The GridKa Tape Storage: various performance test results and current improvements

Haykuhi Musheghyan ,[1] *Andreas* Petzold ,[1] *Andreas* Heiss ,[1] *Doris* Ressmann ,[1] and *Martin* Beitzinger[1]

[1]Karlsruhe Institute of Technology

Abstract. Data growth over several years within HEP experiments requires a wider use of storage systems for WLCG Tiered Centers. It also increases the complexity of storage systems, which includes the expansion of hardware components and thereby complicates existing software products more. To cope with such systems is a non-trivial task and requires highly qualified specialists. Storing petabytes of data on tape storage is a still the most cost-effective way. Year after year, the use of a tape storage increases, consequently a detailed study of its optimal use and verification of performance is a key aspect for such a system. It includes several factors, such as performing various performance tests, identifying and eliminating bottlenecks, properly adjusting and improving the current GridKa setup, etc.
At present, GridKa uses dCache as the storage system in frontend and TSM as the tape storage backend. dCache provides a plugin interface for exchanging data between dcache and tape.
TSS is a TSM-based client developed by the GridKa team. TSS has been in production for over 10 years. The interaction between the GridKa dCache instance and TSM is accomplished using additional scripts that can be further optimized to improve the overall performance of the tape storage.
This contribution provides detailed information on the results of various performance tests performed on the GridKa tape and significant improvements of our tape storage performance.

1 Introduction

Grid Computing Center Karlsruhe (GridKa) is the German WLCG tier-1 center in Karlsruhe that supports four main LHC experiments (Alice, Atlas, CMS, LHCb). It is also the German regional grid computing center for non-LHC HEP experiments (Belle2, BaBar, Auger, Compass). All these experiments each year produce terabytes and/or petabytes of data that need to be safely stored, reliably and fast accessed on demand. Storing all these data on disk storage system is very costly, therefore using tape storage system is still the most cost-effective solution.

It has a multi-layer and complex infrastructure. It consists of many components that must be used in the most efficient and optimal way in order to provide highly available and reliable storage system. For a long time, GridKa uses dCache[1] as a disk and IBM Spectrum Protect (IBM SP)[2], formally known as Tivoli Storage Manager (TSM) as a tape storage system. GridKa uses only one Oracle SL8500 library for tape storage system. As a tape client, GridKa uses its own developed software, the so-called Session Server-client (TSS-client)[3]. Compared to the dsmc-client[4], which is the official IBM SP client, TSS-client allows multiple recall/migrate queues, offers a control over queue priorities and storage classes, sorts files located on tapes, masks transient IBM SP errors, optimizes simultaneous access to the same tape cartridge, reports mount times and tape access efficiency and has a simple command line interface to the IBM SP database. TSS-client is using IBM SP API and is located between dCache disk pools and a IBM SP as displayed in Figure 1.

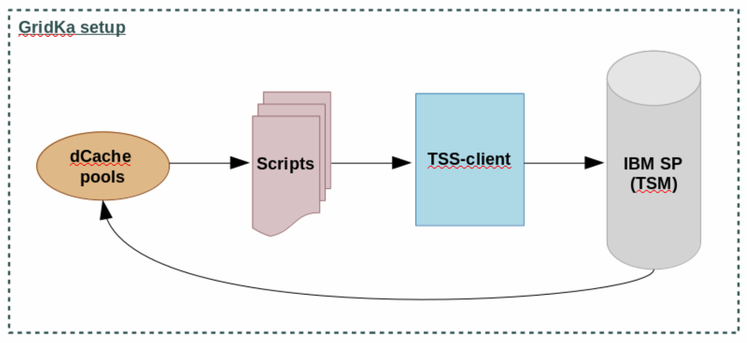

Figure 1: GridKa storage system: Interaction between dCache pools and IBM SP

After the first recall test within ATLAS Distributed Computing (ADC) Tape Carousel in 2018, various local tests were performed on GridKa tape system. The main goal of these tests is a detailed study of the current tape setup, identifying and eliminating bottlenecks, properly adjusting and improving the current setup that will lead to increase of the overall tape storage system performance.

All main tests performed on GridKa tape are listed below in this paper.

2 Various performance test results

2.1 Recall files directly and indirectly from tape

One of the key aspects for the storage systems is scalability. For tape storage systems in general, and in particular for the GridKa tape setup, the request queue length plays an important role and has a significant impact on the overall performance of the tape storage system. The longer the queue, the better the sorting of files by location on the tape and, therefore, more files can be recalled from the same tape with one mount. Consequently, less tape remounts and less time is spent finding the location of the file on the tape and, as a result, better performance.

The Figure 1 displays the current production setup of GridKa storage system. This setup has a certain limit on the maximum number of simultaneous requests, which is 2.000, due to limitation on the memory usage. Thus, dCache maintains an active queue with maximum 2.000 requests. A trivial change of this number ends up with the crash of the dCache pool.

During the test, it turned out that a standalone TSS-client is capable of serving up to 30.000 simultaneous requests. TSS-client builds and sorts the queues for the tape cartridge based on the provided file list.

Figure 2 displays the differences between the tape recall throughput when the maximum number of simultaneous requests was 2.000 and 30.000.

On the upper part of the Figure 2 is the result of the first recall test within ADC Tape Carousel in 2018 with 2.000, and on the lower part is the local test result with 30.000 simultaneous requests.

During the local test, all parameters (the number of used drives, file list, etc.) remained the same as during the ADC Tape Carousel test, except for the maximum number of simultaneous requests. As a result of these tests, the average tape recall throughput ≈400MB/s and ≈600MB/s was achieved, respectively. This is already a good achievement.

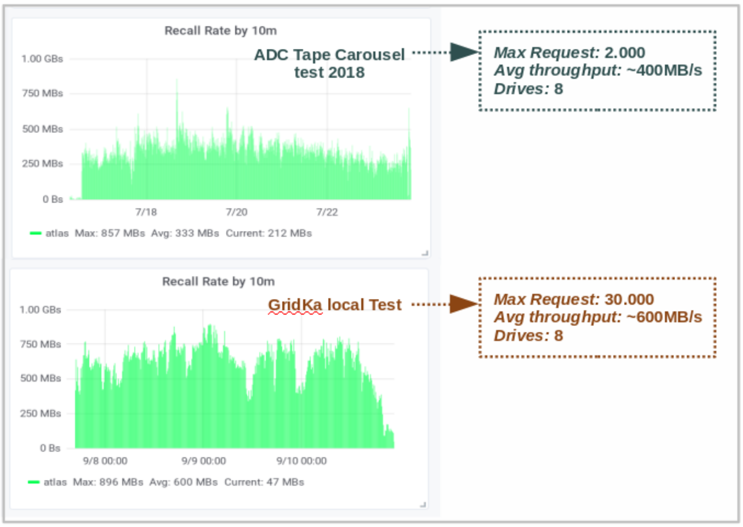

Figure 2: The difference in tape recall throughput when the maximum number of simultaneous requests was 2.000 and 30.000

More details about GridKa tape setup can be found in the following article [5].

2.2 Duplicated files on tape

During one of the local tests, it was detected that the same file was written to the same and different tapes several times. This is a classic situation when dCache tries to write a file to a tape, but for understandable reasons (for example, tape system is not available, the dCache pool is down at the time the store request to the tape is completed or any kind of failure when it is impossible to guarantee that the file was successfully written to tape) it does not get an appropriate return code, which indicates that the file was written to tape successfully. In this case, dCache will keep trying to write the file to a tape again and again until it gets back the appropriate return code as an indication of a successful write to the tape. There was a case when a single file was written to tape 96 times. As a result it, ended with 96 copies on 5 different tapes.

The presence of duplicates in a tape storage system is not good in general, as it affects the overall performance of a tape storage system. Usually, the presence of duplicates involves the distribution of files over many more tapes, than in a case of non-duplicates. Typically, the first copy of a file goes to tape along with other files that belong to the same dataset. If the file is written again later, it will be on another tape, therefore increases the number of tape mounts during the recall of the dataset. Tapes will be mounted over and over again, although only for recalling a single or a few files from the particular tape. TSS-client will try to recall only the latest written version of the file, which is not optimal. This is very noticeable during any recall campaign, where terabytes or even petabytes of data needed to be recalled from a tape storage system.

This was a design decision, better to have a file written more than once on a tape than not to have it at all. The approach solves the issue of data loss.

For the test, a list of files from 10 randomly picked up tapes was generated. This list has a total of 27.079 files. TSS-client started to sort and build queues based on the given file list. It was expected to have a total of 10 queues, one queue for each tape file list. As a result, there were 58 queues built by TSS-client instead of 10 as displayed in the Figure 3. This means, that for recalling 27.079 files from the tape storage system, 58 different tapes will be mounted.

IBM SP database stores information, about files that were written to tape and TSS-client builds queues based on the last active copy of a file.

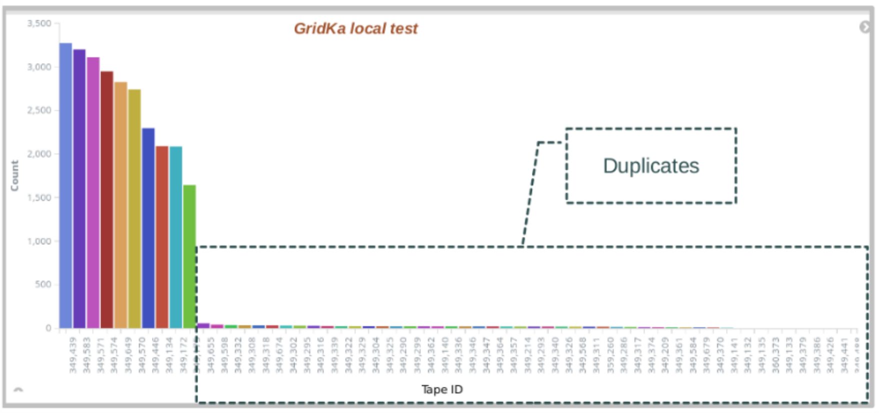

Figure 3: File distribution on different tapes: Duplicate files on tape

This is a convincing sign that some files were written several times, on the same or on different tapes.

In our current setup, all duplicates were detected and removed from the IBM SP database. To remove duplicates, the oldest version of the file was recalled from the tape, then the checksum of the file was compared to verify the correctness of the file, and after that all newer versions of this file were removed from tape.

2.3 Recall the entire tape cartridge

Another bunch of tests were performed on the GridKa tape system to measure the difference in tape recall performance when recalling the entire file list and individual files from tapes. The test was done several times with a different number of tape cartridges and tape drives, and the results were similar. The results of two diverse tests using two different file lists (a file list of the entire tape and a file list of individual files from the same tape) with a different number of tapes (6 and 10 tapes) are presented here in this Section.

This test displays the overall tape recall throughput when the entire file list of a single tape cartridge was recalled. Figure 4 displays the result of recalling the file list of the entire

6 (a total of 15.536 files) and 10 (a total of 25.997 files) tape cartridges with the number of tape drives 6 and 10, respectively. Within the current GridKa tape setup, the maximum recall throughput was ≈900MB/s and ≈1,5GB/s for 6 and 10 tape drives, respectively, which corresponds to ≈150MB/s per tape drive. The average file size during the test was 2.0GB.

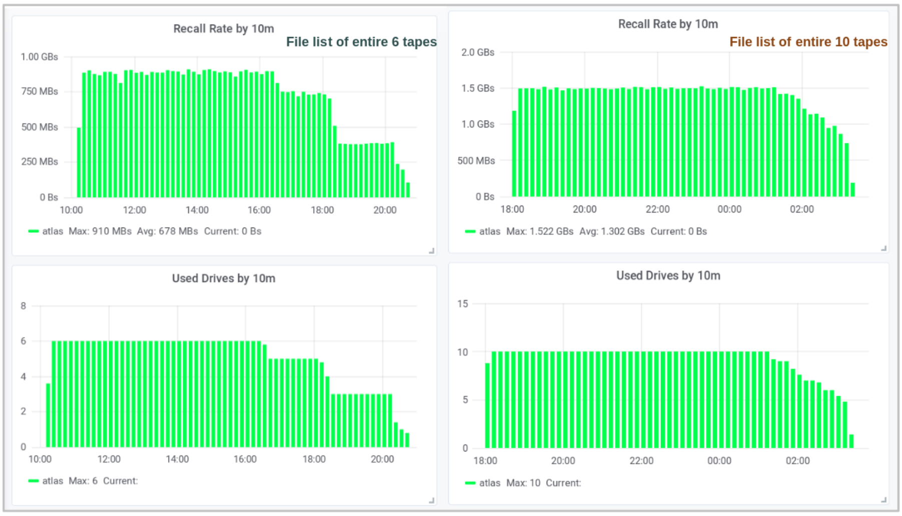

Figure 4: Read the file list of the entire tapes (6 and 10 tape cartridges)

Figure 5 displays the result of recalling individual files from 6 (a total of 12.787 files) and 10 (a total of 21.071 files) tape cartridges. In this case, the maximum recall throughput was ≈850MB/s and ≈1,2GB/s for 6 and 10 tape drives, respectively, or an average of these two maximums ≈130MB/s per tape drive. The average file size during the test was 2.0GB.

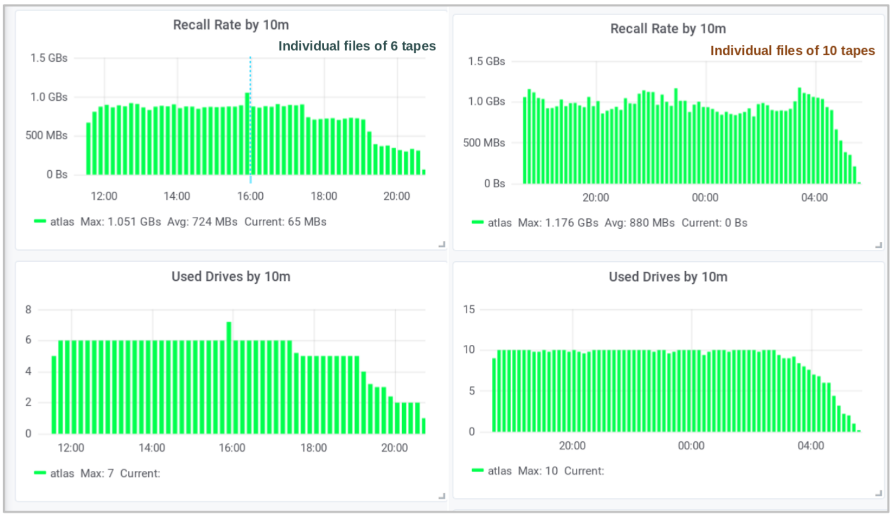

Figure 5: Read individual files from tapes (6 and 10 tape cartridges)

Recalling the entire file list of a single tape cartridge is much faster, than recalling individual files from a single tape cartridge as files are recalled one after another in sequence without spending any time on finding an accurate file position on the tape cartridge.

2.4 Recall/migrate large files from/to tape

Two other tests were performed on the GridKa tape system. The first test case is when small and large files were migrated or written to tapes and the second test case is when small and large files were recalled or read from tapes. The purpose of these two tests is to measure the difference in tape migrate and recall performance when small and large files were written and then read to and from tape. As small files, it was considered less than 1.0GB of file size and for large files the file size was larger than 9.0GB.

Figure 6 displays the first test case. On the left side of Figure 6 displayed the result of writing large files to tape using only one tape drive. In this case, the maximum migrate throughput was ≈175MB/s and the average migrate throughput was ≈110MB/s for a single tape drive, while on the right side of Figure 6 small files were written to tape. Here the maximum migrate throughput was only ≈775MB/s for 8 tape drives (see time period between two dashed lines). It corresponds to ≈95MB/s, and the average migrate throughput was ≈75MB/s for a single tape drive.

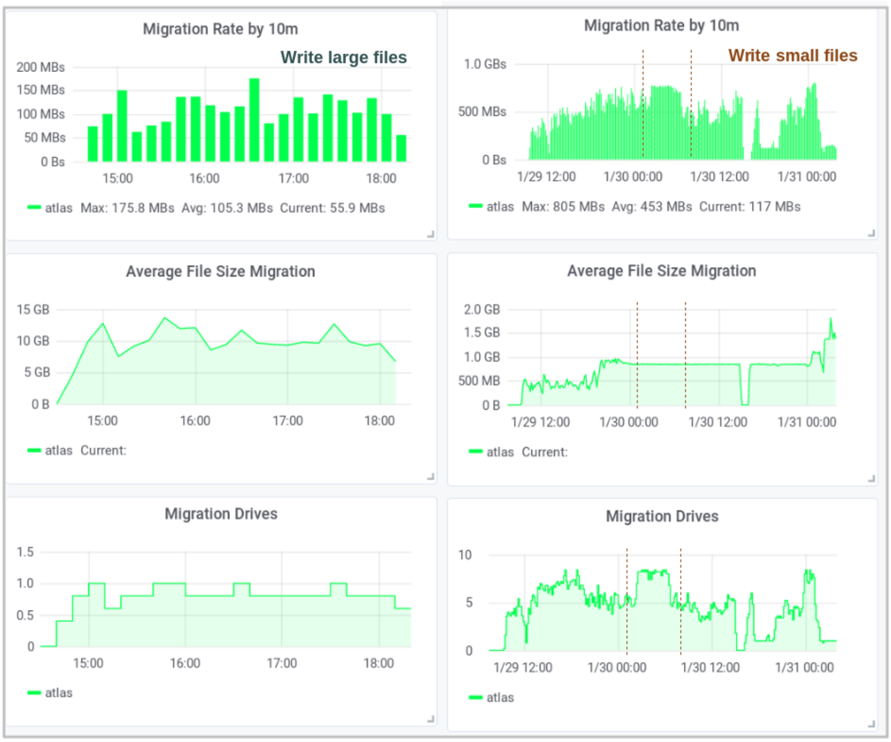

Figure 6: Writing small (less than 1.0GB file size) and large (larger than 9.0GB file size) files to tapes

Figure 7 displays the second test case. On the left side of Figure 7 displayed the result of reading large files from tape using only one tape drive. In this case, the maximum recall throughput was ≈210MB/s achieved and the average recall throughput was ≈160MB/s for a single drive, while in the right side of the figure, completely different results are presented for small files. The maximum recall throughput was ≈400MB/s for 8 tape drives, which corresponds to ≈50MB/s, and the average recall throughput is ≈30MB/s for a single tape drive.

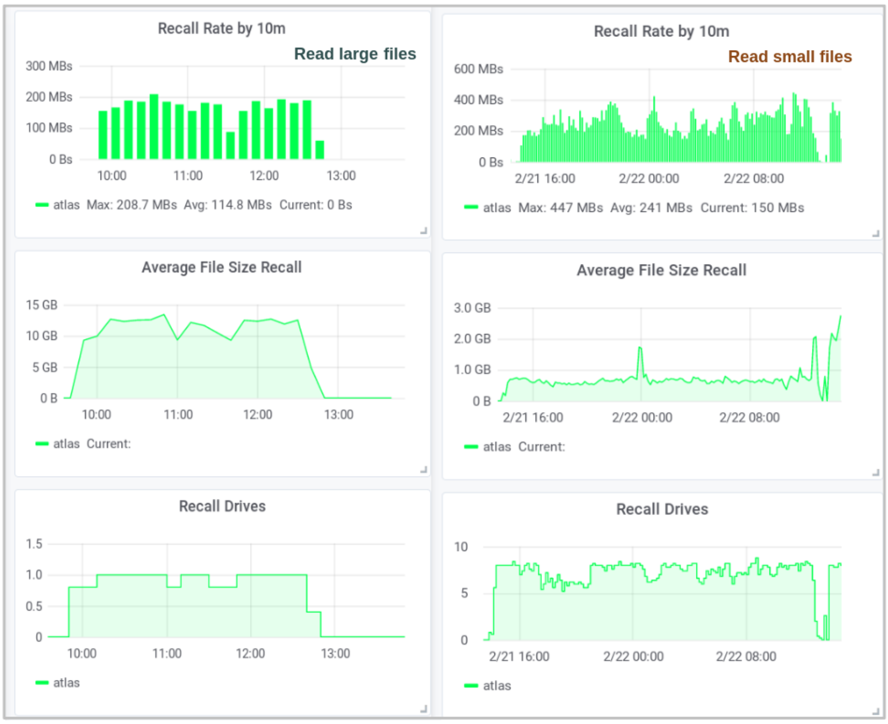

Figure 7: Reading small (less than 1.0GB file size) and large (larger than 9.0GB file size) files from tapes

The overall tape throughput depends on many factors. One of these factors is the file size, which directly affects the overall throughput of the tape storage system. The larger the file size, the better in both cases for migrating a file to tape and for recalling a file from tape.

3 Summary and conclusions

Starting from July 2018, various performance tests were performed on the GridKa tape storage system. During these tests, three main bottlenecks were detected and these are:

- Increase the number of simultaneous requests from 2.000 to 30.000. Increasing the number of simultaneous requests guarantees more than ≈50% of improvement in the overall tape recall throughput.
- Remove duplicated files from tape. Removing duplicated files from tape significantly reduces the number of tape mounts and therefore affecting the overall tape recall throughput in a positive way.

- Write and then read large files (larger than 9.0 GB). File size is crucial. Writing and then reading large files increases the overall tape recall throughput ≈3.0 times.

These bottlenecks can be eliminated both on GridKa and on the Virtual Organization (VO)[6] side. Bottleneck elimination, is not an easy task, and requires certain effort in terms of new software development, new installations and configuration changes, network extension, etc..

In the test setup, at least ≈50% better performance of the overall tape storage system was achieved, which is not the case for the production setup in the current time frame.

Having minimum 50% of improvement in the overall tape storage system throughput is already an significant enhancement for GridKa, taking into account the complexity and multi-layer infrastructure of it.

Currently, an intensive work is ongoing towards bringing above mentioned test results into production.

References

[1] dCache storage system, https://www.dcache.org/

[2] IBM Spectrum Protect (IBM SP), https://www.ibm.com/support/knowledgecenter/de/SSEQVQ/landing/welcome_sseqvq.html

[3] J. van Wezel, D. Ressmann, "TSM as tape storage backend for disk pool managers", HEPiX Spring Meeting, Roma, I, April 3-7, (2006)

[4] dsmc-client, https://www.ibm.com/support/knowledgecenter/SSEQVQ_8.1.6/client/c_start_cmdline.html

[5] Ressmann, Doris, et al., "The GridKa Tape System: status and outlook.", EPJ Web of Conferences. **Vol. 214.** EDP Sciences, 2019.

[6] Andreeva, J., Boehm, M., Gaidioz, B. et al. Experiment Dashboard for Monitoring Computing Activities of the LHC Virtual Organizations. J Grid Computing **8**, 323–339 (2010). https://doi.org/10.1007/s10723-010-9148-x