL when using the ALB tape format, and of 47% using the ALC format. The performance gains are less large on IBM drives (46% for ALB, 31% for ALC) as they provide a built-in optimisation mechanism for small file writing. It should be noted that these performance improvements can only be achieved if there is no other contention (e.g. network or disk server saturation).

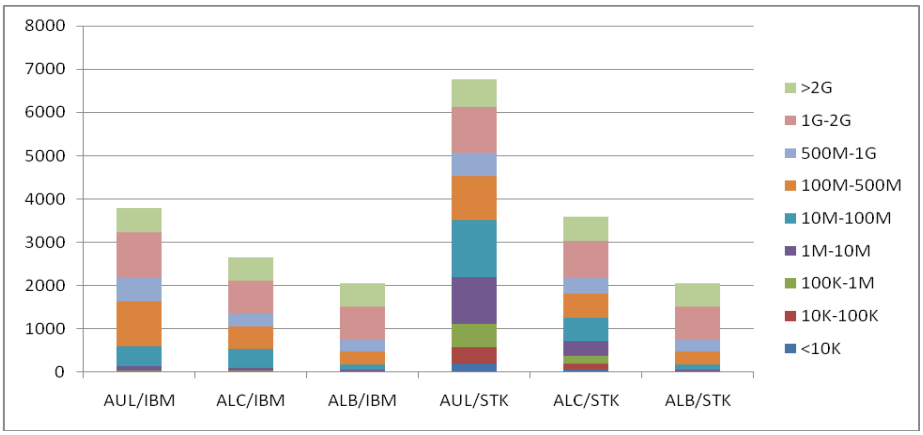

Figure 5: Time to migrate (in days) all existing CASTOR files at CERN, using three different formats, on IBM and Sun/STK drives.

4.2 Changes to the CASTOR architecture

In order to use the new tape file format the CASTOR architecture has to be modified to include one or more components to pack user files into aggregations. Figure [5] shows the area of the existing CASTOR architecture that needs to be modified.

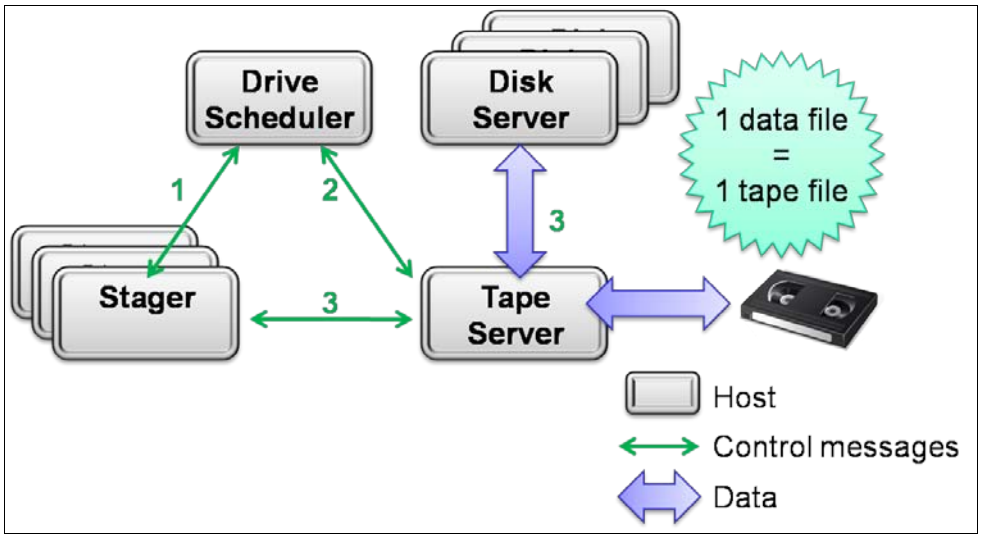

Figure 6: Current CASTOR tape architecture (simplified)

The above components work together to migrate and recall user files to and from tape. In step 1 the stager requests a tape drive, in step 2 the drive scheduler allocates a free drive and in step 3 the data is transferred to/from disk/tape based on the list of files given by the stager. Note that in the current architecture one user file maps to one tape file.

Figure [6] shows the proposed new architecture. Two new components will be added: the tape gateway and the tape aggregator. The tape aggregator will pack user files into aggregations on the tape server as they are read from the disk servers. The tape gateway will control the tape aggregator, deciding which files go into or come out of an aggregation. Both will deal with aggregations atomically, i.e. the stager is informed if either all files belonging to an aggregation are successfully written to tape, or the whole aggregation is discarded in case of failure.

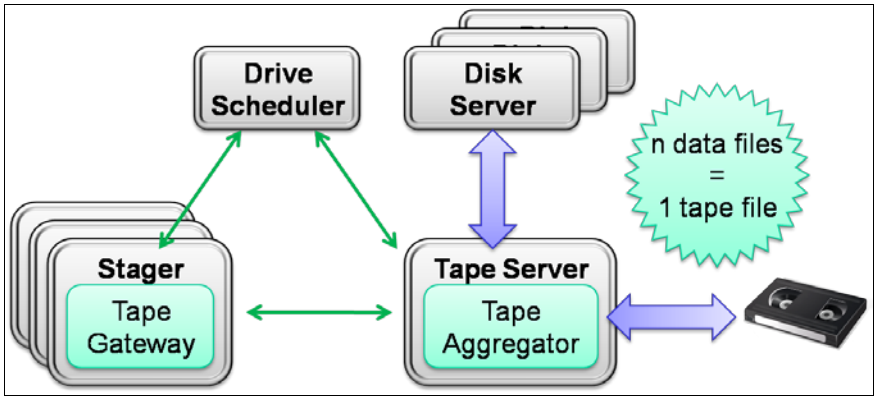

Figure 7: New CASTOR tape architecture

The tape gateway and aggregator components will work alongside the existing stager and tape server components allowing for a smooth transition to the new architecture. They will also continue to support the current AUL tape format both in reading and writing.

4.3 Related work

Unfortunately, there is little public technical information on how other mass storage systems address the tape mark performance problem. The HPSS collaboration ([3]) has included in their recent HPSS release 7.1 a native architecture for small files, with the capability to aggregate multiple small files into an aggregated tape segment when they are migrated from disk to tape ([4]). Backup systems such as IBM's TSM [5] address the problem by using disk caches where small files are aggregated into larger ones, before migrating them to tape. This however requires a separate disk cache layer dedicated to tape, and adds delays in terms of latency before the data is safe on tape.

6. Current status and future work

The tape request access control and mount prioritization mechanisms, as well as the recall/migration policy framework, have been deployed at CERN and at the Tier-1 CASTOR instances. However, recall policies are not yet used at CERN. This is mainly for two reasons: On one hand, recall policies are not appropriate for interactive users, who should be accessing data which already has been staged. On the other hand, they cause inefficiency on batch systems as jobs may have to wait for the data longer.

The components for supporting file aggregations and the new ALB format are being prototyped. A full release is planned for November 2009, with a gradual deployment over 2010. In parallel, we will continue to investigate how the concept of aggregations further exploited by the CASTOR stager system. The aggregation format is internal to the tape layer and not visible from higher levels, which still continue to see individual files. In particular, we will investigate how to group logically related files which are likely to be accessed in common. This could be achieved e.g. by defining migration policies which group files belonging to the same namespace directory into one tape stream. In that case, aggregations of logically related files could be read out completely with appropriate recall policies.

References

[1] CASTOR homepage. http://cern.ch/castor
[2] Using Magnetic Tape Labels File Structure. Document SC26-4565-01. International Business Machines Corporation, 1991.
[3] High Performance Storage System (HPSS). http://www.hpss-collaboration.org/hpss/index.jsp

[4] HPSS Roadmap 2009. http://www.hpss-collaboration.org/hpss/about/HPSSRoadmap2009.pdf
[5] IBM Tivoli Storage Manager (TSM). http://www-01.ibm.com/software/tivoli/products/storage-mgr/

Acknowledgements

Many thanks to the CERN Tape Operations team for their valuable ideas, and their input on tape usage and statistics used for this paper.