Technical Report

Understanding and Using vStorage APIs for Array Integration and NetApp Storage

Peter Learmonth, NetApp
November 2010 | TR-3886

TECHNICAL OVERVIEW

This document describes the technical aspects of NetApp support and integration with VMware® vStorage APIs for Array Integration (VAAI), as well as how to deploy and use this technology. VAAI is a set of application programming interfaces (APIs) and SCSI commands that offload certain I/O-intensive tasks to NetApp® storage systems. By integrating with vStorage APIs, NetApp enables advanced storage capabilities to be accessed and executed from familiar VMware interfaces, improving manageability, performance, data mobility, and data protection.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

The wide adoption of virtualization, particularly VMware vSphere™, by businesses worldwide demonstrates the importance of storage in these deployments. As virtualized data centers scale, heavier demands are placed on the shared storage systems that support these environments. Traditionally, vSphere and its predecessor products have treated advanced storage systems as essentially dumb disk. Much of the work that could have been offloaded to storage was done by theVMware ESX™ servers, the brute force way across the storage network, consuming server compute and storage network resources. VMware and NetApp recognized that there was a better way to perform some of these tasks, and they set about standardizing protocols to offload them to storage.

The result of this joint development effort is vStorage APIs for Array Integration (VAAI). These APIs allow ESX servers to offload virtual machine (VM) cloning and initialization work directly to supported storage systems, eliminating that load from the server compute and storage network resources. One of the VAAI operations also greatly simplifies the locking mechanism in VMFS datastores, allowing greater scalability within VMFS datastores.

2 ABOUT VAAI

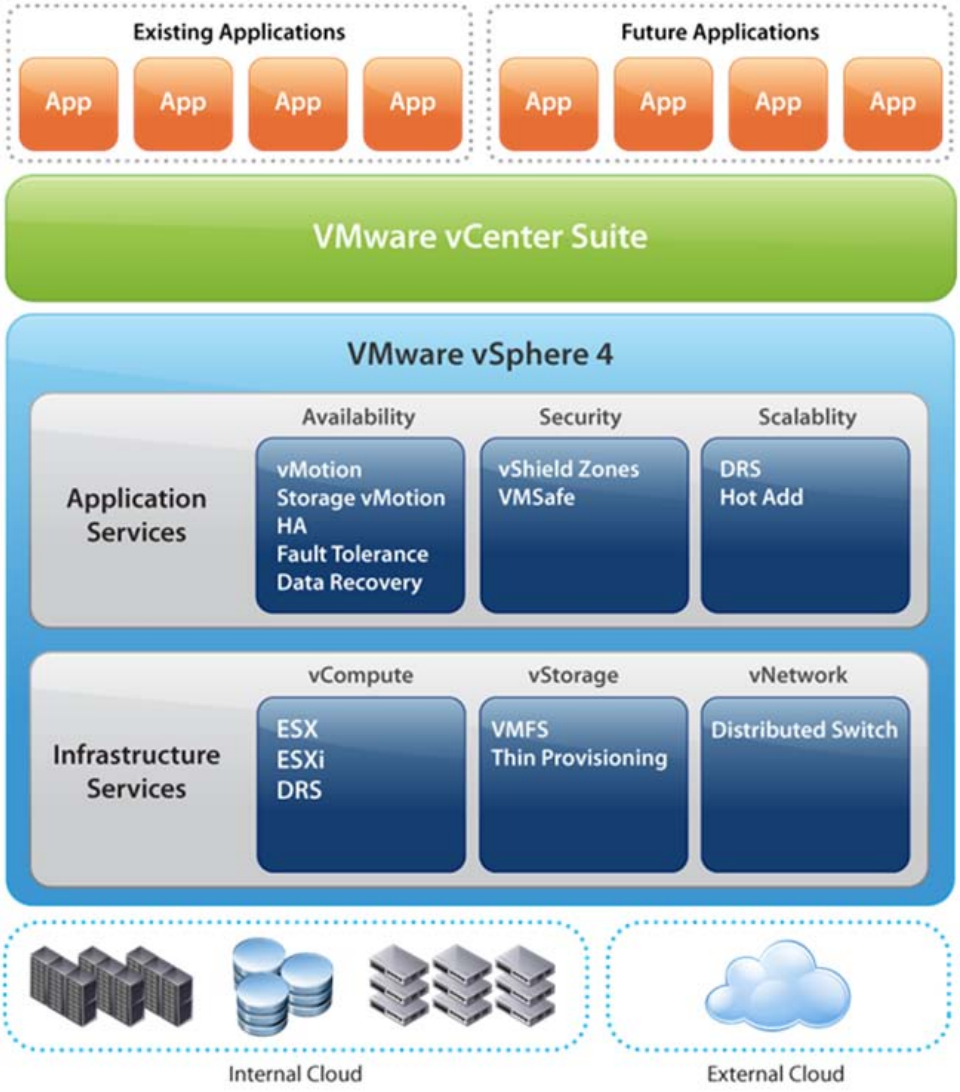
VAAI is a set of APIs and SCSI commands used to offload certain functions that are performed more efficiently on the storage array. In the past, these functions were performed "across the wire," unnecessarily consuming network and compute resources and slowing performance.

With the goals of reducing the consumption of network and compute resources and increasing efficiency, VAAI development was begun in 2007 as a joint effort involving VMware, NetApp, EMC, and EqualLogic. vStorage includes some technologies that predate the vSphere name, including VMFS, Network File System (NFS), multipathing, Site Recovery Manager, and VMware Consolidated Backup.

VMware vSphere is composed of six functional areas. As vSphere evolves, new features can be added to each of the functional areas. For example, vSphere 4.1 adds VAAI to vStorage. The six functional areas of VMware vSphere are:

- Availability
- Security
- Scalability
- Compute
- Storage
- Network

Figure 1) vSphere components (graphic supplied by VMware).



VAAI consists of three components that VMware refers to as *primitives*. A primitive is the underlying technology that a higher-level feature or use case can call. In turn, the primitive can perform a function or request that the function be performed on the storage device on behalf of the primitive. The three primitives in VAAI as of vSphere 4.1 are:

- Full Copy
- Block Zeroing
- Hardware-Assisted Locking

Full-motion animations of these operations, with and without VAAI to demonstrate the difference, are available as part of the NetApp VMworld 2010 demonstrations on YouTube.

2.1 FULL COPY

As defined by VMware, Full Copy "enables the storage arrays to make full copies of data within the array without having to have the ESX Server read and write the data."

USE CASE

There are two use cases for the Full Copy primitive.

- **vCenter™ VM cloning.** vCenter VM cloning makes a full copy of VMs, including any attached virtual disks. A virtual disk in a VMFS datastore is simply a large file. A VM can be cloned within the same datastore as the original or into another datastore simply by making a copy of the VM files.
- **Storage VMotion™.** Storage VMotion is a VMware technology that moves VMs, including any attached virtual disks, from one datastore to another while the VMs are running.

BENEFITS

There are two benefits of offloading the copy or move process to the storage. First, it reduces the use of compute and network resources for the copy operation; and second, the copy and move operations complete faster.

HOW IT WORKS

The underlying technology is an ANSI T10 standard SCSI Command Descriptor Block (CDB), or simply a command called EXTENDED COPY, sometimes abbreviated as XCOPY (not to be confused with the MS-DOS command of the same name). Rather than the ESX server reading every block of the virtual disk to be cloned or moved and then writing it back out to a new location, the ESX server sends a single SCSI command for a set of contiguous blocks, telling the storage to copy the blocks from one location, or logical block address (LBA), to another LBA. The command across the network is tiny, and all the actual work is performed on the storage device. Whether blocks are actually copied or some storage cloning technology is used depends on the storage vendor implementation.

Figure 2 and Figure 3 compare Full Copy cloning without and with VAAI.

Figure 2) Full Copy cloning without VAAI.

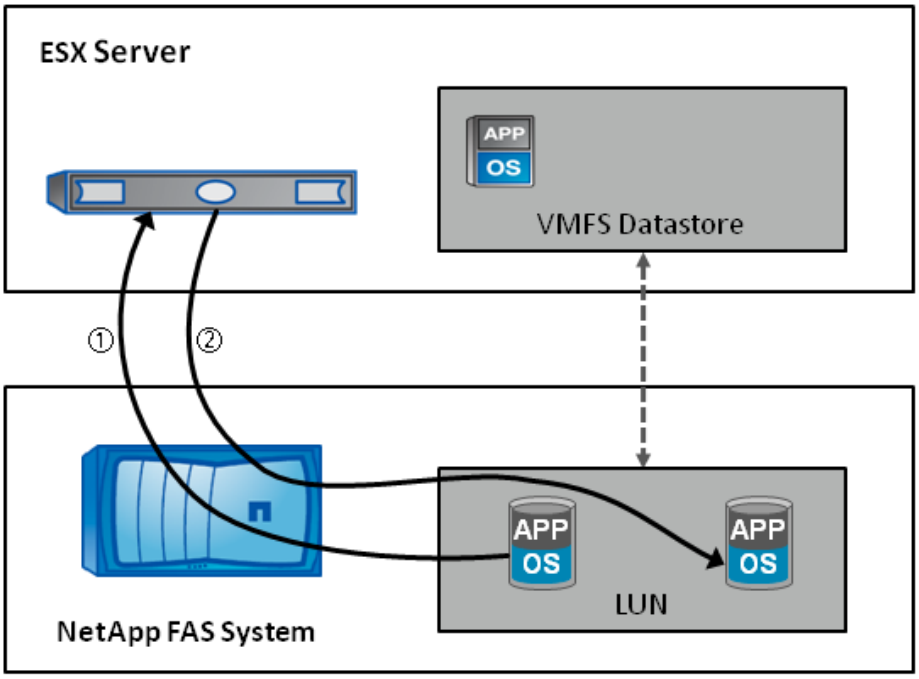
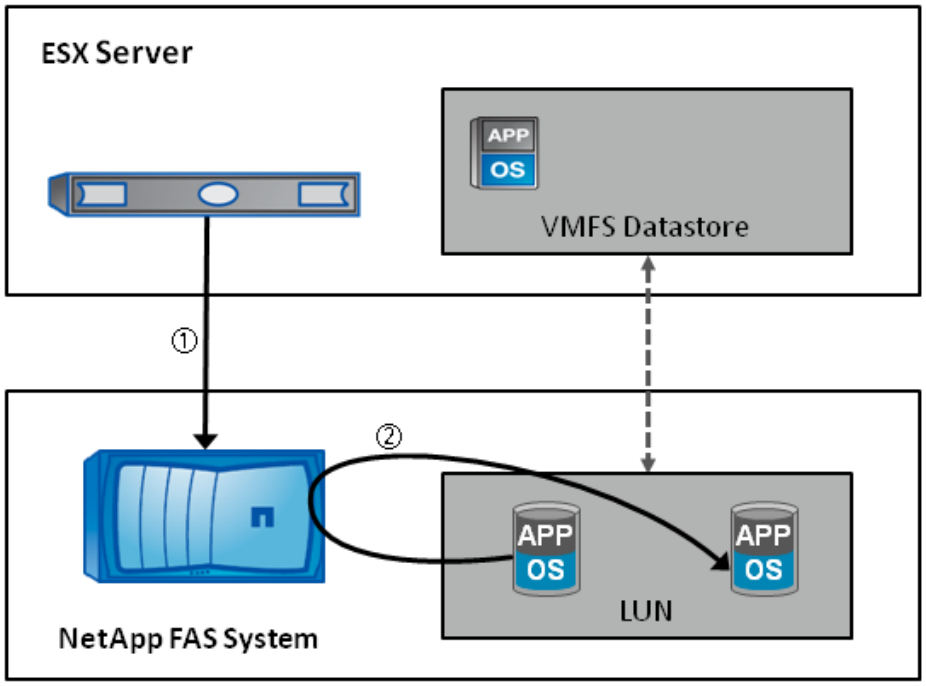


Figure 3) Full Copy cloning with VAAI.



2.2 BLOCK ZEROING

As defined by VMware, Block Zeroing "enables storage arrays to zero out a large number of blocks to speed up provisioning of VMs."

USE CASE

There are two situations in which ESX uses Block Zeroing:

- **Creating virtual disks that are zeroed upon creation.** The more commonly understood use case is creating virtual disks that are zeroed upon creation, which is referred to as *eager zeroed disks*. This virtual disk format is mainly used with VMs configured for VMware Fault Tolerance (FT). Upon creation, the virtual disk is zeroed out and cannot be used until the process completes. There are three reasons why FT virtual disks are eager zeroed:
  - To erase any data from previous VMs or other use of the logical unit number (LUN)
  - To make sure that the full size of the virtual disk is available and allocated, especially when thin provisioning is used
  - To avoid having to zero blocks before each first access, which can have a performance penalty
- **Creating virtual disks that are zeroed the first time they are accessed.** The less-known use of Block Zeroing is in conjunction with the most common virtual disk format, *zeroed-thick*. With zeroed-thick, sometimes referred to as *lazy zeroed* to distinguish it from eager zeroed, the virtual disk can be used immediately after it is created. Blocks still get zeroed, but this happens the first time they are accessed.

BENEFITS

The main benefit of Block Zeroing is that the work of writing zeroes is offloaded to the storage device, reducing redundant use of the compute and storage network resources. A second benefit is that the eager zeroed Virtual Machine Disk (VMDK) creation process may complete faster, allowing faster deployment of VMs, especially FT.

HOW IT WORKS

Block Zeroing uses a standard SCSI command called WRITE SAME. Rather than writing zeroes to each block explicitly, ESX uses the WRITE SAME command to instruct the storage device to write a pattern across a number of sequential blocks. Although the WRITE SAME command allows any pattern to be written, ESX uses the command only to write zeroes.

2.3 HARDWARE-ASSISTED LOCKING

VMware defines Hardware-Assisted Locking as providing “an alternative means to protect the metadata for VMFS cluster file systems and thereby improving the scalability of large ESX server farms sharing a datastore."

USE CASE

Fine-grained locking, the locking of small objects within a datastore, is used two ways in ESX:

- The ESX server must lock the object or region to be changed before changing any VMFS metadata.
- The process that runs VMs must be sure that no other process can write to the virtual disk, so no other ESX server can start a VM that uses those disks.

BENEFIT

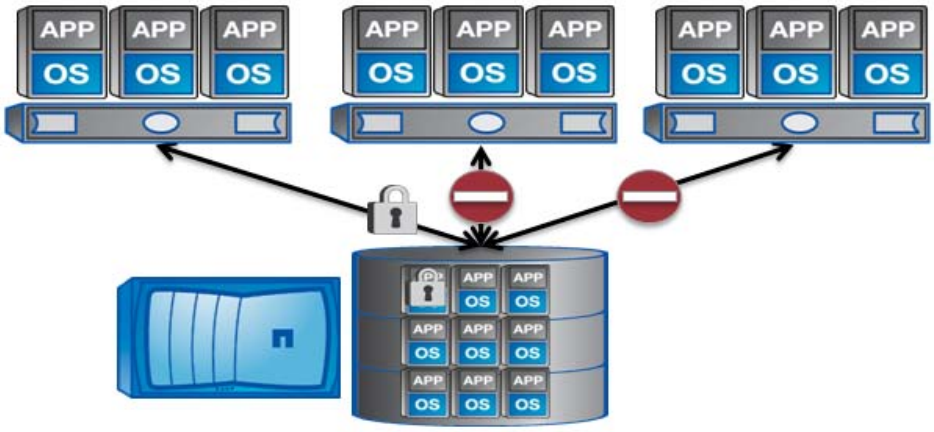
The benefit of Hardware-Assisted Locking is that it practically eliminates LUN-level locking based on SCSI reservations. This means that VMFS datastores can scale to more VMs per datastore and more servers can reliably attach to each LUN. This gives the administrator the flexibility to lay out datastores by business need rather than according to a limit imposed by technology.

THE PROBLEM THAT HARDWARE-ASSISTED LOCKING SOLVES AND HOW IT WORKS

The original implementation of the Virtual Machine File System (VMFS) uses SCSI reservations to guarantee reliable locking when many servers share a LUN. Normally, locking a smaller object such as a piece of metadata or a virtual disk file requires reading the lock and then, if the lock is free, writing to the lock to actually lock the object. With two operations, a read followed by a write, it is possible that another server may already have read the lock and that both servers will write their version of the lock. In that case, both servers think that they have the object. Before VAAI, when attempting to lock a smaller object, the server would lock the entire LUN by using a SCSI reservation, then read, and if free, write the lock. While the reservation is in place, no other server can access the LUN. Although these reservations are typically less than 1ms, many of them in rapid succession can cause a performance plateau with VMs on that datastore.

Figure 4 and Figure 5 compare VMFS locking without VAAI and Hardware-Assisted Locking with VAAI.

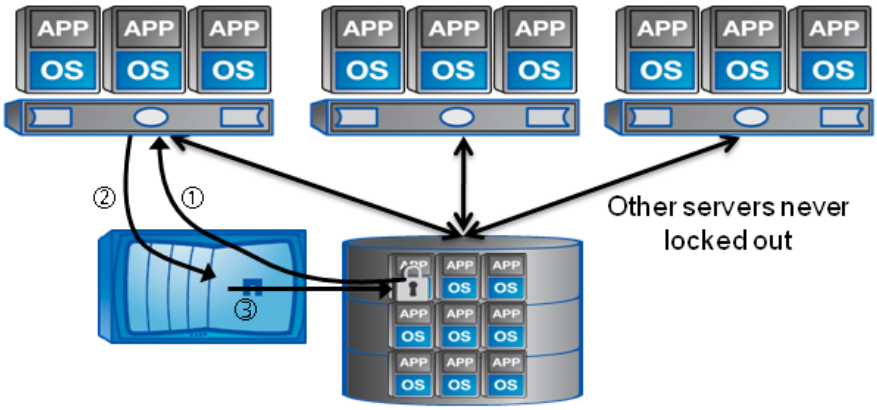
Figure 4) VMFS locking without VAAI.



Hardware-Assisted Locking in vSphere 4.1 uses yet another new SCSI command, COMPARE AND WRITE (CAW), sometimes referred to as *verify and write (VAW)*, especially in NetApp implementation in commands and command output. The ESX server makes a first read on the lock. If the lock is free, the server sends a CAW command that includes the original free contents of the lock, along with the lock data that the server wants to place into the lock. The storage device reads the lock again, compares the current data in the lock to what was in the CAW command, and if they match, writes the desired new data into the lock and returns success. From the SCSI perspective, the CAW read, compare, and write are treated as a single, atomic operation. This means that it is not possible for another server to intervene and write during the CAW operation. It is possible, however, for another server to write (or verify and write) the lock after the first read, which occurs before the CAW operation. If that happens, the CAW of the server does not match and returns a failure.

Figure 5 shows Hardware-Assisted Locking with VAAI employed.

Figure 5) Hardware-Assisted Locking with VAAI.

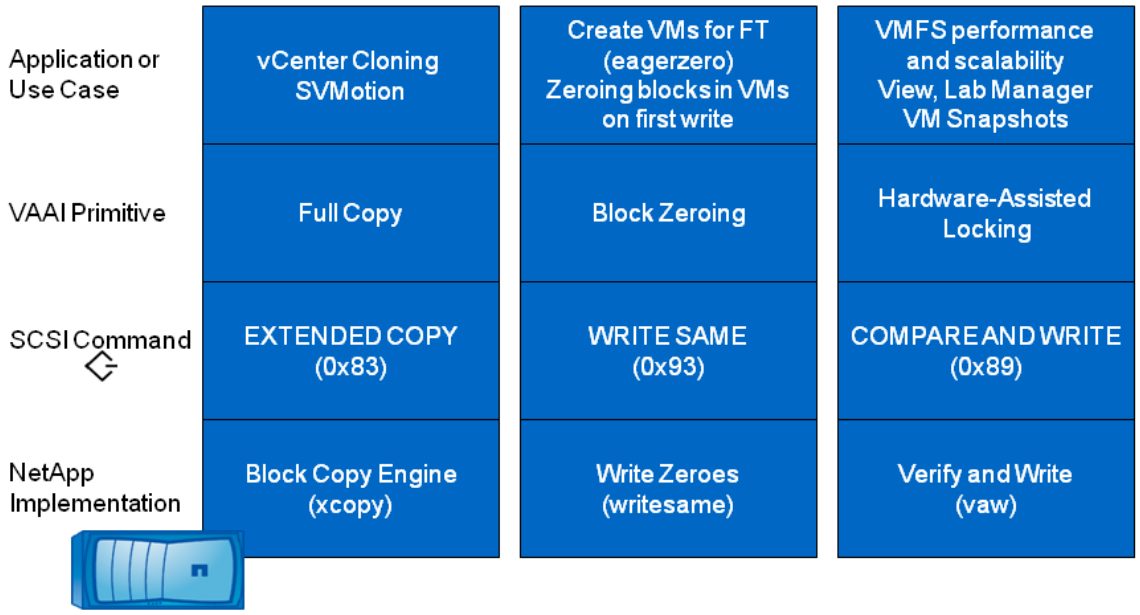


1. The server reads the block containing the lock for the file or other metadata.
2. If free, the server sends a CAW command containing the old and desired lock data.
3. The NetApp controller compares the old data to the data on disk, and if the same, writes the new lock data.
4. (Not shown) If successful, the NetApp controller returns a success status.

2.4 RELATIONSHIP OF USE CASE, PRIMITIVE, AND SCSI COMMAND

At various stages in the development of VAAI, and now in some documentation, the VAAI use case, primitive, SCSI command or CDB, and even some storage vendor terms have been used almost interchangeably. Figure 6 clarifies the terminilogy.

Figure 6) Relationship between use case, primitive, SCSI command, and NetApp implementation.



Note: You might also see the older term for VAW, which is Atomic Test and Set (ATS).

API VERSUS SCSI COMMAND

Strictly speaking, the communication between servers and storage isn't really a set of APIs. The APIs are used at a higher layer for the applications to call the primitive. The primitives invoke the SCSI commands and are sent from the server to the storage. The advantages to this approach are:

- SCSI commands are standardized across the industry. Although SCSI allows for some vendor-proprietary commands, options, and content, the SCSI commands used for VAAI are the same for all vendors. This greatly simplifies the integration work necessary for VMware and storage partners. If storage vendor APIs are used, a plug-in or driver architecture is necessary, with separate development for each vendor.
- With SCSI commands, the traffic for VAAI operations is in-band, meaning no additional network or protocol is necessary.
- No additional security, such as an API user and authentication mechanism, is needed beyond the standard zoning and LUN mapping or masking that is already in place.

2.5 VAAI AND NFS

As of vSphere 4.1, there are no VAAI features for NFS. As a file protocol, rather than a block protocol such as iSCSI, Fibre Channel Protocol (FCP), and Fibre Channel over Ethernet (FCoE), there are no SCSI commands between ESX servers and storage. SCSI commands in VM guests are converted to NFS operations.

However, with NetApp storage and management tools such as the Virtual Storage Console (VSC), there is already a corollary to each of the VAAI primitives for use with NFS.

Rapid cloning in VSC and the former Rapid Cloning Utility uses NetApp FlexClone® technology to make copies of virtual disks very rapidly as part of the rapid cloning of VMs. This completely eliminates block copying for most VM clones.

The main reason for Block Zeroing is to make sure that stale data from old VMs or other objects is not available to new VMs that are assigned to the old blocks. With the NetApp WAFL® file system, blocks are not allocated until data is written and any attempt to read a block that has not been written to yet returns zeroes. In other words, WAFL will never return stale data in reads from new virtual disks in NFS datastores. Further, NFS datastores on NetApp systems use the thin virtual disk format, so it is not even possible to create an eager-zeroed-thick virtual disk.

Finally, volume metadata for NFS datastores is managed centrally on the NFS server, rather than by each ESX server attached to the datastore. Virtual disk locks are implemented as small `.LCK` files rather than as some metadata in the file system. This means that an equivalent of SCSI reservations is not needed. In fact, there is no NFS equivalent to lock an entire file system.

There are, however, still use cases for some offload to storage, such as the use of Storage VMotion to move a subset of the VMs in a datastore to another datastore. Thus VAAI may add capabilities for NFS in some future release of vSphere and NetApp Data ONTAP®.

3 USING AND MANAGING VAAI

If specific configuration requirements are met, there is little more that an administrator must do to enable and use VAAI. The three primitives are on by default in ESX and ESXi 4.1, and VAAI is always enabled on supported NetApp storage systems.

3.1 VAAI REQUIREMENTS

The system and configuration requirements to use VAAI are:

- Servers must be connected to storage by using a block protocol such as iSCSI, FCP, or FCoE.
- Servers must be running ESX or ESXi 4.1.
- The NetApp storage system must be running Data ONTAP 8.0.1 or later.
- The VMFS datastores must be properly aligned on the LUN. VMFS datastores created by using vCenter 2.0 or later or NetApp VSC are correctly aligned by default. This applies to both cloning and eager-zero applications.
- When cloning between datastores, the two datastores must be accessed by using the same protocol. VAAI will not XCOPY between iSCSI and FCP on different LUNs on the same controller because it does not know that the two paths are to the same storage device.

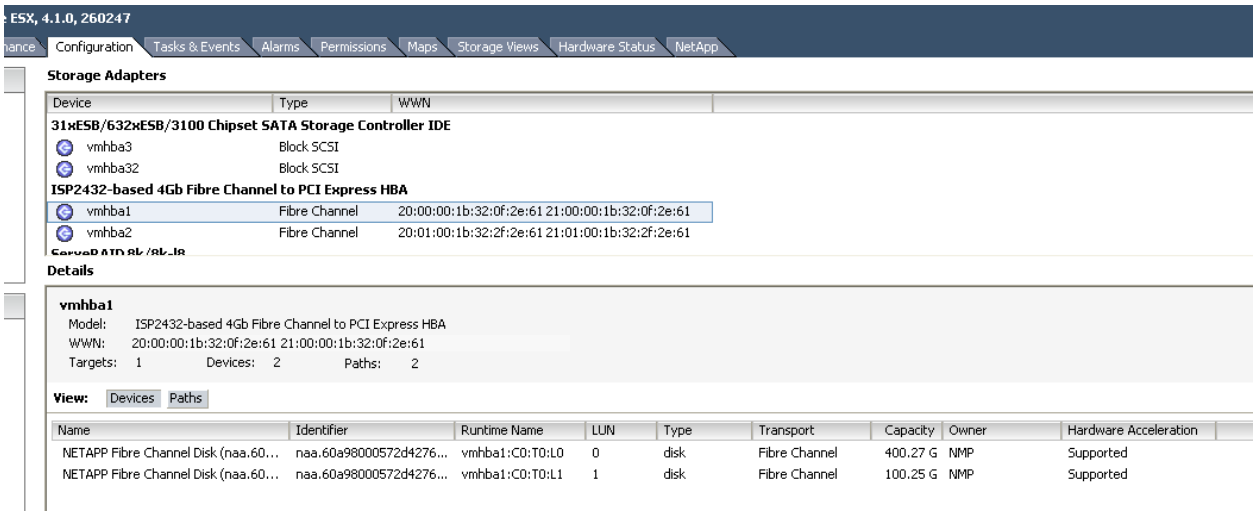
3.2 ENABLING AND DISABLING VAAI

There aren't many reasons to disable VAAI. One reason is to compare VAAI and non-VAAI performance and behavior in your environment with your workload.

VAAI IS ON BY DEFAULT

By default, all three VAAI primitives are on in both Data ONTAP and ESX. ESX uses VAAI automatically, if the storage device supports it. ESX determines whether each primitive is supported simply by trying it against each LUN as needed. After a primitive has been attempted, the status of VAAI goes from unknown to either supported or unsupported in the vSphere client storage view and in the output of `esxcfg-scsidevs -l`.

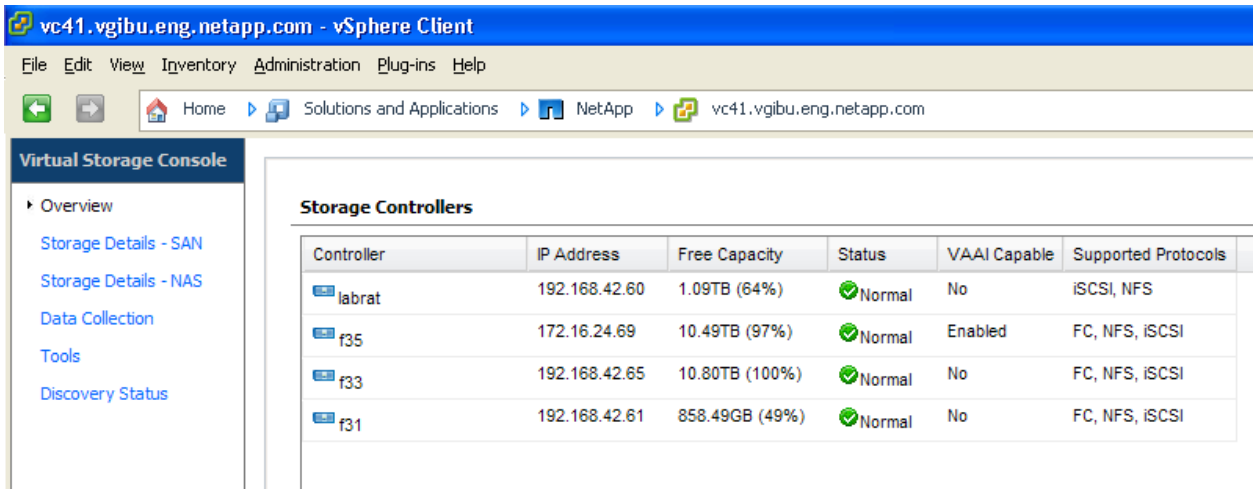
Figure 7) Viewing VAAI status by using the vSphere client.




```
[root@esx8 ~]# esxcfg-scsidevs -l | egrep "Display Name:|VAAI Status:"
  Display Name: Local ServeRA Disk (mpx.vmhba0:C0:T0:L0)
  VAAI Status: unknown
  Display Name: Local USB CD-ROM (mpx.vmhba32:C0:T0:L0)
  VAAI Status: unknown
  Display Name: Local MATSHITA CD-ROM (mpx.vmhba34:C0:T0:L0)
  VAAI Status: unknown
  Display Name: Local IBM-ESXS Enclosure Svc Dev (naa.5005076a0400434d)
  VAAI Status: unknown
  Display Name: NETAPP iSCSI Disk (naa.60a98000572d42766d4a5966456e794e)
  VAAI Status: supported
  Display Name: NETAPP iSCSI Disk (naa.60a98000572d42766d4a59664b51774a)
  VAAI Status: supported
  Display Name: NETAPP Fibre Channel Disk (naa.60a98000572d42766d4a5a4c5264334c)
  VAAI Status: supported
  Display Name: NETAPP iSCSI Disk (naa.60a98000572d42766d4a5a4e486d3979)
  VAAI Status: supported
```

The support status of VAAI is also displayed in the NetApp Virtual Storage Console.

Figure 8) Viewing VAAI status in the Virtual Storage Console.



VAAI CONFIGURATION IN ESX

Each primitive has a separate advanced configuration option in ESX that is set separately for each ESX server. The advanced configuration option can be set by using the CLI, the vSphere client, or the vSphere APIs from the Software Developer's Kit (SDK). Table 1 contains the CLI and vSphere names for the advanced configuration options.

Table 1) VAAI primitive setting names.

Primitive	CLI Name	vSphere Name
Full Copy	/DataMover/HardwareAcceleratedMove	DataMover.HardwareAcceleratedMove
Block Zeroing	/DataMover/HardwareAcceleratedInit	DataMover.HardwareAcceleratedInit
Hardware-Assisted Locking	/VMFS3/HardwareAcceleratedLocking	VMFS3.HardwareAcceleratedLocking

To view and set VAAI options from the CLI, use the command `esxcfg-advcfg`. This is useful for running tests of VAAI versus non-VAAI performance and behavior, especially in a script. To view a setting, use the `-g` or `get` option.

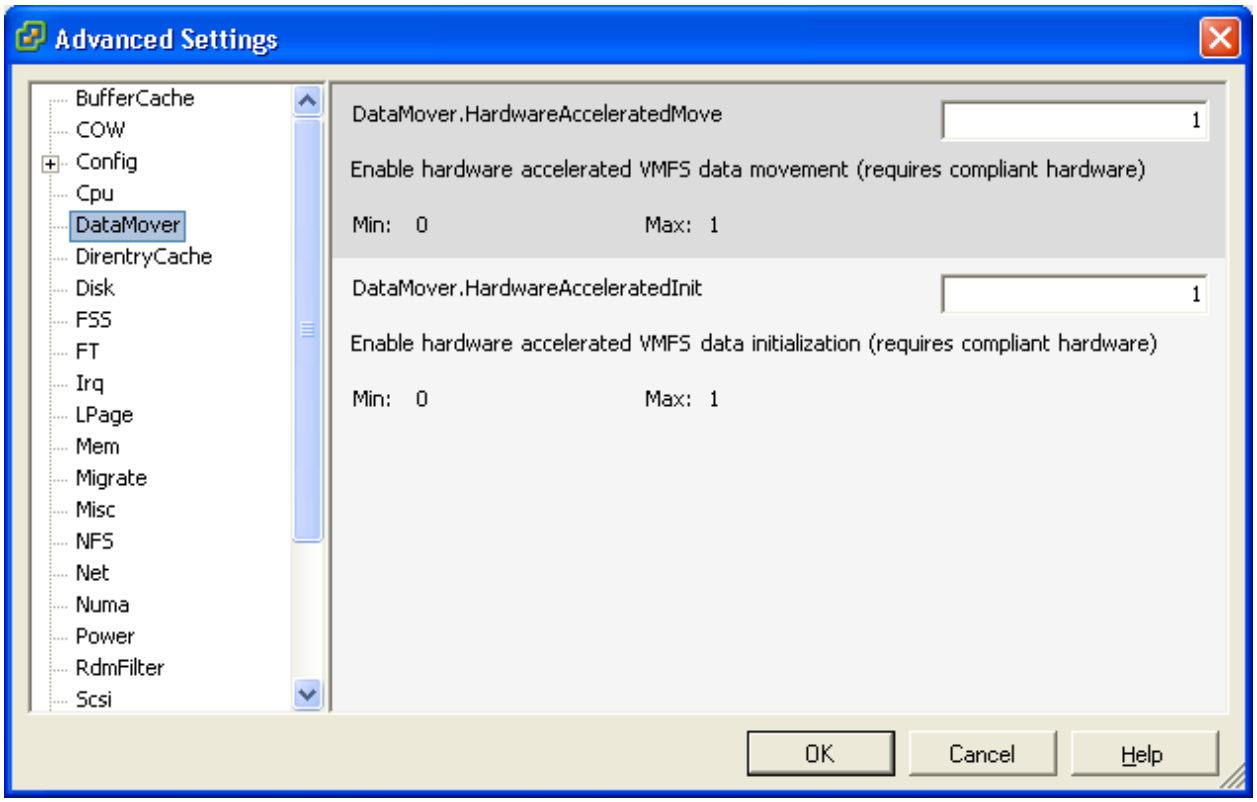
```
[root@esx8 ~]# esxcfg-advcfg -g /DataMover/HardwareAcceleratedMove
Value of HardwareAcceleratedMove is 1
[root@esx8 ~]# esxcfg-advcfg -g /DataMover/HardwareAcceleratedInit
Value of HardwareAcceleratedInit is 1
[root@esx8 ~]# esxcfg-advcfg -g /VMFS3/HardwareAcceleratedLocking
Value of HardwareAcceleratedLocking is 1
```

The `-s` option sets the advanced settings. The following commands turn Full Copy off and then back on.

```
[root@esx8 ~]# esxcfg-advcfg -s 0 /DataMover/HardwareAcceleratedMove
Value of HardwareAcceleratedMove is 0
[root@esx8 ~]# esxcfg-advcfg -s 1 /DataMover/HardwareAcceleratedMove
Value of HardwareAcceleratedMove is 1
```

You can access the VAAI settings under Advanced Settings in the vSphere client that is connected directly to either vCenter or the ESX server. Click the server name and then the Configuration tab. Under Hardware, click Advanced. Advanced settings are organized in a hierarchy. The part of the vSphere API name preceding the dot, as shown in Table 1, is the branch under which the setting is found.

Figure 9) vSphere client showing ESX Advanced Settings → DataMover dialog box.



VAAI CONFIGURATION IN DATA ONTAP

VAAI primitives, referred to as *vStorage* in Data ONTAP, cannot be turned off in any public release of Data ONTAP. If necessary, VAAI primitives must be disabled on the ESX side.

Also, no performance or other tunable options are available. However, there is built-in throttling of VAAI operations, mainly for XCOPY and WRITE SAME operations. XCOPY and WRITE SAME are very small SCSI commands that result in a large amount of work at the back end. In early testing, this work could cause a dramatic performance impact on other non-VAAI workloads. To minimize this impact, NetApp added a throttling mechanism that monitors internal response times and throttles VAAI operations accordingly.

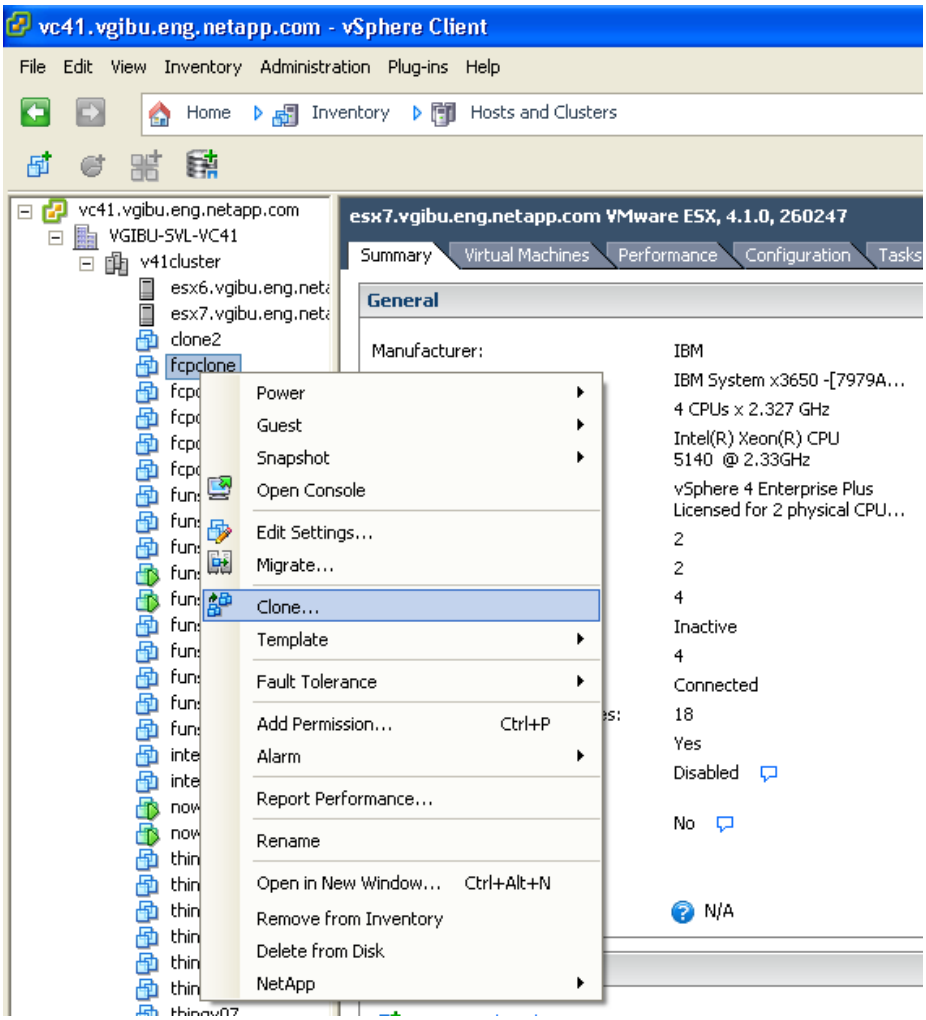
3.3 RUNNING VAAI OPERATIONS

VAAI primitives are used in routine vSphere operations such as creating, cloning, migrating, starting, and stopping VMs. These operations can be executed through the vSphere client for simplicity or from the command line for scripting or to get more accurate timing.

CLONING VMS AND MIGRATING VMS WITH STORAGE VMOTION

To test Full Copy, you can either clone a VM or use Storage VMotion to migrate it to another datastore on the same NetApp storage system connected by way of the same protocol. As shown in Figure 10, in the vSphere client, you right-click the VM and select either Clone or Migrate, then follow the wizard. You can clone to the same datastore or to a different datastore, as long as they are on the same storage system. For Storage vMotion or cold migration, you must migrate to a different datastore on the same storage system. The VM can be running or halted, with or without VMware snapshots.

Figure 10) Accessing the vCenter Migrate and Clone wizards.



From the command line, you can use the `vmkfstools` command to clone virtual disks. Note that this clones only the virtual disk, not the VMX (configuration file) or anything else that makes up the VM. The `vmkfstools` command is useful for performance testing (for example, with the `time` command) and for scripts.

```
[root@esx8 winf35fcp1]# pwd
/vmfs/volumes/f35vmfs1/winf35fcp1
[root@esx8 winf35fcp1]# time vmkfstools -i winf35fcp1.vmdk ../junk/clone2.vmdk
Destination disk format: VMFS zeroedthick
Cloning disk 'winf35fcp1.vmdk'...
Clone: 100% done.

real    0m30.483s
user    0m0.000s
sys     0m0.170s
```

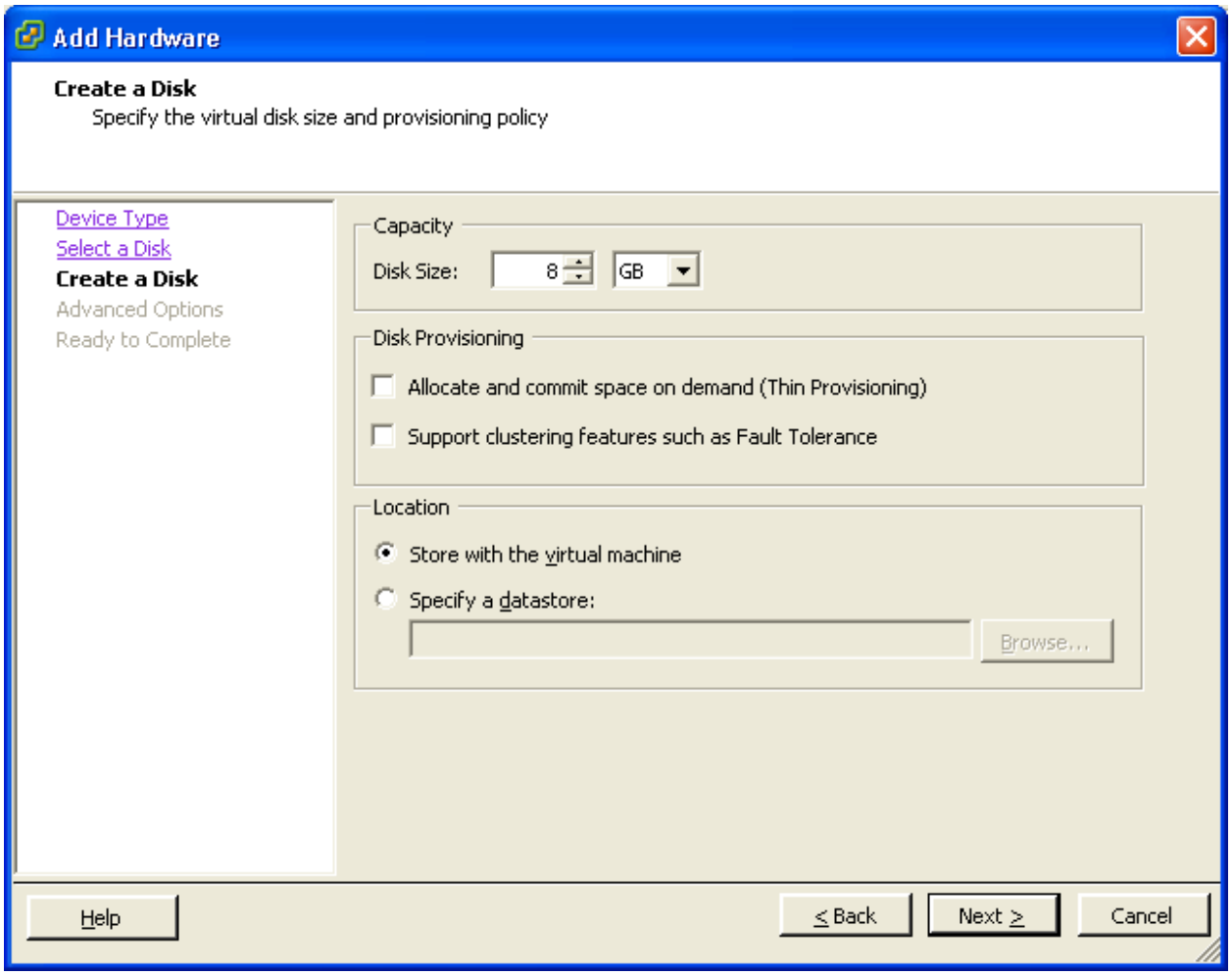
Storage VMotion from the command line requires the vSphere CLI. You can download, install, and run the vSphere CLI from VMware, or you can use the simple vSphere Management Assistant (vMA), which is an appliance VM. The following vMA command migrates a VM between two datastores.

```
[vi-admin@vma ~]$ svmotion --server vc4.vgibu.eng.netapp.com --username administrator --password
something --datacenter VGIBU-SVL-VC4 --vm [vmfs2] vaai1-w2k3/vaai1-w2k3.vmx:vmfs1
```

USING BLOCK ZEROING

Block Zeroing is used in two ways, both invoked by creating either a zeroed-thick virtual disk or an eager-zeroed-thick (EZT) virtual disk. As shown in Figure 11, you select the type of virtual disk in the Create a Disk dialog box where you also specify size and location. Leaving both of the Disk Provisioning checkboxes unchecked (the default) creates a zeroed-thick virtual disk. Checking the Support Clustering Features such as Fault Tolerance checkbox creates an EZT virtual disk.

Figure 11) vSphere client virtual disk type selection.



You can also use the `vmkfstools` command. Prepending the `time` command is useful in obtaining a precise measurement of completion time.

```
[root@esx8 vmfs4]# time vmkfstools -c 1g -d eagerzeroedthick fun.vmdk
Creating disk 'fun.vmdk' and zeroing it out...
Create: 100% done.

real    0m3.294s
user    0m0.000s
sys     0m0.000s
```

When you create a zeroed-thick virtual disk, all blocks are allocated upon creation but are not zeroed until they are first accessed by the VM. The virtual disk is immediately available for use.

With EZT, all blocks are allocated and zeroed upon creation of the virtual disk. The virtual disk is not available until this process completes.

Thin-provisioned virtual disks can also be used. Blocks are neither allocated nor zeroed until first accessed by the VM. The virtual disk is available for use immediately. Allocation of blocks requires changes to VMFS metadata. When this happens, VAW commands occur along with WRITE SAME.

Note: When blocks are zeroed, a larger block than the 512-byte virtual disk block size is zeroed. Typically, zeroing happens in 1MB chunks, but smaller amounts are also sometimes zeroed.

Table 2 summarizes the properties associated with the virtual disk types.

Table 2) Virtual disk types and properties.

Format	Allocated	Zeroed	Available for Use
Zeroed-thick	On create	On first access of VMFS block	Immediately
Eager-zeroed-thick	On create	On creation of VMDK	After zeroing
Thin	On first access of VMFS block	On first access of VMFS block	Immediately

3.4 VIEWING VAAI STATISTICS IN ESXTOP

The `esxtop` command in ESX 4.1 has two new sets of counters for VAAI operations available under the disk device view. Both sets of counters include all three primitives.

To view VAAI statistics using the `esxtop` command, follow these steps:

1. Log in to ESX using an SSH client and set your screen width to at least 130 characters.
2. Enter the `esxtop` command:
   `[root@esx8 ~]# esxtop`
3. Press u to change to the disk device stats view.
4. Press `f` to select fields.

   Note: This selects sets of counters, not individual counters.
5. Press o to select `VAAI Stats` and/or p to select `VAAI Latency Stats`.
6. Optionally, deselect `Queue Stats, I/O Stats`, and `Overall Latency Stats` by pressing `f`, g, and `i` respectively in order to simplify the display.
7. To see the whole LUN field, widen it by pressing `L` (capital) then entering a number (36 is wide enough to see a full NAA ID of a LUN).

```
7:33:57am up 10:10, 134 worlds; CPU load average: 0.01, 0.01, 0.04

DEVICE                                 CLONE_RD CLONE_WR CLONE_F MBC_RD/s MBC_WR/s   ATS
ATSF   ZERO   ZERO_F MBZERO/s
mpx.vmhba0:C0:T0:L0                           0        0       0     0.00     0.00     0
0      0        0     0.00
mpx.vmhba32:C0:T0:L0                          0        0       0     0.00     0.00     0
0      0        0     0.00
mpx.vmhba34:C0:T0:L0                          0        0       0     0.00     0.00     0
0      0        0     0.00
naa.5005076a0400434d                          0        0       0     0.00     0.00     0
0      0        0     0.00
naa.60a98000572d42766d4a5966456e794e  2410     2410       0     0.00     0.00   693
0   4811        0     0.00
```

```
naa.60a98000572d42766d4a59664b51774a     0        0       0     0.00     0.00   278
0   5482        0     0.00
naa.60a98000572d42766d4a5a4457566d4b  9250     6041       0   228.06     0.00  8642
0  21611        0     0.00
naa.60a98000572d42766d4a5a4c5264334c     0        0       0     0.00     0.00   374
0      0        0     0.00
naa.60a98000572d42766d4a5a4e486d3979     0        0       0     0.00     0.00   100
0      0        0     0.00
naa.60a98000572d42766d4a5a725a423771  2394     5603       0     0.00   228.06  2236
0  10246        0     0.00
{NFS}infra                               0        0       0     0.00     0.00     0
0      0        0     0.00
{NFS}nfs99                               0        0       0     0.00     0.00     0
0      0        0     0.00
{NFS}stuff                               0        0       0     0.00     0.00     0
0      0        0     0.00
```

Table 3) VAAI counters in esxtop.

Counter Name	Description
DEVICE	Devices that support VAAI (LUNs on a supported storage system) are listed by their NAA ID. You can get the NAA ID for a datastore from the datastore properties in vCenter, the Storage Details—SAN view in Virtual Storage Console, or using the vmkfstools `-P /vmfs/volumes/<datastore>` command. NetApp LUNs start with naa.60a98000. **Note:** Devices or datastores other than LUNs on an external storage system such as CD-ROM, internal disks (which may be physical disks or LUNs on internal RAID controllers), and NFS datastores are listed but have all zeroes for VAAI counters.
CLONE_RD	Number of Full Copy reads from this LUN.
CLONE_WR	Number of Full Copy writes to this LUN.
CLONE_F	Number of failed Full Copy commands on this LUN.
MBC_RD/s	Effective throughput of Full Copy command reads from this LUN in megabytes per second.
MBC_WR/s	Effective throughput of Full Copy command writes to this LUN in megabytes per second.
ATS	Number of successful lock commands on this LUN. ATS refers to the old COMPARE AND WRITE name—Atomic Test and Set.
ATSF	Number of failed lock commands on this LUN. ATS refers to the old COMPARE AND WRITE name—Atomic Test and Set.
ZERO	Number of successful Block Zeroing commands on this LUN.
ZERO_F	Number of failed Block Zeroing commands on this LUN.
MBZERO/s	Effective throughput of Block Zeroing commands on this LUN in megabytes per second.

Counters that count operations do not return to zero unless the server is rebooted. Throughput counters are zero when no commands of the corresponding primitive are in progress.

Clones between VMFS datastores and Storage VMotion operations that use VAAI increment clone read for one LUN and clone write for another LUN. In any case, the total for clone read and clone write columns should be equal.

3.5 OBSERVING VSTORAGE ON NETAPP STORAGE SYSTEMS

Several counters are available under the Data ONTAP `stats` command. There are separate counters for each VAAI primitive under the vstorage object. Counters are accessed as follows.

```
f35> stats show vstorage
vstorage:vfiler0:xcopy_copy_reqs:23860
vstorage:vfiler0:xcopy_abort_reqs:0
vstorage:vfiler0:xcopy_status_reqs:0
vstorage:vfiler0:xcopy_total_data:24432640
vstorage:vfiler0:xcopy_invalid_parms:0
vstorage:vfiler0:xcopy_authorization_failures:0
vstorage:vfiler0:xcopy_authentication_failures:0
vstorage:vfiler0:xcopy_copy_failures:0
vstorage:vfiler0:xcopy_copyErr_isDir:0
vstorage:vfiler0:xcopy_copyErr_data_unrecov:0
vstorage:vfiler0:xcopy_copyErr_offline:0
vstorage:vfiler0:xcopy_copyErr_staleFH:0
vstorage:vfiler0:xcopy_copyErr_IO:0
vstorage:vfiler0:xcopy_copyErr_noSpace:0
vstorage:vfiler0:xcopy_copyErr_diskQuota:0
vstorage:vfiler0:xcopy_copyErr_readOnly:0
vstorage:vfiler0:xcopy_copyErr_other:0
vstorage:vfiler0:xcopy_intravol_moves:23860
vstorage:vfiler0:xcopy_intervol_moves:0
vstorage:vfiler0:xcopy_one2one_moves:0
vstorage:vfiler0:xcopy_one2many_moves:0
vstorage:vfiler0:writesame_reqs:23985
vstorage:vfiler0:writesame_holepunch_reqs:0
vstorage:vfiler0:writesame_total_data:24449920
vstorage:vfiler0:vaw_reqs:2035
vstorage:vfiler0:vaw_miscompares:0
```

Table 4 provides definitions and units for the vStorage counters that you are likely to encounter in normal operations.

Table 4) vStorage counters, definitions, and units.

Counter Name	Definition	Unit
xcopy_copy_reqs	Number of Full Copy requests (XCOPY SCSI commands) received	Count
xcopy_total_data	Amount of data copied in response to XCOPY requests	kB (1024 bytes)
xcopy_intravol_moves	Number of Full Copy (XCOPY) requests that copied blocks within the same volume; usually clone operations	Count
xcopy_intervol_moves	Number of Full Copy (XCOPY) requests that copied blocks from one volume to another; SVMotion or clone operations	Count
writesame_reqs	Number of Block Zeroing requests (WRITE SAME SCSI commands) received	Count

Counter Name	Definition	Unit
writesame_holepunch_reqs	Number of Block Zeroing requests (WRITE SAME SCSI commands) with unmap bit set received; allows Data ONTAP to free up zeroed blocks	Count
writesame_total_data	Amount of data written as WRITE SAME patterns	kB (1024 bytes)
-vaw_reqs	Number of Hardware-Assisted Locking requests (verify and write SCSI commands) received	Count
vaw_miscompares	Number of times the VAW read payload did not match the block on disk	Count

It is sometimes useful to reset these counters. VAAI counters can be reset by using the `vstorage` command, which is accessed from the advanced privilege mode.

```
f35> priv set advanced
Warning: These advanced commands are potentially dangerous; use
         them only when directed to do so by NetApp
         personnel.
f35*> vstorage zero_counters
```

This command sets all vstorage counters to 0. It is not possible to reset individual counters or all counters of a specific VAAI primitive.

EXAMPLES

In the first example, `vmkfstools -i` was used to clone a VMDK with 1GB of data written in it. Counters for the vstorage object were reset before the test. A total of 1146880kB was copied in 280 XCOPY commands, demonstrating that the primitive requested 4MB to be copied at a time.

```
After XCOPY (vmkfstools -i to copy a VMDK with ~1GB written in it)
f35*> stats show vstorage
vstorage:vfiler0:xcopy_copy_reqs:280
vstorage:vfiler0:xcopy_abort_reqs:0
vstorage:vfiler0:xcopy_status_reqs:0
vstorage:vfiler0:xcopy_total_data:1146880
vstorage:vfiler0:xcopy_invalid_parms:0
vstorage:vfiler0:xcopy_authorization_failures:0
vstorage:vfiler0:xcopy_authentication_failures:0
vstorage:vfiler0:xcopy_copy_failures:0
vstorage:vfiler0:xcopy_copyErr_isDir:0
vstorage:vfiler0:xcopy_copyErr_data_unrecov:0
vstorage:vfiler0:xcopy_copyErr_offline:0
vstorage:vfiler0:xcopy_copyErr_staleFH:0
vstorage:vfiler0:xcopy_copyErr_IO:0
vstorage:vfiler0:xcopy_copyErr_noSpace:0
vstorage:vfiler0:xcopy_copyErr_diskQuota:0
vstorage:vfiler0:xcopy_copyErr_readOnly:0
vstorage:vfiler0:xcopy_copyErr_other:0
vstorage:vfiler0:xcopy_intravol_moves:280
vstorage:vfiler0:xcopy_intervol_moves:0
vstorage:vfiler0:xcopy_one2one_moves:0
vstorage:vfiler0:xcopy_one2many_moves:0
vstorage:vfiler0:writesame_reqs:1
vstorage:vfiler0:writesame_holepunch_reqs:0
vstorage:vfiler0:writesame_total_data:64
vstorage:vfiler0:vaw_reqs:65
vstorage:vfiler0:vaw_miscompares:0
```

In the next example, the `copy` command in a Windows® guest was used to copy a 1GB file into previously unused space in a thin virtual disk. Counters for the vstorage object were reset before the test. Snipped counters were 0. Note that there were 16 VAW requests for 1GB of write, indicating that 64MB of space is allocated from the VMFS to the VMDK at a time.

```
f35*> stats show vstorage
<snip>
vstorage:vfiler0:writesame_reqs:1023
vstorage:vfiler0:writesame_holepunch_reqs:0
vstorage:vfiler0:writesame_total_data:1047552
vstorage:vfiler0:vaw_reqs:16
vstorage:vfiler0:vaw_miscompares:0
```

4 PERFORMANCE

Performance improvements offered by VAAI can be grouped into three categories:

- Reduced time to complete VM cloning and Block Zeroing operations
- Reduced use of server compute and storage network resources
- Improved scalability of VMFS datastores in terms of the number of VMs per datastore and the number of ESX servers attached to a datastore

The actual improvement seen in any given environment depends on a number of factors, discussed in the following section. In some environments, improvement may be small.

4.1 PERFORMANCE OF CLONING, MIGRATING, AND ZEROING VMS

The biggest factor for cloning and Block Zeroing operations is whether the limiting factor is on the front end or the back end of the storage controller. If the throughput of the storage network is slower than the disks can handle, offloading the bulk work of reading and writing virtual disks for cloning and migration and writings zeroes for virtual disk initialization can help greatly.

One example where substantial improvement is likely is when the ESX servers use 1GbE iSCSI to connect to a NetApp storage system that has multipath HA (MPHA) connected Fibre Channel disks. The front end at 1Gbps does not support enough throughput to saturate the back end, which is supported by multiple links at 4Gbps. When cloning or zeroing is offloaded, all that goes across the front end is small commands with a small payload. The work of reading and writing, or just writing as in the case of Block Zeroing, is done by the storage controller directly to disk.

4.2 VMFS DATASTORE SCALABILITY

Documentation from various sources, including VMware professional services best practices, has traditionally recommended 20 to 30 VMs per VMFS datastore, and sometimes even fewer. Documents for VMware Lab Manager suggest limiting the number of ESX servers in a cluster to eight. These recommended limits are due in part to the effect of SCSI reservations on performance and reliability. Extensive use of some features, such as VMware snapshots and linked clones, can trigger large numbers of VMFS metadata updates, which require locking. Before vSphere 4.1, reliable locks on smaller objects were obtained by briefly locking the entire LUN with a SCSI reservation. Any other server trying to access the LUN during the reservation would fail and would wait and retry up to 80 times by default. This wait and retry added to perceived latency and reduced throughput in VMs. In extreme cases, if the other server exceeded the number of retries, errors would be logged in the vmkernel logs and I/Os could return as failures to the VM.

When all ESX servers sharing a datastore support VAAI, Hardware-Assisted Locking can eliminate SCSI reservations, at least reservations due to obtaining smaller locks. The result is that datastores can be scaled to more VMs and attached servers than previously.

NetApp has tested up to 128 VMs in a single VMFS datastore. The number of VMs was limited in testing to 128 because the maximum addressable LUN size in ESX is 2TB, which means that each VM can occupy a maximum of 16GB, including virtual disk, virtual swap, and any other files. Virtual disks much smaller than this generally do not allow enough space to be practical for an OS and any application.

Load was generated and measured on the VMs by using Iometer. For some tests, all VMs had load. In others, such as when sets of VMs were started, stopped, or suspended, load was placed only on VMs that stayed running.

Tests such as starting, stopping, and suspending numbers of VMs were run with Iometer workloads running on other VMs that weren't being started, stopped, or suspended. Additional tests were run with all VMs running Iometer, and VMware snapshots were created and deleted as quickly as possible on all or some large subset of the VMs.

The results of these tests demonstrated that performance impact measured before or without VAAI was either eliminated or substantially reduced when using VAAI, to the point that datastores could reliably be scaled to 128 VMs in a single LUN.

5 REFERENCES

- ANSI T10 SCSI Operation Codes
  www.t10.org/lists/op-num.htm
- NetApp@VMworld 2010 VAAI Demo: vStorage APIs for Array Integration and NetApp
  www.youtube.com/watch?v=fInR5bPBAjw
- VMware KB: vStorage APIs for Array Integration FAQ
  http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1021976

www.netapp.com