

# Does your storage management software understand the value of your data?

This white paper shows how MPSTOR's storage management software stack recognises the value of data and utilises disk tiering strategies to greatly reduce the cost per terabyte of managed storage by mapping Logical Volumes to multiple RAID's of disks with varying quality of service.



## Executive Summary

### Overview

The data storage industry has seen great innovation in disk technology. Disks of varying properties and cost are now available to the user such as Solid State Disk, Enterprise class SAS and Fiber Channel as well as SATA. The challenge posed for storage management software is make the most efficient use of these disks. A well designed storage management software stack recognises the value of data. Data scales in value from low infrequently used data such as archives to high mission-critical, frequently used data such as financial transactions. The value of this data to an enterprise is not equal and therefore a strategy of different drive types with differing costs to store this data is required in order to optimise media costs.

MPSTOR's software stack with in-built Self Organising Storage (SOS) capabilities can address this challenge. It does this by clustering disks in groups by QoS (Quality of Service) and building RAID's of the same QoS level. Logical Volumes allocate RAID space from these RAID's to build a volume with mixed QoS disk. Allocation of space is done by referencing a usage model to select the appropriate disk tier in a tiered storage environment. The graph below shows the cost in US\$ of 16TB of managed storage using different drive mixes, with SATA(E) as enterprise SATA and SATA(D) as desktop SATA.

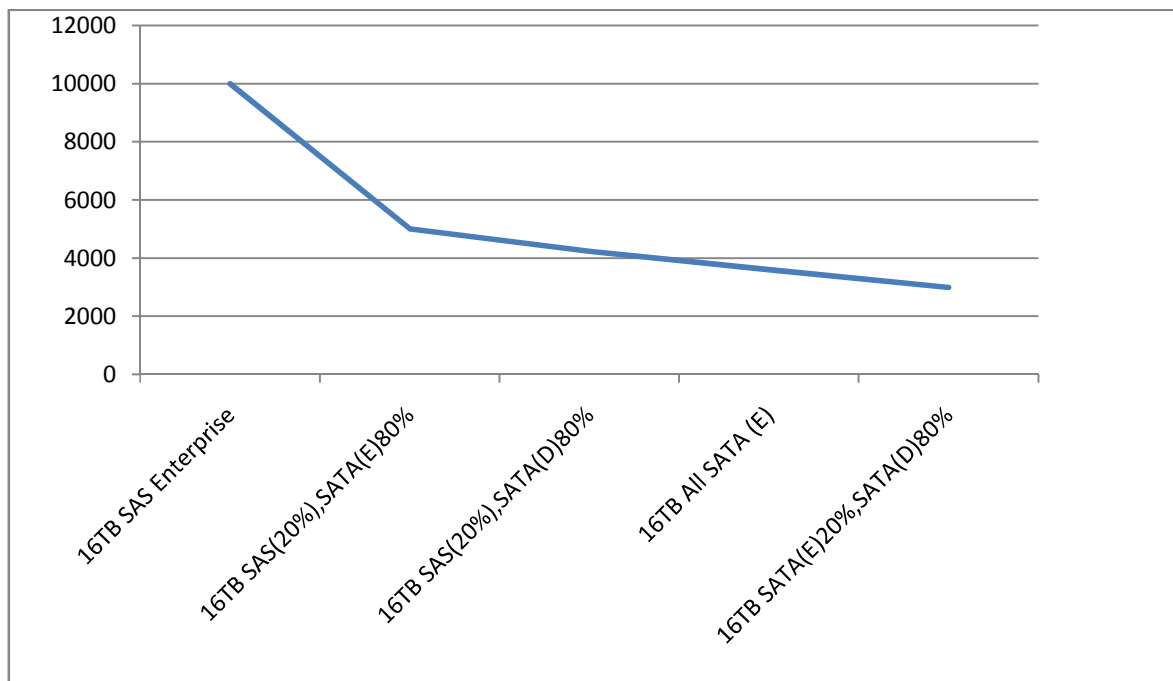


Figure 1

## Business Challenge

The emerging fields of High Capacity Storage and Object Storage Disks (OSD) have arisen in order to address the exponential growth in storage requirements driven by the digitisation of all media such as digital broadcast, file data, video, music, voice mail and archive data.

A strategy is required to store this data efficiently on the media that is most appropriate to its value.

Figure 1 (above) shows the cost of 16TB of storage using different drive mixes. Figure 2 (below) shows typical enterprise data being a mixture of Archive, Business and Critical data, all with varying value to the company.

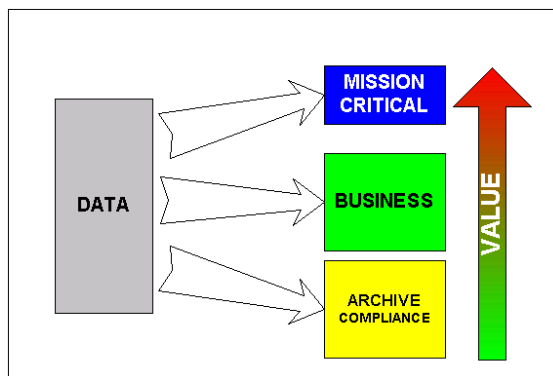


Figure 2

The strategy shown in figure 1 which provisions only SAS Enterprise storage is not an optimal strategy since the low value data is stored on high cost media. Equally the strategy of storing all data on desktop media would not be appropriate for Mission-Critical data.

Clearly a mix of disk tiers is required to give an optimum solution to storing valuable data on valuable media and less valuable data on cheaper media.

Figure 3 shows the relative optimisation value for the different drive mixes.

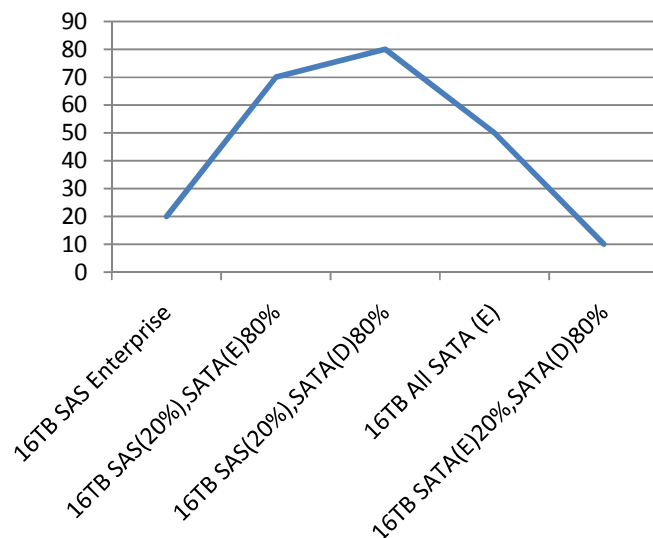


Figure 3

## The Technical Challenge

### Tiered Storage Strategy

To provide an optimal solution to the *problem of managing exponential growth of data of varying value*, an enterprise needs a tiered storage strategy where each tier is made up of disks of a particular class.

### Quality of Service (QoS)

Each disk class has a particular price point and functionality level (Quality of Service). QoS designates the overall performance of a drive. Drives used in top Tiers have a high QoS and drives in the lower tiers have a low QoS.

Drive prices are based on a QoS level, drives with high QoS (Enterprise drives) are expensive and drives with lower QoS are less expensive (e.g. desktop drives).

### Duty Cycles

The QoS of a disk is related to its overall performance, a critical parameter being duty cycle, or the ratio of time the disk heads are moving relative to the time the disks are powered on.

For example, desktop drives are specified to work during a normal workday with a duty cycle of about 10%. Enterprise drives are specified to work 24/7 with a duty cycle of about 60%.

Duty Cycle: % of Power On Time drive head is moving. Duty cycle is directly correlated to the Mean Time Between Failure (MTBF) of the drive.

Class	Duty Cycle	MTBF
Enterprise	60% - 80%	1.2m hours
Midline	15% - 25%	1.2m hours
Desktop/Mobile	5% - 10%	300k hours or warranty notice

An optimal strategy for storage of any given data flow is that frequently accessed data should be stored on High QoS disks, and low/medium accessed data should be stored on low QoS disks

As data flows into a data centre, the Tiered Storage strategy should effectively separate the data into streams of value is required. Once these streams of value have been defined the data can then be routed to the most appropriate storage.

## SOS – Understanding the Value of Data

### Disk Reliability and Duty Cycle Correlation

The data flow to each Tier of the Storage Array needs to be kept within the operational limits of the member disks as otherwise the disks will prematurely fail. Figure 4 shows the correlation between Disk Reliability and Duty cycle for different classes of disks.

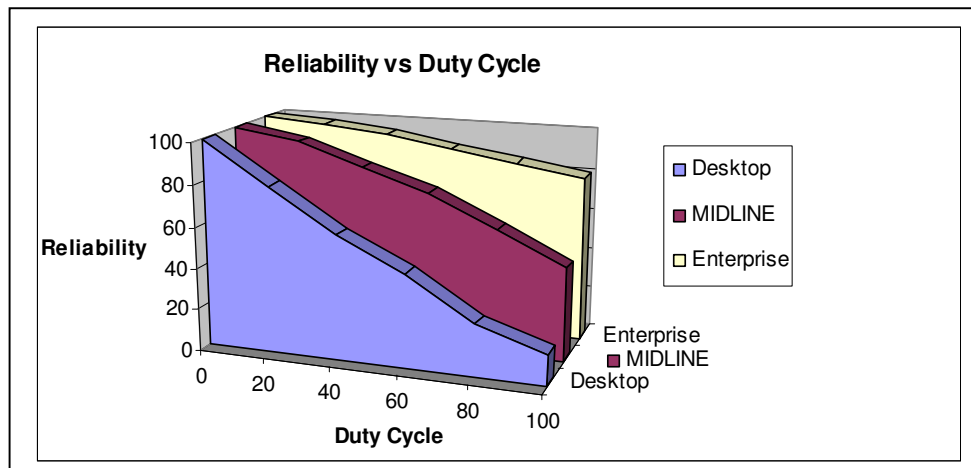


Figure 4

With the increasing complexity of storage system requirements and dramatic increases in required storage capacity there is a need for improved management of resources of data storage systems to optimise extent and reliability of data storage. Several methods of implementing tiered storage are possible.

For example, tiered storage may be managed at a file system level where files are individually tagged with importance levels (either through manual tagging or access counts), and these files are then stored on the appropriate medium.

There are two distinct disadvantages to this approach, namely portability and granularity. If Tiered Storage is to be implemented at a file system level, then the Tiered Storage Server itself must implement each file system type and modify each file system explicitly to support the tiered storage functionality.

The granularity problem arises when large files are required, since a file is associated with only one importance level. Some blocks of a large file may be accessed very often while other blocks are rarely accessed. Since all blocks of the file are stored on media of the same quality there is an inevitable mismatch between the value of certain blocks of data and the quality of the physical medium it is stored on.

## SOS – Understanding the Value of Data

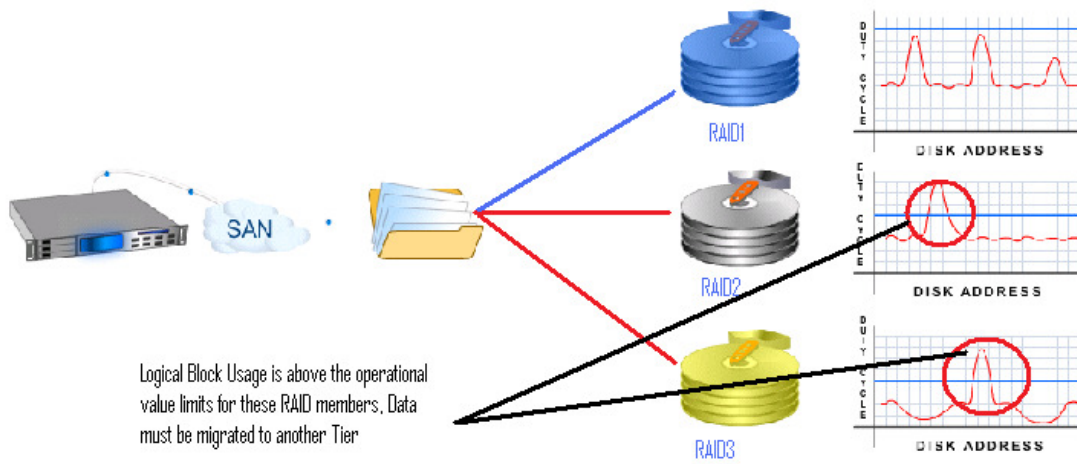


Figure 5

### Creating Logical Volumes

Figure 5 above show the logical block usage over a RAID, in the bottom two RAID stacks certain LBA ranges are used so frequently to push the drives beyond their operational limits.

## SOS – Understanding the Value of Data

### Critical functions of an SOS system

A well designed SOS manager should carry out a number of critical functions:

Feature	Description of Functionality
RAID with QoS	Allows disks to be grouped by Quality of Service and Logical Volumes built from space from RAIDs of varying QoS.
Tiered Logical Volume (TLV)	dynamically migrates data on a block basis between tiers of the TLV to optimise use of the disk resources.
Duty Cycle Limits	monitors use of disk resources and dynamically migrates data to ensure that disks do not exceed a duty cycle limit.
Dynamic Mapping	dynamically modifies mappings between logical volumes and disk resources.
Meta-Data based predictive migration	maintains meta-data allowing it to predict usage of disk resources to automatically migrate data based on statistical time series analysis.
Mapping Modification	modifies a mapping according to predicted hit count to minimise the differences between current and predicted mappings.
Tie-Break Management	executes tie-breaking algorithms according to configurable condition.
Capacity Deficit Management	resolves predicted QoS capacity deficits by swapping blocks of high predicted usage with blocks of low predicted usage and iteratively repeats these steps until all QoS deficits have been resolved. Note : A capacity deficit is a value which indicates the estimated usage above what a tier can sustain.
Predicted Usage Subsets	generates subsets of predicted usage values in which the sum of each subset is no more than a predefined value. The SOS manager should swap large values from sets that exceed their required sum with small values from sets that are less than a

## SOS – Understanding the Value of Data

	required sum. The SOS manager should operate iteratively until capacity deficits are resolved.
--	--

## SOS – Understanding the Value of Data

### Technical Specifications

To investigate SOS in more detail, take a look at the following example with reference to the accompanying drawings in which Fig. 6 illustrates a mechanism by which logical block addresses are mapped to addresses on physical media in a Tiered Logical Volume Manager;

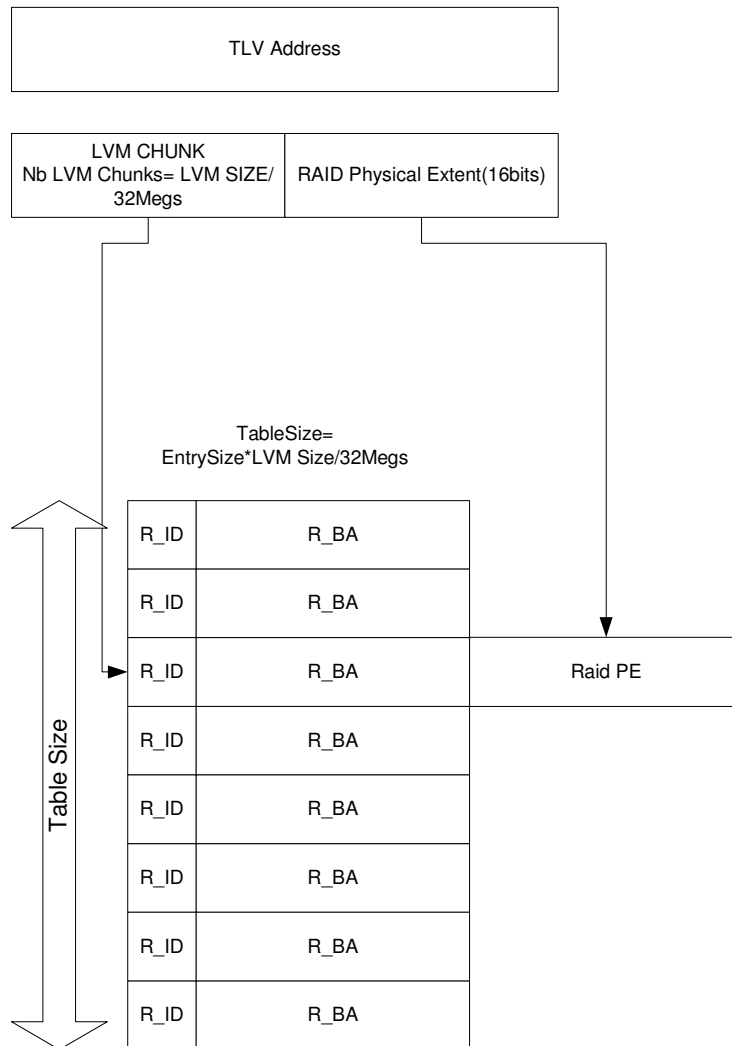


Figure 6

Figs. 7 illustrate an algorithm for resolving predicted QoS capacity where RAID3 has a deficit in capacity usage of 5+1+1+1+1. This is swapped with R1 which has a usage surplus to produce a new system where no individual RAID has a usage capacity deficit.

## SOS – Understanding the Value of Data

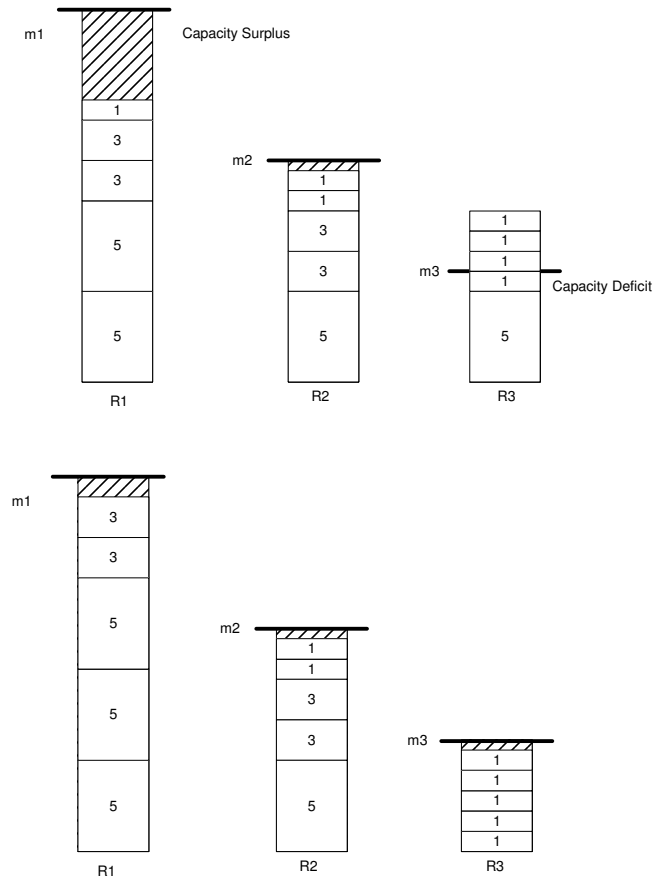


Figure 7

A self-organising storage (SOS) manager manages data storage in a tiered storage disk array. Within the array, disks are organised into Raids and data is striped across the Raids. The Raids are built exclusively with disks from a particular Tier of the Tiered storage Array. Tiered Raids inherit the QoS of its member disks and as such the data flow to the Raid must be kept within the operational limits of the member disks.

### *System Data Flows*

The data flow within a system is composed of the following flows

Host Data Flow	Read Data from Cache
	Write Data to Cache
	Read Data from Disk
	Write Data to Disk
Internal Data Flow	Migration of Data from one Raid to another (during an SOS operation)
	Replication of Data (copying data to/from a remote system)
	Internal data flowRebuild Data (rebuilding the parity of a Raid)

## SOS – Understanding the Value of Data

### *Host Data Flow*

Read and Write requests from a host server can only be sent to a Logical Volume (LV) device within the SOS system. This logical volume appears as a disk volume to the Host server.

An LV within an SOS framework is called a Tiered Logical Volume (TLV). A Tiered Logical Volume is a mapping between Logical Volume space and space on Raids in different tiers of the Disk Array. A TLV appears as one contiguous disk space to the Host, and this logical disk is mapped to space on a number of Raids. The Raid used in each TLV depends on the QoS required by the user for the Volume.

A Volume that is a mission critical volume could be configured to use storage space from only the Highest QoS Raid Tier of the Array, while a Volume with mixed data may be configured to use space from several QoS Tiers.

### *TLV Formulation*

Each TLV address is divided into two components, a RAID\_PE (PE) and an LVM Chunk.

The PE is called a Physical Extent. The size of each Physical Extent is a property of the TLV and is nominally set at 64K 512Byte blocks. This means the first 16 bits of every TLV LBA address is used as an OFFSET within the base address pointed to by the LVM Chunk.

The LVM Chunk is used as an index into a table of R\_ID/R\_BA pairs. The table is called the TLV Map. The number of entries of the TLV Map is equal to the size of the TLV in bytes divided by 32Megabytes.

The R\_ID is an ID, which identifies which Raid the R\_BA is mapped to.

The R\_BA is the base address which is used with the RAID\_PE to determine where the LVM chunk is pointing to in Raid space. The LVM Chunk to R\_ID/R\_BA and RAID\_PE mapping is shown in Figure 6.

### *Process Overview*

As discussed above, storage on a Tiered Logical Volume is organised in chunks of data. For each logical volume chunk there is a corresponding segment on a Raid that provides the actual storage for this chunk. Each Raid has an associated QoS value that is inherited from its constituent disks. This QoS rating can be converted into absolute *hit values*: the total number of times a Raid is accessed must not exceed a predefined limit, and this limit is derived from its disks' duty cycle characteristics.

The manager enforces the restriction that a disk does not exceed its allowed duty cycle by enforcing the corresponding restriction that the total number of accesses on a Raid does not exceed its hit limit in a given time period.

## SOS – Understanding the Value of Data

Each Logical Volume segment must map to a Raid segment, and these mappings are dynamically changed. QoS restrictions on each Raid are then enforced by means of a three-step process:

1. Based on past experience, predict the number of hits on each Logical Volume segment that will occur over the next time period.
2. Using these predicted hit counts, compute a new mapping between Logical Volume segments and Raid segments such that (a) the total number of predicted hits on any Raid does not exceed its designated hit limit, and (b) the difference between the new computed mapping and the current mapping is minimised.
3. Schedule migration operations for any segments that are moved to a different Raid under the new mapping to minimise disruption of incoming IO operations.

### *Segment Hit Prediction*

It is not sufficient to use the number of hits per logical volume segment in time period  $t$  to predict the number of hits on a particular segment in time period  $t + 1$ . A migration strategy based on this information would inevitably be myopic, resulting in undesirable behaviour such as migration thrashing, since data access patterns do not tend to be constant but rather follow statistical patterns.

For example, a particular logical volume segment may be accessed heavily every third day and not at all during the intervening two days. Adopting a reactive migration policy, in which segments are migrated based on the previous day's usage, would in this instance result in the segment being located on low QoS Raids during days of high usage and the segment being located on high QoS Raids for one of the low usage days.

This behaviour is clearly undesirable, and therefore we require an intelligent algorithm to discover statistical patterns in segment usage in order to *predict* periods of high and low usage and *pre-emptively* migrate data segments to the appropriate physical media. To continue the example above, we would pre-emptively migrate the segment in question to a high QoS Raid before its days of high usage and to a low QoS Raid for the days of low usage.

The example above, however, is contrived and only serves to illustrate the basic idea of *predictive* segment migration in contrast to *reactive* segment migration. One fundamental assumption underlies the possibility of predictive segment migration: segment access patterns tend to follow regular, discoverable, statistical patterns. For instance, logical volume segments may experience:

- Constant hit rates
- Linearly increasing/decreasing hit rates
- Exponentially increasing/decreasing hit rates
- Periodically varying hit rates

## SOS – Understanding the Value of Data

Using statistical time-series analysis we can accurately predict the number of hits on each segment for the forthcoming time period. This data is the primary input for the segment mapping generator algorithm, which pre-emptively migrates data to the appropriate medium, based on the forecasted hit rate.

### *Segment Remapping Algorithm*

The segment remapping algorithm computes a new mapping between logical volume and Raid segments such that the total predicted hit count for each Raid does not exceed its specified hit-limit and the difference between the new and old mappings is minimised.

By minimising the difference between the new and old mappings we can fulfil important optimisation criteria. Minimising the number of migrations that need to be undertaken to fulfil the QoS requirements conserves SAN bandwidth and disk usage as well as promoting the stability of the mapping.

Once a mapping meeting these requirements has been computed, the mapping is then passed onto the migration manager, which schedules the migration tasks intelligently ensuring that normal operation of the system is not disrupted by the migration tasks.

More specifically, the segment remapping algorithm takes as input the set of predicted hit values for each logical volume segment, the current mapping between logical volume and Raid segments, and the hit limits for each individual Raid. The output of the algorithm is a new mapping between logical volume and Raid segments that is guaranteed to meet the QoS requirements for each Raid and do so with the minimum number of migrated blocks.

There may be many possible mappings that meet these criteria and so tie-breaking heuristics are used to make intelligent (and configurable) choices between alternative mappings. For example, we may specify a policy that higher QoS Raids should be filled first and low hit-rate segments migrated as more high QoS space is required (Top Down), or alternatively, segments should begin on low QoS Raids and be migrated upwards as demand increases (Bottom Up).

In the case that there is no mapping satisfying the QoS requirements of all Raids in the system there is also a range of heuristic policies that can be chosen from.

For example, we may wish to specify the policy that particular logical volumes should be brought to within the correct QoS limits at the further expense of other logical volumes. Alternatively, we may wish to specify that the QoS costs should be borne equally among all Raids, ensuring that the excessive wear is distributed equally among all disks in the system. Many different heuristic policies for preferential segment fitting are possible, and a general mechanism for supplying such heuristics is an integral part of the segment remapping algorithm.

## SOS – Understanding the Value of Data

### Formulation

The problem of generating a mapping matching the requirements outlined above can be formulated mathematically as follows. A partition of a set  $A$  is an expression of  $A$  as a union of nonempty, disjoint subsets. Thus, a mapping between Logical Volume segments and Raid segments is some partition of the set of logical volume segments, each subset corresponding to a particular Raid. If we let  $P$  be the set (or more precisely, multiset) of predicted hit values for Logical Volume segments, then the required output of our mapping algorithm is a partition of  $P$  such that we have exactly  $k$  subsets and the sum of each subset does not exceed  $m_j$ , where  $k$  is the number of Raids allocated and  $m_j$  is the maximum number of hits for Raid  $j$ .

More formally, let  $P = \{p_1, \dots, p_n\}$  be a multiset of  $n$  non-negative integers and let  $R = \{R_1, \dots, R_k\}$  be a set of  $k$  multisets of non-negative integers such that

$$\bigcup_{j=1}^k R_j = P \text{ and } \bigcap_{j=1}^k R_j = \emptyset.$$

Furthermore, let  $m = \{m_1, \dots, m_k\}$  be a multiset of positive integers. The problem of mapping Logical Volume segments to Raid segments is then transformed into the following. Given  $R = \{R_1, \dots, R_k\}$  and  $m = \{m_1, \dots, m_k\}$ , compute  $R' = \{R'_1, \dots, R'_k\}$  such that

$$|R'_j| = |R_j| \text{ and } \sum_{r \in R'_j} r \leq m_j \text{ for } 1 \leq j \leq k.$$

Any set  $R'$  meeting these requirements is guaranteed to meet the required QoS criteria.

As an example, let  $P = \{1, 1, 4, 5, 6, 7, 8\}$  be the set of predicted Logical Volume segment demand values for the forthcoming time period. Then, suppose that under the current mapping between Logical Volume and Raid segments we have the following distribution of predicted demand values on Raids

$$R_1 = \{1, 8\} \quad R_2 = \{4, 7\} \quad R_3 = \{1, 5, 6\}$$

and suppose, furthermore, that the maximum demand values for these Raids are given by  $m_1 = 3$ ,  $m_2 = 10$  and  $m_3 = 20$ . Since the sum of Raids 1 and 2 exceed these maxima, we must compute a new mapping from Logical Volume to Raid segments to prevent the forecasted violation of the designated usage restrictions for these Raids. The multisets

$$R'_1 = \{1, 1\} \quad R'_2 = \{4, 5\} \quad R'_3 = \{6, 7, 8\}$$

meet our criteria above (i.e. the size of each new set is the same as the original, and the sum of the set  $R'_j$  is no more than its usage limit,  $m_j$ ).

## SOS – Understanding the Value of Data

We can also model the requirement that the number of migration operations is minimised as follows. If there is no set  $R''$  such that

$$|R_j''| = |R_j| \text{ and } \sum_{r \in R_j''} r \leq m_j \text{ for } 1 \leq j \leq k$$

and

$$\sum_{j=1}^k |R_j'' \cap R_j| > \sum_{j=1}^k |R_j' \cap R_j|$$

we are assured that the mapping corresponding to the solution  $R''$  is globally optimal in terms of the number of migration operations. Since the current mapping is given by the set of multisets  $R$  we can quantify the number of migration operations that will be needed to implement the new mapping by considering the sum of the cardinalities of the intersection of each set  $R_j$  from the original mapping with the set  $R_j'$  from the newly computed mapping. The larger the intersections of these sets, the more the sets have in common and hence the fewer migration operations required to implement the corresponding mapping.

### Algorithm

The previous section demonstrated that we could regard the problem of generating a new mapping between Logical Volumes and Raids in terms of generating subsets of the set of predicted usage values, in which the sum of each subset is no more than some predefined value.

The algorithm to solve this problem operates by swapping large values from sets that exceed their required sum with small values from sets that are less than the required sum, and iteratively applies this idea until a solution is found. This is illustrated in Fig 7, where we see three sets of integers  $R_1$ ,  $R_2$  and  $R_3$ , corresponding to Raids 1, 2 and 3.

For each Raid set there is a specified limit,  $m_1$ ,  $m_2$  and  $m_3$  respectively. The integers here refer to the predicted hit counts for particular Logical Volume segments.

Thus, we have predicted that two segments on Raid  $R_1$  will receive five hits, two segments will receive three hits and one segment will receive only one hit.

Summing these hit counts for each Raid we can see that Raid  $R_1$  has a forecasted QoS capacity surplus and  $R_3$  has a forecasted QoS capacity deficit. This means that, under the current mapping between Logical Volume and Raid segments we have predicted that the disks in Raid 3 will exceed their designated QoS access limit.

We have also predicted, however, that Raid  $R_1$  will be well within its operational limits. We can therefore resolve the capacity deficit in  $R_3$  by swapping the block

## SOS – Understanding the Value of Data

of size 5 in R3 with the block of size 1 in R1; the result of this operation is given in Fig. 4, where all Raids are within their designated QoS capacity limits.

The block remapping algorithm operates by iteratively applying this step until all of the capacity deficits have been resolved. The distance of any particular solution from the optimum may also be quantified, and so we can also terminate iteration once a solution of the required quality has been found. In the case that the total capacity surplus is less than the total capacity deficit, no solution exists, and so more space on the high quality Raids must be allocated. The means by which this is obtained is a user-specifiable policy.

### *Migration Manager*

Once the segment-remapping algorithm has computed the new Logical Volume segment to Raid segment mapping, this new mapping is passed to the migration manager. The migration manager ensures that the normal operation of the system is not affected by the process of data migration by monitoring system activity and intelligently scheduling migrations to and from nearby Raids to minimise SAN bandwidth and disk utilisation.

### Summary

It is clear that the block-based Tiered Storage management described above has many advantages. Data is automatically analysed at the *block* level to determine its value and hence the most appropriate storage medium. By exporting raw logical volumes over the Storage Area Network rather than file systems, the storage management is decoupled from file system issues. Furthermore, since block-level access patterns are analysed without knowledge of file structure the granularity problem discussed above is also solved --- data can be analysed at any level of granularity desired to ensure optimal usage of storage at the various price points and levels of functionality.

Another important advantage of the invention is the ability to perform *preemptive* migration of data to the most appropriate medium for forecasted usage. Data access frequencies are rarely constant but instead follow statistical patterns which may be predicted by analysing block access patterns. Using these predictions we can then ensure that blocks of data reside on the most appropriate physical medium and optimal usage of storage resources is obtained.

### About the Author

William Oppermann has extensive experience in the ICT industry and in particular the enterprise data storage industry has worked in new product development for over 20 years. William holds a B.E as well as a Msc. Eng. masters degree in Engineering as well as degree in International sales from DIT. His career includes eight years as VP of Engineering at Raidtec Corporation starting in 1995, developing multiple generations of NAS and SAN systems. William also spent 8 years in France as CEO and founder of consulting company AWAP SA and Technical Director of Telecoms Company SAT-ATLANTIS SA, both of these companies developed leading edge telecoms hardware and software.

## MPSTACKWARE

**MPStackware** provides a full featured Linux based high availability stack compatible with Virtualized hardware frameworks such as XEN and VMWARE.

**MPStackware** is a unified storage stack and supports multiple host connections such as SAS, Fiber Channel, iSCSI and NAS protocols such as CIFS.

**MPStackware** is a Storage Software Management suite that runs on both proprietary and industry standard Intel platforms allowing the OEM complete flexibility in choosing his hardware platform.

**MPStackware** is capable of scaling from low to high end enterprise class servers and allows an OEM to add **MPStackware** to their own hardware to create a full storage management solution.

**MPStackware** is fully supported by a range hardware reference designs allowing an OEM to create a complete hardware storage solution based on MPSTOR technology.

### → Contact Us

Name: William Oppermann

Address: MPSTOR Ltd  
Rubicon Centre  
CIT Campus  
Cork  
Ireland.

Email: [wo@mpstor.com](mailto:wo@mpstor.com)

Website: [www.mpstor.com](http://www.mpstor.com)

Tel: 353 86 8045403

Copyright ©2009 MPSTOR Ltd. All Rights Reserved.

MPSTOR logos, and trademarks or registered trademarks of MPSTOR Ltd or its subsidiaries in Republic of Ireland and other countries.

Other names and brands may be claimed as the property of others. Information regarding third party products is provided solely for educational purposes. MPSTOR Ltd is not responsible for the performance or support of third party products and does not make any representations or warranties whatsoever regarding quality, reliability, functionality, or compatibility of these devices or products.