Using TSM to create a high-performance tape connection

Dr. D. Ressmann, Dr. S. Halstenberg, J. van Wezel
GridKa, Forschungszentrum Karlsruhe, Germany

Abstract

At GridKa an initial capacity of 1.5 PB online and 2 PB background storage is needed for the LHC start in 2007. Afterwards the capacity is expected to grow almost exponentially. No computing site will be able to afford keeping this amount of data in online storage, hence a highly accessible tape connection is needed. This paper describes a high-performance connection of the online storage to an IBM ® Tivoli Storage Manager (TSM) environment.

The performance of a system does not only depend on the hardware, but also on the architecture of the application. The scenario we are describing distributes its files over a large number of file servers, storing their data with the help of a proxy node to tape. Each file server can restore the data independent on the file server which has stored the data originally. Furthermore with the LAN free connection to the tape drives the data transfers bypass the TSM server which otherwise would be a bottleneck. The system is completely transparent to the user.

INTRODUCTION

The Grid Computing Centre Karlsruhe [1] (GridKa) is part of a large science and engineering research institution "Forschungszentrum Karlsruhe [2]" and serves as a Tier-1 centre for all large hadron collider (LHC) experiments (Alice, Atlas, CMS and LHCb). Furthermore GridKa already serves four HEP experiments: Babar, CDF, Compass, and D0. See also [3].

All these experiments produce different types of data with a different life time and access pattern. To keep files on disk which are seldom used and have to be stored for a long time is a very expensive solution, as such storage is typically divided into online and background storage. However a fast and reliable access to the background storage is important, since the data flow from the Tier-0 to the Tier-1 sites have to be guaranteed with a minimum throughput rate. Additionally at the same time of this data flow other access pattern have to be supported for reconstruction and simulation.

As for our planning numbers already in 2006 we provide 800 TB disk and 900 TB tape storage. These numbers increase until 2010 to a disk storage of about 11 PB and a tape storage of 13 PB. This paper describes a possible solution to connect the online storage with high performance to the background storage.

SETUP

Service Challenge

The Service Challenges [4] serve to test the infrastructure needed for full LHC data taking and analysis. It concentrates primarily on the robust file transfers from the Tier-0 centre CERN to all Tier-1 centres which is needed to support the distribution of the raw data. In subsequent steps also the transport of the reduced datasets from Tier-0 to Tier-1, between the Tier-1's and between the Tier-1's and the Tier-2's will be tested. Apart from the bare data transport the challenges will take all necessary software components into account as they become gradually available. Operational aspects will also need to be understood as we upgrade in steps from test systems to a production service. During LHC data taking a 24/7 service has to be provided and the implications in terms of resources (manpower) have to be understood. As the tests gradually become more like a service the LHC experiments will use the production infrastructure for tests of their software and for their data challenges.

During a throughput phase of these service challenge tests we transferred more than 200 MB/s from CERN to GridKa [5] disk to disk. For example during the test on 23rd of January 2006 we used 8 gridFtp doors and 4 pool nodes. On each of these pool nodes we have installed two pools with each 1.4 TB disk space. The disks are mounted over a storage area network (SAN). Since the Service Challenge tests include the whole chain of data analysis, the tape connection is included in these tests. At Forschungszentrum Karlsruhe we have experience using TSM as our tape connection.

Tivoli Storage Manager

TSM is a client-server software product developed by IBM ® to create and manage backup and archives or important data. Ideally a TSM server has a disk storage used for a database containing all information about the location of a file on tape and the utilisation of a tape. Furthermore a disk storage can be used as a buffer for the files before they are written to tape. Additionally the TSM server sends control messages to a tape robot. The tape robot has a picker arm which moves the tapes from a shelf into the tape drives.

Traditionally GridKa users created their archives with the TSM command called "dsmc". This method had some drawbacks since TSM has originally been developed for backup use, and the archive functionality is somehow awkward. If a user starts an archive for a big directory structure

and several TB of data, this archive will take a long time. If during this time some complications occur, e.g. the connection to the TSM server is lost, there exist no automatism to check which files have already been archived and which one haven't. It would be for the user to check manually every file. Furthermore if the user writes a new file within a directory structure it will not automatically been archived. This disadvantages could have been overcome creating own scripts to access TSM manually or by using the Hierarchical Storage Management (HSM) functionality of TSM. However using dCache for this connection, solves this problem and has many additional advantages.

dCache at GridKa

At GridKa we have decided to use dCache [6] as our storage element, since it has an exchangeable HSM connection and is accessible via SRM [7]. dCache keeps track on every file written to tape and schedules the archive automatically till it succeeds by creating a new TSM session for every file. This whole procedure is completely transparent for the user. Among other advantages dCache has different access modules and gives the opportunity to access the storage via grid tools like SRM [7].

The files in our dCache system are accessed with different patterns and from different locations. Therefore we have to partition our dCache setup as shown in figure 1. Because of security reasons we differentiate between internal and external pools. The idea is to create the connection between the internal and external pools only through a tape medium.

Over a dedicated line, raw data are written from CERN with a guaranteed high bandwidth to tape at each Tier-1 centre. One of the main purposes of a Tier-1 centre is to store these raw data and make them available for further use. Any interruption from user jobs has to be strictly prohibited. Therefore we create write-only pools for these raw data and do not allow any pool-to-pool transfers or read-access from these pools. The only purpose of this set of pools is to write with a constant stream to tape. To access these files afterwards, they have to be restored from tape to either an internal or external read pool.

Physicists are reconstructing the raw data and produce for example Event Summary Data (ESD) using our computing cluster. Newly created files like ESD have to be stored on a few separate write pools with tape connection. Lots of user jobs are reading the ESD data in random order and analysing them to create data with a relatively short lifetime. As such these files are stored on disk-only pools. Either these disk-only files are not accessed from outside GridKa, or special security issues have to be considered. Contrary for the ESD data it is unproblematic to make them available for other sites e.g. other Tier-1 or Tier-2 sites, since they are written to tape and the tape library will be our joint between the internal and external network.

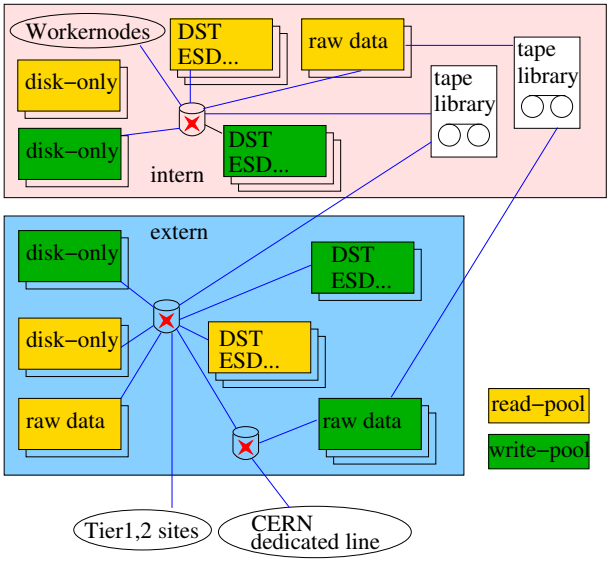

Figure 1: dCache Setup

TAPE CONNECTION

A simple tape connection

One solution for a dCache tape connection using TSM [8] is to use a TSM server as the connection to the tape drives as can be seen in figure 2. In this scenario a program tsmcp which is interfaced to dCache is used. It uses the TSM application programming interface (API) and starts and closes a session for each store or retrieve action. The dCache pool nodes connect to the TSM server, which then writes the files to tape. This solution does not scale, however it is perfectly fine for small sites which only need a tape connection less than 100 MB/s.

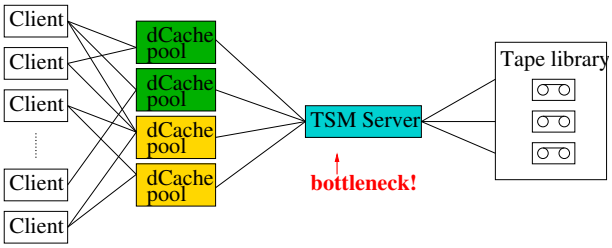

Figure 2: a simple tape connection

To create a scalable and performance solution we introduced storage agents to our setup. A regular TSM server has a database. Whereby a storage agent is a minimised TSM-server without its own database. The storage agent can be installed on a pool node as shown in figure 3, and as such only a management channel to the TSM-server is needed via LAN and the data transfer goes directly via SAN to the tape drives.

Since dCache does its own load balancing, files are stored from one node and retrieved from another node. It is possible to use the same TSM client node name for all

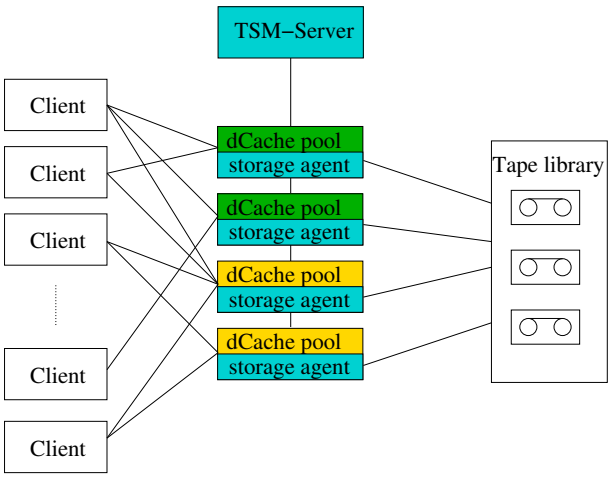

Figure 3: a possible tape connection using storage agents and proxy nodes

clients, if the TSM-server writes the files to tape. However when storage agents are used and the files are all written to the same filespace it is important to introduce proxy nodes. A proxy node is a common name for TSM clients. For example a client with a node name ”alpha” connects to its storage agent and requests to write a file on behalf of ”dCache-proxy”. When a node ”beta” tries to read this file it connects to its own storage agent and requests this file on behalf of ”dCache-proxy”.

The problem with this approach results in an inordinate amount of time used for starting a session. Furthermore the TSM volume selection algorithm starts a cartridge juggle if a file has to be stored. In case the TSM server writes the file directly to tape, the same tape is used till it is full without having to dismount the tape. However when storage agents are used the same tape is being dismounted frequently as can be seen in figure 4.

In this example ”pool A” and ”pool B” have each mounted a tape and start writing to it. Independent if ”pool A” wants to write a new file to the same tape once the first session has finished the TSM server decides to schedule ”tape k” for ”pool C”. As a result ”pool C” has to wait till ”pool A” has dismounted that tape and is able to mount it. If ”pool A” starts a new request while both tapes are still occupied the TSM server will schedule a new tape from its stack to be used. In our case it would have been more desirable to use the new tape directly for ”pool C” however the TSM server tries to optimise the filling of all tapes and as such does not use a new tape for each request as long as it has a tape with some space left. In addition a retrieve action has no way to manage the ordering of the files.

AN IMPROVED PERFORMANCE TAPE SOLUTION

We developed a TSM Session Server (TSS), which creates a connection between the dCache pools and the storage agents using also the TSM API. The TSS is a single executable and opens a TSM session, which is kept alive till a timeout without any request occurs. The advantage of this approach is, that TSM interacts only with a single session and dCache can handle multiple tape flush, retrieval, rename, or queries on log files within the same session. The TSS returns the appropriate return code for each request. As such dCache still keeps the control, which files are on tape and which files did not succeed. Furthermore the TSS sends different types of data to different tape sets, if this information are known from dCache it might also group data which are likely to be retrieved together. Figure 5 demonstrates the same example as explained in figure 4, but using the TSS. It can clearly be seen that the mount action of a storage agent is extremely decreased.

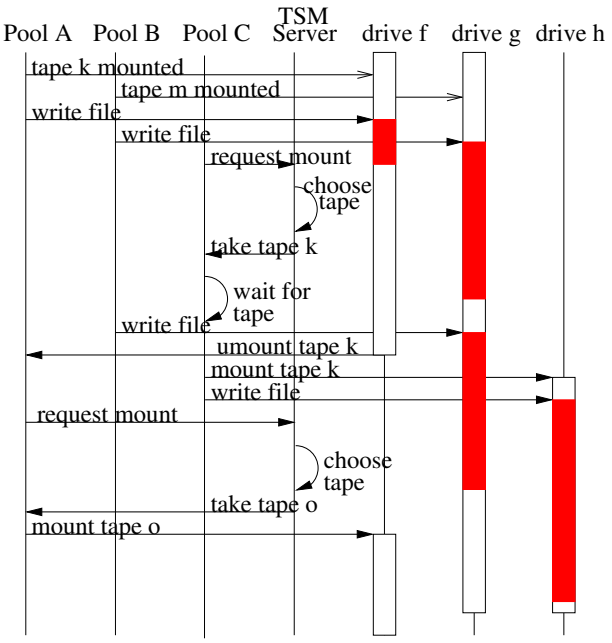

Figure 4: the drawback with storage agents

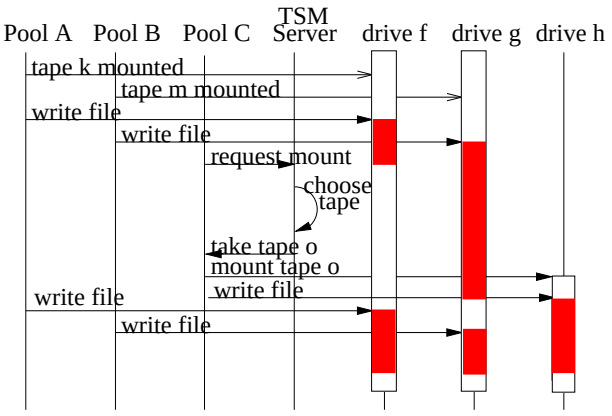

Figure 5: the improved solution using TSS

CONCLUSION

The throughput of a TSM server is limited by the capacity of the machine and this is in our case 100 MB/s. Assuming that one LTO2 drive can write with a maximum speed of 35 MB/s to a tape the TSM server would have to be able to write with a speed of 105 MB/s with only 3 drives connected. Adding additional drives would not create a higher troughput and as such the TSM server is identified as a bottleneck. This initial solution does not scale.

Presuming a storage agent is able to write with the same speed as the TSM server that is 100 MB/s to tape we have a similar problem since adding more storage agents will increase the number of mount actions and as such the throughput is reduced with the number of storage agents used.

While using the TSM Session Server the unnecessary mount actions are reduced and the throughput increases with the number of storage agents. As a result we have created a high-performance and scalable tape connection using TSM. Furthermore the TSS keeps the drives streaming at its maximum rate and creates a clear cut between online and background storage operations. The disk pool management system we use at the moment is dCache, however TSS can be used for other disk pool mangers with little adaptation. The reading operations can be advanced when the disk pool management supports a sort algorithm for these request. This is already announced for dCache. With the current rate of file transactions and the estimated number of files in the future, the TSM database is not expected to become a bottleneck.

ACKNOWLEDGEMENTS

The authors wish to thank people of the department GIS and DASI from IWR at Forschungszentrum Karlsruhe and the German Federal Ministry of Education and Research (BMBF) for their support at various stages in the preparation of the work presented.

REFERENCES

[1] Grid Computing Centre Karlsruhe
http://www.gridka.de

[2] Forschungszentrum Karlsruhe GmbH
http://www.fzk.de

[3] Connecting WLCG Tier-2 Centres to GridKa
A. Heiss, S. Halstenberg, B. Hoeft, D. Ressmann, J. van Wezel
CHEP 06 Proceedings

[4] Service Challenge
https://uimon.cern.ch/twiki/bin/view/LCG/LCGServiceChallenges

[5] Service Challenge Monitoring
http://lxgate24.cern.ch/GRIDVIEW/

[6] dCache
http://www.dcache.org

[7] Storage Resource Manager
http://www-isd.fnal.gov/srm/

[8] Tivoli Storage Manager
http://publib.boulder.ibm.com/infocenter/tivihelp/v1r1/index.jsp