3PAR Thin Conversion: VMware ESX

Migrating from Legacy Storage to 3PAR Utility Storage

Document Abstract:

3PAR® Thin Provisioning software reduces storage costs by using copy-on-write technology that allocates physical disk blocks at the point they are actually written, thus eliminating the need for pre-planning, manual provisioning, and large up-front capacity purchases typical of legacy storage. The 3PAR Gen3 ASIC with Thin Built In™ works with 3PAR Thin Provisioning and data stored on legacy storage to enable the conversion of "fat" file systems to thinner, more efficient volumes on the 3PAR InServ® Storage Server. This document shows how a VMware ESX administrator can prepare for and perform this conversion.

Part Number: vmf2t-wp-09.2

Table of Contents

1 Introduction

3PAR® Thin Provisioning software is a green technology that dramatically cuts capacity, energy, and related costs while substantially alleviating storage and system administration overhead. By allowing the safe overallocation of physical capacity, 3PAR Thin Provisioning lets organizations provision virtual capacity to applications once and then purchase physical capacity only as applications truly require it for written data. Since its introduction, 3PAR Thin Provisioning has continued to set the industry standard against which all other “thin” approaches are measured.

With 3PAR Thin Provisioning, capacity is dedicated and configured autonomically and in small increments from a single, reservationless, comprehensively scalable reservoir, so organizations can provision storage efficiently and without waste. However, starting “thin” has not been an option with many legacy storage vendors. So what about the petabytes of storage that currently reside on expensive and underutilized legacy storage? This technical paper discusses how organizations can address this problem of inefficient and underutilized storage in VMware environments by migrating from “fat” legacy storage to “thin” 3PAR Utility Storage.

2 3PAR Thin Technologies: Start Thin, Get Thin, Stay Thin

In an ideal world, all of your storage would start thin. But for organizations that have been using legacy solutions instead of 3PAR Thin Provisioning, starting thin may not have been an option. 3PAR Utility Storage allows organizations to not only start thin, but also to get thin and to stay thin.

Working in conjunction with 3PAR Thin Provisioning, 3PAR InServ® Storage Servers with Thin Built In™ feature built-in zero detection[1] which enables organizations to intelligently leverage data migration[2] from legacy arrays onto more efficient thin volumes on the InServ while preserving service levels and without disruption or performance impact. 3PAR is the first in the industry to offer storage systems with this zero detection capability designed into the hardware architecture of its arrays to enable wire-speed fat-to-thin conversions. The revolutionary 3PAR Gen3 ASIC is the only processor that provides an efficient, silicon-based mechanism for converting traditional “fat” volumes from other platforms to more efficient “thin” volumes on 3PAR arrays with on-chip, on-the-fly detection of all-zero blocks. With 3PAR, getting thin is now just as simple as starting thin.

[1] The zero-detection features require a 3PAR T- or F-Class InServ Storage Server and InForm Operating System 2.3.1.
[2] 3PAR Thin Conversion is migration-method agnostic. Data migration may be performed using tools provided with the host operating system or with third-party migration tools.

3 3PAR Thin Conversion Process for VMware ESX

3PAR Thin Conversion is a three-step process for migrating data, using OS-native or third-party migration tools, from a fully-provisioned volume (or LUN) on a legacy storage array to a Thin Provisioned Virtual Volume (TPVV) on a 3PAR InServ Storage Server. The process starts with a planning step, which identifies and quantifies free or unused space in order to determine whether or not there is significant benefit to “thinning” the volume. A preparation step is next, during which the volume is cleaned up and unused space overwritten with zeroes. The final step is the migration itself, during which the InServ array detects blocks of zeroes and does not allocate physical storage for them. Each of these steps is briefly described below. A detailed explanation of each step then follows.

Step 1: Assessment

During the first step of 3PAR Thin Conversion, the administrator quickly identifies whether or not the volume might be amenable to thinning based on file system free (or unused) space. In most cases, volumes which are good candidates for thinning can be identified in a minute or two.

There may be opportunities to create additional free space at this point by shrinking databases, emptying trashcans, deleting temporary files, archiving unused files, and performing other types of file system housekeeping. Specific instructions for these actions depend on the applications which use the file system and other details pertaining to how the file system is used. These instructions are beyond the scope of this document. File system cleanup can free up substantial amounts of storage in most file systems, leading to significant benefits from thinning even if the preliminary assessment indicates otherwise.

Step 2: Preparation

The preparation step consists of writing zeroes to the free or unused space in the file system. Writing zeroes does not free the space, but zeroed space detected by the 3PAR Gen3 ASIC during the migration step will not be allocated physical storage, so this step is necessary for thinning the volume.

Step 3: Migration

The final step of volume thinning is migration of the file system, which is the act of copying (or migrating) the volume containing the “fat” file system with zeroed free space to a TPVV on the InServ array.

The following subsections describe in detail the actions required to accomplish these three steps, using examples to illustrate each step.

3.1 Assessment

File systems contain user data, metadata which is used to manage the storage space, and free space. Free space includes storage blocks which have never been used plus blocks from deleted files. In most cases, when a file is deleted, its blocks are simply added to the pool of free space without zeroing or otherwise erasing them.

The core of the assessment step is to determine the number of physical blocks in the file system which are free or unused, i.e., not allocated to user or application data or file system metadata. The methods to make this determination depend upon the Virtual Disk (VMDK) format and guest OS being used.

Once this initial determination has been made, clean the file system to recover as much free space as possible. Actions to consider include shrinking databases[3], emptying trashcans, deleting temporary files, and archiving unused files. FIle system cleanup is a crucial step in order to maximize the space savings from the conversion and may lead to significant benefits from thinning even if the preliminary assessment indicates otherwise.

Finally, for Windows guests, consider running Disk Defragmenter. TPVV storage is allocated in 16-KB increments, so a highly fragmented NTFS file system (containing a mix of used and free space in many 16-KB blocks) can result in reduced effectiveness of Thin Conversion. Defragmenting the file system prior to zeroing free space may be performed to minimize this problem.

3.1.1 Virtual Disk (VMDK) Formats

VMware ESX provides storage volumes for virtual machines called Virtual Disks. Each virtual disk is stored in a VMDK file—a file within the VMFS file system. Each VMDK file appears as a disk LUN to the VMware guest operating system.

[3] Consult the documentation for your database software for specific instructions on how to shrink the database.

VMware offers four different formats for VMDK files:

- **Eager Zeroed Thick** – All VMDK blocks are allocated and zeroed when the VMDK is created.

- **Zeroed Thick** – All VMDK blocks are allocated when the VMDK is created but each block is only zeroed upon first reference.

- **Thick** – All VMDK blocks are allocated when the VMDK is created but are never zeroed by ESX. While more efficient, this creates a significant security exposure by allowing the guest OS to access data which was written to the storage by a former owner of the space.

- **Thin** – VMDK blocks are not allocated until written. Unallocated blocks are read as zeroes. (This behavior is similar to a 3PAR TPVV).

This document describes the conversion process for all four VMDK formats.

The 3PAR Thin Conversion process also varies depending on the guest operating system. This document examines the process using Red Hat Enterprise Linux (RHEL) 5.2 guests and Windows® Enterprise Server 2003 R2 guests.

3.1.2 Quantifying Free Space

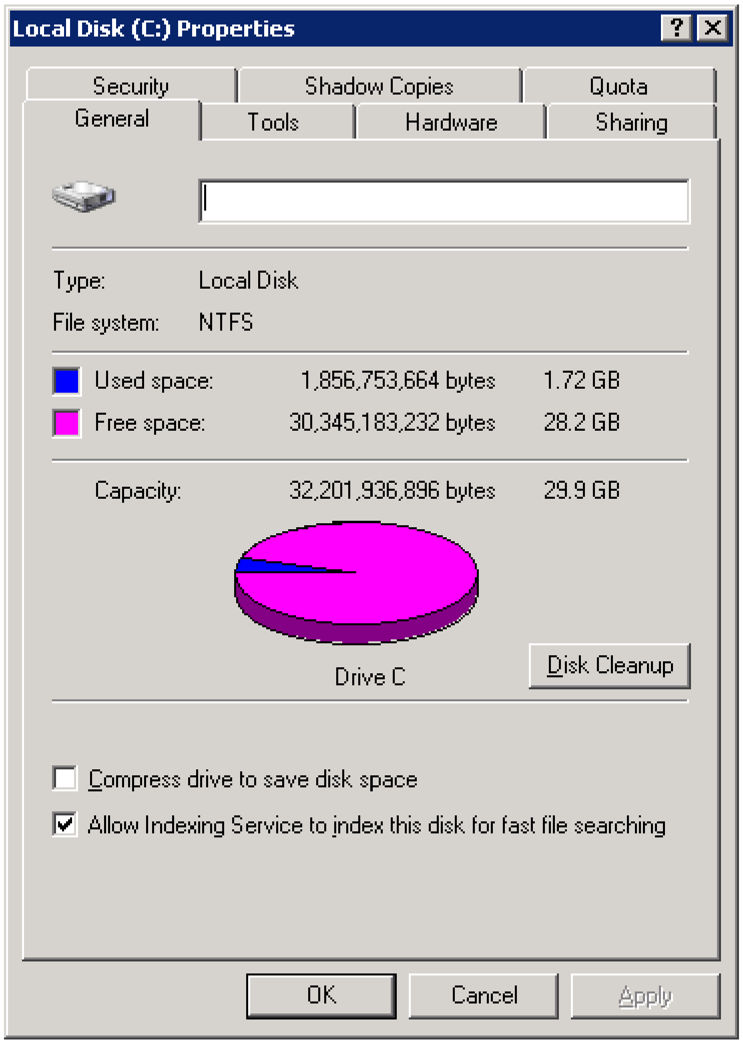
Free space is an attribute of the guest operating system's file system and thus must be determined within the guest OS rather than at the ESX level.

The `df` command will identify the amount of free space in the file system for Linux guests:

```
[root@rhel5 /] # df –h /
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                       27G    3G   23G  12% /
```

In this example, the root file system is 27 GB in size but only 3 GB (12%) is in use; 23 GB is available or free.

For Windows guests, the amount of free space may be determined by examining the properties of the file system as illustrated in Figure 1


Local Disk (C:) Properties

Type: Local Disk
File system: NTFS

🟦 Used space: 1,856,753,664 bytes 1.72 GB
🟪 Free space: 30,345,183,232 bytes 28.2 GB

Capacity: 32,201,936,896 bytes 29.9 GB

☐ Compress drive to save disk space
☑ Allow Indexing Service to index this disk for fast file searching

Figure 1. Properties for Drive C Showing Free Space.

The amount of free space may also be determined using the Windows CHKDSK utility:

```
C:\>chkdsk
The type of the file system is NTFS.

WARNING!  F parameter not specified.
Running CHKDSK in read-only mode.

CHKDSK is verifying files (stage 1 of 3)...
File verification completed.
CHKDSK is verifying indexes (stage 2 of 3)...
Index verification completed.
CHKDSK is verifying security descriptors (stage 3 of 3)...
Security descriptor verification completed.

  31447205 KB total disk space.
   2039428 KB in 10039 files.
      2560 KB in 859 indexes.
         0 KB in bad sectors.
```

```
       77925 KB in use by the system.
       65536 KB occupied by the log file.
    29327292 KB available on disk.

        4096 bytes in each allocation unit.
     7861801 total allocation units on disk.
     7331828 allocation units available on disk.
```

In this example, `29327292 KB available on disk` indicates the amount of free space: 28 GB of free space is available out of 30 GB of total disk space.

3.1.3 Determining the Benefit of Zeroing Free Space

The potential benefit of zeroing free space prior to migrating a file system depends on how much allocated physical space is not used by live data, i.e., the number of file system blocks on the file system's free list. Obviously, if there is relatively little unused space in the allocated physical space then there is little benefit to zeroing the free space to recapture this relatively small unused space. The reality is that file systems often have significant amounts of free space and the free space has been allocated to physical blocks. In this latter case, zeroing this significant amount of free space (to remove traces of deleted files) prior to file system migration will result in substantial storage savings.

3.1.3.1 Benefits of Zeroing Free Space: Thick VMDK Formats

For file systems inside thick VMDKs (including eager zeroed thick and zeroed thick formats), the amount of unused physical storage is equivalent to the amount of free space as determined using `df` (Linux) or disk properties or `CHKDSK` (Windows). Zeroing the free space will be beneficial unless there is little free space.

3.1.3.2 Benefits of Zeroing Free Space: Thin VMDK Format

Thin VMDKs require more careful analysis. Blocks which have never been used will be included in the guest file system's free space but will not have associated physical storage. Blocks which were used by files that have since been deleted (or truncated) will also be in the file system's free space, but in this case they will be associated with physical storage because they were previously written prior to the deletion. From the block-level view of the storage array, these "dirty-but-free" blocks are indistinguishable from used blocks.

If there is little free space in the thin VMDK, zeroing the free space will have little effect, similar to thick VMDKs. Assuming there is significant free space, one must examine how big the thin VMDK is according to ESX. For example, using a VM named `rhel5_vm4`, one would use the following commands:

```
[root@esxhost /] # vdf -h /vmfs/volumes/*/rhel5_vm4/
Filesystem        Size  Used Avail Use% Mounted on
/vmfs/volumes/49a3350b-3ab1799d-6f10-001f29c98e70/rhel5_vm4
                   99G   11G   88G  11% /vmfs/volumes/vmfsfat4

[root@esxhost /] # du -h /vmfs/volumes/*/rhel5_vm4/*-flat.vmdk
8.7G    /vmfs/volumes/vmfsfat4/rhel4_vm4/rhel5_vm4-flat.vmdk
```

The output of the du command in this example indicates that 8.7 GB of physical storage has actually been allocated for the file system.

The amount of space used should be compared to the allocated physical storage as reported by `df` (Linux) or disk properties or `CHKDSK` (Windows):

- If the amount of space used is close to the allocated physical storage size, most of the free space is unallocated and thus will be read as blocks of zeroes. Zeroing the free space in this case will be wasteful since it will require allocation of physical blocks which would have been read as zeroes anyway.

- If physical space usage reported by ESX is much greater than the used space reported by the guest operating system, there is likely to be a significant benefit from zeroing the free space.

There is no fixed threshold at which zeroing will be beneficial for all sites. For example, consider a 50-GB volume to which 30 GB of physical storage has been allocated but only 5 GB is in use by files and metadata. Zeroing the free space before migration will result in post-migration savings of 25 GB, but will require the short-term allocation of an additional 20 GB of physical storage during the preparation step of the migration process.

3.2 Preparation

Preparation consists of writing zeroes over remnants of old data in the free space. The specific steps to do this depend on the guest OS and are described in the following subsections.

3.2.1 Zeroing Free Space: Linux Guest OS

The dd command may be used to zero free space on Linux guests. To minimize any impact on other processes, it is suggested that zeroing using dd be done incrementally rather than in a single step. For example, to zero 23 GB of free space, the following commands should be used to zero one GB at a time:

```
[root@rhel5 /] # mkdir /zerofree
[root@rhel5 /] # for ((i=0;i<23;++i)) do
> echo Generating file /zerofree/$i ...
> nice dd if=/dev/zero of=/zerofree/$i bs=1k count=1m
> done
Generating file /zerofree/0 ...
1048576+0 records in
1048576+0 records out
Generating file /zerofree/1 ...
1048576+0 records in
1048576+0 records out
```

…

Once the zeroing process completes, verify that all of the free space has be written with zeroes:

```
[root@rhel5 /] # **df –h /**
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                       27G   27G     0 100% /
```

Finally, delete the files of zeroes:

```
[root@rhel5 /] # rm –rf /zerofree
```

3.2.2 Zeroing Free Space: Windows Guest OS

Free space on Windows guests is zeroed using the SDelete utility. A description of SDelete and a downloadable copy of the executable (.EXE file) may be found on Microsoft's web site at http://technet.microsoft.com/en-us/sysinternals/bb897443.aspx. Testing showed that SDelete had little performance impact on other system activities.

Once SDelete has been downloaded, using it to zero free space in NTFS is simple:

```
C:\>sdelete -c

SDelete - Secure Delete v1.51
Copyright (C) 1999-2005 Mark Russinovich
Sysinternals - www.sysinternals.com

SDelete is set for 1 pass.
Free space cleaned on C:\
```

3.3 Migration

Volumes are ready to be migrated once their free space has been zeroed. This paper illustrates the use of Storage VMotion to perform the volume migration since it is most likely to be used by VMware administrators. Other migration tools can be used but detailed steps for each of them are beyond the scope of this paper.

Before invoking Storage VMotion, it is necessary to enable zero detection on the InServ to which the VMDKs will be migrated:

```
t400 cli% setvv –pol zero_detect vmfsthin1
```

This command example enables zero detection for the Virtual Volume (VV) named `vmsthin1`. If multiple VMDKs will be moved to the same VV, this step need only be performed once.

Next, perform the migration as follows using Storage VMotion:

```
3(NXDOMAIN):~# svmotion --interactive
```

```
Entering interactive mode.  All other options and environment
variables will be ignored.

Enter the VirtualCenter service url you wish to connect to (e.g.
https://myvc.mycorp.com/sdk, or just myvc.mycorp.com):
https://dl380-04/sdk
Enter your username: Administrator
Enter your password:

Attempting to connect to https://dl380-04/sdk.
Connected to server.

Enter the name of the datacenter: Test
Enter the datastore path of the virtual machine (e.g.
[datastore1] myvm/myvm.vmx): [vmfsfat4] rhel5_vm4/rhel5_vm4.vmx
Enter the name of the destination datastore: vmfsthin4

You can also move disks independently of the virtual machine.  If
you want the disks to stay with the virtual machine, then skip
this step..
Would you like to individually place the disks (yes/no)? no

Performing Storage VMotion.
0% |-----------------------------------------------------------------
--------------------------------------| 100%

#################################################################
#########################
Storage VMotion completed successfully.

Disconnecting.
```

After the conversion, the showvv command can be used to verify the space savings:

```
t400 cli% showvv vmfsthin4
 Id       Name      Type CopyOf BsId Rd   State AdmMB SnapMB userMB
443  vmfsthin4 Base,tpvv    ---  443 RW started   256   3584  51200
-------------------------------------------------------------------
  1   total LD                                    256   3584      0
    total virtual                                   -      -  51200
```

In this example (also illustrated in Figure 2), a 50 GB VLUN is presented to the ESX server (51200 MB of user space), but only 3.5 GB of usable physical storage has been allocated (3585 MB of snap space, not counting parity or mirror space for RAID) plus 256 MB of admin space to store the TPVV-to-physical block mappings.

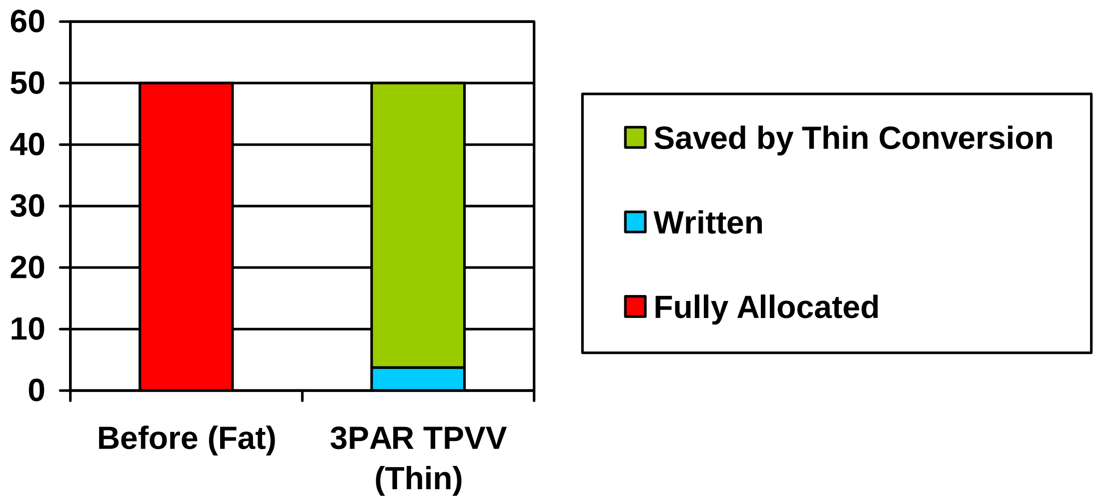


Figure 2: 50 GB LUN Before and After Thin Conversion.

4 Summary

As modern datacenters are constantly asked to do more with less—especially during times of tightening IT budgets—thin technologies are an ideal solution. 3PAR Thin Provisioning dramatically cuts capacity and related costs while substantially alleviating storage and system administration. With 3PAR Thin Provisioning and other 3PAR thin technologies, organizations can not only start thin, but can also get thin and stay thin. 3PAR InServ Storage Servers with Thin Built In™ and zero detection capability give organizations the ability to intelligently migrate data from traditional arrays onto more efficient thin volumes while preserving service levels and without disruption or performance impact. With 3PAR, getting thin has never been so simple.

About 3PAR

3PAR® (NYSE: PAR) is the leading global provider of utility storage, a category of highly virtualized and dynamically tiered storage arrays built for public and private cloud computing. Our virtualized storage platform was built from the ground up to be agile and efficient to address the limitations of traditional storage arrays for utility infrastructures. As a pioneer of thin provisioning and other storage virtualization technologies, we design our products to reduce power consumption to help companies meet their green computing initiatives, and to cut storage total cost of ownership. 3PAR customers have used our self-managing, efficient, and adaptable utility storage systems to reduce administration time and provisioning complexity, to improve server and storage utilization, and to scale and adapt flexibly in response to continuous growth and changing business needs. For more information, visit the 3PAR Website at: www.3PAR.com.