

Optimisation of Grid Enabled Storage at Small Sites

Greig A Cowan
University of Edinburgh, UK

Jamie K Ferguson, Graeme A Stewart
University of Glasgow, UK

Abstract

Grid enabled storage systems are a vital part of data processing in the grid environment. Even for sites primarily providing computing cycles, local storage caches will often be used to buffer input files and output results before transfer to other sites. In order for sites to process jobs efficiently it is necessary that site storage is not a bottleneck to Grid jobs.

dCache and DPM are two Grid middleware applications that provide disk pool management of storage resources. Their implementations of the storage resource manager (SRM) webservice allows them to provide a standard interface to this managed storage, enabling them to interact with other SRM enabled Grid middleware and storage devices. In this paper we present a comprehensive set of results showing the data transfer rates in and out of these two SRM systems when running on 2.4 series Linux kernels and with different underlying filesystems.

This benchmarking information is very important for the optimisation of Grid storage resources at smaller sites, that are required to provide an SRM interface to their available storage systems.

1 Introduction

1.1 Small sites and Grid computing

The EGEE project [1] brings together scientists and engineers from 30 countries to order to create a Grid infrastructure that is constantly available for scientific computing and analysis.

The aim of the Worldwide LHC Computing Grid (WLCG) is to use the EGEE developed software to construct a global computing resource that will enable particle physicists to store and analyse particle physics data that the Large Hadron Collider (LHC) and its experiments will generate when it starts taking data in 2007. The WLCG is based on a distributed Grid computing model and will make use of the computing and storage resources at physics laboratories and institutes around the world. Depending on the level of available resources, the institutes are organised into a hierarchy starting with the Tier-0 centre at CERN (the location of the LHC), multiple national laboratories (Tier-1 centres) and numerous smaller research institutes and Universities (Tier-2 sites) within each participating country. Each Tier is expected to provide a certain level of computing service to Grid users once the WLCG goes into full production.

The authors' host institutes form part of Scot-Grid [2], a distributed WLCG Tier-2 centre, and it is from this point of view that we approach the subject of this paper. Although each Tier-2 is unique in its configuration and operation, similarities can be easily identified, particularly in the area of data storage:

1. Typically Tier-2 sites have limited hardware resources. For example, they may have one or two servers attached to a few terabytes of disk, configured as a RAID system. Additional storage may be NFS mounted from another disk server which is shared with other non-Grid users.
2. No tape storage.
3. Limited manpower to spend on administering and configuring a storage system.

The objective of this paper is to study the configuration of a Grid enabled storage element at a typical Tier-2 site. In particular we will investigate how changes in the disk server filesystems and file transfer parameters affect the data transfer rate when writing into the storage element. We concentrate on the case of writing to the storage element, as this is expected to be the most stressful operation on the Tier-2's storage resources, and indeed testing within the GridPP

collaboration bears this out [3]. Sites will be able to use the results of this paper to make informed decisions about the optimal setup of their storage resources without the need to perform extensive analysis on their own.

Although in this paper we concentrate on particular SRM [4] grid storage software, the results will be of interest in optimising other types of grid storage at smaller sites.

This paper is organised as follows. Section 2 describes the grid middleware components that were used during the tests. Section 3 then goes on to describe the hardware, which was chosen to represent a typical Tier-2 setup, that was used during the tests. Section 3.1 details the filesystem formats that were studied and the Linux kernel that was used to operate the disk pools. Our testing method is outlined in Section 4 and the results of these tests are reported and discussed in Section 5. In Section 6 we present suggestions of possible future work that could be carried out to extend our understanding of optimisation of Grid enabled storage elements at small sites and conclude in Section 7.

2 Grid middleware components

2.1 SRM

The use of standard interfaces to storage resources is essential in a Grid environment like the WLCG since it will enable interoperation of the heterogeneous collection of storage hardware that the collaborating institutes have available to them. Within the high energy physics community the storage resource manager (SRM) [4] interface has been chosen by the LHC experiments as one of the baseline services [5] that participating institutes should provide to allow access to their disk and tape storage across the Grid. It should be noted here that a storage element that provides an SRM interface will be referred to as ‘an SRM’. The storage resource broker (SRB) [6] is an alternative technology developed by the San Diego Supercomputing Center [7] that uses a client-server approach to create a logical distributed file system for users, with a single global logical namespace or file hierarchy. This has not been chosen as one of the baseline services within the WLCG.

2.2 dCache

dCache [8] is a system jointly developed by DESY and FNAL that aims to provide a mechanism for storing and retrieving huge amounts of data among a large number of heterogeneous server nodes, which can be of varying architectures (x86, ia32, ia64). It provides a single namespace view of all of the files that it manages and allows access to these files using a variety of protocols, including SRM. By connecting dCache to a tape backend, it becomes a hierarchical storage manager. However, this is not of particular relevance to Tier-2 sites who do not typically have tape storage. dCache is a highly configurable storage solution and can be easily deployed at Tier-2 sites where DPM is not sufficiently flexible.

2.3 Disk pool manager

Developed at CERN, and now part of the gLite middleware set, the disk pool manager (DPM) is similar to dCache in that it provides a single namespace view of all of the files stored on the multiple disk servers that it manages and provides a variety of methods for accessing this data, including SRM. DPM was always intended to be used primarily at Tier-2 sites, so has an emphasis on ease of deployment and maintenance. Consequently it lacks some of the sophistication of dCache, but is simpler to configure and run.

2.4 File transfer service

The gLite file transfer service (FTS) [9] is a grid middleware component that aims to provide reliable file transfer between storage elements that provide the SRM or GridFTP [10] interface. It uses the concept of channels [11] to define unidirectional data transfer paths between storage elements, which usually map to dedicated network pipes. There are a number of transfer parameters that can be modified on each of these channels in order to control the behaviour of the file transfers between the source and destination storage elements. We concern ourselves with two of these: the number of concurrent file transfers (N_f) and the number of parallel GridFTP streams (N_s). N_f is the number of files that FTS will simultaneously transfer in any bulk file transfer operation. N_s is number of simultaneous GridFTP channels that are opened up for each of these files.

2.5 Installation

Sections 2.2 and 2.3 described the two disk pool management applications that are suitable for use at small sites. Both dCache (v1.6.6-5) and DPM (v1.4.5) are available as part of the 2.7.0 release of the LCG software stack and have been sequentially investigated in the work presented here. In each case, the LCG YAIM [12] installation mechanism was used to create a default instance of the application on the available test machine (See Section 3). For dCache, PostgreSQL v8.1.3 was used, obtained from the PostgreSQL website [13]. Other than installing the SRM system in order that they be fully operational, no configuration options were altered.

3 Hardware configuration

In order for our findings to be applicable to existing WLCG Tier-2 sites, we chose test hardware representative of Tier-2 resources.

1. Single node with a dual core Xeon CPU. This operated all of the relevant services (i.e. SRM/nameserver and disk pool access) that were required for operation of the dCache or DPM.
2. 5TB RAID level-5 disk with a 64K stripe. Partitioned into three 1.7TB filesystems.
3. Source DPM for the transfers was a sufficiently high performance machine that it was able to output data across the network such that it would not act as a bottleneck during the tests.
4. 1Gb/s network connection between the source and destination SRMs, which were on the same network connected via a Netgear GS742T switch. During the tests, there was little or no other traffic on the network.
5. No firewalling (no iptables module loaded) between the source and destination SRMs.

3.1 Kernels and filesystems

Table 1 summarises the combinations of Linux kernels that we ran on our storage element and the disk pool filesystems that we tested it with. As can be seen four filesystems, ext2 [14], ext3 [15], jfs [16], xfs [17], were studied. Support for the first 3 filesystems is included by default in the Scientific Linux [18] 305 distribution. However,

support for xfs is not enabled. In order to study the performance of xfs a CERN contributed rebuild of the standard Scientific Linux kernel was used. This differs from the first kernel only with the addition of xfs support.

Note that these kernel choices are in keeping with the ‘Tier-2’ philosophy of this paper – Tier-2 sites will not have the resources available to recompile kernels, but will instead choose a kernel which includes support for their desired filesystem.

In each case, the default options were used when mounting the filesystems.

Kernel	Filesystem			
	<i>ext2</i>	<i>ext3</i>	<i>jfs</i>	<i>xfs</i>
<i>2.4.21</i>	Y	Y	Y	N
<i>2.4.21+cern xfs</i>	N	N	N	Y

Table 1: Filesystems tested for each Linux kernel/distribution.

4 Method

The method adopted was to use FTS to transfer 30 1GB files from a source DPM to the test SRM, measuring the data transfer rate for each of the filesystem-kernel combinations described in Section 3.1 and for different values of the FTS parameters identified in Section 2.4. We chose to look at $N_f, N_s \in (1, 3, 5, 10)$. Using FTS enabled us to record the number of successful and failed transfers in each of the batches that were submitted. A 1GB file size was selected as being representative of the typical filesize that will be used by the LHC experiments involved in the WLCG.

Each measurement was repeated 4 times to obtain a mean. Any transfers which showed anomalous results (e.g., less than 50% of the bandwidth of the other 3) were investigated and, if necessary, repeated. This was to prevent failures in higher level components, e.g., FTS from adversely affecting the results presented here.

5 Results

5.1 Transfer Rates

Table 2 shows the transfer rate, averaged over N_s , for dCache for each of the filesystems. Similarly 3 shows the rate averaged over N_f .

N_f	Filesystem			
	<i>ext2</i>	<i>ext3</i>	<i>jfs</i>	<i>xfs</i>
1	157	146	137	156
3	207	176	236	236
5	209	162	246	245
10	207	165	244	247

Table 2: Average transfer rates per filesystem for dCache for each N_f .

N_s	Filesystem			
	<i>ext2</i>	<i>ext3</i>	<i>jfs</i>	<i>xfs</i>
1	217	177	234	233
3	191	159	214	219
5	189	155	208	217
10	183	158	207	215

Table 3: Average transfer rates per filesystem for dCache for each N_s .

Table 4 shows the transfer rate, averaged over N_s , for DPM for each of the filesystems. Similarly 5 shows the rate averaged over N_f .

The following conclusions can be drawn:

1. Transfer rates are greater when using modern high performance filesystems like *jfs* and *xfs* than the older *ext2,3* filesystems.
2. Transfer rates for *ext2* are higher than *ext3*, because it does not suffer a journalling overhead.
3. For all filesystems, having more than 1 simultaneous file transferred improves the average transfer rate substantially. There appears to be little dependence of the average transfer rate on the number of files in a multi-file transfer (for the range of N_f studied).
- 4a. With dCache, for all filesystems, single stream transfers achieve a higher average transfer rate than multistream transfers.
- 4b. With DPM, for *ext2,3* single stream transfers achieve a higher average transfer rate than multistream transfers. For *xfs* and *jfs* multistreaming has little effect on the rate.
- 4c. In both cases, there appears to be little dependence of the average transfer rate on the number of streams in a multistream transfer (for the range of N_s studied).

N_f	Filesystem			
	<i>ext2</i>	<i>ext3</i>	<i>jfs</i>	<i>xfs</i>
1	214	192	141	204
3	297	252	357	341
5	300	261	368	354
10	282	253	379	356

Table 4: Average transfer rates per filesystem for DPM for each N_f .

N_s	Filesystem			
	<i>ext2</i>	<i>ext3</i>	<i>jfs</i>	<i>xfs</i>
1	293	277	289	310
3	264	209	313	323
5	264	237	303	307
10	272	234	339	317

Table 5: Average transfer rates per filesystem for DPM for each N_s .

5.2 Error Rates

Table 6 shows the average percentage error rates for the transfers obtained with both SRMs.

5.2.1 dCache

With dCache there were a small number of transfer errors for *ext2,3* filesystems. These can be traced back to FTS cancelling the transfer of a single file. It is likely that the high machine load generated by the I/O traffic impaired the performance of the dCache SRM service. No errors were reported with *xfs* and *jfs* filesystems which can be correlated with the correspondingly lower load that was observed on the system compared to the *ext2,3* filesystems.

5.2.2 DPM

With DPM all of the filesystems can lead to errors in transfers. Similarly to dCache these errors generally occur because of a failure to correctly call `srmsSetDone()` in FTS. This can be traced back to the DPM SRM daemons being

SRM	Filesystem			
	<i>ext2</i>	<i>ext3</i>	<i>jfs</i>	<i>xfs</i>
dCache	0.21	0.05	0	0
DPM	0.05	0.10	1.04	0.21

Table 6: Percentage observed error rates for different filesystems and kernels with dCache and DPM.

badly affected by the machine load generated by the high I/O traffic. In general it is recommended to separate the SRM daemons from the actual disk servers, particularly at larger sites.

Note that the error rate for jfs was higher, by some margin, than for the other filesystems. However, this was mainly due to one single transfer which had a very high error rate and further testing should be done to see if this repeats.

5.3 Comment on FTS parameters

The poorer performance of multiple parallel GridFTP streams relative to a single stream transfer observed for dCache and for DPM with ext2,3 can be understood by considering the I/O behaviour of the destination disk. With multiple parallel streams, a single file is split into sections and sent down separate TCP channels to the destination storage element. When the storage element starts to write this data to disk, it will have to perform many random writes to the disk as different packets arrive from each of the different streams. In the single stream case, data from a single file arrives sequentially at the storage element, meaning that the data can be written sequentially on the disk, or at least with significantly fewer random writes. Since random writes involve physical movement of the disk and/or the write head, it will degrade the write performance relative to a sequential write access pattern.

In fact multiple TCP streams are generally beneficial when performing wide area network transfers, in order to maximise the network throughput. In our case as the source and destination SRMs were on the same LAN the effect of multistreams was generally detrimental.

It must be noted that a systematic study was not performed for the case of $N_f > 10$. However, initial tests show that if FTS is used to manage a bulk file transfer on a channel where N_f is initially set to a high value, then the destination storage element experiences a high load immediately after the first batch of files has been transferred, causing a corresponding drop in the observed transfer rate. This effect can be seen in Figure 1, where there is an initial high data transfer rate, but this reduces once the first batch of N_f files has been transferred. It is likely that the effect is due to the post-transfer negotiation steps of the SRM protocol occurring simultaneously for all of the transferred files. The resulting high load on the destination SRM node causes all subsequent FTS transfer requests to time out, resulting in the transfers failing. It must be noted that our use of a 1GB test file for

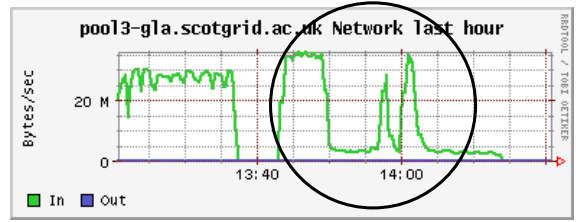


Figure 1: Network profile of destination dCache node with jfs pool filesystem during an FTS transfer of 30 1GB files. Throughout the transfer, $N_f = 15$. Final rate was 142Mb/s with 15 failed transfers.

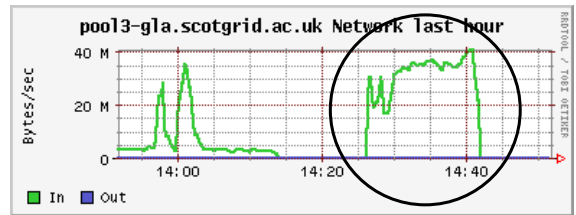


Figure 2: Network profile of destination dCache node with jfs pool filesystem during an FTS transfer of 30 1GB files. $N_f = 1$ at the start of the transfer, increasing up to $N_f = 15$ as the transfer progressed. Final transfer rate was 249Mb/s with no failed transfers.

all transfers will exacerbate this effect. Figure 2 shows how this effect disappears when the start times of the file transfers are staggered by slowly increasing N_f from $N_f = 1$ to 15, indicating that higher file transfer rates as well as fewer file failures could be achieved if FTS staggered the start times of the file transfers. Improvements could be made if multiple nodes were used to spread the load of the destination storage element. If available, separate nodes could be used to run the disk server side of the SRM and the namespace services.

6 Future Work

The work described in this paper is a first step into the area of optimisation of grid enabled storage at Tier-2 sites. In order to fully understand and optimise the interaction between the base operating system, disk filesystem and storage applications, it would be interesting to extend this research in a number of directions:

1. Using non-default mount options for each of the filesystems.
2. Repeat the tests with SL 4 as the base oper-

ating system (which would also allow testing of a 2.6 Linux kernel).

3. Investigate the effect of using a different stripe size for the RAID configuration of the disk servers.
4. Vanilla installations of SL generally come with unsuitable default values for the TCP read and write buffer sizes. In light of this, it will be interesting to study the how changes in the Linux kernel-networking tuning parameters change the FTS data transfer rates. Initial work in this area has shown that improvements can be made.
5. It would be interesting to investigate the effect of different TCP implementations within the Linux kernel. For example, TCP-BIC [19], westwood [20], vegas [21] and web100 [22].
6. Characterise the performance enhancement that can be gained by the addition of extra hardware, particularly the inclusion of extra disk servers.

When the WLCG goes into full production, a more realistic use case for the operation of an SRM will be one in which it is simultaneously writing (as is the case here) and reading files across the WAN. Such a simulation should also include a background of local file access to the storage element, as is expected to occur during periods of end user analysis on Tier-2 computing clusters. Preliminary work has already started in this area where we have been observing the performance of existing SRMs within ScotGrid during simultaneous read and writing of data.

7 Conclusion

This paper has presented the results of a comparison of file transfer rates that were achieved when FTS was used to copy files between Grid enabled storage elements that were operating with different destination disk pool filesystems and a 2.4 Linux kernel. Both dCache and DPM were considered as destination disk pool managers providing an SRM interface to the available disk. In addition, the dependence of the file transfer rate on the number of concurrent files (N_f) and the number of parallel GridFTP streams (N_s) was investigated.

In terms of optimising the file transfer rate that could be achieved with FTS when writing

into a characteristic Tier-2 SRM setup, the results can be summarised as follows:

1. Pool filesystem: jfs or xfs.
2. FTS parameters: Low value for N_s , high value for N_f (staggering the file transfer start times). In particular, for dCache $N_f = 1$ was identified as giving optimal transfer rate.

It is not possible to make a single recommendation on the SRM application that sites should use based on the results presented here. This decision must be made depending upon the available hardware and manpower resources and consideration of the relative features of each SRM solution.

Extensions to this work were suggested, ranging from studies of the interface between the kernel and network layers all the way up to making changes to the hardware configuration. Only when we understand how the hardware and software components interact to make up an entire Grid enabled storage element will we be able to give a fully informed set of recommendations to Tier-2 sites regarding the optimal SRM configuration. This information is required by them in order that they can provide the level of service expected by the WLCG and other Grid projects.

8 Acknowledgements

The research presented here was performed using ScotGrid [2] hardware and was made possible with funding from the Particle Physics and Astronomy Research Council through the GridPP project [23]. Thanks go to Paul Millar and David Martin for their technical assistance during the course of this work.

References

- [1] Enabling Grids for E-scienceE
<http://www.eu-egee.org/>
- [2] ScotGrid, the Scottish Grid Service
<http://www.scotgrid.ac.uk>
- [3] GridPP Service Challenge Transfer tests
http://wiki.gridpp.ac.uk/wiki/Service_Challenge_Transfer_Tests
- [4] The SRM collaboration
<http://sdm.lbl.gov/srm-wg/>

- [5] LCG Baseline Services Group Report
<http://cern.ch/LCG/peb/bs/BSReport-v1.0.pdf>
- [6] The SRB collaboration
http://www.sdsc.edu/srb/index.php/Main_Page
- [7] The San Diego Supercomputing Center
<http://www.sdsc.edu>
- [8] dCache collaboration
<http://www.dcache.org>
- [9] gLite FTS
<https://uimon.cern.ch/twiki/bin/view/LCG/FtsRelease14>
- [10] Data transport protocol developed by the Globus Alliance
http://www.globus.org/grid_software/data/gridftp.php
- [11] P. Kunszt, P. Badino, G. McCance, The gLite File Transfer Service: Middleware Lessons Learned from the Service Challenges, CHEP 2006 Mumbai.
<http://indico.cern.ch/contributionDisplay.py?contribId=20&sessionId=7&confId=048>
- [12] LCG generic installation manual
<http://grid-deployment.web.cern.ch/grid-deployment/documentation/LCG2-Manual-Install/>
- [13] PostGreSQL Database website
<http://www.postgresql.org/>
- [14] R. Card, T. Ts'o, S. Tweedie (1994). "Design and implementation of the second extended filesystem." Proceedings of the First Dutch International Symposium on Linux. ISBN 90-367-0385-9.
- [15] Linux ext3 FAQ
<http://batleth.sapientsat.org/projects/FAQs/ext3-faq.html>
- [16] Jfs filesystem homepage
<http://jfs.sourceforge.net/>
- [17] Xfs filesystem homepage
<http://oss.sgi.com/projects/xfs/index.html>
- [18] Scientific Linux homepage
<http://scientificlinux.org>
- [19] A TCP variant for high speed long distance networks
<http://www.csc.ncsu.edu/faculty/rhee/export/bitcp/index.htm>
- [20] TCP Westwood details
<http://www.cs.ucla.edu/NRL/hpi/tcpw/>
- [21] TCP Vegas details
<http://www.cs.arizona.edu/protocols/>
- [22] TCP Web100 details
<http://www.hep.ucl.ac.uk/~ytl/tcpip/web100/>
- [23] GridPP - the UK particle physics Grid
<http://gridpp.ac.uk>