



IBM Solutions Technical Brief

**Configuring IBM TotalStorage
for Oracle OLTP Applications**

**Dale Martin
IBM Oracle Solutions
Advanced Technical Support**

**Version: 2.0
Date: 07 July 2005**



1. INTRODUCTION.....	3
2. REMINDERS.....	4
3. TRADEMARKS.....	4
4. FEEDBACK.....	4
5. ESS CONFIGURATION OPTIONS.....	5
5.1. <i>Disk Size and Speed.....</i>	<i>5</i>
5.2. <i>Subsystem Capacity.....</i>	<i>5</i>
6. AIX AND RDBMS SETUP/TUNING	6
7. ESS SETUP	6
7.1. <i>ESS Array Formatting Options</i>	<i>6</i>
7.2. <i>RAID-5 vs. RAID-10.....</i>	<i>7</i>
7.3. <i>Arrays Across Loops (AAL).....</i>	<i>9</i>
7.4. <i>Deciding on a Logical Unit Number (LUN) Size Standard.....</i>	<i>9</i>
7.5. <i>Determining Data Isolation Requirements.....</i>	<i>10</i>
7.6. <i>Creating LUNs.....</i>	<i>11</i>
8. SERVER SETUP	13
8.1. <i>Subsystem Device Driver (SDD) Setup</i>	<i>13</i>
8.2. <i>Create Volume Groups</i>	<i>15</i>
8.3. <i>Create Logical Volumes</i>	<i>19</i>
8.4. <i>Create Data Files.....</i>	<i>20</i>
9. VARIATIONS ON THE “STRIPING AND SPREADING” THEME	21
9.1. <i>The SAME (Stripe and Mirror Everything) Methodology.....</i>	<i>21</i>
9.2. <i>Oracle10g Automatic Storage Management (ASM).....</i>	<i>21</i>
9.3. <i>IBM General Parallel File System (GPFS).....</i>	<i>22</i>
9.4. <i>“Striping and Spreading” with other I/O Subsystems</i>	<i>23</i>
10. CONCLUSIONS	26
11. REFERENCES.....	26
12. ACKNOWLEDGEMENTS	27

1. Introduction

Most of this paper focuses on Oracle data layout on the IBM Enterprise Storage Server (ESS) or “Shark”. The same techniques discussed for ESS are applicable to other IBM TotalStorage products as well as some competitive offerings.

The ESS has gained widespread acceptance in the marketplace. This is due in part to its industry leading performance, advanced functions such as FlashCopy and PPRC (Peer-to-Peer-Remote-Copy) and very attractive Total Cost of Ownership (TCO).

However, as with all I/O subsystems, good planning and data layout can make the difference between having excellent I/O throughput and application performance, and having poor I/O throughput, high I/O response times and correspondingly poor application performance. It therefore isn’t surprising that first-time ESS customers ask for advice on how to optimally layout their ESS subsystem.

In many cases, I/O performance problems can be traced directly to “hot” files that cause a bottleneck on some critical component (e.g. a single physical disk). This can occur even when the overall I/O subsystem is fairly lightly loaded. When bottlenecks occur, Storage or Data Base Administrators may have to identify and manually relocate the high activity data files that were contributing to the bottleneck condition. This tends to be a very resource intensive and often frustrating task. As the workload content changes in concert with the ebb and flow of normal business cycles (e.g. hour by hour through the business day or day by day through the accounting period), bottlenecks may mysteriously appear and disappear or migrate over time from one datafile or device to another.

Generally, I/O (and therefore application) performance will be best when the I/O activity is evenly spread across the entire I/O subsystem. This paper offers a straightforward, nearly foolproof technique for achieving highly balanced I/O activity across the entire ESS subsystem – thereby optimizing overall I/O performance. This technique requires minimal knowledge of the specific application(s) to be deployed or their underlying I/O characteristics and virtually eliminates the need for manual data file placement or ongoing I/O performance tuning. Conceptually, the technique simultaneously employs two separate but complementary I/O balancing strategies:

- The use of hardware RAID-5 and/or RAID-10 technology within the ESS subsystem. The fine granularity “striping” inherent to the RAID-5 and RAID-10 algorithms results in highly balanced I/O activity across all of the physical disks in the RAID array.
- The use of AIX Logical Volume Manager (LVM) to physically partition data files across multiple ESS arrays, thereby “spreading” I/O activity across all the arrays in the ESS subsystem.

This “striping and spreading” strategy has been used in a number of large online Enterprise Application Solutions implementations with excellent results.



While this paper's focus is on Enterprise Application Solutions (EAS) applications, this technique may be used successfully for many other types of applications. Although, no single technique works best for all possible workload situations, the one described here should perform satisfactorily for most customer workloads.

This paper is oriented toward AIX/pSeries server platform. However, this technique may also be adapted to other platforms that support a functionally rich Logical Volume Manager (LVM). While Oracle RDBMS (Relational Data Base Management System) is occasionally referred to, these general techniques are also applicable to DB2 UDB, Informix or other RDBMS environments.

2. Reminders

Copyright 2003/2005 IBM Corporation. All Rights Reserved.

Neither this documentation nor any part of it may be copied or reproduced in any form or by any means or translated into another language, without the prior consent of the IBM Corporation. The information in this paper is provided by IBM on an "AS IS" basis. IBM makes no warranties or representations with respect to the content hereof and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. IBM assumes no responsibility for any errors that may appear in this document. The information contained in this document is subject to change without any notice. IBM reserves the right to make any such changes without obligation to notify any person of such revision or changes. IBM makes no commitment to keep the information contained herein up to date.

Version 2.0, published July 7, 2005

3. Trademarks

IBM, RS/6000, pSeries, TotalStorage, and Enterprise Storage Server are trademarks or registered trademarks of the International Business Machines Corporation.

Oracle, Oracle8, Oracle8*i* and Oracle9*i* are trademarks or registered trademarks of Oracle Corporation.

UNIX is a registered trademark in the United States and other countries exclusively through X/Open Company Limited.

All other products or company names are used for identification purposes only, and may be trademarks of their respective owners.

4. Feedback

Please send comments or suggestions for changes to dalem@us.ibm.com

5. ESS Configuration Options

When configuring/ordering an ESS subsystem, there are a number of choices or options that can affect the potential performance or I/O capacity of the subsystem. It is recommended that you work with your IBM TotalStorage sales representative or IBM Business Partner to determine what specific ESS configuration will meet your detailed performance requirements. This section provides a high level overview of some of those options, but should be considered as an introduction to the issues, rather than specific advice for your particular environment.

5.1. Disk Size and Speed

Storage vendors and customers are jointly beating an unmistakable path to higher capacity disks. The reasons are simple: lower cost, less floor space, and easier to manage. Each generation of disks typically doubles in capacity. These larger capacity disks attract more data, more users, more I/O operations per second, and potentially greater queuing delays. Despite all of these cautions, the historical trend continues. It all works because:

- ***Disks keep getting faster.*** Higher rotation speeds and increased density usually improve performance of each generation approximately 30%.
- ***Disk subsystems keep getting faster.*** larger caches, higher bus speeds, striping, and better attachment technology all contribute.
- ***Users change behavior.*** Less expensive storage attracts more types of application data, much of which has never been “online”, and much of which has lower access requirements.

Currently, there are several available ESS disk options; 18.2, 36.4, 72.8 or 145.6 GB capacity 10k rpm drives, and 18.2, 36.4 or 72.8 GB capacity 15k rpm drives.

Performance of the 10k rpm 72.8 GB drives is acceptable for the majority of customer workloads and they offer the most attractive choice for most customers, due to their ability to deliver large storage capacity in a small footprint. The 15k rpm 36.4 and 72.8 GB drives offer higher performance and are suitable for applications with relatively high access density (I/Os per second per gigabyte) and/or critical I/O response time requirements.

The largest capacity 145.6 GB drives might be suitable in the following circumstances:

- For less demanding database applications with a low I/O access density requirement (< .7 I/Os per second per gigabyte).
- For applications with very heavy sequential content, such as decision support applications. Sequential workloads tend to stress other components of the disk subsystem (e.g., fibre interfaces, internal data paths, etc.) in comparison to the effects of disk contention.
- Combined with RAID-10. This may provide a reasonable cost compromise in circumstances in which RAID-10 is the customer’s preferred choice.

5.2. Subsystem Capacity

Disk subsystem capacities have historically been increasing, for most of the same reasons as the disks themselves. For most usage scenarios for today’s applications, the “sweet spot” for the ESS 800 is around 6.7 TB of usable capacity (128 72.8 GB disks in a RAID-5 configuration) per

subsystem. A 6.7 TB configuration should provide acceptable performance for applications with up to twice the industry average access density of about .6 I/Os per second per GB. A 6.7 TB ESS 800 is therefore a reasonable “starting point” for most customers looking for the best overall balance between performance, capacity and price/performance. Not surprisingly, the average shipped capacity per ESS 72.8 GB subsystem exceeds 5 TB.

Example

Throughout this paper, we will be developing an ESS implementation scenario to demonstrate the “striping and spreading” technique. Information regarding this scenario will be contained within shaded boxes (like this one) scattered throughout the paper.

For our example, let’s assume that we have ordered a basic “starting point” subsystem – an ESS 800 with 128 10k rpm 72.8 GB disks.

6. AIX and RDBMS Setup/Tuning

There are a number of AIX tuning parameters that can affect I/O activity and performance, such as those having to do with asynchronous I/O and Journaled File System (JFS or JFS2) cache. Database specific parameters, such as the size of the database buffer cache, log file cache and database block size can also affect I/O performance.

Refer to the appropriate database documentation for recommended AIX and database parameter settings. For Oracle database, a good starting point is the Administrator’s Reference document. (For example, see Appendix A of the “Oracle9i Administrator’s Reference”.)

7. ESS Setup

7.1. ESS Array Formatting Options

ESS storage is typically organized into sets of 8 disks. Disks are installed in “eight packs”, which are Field Replaceable Units (FRUs) containing 8 Disk Drive Modules (DDMs). RAID arrays are also organized into sets of 8 disks, though not mapping directly to 8-packs, and may be configured in one of four possible ways (depending on the ESS model and physical location of the disk eight pack within the ESS):

RAID-5 (all ESS models)

- 6+P+S (7 disks including parity with 1 spare disk)
- 7+P (8 disks including parity)

RAID-10 (ESS 800 only)

- 3+3 (3 primary disks, 3 mirror disks and 2 spare disks)
- 4+4 (4 primary disks and 4 mirror disks)

In the ESS 800, the customer has control over the array type (RAID-5, RAID-10) and array types can be intermixed. In the older E and F models, only RAID-5 is supported. ESS requires at least two spare DDMs per internal SSA loop. The number of disks used in a particular array is dependent on the array order within a loop. For RAID-5, the first two arrays defined on a loop must be 6+P configurations. For RAID-10, the first array defined on a loop must be a 3+3 configuration. RAID-5 and RAID-10 arrays may also be intermixed on the same loop. If the first array defined on a loop is RAID-5 and the second array defined on the loop is RAID-10, there will be 3 spares on the loop.

7.2. RAID-5 vs. RAID-10

Historically, database vendors and consultants have tended to warn against using RAID-5 technology for applications having a lot of update activity. . Several factors have contributed to this position, which include:

- The classic RAID-5 “write penalty”, which refers to the fact that up to 4 internal I/Os are required for each external I/O request:
 1. read old data block
 2. read old parity block
 3. write new data block
 4. write new parity block

- Poorly-optimized implementations of RAID-5 in open systems environments. Some of these implementation lacked write cache, failed to stripe the data across disks, or were sometimes implemented in software.

However, in most cases, this “write penalty” does not affect application performance when using an ESS RAID-5 configuration. ESS normally has 100% cache write hits. This means that ESS returns a “write complete” status to the server as soon as the data block has been written to cache and to Non Volatile Storage (NVS). Therefore, as long as the backend (physical) disk configuration can support the sustained I/O rate required by the application, write I/O performance is not determined by the RAID-5 or RAID-10 choice.

Another “classic” recommendation is to never place RDBMS log files on RAID-5 devices because the RAID-5 design “cannot support high sequential write activity”. The ESS (as in all other IBM hardware RAID-5 capable storage offerings) uses a “full stripe” parity generation algorithm that eliminates the need to read existing data or parity in order to generate the new parity information. Whereas a “classic” RAID-5 implementation may require 4 physical I/Os per logical I/O write request, the ESS “full stripe” RAID-5 write requires at most 1.17 physical I/Os per logical I/O write request (1 parity write for every 6 data writes). For high volume sequential write applications, this is an advantage for RAID-5 over RAID-10 – which may require 2 physical writes (1 to each copy) per logical write request.

Note: JFS and JFS2 file systems intersperse inode data with the actual file data. Depending on how frequently inode blocks occur, this can potentially disrupt “full stripe” write activity within the ESS. This potential issue can be avoided by placing files with heavy sequential write characteristics (e.g. redo log files) on raw devices.

As the following table shows, RAID-5 provides for a substantially higher effective capacity than RAID-10 - given an equivalent number of physical disks.

Effective Eight Pack Capacity in Gigabytes (10⁹)

Disk Size →	18.2 GB	36.4 GB	72.8 GB	% Effectiveness
Physical eight-pack capacity	145.60	291.20	582.40	
RAID-5 (6+P)	105.20	210.45	420.92	72%
RAID-5 (7+P)	122.74	245.53	491.08	84%
RAID-10 (3+3)	52.50	105.12	210.39	36%
RAID-10 (4+4)	70.00	140.16	280.52	48%

Note: The ESS E and F models also support JBOD (Just a Bunch Of Disks) configurations. JBOD support is not available on the ESS 800. JBOD does not provide any data protection against disk failures or support the automated I/O balancing that is inherent to RAID-5 and RAID-10 arrays. For these reasons, we **DO NOT** recommend the use of JBOD.

The following table summarizes the major RAID-5 vs. RAID-10 attributes.

RAID-5 vs. RAID-10 Comparison

	RAID-5	RAID-10
Sequential Read	Excellent	Excellent
Sequential Write	Excellent	Good
Random Read	Excellent	Excellent
Random Write	Fair	Excellent
\$ per MB	Excellent	Fair

For most customer workloads, RAID-5 provides comparable performance to RAID-10, with considerably lower cost per Megabyte. This makes RAID-5 the default choice for most applications. RAID-10 should be considered for workloads with a high percentage of random write activity and high I/O access densities. There is no “one size fits all” answer, but a basic rule-of-thumb would be to consider using RAID-10 if the random write content exceeds 25% and the peak sustained I/O rate is expected to exceed 50% of the array capacity. When in doubt, your IBM Storage rep or business partner can help model your workload and recommend a choice appropriate to your workload.

Example Cont'd...

We will be using RAID-5 arrays. With a total of 16 disk eight packs (128 disks) on 8 internal SSA loops, all RAID-5 arrays will be of the 6+P type in order to satisfy the requirement of having two spare drives per loop.

7.3. Arrays Across Loops (AAL)

Arrays Across Loops (AAL) is an optional feature (#9903) on the ESS 800 whereby disk arrays are spread across both loops on the SSA device adapter pair. AAL configurations can support up to a 1.8 times improvement in sequential bandwidth performance on a single array as compared to non-AAL configurations. This can potentially benefit Oracle sequential operations, such as table scans and database backup/recovery activities. AAL does require 8-packs to be installed in multiples of 4 (initial capacity as well as upgrades), so it may not be appropriate for some environments with relatively small storage capacity requirements.

7.4. Deciding on a Logical Unit Number (LUN) Size Standard

With ESS, the minimum configurable LUN size is 0.1 GB and the maximum is the total effective capacity of the RAID array that the LUN is defined on. In most cases, the choice of LUN size has minimal effect on performance. However, in an effort to simplify Storage Administration tasks, customers may wish to adopt a LUN size standard. This allows LUNs to be allocated and subsequently de-allocated and re-allocated in an orderly fashion, without wasting space. A consistent LUN size is also a key component of the “striping and spreading” technique.

Since the usable capacity of any ESS rank is some multiple (6, 7, 3 or 4) of the disk size, using the physical disk size (roughly) as the standard LUN size allows for efficient allocation of the available ESS disk capacity, even when multiple array configurations are used. This would equate to a LUN size of 17.5 GB (for 18.1 GB drives), 35.0 GB (for 36.2 GB drives), 70.0 GB (for 72.8 GB drives) or 140.0 GB (for 145.6 GB drives).

When there is a mix of physical disk sizes in your environment, consider basing LUN size on the size of the smallest disk. Or, for environments where ESS storage is shared across a large number of relatively small servers and smaller allocation units are desirable, consider some fraction of the disk size – such as 11.6, 14.0, 17.5, 23.3, or 35.0 GB (when using 72.8 GB drives). It is also possible to define more than one LUN size standard for an enterprise (e.g. 70.0 GB for large environments and 14.0 GB for small environments). Having multiple standard LUN sizes somewhat increases the complexity of the storage management task, but may provide for somewhat more efficient storage allocation if properly managed.

Note: AIX and ESS use different metrics for representing disk or memory capacity:

- ESS uses base 10 definitions of “Gigabytes” (abbreviated GB) = $10^9 = 1,000,000,000$ bytes and “Megabytes” (abbreviated MB) = $10^6 = 1,000,000$ bytes.
- AIX uses base 2 definitions of “Gibibytes” (abbreviated GiB) = $2^{30} = 1,073,741,824$ bytes and “Mebibytes” (abbreviated MiB) = $2^{20} = 1,048,576$ bytes as defined in the defined in the IEC 60027-2 standard.

Therefore, the “apparent” size of a given LUN will be smaller in AIX than it is on the ESS. For example, a 70.0 GB (70×10^9) ESS LUN will appear to AIX as roughly 65 GiB (65×2^{30}) in size. Throughout this paper, we use MB or GB to refer to base 10 values (10^6 or 10^9) and MiB or GiB to refer to base 2 values (2^{20} or 2^{30}).

As we will further discuss in the “**8.2 Create Volume Groups**” section, the AIX Logical Volume Manager (LVM) has limits on minimum Physical Partition size and the maximum number of Physical Partitions per Physical Volume. From an AIX LVM point of view, “optimal” LUN sizes are determined by the formula $n \times 1016 \times 1$ MiB (where $n = 1$ to 64, depending on a number of constraints). This “set” of possible LUN sizes is somewhat different from those suggested above. The AIX “optimum” LUN sizes may result in slightly better performance (through smaller Physical Partition sizes) and allow for Volume Groups with slightly larger capacity. However, since the ESS Physical Disk sizes are not evenly divisible by the optimum AIX LUN sizes, their use will typically reduce the effective ESS storage capacity a few percentage points. For example, with a 6+P array of 36.2 GB drives, it is possible to get six 35.0 GB LUNs for a total capacity of 210 GB, or six 34.0 GB LUNS ($n=32$) for a total capacity of 204 GB – a 3% difference.

Example Cont’d...

We will use physical drive size as the “default” LUN size standard - 70 GB usable after sector formatting overhead. This allows for 6 LUNs per RAID-5 array. Since our ESS 800 subsystem has 16 arrays, we will a total of 96 LUNs. 96 LUNs of 70 GB each yields 6.72 TB of usable capacity.

7.5. Determining Data Isolation Requirements

In the past Oracle has recommended that several categories of RDBMS files be isolated from each other (e.g. separate redo logs from data, indexes from tables, isolate rollback segments etc.). However, isolating each class of data file on its own set of arrays can result in sub-optimization of the overall ESS subsystem performance. In general, the best overall performance can be achieved if storage is managed with the minimal number of physical data groupings or pools.

One non-performance related consideration is that it is generally recommended practice to isolate recovery related data (e.g. active redo logs, archive logs, database backups etc.) from the primary user data. The “Oracle9i Backup and Recovery Concepts Release 2 (9.2)” manual defines the “golden rule of backup and recovery” as “*the set of disks or other media that contain the redundancy set [i.e. the set of files needed to recover from the failure of any Oracle datafile] should be separate from the disks that contain the datafiles, online redo logs, and control files*”.

This strategy ensures that the failure of a disk that contains a datafile does not also cause the loss of the backups or redo logs needed to recover the datafile”.

The RAID-5 and RAID-10 architectures both protect against single disk failures within an array. Although exceedingly unlikely, double disk failures do occasionally occur. In this event, some or all of the data on that array could be lost. Therefore, if protection against double disk failures is critical, all recovery related data for a given database instance should be placed on separate ESS arrays (on separate ESS subsystems if practical) than the data being protected/backed up.

Since indexes can be rebuilt from the table data, it is not critical that index related .dbf files be physically isolated from the recovery related files.

Since the Oracle control files, online redo logs and archived redo logs are crucial for most backup and recovery operations, at least two copies of these files are normally maintained on different physical disks. And, ideally, both sets of these files should be physically isolated from the base user data.

Example Cont'd...

In order to provide protection against double disk failures, we will isolate all recovery related data on a separate set of ESS arrays than where the user data is stored. In addition, the Oracle control files, online redo logs and archived redo logs will be multiplexed, with the multiplexed copies physically isolated from each other as well as from the base user data.

7.6. Creating LUNs

Use the ESS Specialist to create LUNs of the “standard” size determined above. LUNs are typically assigned to the host (or multiple hosts in a RAC or OPS environment) that will use them at creation time. Subject to any data isolation requirements (see above section), LUN assignments should be done in such a way that at any given time, the assigned LUNs are evenly distributed across all of the disk groups (ranks) in the ESS – Assuming a 16 rank (128 disk) ESS subsystem, after 16 LUNs have been assigned, there should be one assigned LUN per array, after 32 LUNs assigned, there should be two assigned LUNs per array etc. Within the “evenly distributed” constraint defined above, LUNs should be assigned in a quasi-random fashion. A strict round-robin assignment can potentially result in undesirable “convoy” affects for certain types of sequential workloads.

If multiple servers are attached (either direct attach or via a SAN) to the same ESS, the LUNs assigned to any particular server should be even distributed across the ESS arrays.



Example Cont'd...

We will initially allocate LUNs for two separate Oracle instances, DB01 on ServerA and DB02 on Server B. DB01 requires 1,700 GB for table/index data and 2 x 100 GB for recovery related data and DB02 requires 1,350 GB for table/index data and 2 x 50 GB for recovery related data. Therefore, we will need 29 LUNs ($1,700/70 = 25 + 2 \times 100/70 = 4$) for DB01 and 22 LUNs ($1,350/70 = 20 + 2 \times 50/70 = 2$) for DB02.

The table below is a logical view of the ESS layout, showing the initial LUN assignment within the ESS. Remember, in our ESS configuration, there are 16 arrays, with six (6) 70 GB LUNs per array. The array designations (down the left side) are in Device Adapter Pair / Cluster / Loop notation. For example, 11B = Device Adapter Pair 1, Cluster 1, Loop B. Volume (LUN). The relative Volume (LUN) number within a given array are shown along the top.

Note that in order to isolate base data from recovery related data, volumes from Device Adapter Pair 3, Cluster 2, Loop A (and likewise Device Adapter Pair 2, Cluster 1, Loop A) are not used any DB01-Datxxx LUNs, since they already contain DB01_Recvxx LUNs. And, volumes from Device Adapter Pair 4, Cluster 2, Loop A (and likewise Device Adapter Pair 3, Cluster 1, Loop A) are not used any DB02-Datxxx LUNs, since they already contain DB02_Recvxx LUNs.

	001	002	003	004	005	006
42B	DB01-Data12	DB01-Data20	DB02-Data10			
42A	DB01-Data3	DB01-Data23	DB02-RecvA1			
32B	DB01-Data14	DB01-Data18	DB02-Data6			
32A	DB01-RecvA1	DB01-RecvA2	DB02-Data14			
22B	DB01-Data9	DB01-Data25	DB02-Data15			
22A	DB01-Data4	DB01-Data15	DB02-Data7	DB02-Data19		
12B	DB01-Data1	DB02-Data2	DB02-Data17			
12A	DB01-Data6	DB01-Data21	DB02-Data12			
41B	DB01-Data13	DB02-Data1	DB02-Data4	DB02-Data18		
41A	DB01-Data2	DB01-Data19	DB02-Data9			
31B	DB01-Data5	DB02-Data3	DB02-Data16			
31A	DB01-Data7	DB01-Data24	DB02-RecvB1			
21B	DB01-Data11	DB01-Data16	DB02-Data5	DB02-Data20		
21A	DB01-RecvB1	DB01-RecvB2	DB02-Data11			
11B	DB01-Data8	DB01-Data22	DB02-Data13			
11A	DB01-Data10	DB01-Data17	DB02-Data8			

- = Assigned to Server A (DB01)
- = Assigned to Server B (DB02)
- = Not Assigned

8. Server Setup

The “striping” affects of RAID-5 and RAID-10 provide effective I/O balancing **within** individual arrays in the ESS and can potentially offer better performance with less manual intervention than alternative non-striped I/O subsystems. However, poor data placement can still result in I/O activity that is not well balanced **across** all of the arrays in the ESS. If the I/O demand against a single array exceeds the I/O capacity of that array, poor overall performance can result, even when the other arrays in the ESS are underutilized.

The objective of the “spreading” strategy described in this section is to evenly balance I/O activity across all of the arrays in the ESS subsystem, without requiring detailed knowledge of application workload or specific data file i/o characteristics, and with minimal ongoing performance “tuning” requirements.

8.1. Subsystem Device Driver (SDD) Setup

SDD can be used to automatically load balance I/O activity across the available host I/O adapters and/or provide alternate paths if one (or more) of the available adapters fails. In most cases, the “spreading” strategy to be described here should provide for fairly well balanced activity across the host adapters with or without the use of SDD. However, SDD is still recommended for High Availability (HA) environments in order to avoid outages due to individual host adapter failures.

When multiple paths are defined for a single ESS LUN, each path will show up as a single AIX hdisk. The smitty “Define and Configure All Data Path Devices” function can be used to create vpath definitions. “vpath” is a logical grouping of hdisks, where each vpath represents all of the possible paths (hdisks) to a particular ESS LUN.

Multi-Path I/O (MPIO) support was introduced in AIX 5.2. For those I/O subsystems that support it (including ESS), the AIX MPIO feature may be used in place of proprietary multi-path software (e.g. SDD) provided by the storage vendor. This is particularly advantageous in heterogeneous storage environments where the proprietary multi-path software from one vendor may not be able to coexist on the same AIX server with another vendor’s multi-path software. ESS SDDPCM MPIO drivers for AIXL 5L v5.2 and later and the associated documentation are available at: <http://www-1.ibm.com/support/docview.wss?uid=ssg1S4000201>.

Note: The SDD driver is not compatible with the Automatic Storage Management (ASM) feature of Oracle 10g. This is because the vpath (represented by /dev/rvpathn) interface is only accessible by the root user even if the ownership and permissions are changed. Any one of the individual disk paths (hdiskn) that make up the vpath could be used in the ASM configuration but all the benefits of multipathing will be lost. Therefore, in order to provide multi-path support with ASM, the ESS SDDPCM MPIO driver should be used. Once installed, a single hdisk is created that represents all virtual paths to the hdisk and its corresponding /dev/rhdiskn interface is accessible by an oracle user. The /dev/rhdiskn interface name can then be used when configuring ASM.



Example Cont'd...

In our example, we will be using SDD, so vpath names will be assigned to each grouping of hdisks corresponding to a particular ESS LUN. When vpaths are assigned a group of newly assigned ESS LUNs, they are normally created in ESS Device Adapter Pair / Cluster / Loop / Volume order (as shown in the table on the next page). However, you should be sure to verify the vpath name assignments are as you expect.

	001	002	003	004	005	006
42B	SvrA vpath15 DB01-Data12	SvrA vpath28 DB01-Data20	SvrB vpath18 DB02-Data10			
42A	SvrA vpath14 DB01-Data3	SvrA vpath27 DB01-Data23	SvrB vpath17 DB02-RecvA1			
32B	SvrA vpath11 DB01-Data14	SvrA vpath25 DB01-Data18	SvrB vpath14 DB02-Data6			
32A	SvrA vpath10 DB01-RecvA1	SvrA vpath24 DB01-RecvA2	SvrB vpath13 DB02-Data14			
22B	SvrA vpath7 DB01-Data9	SvrA vpath22 DB01-Data25	SvrB vpath10 DB02-Data15			
22A	SvrA vpath6 DB01-Data4	SvrA vpath21 DB01-Data15	SvrB vpath9 DB02-Data7	SvrB vpath20 DB02-Data19		
12B	SvrA vpath3 DB01-Data1	SvrB vpath0 DB02-Data2	SvrB vpath6 DB02-Data17			
12A	SvrA vpath2 DB01-Data6	SvrA vpath18 DB01-Data21	SvrB vpath5 DB02-Data12			
41B	SvrA vpath13 DB01-Data13	SvrB vpath2 DB02-Data1	SvrB vpath16 DB02-Data4	SvrB vpath21 DB02-Data18		
41A	SvrA vpath12 DB01-Data2	SvrA vpath26 DB01-Data19	SvrB vpath15 DB02-Data9			
31B	SvrA vpath9 DB01-Data5	SvrB vpath1 DB02-Data3	SvrB vpath12 DB02-Data16			
31A	SvrA vpath8 DB01-Data7	SvrA vpath23 DB01-Data24	SvrB vpath11 DB02-RecvB1			
21B	SvrA vpath5 DB01-Data11	SvrA vpath20 DB01-Data16	SvrB vpath8 DB02-Data5	SvrB vpath19 DB02-Data20		
21A	SvrA vpath4 DB01-RecvB1	SvrA vpath19 DB01-RecvB2	SvrB vpath7 DB02-Data11			
11B	SvrA vpath1 DB01-Data8	SvrA vpath17 DB01-Data22	SvrB vpath4 DB02-Data13			
11A	SvrA vpath0 DB01-Data10	SvrA vpath16 DB01-Data17	SvrB vpath3 DB02-Data8			

	= ASSIGNED TO SERVER A (DB01)
	= Assigned to Server B (DB02)
	= Not Assigned

8.2. Create Volume Groups

Once vpath names (or hdisk names if SDD is not being used) have been assigned, one or more Volume Groups (VGs) can be created using those vpaths (or hdisks). This can be done using `smitty` or the “`mkvg`” or “`mkvg4vp`” AIX command.

If the recovery related data is to be isolated from the regular database data, one (or more) VG(s) should be created for the recovery related data and one (or more) VG(s) should be created for the database data. Generally, using a small number of VGs will provide more opportunity for overall ESS subsystem performance than using a large number of VG. However, AIX Logical Volume Manager (LVM) restrictions on the number of Physical Volumes (PVs) in a volume group, Physical Partition size, etc. may more or less require the use of multiple VGs for large database environments. If multiple VGs are required, try to place several vpaths (or hdisks) in each Volume Group (preferably one vpath/hdisk from every ESS array or rank).

AIX supports multiple Volume Group types:

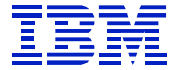
- **Default** VGs are created with the `mklv` or `mkvgr4vp` command can accommodate up to 255 Logical Volumes (LVs) and 32 Physical Volumes (PVs).
- **Big** VGs are created by using the “-B” flag on the `mkvg` or `mkvg4vp` command. They will support up to 512 LVs and 128 PVs.
- **Scalable** VGs were introduced in AIX 5.3 and are created by using the “-S” flag on the `mkvg` or `mkvg4vp` command. They can accommodate up to 1024 PVs and 256 LVs by default. These limits may be increased via the `mklv` “-v” and “-P” options.

For Oracle, we recommend the use of either **Scalable** or **Big** VGs, particularly when Raw Devices are to be used. See the discussion on Physical Partition size below as well as the “8.3.1 The AIX Logical Volume 4K Offset” section for more information.

When creating the LVs (see next section), we will be “spreading” each LV across as many ESS LUNs (vpaths or hdisks) as possible. In order to do that optimally, we want to set the Physical Partition (PP) size for the VG as small as possible (using the `mkvg` or `mkvg4vp` “-s” option), given the size and number of PV (vpaths or hdisks) in the VG. PP spreading works very well for most OLTP workloads.

For single stream sequential I/O (for example, a single connected user doing a tablespace scan) performance will be limited to the capacity of a single ESS array. If higher single stream sequential throughput is required, this can be accomplished through the use of LVM small granularity striping (512K or smaller stripe size) rather than PP spreading. However, caution should be used when doing LVM small granularity striping over ESS RAID-5 or RAID-10 arrays, because the LVM striping can potentially defeat the ESS sequential detect/prefetch cache management algorithms.

In AIX 5.2 and 5.3, there are a number of LVM enhancements that may make the use of LVM striping more desirable, even for OLTP workloads. See the “8.2.1 Physical Partition Spreading vs. LVM Striping” section for more details.



For Standard and Big VGs, there is a default limitation of 1016 PPs per PV. Therefore, in order to fully utilize the space on a given PV, the PP size must be $\geq PV / 1016$. However, if a VG is going to contain less than the maximum number of PVs (32 for Default or 128 for Big), it may be possible to adjust the “factor value” (mkvg or mkvg4vp “-t” option) to allow for more than 1016 PPs per PV.

The following table shows the optimal “-s” and “-t” settings for some possible PV (LUN) sizes and number of volumes in the VG. It is always a good idea to leave room in every VG for at least one additional PV to be added at a later time.

# PVs	Factor	PP size for given LUN size					
		4.3 GB	8.7 GB	17.5 GB	35.0 GB	70.0 GB	140.0 GB
1-2	64	1	1	1	1	2	4
3-4	32	1	1	1	2	4	8
5-8	16	1	1	2	4	8	16
9-16	8	1	2	4	8	16	32
17-32	4	2	4	8	16	32	64
33-64	2	4	8	16	32	64	128
65-128	1	8	16	32	64	128	256

Scalable VGs support up to 32,768 PPs by default, although a maximum limit of 32,512 PPs per LV still applies. The “-t” (factor) option is ignored when creating Scalable VGs, so the above table is not applicable to Scalable VGs.

Example Cont'd...

For DB01, we will create one VG for each copy of the recovery related data and one VG for the remaining database data. Each recovery related VG will contain 2 Physical Volumes. Allowing for future growth, we will use a “factor” of 32 and a PP size of 4 MiB.

```
mkvg4vp -B -t 32 -s 4 -y DB01_RECOV_VGA vpath10 vpath24
mkvg4vp -B -t 32 -s 4 -y DB01_RECOV_VGB vpath4 vpath19
```

The other VG for DB01 will contain 25 Volumes. Allowing for future growth, we will use a “factor” of 4 and a PP size of 32 MiB. Note that if we expected to have a large number of small heavily accessed tables, we may want to create multiple VGs so that the PP size could be reduced to allow for more of the small tables to be spread across multiple volumes in the volume group.

```
mkvg4vp -B -t 4 -s 32 -y DB01_DATA_VG1 vpath15 vpath14 vpath11 vpath7
vpath6 vpath3 vpath2 vpath13 vpath12 vpath9 vpath8 vpath5 vpath1 vpath0
vpath18 vpath21 vpath22 vpath25 vpath27 vpath28 vpath16 vpath17 vpath20
vpath23 vpath26
```

We won't show the details here, but we would create similar VG for the DB02 database.

8.2.1. Physical Partition Spreading vs. LVM Striping

When choosing a PP size or LVM stripe size for VGs containing RAID-5 or RAID-10 based LUNs, it is generally a good idea to choose a value that is several times the stripe size used in the underlying RAID array – typically, 1M or larger. The one major exception to this would be for single-threaded sequential I/Os streams with a throughput capacity greater than the throughput capacity of a single RAID array. In that case, small granularity LVM striping (e.g. 64K or 128K) offers the ability to spread that single-threaded I/O stream across multiple physical RAID arrays and potentially improving throughput performance. This technique is generally discouraged for most other workloads, because fine granularity LVM striping can potentially impair the ability of the ESS to detect and optimize for sequential I/O workloads.

Up to and including AIX 5.1, LVM only supported stripe sizes of up to 128K. This means that historically, PP spreading was the only effective way of meeting the desired goal of a 1M or larger stripe size. This has changed somewhat in more recent AIX releases. In AIX 5.2, stripe sizes of up to 1M are supported and in AIX 5.3, stripe sizes of up to 128M are supported.

These recent LVM enhancements make it much easier and attractive to implement the “striping and spreading” technique with LVM striping and, in some cases, LVM striping may now be preferred over PP spreading. One potential advantage of LVM striping is the ability to choose a relatively small stripe size (e.g. 1M). With PP spreading, the minimum allowable PP size is a factor of the size and number of Physical Volumes in the VG. It isn't at all uncommon to see minimum PP sizes in the 16MiB, 32MiB, or larger range. If the PP size is

too large, it can reduce the overall effectiveness of the “striping and spreading” technique – the larger the PP size, the less opportunity to spread small, “hot” objects across all the available arrays. LVM striping now allows us to pick a stripe size that is large enough (e.g. several times the ESS RAID array stripe size), but not “so large” as to reduce the overall effectiveness of the spreading technique.

One potential advantage of PP spreading over LVM striping is that if the LV (and underlying VG) needs to be extended, any number (1 or more) PVs can be added to the VG. A ‘reorgvg’ can then be performed to redistribute PPs for the LVs within that VG. As of AIX 5.3, striped LVs may be extended to additional PVs, but with the restriction that PVs must be added in multiples of the LV striping width. (For example, if the original LV is striped over 8 PVs, 8 additional PVs must be used to extend the LV.)

To simplify administration, customers may prefer to create VGs with some fixed number of PVs (e.g. 8 or 16). When additional storage is required, a new VG (and LVs) could be created rather than extending the existing VG and LVs. The decision whether to extend existing VGs (and LVs) vs. creating new ones may be influenced by other factors as well. For example, if HACMP is used to provide server-based failover of an Oracle environment, failover times may be affected by the number of VGs (and LVs) that are failed over.

8.3. Create Logical Volumes

Now that the Volume Group(s) have been defined, the Logical Volumes can be created. We will “spread” the data for each Logical Volume across the maximum number of the Physical Volumes within the Volume Group by using the `mklv -e x` option from the command line or through `smitty`. Spreading greatly reduced the risk of I/O hotspots caused by one or more heavily accessed datafiles.

When Raw Devices are going to be used for Oracle Database files, but sure to adhere to existing Oracle limitations on the maximum data file (raw device size). Also note that currently an AIX Logical Volume may not have more than 32,512 Physical Partitions.

Example Cont'd...

For DB01, let's initially create a total of 6 Logical Volumes, 3 for data and 3 for index. The Logical Volumes for data will each be 4 GiB in size (128 32 MiB partitions) and the Logical Volumes for indexes will each be 1 GiB in size (32 32 MiB partitions). Note that Logical Volumes for both index and table (row) data can be contained within the same Volume Group.

```
mklv -e x -y DB01_DATA_01 DB01_DATA_VG1 128
mklv -e x -y DB01_DATA_02 DB01_DATA_VG1 128
mklv -e x -y DB01_DATA_03 DB01_DATA_VG1 128
mklv -e x -y DB01_INDX_01 DB01_DATA_VG1 32
mklv -e x -y DB01_INDX_02 DB01_DATA_VG1 32
mklv -e x -y DB01_INDX_03 DB01_DATA_VG1 32
```

For simplicity, we have not shown the creation of Logical Volumes for such things as Oracle executable libraries, rollback segments, redo logs etc. These would be required for a complete Oracle implementation.

8.3.1. The AIX Logical Volume 4K Offset

When using standard or Big VGs, the first 4K (4096 bytes) of each AIX Logical Volume is normally reserved for the Logical Volume Control Block (LVCB). This means that the first Oracle data block begins at a 4K offset into the Logical Volume. When fine granularity striping is used (either within AIX LVM or within ESS RAID-5 or RAID-10 arrays), this can result in a slight I/O performance degradation when an Oracle DB Block Size greater than 4K is used. (An 8K DB Block Size is typical for OLTP applications and a 16K DB Block size is typical for Data Warehouse applications.) This is because every few DB blocks are physically split across device boundaries – with the first part of the DB block residing on one physical disk and the remainder of the DB block residing on another physical disk. This can result in two physical I/Os being required to read or write a single DB block. The larger the DB Block size used, the higher the percentage of split blocks and the greater the potential for I/O performance degradation.

As of Oracle9i Release 2 (or later), it is possible to create zero offset LVs for Oracle raw devices. This is recommended for new Oracle implementations or for existing applications with extremely high I/O performance requirements. For 9i Release 2, you must be at 9.2.0.3 or later, or have applied Oracle patch (bug **2620053**).

For AIX 5.3 (or later) environments, use Scalable VGs (mkvg -S option). Provided Oracle is at the required release/patch levels (see above), it will automatically begin writing at offset zero in any Raw Device LV contained within a Scalable VG. This is because LVCBs for LVs created within Scalable VGs are not located at the beginning of the LV.

For AIX 5.2 or earlier, zero offset LVs can be created within Big VGs (mklv -B option). For AIX 5.1, e-fix (APAR) **IY36656** is required and for AIX 4.3, **IY38578** is required. Once the prerequisite (Oracle and AIX) software is installed, the following can be done to take advantage of the zero offset feature:

- Create a “big” Volume Group using the mkvg -B flag.
- Create one or more Logical Volumes in that Volume group using the mklv -T O flag. The “-T O” option indicates to Oracle that it can safely use a 0 offset for this Logical Volume.

Note: The mklv “-T O” option is only supported for LVs created in Big VGs.

In order to eliminate the 4K offset for an existing Oracle database, new Logical Volumes must be created and the existing data must be migrated to the new Logical Volumes using normal migration procedures.

8.4. Create Data Files

If you will be using Raw Devices, this step is omitted. However, if you are using the Journaled File System (JFS) to manage your data files, you would create one or more data files (via the Oracle CREATE/ALTER TABLESPACE “ADD DATAFILE” clause) per AIX Logical Volume. Large, heavily accessed data files are sometimes subject to inode contention. We generally recommend that individual RDBMS related data files be no more than 2 GiB in size in order to keep inode contention at a minimum.

Oracle reads the header block of all database data files at database startup to determine whether or not database recovery is required. Therefore in general, the larger the number of data files, the longer the startup time. And, since all open data files need to be closed during a normal database shutdown, shutdown is also somewhat proportional to the number of data files. For High Availability (HA) environments that employ single instance HACMP failover (as opposed to doing an Oracle9i Real Application Clusters (RAC) coordinated failover), reducing the number of data files may reduce time required for failover.

Oracle provides I/O performance information at a data file (or raw device) level. Therefore, having a large number of data files may provide more accurate information regarding the I/O characteristics of particular database objects (e.g. tables, indexes). For some applications, this may be an argument for having more rather than fewer data files.

Example Cont'd...

In our example, we will be using raw devices, so individual data files will not be created.

9. Variations on the “Striping and Spreading” Theme

9.1. The SAME (Stripe and Mirror Everything) Methodology

A few years ago, Oracle (in conjunction with EMC) began recommending the use of the SAME (Stripe and Mirror Everything) Methodology for layout of Oracle database files. SAME has two fundamental goals:

- Provide protection against single disk failures. Mirroring is one way to accomplish that.
- Reduce I/O hotspots by balancing I/O activity across multiple physical disks.

The “striping and spreading” technique outlined in this paper is entirely consistent with SAME:

- ESS RAID-10 and RAID-5 arrays both provide protection against single disk failures.
- ESS RAID-10 and RAID-5 both balance I/O activity across all the physical disks in a RAID-10 or RAID-5 array.

However, ESS and the “striping and spreading” technique offers a number of additional benefits:

- The choice of RAID-10 (mirroring) or RAID-5. RAID-5 provides comparable performance to RAID-10 for most customer workloads with better price/performance.
- Good, automatic I/O balancing across all arrays within the ESS subsystem as well as within a single ESS RAID-5 or RAID-10 array.

9.2. Oracle10g Automatic Storage Management (ASM)

Automatic Storage Management (ASM) is a new Oracle 10g component which provides integrated file system and volume manager features for Oracle database files. ASM allows Database Administrators to define storage pools (disk groups). Oracle then manages the allocation and placement of Oracle files within those pools.

ASM stripes Oracle objects across all the available disks in a disk group using either a 1MiB (course) or 128KB (fine) stripe size. Normally, you would want to use the 1MiB striping option. 128KB striping is potentially applicable to such things as redo log files where it is desirable to split large I/O requests into smaller parallel I/Os to multiple disks (hdisks). However, as already discussed, small granularity striping at the host level can potentially disrupt the sequential read/write algorithms within the ESS. At file creation time, the DBA can decide whether or not to use this fine-grained striping option for particular files. Otherwise, file type specific templates control default behavior within the disk group.

Since ASM internally “spreads” file extents across all of the hdisks in the ASM disk group, from a data and I/O distribution standpoint, it functions similarly to PP spreading and/or LVM (large granularity) striping in non-ASM environments. However, ASM does not use a pure “round

robin” algorithm for striping across the available disks. Rather, it stripes based on current unused capacity of each available disk – disks having larger capacity (more unused space) will end up having more extents placed on it than disks having smaller capacity (less unused space). For this reason, it is extremely important that all of the hdisks in any given ASM disk group have the same size and performance characteristics (e.g. same RAID configuration, same underlying physical disk capacity same disk RPM, etc.).

The implementation steps for ASM and non-ASM environments are essentially identical through section 8.1 above. After that, one or more ASM disk groups should be defined using similar criteria to what has been discussed for setting up AIX Volume Groups in section 8.2.

ASM also has the ability to do 2-way and 3-way file mirroring:

- If “external” redundancy is specified, this indicates that Oracle will only maintain a single copy of the file. With RAID-5 or RAID-10 based I/O subsystems, “external” redundancy is normally sufficient, since the RAID-5 and RAID-10 architectures protect against single disk failures.
- If “normal” or “high” redundancy is specified, Oracle will maintain multiple copies (2 or 3 respectively) of the file, with each copy being placed in a different “failure group”. Failure groups are used to ensure that each copy is placed on a mutually exclusive set of AIX hdisks. “Normal” or “high” redundancy can be useful when non-RAID protected I/O subsystems are used, or where a very high degree of data protection (e.g. against double disk, or total I/O subsystem failures) is required.

If ASM mirroring is used, make sure that the physical ESS RAID arrays used to back hdisks (LUNs) in any given failure group are mutually exclusive to those RAID arrays used to back hdisks (LUNs) in the other failure group(s) involved in ASM mirroring. And, if the intent is to protect against total I/O subsystem (ESS) failures, make sure that the RAID arrays associated with different failure groups are on different ESS subsystems.

Since ASM mirroring is server software based, it can potentially involve higher overhead (CPU, redundant I/O adapter and SAN traffic, etc.) than hardware based alternatives. Prior to implementing ASM mirroring, take a moment to consider I/O subsystem options that might be available. For example:

- RAID-5 and RAID-10 arrays provide data redundancy protection against single disk failures.
- PPRC can be used to mirror LUNs within a single ESS subsystem or between multiple ESS subsystems.

9.3. IBM General Parallel File System (GPFS)

When implementing Oracle Real Application Clusters (RAC) environments, many customers prefer using a clustered file system. GPFS is IBM’s clustered file system offering for Oracle RAC on AIX. Other Oracle files, such as the ORACLE_HOME executable libraries, archive log directories do not have to be shared between instances. These can either be placed on local disk (e.g. using JFS2 file systems), or a single copy can be shared across the RAC cluster using

GPFS. Potentially, this provides some administrative advantages, such as having only a single physical ORACLE_HOME to maintain, ensuring that archive logs are always available (even when one or more nodes is down) when recoveries are required, etc.

When used with Oracle, GPFS automatically stripes files across all of the available disks within a GPFS file system using a 1MiB stripe size. Therefore, GPFS provides data and I/O distribution characteristics similar to PP spreading, LVM (large granularity) striping and/or ASM coarse grained striping techniques.

The implementation steps up through section 8.1 above may be followed for GPFS. At that point, one or more GPFS file systems (with one or more hdisks assigned per file system) may be created.

9.4. “Striping and Spreading” with other I/O Subsystems

The Striping and Spreading approach can be adapted to most other available I/O subsystems as well. However, it is important to have an understanding of the characteristics of the physical disks and array configurations on the back end, as well as an understanding of how LUNs are mapped to those physical backend structures.

9.4.1. IBM DS6000 and DS8000 Series

The DS6000 and DS8000 series storage subsystems are follow-on products to the ESS. The same techniques discussed in this paper may be applied to DS6000 and DS8000 series subsystems.

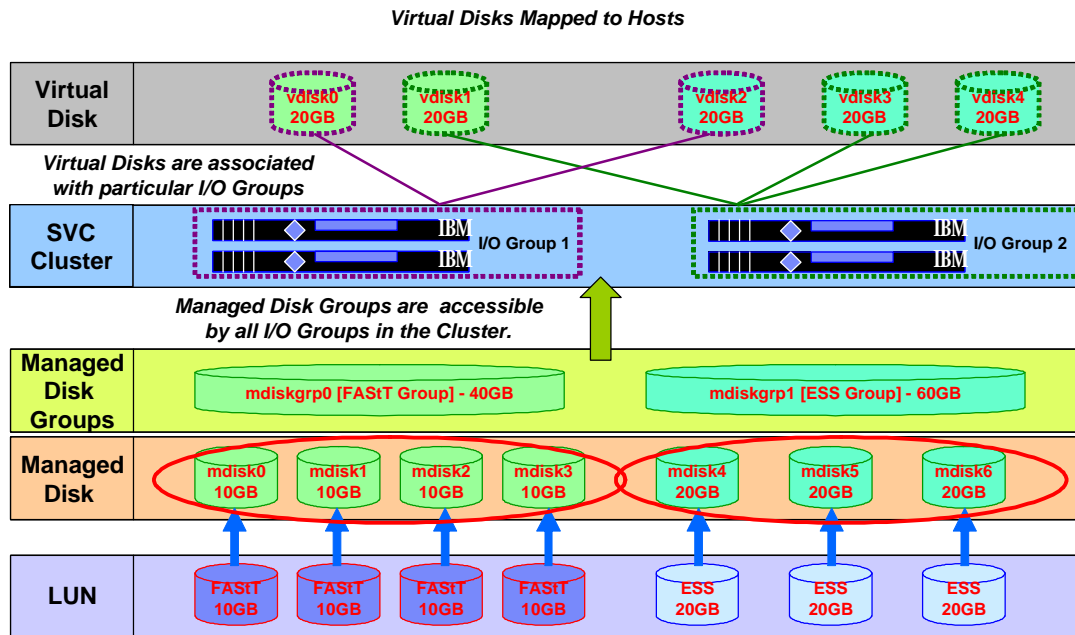
9.4.2. IBM DS4000 Series (FAStT)

The DS6000 series (formerly called the FAStT family) is a set of mid-range storage systems. They support RAID-0, 1, 3, 5 and 10 array types. RAID-5 or RAID-10 would normally be used when implementing the Spreading and Striping strategy discussed in this document.

9.4.3. IBM SAN Volume Controller

The IBM TotalStorage SAN Volume Controller is a member of the IBM TotalStorage virtualization solutions family and is designed to reduce the complexity and costs of managing SAN-based storage. The SAN Volume Controller is an in-band virtualization solution which manages a number of back-end storage controllers and maps the physical storage within those controllers to logical disk images that are presented to servers connected to the SAN.

The following diagram shows the various layers within the SAN Volume Controller mapping structure.



There are a number of different Managed Disk (mDisk) to Virtual Disk (vDisk) mapping functions available with the SAN Volume Controller:

- **Striped:** Each vDisk is mapped to a number of mDisks in a Managed Disk Group (MDG). The extents on the vDisk are striped over the mDisks. Striping is done at the extent level and is therefore coarse grained striping, similar to PP spreading within AIX. This is the default mapping policy.
- **Sequential:** A vDisk is mapped to a single mDisk in a MDG. There is no guarantee that sequential extents on the mDisk map to sequential extents on the vDisk.
- **Image:** Provides a one-to-one mapping of extents on an mDisk to the extents on the vDisk.

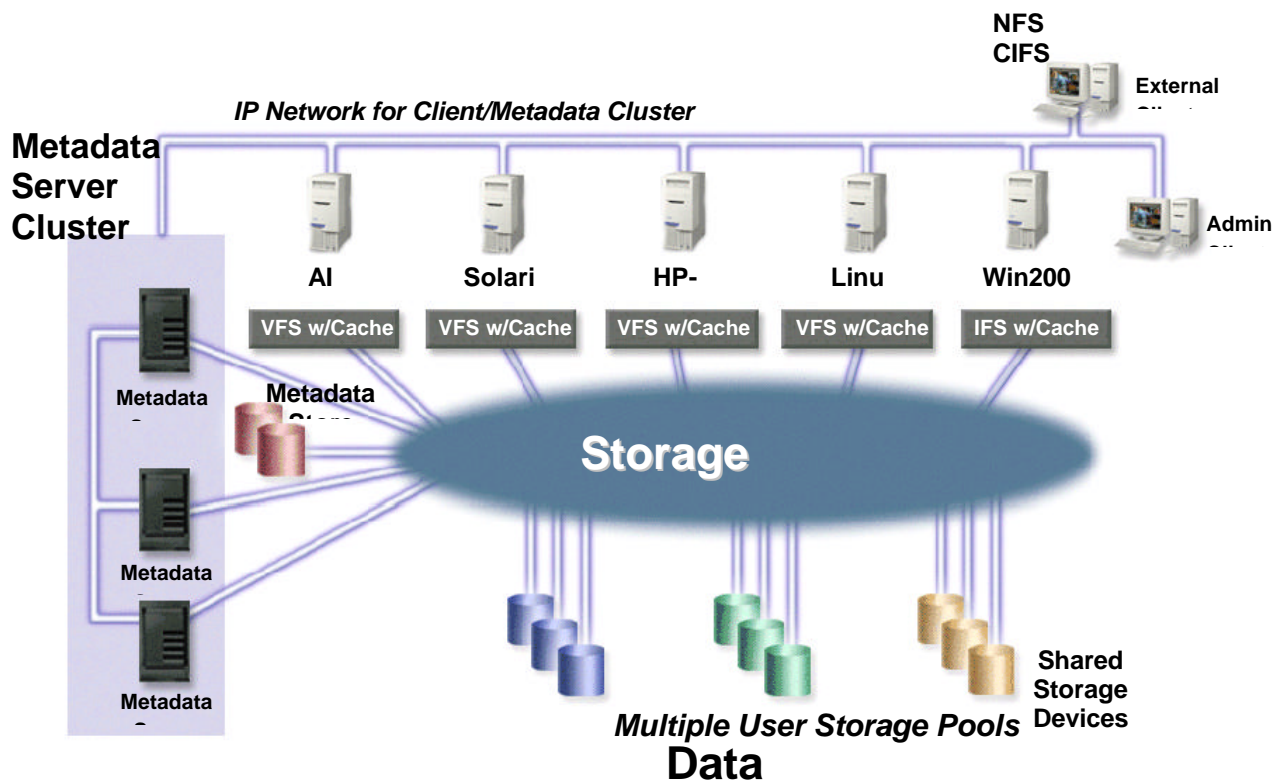
For striped or sequential vDisks, there is no ordering requirement in the vDisk to mDisk extent mapping. For example, it is possible that adjacent extents on an mDisk could be mapped to different vDisks, adjacent extents on the same mDisk could be mapped to widely separated extents on the same vDisk, or to adjacent extents on the same vDisk.

Ideally, LUNs on the back-end storage subsystem should be striped (RAID-5 or RAID-10). If the ‘Sequential’ or ‘Image’ mapping policy is used, AIX PP spreading or course granularity LVM striping should be used across multiple vDisks that are backed by different mDisks. The ‘Striped’ mapping policy can potentially be used instead of AIX level PP spreading or LVM striping. However, the resulting extent mapping can be less predictable given the lack of an ordering requirement for ‘Striped’ vDisks.

9.4.4. IBM SAN File System

The IBM SAN File System is another member of the IBM TotalStorage virtualization solutions family. It is an out-of-band storage virtualization implementation, where the data and metadata (data about the data) are stored in different places. This involves the use of a Metadata server which stores metadata information and provides request authorization, locking and control services for data access requests.

The SAN File System Metadata server (MDS) is a SAN attached server that communicates with application servers to server the metadata. Once a SAN File System client is installed on the application servers, no changes are required to any applications using SAN File System data, since it emulates the syntax and behavior of local file systems.



SAN File System volumes (LUNs) are grouped into storage pools. The System Pool is used for file metadata and other system metadata which is shared across the SAN File System cluster. One or more User Pools are created to store user data. Policy Rules are used to determine which User Pool an individual file is placed in.

As with the other I/O subsystem alternatives discussed in this paper, it is preferable to place multiple striped LUNs into a SAN File System User Pool.

10. Conclusions

I/O subsystem performance can have a significant effect on overall application performance. Manual I/O “hotspot” management can be a very resource intensive activity.

The “striping and spreading” technique described in this paper provides for balanced I/O activity across the ESS subsystem without requiring detailed knowledge of the application data and I/O characteristics and without significant ongoing “hotspot” management. This technique may also be easily adapted to a number of other file system and/or I/O subsystem options.

This “striping and spreading” strategy has been used in a number of large customer Enterprise Application Solutions (OLTP) application implementations with excellent results.

11. References

[1] IBM TotalStorage Enterprise Storage Server Introduction and Planning Guide, IBM Publication Number GC26-7444-00

[2] IBM TotalStorage Enterprise Storage Server Model 800 Performance whitepaper by Bruce McNutt, July 2002

[3] AIX 5L Version 5.1 Commands Reference, IBM Publication SBOF-1877

[4] AIX 5L Version 5.2 Commands Reference (Volumes 1 – 6), IBM Publication Numbers SC23-4115, SC23-4116, SC23-4117, SC23-4118, SC23-4119, SC23-4120

[5] AIX 5L Version 5.3 Commands Reference (Volumes 1 – 6), IBM Publication Numbers SC23-4888, SC23-4889, SC23-4890, SC23-4891, SC23-4892, SC23-4893

[6] Oracle8i Administrator’s Reference – Release 3 (8.1.7) for AIX-Based Systems – Oracle Part Number A85348-01

[7] Oracle9i Administrator’s Reference – Release 2 (9.2.0.1.0) for UNIX Systems: AIX-Based Systems, Compaq Tru64 UNIX, HP 9000 Series HP-UX, Linux Intel and Sun Solaris – Oracle Part Number A97297-01

[8] Oracle 10g Administrator’s Reference – Release 1 (10.1) for UNIX Systems: AIX-Based Systems, hp HP-UX PA-RISC (64-bit), hp Tru64 UNIX, Linux x86 and Solaris Operating System (SPARC) – Oracle Part Number B10812-01

[9] The IBM TotalStorage DS8000 Series: Concepts and Architecture Redbook, IBM Publication Number SG24-6452-00

[10] The IBM TotalStorage DS6000 Series: Concepts and Architecture Redbook, IBM Publication Number SG24-6471-00

[11] IBM TotalStorage Solutions Handbook Redbook, IBM Publication Number SG24-5250-05

[12] IBM TotalStorage SAN Volume Controller and SAN Integration Server Redbook, IBM Publication Number SG24-6423-01

[13] IBM Total Storage: Introducing the SAN File System Redbook, IBM Publication Number SG24-7057-02

12. Acknowledgements

Several people provided technical input and guidance for this paper, including:

- John Aschoff – Database and Performance, IBM Storage Systems Group
- Dan Braden – pSeries Support, IBM Advanced Technical Support - Americas
- Martin Carangelo – Enterprise Application Solutions, IBM Advanced Technical Support – Americas
- Ralf Schmidt-Dannert – Enterprise Application Solutions, IBM Advanced Technical Support - Americas
- Jaymin Yon – Enterprise Application Solutions, IBM Advanced Technical Support - Americas