

# HFS TSM Disk-based Backup Pilot Report

Ian Smith

OUCS - August 2011

# Table of Contents

1. Introduction .....	3
2. Data Deduplication .....	3
2.1. Dedupe Standalone Appliances .....	3
Pros .....	4
Cons .....	4
2.2. TSM deduplication .....	4
Pros .....	5
Cons .....	5
3. Test Environment .....	5
3.1. Host Machine .....	5
3.2. Disk/Filesystem Setup .....	6
3.3. TSM Server configuration .....	6
3.4. Test Data .....	6
4. Test-suites .....	7
4.1. Test suite 1 .....	7
4.2. Test suite 1 performance results .....	7
4.3. Deduplication results .....	8
5. Evaluation and conclusions .....	9
5.1. Deduplication .....	9
5.2. Performance .....	9
6. Recommendations .....	10

# 1. Introduction

Disk-based backup has predictably become more popular as disk capacity has risen and the cost-per-terabyte has fallen. Different disk technologies ( SAS, SATA , and Fibre Channel (FC) ) offer a range of performance over a range of price, such that a solution can be crafted for most budgets.

For some time, disk has been utilised in Virtual Tape Libraries (VTLs ) where the underlying disk is presented to the backup application as a number of tape volumes. This has allowed some applications geared more to using and managing tape to be able to take those advantages that disk offers: namely speed of access, capacity and ease of management.

Additional to the above has been the recent introduction of de-duplication technologies that variously claim to offer de-duplication ratios anywhere from 10 to 500<sup>1</sup>. This translates to space savings on disk that further drive down the cost of backup storage.

Accordingly, as part of the HFS infrastructure upgrade, this study has undertaken to investigate whether disk-based backup utilising data deduplication techniques might offer cost and/or functional enhancements to the HFS Backup Service.

## 2. Data Deduplication

Data deduplication is a method of reducing storage needs by eliminating redundant data. Only one unique instance of the data is actually retained on storage media while subsequent copies are replaced with a pointer to the unique data copy. The typical example cites an email system containing 100 instances of the same one megabyte file attachment. If the email platform is backed up or archived, all 100 instances are saved, requiring 100 MB storage space. With data deduplication, only one instance of the attachment is actually stored; each subsequent instance is just referenced back to the one saved copy. In this example, a 100 MB storage demand could be reduced to only 1 MB.

While an extreme example, it can be seen that the key to data de-duplication is for there to be an element of similarity among the data for it to yield any storage savings. The simplistic example of an email attachment serves for illustrative purpose, but in reality most deduplication engines look at blocks (or 'chunks') of data of either fixed or varying size at the sub-file level rather than looking solely at the level of an entire file for data redundancy.

Further detail on the basics data deduplication have been sufficiently described elsewhere and it is not the intention to repeat this here. However, it is useful to examine here how some features of the various methods of data deduplication impacted on the setup of the current test environment and how they may impact on any future implementation in a production backup service.

### 2.1. Dedupe Standalone Appliances

A number of standalone appliances are available from a variety of vendors<sup>2</sup>. These storage

---

1 <http://emc.dcig.com/2011/02/getting-real-about-deduplication.html>

2 <http://www.backupcentral.com/mr-backup-blog-mainmenu-47/13-mr-backup-blog/348-target-deduplication-appliance-performance-comparison.html>

appliances sit in their own chassis with a mix of ethernet, SCSI and FC host connections and a bunch of (expandable) SATA disk pre-configured in a RAID configuration. The dedupe engine will either deduplicate the data inline (i.e. as it is received) or after the data has been stored. The resultant data on disk can be shared according via common protocols (CIFS, NFS etc) while remaining a backup target in a disk-to-disk backup environment. Alternatively the appliance may act as a VTL and the deduped data held on virtual tapes emulating a standard format.

## Pros

- Deduplication processing is offloaded from the TSM server and host.
- Highly configurable: offer in-line and scheduled deduplication of data.
- If data from multiple TSM servers is stored on the storage device, the wider scope of data allows greater storage savings to be achieved (deduplication across TSM servers).
- Offer easy multi-site replication of deduplicated storage.

## Cons

- 'Black box' implementation of deduplication can complicate problem resolution between vendor and TSM support if data was lost or corrupted.
- Cost – price / TB is considerably higher than a standard disk server.

## 2.2. TSM deduplication

Data deduplication was a capability introduced with TSM version 6.1. It is only offered out-of-band (i.e. after initial storage) and can only be performed on data on disk, not on tape. Thus, essentially the architecture involves taking data into a (probably disk-based) staging area and then deduping it. There then follows a three-stage process:

1. copying the data to a second, permanent disk storage area.
2. identifying the duplicate data against an internal index and replacing the data chunk(s) with an index pointer.
3. reclaiming the storage space previously held by duplicate data chunks.

Some constraints are either enforced or configurable, so that the process is manageable: It is recommended that at least one copy of all the data is held in a non-deduplicated (cf fully constructed) state *before* that data is subject to deduplication. There are configurable size limits on how large a candidate object can be before it is excluded from dedupe processing. The process of identifying the duplicate data is extremely processor and memory intensive and recommended resources for a TSM server running deduplication need to be increased significantly over and above those sufficient for a similar server not running deduplication. Deduplication will also significantly increase the amount of log and database space required. Significantly, if deduplication is

subsequently turned off, the deduplicated objects are not 'reconstructed', but remain. This is critical: turning off deduplication leaves a legacy of increased database space due to preserving the index table that is key to accessing duplicate objects. The only method to 'wipe' the deduplication from an TSM server once it has been implemented, is to rename the existing the client backups and require that clients backup their data again.

## Pros

- TSM deduplication is implemented by the TSM server and the tracking of indexed items remains within the TSM server. There can be no argument about the origins of 'lost' or 'corrupt' data, should such a case arise.
- The feature is included in the standard TSM EE licence at no extra charge.
- It can utilise any disk.

## Cons

- Cannot deduplicate real-time , only out-of-band.
- Requires extrae CPU and memory resources on the TSM host machine.
- Inflates the DB and LOG space.
- Not easy for a TSM server to revert back to completely non-deduplicated data.
- Data scope restricted to one TSM server. Cannot dedupe across TSM servers.

## 3. Test Environment

In spite of some of the advantages offered by standalone dedupe appliances, the main strand of enquiry of this study was to determine if data deduplication was a viable adjunct to a disk-based backup service within the existing HFS backup service. To this end, the major questions to be asked, were: does the Oxford data deduplicate well ? And, how can native TSM data deduplication be implemented on existing and available disk ? To this end, the purchase and deployment of a deduplication appliance was not considered justified for the purposes of this pilot study.

### 3.1. Host Machine

The test harness consisted of a logical partition (LPAR) provisioned from an IBM P7-770 server. The LPAR was initially allocated a maximum of 6 processors (symmetrical-multi-threaded enabled) and 16GB of RAM. Levels of AIX 6.1 and TSM-DB2 6.2.2.3 were installed to mimic production machines and services.

### **3.2. Disk/Filesystem Setup**

The TSM database, activelog and mirrorlog were placed on 15K SAS disks. The database was spread across 8 disks, the logs across a separate 4 disks, in RAID10 configurations. Production systems at Oxford run on fast FC disk, but capacity on these for a test environment was not available without compromising either test or production operations. In line with recommended best-practice<sup>3</sup>, the database and transaction logs were located on the fastest disk available (15K SAS) and with best read-write characteristics ( read cache off and write cache on ).

Raw client data was held typically on 7K SATA disk in a number of RAID5 groups – the precise number of spindles dependant on the amount of source data being manipulated. The disk storage used for the deduplicated data was held on separate disks, their characteristics and configuration varied across test-suites in order to examine the effect of different disk-configurations on the speed of the de-duplication processes.

### **3.3. TSM Server configuration**

In general the TSM server was configured as per the recommendations contained within the IBM Best Practices document for implementing deduplication<sup>4</sup>. Two deviations from these were:

1. The requirement that a copy of an object must first be stored in its full form before becoming a candidate for deduplication processing was switched off for the purposes of this pilot. The client data used was a copy already and completely separate from the copies held in the production backup service.
2. The parameter that controls the maximum size of file to be stored in a deduplicated storage pool was left unset instead of being set to slightly above the size limit for files that can be considered for deduplication. The practical effect of this was, however, the same; namely that some objects beyond 300GB in size would be stored but would not be processed as candidates for deduplication.

### **3.4. Test Data**

As noted in section 2.2, once turned on, deduplication cannot be removed without requiring a fresh full backup from the clients. This essentially precluded the preferred method of taking a production TSM server, making a fourth copy of the data and deduplicating it, as there would have been no easy back-out path from this without affecting the production service.

Accordingly, the only method of accessing real client data while guaranteeing the integrity of the production backup service (and client backups) was to export a copy of the data to a test TSM server. This was done, with the caveat that the export process itself is quite slow as it updates the TSM database in the target server.

---

<sup>3</sup> Tivoli Storage Manager V6.1 Technical Guide [p53-54]

<sup>4</sup>

<http://www.ibm.com/developerworks/wikis/display/tivolistoragemanager/Data+deduplication+best+practices+for+Tivoli+Storage+Manager+V6.2>

Four data sets were used:

Set 1: Approximately 2TB of client data (>3 million files) backed up via the HFS 'backup-over-vpn' service. Over 80 different client machines and a mix of client OS platforms.

Set 2: 10TB of server data from 3 large Linux server clients.

Set 3: 7.5TB of server data from 8 server clients running Win2008.

Set 4: 12.5TB of client data from 16 desktop clients running Mac OSX, plus a large Windows client and a large Linux client.

## **4. Test-suites**

### **4.1. Test suite 1**

A series of 8 environments were set up in which the 2TB (backup-over-VPN) data was copied, the duplicates identified and the space in the deduplication pool was reclaimed. Variables manipulated and compared with a view to determining comparative performance were:

- SATA vs SAS disk for the target disks.
- SATA vs SAS disk for the source disks.
- RAID 5 vs RAID10
- Disk segment size.
- Available memory.
- Available Processor capacity.

### **4.2. Test suite 1 performance results**

In general we did not see any difference in the performance of SAS against SATA disk. This was perhaps a surprise, but figures confirmed a throughput of between 130-200MB/s for both disk types. This figure improved slightly with more memory (24GB against 16GB) allocated to the system to peak at 250MB/s – as probably more space was allocated to the file system cache. In general across similar configurations, however, both SAS and SATA disks showed similar throughput rates.

Predictably, RAID configuration showed a difference in throughput rates as the RAID-5 read-modify-write penalty of parity affected data written to disk against RAID-10. However, the differences were small, in the order of 20%.

The three major constituent processes of deduplication; backup, identification of duplicates, space

reclamation, are all fairly resource-hungry. However, the identify duplicates stage was by far the most demanding of cpu and memory resources, at times utilising the entire processor resource allocated to the system and, up to 50% of all memory in a system provisioned with 24GB of memory.

While the identification phase of deduplication appeared to be bound by cpu and memory resources, its performance was a little confounded by the discovery that candidate data for identification was read from the original source data pool, *not* the deduplicate pool into which it had been copied. This may, however, not have had too much effect as the speed of data-duplicate identification was, as noted, limited by cpu and memory resources.

Disk segment size did appear to have an effect on data transfer times – with an optimum of 64k or 128k allowing around a 60% increase in throughput over the least optimum configuration.

The major finding was the average time the local cpu spent in I/O wait – i.e. waiting for disk I/O to complete was around 14-16% in the initial data transfer. With 6 processors defined to the system this indicates that all threads were waiting on disk I/O to complete.

### 4.3. Deduplication results

Data deduplication is typically discussed as a percentage as follows:

$$100 \times ( ( \text{original bytes stored} - \text{final bytes stored} ) / \text{original bytes stored} ),$$

or as a ratio:

$$( 1 / ( \text{final bytes stored} / \text{original bytes stored} ) ).$$

So a deduplication percentage of 50% ( half the data is not stored as it is duplicate), equates to a 2:1 ratio. A dedupe percentage of 75 ( three quarters of the data is duplicate and not stored ) equates to a 4:1 ratio.

A summary table of deduplication results is below.

Test Suite	Data (GB)	Time to identify duplicate data (hh:mm)	Data not stored (GB)	Dedupe %age	Dedupe Ratio
1	2008	03:26	853	42%	1.73 : 1
2	10384	11:02	895	8%	1.10 : 1
3	7386	15:15	3719	50%	2 : 1
4	11561	54:75	6335	55%	2.21 : 1

## 5. Evaluation and conclusions

### 5.1. Deduplication

It is fair to say that the results from this pilot were underwhelming. The deduplication ratios attained were far below figures mentioned by others when discussing the benefits of deduplication<sup>5</sup>. However, some caveats are in order that affect the interpretation of these results:

- The TSM backup algorithm is a 'forever incremental' one as opposed to a traditional model where regular full backups are taken. An incremental model will result naturally in less duplicate data stored. Repeated full backups will lead to greater duplication.
- The larger the data pool, the greater the chance of finding duplicate data and therefore the greater the dedupe ratios to be attained. This pilot was constrained by the need to export the data out of a production system – which was slow and limited the total size of the candidate data pool.
- Some objects deduplicate better than others. The Windows System files and Exchange Mail backups are two cases. The former case was not included in this study due to a TSM software bug in the export of such data proceeding very slowly. The Exchange data was not an initial candidate for the pilot study, but will be included in future trials.
- There is a decreasing saving of storage space for each increment of deduplication after the 2:1 dedupe ratio is reached. At 2:1 you save 50% of disk storage, at 4:1 you save another 25% ( 75% in total ), at 8:1 you save another 12.5%, at 16:1 you save another 6.25%. In fact the difference between a 10:1 dedupe ratio and a 50:1 ratio is just 8% storage space saved. It is therefore the initial lower ratios that should be striven for.

### 5.2. Performance

It is clear that there is no such thing as a 'free lunch' with this technology. The process is time-consuming and extremely resource intensive, requiring large amounts of CPU and memory. Some important points arose as follows:

- Identification of duplicates is the most resource intensive process and is advised to be scheduled not to run concurrently with other housekeeping tasks on the TSM server.
- The length of time dedupe identification took was obviously a function of the amount *and nature* of the data. Some data was processed at around 1TB per hour, some at half that speed. As the total amount of deduplicated data increased, so the master index of items will grow and identification will take longer. The figures from test suite 4 indicate that this process could take a considerable amount of time.
- Data storage space is not immediately freed after duplicate data is identified. In our current environment, there would be a 28-day delay before the disk volumes freed of space could be re-used. This is a function of the window we allow for restoring the TSM server database to

---

5 <http://thebackupblog.typepad.com/thebackupblog/2009/04/actual-deduplication-ratios.html>

a previous state (in case of gross corruption, for example).

- Disk access is mainly random in nature and thus benefits from higher spin speed of SAS disks over SATA.
- Disk segment size of 64k or 128k can also offer a performance increment of +60%.
- As with most disk access, the greater number of spindles accessed for read or write, the better the performance.
- Client restore of data can potentially be slower than currently for large files and large data restores, as the deduplicated data may need to be accessed off multiple disk volumes in its reconstruction. The server-side DB2 LOCKLIST variable may need to be adjusted in cases where large files (>300GB) are deduplicated. The TSM server parameter managing the number of volumes which can be accessed simultaneously by a client also needs to be raised from a default value of 1 to the maximum number of disk volumes concurrently accessed by a client restoring deduplicated data.

## 6. Recommendations.

The study yields promising and intriguing results about how far Oxford backup data can be deduplicated and how much disk storage space can be saved thereby. To an extent, the current backup policies in force across the service, namely only two versions, a restricted retention period for old versions of data and a policy of excluding system files (for desktop accounts), all tend for the backups to consist of data unique to the local system and user. That said, it is clear that the more candidate data there is, the greater the potential duplication can be found.

This study identified that the entire process of implementing server-side deduplication would be subject to local resource limits. What these limits actually would be was complicated by the fact that the test environment was sharing CPU and memory resources with production services. As such, it was never possible to give the system unlimited resources to see where these limits might be. We can say, however, that at least 16GB and several processors be dedicated to a single TSM server instance performing deduplication. Currently our 12 TSM instances dynamically share 12 processors between them. As such, it is clear that server-side deduplication is out of reach in our current environment.

Even with adequate CPU and memory resources it appears there are limits to the amount of data that can be deduplicated in the TSM working day. The index file will grow and searching this is a natural brake on the speed of duplicate identification. IBM advice<sup>6</sup> that server-side dedupe should be considered for a daily data intake up to 6TB per day, resource considerations notwithstanding, accords with our findings.

One solution is to offload the process elsewhere. The market is now well-populated with standalone disk-servers that implement deduplication at the disk-controller – see DataDomain<sup>7</sup> or IBM's ProtectTier Gateway and Appliance products<sup>8</sup>. The pros and cons of these have been noted earlier in this document, but suffice to say that their impressive throughput and capacity (up to 900MB/sec

---

6 Tivoli Storage Manager V6.1 Technical Guide [p151]

7 <http://www.datadomain.com/products/>

8 <http://www-03.ibm.com/systems/storage/tape/enterprise/virtual.html>

and 1PB for the IBM TS7680) come at a considerable cost.

With TSM version 6.2, a second solution to offloading the deduplication process is now available. At this level of both server and client software, the deduplication process can be performed by the client on the client system. This will offer scalability lacking in a server-side solution by sharing the processing burden across multiple client systems. The downsides will be that client backups will take longer and the data is deduplicated *before* it reaches the TSM server and therefore *before* a backup has been made. This presents a small risk of data loss if the deduplicate index is corrupted. Additionally the TSM server will still have to manage the a potentially large index of duplicates and search this on behalf of the client processes. Nevertheless, the scalability argument is a powerful one.

Data deduplication is a tradeoff between storage savings and processing impact. This study has shown that this tradeoff is not automatically achieved and may only be worthwhile when deduplication is targeted at specific data sets. This investigation will continue further to identify where those savings are to be made and the costs incurred.