Hitachi Dynamic Provisioning
The Unofficial Best Practice Guide

RupturedMonkey.com
Removing the monkey from the storage manager's back...

http://blogs.rupturedmonkey.com

V1.3

Contents

UPDATE TRACKER

Added in v1.3 –

Zero Page Reclaim Theory in Section 6 (two pages)

1. Introduction

It is intended that this document will help you in your deployment of HDP/ThP, and give you a broad understanding of the technology as well as help you ask the right questions of your HDS and HP representatives.

Although the document originated from a Best Practices/Cheat Sheet, it has matured and evolved thanks to feedback and insights from many HDS and HP customers using the technology, as well as some helpful and knowledgeable people at HDS. Hopefully it is on the way to becoming a true community effort.

This document is written to correspond as much as possible with the latest version of microcode on a HDS USP-V. It will be updated accordingly, by the author, on a best effort basis.

Cross-reference of important technology names between HDS and HP

HDS name	HP name
HDP	ThP
USPV	XP24000
DP-VOL	V-VOL
ShadowImage	Business Copy
TrueCopy	Continuous Access
Storage Navigator	Remote Web Console / CommandView Basic

The above are registered trademarks of Hitachi Ltd, HDS and HP.

Throughout this document the author will use the HDS terms.

DISCLAIMER: At the time of writing this document, I do not work for Hitachi, HDS or HP. The information contained within this document is from my personal knowledge and experience with Hitachi Dynamic Provisioning software, and is in many instances my own opinion. Some recent additions have been added as a result of feedback received by the author. As such, this document should not be considered authoritative. Your HDS or HP representatives should have the latest and greatest Best Practice advice.

Also, the information in this document is subject to being rendered out of date due to new releases of microcode and updated best practice advice from the vendors. However, all effort is made to keep it up to date and useful to those who work with HDP and help us all implement and use HDP appropriately.

For more information from the author regarding HDP, see –
http://blogs.rupturedmonkey.com/?cat=18

For more information from the author concerning storage technologies, especially relating to Hitachi (HDS, HP) storage, see –
http://blogs.rupturedmonkey.com

2. POOL Pre-requisites and Restrictions

- The required Shared Memory DIMMs **must** be installed to the DKC to enable HDP to work. HDP has dedicated DIMMs for the storing of HDP constructs such as the Dynamic Mapping Table (DMT) etc during HDP operations. If they are not physically installed, HDP will not work.

- All LDEVs used to create a Pool, referred to as Pool-VOLs, must be OPEN-V and between 8GB and 4TB in size.

- Each Pool can be comprised of a maximum of 1024 LDEVs (Pool-VOLs).

- A maximum of 128 Pools can be created per subsystem. This is the combined total of HDP and COW Pools.

- A Pool can be between 8GB and 4096TB.

- All Array Groups used to build a Pool must be mapped to the same CLPR.

- An LDEV cannot be used as a Pool-VOL if it has a path definition (that is, presented to a front end port).

- Once an LDEV is added to a Pool, as a Pool-VOL, it cannot be removed without deleting the **entire** Pool. **Double check before adding Pool-VOLs!**

- You cannot delete a Pool that has DP-VOLs associated with it.

- 8192 DP-VOLs can be associated with each Pool.

- Only OPEN-V DP-VOLs can be created.

- With some early versions of HDP, DP-VOLs could not be dynamically expanded. Functionality to dynamically expand DP-VOLs is now available to certain Operating Systems under certain circumstances. See Section 4 “DP-VOL Size Recommendations” for further details.

- It is recommended that Windows servers be placed in Host Storage Domains with a host mode setting of 2C (Windows Extension). This is more future proof than 0C and will be a pre-requisite for future enhancements to HDP and other HDS software. No additional overhead is incurred by using mode 2C.

3. POOL Performance

HDP Pools can be configured in many ways, and the optimum configuration for one environment may not be the optimum configuration for another. This section outlines 2 areas of best practice in relation to Pool performance –

3.1 **Global Best Practices**, which should apply to most, if not all, configurations.
3.2 **Local Best Practices**, which should be tuned according to specific requirements

3.1 Global Best Practices for Pool Configuration

To achieve the most <u>*consistent*</u> and <u>*predictable*</u> performance –

- The more physical spindles that sit behind a POOL, the better the performance of that Pool. This translates to Array Groups. Spindles Win Prizes.
- Use and balance Array Groups from different DKA/BED pairs or different external controllers where possible.
- Do not intermix internal and external LDEVs in the same Pool, even if in the same CLPR.
- Each Array Group should comprise a single large LDEV occupying the entire space of the Array Group*.
- A single Array Group should never contain both normal LDEVs and Pool-VOLs.
- When adding additional space to a Pool, it is recommended that you add additional space in the same quantity and configuration as the original allocation (same number of underlying AG's, RAID type, spindle speed and capacity….). This will keep striding and performance consistent.
- The first time that a Pool is created, and then each time a full new set of LDEVs are added, the **Optimize** button should be used. This ensures striping over all of the Array Groups, including the newly added space.
- Assign DP-VOLs to the same CLPR as the Array Groups that are used to build the associated Pool.

3.2 Local Best Practices for Pool Configuration

This section will attempt to explain some of the theory behind **why** you might configure a Pool in a certain way. Three possible Pool configurations are presented and addressed –

3.2.1 Homogeneous Pools
3.2.2 Heterogeneous Pools
3.2.3 Multiple Pools

3.2.1 Homogeneous Pools

Homogeneous Pools are defined (by the author) as Pools in which all Array Groups/Pool-VOLs comprising the Pool are configured identically. E.g. –

- Common RAID level
- Common spindle type – FC/SAS/SATA/SSD....
- Common spindle capacity
- Common spindle performance characteristics (RPM)

Advantages - Homogeneous Pools allow for the most predictable and consistent performance. All spindles in the Pool will have the same performance characteristics and overhead. No single disk is guaranteed to become a bottle-neck or slow-spot due its inherent characteristics.

Disadvantages - ?

3.2.2 Heterogeneous Pools

Heterogeneous Pools are defined (by the author) as Pools which contain Array Groups/LDEVs of differing configurations. E.g. –

- Differing RAID levels
- Differing spindle types – FC/SAS/SATA/SSD
- Differing spindle capacities
- Differing spindle rotational speeds (rpm)

Heterogeneous Pools are allowed but **not** recommended other than the following scenario –

- Larger capacity drives may be added to an existing Pool when the original drive size is no longer available. However, it is recommended to keep to the same RAID level and rotation speed (RPM). For example, a Pool created with 146GB 15K FC spindles may be grown using 300GB 15K FC spindles when the 146GB disks are no longer available for purchase. It should be noted, however, that the larger capacity disks will have more Pool Pages and as a result have a higher access density making them more likely to become hot-spots/slow-spots.

It is not recommended to intermix RAID levels, interface and protocol type, RPM or technology (Flash SSD and rotational magnetic media). Performance will be established at the lower performing resource. There is little to be gained by trying to enhance a Pools performance by adding a higher performing Drives. Instead, adding Drives at the same RAID, and RPM is the best approach.

3.2.3 Multiple Pools

A single large HDP Pool, rather than multiple smaller Pools, may appear to offer the best performance. This idea is usually based on the principle that spindles win prizes, or in other words, the more spindles behind a Pool or LUN, the wider the striping and therefore the better the performance. However, because of the way that Pool pages are assigned and the current Page size of 42MB, a noticeable performance increase may not be seen.

There is also a concern from many customers that the larger a Pool the greater the potential impact of a failed Array Group. Almost everyone I speak with voices this concern at some point.

Multiple Pool offer the following potential benefits –

- Tiering. Best Practice is to use Homogeneous Pools. Multiple Homogeneous Pools of differing configurations (RAID, RPM, drive capacity....) provides for tiering. E.g.
    - Pool 1 (Tier 2) = 146GB 15K FC RAID1 (4+4)
    - Pool 2 (Tier 3) = 750GB 7.2K SATA II RAID6 (6+2)
- Workload isolation. If certain applications have heavily conflicting workloads or require maximum IOPs from the Pool at the same time, they should be placed on separate Pools.
- Lowering the impact of a failed Array Group. A failed Array Group will only affect a single Pool**

It is common practice for most customers to have multiple Pools on the same USP-V.

Other notes on multiple Pools

A single Pool can usually handle random and sequential workloads, such as database file and log file activities, without a noticeable performance impact. It is not a de facto requirement to have separate Pools to isolate such workloads. However, a single DP-VOL should not be assigned both random and sequential workloads.

It may, however, be beneficial to have separate Pools for heavily conflicting workloads. An example being - putting an application which saturates disk overnight, while staging backups to disk etc, on the same Pool as an application that runs critical overnight batch jobs. Or to separate Logs from data files for application recoverability purposes. You may also want to place ShadowImage P-VOLs and S-VOLs in separate Pools.

If the user is overly concerned about the failure of an Array Group within a Pool, they should consider RAID6.

* Creating 1 large LDEV (Pool-VOL) occupying the entire space of an Array Group is the recommendation if you want the largest possible Pool. However, if the capacity of the Array Group is larger than the maximum size of a single LDEV, multiple equally sized large LDEVs should be created to occupy the entire space of the Array Group. There no longer a recommendation to create multiple LDEVs per Array Group to assist *Striding*.

** The probability of an Array Group, participating in HDP Pool operations, failing is exactly the same as an Array Group not participating in HDP Pool operations. However, the impact can be significantly worse due to the fact that potentially all DP-VOLs in a Pool will be affected by the failure.

4. DP-VOL size recommendations

- As of ucode V03+1 the ability to dynamically expand **the last DP-VOL in a V-VOL Group** is available for Windows 2008. With ucode v04a this feature is extended to AIX and Red Hat Linux. This functionality will likely be extended to more Operating Systems in the future.*

- If creating a very large number of small DP-VOLs it may be worth sizing them in multiples of 42MB in alignment with HDP Page size.

- It is often seen by customers as a good idea to stick existing "standard" LUN sizes already used. This helps keep things simple with the business and rest of the IT department as they are often used to certain common LUN sizes.
  - This will also assist if you ever need to migrate off a Pool to standard sized LDEVs
  - This may help with some array based migration options.

* To be able to dynamically expand a DP-VOL, it must be the last DP-VOL in a V-VOL Group. This may change with future versions of ucode. For detailed explanation of this, and the theory behind it, see the authors following article – http://blogs.rupturedmonkey.com/?p=190

V-VOL groups will be 4TB in size in future releases of ucode.

5. Other Considerations

Monitoring

- Currently the only practical level of alerting is at the Pool level. DP-VOL alerting is not considered useful at the time of writing this document. Future releases of ucode and possibly HTnM version 6.0 may add additional monitoring options.

The following items should be considered if there is a requirement to utilise the oversubscription feature of HDP -

- NTFS partitions should be formatted with the Windows **Quick Format** option. A Windows **Full Format** on Windows Server 2008 beta version has been seen to consume all possible pages for a DP-VOL.
- Filesystem defrags should be avoided wherever possible (***online*** application defrags, as seen in Exchange, SQL Server, Active Directory and Oracle are fine as they do not walk all over the filesystem).
- Veritas Volume Manager is the recommended tool to be used when migrating old systems from traditional LDEVs to DP-VOLs as this is friendly to the oversubscription feature of HDP.
- It is recommended to use appropriate application tools to shrink disk space usage as much as possible before using tools to migrate old systems to DP-VOLs
- After migrating a standard (thick) LDEV to a DP-VOL it is good practice to perform a Zero Data Discard operation (Zero Page Reclaim). This operation requires microcode 60-04-04 and the associated upgrade to Storage Navigator.
- Other USP-V program products are “HDP aware”. As such, you can make additional savings with the HDP-HDP copy and replication operation where only the allocation blocks of a DP-VOL are copied.
    - An example being distance replication products where the P-VOL and S-VOL are both DP-VOLs. In this scenario the USP-V will only replicate the allocated blocks, saving on overhead and bandwidth and bring down Initial Copy operation times etc.

The Windows NTFS file system is not “Thin” friendly. This is because the NTFS file system uses a form of Least Recently Used (LRU) algorithm when choosing clusters/blocks to allocate to writes. Basically, even after deleting data from an NTFS file system, subsequent writes will be written to previously unallocated filesystem blocks/clusters in preference to the space released by delete operation. This behaviour assists the Microsoft Undelete functionality of NTFS and is at the time of writing not configurable.

6. Theory

6.1 Pool and Free Page List Theory

The following is a brief overview of HDP theory based on the authors own personal and limited understanding. Feedback and clarifications are very welcome. It is included to help those implementing HDP better understand the technology and aid them in the decisions they make.

An HDP **Pool** is created from multiple LDEVs referred to as **Pool-VOLs**. Pools should be created according to Best Practices outlined earlier in this document.

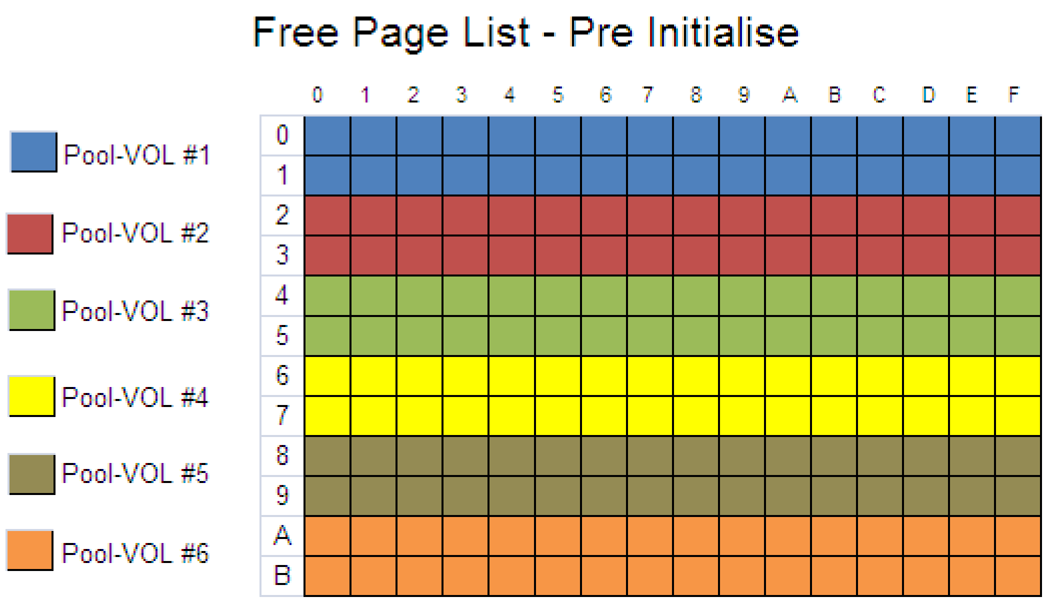
Once a Pool is created, its space is divided into 42MB extents referred to as **pages**. This **page** is the basic unit of allocation, and is fixed at 42MB. See http://blogs.rupturedmonkey.com/?p=182

When an HDP Pool is first created, all of its pages are **free**, as they are not allocated (loaned) to any DP-VOLs.

DP-VOLs are virtual volumes created and **associated** with an HDP Pool but mapped into the CU:LDEV address space and then to front end ports just like normal LDEVs. When a DP-VOL is initially created it is allocated (loaned) no pages. Therefore it consumes no capacity.

Pages are allocated to DP-VOLs on demand.

The **Free Page List** is an HDP metadata related table that is referenced when choosing which page to allocate to a DP-VOL. When an HDP Pool is first created, all Pool-VOLs used to create the Pool have their pages grouped together in a concatenation type configuration as illustrated below –

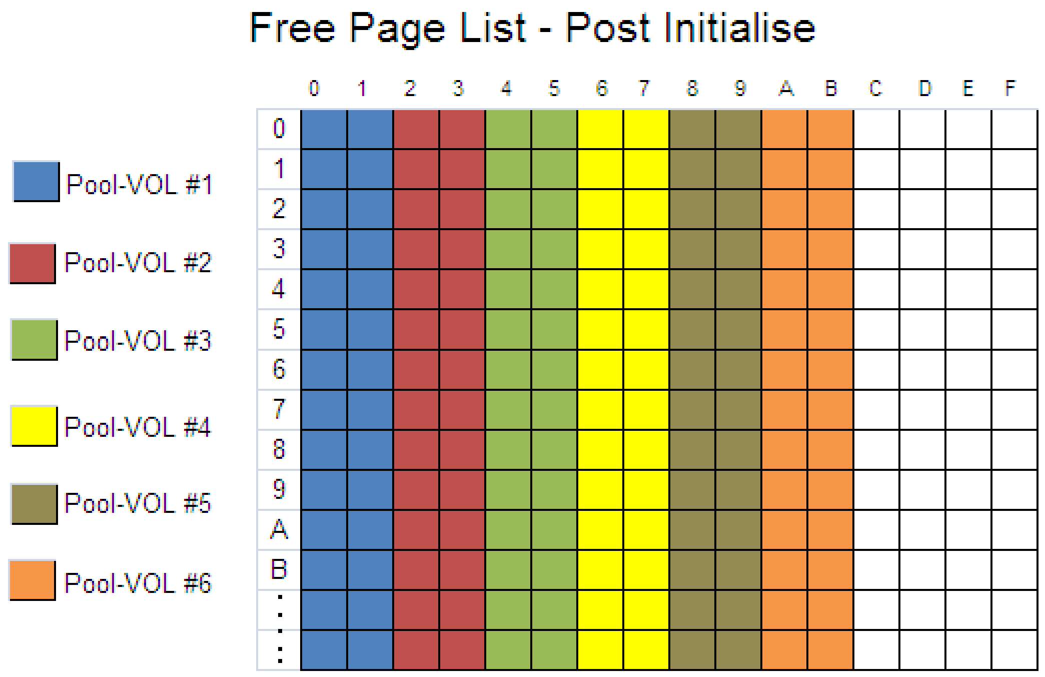


Pages are allocated from the Free Page list in series on a first-come-first-served basis starting at location 0:0 and then moving on to 0:1, 0:2, 0:3........ 1:0, 1:1, 1:2...... To the best of the authors knowledge there is no additional logic employed, when selecting pages, to attempt to evenly distribute the spread of pages for each DP-VOL over the entire Pool.

With the Free Page List in its pre-initialised status, all pages Pool-VOL #1 will be allocated to DP-VOLs prior to any pages on Pool-VOL #2 being allocated etc.

Prior to allocating pages from a Pool (making the Pool production ready) it is highly recommended to **Initialise** the Pool by clicking the **Initialize** button.

Performing the Initialise operation re-arranges the pages in the Free Page List in more of a striping formation –



After the Free Page List is initialised, pages will be allocated more evenly over the Pool-VOLs in the Pool, similar to a stripe. Assuming the example above, two consecutive pages from Pool-VOL #1 will be allocated before switching to Pool-VOL #2........

NOTE: *The Free Page List is an HDP construct that is referenced in HDS documentation. However, the author is not familiar with its actual structure. The diagrams above are for illustrative purposes only and should not be taken as actual or authoritative. Knowledge of the internal structure of the Free Page List may assist in Pool construction decisions, but the author is not in possession of such information.*

6.2 Zero Page Reclaim Theory

As of ucode 60-04-04 and with the associated upgraded version of Storage Navigator, the USP-V supports Zero Page Reclaim operations on a per DP-VOL basis. The Storage Navigator GUI refers to this as a Discard Zero Data operation.

When you select to perform a Zero Page Reclaim operation on a DP-VOL, the USP-V will search the Pages allocated (loaned) to that DP-VOL, looking for Pages that have no data. Or to be more correct, it searches for pages that contain only zeros.

Sidestep: Most storage arrays will zero out a volume when it is RAID formatted, basically writing zeros to all blocks comprising the volume. This helps the XOR parity calculation.

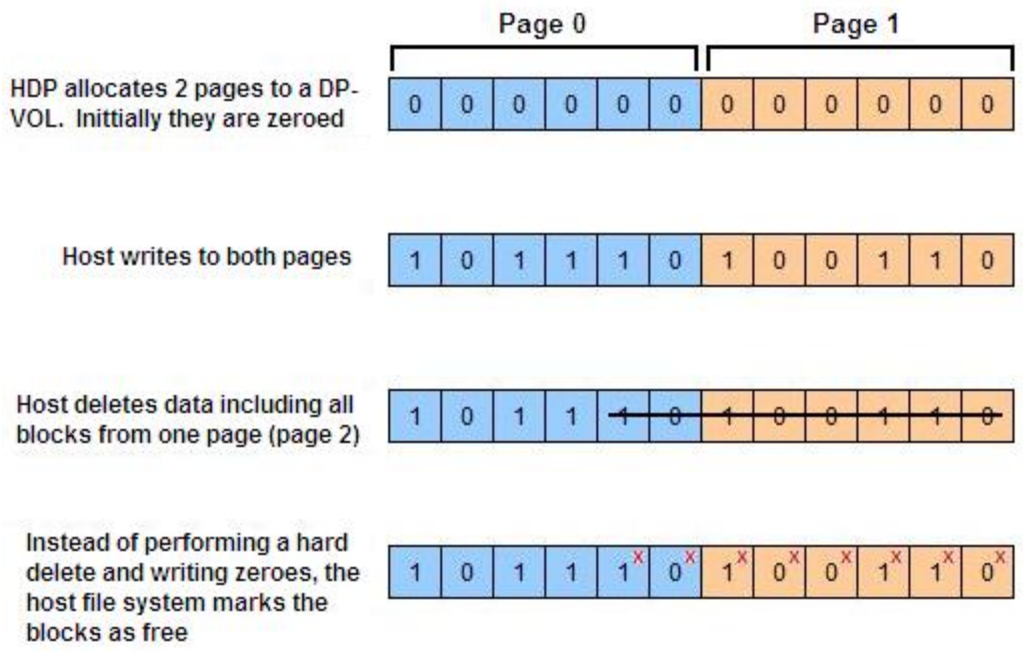
So any page that the array finds comprised entirely of zeros, it assumes is unused and breaks the association between the page and the volume. This has the effect of placing such pages back into the Free Pool. This has obvious capacity saving benefits and is a useful tool for any Thin Provisioning array to have up its sleeve.

How useful this functionality is relies on a few factors. Two of which include; page size, and file system behaviour

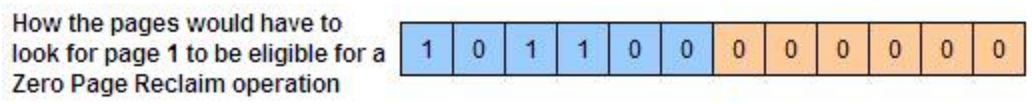
Page size. It would seem, at first glance, that the smaller the page size, the more opportunity for reclaiming space. The theory behind this being that there is more chance of finding pages comprised entirely of zeros if your page is relatively small. Conversely you are less likely to find multiple megabytes worth of consecutive zeros, as would be required if you have a larger extent size. However, this may have little effect in practice, especially after migrations from thick to thin.

File System Behaviour. One point in relation to file systems is that many file systems do not zero out space when a file is deleted. Instead they take the lazy approach and simply mark the blocks as free but leave the original data in place.

While this soft delete behaviour may prove useful should you need to recover a deleted file, it is not friendly to Zero Page Reclaim operations. Put in other words, freeing up 150GB of space on a volume does not normally (depends on your file system behaviour….) write zeros and make the space eligible for Zero Page reclaim. See diagram below -


Page 0
Page 1
HDP allocates 2 pages to a DP-VOL. Inittially they are zeroed
0 0 0 0 0 0 0 0 0 0 0 0
Host writes to both pages
1 0 1 1 1 0 1 0 0 1 1 0
Host deletes data including all blocks from one page (page 2)
1 0 1 1 1 0 1 0 0 1 1 0
Instead of performing a hard delete and writing zeroes, the host file system marks the blocks as free
1 0 1 1 1 0 1 0 0 1 1 0

The above diagram is an oversimplification but illustrates the point. For Extent 1 to be a candidate for Zero Page Reclaim, the file system would have to hard delete (zero out) all deleted data as shown in the diagram below -


How the pages would have to look for page 1 to be eligible for a Zero Page Reclaim operation
1 0 1 1 0 0 0 0 0 0 0 0

With present day file systems, the best use case for Zero Page Reclaim may be after migrating volumes from classical thick volumes to new thin dynamic volumes. For example, this will work well with a 500GB traditional thick volume where only 200GB has been written to. When migrating this to a thin volume you will more than likely be able to reclaim the untouched 300GB at the end of the volume to the Free Pool.

7. Further Information

- In the event of a PS-OFF, the DMT, stored in Shared Memory is de-staged to the SVP hard disks (assuming Mode 460 is set). This is in addition to the real-time recording of DMT changes to the reserved area in the first Pool-VOL of each Pool.

- After performing a Windows **Quick Format** (default cluster size, NTFS) on a 5GB HDP volume, it registered as consuming 6% of its 5GB space. After the same Windows **Quick Format** operation on a 25GB HDP volume, the Storage Navigator GUI reported as consuming 2% of the available space.

- Read operations directed to an area of a DP-VOL that is not already allocated space do not cause space to be consumed.

- On Windows Server 2003 a full format consumes the same amount of space as a quick format. However, Windows 2008 has been seen to consume all pages of a DP-VOL when a full format is performed. This was on a pre-release version of Windows 2008!

For more information regarding the 42MB Page size see -
http://blogs.rupturedmonkey.com/?p=182

For more general information from the author regarding HDP see –
http://blogs.rupturedmonkey.com/?cat=18

For more general information re storage, and in particular HDS and HP storage, see -
http://blogs.rupturedmonkey.com

Links to HDS webtech sessions will be added as they become available.

To contact the author directly - nigel@rupturedmonkey.com

Feedback welcome!

Thanks to everyone reading and using the document and especially those providing feedback and insights.