

Hitachi Dynamic Provisioning Software Best Practices Guide

Guidelines for the Use of Hitachi Dynamic Provisioning Software with the
Microsoft® Windows Operating System and Microsoft SQL Server 2005,
Microsoft Exchange 2003 and Microsoft Exchange 2007

White Paper

By Steve Burr, Hitachi Data Systems Global Solution Services

January 2008



Executive Summary

Hitachi Dynamic Provisioning software has the potential to provide significant benefits to storage and system administrators. These may include simplified management, improved performance and better control of costs. However, to achieve the maximum benefits from Hitachi Dynamic Provisioning, there are best practices to follow when configuring and managing the application environment. There are also some management actions that are best to avoid or closely control as they could negate the benefits. This document collects some best practices for using Hitachi Dynamic Provisioning software in a Microsoft® Windows environment with some of the more common Windows Server applications, such as Microsoft SQL Server and Microsoft Exchange.

Although the recommendations documented here may generally represent good practices, configurations may vary. Please contact your Hitachi Data Systems representative, or visit Hitachi Data Systems online at <http://www.hds.com> for further information regarding solutions by Hitachi Data Systems.



Document Revision Level

Revision	Date	Description
1.0	January 2008	Initial Release

Authors

The information included in this document represents the expertise of a number of skilled practitioners. Contributors to this document include:

- Scott Brown, Steven Burns, Rick Freedman, Richard Jew, Larry Korbus, Jean Francois Massé and Larry Meese



Contents

Introduction	1
Understanding Hitachi Dynamic Provisioning Software	1
Dynamic Provisioning Overview	3
When and How Pool Space Is Consumed	5
Disk Initialization	5
Volume Management — Windows Dynamic Disks	5
Volume or Partition Creation	6
File System Creation	6
Application Objects and Their Usage	6
SQL Server 2005 Executive Summary	8
SQL Server 2005 Detail	8
SQL Server 2005 Testing Detail	11
Exchange 2003 and 2007 Executive Summary	13
Exchange 2003 and 2007 Detail	13
Replication and Hitachi Protection Manager Software	16
Management Tasks	16
Technology Discussion of the Use of Hitachi Dynamic Provisioning Software	17
Metadata	17
File System Storage Reclaim	18
Dynamic Partition Expansion with Hitachi Dynamic Provisioning	20
Dynamic Partition Expansion in Windows	21
Exchange 2007 Extended Example	21
Monitoring	22
RAID Manager Monitoring	22
Performance Factors	23
Recommendations for Adding Disks to a Pool	24
Capacity Planning Factors	25
Appendix A — Hitachi Tuning Manager with Hitachi Dynamic Provisioning Software	26
Appendix B — Testing Methodology Used	30
NTFS File System Tests	30
Exchange 2003 Tests	30
SQL Server 2005 Tests	31
Exchange 2007 Tests	31
Appendix C — Hitachi Dynamic Provisioning Software Terms	32



Hitachi Dynamic Provisioning Software Best Practices Guide

Guidelines for the use of Hitachi Dynamic Provisioning Software with the Microsoft® Windows Operating System and Microsoft SQL Server 2005, Microsoft Exchange 2003 and Microsoft Exchange 2007

White Paper

By Steve Burr, Hitachi Data Systems Global Solution Services

Introduction

This best practices white paper discusses the use of Hitachi Dynamic Provisioning software, one of the latest innovations in Hitachi storage technology. Provisioning of storage can be a challenging task. Traditional static methods of storage provisioning require the administrator to make decisions at an early stage such as: capacity purchase, configuration size and allocation. These decisions are not always correct, possibly leading to performance issues and waste, as well as making storage administration tasks overly complex. Hitachi Dynamic Provisioning allows the administrator to defer allocation decisions to a later time because storage is only allocated when it is actually used.

Like any new technology, the best use of Hitachi Dynamic Provisioning software implies a few changes in management practices. This document discusses these best practices in the context of applications running on the Windows operating system platform. These best practices have been derived as a result of tests and analysis performed in the Hitachi Data Systems laboratories with Microsoft SQL Server 2005, Microsoft Exchange 2003 and Microsoft Exchange 2007 applications.

Important observation: Using Hitachi Dynamic Provisioning does not mean you have to stop using static provisioning methods. Both are available simultaneously, and most sites will likely use both technologies.

Understanding Hitachi Dynamic Provisioning Software

Over time, storage solutions have seen a number of significant developments. Key ones include:

- Direct attached storage
- RAID systems
- Storage replication
- Storage virtualization
- Tiered storage
- Dynamic provisioning



As users have adopted and deployed these technologies, they have seen the following benefits:

- Increased storage reliability
- Increased maintainability — online maintenance and upgrade — “once on, never off”
- Increased performance and, more importantly, increased control over that performance
- Improved manageability
- Improved cost savings and a greater control over costs
- Improved ability to choose the correct performance and price point for every application and to have the flexibility to change it as the environment changes
- Reduced waste, including further leveraging of legacy assets

These benefits are facilitated by the high performance available within the storage controller, such as with the Hitachi Universal Storage Platform™ V and Universal Storage Platform VM, which are the latest generations of Hitachi storage and the first to support dynamic provisioning.

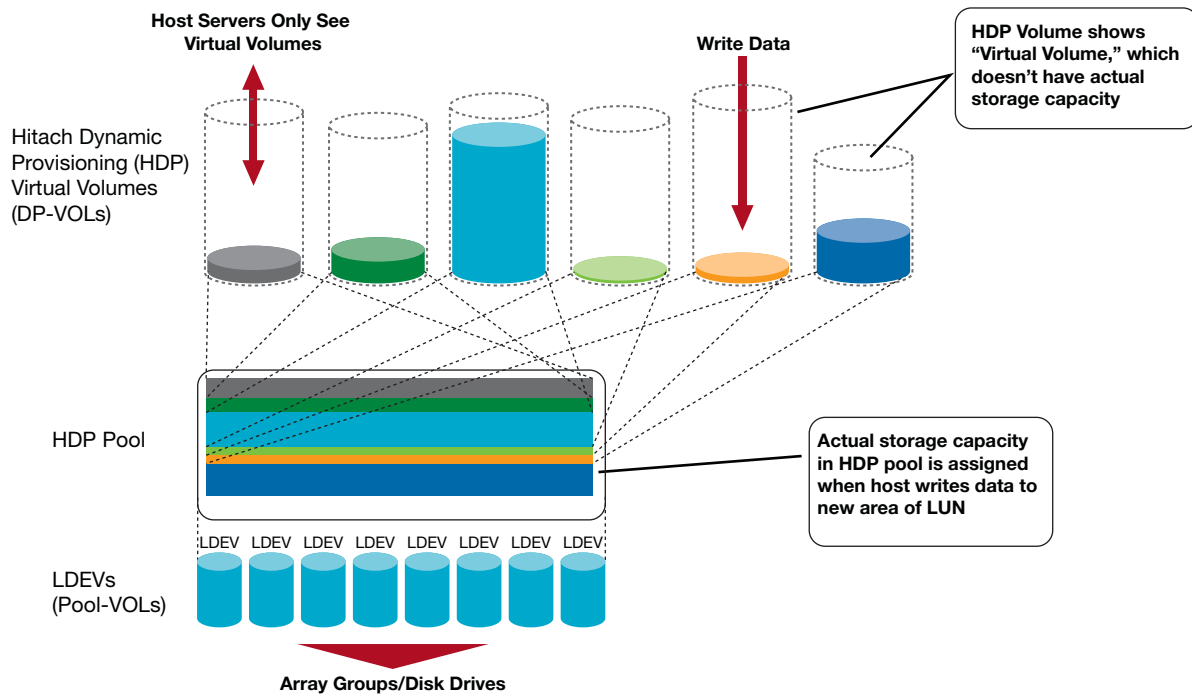
Dynamic provisioning is an alternative to static provisioning. Rather than make all storage allocation decisions at an early stage, they can be delayed until capacity is actually required, while the application grows over time. This gives vastly greater flexibility in a number of areas. Newly purchased storage is not directly allocated to specific servers but instead is allocated to one of the configured storage pools. As application space is used, it is allocated directly from the pool only when needed.

Hitachi Dynamic Provisioning separates application storage allocation from physical allocation, enabling storage administrators to quickly deploy dynamic provisioned volumes for faster application deployment, without the risk of leaving storage capacity stranded. Adding new physical capacity to the storage system is made easier as it does not affect application availability, thus simplifying storage management. Once allocated, the Universal Storage Platform V or Universal Storage Platform VM will automatically assign physical disk capacity as needed. Dynamic Provisioning enables the movement to a just-in-time model of storage provisioning that allows the incremental addition of capacity to common storage pools. This eliminates the waste from the storage infrastructure and reduces the total cost of storage ownership.

With Dynamic Provisioning, administrators can manage the storage allocations of each application less frequently. Virtual volumes and the available physical disk space are constantly monitored to notify administrators in advance when additional capacity is required, helping to ensure applications will not run short of storage.

Using wide striping techniques, Dynamic Provisioning automatically spreads the I/O load of all applications accessing the common pool across the available spindles (see Figure 1). This process should reduce the chance of hot spots and optimizes I/O response times, which can lead to consistently high application performance.

Figure 1. Dynamic Provisioning – Virtual Volumes



Dynamic Provisioning automatically spreads the I/O load of all applications accessing the common pool across the available spindles.

Dynamic Provisioning Overview

Hitachi Dynamic Provisioning (HDP) uses two primary components: a pool of physical storage capacity (HDP Pool) and virtual volumes called DP-VOLs.

The HDP Pool is composed of normal LDEVs known as pool volumes (Pool-VOL). For performance reasons there will normally be multiple pool volumes from multiple Array Groups. The total physical capacity of a pool is the sum of the capacities of the pool volumes in that particular pool.

After a DP-VOL is associated with a HDP Pool, a path definition is created to assign the DP-VOL to an application host server. As the application writes data to the DP-VOL, the Universal Storage Platform V or Universal Storage Platform VM will allocate physical capacity from the HDP Pool while monitoring multiple thresholds to alert administrators when more physical pool capacity is needed.

A DP-VOL is a dynamic provisioning virtual volume. The term V-VOL is sometimes used for virtual volume; this can be misleading because other sorts of virtual volumes exist, such as the V-VOL used in Hitachi Copy-on-Write Snapshot software. All DP-VOLs are V-VOLs but not all V-VOLs are DP-VOLs.

When you wish to create a DP-VOL, you use the dynamic provisioning section of the V-VOL tab of Storage Navigator. Similarly, you create a dynamic provisioning pool or copy-on-write pool from the pool tab.

Table 1 walks through the key tasks performed by a storage administrator using static or dynamic provisioning.

Table 1. Key Tasks of Static and Dynamic Provisioning

Static Provisioning	Dynamic Provisioning
Group a number of physical spindles together and select a RAID level appropriate to the application.	Group a number of physical spindles together and select a RAID level appropriate to the applications. These “RAID groups” are then assembled into one of the dynamic provisioning pools.
Choose allocation unit sizes appropriate for the application. Typically this would be a number of gigabytes based on the site’s selected “standard” volume sizes. <i>Inefficiencies are inherent in using “standard” size volumes. These volumes tend to be oversized relative to the user’s requirements and therefore fundamentally are underutilized. This step can be a challenging one and is a key aspect of application capacity planning.</i> <i>Few applications will continue to operate if their storage capacity runs out, so the only safe way the system administrator can operate is to make the best plan possible and then include an overhead (buffer or fudge-factor).</i> <i>Overhead figures of 20 percent or more are common recommendations in this planning stage. Analysts have reported measured underutilization numbers in the 50 percent or more range.</i>	With Hitachi Dynamic Provisioning software, there is one physical allocation unit size that doesn’t need to be tuned for the application. The unit of allocation is a 42MB “page”. Pages are not allocated directly to any volume; they are allocated on use. This is totally transparent to the user. Any volume size can be easily created without detracting from manageability and therefore can improve utilization.
These “standard” size units are mapped to servers and used in applications. <i>At this point all disk space equal to the volume size has been consumed.</i>	The virtual volumes are mapped to servers and used in applications. <i>At this point no disk space has been consumed.</i>
The storage is formatted and accessed in the usual way.	The storage is formatted and accessed in the usual way, but space is only allocated from the pool when the server writes to the disk.
As storage needs grow, administrators repeat parts of this process to add more storage for the application. <i>Unfortunately, “another LUN” is rarely what applications want. All too often there is the thought “if only I had chosen a larger LUN in the first place.”</i> Some methods are available for easing this, specifically, online LUSE expansion.	As storage needs grow, administrators would not need to allocate additional storage; it would be allocated on use from the pool. As the pool is consumed, additional storage capacity can be procured and added nondisruptively.
<i>The administrator at this point can be presented with a challenging landscape — they have the amount of storage they require, but the data is often in the wrong location. A long, involved process of data relocation may be required to get everything correct.</i>	<i>Data movement is rarely needed.</i> <i>Data relocation may still be required, but it is more likely to be based on organizational requirements.</i>
<i>Even where storage capacity is adequate, performance balance is rarely correct.</i> <i>Even if initially designed correctly, environment changes soon prevent it remaining so.</i> <i>To maintain the required level of performance at a reasonable cost implies further data relocation.</i> Some methods are available for easing this; specifically, Hitachi Tiered Storage Manager software will permit online control over volume placement.	<i>One of the benefits of Hitachi Dynamic Provisioning is performance leveling. Performance hot spots are naturally smoothed out.</i>



Dynamic Provisioning sounds ideal: you only use and always have available exactly the storage you need. But a new storage metaphor inevitably implies some changes in storage management. This is because the designs of operating systems, file systems and applications were not originally optimized for dynamic provisioning.

Developers of operating systems, file systems and applications have evolved their designs over many of years. However, they could not have designed these with the capabilities of Hitachi Dynamic Provisioning software in mind. In fact, if you read many white papers on tuning or data layout, you will notice there is little attention to anything more sophisticated than direct attached storage.¹

The issue we have to be aware of is that many of the designs use storage without considering the cost of doing so.

When and How Pool Space Is Consumed

Hitachi Dynamic Provisioning pool space may be “consumed”² at a number of stages:

1. When the DP-VOL is initialized by the operating system
2. If DP-VOLs are put under volume management
3. When operating system volumes are created (either by partitioning or by creating disk group volumes)
4. When file systems are created (in Windows — NTFS formatting)
5. When application objects are created (for example: creation of edb, mdf, ldf or log files)
6. As data is written to the DP-VOL by the application
7. When management activities are performed

We will examine each of these.

Disk Initialization

Initialization of disks occurs the first time the DP-VOL is written and so it requires allocation of space. On Windows, initialization will write metadata³ to the beginning of the DP-VOL only. One page of pool space will be allocated. This is not always the case on non-Windows platforms; see the Metadata section later in this white paper.

Volume⁴ Management — Windows Dynamic Disks

Converting a disk to a Windows dynamic type and grouping these disks to form a disk group will not allocate additional pool space. However, creating certain types of volumes from the disk group may use pool space. Creating volumes is covered in the next section. Because of their limitations, the use of dynamic disks on Windows is relatively rare. They are not supported on clusters or in many application configurations. They have limitations when used with disk replication products such as Hitachi ShadowImage® Replication software. Many reasons for using dynamic disks are minimized by virtualized storage. For example, implementation of mirrors or

¹ For the optimal solution, further evolution in operating system design is required, cognizant of dynamic provisioning.

² To be precise, consumed means when Dynamic Provisioning is required to allocate a page of storage from the pool. Page allocation is always a consequence of someone writing to an area of a volume. This allocation is permanent until the volume is deleted.

³ See the Technological section later in this white paper for further discussion of metadata.

⁴ The terms dynamic, volume and disk can get confusing here as they are both used by Hitachi Data Systems and Microsoft. The Windows term “disk” is what Hitachi Data Systems would call a LUN. This would be an LDEV, DP-VOL or V-VOL. The Windows term volume is a single section of disk space normally used to hold a file system. A Windows basic disk can be partitioned into multiple volumes. Multiple Windows dynamic disks can be used to form multiple Windows volumes.



RAID sets is far better done on the storage system itself. Spanned and striped volumes are sometimes recommended for use with Hitachi storage as they provide methods for coping with performance and capacity planning issues. Dynamic provisioning further reduces the requirement for dynamic disks in Windows; however, they will work equally well with Hitachi Dynamic Provisioning.

Volume or Partition Creation

Metadata written while the first primary partition on a Basic Disk is created is likely to be in the same page used for initialization. Additional partitions will consume an additional page each. For best performance, it is recommended to align your disk partitions to a storage system's appropriate value. The default alignment is 63 blocks, which is far from optimum for any storage. For Hitachi storage, an alignment of 64 blocks has been shown to perform better. This can be accomplished with the diskpar or, on Windows 2003 SP1 and above, the diskpart tool. Alignment of partitions does not consume additional pool space (except the tiny fraction of space wasted by padding the partition). There are many articles on this: for example, "How to Align Exchange I/O with Storage Track Boundaries" at <http://technet.microsoft.com/en-us/library/aa998219.aspx>.

You do not have to allocate all of the available disk space to a partition. You will see later that reserving space for future growth can provide additional management flexibility and control.

The pool space consumed by creating a volume will depend on the volume type:

- Basic volumes and simple dynamic volumes allocate no additional pool space.
- Mirrored dynamic volumes will copy all of the data from the volume of one disk to the other disks, even though they contain no usable data at this point. They are not recommended with either dynamic or static provisioning.
- RAID-5 dynamic volumes allocate all the configured pool data even though no usable data is there. They are not recommended with either dynamic or static provisioning.
- Striped and spanned volumes allocate no additional pool space; however, Dynamic Provisioning is likely to be a better way to deliver the performance and flexibility features they provide.

File System Creation

Creation of an NTFS File system will only write to a limited proportion of the disk (corresponding to metadata such as the Master File Table). The amount of space used depends on the file system size. For small volumes (c. 1GB), this is accommodated in the initial page. For larger volumes, additional pages are allocated, but generally this is less than one percent of the file system size. To guarantee that the minimum space is written, always use quick formatting. Slow formatting is equally efficient on Windows 2003 but tests on Windows 2008 Beta showed that slow format wrote more data. There is no benefit in slow formatting with RAID systems because all devices are pre-formatted.

Application Objects and Their Usage

The goal of this section is to start to look at the question, "How do I make sure that my application keeps to a controlled amount of space? Specifically, how do I ensure that they don't allocate more space than I intended them to?" Controlling space is important as it directly controls the cost, which is one of the benefits provided by Dynamic Provisioning software. We will start by looking at file systems and general principals of applications and then look at some key applications in more detail.

File Systems – NTFS

After metadata allocation, storage space reclaim is the next most important factor in maintaining a space-efficient file system. This is described in the Technology section later in this paper. The Windows NTFS file system has poor storage reclaim characteristics. If Dynamic Provisioning is used with virtual volumes that are greatly oversized then there is a risk that space will be allocated unnecessarily. Controlling the way the file system is used can

prevent this from occurring but if you have no control (such as a general file system share for user storage) then the only safe approach is by using dynamic partition expansion. This technique is discussed in the technology section below. Fortunately, many applications have predictable access patterns and are a good match with Dynamic Provisioning.

General Principals of Application Data Using NTFS

Many applications have classes of data with similar characteristics. In many applications, you can often identify data classes corresponding to the following:

- Database files. (Examples: Exchange edb file, SQL .mdf file, Sybase database file, Oracle tablespace files)
- Log files. (Examples: Exchange .log files, SQL .ldf file)
- Management and backup files. (Examples: NTBackup .bkf files, SQL Server .trn or .bak files, Archive Redo Logs, the Exchange System directory)

Example of good behavior:

Create File X, Y, Z

While

Append to File X, Y Z

End While

Examples of behavior that will waste pool:

While

Create File X_n, X_{n+1}, X_{n+2}

Delete File X_m, X_{m+1}

End While

While

Create File X⁵

Delete File X

End While

Note that sometimes one file is involved, sometimes many files. It is important to know if these data classes will or will not overallocate when used with Dynamic Provisioning software. It is possible to predict how a data class will perform, but you should try to obtain a recommendation or perform an evaluation of an untested data class. If uncertain, you can always use dynamic partition expansion as an additional safety “cap.” This is to be recommended if you consider an installation might be subject to a rebuild (for example, some test environments). Dynamic partition expansion is described later in this paper.

One simple method for prediction is: if the quantity and names of files remain constant as the application runs and data files just grow as data is added to the application, then this is likely to be space efficient.

If multiple files are continuously being created, then you can predict that the pool space allocated is highly likely to grow unacceptably — it does not matter that the files are subsequently deleted. With NTFS additional space will be allocated.

⁵ This is confusing at first sight. You might think, “it is the same name, it is the same file, and therefore it will re-use the space.” This is regrettably wrong. It may have the same path and file name but from Windows point of view, it is a new file that just happens to have the same name — so it creates new space. UNIX has the same behaviour but its operation is clearer. On UNIX, a file may have multiple names or no name at all. The actual data is addressed not by name but by number, the inode number. If you delete a file and re-create it, then the file will have a different inode number.

Renaming a file does not change the pool space allocated, as the data is not moved.

In addition to file creation/deletion, many applications present configuration parameters to control their storage allocation strategy. In database terminology this is known as the physical schema. Many authors recommend using the application equivalent of automatic file expansion. This method does indeed work well with dynamic provisioning and is our recommendation. However, manual expansion methods work too. The key is to keep the initial data allocation to a moderate amount.

Other general guidelines are: Do not mix database files with other data in the same file system. Do not use defragmentation tools; they waste space and may reduce performance.

SQL Server 2005 Executive Summary

SQL Server can maintain space efficiency with Dynamic Provisioning under the conditions shown in Table 2:

Table 2. Maintaining SQL Server

Database Files	Files can be on either capped or unconstrained Hitachi Dynamic Provisioning (HDP) software volumes. SQL Server initial sizes should be kept to a minimum and planned with appropriate growth and size limits. Follow normal guidelines for Windows NTFS file systems on HDP volumes.
Log Files	Files can be on either capped or unconstrained HDP volumes. SQL Server initial sizes should be kept to a minimum and planned with appropriate growth and size limits. Follow normal guidelines for Windows NTFS file systems on HDP volumes.
Backup Files	Files should be on static volumes or HDP volumes capped by dynamic partition expansion.

SQL Server 2005 Detail

SQL Server stores the majority of its data in a database file (normally with an .mdf extension). Often this file will exist in a file system on its own. If you wish to grow a database larger than one disk, you are allowed to span the data onto subsequent disks by creating additional database extension files (.ndf). SQL Server 2005 also has a database log file (.ldf extension), which is often stored on its own disk because its access pattern (sequential write) is very different to the database (random). SQL Server has its own streaming backup mechanism, which may be used to generate .bak and .trn files for the database and log backups. SQL Server has configuration parameters — SIZE, FILEGROWTH and MAXSIZE — that set the allocation strategy for its database and log files. The parameters used are illustrated in the following SQL code to create a database:

```
CREATE DATABASE mydb ON PRIMARY (
    NAME = mydb1,
    FILENAME = 'S:\mountpoint\data1\mydb1.mdf',
    SIZE = 42MB,
    MAXSIZE = 42000,
    FILEGROWTH = 42)
LOG ON (
    NAME = mydblog1,
    FILENAME = 'S:\mountpoint\log1\mydblog1.ldf',
    SIZE = 42MB,
    MAXSIZE = 10000,
    FILEGROWTH = 42)
```



Each file is created with an initial SIZE and will autoextend by FILEGROWTH when the currently allocated space runs out, but it will stop if it reaches MAXSIZE. This is a perfect structure for use with Dynamic Provisioning software. These parameters will have equivalent values in the capacity plan.

For the database, the values of SIZE, FILEGROWTH and MAXSIZE will be determined by the capacity plan. A trade-off exists between space consumed and performance. Very small values of FILEGROWTH (<21MB) are not recommended as they lead to fragmentation. It makes sense for FILEGROWTH to be a multiple of a page, 42MB, as this will improve on-disk contiguity. If SIZE and FILEGROWTH are large, then pages are allocated more rapidly, but they will have high contiguity. However, contiguity is generally less important with dynamic provisioning than static provisioning. MAXSIZE is more straightforward, and it is used to make sure that your application does not “go out of control.” This is an additional control over your space. You will wish to set SIZE to a value to create an initial allocation of data that is at least equal to the amount of data you plan to import into the database.

The recommendations for managing SQL Server 2005 with Hitachi Dynamic Provisioning are outlined below. This advice is probably equally valid for previous versions of SQL Server, too.

1. Start any project with migration planning (where appropriate) and capacity planning, including a plan to monitor capacity. None of this is fundamentally different from traditional SQL Server planning. This would consider:
 - a. Method and frequency of reviews (say once per week for “are we on plan”)
 - b. Estimate of the time needed to procure more storage (worst case) to ensure that capacity can be available at the required time⁶
 - c. For the database:
 - i. Starting database size to be imported (SIZE)
 - ii. Table of the predicted size over time (to determine initial FILEGROWTH based on the review period)
 - iii. Table of the exceptional size over time (to give initial MAXSIZE and partition size)
 - d. Log file capacity is based on expected transaction rate. In some cases, this is based on the business plan (number of customer purchases) but may need to be estimated (number of enquiries):
 - i. Initial log size
 - ii. Growth prediction
 - iii. Exceptional size
2. The database file can be created on relatively unlimited-sized HDP LUNs. For example, starting points like “the largest the database could ever grow to” multiplied by a factor of two or more. In normal use, a database file is unlikely to grow suddenly.
3. We recommend partitioning the database disk with the same value that we had planned for MAXSIZE. In fact, these two values would be changed in tandem. Warning: the partition size includes metadata; MAXSIZE does not.
4. The database file would be created with initial SIZE, FILEGROWTH and MAXSIZE values. These values can be changed during use. Ideally all values will be multiples of a page, 42MB.
 - a. The value for SIZE will be immediately allocated in the HDP Pool, so keep this as low as possible. However, if data will be immediately imported, include this in the initial SIZE.

⁶ There is a “catch-22” here. We are trying to minimize costs, but that means that we may need to be more agile when procuring new equipment. This will depend on how the budget is managed and whether budget reserves are permitted.

- b. FILEGROWTH should rarely be less than one page, 42MB. This value is a trade-off: too small and the database may be fragmented; too large and pool is wasted. You should probably aim for the database to be extended once per day. Database systems generally try quite hard to avoid fragmentation; storing “associated data” in the same or near disk blocks so that required data is either already fetched or the seek distance is small.
- c. MAXSIZE should never be more than you can afford. This value can prevent the database consuming too much pool space.

Example: Leveraging Capacity Leveling

A company has separate databases, each supporting one of its product lines. Planning for the quantity of data at such a micro level can be difficult. But with static provisioning that is what is required. Good sales in one product would lead to insufficient capacity, while another product may sell less than predicted. Applying the same planned growth factor to both product lines will lead to long-term wastage. Dynamic provisioning will level this out, supplying capacity where it is needed.

Static Provisioning

Product	Estimate	Growth	Contingency
A1	100GB	50GB	50GB
A2	200GB	50GB	50GB
A3	300GB	100GB	50GB
Total:	600GB	+ 200GB	+ 150GB = 950GB

Dynamic Provisioning

Product	Estimate	Growth	Contingency
Group A	600GB	0GB	0GB
Total:	600GB		

We are able to make a more accurate estimate at the group level. We budget to purchase the growth capacity but only do so when needed.

- 5. Extension database files (.ndf) should not be needed. Their spanned nature does not scale or level the performance load.
- 6. The log file (.ldf) is a little more sensitive to planning than the database as it is less controllable and predictable. Relatively unlimited-sized HDP LUNs may still be safe, but we would recommend partitioning them based on expected cost and use.
 - a. The logfile will be receiving burst sequential write traffic with the same disk areas being re-used, so contiguity will be beneficial to reduce seek overhead. You should aim for moderately large SIZE and FILEGROWTH values, such as 420MB.
 - b. The partition size should be the same value planned for MAXSIZE. They would be changed in tandem.
- 7. The log file can grow suddenly, for example when some management tools or reports are used. The log file never reduces in size; logfile space is internally deleted and re-used. This re-use is efficient, unlike NTFS space re-use. Space is recovered routinely when backup with log truncation is executed. If the backup process fails, the logs could grow large unexpectedly.

8. Because of its role in controlling log file use, backup should be part of the implementation planning, which includes:
 - a. Monitoring to be sure backup and truncation have occurred
 - b. Possibility of missed backups in the “maximum log size” calculation and configuration
9. Do not mix files of greatly different types in the same file system unless you are sure the space of all types will be controlled. Do not mix “general space for user files” with application data. Ideally the mdb and ldb files will have their own DP-VOL and file system. This is the Microsoft recommendation for SQL Server, too.
10. Do not defragment file systems on HDP volumes, including those containing database or log files. In a Hitachi Dynamic Provisioning environment, defragmentation of NTFS is rarely space efficient with any data.
11. Other files (.trn, .bak) will be on unmanaged file systems. This can be on static provisioned storage or on dynamic provisioned storage that is tightly controlled; the recommendation would be to make an estimate and create a partition of that value. This class of data would be a candidate for a scripted “autoexpand” solution. For backup via Hitachi ShadowImage Replication software, see the section later in this paper.
12. Beware of space efficiency when copying or moving application files around. Copying to a normal LDEV is efficient, but if files are copied back even if the originals are deleted, then too much space will be consumed. If you need to do this:
 - a. Prepare the application for the change (offline or detach the database).
 - b. Copy all data off the dynamic provisioned disk.
 - c. Delete any associated cluster resources.
 - d. Recreate the file system (quick format) and cluster resources and then copy the required data back using file level tools.

This process cannot recover pool space but it will avoid using more. It is effective if the data quantity copied back will soon be greater than or equal to that copied off.

13. If you overallocate a volume by mistake and need to reclaim all the space, the following process can be used:
 - a. Prepare the application to make the LUN disappear (offline or detach the database).
 - b. Copy data off the dynamic provisioned volume.
 - c. Delete any associated cluster resources.
 - d. Delete the drive letter or mount point, file system and partition.
 - e. Delete the LUN and DP-VOL, recreate the DP-VOL and remap it to the servers.
 - f. Once mapped to the server, you can recreate the file system, drive letters or mount points and cluster resources.
 - g. Copy the data back using file level tools.

This process can recover pool space.

SQL Server 2005 Testing Detail

This section shows some of the detail of the SQL Server testing. A powerful method for understanding how dynamic provisioning is working with an application is a simple chart comparing pool space used with file space used. By comparing the two lines we can see:

- If data is deleted from a file system the corresponding line will dip. However, the pool space line will never dip; this is impossible.
- Pool space used will always be greater than or equal to the file space used.

File system metadata is rarely included in operating system reports of file space used. The pool space used will therefore be greater than the file system space reported. Often this can be pre-calculated and the graph corrected. This makes comparison of the lines more valuable.

Use of new file system space is rarely uniform; there are steps in the graph. There will be steps in the pool usage graph too, but it is very unlikely that they would track one another exactly. Steps in the pool graph will of course be multiples of 42MB.

Although the lines could never be exactly parallel, whether they track one another over the long term is interesting information. Close tracked lines are the best behavior and what we are looking for.

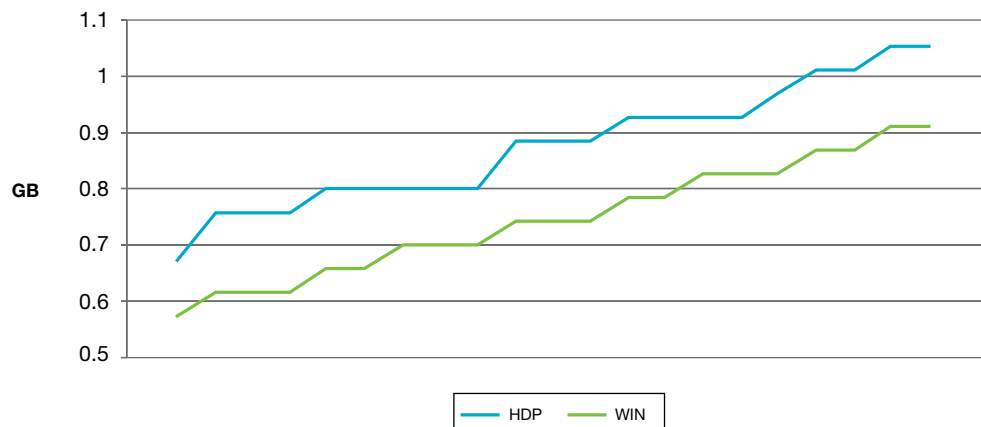
The problem behaviors we are looking for are:

1. Pool space initially allocated is a large proportion of the DP-VOL size. The ideal file system would only have a few percent of overhead. If this is seen, it normally means that the file system is grabbing a lot of pool space for metadata. We expect the lines to get closer to one another as the file system grows. However, in some cases this is not until the file system is completely full. Windows NTFS does not show this behavior. (Solaris UFS and HP-UX HFS exhibit this behavior).
2. Pool space used constantly grows faster than the file system. This most likely indicates that the file system is leaving unused gaps in the disk map. A small diversion is probably acceptable but a large difference may indicate additional strategies need to be followed. (Linux EXT3 and OCFS2 exhibit this behavior).
3. Pool space used grows faster than the file system but only occurs after file deletions. This indicates that file system disk blocks are not being re-used. Windows NTFS has this behavior so we need to ensure this does not result in wasting too much space.

The following charts show the amount of space used on the file system⁷ by a database MDF file (WIN) and the equivalent pool space consumed (HDP).

As shown in Figure 2, the SQL Server FILEGROWTH value for the database was 42MB. This chart is a section after several days of testing. The important fact is that the “HDP Pool space used” and the “Windows capacity used” lines have remained parallel, so space is not being wasted.

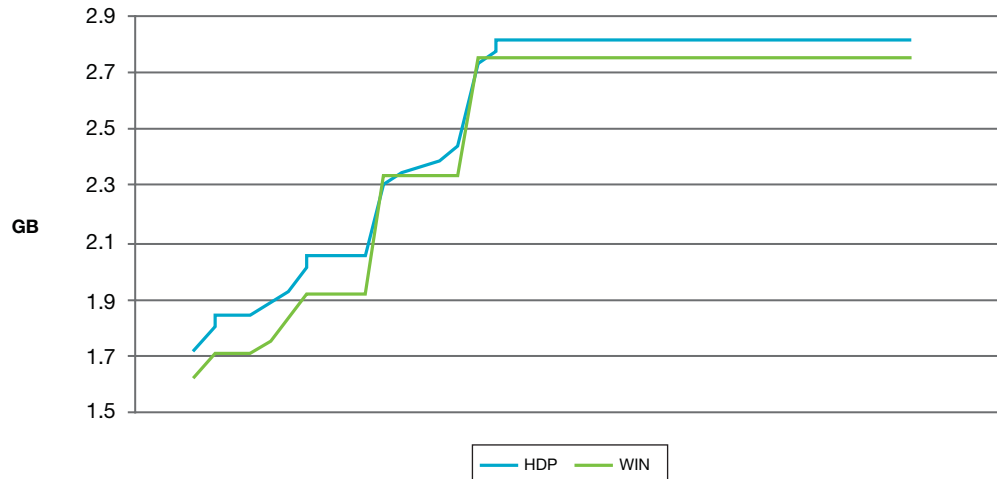
Figure 2. SQL Server Data (42MB Increment)



⁷ Note: the overhead associated with the file allocation areas has been included in the file system figure so the numbers could be directly compared.

Figure 3 shows the database after the FILEGROWTH value for the database was changed to 420MB. This uses a little more space but shows improved locality. This parameter can be changed while the database remains online. As before, the lines remain parallel.

Figure 3. SQL Server Data (420MB Increment)



Exchange 2003 and 2007 Executive Summary

Exchange 2003 and 2007 can maintain space-efficiency with Dynamic Provisioning under the conditions shown in Table 3.

Table 3. Space-efficiently Maintaining Microsoft Exchange 2003 and 2007

Database files	Can be on either capped or unconstrained, Hitachi Dynamic Provisioned (HDP) volumes. Follow normal guidelines for Windows NTFS file systems on HDP volumes.
Log files	Can be on either static volumes or HDP volumes capped using dynamic partition expansion. Never use large, unconstrained HDP volumes.
System Directory Files	Can be on either static volumes or HDP volumes capped by dynamic partition expansion.
Backup Files	May depend on the backup tool. Use static volumes or HDP volumes capped by dynamic partition expansion.

Exchange 2003 and 2007 Detail

Although there are many differences between Exchange 2003 and 2007⁸, the Jet database versions underlying them are similar and use the same file names and structure. From the operation view of Dynamic Provisioning, they act the same. Their performance characteristics are quite different, however, and this needs to be taken into account during performance planning.

Here are the guidelines for using Dynamic Provisioning with Exchange 2003 and 2007. It starts with the key data areas and then considers more general issues:

⁸ Exchange 2000 has not been tested, as it is unlikely to be deployed with Dynamic Provisioning; however, it is expected to perform in the same manner or in a manner similar to Exchange 2003.

- **Database files (.mdb)** — each database file holds an Exchange Message Store (a group of mailboxes). There are up to five Message Stores per storage group. Message Stores may be grouped on the same volume or spread across multiple volumes. Multiple Messages Stores in a storage group would have poorer contiguity. Storage Group is the Jet equivalent of a database. There are no storage physical schema parameters in Jet equivalent to SQL Server's initial allocation. One server/cluster can support up to 50 storage groups (Exchange 2007) or 4 storage groups (Exchange 2000 and 2003).

Each storage group should ideally be on its own HDP volume.

The default installation of Exchange will perform mailbox management every night. This is a safe activity and it will reclaim space that can be later reused. This is in contrast to NTFS defragmentation.

Do not defragment file systems on HDP volumes. In a Dynamic Provisioning environment, defragmentation of NTFS is not space efficient on any data.

Beware of space efficiency when copying or moving application files around. Copying to a normal LDEV is efficient, but if files are copied back even if the originals are deleted, then too much space will be consumed. If you need to do this:

1. Prepare the application for the change (probably offlining the message store).
2. Copy all data off the dynamic provisioned disk.
3. Delete any cluster resources.
4. Recreate the file system (quick format) and cluster resources, and then copy the required data back using file level tools.

This process cannot recover pool space but will not use more. It is therefore effective if the data quantity copied back will soon be greater than or equal to that copied off.

If you overallocated a volume by mistake and need to reclaim all the space, the following process can be used:

1. Prepare the application to see the LUN disappear (probably by offlining the message store).
2. Copy data off the dynamic provisioned volume.
3. Delete any cluster resources.
4. Delete the drive letter or mount point, file system and partition.
5. Delete the LUN and DP-VOL, recreate the DP-VOL and remap it to the servers.
6. Once mapped to the server, you can recreate the file system, drive letters or mount points and cluster resources.
7. Copy the data back using file level tools.

This process can recover pool space.

The recommendations in the management tasks section must be followed. The Exchange management console has tasks to relocate information. These are copy activities that need to be performed with care. Example: I wish to move a mailbox store from one volume to another volume. This will allocate pool space, as one would expect. If you then need to move another store into the original volume, you must follow the appropriate volume recycle procedure first.

- **Log files (.log)** — Exchange generates a new log file for every 1MB (Exchange 2007) or 5MB (Exchange 2000 or Exchange 2003) of transaction data. The number of log files per Storage Group is unlimited. The rate they are created is dependant on the rate at which messages are sent and received (this rate may change unpredictably and might be subject to Denial of Service attacks).

Log files are not truncated (deleted) until they have been backed up.

If not capped, the space allocated to an Exchange log volume will continue to grow — rapidly. The viable alternatives available, in order of recommendation are:

- Dynamic provisioning with dynamic partition expansion
- Dynamic provisioning with a small virtual volume size
- Static provisioning with a small static size

Example: Log Directory Growth

The log volumes of a large Exchange environment are configured based on a plan for daily backup and truncation of logs.

The tape library suffers a catastrophic failure and will take four days to repair.

With static provisioning, the administrator is unlikely to have allocated five times the planned capacity for logs. He will probably be forced to manually delete the logs and take the risk that an outage will lose up to five days of e-mail messages.

With dynamic provisioning, the administrator initially defined a large virtual volume capped to a smaller partition size. The administrator can now increase the “cap” on the log drive, a task of a few seconds.

The first has the greatest flexibility in size and performance; the second only flexibility in performance; the third is the least flexible.

A Log Directory should be on its own Dynamic Provisioning volume. Because of its 100 percent sequential write characteristic, it may perform better if in a different pool to the databases.

There is an Exchange mode called circular logging where, to save space, logs are deleted as soon as applied, soon after their generation. This severely limits the flexibility of backup recovery as continuous logs are essential to permit roll-forward of the database. For this reason, this option is rarely used. The behavior of Dynamic Provisioning is identical, whether circular logging is enabled or not. The space saving will not occur as multiple files are still generated and deleted.

Do not mix files of greatly different types in the same file system unless you are sure the space of all types will be controlled. Do not mix “general space for user files” with application data. Ideally the edb file will have its own DP-VOL and file system. This is the Microsoft recommendation for Exchange, too.

- **The Exchange system directory (mailroot)** — this directory has multiple uses and is often neglected in planning. By default, mailroot is placed on the C: drive but in certain scenarios this can lead to problems. Mailroot holds “pending” and “rejected” traffic. This can receive a surprisingly high amount of traffic. For example, if one server is shut down, all the mail to it from other servers will go to mailroot. Filling the mailroot directory is a common cause of Exchange server crashes. If an HDP volume is used for mailroot, it should be on space capped by a partition.

Mailroot should be replicated for resilience; as soon as a failure occurs, mailroot will be used to store pending e-mail. If this failure is the start of a disaster, then this pending data will be lost forever when the rest of the site fails.

- **Backup files** — unlike SQL Server, Exchange does not have its own backup mechanism or file format. How this performs with Dynamic Provisioning will depend on the backup application used. However, if backup to an HDP volume is used, this is most likely to be unmanaged space that should be capped to be effective.

For backup via Hitachi ShadowImage Replication software, see the section later in this paper. Backup using Hitachi Protection Manager and Hitachi ShadowImage Replication software is recommended for a zero disruption approach. It works well with Dynamic Provisioning volumes.

Replication and Hitachi Protection Manager Software

Hitachi Dynamic Provisioning currently supports replication with Hitachi ShadowImage Replication software (Contact Hitachi Data Systems for the latest Hitachi replication products supported by Hitachi Dynamic Provisioning). You would nearly always use Hitachi Dynamic Provisioning software to provision volumes for both primary and secondary volumes. ShadowImage Replication software is aware of which areas of the volumes are and are not allocated. The configured size of both volumes must be the same. If you use ShadowImage to copy a volume, the copy will be equally space efficient. Protection Manager, when used with ShadowImage Replication software will also operate efficiently. To minimize contention and maintain best efficiency, ShadowImage secondary volumes should use a different pool from the primary. Restore actions should not consume additional space as the volume has been derived from the primary. The only exception might be if a secondary volume was mounted and then changed significantly, increasing pool space of the secondary volume. If restored, the primary would also increase in size.

Example: We are using Hitachi Dynamic Provisioning on an Exchange database. The volumes are sized at 1TB for future expansion, but they only have 300GB of space allocated. By using Hitachi Protection Manager, we can create ShadowImage backups of the database that will be on volumes configured to 1TB, but they would also only have 300GB of space allocated.

Management Tasks

This section looks at management tasks performed on overprovisioned⁹ dynamic provisioned volumes on NTFS. It is important that management tasks do not cause pool space to grow. If volumes are to be fully allocated or are capped by some method, these recommendations are not needed.

Disk quotas cannot be used to prevent pool space wastage; they will limit how much data a user can retain but they could still create and delete large capacity allocations, using up all of the pool.

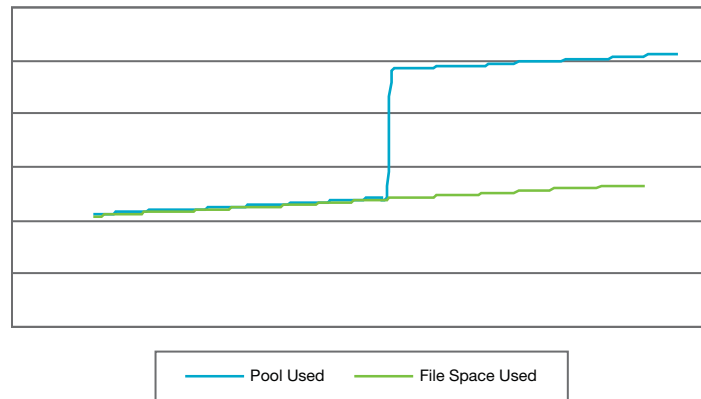
A few management tools cause excessive use of HDP pool space and they should be avoided or used with care. Some tools can allocate new pool space very rapidly. Tools that perform low-level block level writes should normally be avoided. Fortunately, Windows has few of these.

One key thing to avoid is defragmenting overprovisioned volumes. Defragmentation tools all look for and use the largest block of space they can find. This inevitably involves allocating pool space unnecessarily. In some cases the allocated space in a pool will double within seconds if defragmentation is performed. Due to the nature of Hitachi Dynamic Provisioning defragmentation does not have the beneficial effect on performance as shown with direct attached disks. Defragmentation may actually reduce performance on Dynamic Provisioned volumes. Note

⁹ DEFINITION: Overprovisioned (also overallocated, oversubscribed) — is where more space is allocated to a volume than you currently require. When volumes are overprovisioned then you can cope with unexpected increases in capacity demand. However, you will wish to control your applications to ensure they don't waste this space. Dynamic Provisioning allows you to consider overprovisioning, but it is not essential to use it to gain benefits from Dynamic Provisioning.

that during operation shown in Figure 4, the file system is, wrongly, defragmented, which immediately doubles the pool used.

Figure 4. Example of Exchange Database Defragmentation



Some complex tools such as consistency checkers or sort tools make use of temporary files or directories. The default location for these is often the same volume as the one managed. This should, of course, be avoided as the temporary space used will not be immediately reused.

Copying data into a file system must be performed by file level tools (such as copy, robocopy and file restore); block level copy tools will often result in an inefficient use of disk.

If you copy data from a file system onto the same disk, the pool space allocated could increase and will not be reclaimed until the disk is fully allocated. Similarly if you create, delete and recreate a significant proportion of files, the pool space will be double allocated. Renaming is generally a safe activity.

When copying of data cannot be avoided (such as with test and development environments) then the following methods can be used:

- If you are going to put back the same amount of data onto a volume, then you can use the operating system to delete and recreate the file system before copying data into it.
- If you are going to put back less data on the disk and wish to reclaim space, then you must delete the volume and reformat it to reclaim the space. A new volume can then be created.

Technology Discussion of the Use of Hitachi Dynamic Provisioning Software

All the previous discussions were related to “what happens” and “how to control what happens.” This section is different and discusses “why.” It is technical in nature and some aspects of operating system designs are included to provide a better understanding of Hitachi Dynamic Provisioning.

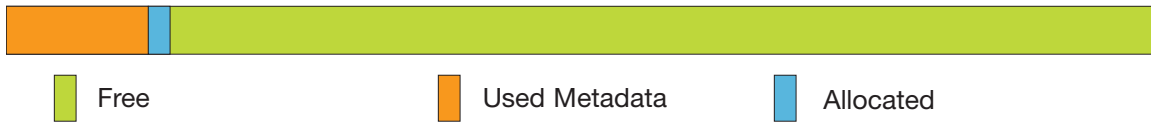
Metadata

When a volume manager diskgroup or file system is initialized, initial metadata is created. This information will eventually include: directories, file names, dates and times, security data and the locations of allocated and free disk blocks (the free pool). This metadata is a normal and essential part of the operation of an operating system and it is the reason why a disk of size X GB never actually holds X GB.

With file systems like NTFS, a single area for metadata is created at the beginning of the disk. Because Dynamic Provisioning data is allocated in 42MB pages, the amount of metadata pool space allocated will be rounded up to

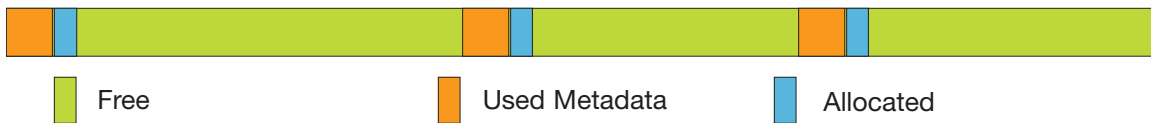
the nearest page boundary. The quantity is not published but in testing has been seen to be less than one percent. This space will soon be filled with data, as shown in Figure 5.

Figure 5. Metadata with File Systems Like NTFS



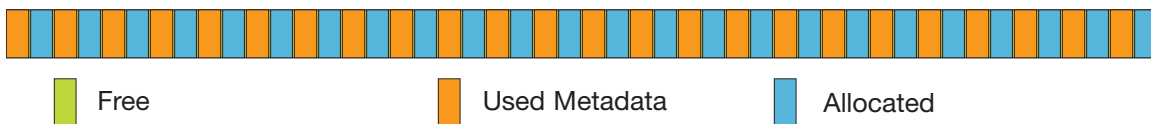
In some file systems, the metadata is not a single unit but spread over several parts of the disk. An example of this is Linux XFS, which writes metadata every 2GB, which would allocate two percent of the configured pool. This pre-allocated space is unlikely to be filled until the disk is nearly full, as shown in Figure 6.

Figure 6. Metadata with File Systems Like Linux XFS



Some file systems write small amounts of data at frequent intervals. The HP-UX file system HFS writes data every 10MB. This will allocate all the pool storage of the volume and no space saving is possible. This is shown in Figure 7. This pre-allocated space is unable to be filled until the disk is nearly full. There will of course still be the potential performance benefits that Dynamic Provisioning offers.

Figure 7. Some File Systems Allocate All Pool Storage of the Volume – Disallow Space Savings



Intermediate examples exist where although less than 100 percent is allocated, such a large proportion is allocated that only marginal storage savings are possible:

- Linux EXT2 and EXT3 will write metadata every 128MB, which would allocate 30 percent of the configured DP-VOL.
- AIX JFS with an 8MB allocation unit would allocate 100 percent of the configured DP-VOL. However, AIX JFS with a 64MB allocation unit would allocate 65 percent of the configured DP-VOL.

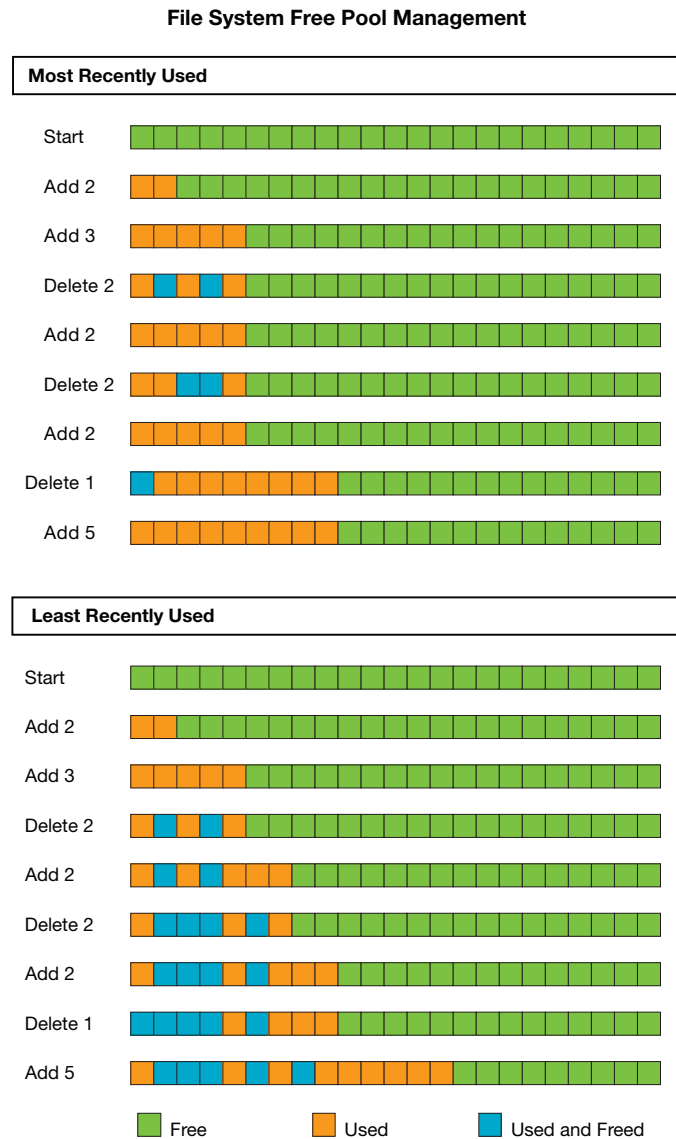
File System Storage Reclaim

All file systems have a “free data pool.” This pool will have different names for each file system, such as: free-queue, free-list or file table. All have some essential characteristics:

1. When requests for space arrive (either for new files or to expand an existing one) blocks are taken from the free pool and allocated to the file. This request will always be associated with a disk write, so the underlying HDP pool blocks will become allocated.
2. Files will also be deleted (or truncated). This involves putting the now unused blocks back into the free pool. The storage is not informed when data is freed (there is no way to do so); the server just stops accessing that area of space.

There are two major categories of pool management¹⁰: Most Recently Used or Least Recently Used, as shown in Figure 8.

Figure 8. Behaviour of MRU and LRU Allocation Algorithms Given the Same Sequence. Notice how the LRU algorithm touches more pages.



Most Recently Used (MRU) — in this strategy, blocks that have just been freed are allocated in preference to blocks that have not yet been used. Freed blocks are placed at the front of the free-queue.

Least Recently Used (LRU) — in this strategy, blocks that have not yet been used are allocated in preference to blocks that have just been freed. Freed blocks are placed at the end of the free-queue.

¹⁰ Actually there are hundreds of strategies. All, however, will have either a least- or most-recently used characteristic. The goals of free pool management are: minimizing fragmentation and maximizing contiguity. This might involve multiple queues for different sized allocations or sorted queues that allow fragmented data to be easily coalesced.

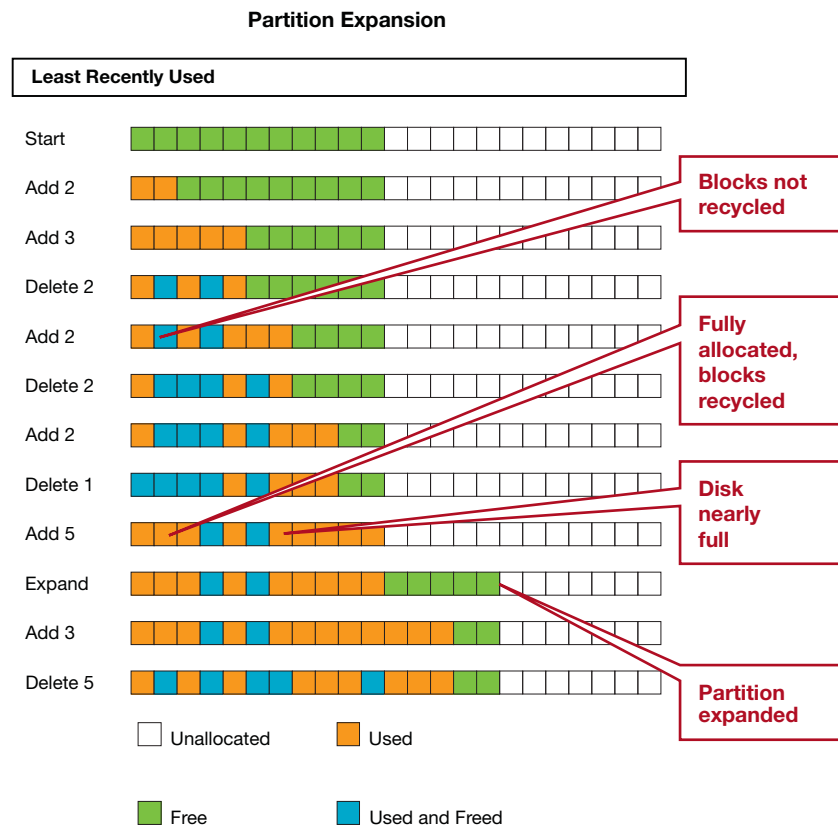
The Most Recently Used strategy is the most thin provisioning-friendly; the amount of pool storage allocated will, generally, be equal to the largest the file system has grown.

The Least Recently Used strategy is less friendly to thin provisioning; if space is regularly created and deleted, the new space will all be coming from new areas of disk. Dynamic Provisioning will have to allocate more and more pages from the pool. Unfortunately, Windows NTFS uses an LRU strategy. At first sight this might imply that Dynamic Provisioning and Windows are not best suited. But there are strategies that will “keep Windows on a diet.” These strategies depend on the type of application data.

Dynamic Partition Expansion with Hitachi Dynamic Provisioning

The poor storage reclaim characteristics of NTFS can be avoided with a straightforward strategy. When you know the file access pattern of your application is likely to overuse pool space, you can still continue to create oversized devices, but you would not use all the space at the outset. You can use Windows partitioning to control the actual allocation. NTFS partitions of any size can be created and may be resized online. It only takes a few seconds to allocate additional space. This is the best strategy to adopt for “unmanaged space” and spaces where files are continuously created and deleted, such as log files. (See Figure 9.)

Figure 9. Use of a Dynamic Partition.



By initially capping the space, the file system is forced to recycle space instead of allocating more. Only when the partition is nearly full do we expand the partition and potentially use additional pool space.

Dynamic Partition Expansion in Windows

Performing partition expansion is simple. Below is a script to expand disk 51 to 600MB.

```
echo select disk 51 > diskpart.scr
```

```
echo select partition 1 >> diskpart.scr
```

```
echo extend size=600 >> diskpart.scr
```

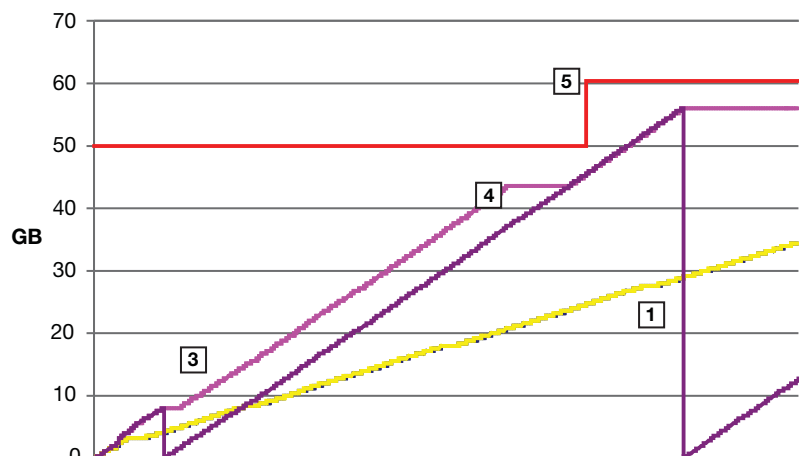
```
diskpart /s diskpart.scr
```

This script was used in the next example.

Exchange 2007 Extended Example

Figure 10 shows an example of Exchange 2007 in use over an extended period. The configuration has been done according to the guidelines. Although Exchange 2007 was used for this sample, Exchange 2003 would perform identically.

Figure 10. An Extended Example of Use for an Exchange 2007 Configuration



- 1** These lines show the database edb file size and the matching HDP allocated pool. There are two lines in the graph but they are on top of one another. This illustrates that the used and allocated space track one another accurately. Close examination at point [1] (and other places on the line) will show a slight flattening. This is the Exchange Mailbox Management Process successfully recovering space when scheduled each night.
- 2** This line shows the total file system space used for the log files. You will observe that this goes back to zero twice. This is because the database has been backed up and the logs truncated.
- 3** This line shows the pool space allocated for the log files. It cannot return to zero. Dynamic Provisioning volumes cannot do this without being destroyed. At point [3] you will see pool pages start to be consumed again, even though space has been recovered. This issue with NTFS was discussed in the section on File System Storage Reclaim.
- 4** At point [4] the file system space has all been allocated so NTFS starts to reuse blocks. The line flattens and rejoins line [2].
- 5** This line shows the size of the NTFS partition. This implementation follows the guideline and uses dynamic partition expansion. The allocated size (100GB) is larger than the partition created on Windows (50GB). This caps the space to 50GB. At point [5] we show dynamic partition expansion. It is clear that the logs will overflow the 50GB allocated; so the administrator increases it to 60GB.

The use of Hitachi Dynamic Provisioning (HDP) for both database and log files is shown, following the guidelines. The database was subjected to a constant load.

Monitoring

You may wish to monitor the use of HDP Pools quite carefully. There are several techniques available:

1. The Dynamic Provisioning software will generate alerts if the pool exceeds its configured thresholds.
2. The above alerts can be configured to be passed through SNMP TRAPs to your management system.
3. Hitachi Tuning Manager software has additional features for reporting the trends of pool and LDEV usage. See “Hitachi Tuning Manager Hardware Reports Reference” for more details.
4. In addition, you can configure Tuning Manager to generate alerts when usage exceeds configured values or if usage trends look dangerous.
5. RAID Manager can be used to report on pool and individual DP-VOL usage. This can be used to develop custom monitoring solutions.
6. You may wish to combine the above monitoring with scripts to adjust dynamic partitions.

RAID Manager Monitoring

RAID Manager can give you an instantaneous view of pool usage at the DP-VOL and pool level.

raidvchkdsp -g HDP -fx -v aou

<i>Group</i>	<i>PairVol</i>	<i>Port#</i>	<i>TID</i>	<i>LU</i>	<i>Seq#</i>	<i>LDEV#</i>	<i>Used(MB)</i>	<i>LU_CAP(MB)</i>	<i>U(%)</i>	<i>T(%)</i>	<i>PID</i>
HDP	MDF	CL6-B-1	18	35	10044	b906	2814	93750	1	70	9
HDP	LDF	CL6-B-1	18	36	10044	b907	4032	93750	1	70	9
HDP	AUX	CL6-B-1	18	37	10044	b908	8442	93750	1	70	9

The above report shows the status of a group configured in RAID Manager. Suitable Group and PairVol names have been chosen to help identify the use. If groups are not configured, then the similar raidvchkscan command can be used. The first data line shows: Group HDP contains a Volume MDF. This is mapped as Host Storage Domain CL6-B-1, target ID 18 and LUN 35. The storage serial number is 10044 and the LDEV number is hex B906. The application (SQL Server, MDF file) has used 2814MB of the allocated capacity 93750MB. The pool threshold is 70 percent. This volume is associated with pool 9, and 1 percent of this pool is currently used. Note: the threshold and usage are for the total pool, not the LDEV. To get LDEV usage ratio, you need to divide 2814 by 93750.

raidvchkscan -v pida

<i>PID</i>	<i>POLS</i>	<i>U(%)</i>	<i>AV_CAP(MB)</i>	<i>TP_CAP(MB)</i>	<i>W(%)</i>	<i>H(%)</i>	<i>Num</i>	<i>LDEV#</i>	<i>LCNT</i>	<i>TL_CAP(MB)</i>
009	POLN	1	819714	820302	80	70	16	40	104	3334687
010	POLN	0	3286038	3286038	80	70	64	28672	154	11426142

The above report shows the status of the pools on the attached storage. The first data line shows: pool 9 is in a normal state (POLN), 1 percent of the pool has been used, 819714MB is available from a starting size of 820302MB. The warning threshold is 80 percent and the high water mark is 70 percent. There are 16 LDEVs in this pool and the first pool volume is LDEV 40. The pool has 104 DP-VOLs using it; they are configured to be able to expand up to 3334687MB. However, 3334687MB is greater than 820302MB, so we cannot grow the volumes to this size without increasing the pool size first. We would call this an overprovisioned pool.

For more information on RAID Manager commands see: “Hitachi Command Control Interface (CCI) User and Reference Guide.”

Performance Factors

Tests have shown that the performance design and consequent disk design recommendations for dynamic provisioning are similar to static provisioning. But for dynamic provisioning, the requirement is on the pool, rather than the array group. Further, the pool performance requirement (number of IOPS) will be the aggregate of all applications using the same pool (which is also true if a static array group supports multiple applications).

Pool design and use depends on the application performance required. Guidelines for dynamic provisioning pool design should be followed; for example, a minimum number of drive groups, recommended LDEVs per pool and LDEVs per drive group.

Application performance and capacity requirements must be met for the design of Hitachi Dynamic Provisioning pools. You may need to evaluate more than one workload at the same time. This will give recommendations for: RAID level, the minimum number of spindles necessary for performance and the number of spindles required for the capacity. Table 4 shows general requirements for analyzing applications.

Table 4. Application Capacity and Performance Requirements

Application Performance/Capacity Pattern		Application Capacity Required	
		Small Capacity	Large Capacity
Application Performance Needed	Low IOPS	1. Minimum pool size always applies, so this workload must be accommodated by combining it with other applications.	2. This is a good candidate for Hitachi Dynamic Provisioning. It may have spare performance for supporting a high-IOPS application.
	High IOPS	3. This is the problematic area, as the number of disks needed for performance will exceed the number of disks required for capacity. This workload must be combined with type 1 or type 2 in a common pool.	4. This is a potential candidate for Hitachi Dynamic Provisioning, possibly with a separate pool (or pools).

It is not in the scope of this document to give detailed operations or guidelines for pool design. However, there are a number of standard recommendations:

1. You should not have different drive types for Pool-VOLs registered in a pool.
2. It is a best practice to have the same RAID level for all Pool-VOLs registered to the same pool.
3. For performance reasons, it is best to dedicate array groups to a pool. Four or more array groups are recommended for performance sensitive applications. Distribute array groups evenly over multiple back-end directors if possible.
4. Multiple Pool-DEVs are recommended from each array group. Four are recommended.
5. Consider pool expansion; do not use the maximum number of Pool-DEVs early.
6. External devices may be used in a pool but should not be in a pool along with internal devices.
7. External Pool-VOLs with different cache modes cannot be intermixed in the same pool.
8. The Pool-DEVs used to construct a pool must be between 8GB and 4TB in size.
9. You cannot remove Pool-DEVs from a pool.

10. Pools should always be initialized (optimized) after creation.

Additional guidelines are available from your Hitachi Data Systems representative. Also see the “Hitachi Universal Storage Platform™ V and Universal Storage Platform VM Dynamic Provisioning User’s Guide.”

For multiple, large or performance-intensive databases, the performance requirements may justify or require the database to have a separate pool or pools. This is not dissimilar to static provisioning where one might dedicate array groups to an application. Separate pools for each database and separate pool or pools for logs will have benefits but may not always be justifiable unless the capacity is large.

Leveraging Performance Leveling

A heavy online transaction processing (OLTP) application might need a peak IOPS requirement that can be met by four array groups in a RAID-10 2+2 configuration. However, the application only requires storage of one array group.

We also have a requirement for a file archive with minimal I/O load but a significant amount of storage, say four array groups.

A possible pool design would be a single HDP pool with five RAID-10 array groups.

Leveraging Capacity Leveling

Exchange is a performance critical application. Many Hitachi storage customers employ RAID-5 in all their applications without issues. Exchange is one of the few applications where RAID-10 is often justified. Fortunately, Exchange databases are also relatively large so they need large numbers of disks to support the data. This quantity of disks can support the required I/O load provided the load is evenly distributed.

For example: an administrator follows Microsoft guidelines and places organizational groups in the same mailbox store. To maintain performance, they have balanced this workload across multiple storage systems. But a policy change means that the members of one group are all given a BlackBerry®. Microsoft estimates that a BlackBerry user generates 3-12 times the I/O load of a normal user. This could require a costly rebalancing exercise on static storage.

With Hitachi Dynamic Provisioning, it would not matter that the workload was unbalanced since Dynamic Provisioning would naturally level the load.

Recommendations for Adding Disks to a Pool

The high performance of Hitachi Dynamic Provisioning software is enabled by the fact that a large number of disks are used in the pool. However, if additional disks are added to a pool, data allocated after that time may experience different performance characteristics. The amount of pool left when you add storage will have a significant impact.

Two extreme examples will illustrate this:

I start with a pool constructed from four array groups. All data allocated will have a similar performance of four “units.”

1. If I use all of the pool up and then add two storage array groups, data allocated after this would have a performance of two units. (This is actually the same as creating a new pool with only two storage array groups).
2. If instead I use none of the storage and add two storage array groups, data allocated after this would have a performance of six units. (This is actually the same as creating a new pool with six storage array groups).

Obviously, no real environment would have either of these extremes. The recommendations for adding disks to a pool are:

- Do not wait until the last minute to add to the pool unless you plan to add at least the same number of array groups or you are happy that newly allocated data performs more slowly.
- Always add relatively large amounts of storage to the pool.

Capacity Planning Factors

At first sight, some capacity planning calculations might seem simple: for 1000 mailboxes, each with a quota of 100MB, the result is 100GB of space required. But additional factors consume extra space.

One is the Exchange “dumpster,” an area of the Exchange mailbox where deleted e-mails are retained before permanent deletion. This does not use up the users quota, but it does consume space on disk. You can try to plan for the additional space for the dumpster; basing the calculation on expected message receipt rate, but this can only be a rough prediction. Using traditional static provisioning, the system manager must allocate additional storage, over and above this prediction, to ensure his Exchange system does not crash. This headroom space should, however, never be used. If it is used, your planning was wrong and there is a danger that future usage will cause Exchange to crash.

Other application data spaces are also usage rate dependant. A bank may be able to predict quite accurately how many customers it has, so planning this space may be relatively easy. But “transactions per customer” can vary greatly. Some retail corporations are very seasonal; for example Christmas sales may compose a significant proportion of the company’s annual business. Planning for this can be difficult and computer issues in the holiday season are becoming standard. “We haven’t enough storage to record the huge number of orders for our new product” is not going to be an acceptable excuse in the boardroom. “We have the storage but we’re going to have to close the Web site for half a day to move all the data around” is little better. Even our “expected number of customers” calculation can be thrown out by a particularly successful marketing plan. To summarize: if the rest of the business is doing their job – the IT department has a planning problem.

Dynamic Provisioning can give us additional tools for reducing problems with capacity planning. However, it does not mean:

- Capacity planning is no longer required.
- There is no longer a need to monitor space usage.
- Managing storage can be avoided.

You still have to do all these proper storage management practices but hopefully with lower risks.

Appendix A – Hitachi Tuning Manager with Hitachi Dynamic Provisioning Software

Hitachi Tuning Manager software is a comprehensive storage management tool to monitor, trend and analyze performance data to effectively manage storage using Hitachi Dynamic Provisioning software. Tuning Manager reports and alerts are customized from the base data sets. New data sets supporting Dynamic Provisioning include: pool configuration, pool usage trends, virtual volume configuration and virtual volume usage trends. Dynamic Provisioning specific parameters include: FREE_CAPACITY, POOL_ID, POOL_VOLUME_COUNT, STATUS, THRESHOLD, TOTAL_ACTUAL_CAPACITY, TOTAL_MANAGED_CAPACITY, USAGE_RATE, USED_CAPACITY, VIRTUAL_VOLUME_COUNT, and WARNING_THRESHOLD.

In addition, standard data sets, such as Logical Device configuration, and LUSE configuration now include additional fields to monitor Dynamic Provisioning parameters. Parameters include: POOL_ID, and VOLUME_TYPE.

By combining standard and Dynamic Provisioning-specific data many powerful reports can be constructed.

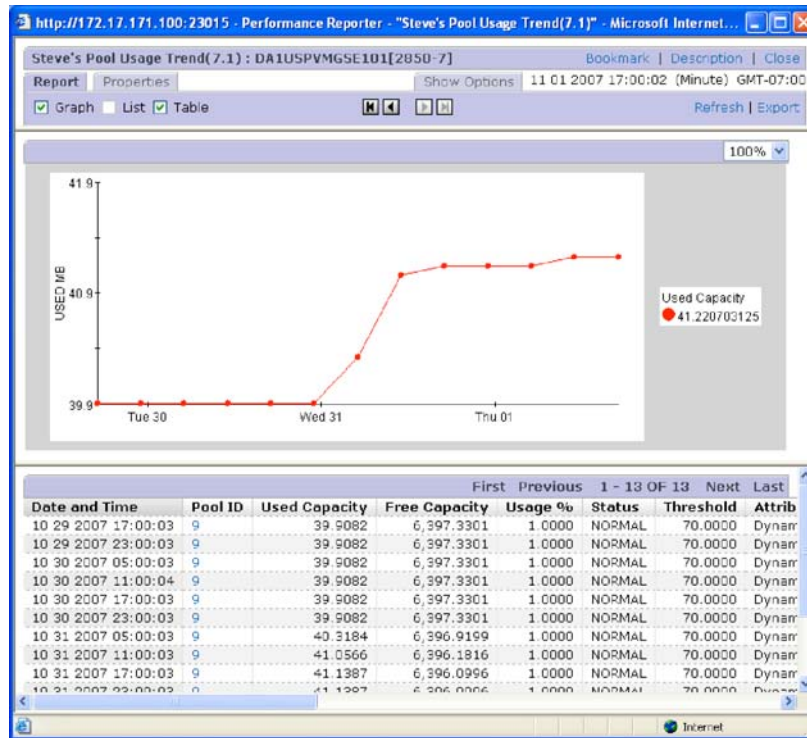
To obtain performance information related to the usage of the array group that contains pool volumes for Hitachi Copy-on-Write Snapshot software (formerly QuickShadow) or Dynamic Provisioning, you need to specify Y or y for Unassigned Open Volume Monitoring when setting up the instance environment for the Agent for RAID.

As well as storage data, Tuning Manager has agents for collecting data from: servers (such as CPU, memory and file system usage) SAN components (port I/O, errors) and agents for specific applications such as Oracle, Exchange, SQL Server and IBM® DB2®. These sources can be combined to give an end-to-end view of an application and all the resources it depends upon.

A few example reports follow to illustrate some of the possibilities when using Tuning Manager with Dynamic Provisioning.

Figure A1 shows a customized report that examines the trend in pool usage for Pool_ID 9 over a three-day period.

Figure A1. Customized Report—Three Day Pool Usage

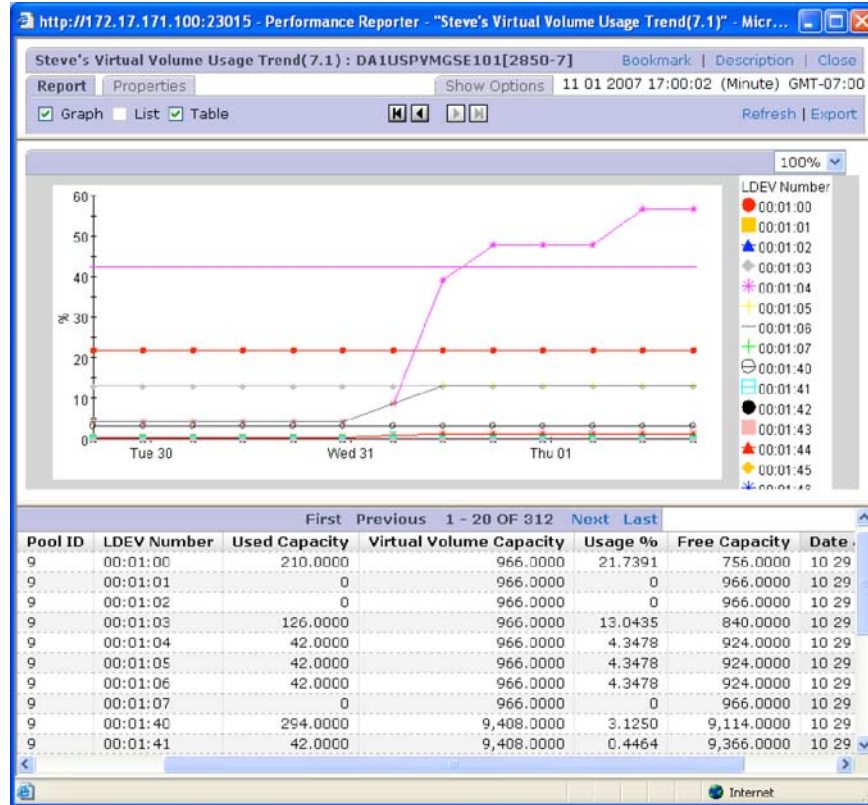


Customized alerts can be created to monitor pool usage in a number of ways. If thresholds are exceeded then one of the following options can be configured:

- Sending of an e-mail
- Raising of a system alert
- Creating an SNMP TRAP
- Running a program: possibly one to use Hitachi Device Manager scripting to configure additional storage for the pool.

Figure A2 shows the trend of pool usage for a selected range of Virtual Logical devices that use a specified pool. LDEV 00:01:04 is growing rapidly and is already 60 percent of its configured capacity.

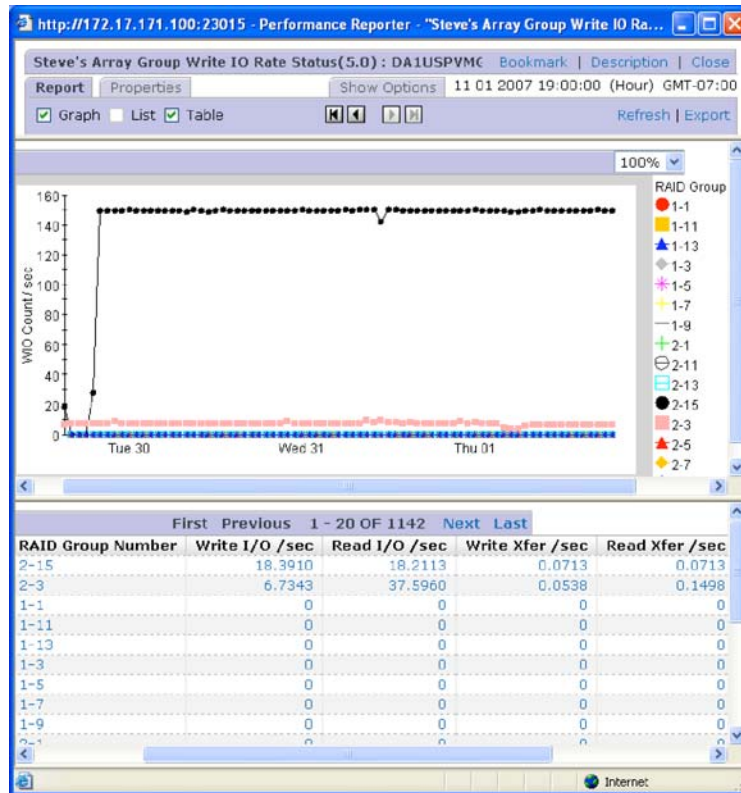
Figure A2. Customized Report – Virtual Volume Pool Usage



Of course standard Tuning Manager reports are also of value when analyzing Dynamic Provisioning data.

Drive group reports such as the screen shown in Figure A3 will be relevant for monitoring the performance of both static provisioned volumes and array groups allocated to Hitachi Dynamic Provisioning pools.

Figure A3. Drive Group Report



Data on the performance or use of V-VOL groups are not included in any of the standard reports; this would have little benefit as the V-VOL group has no physical significance.

In a similar way, standard or customized LDEV reports can be used to monitor:

- Static provisioned LDEVs
- Dynamic provisioned LDEVs
- LDEVs used to construct dynamic provisioning pools

Appendix B – Testing Methodology Used

This document required the testing of similar aspects of four classes of application

1. General use of the NTFS file system — tests giving us the required understanding for the application testing which is all based on NTFS files
2. Exchange 2003 use in normal scenarios
3. SQL Server 2005 use in normal scenarios
4. Exchange 2007 use in normal scenarios

The goal was to identify the consumption of pool, as reported by RAID Manager and compare that against the “size of the file system” as reported by Windows. To be comparable, the latter number has to be corrected to take account of the metadata that Windows initially consumes (and does not report). These figures were monitored at each stage of all the testing.

NTFS File System Tests

The core operations tested were in three phases:

Creation

The normal processes of file system creation were followed: creation of partitions and creation of file systems. At each step, RAID Manager and Operating System commands were run to monitor pool space used and file system space used.

Soak Testing

Scripts were written which would exercise normal file operations a large number of times, with different file sizes: creation of files, deletion of files and file append operations. Several copies of these scripts were run in parallel as we wish to create internal block fragmentation. Stating of file deletion was delayed until the write-only characteristics were fully determined. Several GB of data was generated.

Management Operation Tests

At the end of testing, we tested file partition expansion to prove it does not waste pool. Defragmentation was also tested at the end because, as predicted, it wastes pool space.

Exchange 2003 Tests

Creation

We expected different characteristics from Exchange message stores, Exchange logs and Exchange backup files, so they were partitioned on separate HDP LUNs. Once the file systems were created, Exchange System Manager was used to create the files in the normal way (one SG, one SG per LUN, two MB per SG).

Exchange 2003 Soak Testing

Rather than write scripts for Exchange 2003, we used a tool that generates a representative workload. The tool used was Microsoft’s Exchange LoadSim. This uses the same APIs as real Exchange clients to run through a representative daily workload for thousands of users. E-mails and appointments were sent, read, deleted and replied to. Single addresses and distribution lists are used. Messages of various sizes were used with and without attachments. We wanted to ensure that data was deleted rapidly to see if it was correctly recycled, so data retention policy was set to zero; apart from that the installation was standard. Testing ran for 10 days, 4.8m transactions occurred, 1.3m messages were read, 1.25m messages were delivered and 350K were deleted.

Management Operation Tests

Exchange logs will grow forever unless truncated. This is performed at backup time. Backup was performed several times during testing to confirm operation. Hitachi Protection Manager software was used to back up the Exchange 2003 system. Mailbox management was left to its default nightly schedule to determine whether it caused problems. At this time, we also tested file partition expansion to prove it does not waste pool and to prove it forced block recycling. File system defragmentation was run at the end because, as predicted, it wastes pool space.

SQL Server 2005 Tests

Creation

We expected different characteristics from SQL Server database files, SQL Server logs and SQL Server backup files, so they were partitioned on separate HDP LUNs. Once the file systems were created, SQL Manager was used to create the database files in the normal way. The datafiles and logfiles were created with initial size 42MB, growth 42MB, and a large maximum size. System databases were left on the C drive, as in a normal installation.

SQL Server 2005 Soak Testing

There was no equivalent of LoadSim for SQL Server. Scripts were written to create and randomly populate and delete records from a database. During several days of testing, 15 million records were added or updated, and 8 million were deleted.

Management Operation Tests

SQL Server logs will grow forever unless truncated. This is performed at backup time. Backup was performed several times during testing to confirm operation. SQL Server backup was used to perform this.

Exchange 2007 Tests

Creation

We expected different characteristics from Exchange message stores, Exchange logs and Exchange backup files, so they were partitioned on separate HDP LUNs. Once the file systems were created, Exchange System Manager was used to create the file systems in the normal way (one SG, one SG per LUN, two MB per SG).

Exchange 2007 Soak Testing

Scripts and a load generator were used for Exchange 2007 testing. The scripts sent randomly sized e-mails (a subset of the Loadsim messages with and without attachments). Messages were also deleted randomly. These scripts permitted a controlled examination of the create/delete process. In addition, the Microsoft tool Exchange LoadGen was used. This uses the same APIs as real Exchange clients to run through a representative daily workload for thousands of users. E-mails and appointments were sent, read, deleted and replied to. Single addresses and distribution lists were used. Messages of various sizes were used with and without attachments. We wanted to ensure that data was deleted rapidly to see if it was correctly recycled, so data retention policy was set to zero; apart from that the installation was standard.

Management Operation Tests

Exchange logs will grow until Exchange limits are reached unless truncated. This is performed at backup time. Backup was performed several times during testing to confirm operation. The NT Backup tool was used to back up and truncate the logs. Mailbox management was left to its default nightly schedule to determine whether it caused problems. At this time, we also tested file partition expansion to prove it does not waste pool and to prove it forced block recycling. File system defragmentation was run at the end because, as predicted, it wastes pool space.

Appendix C – Hitachi Dynamic Provisioning Software Terms

The following Hitachi Dynamic Provisioning terminology and their respective definitions have been used throughout this paper:

- **Array Group** – a grouping of multiple disk drives combined to form a single pool of data with a specified RAID level.
- **LDEV (Logical Device)** – a volume created from the space within a particular array group. An LDEV is partitioned from that array group's raw space.
- **HDP Pool (pool)** – a pool of physical storage capacity that is allocated to dynamic provisioned virtual volumes (DP-VOLs). The actual data is stored here in 42MB pages and it is also referred to within this document simply as "pool."
- **Pool Volume (Pool-VOL)** – a regular LDEV used to construct a pool. For performance reasons a pool will normally have multiple pool volumes from multiple array groups.
- **V-VOL (Virtual Volume)** – a volume where space is not allocated statically at creation. Only the maximum size is specified, up to 2TB. Space will be allocated from a pool as it is required. Hitachi Dynamic Provisioning and Hitachi Copy-on-Write Snapshot software products currently use V-VOLs.
- **DP-VOL (Dynamic Provisioned Virtual Volume)** – a Hitachi Dynamic Provisioning volume created from an HDP pool. It is one of the virtual volume types supported by the Hitachi Universal Storage Platform™ V and Universal Storage Platform VM. A DP-VOL is a member of a V-VOL group.
- **LUN (Logical Unit Number)** – the host-visible identifier assigned by the user to an LDEV when mapped to a Storage Port or Host Storage Domain.
- **V-VOL Group** – a container for virtual volumes. One or more V-VOLs are configured within this space, each with a specified maximum size. The total of these sizes must be less than the size of the V-VOL Group. The maximum size currently supported for a V-VOL Group is 3TB. A V-VOL Group can be thought of as a virtual array group (in reports a normal array group might show as 2-4, a dynamic provisioned V-VOL group as X5-1 and a Copy-on-Write Snapshot V-VOL group as V2-1).



Corporate Headquarters 750 Central Expressway, Santa Clara, California 95050-2627 USA

Contact Information: + 1 408 970 1000 www.hds.com / info@hds.com

Asia Pacific and Americas 750 Central Expressway, Santa Clara, California 95050-2627 USA

Contact Information: + 1 408 970 1000 www.hds.com / info@hds.com

Europe Headquarters Sefton Park, Stoke Poges, Buckinghamshire SL2 4HD United Kingdom

Contact Information: + 44 (0) 1753 618000 www.hds.com / info.uk@hds.com

Hitachi is a registered trademark of Hitachi, Ltd., and/or its affiliates in the United States and other countries. Hitachi Data Systems is a registered trademark and service mark of Hitachi, Ltd., in the United States and other countries.

ShadowImage is a registered trademark and Universal Storage Platform is a trademark of Hitachi Data Systems Corporation.

IBM and XRC are registered trademarks of International Business Machines Corporation.

Microsoft is a registered trademark of Microsoft Corporation.

Hitachi Data Systems has achieved a Microsoft Competency in Advanced Infrastructure Solutions.

BlackBerry is a registered trademark of Research in Motion Ltd.

All other trademarks, service marks and company names are properties of their respective owners.

Notice: This document is for informational purposes only, and does not set forth any warranty, express or implied, concerning any equipment or service offered or to be offered by Hitachi Data Systems. This document describes some capabilities that are conditioned on a maintenance contract with Hitachi Data Systems being in effect, and that may be configuration-dependent, and features that may not be currently available. Contact your local Hitachi Data Systems sales office for information on feature and product availability.

Hitachi Data Systems sells and licenses its products subject to certain terms and conditions, including limited warranties. To see a copy of these terms and conditions prior to purchase or license, please go to <http://www.hds.com/corporate/legal/index.html> or call your local sales representative to obtain a printed copy. If you purchase or license the product, you are deemed to have accepted these terms and conditions.

© Hitachi Data Systems Corporation 2008. All Rights Reserved
WHP-270-00 January 2008