# VMware® Virtual SAN Diagnostics and Troubleshooting Reference Manual

Cormac Hogan
Storage and Availability Business Unit
VMware
v 1.0/March 2015

vmware®

# 1. Introduction

VMware's Virtual SAN is designed to be simple: simple to configure, and simple to operate. This simplicity masks a sophisticated and powerful storage product.

The purpose of this document is to fully illustrate how Virtual SAN works behind the scenes: whether this is needed in the context of problem solving, or just to more fully understand its inner workings.

## Health Services

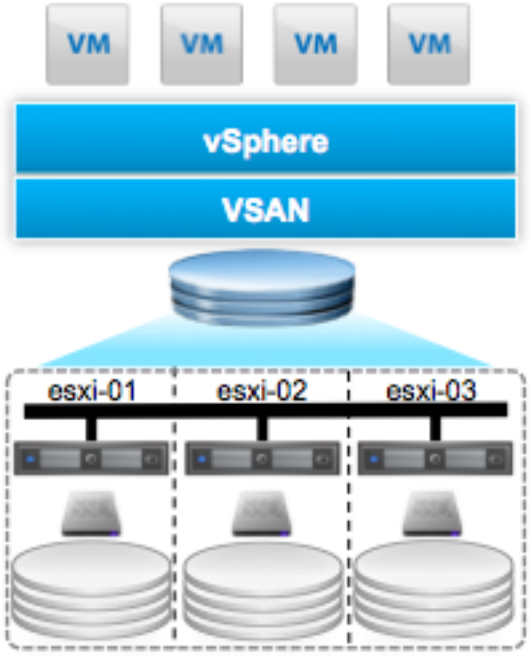
Virtual SAN 6.0 comes with a Health Services plugin. This feature checks a range of different health aspects of Virtual SAN, and provides insight into the root cause of many potential Virtual SAN issues. The recommendation when triaging Virtual SAN is to begin with the Virtual SAN Health Services. Once an issue is detected, the Health Services highlights the problem and directs administrators to the appropriate VMware knowledgebase article to begin problem solving.

Please refer to the *Virtual SAN Health Services Guide* for further details on how to get the Health Services components, how to install them and how to use the feature for troubleshooting common Virtual SAN issues.

# 2. What is VMware Virtual SAN (VSAN)?

VMware Virtual SAN is a new software-defined storage tier for VMware vSphere, bringing the benefits of the software-defined data center to storage. By clustering server magnetic disks and flash devices in hybrid condifurations, or many flash devices in all-flash configurations, Virtual SAN creates a flash-optimized, highly resilient shared datastore designed for virtual environments.

Virtual SAN aggregates the local or direct attached storage of ESXi hosts and makes them appear like a single pool of shared storage across all the hosts. It introduces a converged storage-compute platform – virtual machines are running on ESXi hosts as per usual, while a small percentage of CPU and memory resources are used to serve the storage needs of the same virtual machines.




Based on a hypervisor-converged distributed architecture embedded in the vSphere kernel, Virtual SAN can make optimal data placement decisions and I/O optimizations. Because Virtual SAN sits directly in the I/O data path, the product is able to deliver the highest levels of performance, scalability, and resilience without taxing the CPU with additional overhead.

Since Virtual SAN has a distributed architecture, virtual machines can be deployed in a way that no single hardware failure will impact their availability.

Virtual SAN also differs from other storage products in its policy-based approach to storage management. This management architecture enables administrators to

specify storage attributes such as capacity, performance, and availability - in the form of simple policies - on a per-VM basis. These policies, governed by service-level agreements (SLAs), dynamically self-tune and load-balance the system so that each virtual machine has the right level of resources. The system can adapt to ongoing changes in workload conditions to ensure that each virtual machine has the storage resources it needs.

Virtual SAN distributed architecture leverages enterprise-grade flash devices such as SSDs for high-performance read/write caching and in the 5.5 release, utilizes magnetic disks for cost-effective data persistence. In the 6.0 release, VMware supports flash devices for both the caching layer and the capacity layer. The Virtual SAN datastore granularly scales up by adding more disks or scales out by adding more hosts, allowing users to configure the system to meet their needs flexibly. This document will cover both the 5.5 and 6.0 versions of Virtual SAN.

Virtual SAN is deeply integrated with the VMware stack. It gets optimal performance by having critical code paths run natively in the kernel, and also works seamlessly with other core products and features of VMware.

Virtual SAN includes a number of essential tools for monitoring operation and performance. The purpose of this reference manual is to assist with diganosing and troubleshooting of Virtual SAN, and assist with how to use these tools for problem solving and getting to a root cause of an issue.

## Common troubleshooting scenarios with Virtual SAN

As Virtual SAN is a software-based storage product, it is entirely dependent on the proper functioning of its underlying hardware components such as the network, the storage I/O controller, and the storage devices themselves.

As Virtual SAN is an enterprise storage product, it can put an unusually demanding load on supporting components and subsystems, exposing flaws and gaps that might not be seen with simplistic testing or other, less-demanding use cases.

Indeed, most Virtual SAN troubleshooting exercises involve determining whether or not the network is functioning properly, or whether the Virtual SAN VMware Compatibility Guide (VCG) has been rigorously followed.

As Virtual SAN uses the network to communicate between nodes, a properly configured and fully functioning network is essential to operations. Many Virtual SAN errors can be traced back to things like improperly configured multicast, mis-matched MTU sizes and the like. More than simple TCP/IP connectivity is required for Virtual SAN.

Virtual SAN uses server-based storage components to recreate the functions normally found in enterprise storage arrays. This architectural approach demands a **rigorous discipline** in sourcing and maintaining the correct storage I/O controllers, disks, flash devices, device drivers and firmware, as documented in the VMware Virtual SAN VCG. Failure to adhere to these guidelines will often result in erratic performance, excessive error messages or both.

This guide demonstrates how to verify that your Virtual SAN is working correctly, and if it is not working well, how to find the cause.

## Layout of this document

This document will first introduce the reader to the various tools that are available for troubleshooting Virtual SAN. The reader will also be shown various commands and tools in chapter 3 that will allow them to determine whether or not their hardware is on the VMware Compatibility Guide for Virtual SAN. This is most important.

Once a reader has confirmed that they are using supported hardware, the primary purpose of chapter 4, the document examines tools and commands that are available for on-going proactive monitoring and reactive troubleshooting of Virtual SAN networking, storage, and performance. This makes up the bulk of the remainder of the reference manual. In most all cases, the reader is provided with a description of why the command is useful, and examples of the expected output from many of the commands is shown.

An overview on how to use certain VSAN specific tools in provided. Notably, the reader is shown how to use the Ruby vSphere Console (RVC) and VSAN Observer effectively.

In order to understand the output displayed in some of these commands, the reader is provided with certain internal details of Virtual SAN. However the reader needs to be aware that some outputs from the tools are primarily for the purposes of VMware's support and engineering staff. This is highlighted in the text.

# 3. Tools to troubleshoot Virtual SAN

While most normal Virtual SAN operations can be done without ever leaving the vSphere web client user interface, additional tools are available for diagnostics and troubleshooting purposes. Investing the time to set up and learn these tools will give a deeper, more complete view of what Virtual SAN is doing behind the scenes.

## vSphere Web Client

The vSphere Web Client is the primary tool to configure and manage Virtual SAN. The vSphere Web Client is used to configure storage policies, and monitor their compliance. It also can be used to inspect underlying disk devices and how they are being used by Virtual SAN.

As a troubleshooting tool, the vSphere Web Client can be configured to present specific alarms and warnings associated with Virtual SAN. It will also highlight certain network misconfiguration issues, and whether or not hardware components are functioning properly.

Additionally, the vSphere Web Client can provide at-a-glance overviews of individual virtual machine performance, and indicate whether or not Virtual SAN is recovering from a failed hardware component.

The vSphere Web Client is the logical place to start when diagnosing or troubleshooting a suspected problem.

Although the vSphere Web Client does not include many of the low-level Virtual SAN metrics and counters, it does have a pretty comprehensive set of virtual machine metrics. You can use the performance charts in the vSphere Web Client to examine the performance of individual virtual machines running on Virtual SAN.

## ESXCLI

Every ESXi host supports a direct console that can be used for limited administrative purposes: starting and stopping the system, setting parameters and observing state. As such, ESXCLI is an important tool in regards to Virtual SAN diagnostics and troubleshooting. ESXCLI works through the use of individual namespaces that refer to different aspects of ESXi, including Virtual SAN. To see what options are available to ESXCLI for monitoring and troubleshooting Virtual SAN, simply type:

```
esxcli vsan
```

at the ESXi shell, and a list of commands will be displayed.

Keep in mind that ESXCLI can only communicate with one host or ESXi instance at a time. To look at cluster wide information, the Ruby vSphere Console (RVC) should be used.

There are lots of ways to display similar information using the ESXCLI. In this reference manual, the focus will be on the best one or two ways to get the information.

## Ruby vSphere Console - RVC

Ruby vSphere Console is a Ruby-based expandable management platform from which you can utilize any API vCenter exposes.

It can simply be described as a command-line console UI for VMware ESXi hosts and VMware vCenter Server.

The vSphere inventory is presented as a virtual filesystem, allowing you to navigate and run commands against managed entities using familiar shell syntax, such as "`cd`" to change directory and "`ls`" to list directory (inventory) contents. RVC has been extended to provide a wealth of useful information about Virtual SAN health and status.

RVC is included in both the Windows and Linux versions of vCenter Server since 5.5U1.

## VSAN Observer

VSAN Observer is a monitoring and troubleshooting tool that is available with RVC, and enables analysis of a Virtual SAN Cluster. VSAN Observer is launched from RVC. Its captures low-level metrics from Virtual SAN and presents them in a format that is easily consumable via a web browser. It is a tool that is typically deployed to assist with monitoring and troubleshooting Virtual SAN issues. It monitors Virtual SAN in real-time.

Virtual SAN can be examined either from the perspective of physical resources (CPU, Memory, disks) with a wealth of different metrics. It can also be monitored from a virtual machine perspective, allowing resources consumed by the virtual machine to be examined, and whether or not a virtual machine contends with other virtual machines for resources, etc.

Two key points here:

- VSAN Observer is used for performance troubleshooting
- VSAN Observer is built on RVC

## Third party tools

As Virtual SAN uses third-party storage I/O controllers and flash devices, it may be necessary to use specific tools provided by these vendors to configure and check status.

Using these tools, e.g. `MegaCLI` from LSI or the `hpssacli` extension to ESXCLI from HP, you can check the status of the storage controller as well as the disk devices attached to the controller. You can also verify that the cache is configured properly on the controller (e.g. disabled, or set to 100% read, in both cases disabling write cache).

You may also find that you PCI-E flash device vendor also supplied tools for checking configuration and status. For example, FusionIO/SanDisk is one vendor that provides such a set of tools.

We will look at how to use these tools in various places in this reference manual.

## Troubleshooting tools summary

With this set of tools at your disposal, you should be in a position to monitor, diagnose and troubleshoot Virtual SAN. The vSphere Web Client and ESXCLI commands are covered in the official vSphere documentation, and only the commands relevant to Virtual SAN are covered in this reference manual. The complete official vSphere documentation can be found at the following link: https://www.vmware.com/support/pubs/.

This reference manual will focus on all aspects of RVC and VSAN Observer, with a goal of getting the reader to a point where they can confidently troubleshoot Virtual SAN issues. The combination of the vSphere Web Client, ESXCLI, RVC and VSAN Observer represent a very complete and powerful toolset to fully observe the inner workings of Virtual SAN. The remainder of this reference manual will focus on how to use these tools to diagnose and troubleshoot Virtual SAN environments.

# 4. VMware Compatibility Guide & Virtual SAN

The VMware Compatibility Guide (VCG, also known as the Hardware Compatibility List or HCL) - http://www.vmware.com/resources/compatibility/search.php - contains lists of devices and components that have been successfully tested and qualified to work with Virtual SAN. It includes specific model numbers, recommended drivers and firmware.

It is imperative that you verify that VMware supports the hardware components *(including driver and firmware versions*) that are being used in a Virtual SAN cluster. Many of the issues reported by customers for Virtual SAN relate to unsupported hardware, drivers or firmware. In this section, we will show you how to verify that your infrastructure supports Virtual SAN.

***WARNING: failure to precisely adhere to the guidelines provided in the VMware Compatibility Guide may result in erratic performance, unusual error conditions and/or potential data loss. VMware support may not be in a position to assist you if the components used to build a Virtual SAN environment are not on the VCG.***

The hope is that the Virtual SAN hardware has been checked and verified before deployment, but should there be a need to check whether or not a Virtual SAN cluster is running with supported components, the procedures outlined here will help to verify.

## Checking vSphere software versions

While VMware supports Virtual SAN running with vSphere 6.0, 5.5U2 and 5.5U1, it would always recommend running the latest versions of vSphere software, both ESXi and vCenter Server. Ensure that your vCenter server version, visible from the vSphere web client UI is synchronized with the ESXi versions, visible via the `vmware -v` command run from the ESXi command line.

VMware had a large number of participants in the Virtual SAN BETA in vSphere 5.5. VMware does not support BETA code in production environments obviously, and significant changes were made to the on-disk format of Virtual SAN before it was made generally available (GA). So although Virtual SAN code is available in vSphere 5.5, the Virtual SAN code is beta quality and **you cannot use this in production.** You must use vSphere 5.5U1, 5.5U2 or later as mentioned earlier.

Be aware that VMware does not support upgrading a BETA version of Virtual SAN to a GA version. In such cases, **a fresh deployment of Virtual SAN is required**, i.e. a fresh deployment of vSphere 5.5U1, 5.5U2, etc. **Do not attempt to upgrade** from 5.5 to 5.5U1 or 5.5U2 if you were previously using the beta version of Virtual SAN, and wish to use the GA version.

Anyone who is considering an evaluation of Virtual SAN must pick up the latest versions of software. VMware continuously fixes issues encountered by customers, so by using the latest version of the software, you avoid issues already fixed.

## A note about Virtual SAN Ready Nodes

There are two ways to build a Virtual SAN Node:

- Build your own based on certified components

or

- Choose from list of Virtual SAN Ready Nodes

Virtual SAN Ready Node is a validated server configuration in a tested, certified hardware form factor for Virtual SAN deployment, jointly recommended by the server OEM and VMware. Virtual SAN Ready Nodes are ideal as hyper-converged building blocks for larger datacentre environments looking for automation and a need to customize hardware and software configurations. Further details on Virtual SAN Ready Nodes can be found here:

http://partnerweb.vmware.com/programs/vsan/Virtual%20SAN%20Ready%20Nodes.pdf

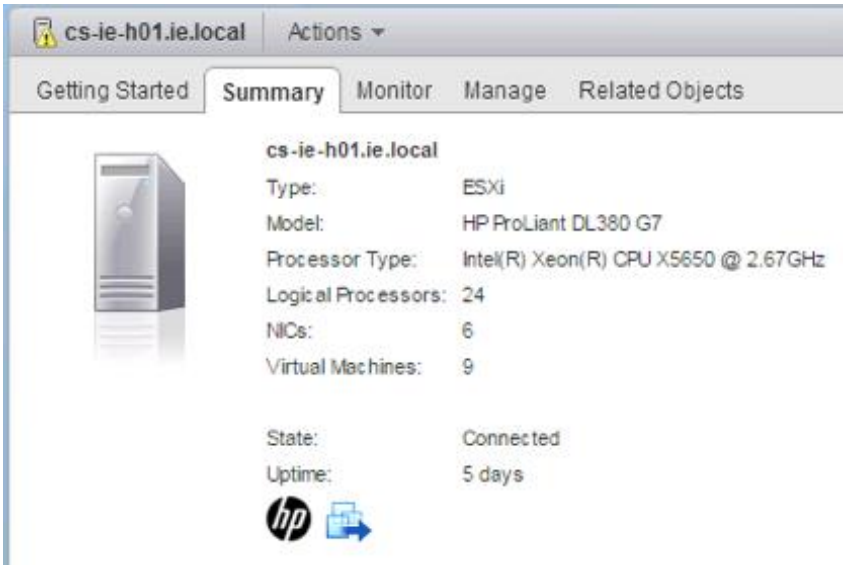
## A note about VMware EVO:RAIL

Another option available to customer is VMware EVO:RAIL™. EVO:RAIL combines VMware compute, networking, and storage resources into a hyper-converged infrastructure appliance to create a simple, easy to deploy, all-in-one solution offered by our partners. EVO:RAIL software is fully loaded onto a partners' hardware appliance and includes VMware Virtual SAN. Further details on EVO:RAIL can be found here:

http://www.vmware.com/products/evorail

# Checking host/server compatibility

The first step is to ensure that the servers are supported for Virtual SAN. The vSphere web client provides information about the server that, in turn, will allow you to check the VCG for supportability.




There are, of course, other ways of capturing this information, such as through the use of ESXCLI command.

### esxcli hardware platform get

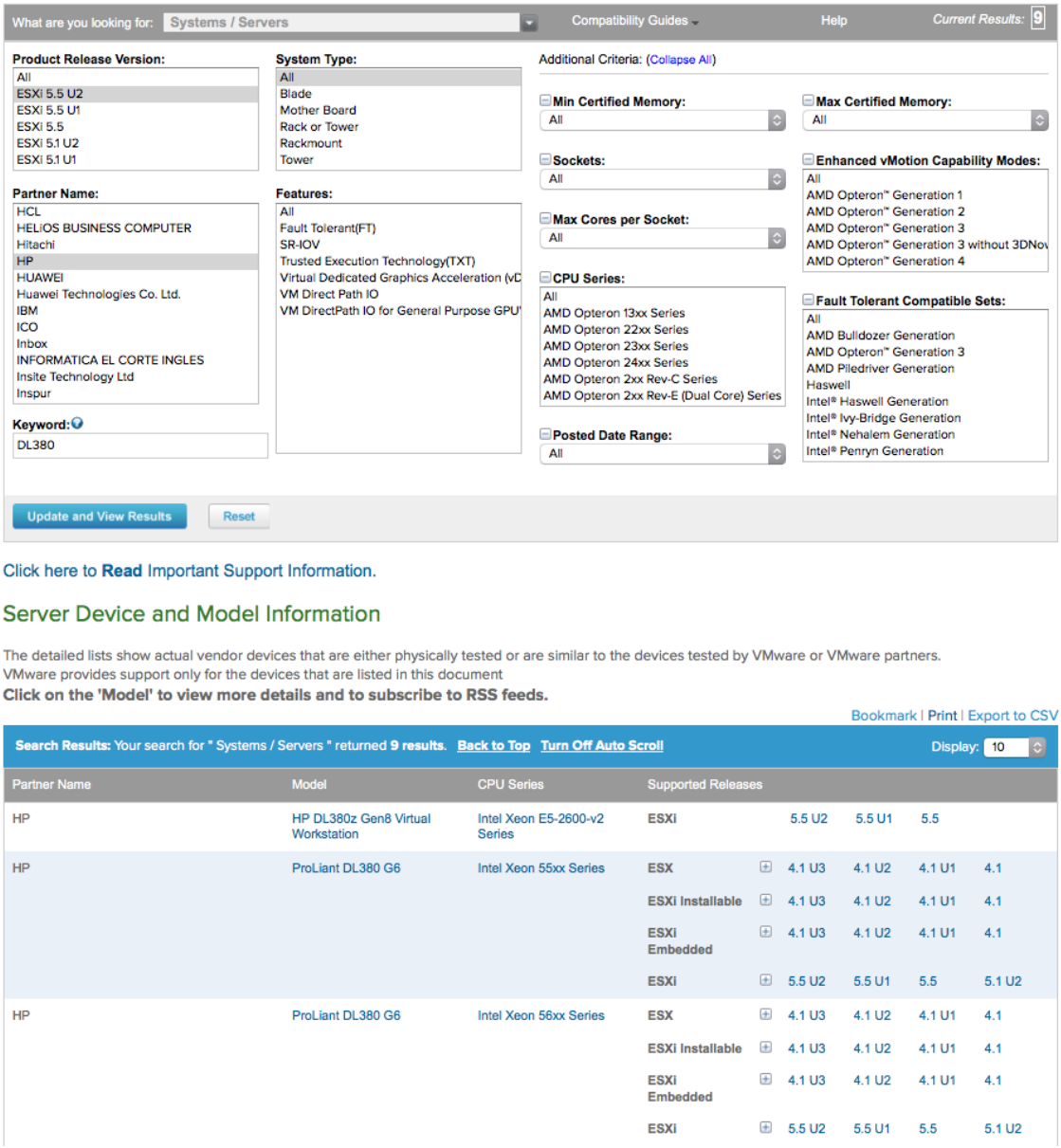
Details about your host/server are shown from the ESXCLI command `esxcli hardware platform get`. As can be seen from the output below, this host is an HP Proliant DL380 G7:

```
~ # esxcli hardware platform get
 Platform Information
    UUID: 0x39 0x33 0x38 0x35 0x36 0x36 0x5a 0x43 0x32 0x31 0x31 0x34 0x30 0x42 0x32 0x30
    Product Name: ProLiant DL380 G7
    Vendor Name: HP
    Serial Number: CZ21140B20
    IPMI Supported: true
 ~ #
```

***Caution**: Note that this host is shown for illustration and troubleshooting purposes only. This server generation from HP doesn't have a controller on the VSAN VCG, as shall be seen shortly when we look at the supportability of other components. The first set of controllers from HP for Virtual SAN on the VCG is supported with the G8 family only.*

# Verify server support via VCG

Once the server type has been identified, navigate to the VCG, select System/Servers in the "What are you looking for:" field, select appropriate product release version and partner name, and verify that the server is supported.




This completes step 1, and at this stage the server has been confirmed as supporting ESXi. The next step entails checking whether the other components on the host are supported for running Virtual SAN.

# Checking host memory requirements

The number of disk groups and the number of disks that can be supported in a disk group is directly related to the amount of host memory available on an ESXi host in the Virtual SAN Cluster. At a minimum, it is required that a host has at least 6GB of memory. This will allow you to create a single disk groups, with one flash device and one magnetic disk per host in hybrid configurations, or one flash cache device and one flash capacity device in all-flash configurations. However, with only 6GB of memory, there will be very little will be available for VMs to consume.

If the Virtual SAN configuration requires a host to contain the maximum number of disks (7 magnetic disks x 5 disk groups in both all-flash and hybrid configurations), then the host must contain 32GB of memory. These guidelines are the same for Virtual SAN 5.5 and 6.0, as well as hybrid and all-flash configurations.

One final consideration is that the amount of host memory also impacts the number of components that can be created on the Virtual SAN cluster. With 8GB of memory, the maximum number of components supported is 750/host. With 32GB, one can create the maximum number of components/host, which are 9,000. Components will be covered in greater detail later in this manual.

## Symptoms of host memory shortage

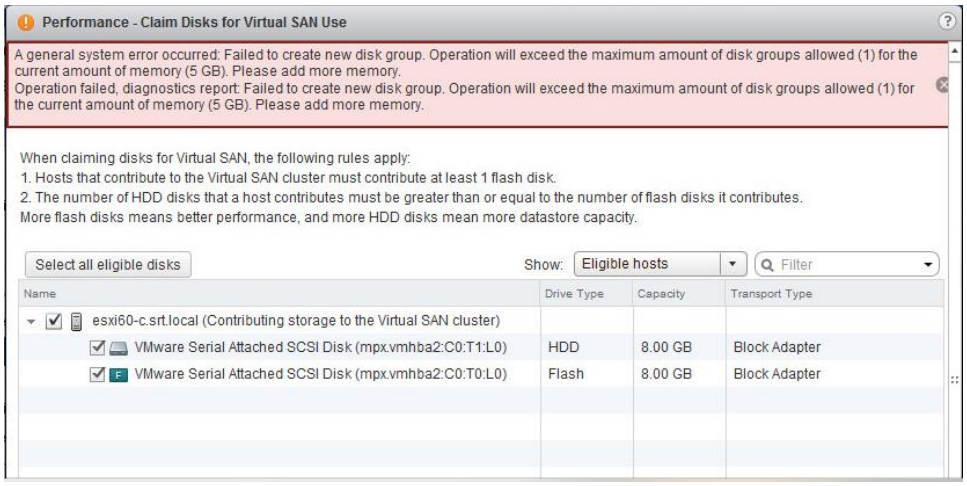
A symptom when there is a lack of host memory is that the following error is displayed in the VMkernel log when you try to create a disk group. The operation itself fails silently in the vSphere web client in vSphere 5.5. The ESXi host VMkernel logs need to be examined to find the root cause of the problem:

```
2013-12-10T17:00:49.878Z cpu0:1000019952)WARNING: LSOMCommon:
LSOM_GlobalMemInit:1000: Unable to create LSOM slabs

2013-12-10T17:00:50.290Z cpu0:1000019952)WARNING: PLOG: PLOG_InitDevice:145:
Failed to initialize PLOG's memory: Admission check failed for memory resource

2013-12-10T17:00:50.296Z cpu0:1000019952)WARNING: PLOG: PLOGProbeDevice:3585:
Failed to init PLOG device <mpx.vmhba1:C0:T2:L0:1>
```

A similar error is thrown when trying to add new disks to a disk group:

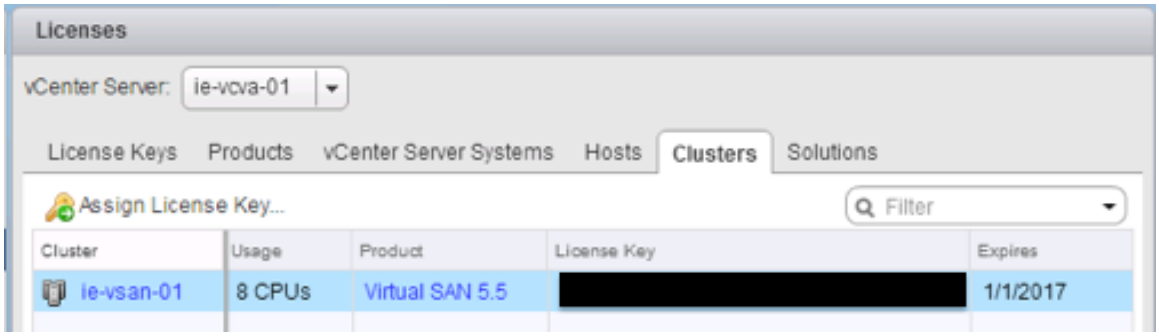
```
2013-04-23T15:22:14.831Z              cpu1:1000027564)WARNING:              PLOG:
PLOGInitAndAnnounceMD:3780:    Unable    to    allocate    or    get    recovered
mpx.vmhba2:C0:T6:L0:2  state5257150e-b9d8-a627-4011-5a90cee6e862  failed  Out  of
memory
```

In 6.0, this behavior has been improved considerably and a warning is now displayed in the vSphere web client if a shortage of memory resources is encountered when trying to create a disk group.




## License check

It is important to ensure that the Virtual SAN cluster is correctly licensed. The easiest way to do this is via the vSphere Web Client Licensing section. Ensure that there are enough licenses for each of the hosts in the cluster, and that licenses are not over-committed; in other words, ensure that there are enough licenses to cover all of the CPUs in all of the hosts in the Virtual SAN cluster.




Note that hosts that are not contributing to the Virtual SAN datastore in the cluster, and are only consuming the storage, still require a valid Virtual SAN license.

Note that Virtual SAN 6.0 introduces an additional license requirement for all-flash configurations. If you wish to use all-flash functionality, an additional all-flash license is required.

## Homogenous host configurations

As a best practice, VMware recommends having identical, uniform configurations across all ESXi hosts participating in a Virtual SAN cluster. While Virtual SAN will work with different controllers, disks, network interface cards and flash devices in different hosts, it can lead to imbalance in the deployment of objects and components across the cluster, which in turn can lead to degraded performance.

Moreover, it can also compromise availability. An imbalanced cluster makes for a larger fault domain in the cluster. Failure of hosts with larger amounts of storage can result in higher probability of data unavailability, as there may not be enough storage capacity remaining in the cluster to rebuild the missing virtual machine components.

This is the reason why VMware strongly recommends identical configurations on each host, including number of disks and disk group and cache sizes. There is also a best practice requirement to have a 1:10 flash cache to capacity layer ratio. This will be elaborated on in greater detail later in the manual.

Troubleshooting will also be made easier with homogenous host configurations.

## A note about multiple controllers and SAS expanders

Virtual SAN supports multiple storage I/O controllers per ESXi host. The maximum number of capacity layer disks per host is 35 (7 disks per disk group, 5 disk groups per host). Some controllers have 16 ports and therefore allow up to 16 disks behind one controller. This means that with just two controllers in one host, a design can get close to the maximums of 35 disks. However, some controllers only have 8 ports, so a total of 5 controllers would be needed to reach the maximum (5 flash disks and 35 capacity layer disks).

SAS Expanders are sometimes considered to extend the number of storage devices that can be configured with a single storage I/O controller. VMware has not extensively tested SAN expanders with VSAN, and thus does not encourage their use. In addition to potential compatibility issues, the use of SAS expanders may impact performance and increase the impact of a failed disk group. Validation of SAS expanders is still a work in progress.

# Part 1 - Gathering controller/flash device information

**Objective**: Gather information from the ESXi host about the storage I/O controller and flash device so that supportability can be checked.

Verifying that the storage I/O controller and flash devices are supported is extremely important. This can be achieved on a per host basis by running some of the commands available in the ESXCLI, and then checking the information against the VCG.

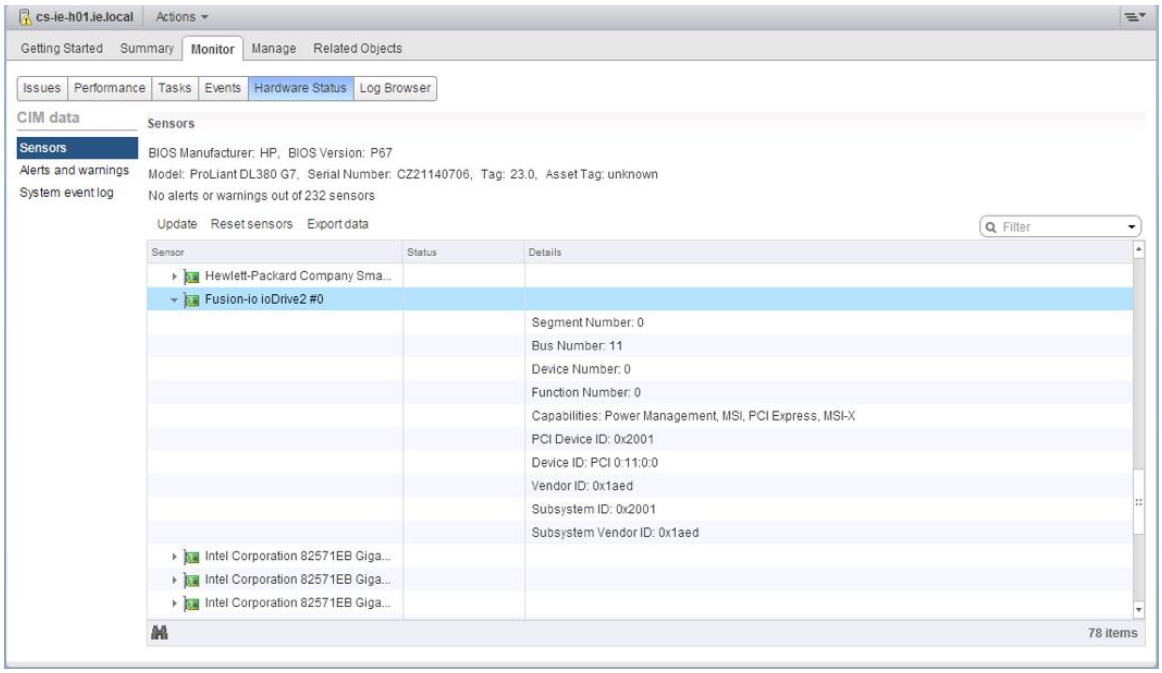
Note that this checking of storage I/O controller and flash devices should be done proactively on a continual basis as partners update drivers and firmware to address issues in their respective controllers. An administrator should check the configuration details at initial system build, and also when new components are added to the cluster. As the VCG is regularly updated, the administrator should regularly check for newer versions as these may contain new drivers and firmware versions that include fixes for bugs and performance.

In this section of the troubleshooting reference manual, an example walk-through is shown where the storage I/O controller as well as flash and disk devices are checked, along with vendor, device driver and firmware information. In part 2, this information is checked against the VCG to make sure that the configuration is supported.

### Using the vSphere web client UI to capture device info

In the host hardware status screen-shot shown below, taken from the vSphere web client, there is a Fusion-IO ioDrive2 PCI-E flash device. Using the Monitor and Hardware Status tab, the PCI can be expanded and the specific device may be located under PCI.

In this case the device is Fusion-io ioDrive2 and when the ioDrive2 device is then expanded, information such as the Device ID, Vendor ID, and other useful information can be gleaned. This can be very helpful in verifying if the device is supported for use with Virtual SAN, although in this case the firmware of the device is not visible. This must be retrieved via the command line.




If there are many devices in the list, it may not be easy to navigate. The filter option can help if you type in part of the name of the device. Fortunately, there are other ways of capturing device information other than the vSphere web client. Let's begin by looking at per host information.

## Using ESXCLI to capture device info

Since not everything is visible in the UI, there are options to capture the information in other ways. ESXCLI commands will give most of what is required to verify that a storage controller model is supported, as well as the driver and firmware versions. It can also be used to identify a flash device, and verify that it is also supported.

## esxcli storage core device list

Please note that while VSAN is distributed across all nodes in the cluster, the esxcli storage commands are per-host, and do not have a global view of disks. This command is run from the CLI of an ESXi 5.5 host (there is additional detail in the 6.0 version of the command which will be looked at shortly). The command displays information about a disk drive.

```
~ # esxcli storage core device list
 naa.600508b1001c4b820b4d80f9f8acfa95
    Display Name: HP Serial Attached SCSI Disk
(naa.600508b1001c4b820b4d80f9f8acfa95)
    Has Settable Display Name: true
    Size: 139979
    Device Type: Direct-Access
    Multipath Plugin: NMP
    Devfs Path: /vmfs/devices/disks/naa.600508b1001c4b820b4d80f9f8acfa95
    Vendor: HP
    Model: LOGICAL VOLUME
```

```
   Revision: 3.66
   SCSI Level: 5
   Is Pseudo: false
   Status: degraded
   Is RDM Capable: true
   Is Local: false
   Is Removable: false
   Is SSD: false
   Is Offline: false
   Is Perennially Reserved: false
   Queue Full Sample Size: 0
   Queue Full Threshold: 0
   Thin Provisioning Status: unknown
   Attached Filters:
   VAAI Status: unknown
   Other UIDs: vml.0200060000600508b1001c4b820b4d80f9f8acfa954c4f47494341
   Is Local SAS Device: false
   Is USB: false
   Is Boot USB Device: false
   No of outstanding IOs with competing worlds: 32

eui.a15eb52c6f4043b5002471c7886acfaa
   Display Name: Local FUSIONIO Disk (eui.a15eb52c6f4043b5002471c7886acfaa)
   Has Settable Display Name: true
   Size: 1149177
   Device Type: Direct-Access
   Multipath Plugin: NMP
   Devfs Path: /vmfs/devices/disks/eui.a15eb52c6f4043b5002471c7886acfaa
   Vendor: FUSIONIO
   Model: IODRIVE
   Revision: v1.0
   SCSI Level: 5
   Is Pseudo: false
   Status: on
   Is RDM Capable: false
   Is Local: true
   Is Removable: false
   Is SSD: true
   Is Offline: false
   Is Perennially Reserved: false
   Queue Full Sample Size: 0
   Queue Full Threshold: 0
   Thin Provisioning Status: yes
   Attached Filters:
   VAAI Status: unknown
   Other UIDs: vml.0100000000313231304430393235494f44524956
   Is Local SAS Device: false
   Is USB: false
   Is Boot USB Device: false
   No of outstanding IOs with competing worlds: 32
```

There is a lot of useful information to be gleaned from this output. One can see that the magnetic disks (Is SSD: false) seems to be presented to the ESXi host from a HP controller (Vendor: HP), and we can see the flash device (Is SSD: true) seems to be presented from a Fusion-IO controller.

Note that the status of the magnetic disk shown as *degraded* in the above output. The reason for this is because there is only a single path to the device. If there were multiple paths, then this “degraded" state is not present. This is not a concern for local disks, as in almost all local disk configurations, there will only be one path to the devices.

## Pass-through or RAID-0

"Logical Volume" in the Model field of the previous ESXCLI output implies that there is a RAID Volume configuration on the disk, most likely a RAID-0. There is a reason why a device would need to be configured as RAID-0. Certain storage controllers can pass disk devices directly to the ESXi host – this is what we term pass-through mode. Other storage controllers require each disk device to be configured in a RAID-0 volume before an ESXi host can recognize them; in other words they do not support pass-through mode. Shortly the benefits of one configuration over another will be discussed.

To determine if a controller supports either pass-through mode or RAID-0 mode (or indeed both), refer to the VCG. Here is an example of various HP storage I/O controllers on the VCG, and which mode they support.

**Search Results:** Your search for " Virtual SAN IO Controller " returned **7 results.** Back to Top Turn Off Auto Scroll Display: 10

| Brand Name | Model | Feature | Product Description | Queue Depth | Supported Releases |
|---|---|---|---|---|---|
| HP | H220 | Virtual SAN Pass-Through | **Device Type:** SAS **VID:** 1000 **SVID:** 1590 **DID:** 0087 **SSID:** 0041 | 600 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | H220i | Virtual SAN Pass-Through | **Device Type:** SAS **VID:** 1000 **SVID:** 1590 **DID:** 0087 **SSID:** 0044 | 600 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | H222 | Virtual SAN Pass-Through | **Device Type:** SAS **VID:** 1000 **SVID:** 1590 **DID:** 0087 **SSID:** 0043 | 600 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | Smart Array P220i | Virtual SAN RAID 0 | **Device Type:** SAS **VID:** 103C **SVID:** 103c **DID:** 323B **SSID:** 3355 | 1020 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | Smart Array P420 | Virtual SAN RAID 0 | **Device Type:** SAS **VID:** 103C **SVID:** 103c **DID:** 323B **SSID:** 3351 | 1020 | ESXi 5.5 U2 ESXi 5.5 U1 |

This next ESXCLI command will display further information about the adapters/controllers. Note that this will publish all adapters, including network adapters, not just storage I/O controllers.

## esxcli core storage adapter list

```
 ~ # esxcli storage core adapter list
 HBA Name  Driver        Link State  UID                   Description
 --------  ------------  ----------  --------------------  ------------------------------
------------------------------
 vmhba0    ata_piix      link-n/a    sata.vmhba0           (0:0:31.2) Intel Corporation
ICH10 4 port SATA IDE Controller
 vmhba1    hpsa          link-n/a    sas.5001438013ebe0d0  (0:5:0.0) Hewlett-Packard
Company Smart Array P410i
 vmhba32   usb-storage   link-n/a    usb.vmhba32           () USB
 vmhba33   bnx2i         unbound     iscsi.vmhba33         Broadcom iSCSI Adapter
 vmhba34   bnx2i         unbound     iscsi.vmhba34         Broadcom iSCSI Adapter
 vmhba35   bnx2i         unbound     iscsi.vmhba35         Broadcom iSCSI Adapter
 vmhba36   bnx2i         unbound     iscsi.vmhba36         Broadcom iSCSI Adapter
 fioiom0   iomemory-vsl  link-n/a    unknown.fioiom0       (0:11:0.0) Fusion-io ioDrive2
 vmhba37   ata_piix      link-n/a    sata.vmhba37          (0:0:31.2) Intel Corporation
ICH10 4 port SATA IDE Controller
 ~ #
```

This information displayed above can also be observed by using the following command:

### esxcfg-scsidevs –a

```
~ # esxcfg-scsidevs -a
 vmhba0  ata_piix     link-n/a  sata.vmhba0            (0:0:31.2) Intel Corporation ICH10
4 port SATA IDE Controller
 vmhba1   hpsa          link-n/a   sas.5001438013ebe0d0 (0:5:0.0) Hewlett-Packard Company
Smart Array P410i
 vmhba32 usb-storage    link-n/a  usb.vmhba32           () USB
 vmhba33 bnx2i          unbound   iscsi.vmhba33         Broadcom iSCSI Adapter
 vmhba34 bnx2i          unbound   iscsi.vmhba34         Broadcom iSCSI Adapter
 vmhba35 bnx2i          unbound   iscsi.vmhba35         Broadcom iSCSI Adapter
 vmhba36 bnx2i          unbound   iscsi.vmhba36         Broadcom iSCSI Adapter
 fioiom0 iomemory-vsl   link-n/a  unknown.fioiom0       (0:11:0.0) Fusion-io ioDrive2
 vmhba37 ata_piix    link-n/a sata.vmhba37             (0:0:31.2) Intel Corporation ICH10
4 port SATA IDE Controller
 ~ #
```

The commands include good descriptions of the adapters to allow administrators to determine exactly which storage controllers and flash devices are running on a host. The storage controller is a **HP Smart Array P410i**, and the PCI-E flash device is a **Fusion-IO ioDrive2**. The driver used by the adapters is also shown.

***Caution**: Note that this HP Smart Array 410i is shown for illustration and troubleshooting purposes only. This controller from HP is not on the Virtual SAN VCG, as we shall see shortly.*

Note also that the description field contains entries such as (0:5:0.0) and (0:11:0.0). This refers to the adapter PCI slot location and this information can be used later with some additional commands to verify that this is the adapter used by Virtual SAN.

### Handling multiple controllers

At this point, using the previous ESXCLI command, a list of controllers and disk drives has been found. However, if there were multiple SCSI controllers on the host, how would one determine which disk was associated with which controller? This next ESXCLI command allows us to do just that.

### esxcli storage core path list

Note that the adapter is listed in the output of each device:

```
~ # esxcli storage core path list

 unknown.fioiom0-unknown.0:0-eui.a15eb52c6f4043b5002471c7886acfaa
 UID: unknown.fioiom0-unknown.0:0-eui.a15eb52c6f4043b5002471c7886acfaa
    Runtime Name: fioiom0:C0:T0:L0
    Device: eui.a15eb52c6f4043b5002471c7886acfaa
  Device Display Name: Local FUSIONIO Disk (eui.a15eb52c6f4043b5002471c7886acfaa)
    Adapter: fioiom0
```

```
    Channel: 0
    Target: 0
    LUN: 0
    Plugin: NMP
    State: active
    Transport: parallel
    Adapter Identifier: unknown.fioiom0
    Target Identifier: unknown.0:0
    Adapter Transport Details: Unavailable or path is unclaimed
    Target Transport Details: Unavailable or path is unclaimed
    Maximum IO Size: 1048576

 sas.5001438013ebe0d0-sas.1438013ebe0d0-naa.600508b1001c258181f0a088f6e40dab
UID:sas.5001438013ebe0d0-sas.1438013ebe0d0-naa.600508b1001c258181f0a088f6e40dab
    Runtime Name: vmhba1:C0:T0:L1
    Device: naa.600508b1001c258181f0a088f6e40dab
    Device      Display      Name:      HP      Serial      Attached      SCSI      Disk
(naa.600508b1001c258181f0a088f6e40dab)
    Adapter: vmhba1
    Channel: 0
    Target: 0
    LUN: 1
    Plugin: NMP
    State: active
    Transport: sas
    Adapter Identifier: sas.5001438013ebe0d0
    Target Identifier: sas.1438013ebe0d0
    Adapter Transport Details: 5001438013ebe0d0
    Target Transport Details: 1438013ebe0d0
    Maximum IO Size: 4194304
```

There is a lot of information included in this command output. However there is a Fusion-IO adapter called **fioiom0** and there is a HP adapter called **vmhba1**. Possibly a better way to display this information, one line at a time, is to use `esxcfg-scsidevs –A`. The command simply displays which device is attached to which controller, without all of the other ancillary information shown above.

### esxcfg-scsidevs –A

```
~ # esxcfg-scsidevs -A
 fioiom0     eui.a15eb52c6f4043b5002471c7886acfaa
 vmhba0      mpx.vmhba0:C0:T0:L0
 vmhba1      naa.600508b1001c4b820b4d80f9f8acfa95
 vmhba1      naa.600508b1001c4d41121b41182fa83be4
 vmhba1      naa.600508b1001c846c000c3d9114ed71b3
 vmhba1      naa.600508b1001c258181f0a088f6e40dab
 vmhba1      naa.600508b1001cc426a15528d121bbd880
 vmhba1      naa.600508b1001c51f3a696fe0bbbcb5096
 vmhba1      naa.600508b1001cadff5d80ba7665b8f09a
 vmhba1      naa.600508b1001cd0fe479b5e81ca7ec77e
 vmhba32     mpx.vmhba32:C0:T0:L0
 ~ #
```

This displays which adapter a particular SCSI device is attached to. Remember that the Fusion-IO adapter is called **fioiom0** and that the HP adapter is **vmhba1**, so here we can see the disks associated with each adapter. Either a EUI identifier, an MPX identifier or an NAA identifier, identifies the SCSI devices. Different SCSI devices from different vendors identify themselves in different ways, although all are supported. The Fusion-IO is using EUI, whereas the HP devices use an NAA identifier.

### A note about SCSI identifiers

NAA stands for Network Addressing Authority identifier. EUI stands for Extended Unique Identifier. The number is guaranteed to be unique to that LUN. The NAA or EUI identifier is the preferred method of identifying LUNs and the storage device generates the number. Since the NAA or EUI is unique to the LUN, if the LUN is presented the same way across all ESXi hosts, the NAA or EUI identifier remains the same. Some devices do not provide the NAA number described above. In these circumstances, an MPX Identifier is generated by ESXi to represent the LUN or disk. The identifier takes the form similar to that of the canonical name of previous versions of ESXi with the "mpx." prefix. This identifier can be used in the exact same way as the NAA Identifier described above." Refer to VMware KB article 1014953 for further information.

### Displaying disk drive information

At this point the controllers/adapters have been identified. If more information regarding the disks drives is required, this next command lists all logical devices on the ESXi host on a single line, with limited information:

### esxcfg-scsidevs –c

```
~ # esxcfg-scsidevs -c
 Device  UID                                                  Device  Type         Console  Device
Size      Multipath PluginDisplay Name
 eui.a15eb52c6f4043b5002471c7886acfaa                                          Direct-Access
/vmfs/devices/disks/eui.a15eb52c6f4043b5002471c7886acfaa    1149177MB  NMP          Local
FUSIONIO Disk (eui.a15eb52c6f4043b5002471c7886acfaa)
 mpx.vmhba0:C0:T0:L0                                                                CD-ROM
/vmfs/devices/cdrom/mpx.vmhba0:C0:T0:L0                          0MB    NMP     Local hp CD-
ROM (mpx.vmhba0:C0:T0:L0)
 mpx.vmhba32:C0:T0:L0                                                          Direct-Access
/vmfs/devices/disks/mpx.vmhba32:C0:T0:L0                      1917MB     NMP     Local USB
Direct-Access (mpx.vmhba32:C0:T0:L0)
 naa.600508b1001c258181f0a088f6e40dab                                          Direct-Access
/vmfs/devices/disks/naa.600508b1001c258181f0a088f6e40dab   139979MB   NMP        HP  Serial
Attached SCSI Disk (naa.600508b1001c258181f0a088f6e40dab)
 naa.600508b1001c4b820b4d80f9f8acfa95                                          Direct-Access
/vmfs/devices/disks/naa.600508b1001c4b820b4d80f9f8acfa95   139979MB   NMP        HP  Serial
Attached SCSI Disk (naa.600508b1001c4b820b4d80f9f8acfa95)
 naa.600508b1001c4d41121b41182fa83be4                                          Direct-Access
/vmfs/devices/disks/naa.600508b1001c4d41121b41182fa83be4   139979MB   NMP        HP  Serial
Attached SCSI Disk (naa.600508b1001c4d41121b41182fa83be4)
 naa.600508b1001c51f3a696fe0bbbcb5096                                          Direct-Access
/vmfs/devices/disks/naa.600508b1001c51f3a696fe0bbbcb5096   139979MB   NMP        HP  Serial
Attached SCSI Disk (naa.600508b1001c51f3a696fe0bbbcb5096)
 naa.600508b1001c846c000c3d9114ed71b3                                          Direct-Access
/vmfs/devices/disks/naa.600508b1001c846c000c3d9114ed71b3   139979MB   NMP        HP  Serial
Attached SCSI Disk (naa.600508b1001c846c000c3d9114ed71b3)
 naa.600508b1001cadff5d80ba7665b8f09a                                          Direct-Access
/vmfs/devices/disks/naa.600508b1001cadff5d80ba7665b8f09a   139979MB   NMP        HP  Serial
Attached SCSI Disk (naa.600508b1001cadff5d80ba7665b8f09a)
~ #
```

This is a list of all the SCSI devices on this ESXi host, including their sizes.

## Using ESXCLI to capture storage controller info

Further information about the controllers/adapters, in particular the PCI identifiers, can be extremely useful when trying to figure out if the device is supported or not since this information can be used to search the VMware Compatibility Guide. This is useful in the case where a storage controller has been rebranded, as shall be seen shortly.

## esxcli hardware pci list

The first line of the PCI list output shows the PCI slot information. This can be used to ensure that the correct adapter is being queried.

Once the correct adapter is confirmed, the Vendor IDs, Device IDs, Sub-Vendor IDs and Sub-Device IDs can be used to verify that this adapter is on the VMware Compatibility Guide. Some vendors rebrand devices to reflect their own particular range of products; however the vendor and device ids remain the same. Therefore it is not uncommon to find that one vendor's controller is not listed under the vendor name in the VCG, but in fact it appears in the VCG listed under a different vendor.

```
 ~ # esxcli hardware pci list

<<truncated>>

 000:005:00.0
    Address: 000:005:00.0
    Segment: 0x0000
    Bus: 0x05
    Slot: 0x00
    Function: 0x00
    VMkernel Name: vmhba1
    Vendor Name: Hewlett-Packard Company
    Device Name: Smart Array P410i
    Configured Owner: Unknown
    Current Owner: VMkernel
    Vendor ID: 0x103c
    Device ID: 0x323a
    SubVendor ID: 0x103c
    SubDevice ID: 0x3245
    Device Class: 0x0104
    Device Class Name: RAID bus controller
    Programming Interface: 0x00
    Revision ID: 0x01
    Interrupt Line: 0x0a
    IRQ: 10
    Interrupt Vector: 0x2c
    PCI Pin: 0xa8
    Spawned Bus: 0x00
    Flags: 0x0201
    Module ID: 4177
    Module Name: hpsa
    Chassis: 0
    Physical Slot: 255
    Slot Description:
    Passthru Capable: true
    Parent Device: PCI 0:0:1:0
    Dependent Device: PCI 0:5:0:0
    Reset Method: Link reset
    FPT Sharable: true
```

```
000:00b:00.0
   Address: 000:00b:00.0
   Segment: 0x0000
   Bus: 0x0b
   Slot: 0x00
   Function: 0x00
   VMkernel Name: fioiom0
   Vendor Name: Fusion-io
   Device Name: ioDrive2
   Configured Owner: Unknown
   Current Owner: VMkernel
   Vendor ID: 0x1aed
   Device ID: 0x2001
   SubVendor ID: 0x1aed
   SubDevice ID: 0x2001
   Device Class: 0x0180
   Device Class Name: Mass storage controller
   Programming Interface: 0x00
   Revision ID: 0x04
   Interrupt Line: 0x0a
   IRQ: 10
   Interrupt Vector: 0x5d
   PCI Pin: 0x30
   Spawned Bus: 0x00
   Flags: 0x0201
   Module ID: 4178
   Module Name: iomemory-vsl
   Chassis: 0
   Physical Slot: 2
   Slot Description: PCI-E Slot 2
   Passthru Capable: true
   Parent Device: PCI 0:0:8:0
   Dependent Device: PCI 0:11:0:0
   Reset Method: Link reset
   FPT Sharable: true

<<truncated>>
```

As well as PCI Address, there is the vendor id, device id, sub-vendor id and sub-device id of the PCI adapter. If this is a rebranded device, this is the information required to verify that it is supported via the VCG.

The other useful piece of information from this output is the device driver, referred to as the "Module Name" above. For HP, the driver is the hpsa; for Fusion-IO it is the iomemory-vsl.

Now that the adapter information and driver information has been retrieved, the next step is to check that the device driver version that is running on the host is supported. The supported version of a driver may change as issues are discovered, and the driver version may need to be updated for reasons of reliability or performance or both.

The driver names are already visible in the previous adapter list output, but it does not have the version. One command, `vmkload_mod -s`, can provide this information.

### vmkload_mod –s

```
~ # vmkload_mod -s hpsa
```

```
 vmkload_mod module information
  input file: /usr/lib/vmware/vmkmod/hpsa
  License: GPL
  Version: Version 5.5.0.74-1OEM, Build: 1331820, Interface: 9.2 Built on: Aug  5 2014
```

This is the driver from the vSphere 5.5U2 HP ISO image (OEM – Original Equipment Manufacturer). This ISO image bundles an async driver for the HP Smart Array (async meaning that it is a driver not included with the official ESXi builds; those drivers are referred to as inbox drivers).

A more elegant way of querying a driver version is via `esxcli system module get –m`. Here is an example taken from an ESXi 6.0 host, not from our walk-through.

### esxcli system module get -m

```
esxcli system module get -m hpsa
   Module: hpsa
   Module File: /usr/lib/vmware/vmkmod/hpsa
   License: GPL
   Version: Version 6.0.0.44-4vmw, Build: 2391873, Interface: 9.2 Built on: Dec 27 2014
   Build Type: release
   Provided Namespaces:
   Required Namespaces: com.vmware.driverAPI@9.2.3.0, com.vmware.vmkapi@v2_3_0_0
   Containing VIB: scsi-hpsa
   VIB Acceptance Level: certified
```

Next the FusionIO driver, also displayed in the pci list command previously, is checked.

```
~ # vmkload_mod -s iomemory-vsl
 vmkload_mod module information
  input file: /usr/lib/vmware/vmkmod/iomemory-vsl
  Version: Version 3.2.8.1350, Build: 1198610, Interface: 9.2 Built on: May  8 2014
  License: proprietary
  Name-space: com.fusion-io.scsi-iomemory-vsl#9.2.2.0
```

***Warning**: At this point, it is once again imperative that you use a supported device driver. Failure to do so can result in degraded functionality or degraded performance or both. Check the VCG regularly for driver updates that contain patches and fixes.*

As shown, there are various ways of getting similar information from ESXi via the ESXCLI commands. The previous command reported driver version, but the following command provides another way of getting software versions, including those of the driver that have been installed on the ESXi host from third parties. As stated in the introduction to tools, there are many ways of doing the same thing with ESXCLI. This command is shown simply for illustrative purposes, so that there are other options available for gathering information:

### esxcli software vib list

```
[root@cs-ie-h01:~] esxcli software vib list
 Name              Version                             Vendor  Acceptance Level  Install Date
 ---------------  ----------------------------------  ------  ----------------  ------------
```

```
mtip32xx-native    3.8.5-1vmw.600.0.0.2159203      VMWARE  VMwareCertified   2014-11-07
ata-pata-amd       0.3.10-3vmw.600.0.0.2159203     VMware  VMwareCertified   2014-11-07
ata-pata-atiixp    0.4.6-4vmw.600.0.0.2159203      VMware  VMwareCertified   2014-11-07
ata-pata-cmd64x    0.2.5-3vmw.600.0.0.2159203      VMware  VMwareCertified   2014-11-07
ata-pata-hpt3x2n   0.3.4-3vmw.600.0.0.2159203      VMware  VMwareCertified   2014-11-07
ata-pata-pdc2027x  1.0-3vmw.600.0.0.2159203        VMware  VMwareCertified   2014-11-07
<<truncated>>
```

Note that the output above states “VMwareCertified”. This does not imply that the driver is “Virtual SAN certified”.

***Warning**: Care must be exercised if using a “custom ISO” or a “rollup ISO”. Customers may end up in a situation where the driver version found on the ISO is not yet on the VCG (in other words, customers have upgraded from a supported version to an unsupported version of the driver). Customers must pay special attention when faced with these upgrade situations.*

## Using fio-status (Fusion-IO command) to check settings

The final thing to check is the firmware version of the Fusion-IO device. There does not appear to be an ESXCLI command to display this information. Fortunately Fusion-IO has provided such a tool and it is installed on the ESXi host along with the other software components from Fusion-IO. This command provides driver version, firmware version and product number, which is what is listed on the VCG.

```
 ~ # fio-status

 Found 1 ioMemory device in this system
 Driver version: 3.2.8 build 1350

 Adapter: Single Controller Adapter
          Fusion-io ioDrive2 1.205TB, Product Number:F00-001-1T20-CS-0001, SN:1210D0925,
FIO SN:1210D0925
          External Power: NOT connected
          PCIe Power limit threshold: 24.75W
          Connected ioMemory modules:
            fct0: Product Number:F00-001-1T20-CS-0001, SN:1210D0925

 fct0     Attached
          ioDrive2 Adapter Controller, Product Number:F00-001-1T20-CS-0001, SN:1210D0925
          Located in slot 0 Center of ioDrive2 Adapter Controller SN:1210D0925
          PCI:0b:00.0
          Firmware v7.1.15, rev 110356 Public
          1205.00 GBytes device size
          Internal temperature: 55.12 degC, max 58.08 degC
          Reserve space status: Healthy; Reserves: 100.00%, warn at 10.00%
          Contained VSUs:
            fioiom0:      ID:0, UUID:a15eb52c-6f40-43b5-8a48-84c7886acfaa

 fioiom0 State: Online, Type: block device
          ID:0, UUID:a15eb52c-6f40-43b5-8a48-84c7886acfaa
          1205.00 GBytes device size
```

There is now enough information gathered (model, vendor, driver, firmware) that supportability might be checked on the VMware Compatibility Guide for Virtual SAN.

# Part 2 - Verify hardware support against VMware Compatibility Guide

**Objective**: Now that the information has been gathered, the next step is to navigate to the Virtual SAN VCG to verify that the information previously gathered from the controller and flash devices on the ESXi host is indeed supported by Virtual SAN. Once again here is the link to the VCG, and the Virtual SAN category:

http://www.vmware.com/resources/compatibility/search.php?deviceCategory=vsan

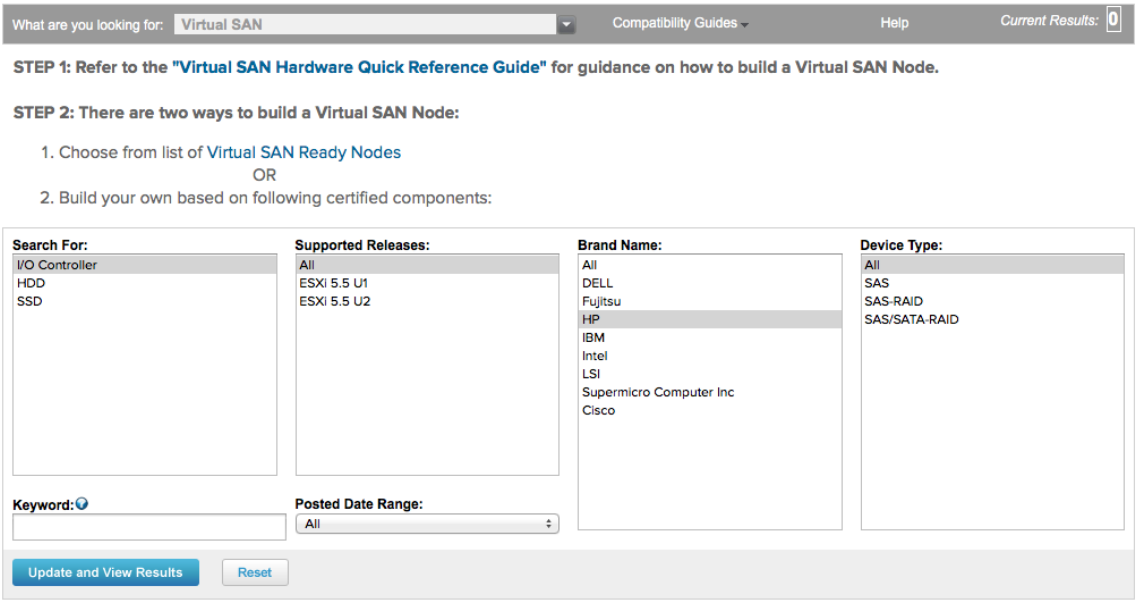
***Caution**: the examples presented here reflect the contents of the VCG at a specific point in time. The Virtual SAN entries in the VCG are constantly being updated. Use the VCG as your authoritative and accurate source.*

### Information gathered

| Device | Vendor | Model | Driver | Firmware |
|---|---|---|---|---|
| Storage I/O Controller | HP | HP 410i | 5.5.0.74-1OEM | N/A |
| PCI-E flash device | FusionIO | F00-001-1T20-CS-0001 | 3.2.8 build 1350 | v7.1.15 |

## Checking storage controller/adapter supportability

The first area of focus is whether or not the HP Smart Array 410i is on the VCG and if it is supported. From the VCG for Virtual SAN, select the I/O Controller, then HP as the brand name. This will search all releases and all device types.




Click on the update and view results to see a list of supported HP controllers:

Click on the 'Model' to view more details and to subscribe to RSS feeds.

Bookmark | Print | Export to CSV

**Search Results:** Your search for " Virtual SAN IO Controller " returned **7 results.** Back to Top Turn Off Auto Scroll Display: 10

| Brand Name | Model | Feature | Product Description | Queue Depth | Supported Releases |
|---|---|---|---|---|---|
| HP | Smart Array P822 | Virtual SAN RAID 0 | **Device Type:** SAS **VID:** 103c **SVID:** 103c **DID:** 323b **SSID:** 3353 | 1020 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | Smart Array P420i | Virtual SAN RAID 0 | **Device Type:** SAS **VID:** 103C **SVID:** 103c **DID:** 323B **SSID:** 3354 | 1020 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | Smart Array P420 | Virtual SAN RAID 0 | **Device Type:** SAS **VID:** 103C **SVID:** 103c **DID:** 323B **SSID:** 3351 | 1020 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | Smart Array P220i | Virtual SAN RAID 0 | **Device Type:** SAS **VID:** 103C **SVID:** 103c **DID:** 323B **SSID:** 3355 | 1020 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | H222 | Virtual SAN Pass-Through | **Device Type:** SAS **VID:** 1000 **SVID:** 1590 **DID:** 0087 **SSID:** 0043 | 600 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | H220i | Virtual SAN Pass-Through | **Device Type:** SAS **VID:** 1000 **SVID:** 1590 **DID:** 0087 **SSID:** 0044 | 600 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | H220 | Virtual SAN Pass-Through | **Device Type:** SAS **VID:** 1000 **SVID:** 1590 **DID:** 0087 **SSID:** 0041 | 600 | ESXi 5.5 U2 ESXi 5.5 U1 |

There is no entry for the HP Smart Array 410i, which implies that this is not a supported controller for Virtual SAN and therefore should not be used with Virtual SAN. However, this is the status right now – this may change in the future.

## Understanding RAID-0 versus pass-through

In the "Feature" column above, you can see references to RAID-0 and Pass-Through. (Pass-through is sometimes referred to as JBOD or HBA mode).

This is a reference to how controllers present their magnetic disks and flash devices (SSD) to ESXi hosts. It means that certain controllers are only supported when they present their disks directly to the ESXi hosts participating in Virtual SAN. Other controllers are only supported when they present each disk to the ESXi host as a RAID-0 volume. Some controllers can support both pass-through and RAID-0 volumes. Pass-through is preferred.

In certain cases, ESXi cannot detect disks on a controller unless they are placed in pass-through mode. Configuring each disk as a "RAID-0" group of one allows them to be detected by Virtual SAN.

***Caution**: If a controller is supported in RAID-0 mode only, and the disks are configured in pass-through mode, some disks might encounter permanent errors frequently which results in VM objects going degraded. For the configuration to be supported, their disks need to be reconfigured as RAID-0. Therefore if the VCG says use RAID-0, use RAID-0 and not pass-through, even though it might appear to work. This misconfiguration and failure to adhere to the VCG has led to a number of escalations in the past. Ensure this is checked before deployment. This will have to be done via the BIOS of the card, or via third party tools.*

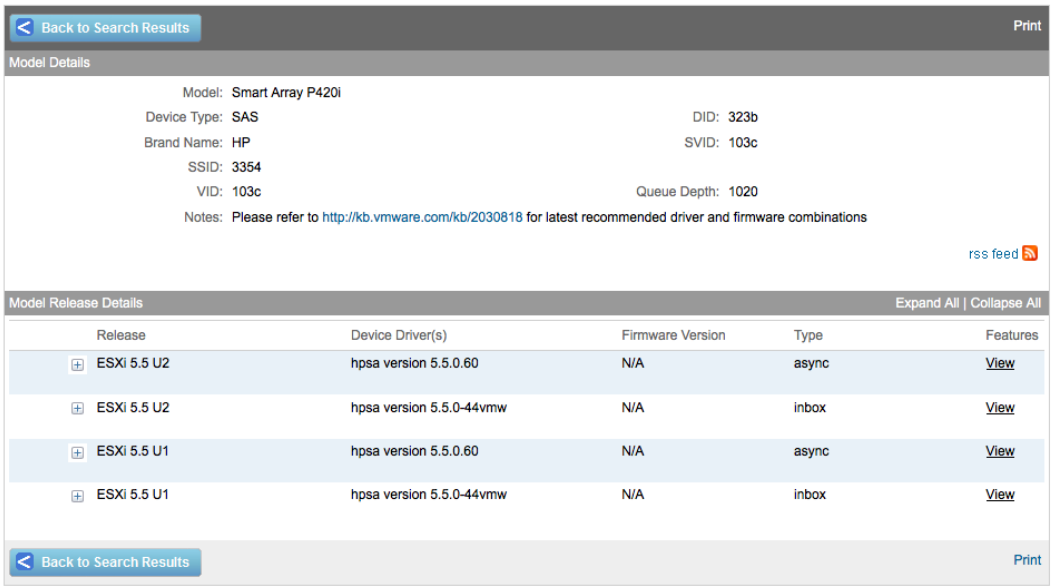
Also note that pass-through is preferred over RAID-0 for ease of disk drive replacement. When a drive needs replacing, if it is in pass-through mode, it is a simple matter of plug and replace. When RAID-0 is configured on the drive, this is an additional step that needs to be carried out on the new disk before the drive can be fully replaced.

Another caveat of using RAID 0 is that the disk type and SMART (Self-Monitoring, Analysis and Reporting Technology) information is not passed to the ESXi host because the controller blocks it.

## Checking storage controller/adapter driver and firmware

Although the P410i is not support with Virtual SAN, it is worth continuing on with the walk-though to check out the rest of the configuration. If the controller used in the configuration had been validated against the VCG, and shows up supported, the next step is to check if the driver version is supported.

In the VCG, click on the adapter model which in this example is the HP Smart Array 420i (since the 410i is not on the VCG). Navigate to the screen that lists the driver versions. Using the `vmkload_mod -s` command earlier, the driver version of the *hpsa* on the setup is **`5.5.0.74-1OEM`**. This is an async driver from the vSphere 5.5U2 HP ISO image.




The version of the *hpsa* driver (5.5.0.74-OEM) on the hosts is unsupported. The only supported driver versions, both inbox and async, do not include this version of the driver.

In this sample walk-through configuration, both the controller (410i) and driver (5.5.0.74-OEM) are unsupported for use with Virtual SAN. Neither should be used with Virtual SAN.

## A note on OEM ESXi ISO images

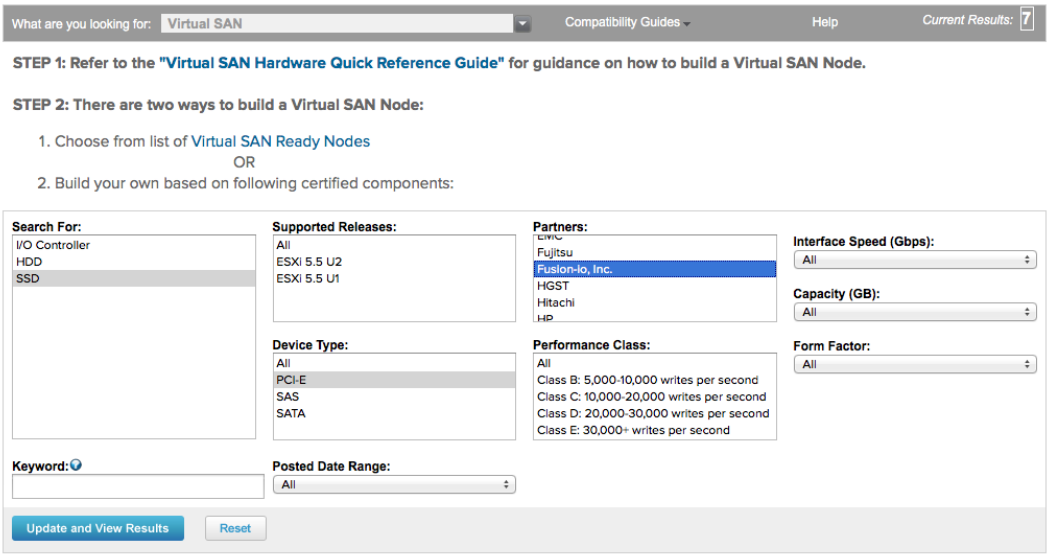
The ESXi ISO image from HP was used to deploy the ESXi hosts. However that image contained drivers that were not supported by Virtual SAN.

Other Equipment Manufacturers (OEMs), through the use of these OEM images, can provide new drivers that they have qualified for general ESXi operations. However, not all OEMs will carry out the storage controller qual/test/cert suite for Virtual SAN. Therefore, the OEM ESXi ISO images may well contain storage controller drivers that have never been certified on Virtual SAN. Use of these non-certified drivers can result in unpredictable behavior on Virtual SAN.

As VMware's OEM partners ramp up their own testing capabilities, the OEM ESXi ISOs start appearing with Virtual SAN-qualified drivers, etc. This will begin in 2015, but will be an on-going process. Until that happens, please continue to use the VCG for Virtual SAN supported drivers.

## Checking a Fusion-IO flash device: model

This environment is using a Fusion-IO PCI-E flash device. The next step is to check if it is supported. Although it is a PCI-E flash device, one must search on SSD in the VCG and select PCI-E as the device type. Finally select Fusion-IO as the partner as shown below.




Fusion-IO has a number of different models of PCI-E flash cards on the VCG, including "ioFX" and "ioScale". In the `esxcli hardware pci list` and the `esxcli storage adapters list` commands that we ran earlier, the device was reported as an "iodrive2", and that is what the search is focused on. To narrow the list of devices returned by the search, place `iodrive2` as a keyword and update the results. This still provides us with a number of supported devices, although interestingly, they are all of different sizes.

The ESXCLI command outputs gathered previously showed that the PCI-E card in question was approximately 1200GB in size. This information can be used to narrow the search criteria.

**Click on the 'Model' to view more details and to subscribe to RSS feeds.**

Bookmark | Print | Export to CSV

**Search Results:** Your search for " Virtual SAN SSD " returned **8 results.** Back to Top Turn Off Auto Scroll Display: 10

| Partner Name | Model | Product Description | Supported Releases |
|---|---|---|---|
| Fusion-io, Inc. | ioDrive2 Duo F01-001-1T20-DS-0001 | **Device Type:**PCI-E<br>**Capacity:** 1200GB<br>**Endurance Class:** Class D: 20+ DWPD<br>**Performance Class:** Class E: 30,000+ writes per second<br>**Flash Technology:** SLC SSD | ESXi 5.5 U2<br>ESXi 5.5 U1 |
| Fusion-io, Inc. | ioDrive2 Duo F01-001-2T41-CS-0001 | **Device Type:**PCI-E<br>**Capacity:** 2410GB<br>**Endurance Class:** Class B: 5-10 DWPD<br>**Performance Class:** Class E: 30,000+ writes per second<br>**Flash Technology:** MLC SSD | ESXi 5.5 U2<br>ESXi 5.5 U1 |
| Fusion-io, Inc. | ioDrive2 F00-001-1T20-CS-0001 | **Device Type:**PCI-E<br>**Capacity:** 1205GB<br>**Endurance Class:** Class B: 5-10 DWPD<br>**Performance Class:** Class E: 30,000+ writes per second<br>**Flash Technology:** MLC SSD | ESXi 5.5 U2<br>ESXi 5.5 U1 |
| Fusion-io, Inc. | ioDrive2 F00-001-365G-CS-0001 | **Device Type:**PCI-E<br>**Capacity:** 365GB<br>**Endurance Class:** Class B: 5-10 DWPD<br>**Performance Class:** Class E: 30,000+ writes per second<br>**Flash Technology:** MLC SSD | ESXi 5.5 U2<br>ESXi 5.5 U1 |
| Fusion-io, Inc. | ioDrive2 F00-001-400G-DS-0001 | **Device Type:**PCI-E<br>**Capacity:** 400GB<br>**Endurance Class:** Class D: 20+ DWPD<br>**Performance Class:** Class E: 30,000+ writes per second<br>**Flash Technology:** SLC SSD | ESXi 5.5 U2<br>ESXi 5.5 U1 |
| Fusion-io, Inc. | ioDrive2 F00-001-600G-DS-0001 | **Device Type:**PCI-E<br>**Capacity:** 600GB<br>**Endurance Class:** Class D: 20+ DWPD<br>**Performance Class:** Class E: 30,000+ writes per second<br>**Flash Technology:** SLC SSD | ESXi 5.5 U2<br>ESXi 5.5 U1 |

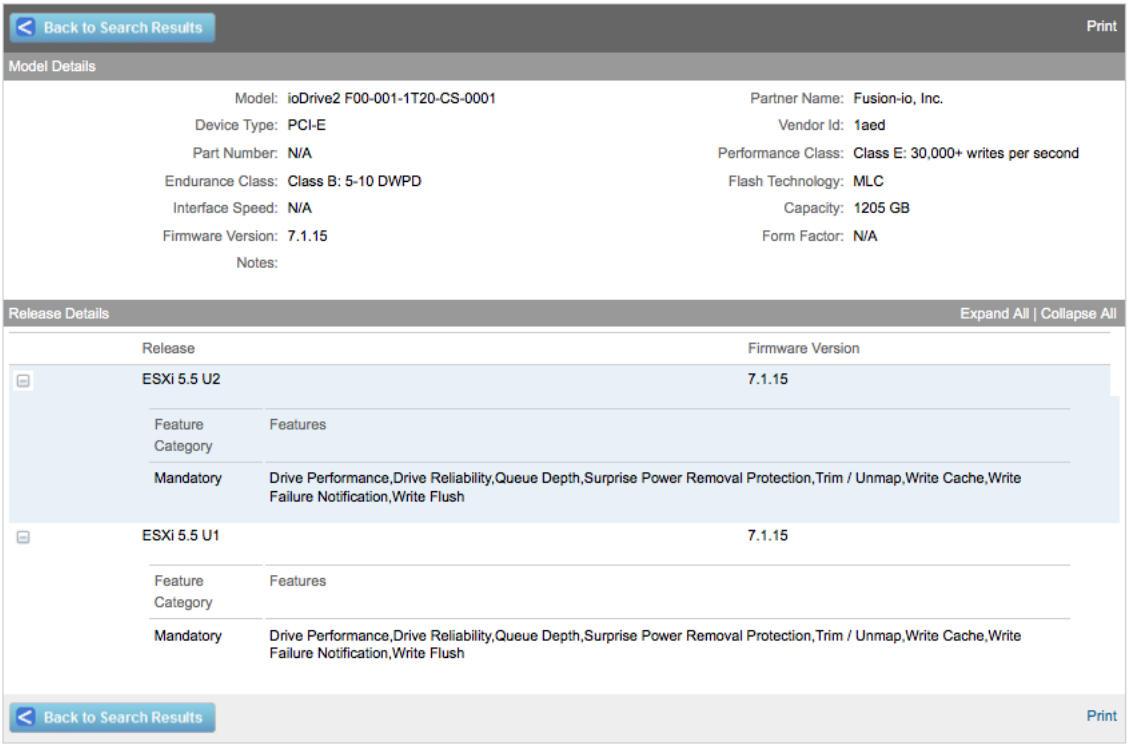
The *F00-001-1T20-CS-001* model appears to be approximately the same size as the device (1205GB) on the host.

Of course a better way to identify the flash device other than by capacity is to use specific Fusion-IO commands. These commands display adapter model, as well as driver and firmware versions. The `fio-status -a` command revealed that this was indeed the *F00-001-1T20-CS-001* model.

The Fusion-IO command `fio-status -a` displays even more information, such as whether or not the cache is protected during a power down, and what is the expected lifespan/endurance of the card.

## Checking a Fusion-IO flash device: firmware

Clicking on the model in the above output takes us to the models details view:



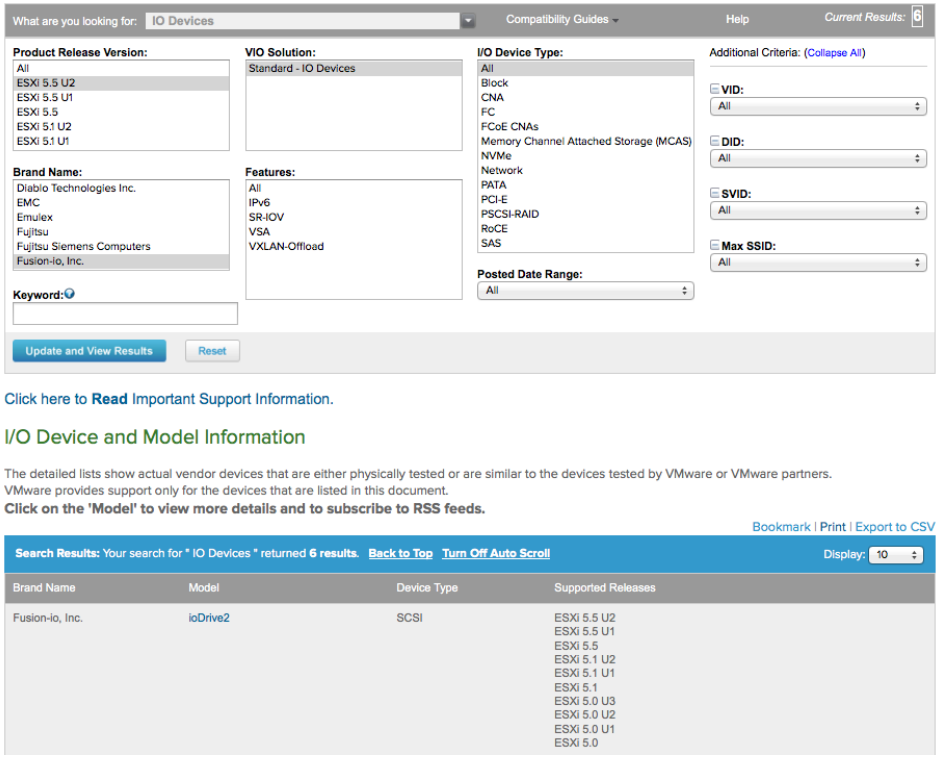

From this details screen, the PCI Vendor ID information as well as the firmware information can be checked. The vendor id certainly matches the Fusion-IO Vendor ID, and the firmware version of the card is displayed, but there is no driver version.

Highlighted above is the firmware version above and as can be clearly seen, the VCG is recommending the same version as the host is running - **`7.1.15`**.

If there was an outdated version of the firmware on the PCI-E flash device. The vendor, in this case Fusion-IO, should be consulted for the location of the latest firmware, and the vendor documentation should be consulted for steps on how to upgrade the firmware on the PCI-E card.

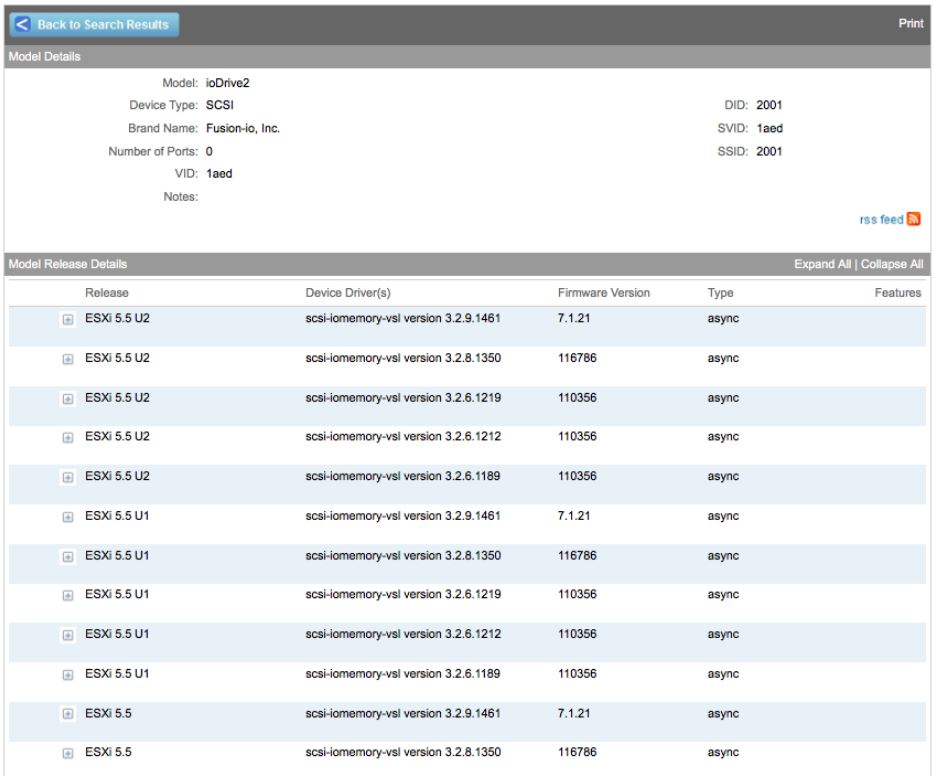
## Checking a Fusion-IO flash device: driver

A different part of the VCG must be navigated to in order to find the driver version. This means searching the VCG one more time, but this time you need to select **I/O Devices** rather than Virtual SAN. Select "Fusion-IO Inc." as the brand name and update the results.




The ioDrive2 model listed.

To get the supported driver versions, click on the "ioDrive2" link in the Model column. This will display the list of supported drivers.



Back to Search Results — Print

Model Details

Model: ioDrive2
Device Type: SCSI
Brand Name: Fusion-io, Inc.
Number of Ports: 0
VID: 1aed
Notes:
DID: 2001
SVID: 1aed
SSID: 2001

rss feed

Model Release Details — Expand All | Collapse All

| Release | Device Driver(s) | Firmware Version | Type | Features |
|---|---|---|---|---|
| ESXi 5.5 U2 | scsi-iomemory-vsl version 3.2.9.1461 | 7.1.21 | async | |
| ESXi 5.5 U2 | scsi-iomemory-vsl version 3.2.8.1350 | 116786 | async | |
| ESXi 5.5 U2 | scsi-iomemory-vsl version 3.2.6.1219 | 110356 | async | |
| ESXi 5.5 U2 | scsi-iomemory-vsl version 3.2.6.1212 | 110356 | async | |
| ESXi 5.5 U2 | scsi-iomemory-vsl version 3.2.6.1189 | 110356 | async | |
| ESXi 5.5 U1 | scsi-iomemory-vsl version 3.2.9.1461 | 7.1.21 | async | |
| ESXi 5.5 U1 | scsi-iomemory-vsl version 3.2.8.1350 | 116786 | async | |
| ESXi 5.5 U1 | scsi-iomemory-vsl version 3.2.6.1219 | 110356 | async | |
| ESXi 5.5 U1 | scsi-iomemory-vsl version 3.2.6.1212 | 110356 | async | |
| ESXi 5.5 U1 | scsi-iomemory-vsl version 3.2.6.1189 | 110356 | async | |
| ESXi 5.5 | scsi-iomemory-vsl version 3.2.9.1461 | 7.1.21 | async | |
| ESXi 5.5 | scsi-iomemory-vsl version 3.2.8.1350 | 116786 | async | |

Here, ESXi 5.5U2 (which is the version of ESXi that we are running), lists a number of different device drivers. At the top of the list is version **`3.2.9.1461`** of the ioDriver. When `vmkload_mod -s` command was run on the iomemory driver for Fusion-IO, it reported a driver version of **`3.2.8.1350`**. Although the driver is down-rev from the latest version from Fusion-IO, it is still supported for Virtual SAN, as it appears second on the list above. While this earlier driver is supported, it would be strongly recommended to upgrade to the newer version to ensure all possible known issues are avoided.

### Walk-through results

| Device | Vendor | Model | Driver | Firmware |
|---|---|---|---|---|
| Storage I/O Controller | HP | HP 410i | 5.5.0.74-1OEM | N/A |
| PCI-E flash device | FusionIO | F00-001-1T20-CS-0001 | 3.2.8 build 1350 | v7.1.15 |

The 410i controller is clearly unsupported. The FusionIO driver is down-rev.

## Flash considerations in version 5.5

While the focus in this scenario has been on a PCI-E flash device, there is of course full support for Solid State Disks on Virtual SAN. Of particular interest is the class of flash device. There are two classes, performance and endurance. Performance is measured in the number of writes per second, whereas endurance, in Virtual SAN 5.5, is measured in full disk writes per day (DWPD).

When deciding which class of flash device to use with your Virtual SAN deployment, these are the two features that require serious consideration.

Performance classes are as follows:

| Class | Writes per second |
|---|---|
| **B** | 5,000 – 10,000 |
| **C** | 10,000 – 20,000 |
| **D** | 20,000 – 30,000 |
| **E** | 30,000+ |

Endurance classes for Virtual SAN 5.5 are as follows:

| Class | Disk Writes Per Day |
|---|---|
| **A** | 1-5 |
| **B** | 5-10 |
| **C** | 10-20 |
| **D** | 20+ |

However for Virtual SAN 6.0, endurance is now considered using the metric TBW, or Terabytes Written.

## Flash Considerations in version 6.0

For Virtual SAN 6.0, the endurance class has been updated to use Terabytes Written (TBW), over the vendor's drive warranty. Previously the specification was full Drive Writes Per Day (DWPD).

By quoting the specification in Terabytes Written (TBW), VMware allows vendors the flexibility to use larger capacity drives with lower full Drive Writes Per Day (DWPD) specifications.

For instance, a 200GB drive with 10 full DWPD is equivalent to a 400GB drive with 5 full DWPD from an endurance perspective. If VMware kept a specification of 10 DWPD, the 400 GB drive with 5 DWPD would be excluded from the Virtual SAN certification.

By changing the specification to 2 TBW per day, both the 200GB drive and 400GB drives are qualified - 2 TBW per day is the equivalent of 5 DWPD for the 400GB drive and is the equivalent of 10 DWPD for the 200GB drive.

Of course, this is also a useful reference for the endurance of flash devices used on the capacity layer, but these devices tend not to require the same level of endurance as the flash devices used as the caching layer.

| Class | SSD Tier | TBW (Per Day) | TBW (5 years) |
|---|---|---|---|
| **A** | All Flash Capacity | 0.2 | 365 |
| **B** | Hybrid – Caching | 1 | 1825 |
| **C** | All Flash – Caching (medium workloads) | 2 | 3650 |
| **D** | All Flash – Caching (high workloads) | 3 | 5475 |

## All-Flash device considerations

VMware is supporting flash drives for capacity in the 6.0 versions of Virtual SAN. Flash drives have shown themselves notoriously sensitive to firmware levels and requiring updates. Once flash drives for capacity start to appear on the VCG, ensure that you use the most recent versions of drivers and firmware.

## Magnetic disk considerations

While there is no need to check drivers/firmware of your hard disk drives (HDD aka magnetic disks), one should continue to check that the SAS or SATA drives planned for the Virtual SAN deployment are supported. The main feature of the drives is the RPM, revolutions per minute. RPMs are as follows:

| Drive Type | RPM |
|---|---|
| **NL-SAS (Near line SAS)** | 7200 |
| **SAS** | 10000 |
| **SAS** | 15000 |
| **SATA** | 7200 |

As you can see, SAS drives can perform much better than NL-SAS and SATA, with the only NL-SAS and SATA disks being 7200 RPMs. NL-SAS disks, which are basically SATA disks with a SAS daughter card, should be treated just like SATA disks. They have the same performance and characteristics as SATA drives. For performance at the magnetic disk layer, serious consideration should be given to the faster SAS drives.

## External storage enclosure considerations

Blade servers are becoming increasingly popular but these do not use direct attached storage for the most part. Most blade server deployments use external storage enclosures.

VMware is supporting limited external storage enclosure configurations in the 6.0 versions of Virtual SAN. Once again, if you plan to use external storage enclosures with Virtual SAN, ensure the VCG is adhered to with regards to versioning for these devices. Customers will be expected to configure the externals storage enclosures such that a disk is only visible to one host.

## Processor power management considerations

While not specific to Virtual SAN, processor power management settings can have an impact on overall performance. Certain applications that are very sensitive to processing speed latencies may show less than expected performance when processor power management features are enabled. A best practice is to select a 'balanced' mode and avoid extreme power-saving modes. There are further details found in VMware KB article 1018206.

## VCG check summary

There are obviously quite a few tasks to do to bring the configuration that was used as an example up to a supported state.

- Hosts that appear in the server section of the VCG, e.g. HP DL380 G7, don't necessarily mean that they are supported for Virtual SAN. HP DL380 G7 hosts do not support any of the controllers necessary to run Virtual SAN. G8 servers would be required from HP to support Virtual SAN.

- Ensure that the controller is supported. The HP 410i controllers in this example will need to be replaced with a different model that appears on the VCG at the time of writing, perhaps a HP 420i.

- Ensure that the controller used is configured with a supported mode, either pass-through or RAID-0. This should be done at deployment, as changing to a new mode may involve an ESXi reinstall.

- Ensure that the driver is supported for Virtual SAN. The *async* driver that is installed as part of the HP customized ISO is also not supported at the time that this research was done, so this will need to be changed for a supported *inbox* driver.

- Ensure that the flash device or SSD is supported. The Fusion-IO PCI-E card is on the VCG but while the driver is running a supported version, the firmware will need to be upgraded to make it supportable.

- Using tools from third party vendors can help to quickly retrieve driver and firmware information about the third party's product

- When choosing flash devices for cache, consider both performance (writes per second) and endurance (TBW/DWPD) characteristics of the drives.

- When choosing flash devices for capacity in all-flash configuration, consider the endurance characteristics of the drives. These devices do not need to be as high performing as the flash devices used at the cache layer.

- When choosing magnetic disks for hybrid configurations, consider performance characteristics (RPM) of the drives.

These are exactly the sorts of checks that would need to be made against the VCG to verify that you have a configuration that supports Virtual SAN.

# 5. Virtual SAN software components

This section aims to provide a brief overview of some of the Virtual SAN software components. Many of these components are not necessary to understand for day-to-day use of Virtual SAN. However when it comes to troubleshooting, you may see them referenced from time to time in logs, RVC commands and especially VSAN Observer outputs. Therefore, for completeness sake, a brief overview is provided.

### Local Log Structured Object Management - LSOM

**LSOM** works at the physical disk level, both flash devices and magnetic disks. It handles the physical storage for Virtual SAN components on the local disks of the ESXi hosts. It also handles the read caching and write buffering for the components.

### Distributed Object Manager - DOM

**DOM** is responsible for the creation of virtual machine storage objects from local components across multiple ESXi hosts in the Virtual SAN cluster by implementing distributed RAID. It is also responsible for providing distributed data access paths to these objects. There are 3 roles within DOM; client, owner and component manager.

- **Client:** Provides access to an object. There may be multiple clients per object depending on access mode.

- **Owner:** Coordinates access to the object, including locking and object configuration and reconfiguration. There is a single DOM owner per object. All objects changes and writes go through the owner. Typically the client and owner will reside on the same host, but this is not guaranteed and they may reside on different hosts.

- **Component Manager:** Interface for LSOM and the physical disks.

A node's DOM may play any of the three roles for a single object

### Cluster Level Object Manager - CLOM

**CLOM** ensures that an object has a configuration that matches its policy, i.e. stripe width or failures to tolerate, to meet the requirements of the virtual machine. Each ESXi host in a Virtual SAN cluster runs an instance of ***clomd***, which is responsible for the policy compliance of the objects. CLOM can be thought of as being responsible for the placement of objects and their components.

## Cluster Monitoring, Membership and Directory Services - CMMDS

**CMMDS** discovers, establishes and maintains a cluster of networked node members, for example, when a new host is added to the Virtual SAN cluster. It manages the inventory of items such as Nodes, Devices, Networks and stores metadata information such as policies, distributed RAID configuration, etc.

## Reliable Datagram Transport - RDT

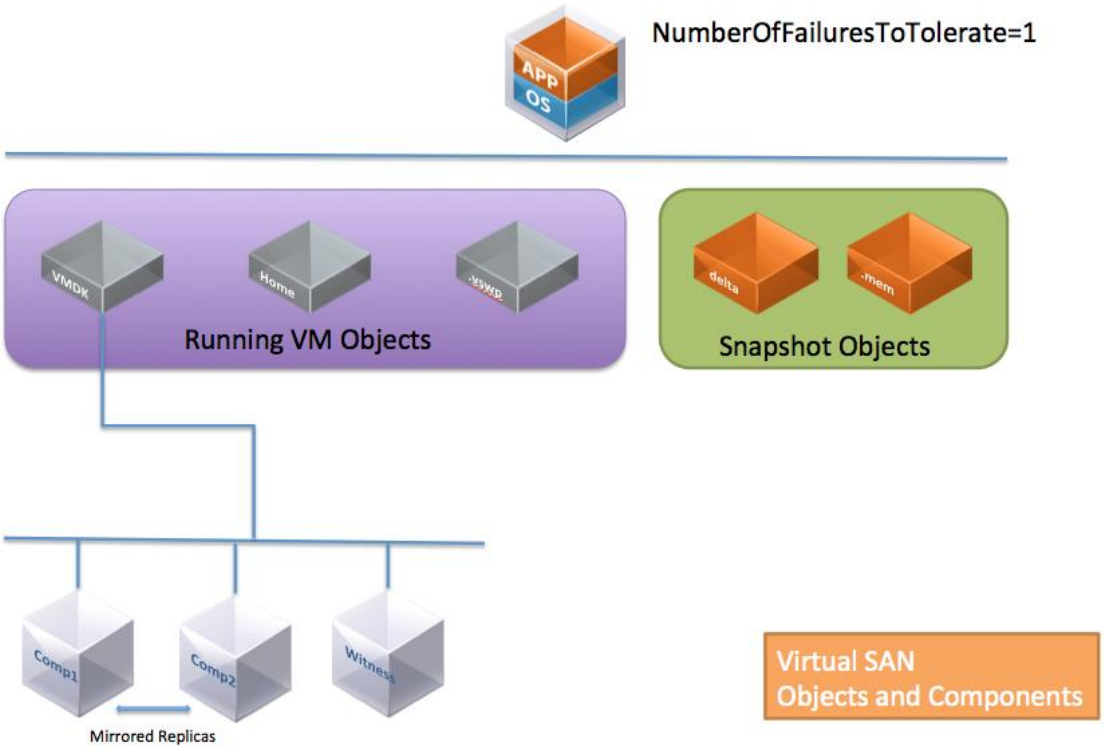
RDT, the reliable datagram transport, is the communication mechanism within Virtual SAN. It is uses TCP at the transport layer and it is responsible for creating and destroying TCP connections (sockets) on demand.

# 6. Understanding Availability & Accessibility

Virtual SAN is an object store. Understanding how failures impact virtual machine availability and accessibility is critical to ensure that you have a successful evaluation. This section looks at how a virtual machine is made up of objects and components, and how failures in the cluster can impact the objects and components and of course overall virtual machine availability and accessibility.

## Objects and components

A virtual machine deployed on a Virtual SAN datastore is comprised of a set of objects. These are the VM Home Namespace, the VMDK, and the VM Swap when the VM is powered on. In the case where a snapshot is taken, there are the delta VMDK objects and the VM memory snapshot object when memory state is captured as part of the snapshot:




Each of these objects is comprised of a set of components, determined by capabilities placed in the VM Storage Policy. For example, if *NumberOfFailuresToTolerate=1* is set in the VM Storage Policy, then some of the objects would be mirrored/replicated, with each replica being a component. If *NumberOfDiskStripesPerObject* is greater than 1 in the VM Storage Policy, then the object is striped across multiple disks and each stripe is said to be a component of the object.

***NOTE: For an object to be accessible in VSAN, more than 50 percent of its components must be accessible.***

## What is a replica?

Replicas are copies of the virtual machine storage objects that are instantiated when an availability capability is specified for the virtual machine. The availability capability dictates how many replicas are created. It enables virtual machines to continue running with a full complement of objects when there are host, network or disk failures in the cluster.

## What is a witness?

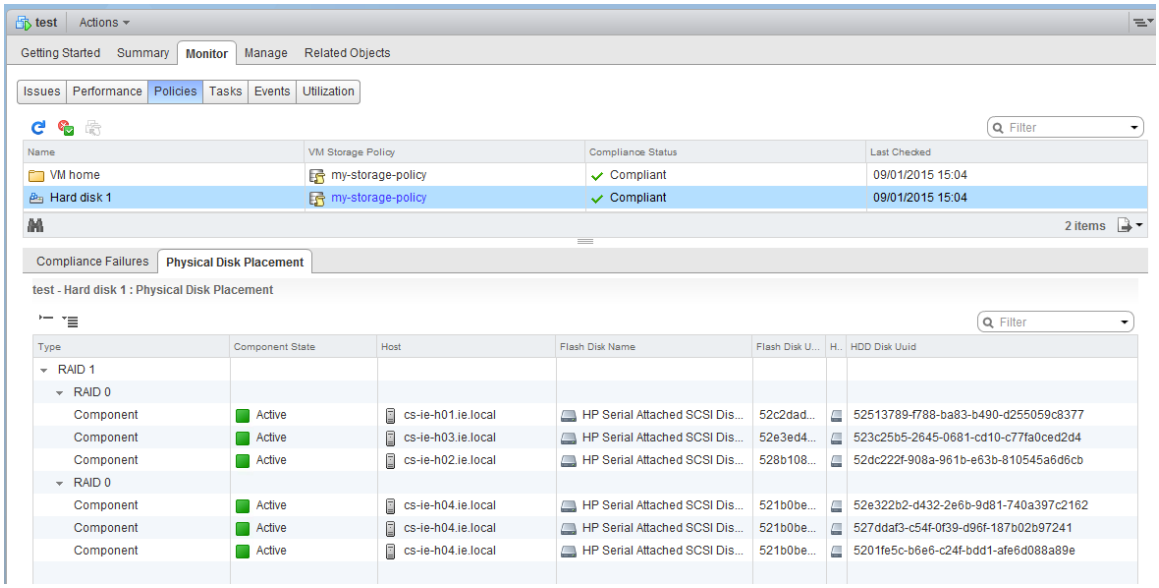
Another component that is part of a virtual machine object is a witness. Virtual SAN witnesses are used to provide an availability mechanism to virtual machines – they are not there to provide an availability mechanism to ESXi hosts/nodes participating in the Virtual SAN cluster.

In Virtual SAN 5.5, witnesses are part of every storage object that needs to tolerate a failure. They are components that do not contain data, only metadata. Their purpose is to serve as tiebreakers when availability decisions are made in the VSAN cluster and to have a vote when a quorum is needed in the cluster. A witness consumes about 2MB of space for metadata on the VSAN datastore when the on-disk format is v1, and about 4MB of space when the on-disk format is v2.

In Virtual SAN 5.5, for a virtual machine deployed on a Virtual SAN datastore to remain available, greater than 50% of the components that make up a virtual machine's storage object must be accessible. If less than 50% of the components of an object are accessible across all the nodes in a Virtual SAN cluster, that object will no longer be accessible on a Virtual SAN datastore, impacting the availability of the virtual machine. Witnesses play an import role in ensuring that more than 50% of the components of an object remain available.

**Important**: This behaviour changed significantly in Virtual SAN 6.0. In this release, a new way of handling quorum is introduced. There is no longer a reliance on "more than 50% of components". Instead, each component has a number of "votes", which may be one or more. To achieve quorum in Virtual SAN 6.0, "more than 50 percent of votes" is needed.

Take the following example of a Virtual SAN 6.0 deployment, which has a virtual machine, deployed with *NumberOfFailuresToTolerate = 1* and *NumberOfDiskStripesPerObject* = 3. In this the votes are distributed such that we can still guarantee virtual machine access in the event of a failure without the need for witnesses:




Note that at least "2n+1" hosts are needed to be able to tolerate "n" number of failures in the cluster. If the data components of an object do not span 2n+1 hosts, witnesses will still be needed in Virtual SAN 6.0. This appreciation of the relationship between virtual machines, objects and components will help with understanding the various Virtual SAN failure scenarios.

This table may also help you to understand the relationship between *NumberOfFailuresToTolerate*, component count and witnesses in Virtual SAN 5.5.

| Number Of Failures To Tolerate | Component count | Number of hosts needed | Component types | Number of components needed for object availability |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 x Replica | 100% or 1 component |
| 1 | 3 | 3 | 2 x Replica + 1 witness | > 50% or 2 components |
| 2 | 5 | 5 | 3 x Replica + 2 witness | > 50% or 3 components |

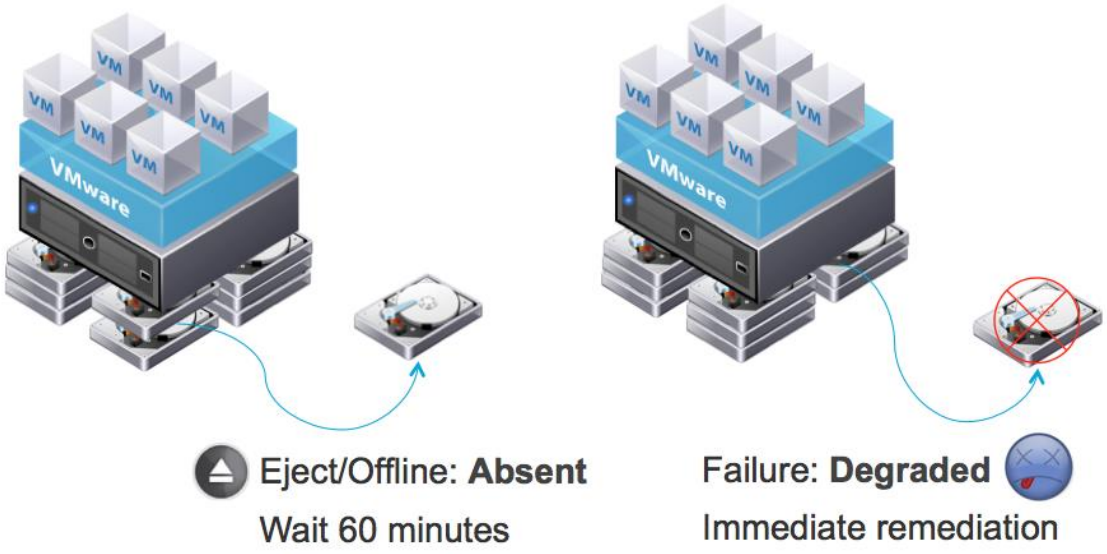
This next section looks at the different ways in which objects and components are affected by errors in Virtual SAN.

## Failures: absent vs. degraded

Virtual SAN has 2 types of failure states for components: ABSENT and DEGRADED.

- A component is **DEGRADED** if Virtual SAN has detected a failure from which it believes the component will not return (e.g. a component residing on a failed disk drive)
- A component is **ABSENT** if Virtual SAN has detected a failure, but Virtual SAN believes that the component may re-appear with all data intact (e.g. an ESXi host has been rebooted, and thus all components on disks on that ESXi host are impacted)

An ABSENT state reflects a transient situation that may or not resolve itself over time, and a DEGRADED state is a permanent state.




It is interesting to note the difference between a disk that has been hot unplugged or offlined and a disk that has actually failed, as shown above. Since the disk that was unplugged or offlined may be reinserted or onlined, Virtual SAN treats the components residing on such a disk as ABSENT. If a disk has a permanent failure, the components are marked as DEGRADED.

When a component is marked as ABSENT, Virtual SAN will wait for 60 minutes (by default) for the components to become available once again. If they do not, and the timer expires, Virtual SAN will begin rebuilding the components elsewhere in the cluster. DEGRADED components are rebuilt immediately.

Note that both ABSENT and DEGRADED are treated as a single failure by Virtual SAN; neither has any more impact than the other when it comes to managing failures-to-tolerate on the cluster. A component that is marked as ABSENT and a

component that is marked as DEGRADED are both considered a single failure. If a VM is configured to tolerate a single failure, a VM remains available in the event of a component going DEGRADED or ABSENT.

In order to rebuild components when a failure occurs, Virtual SAN will search for hosts that satisfy placement rules like that 2 mirrors may not share hosts or fault domains. It also looks for disks with free disk space.

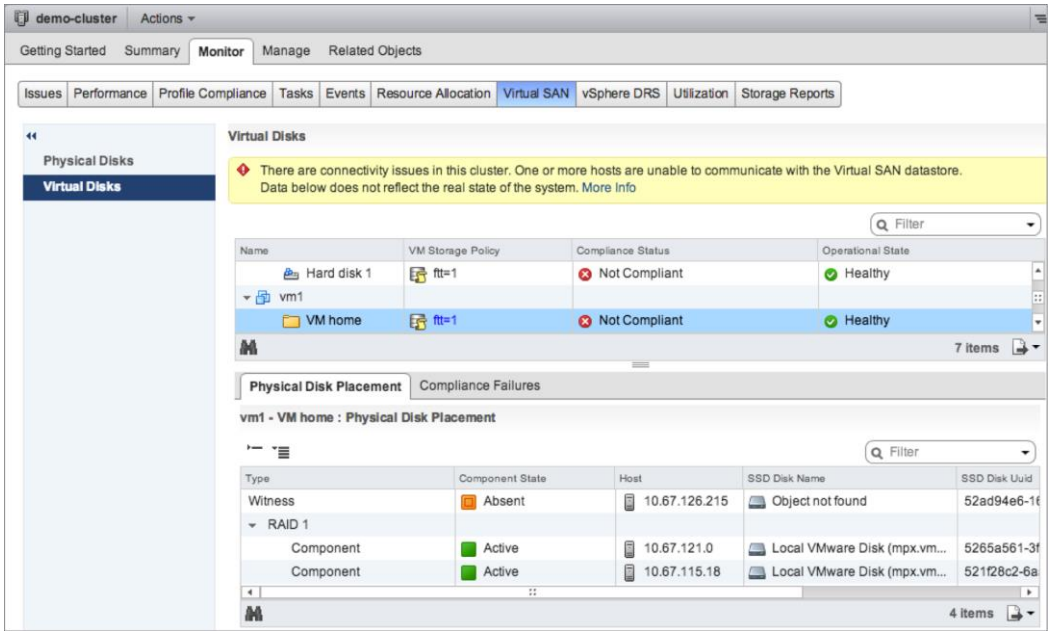
If such hosts and disks are found, Virtual SAN will create new components on those hosts and disks, and start the recovery process if there is enough capacity and resources in the cluster. However, the exact timing of this behavior depends on the failure state of the component. If the component is DEGRADED, i.e. Virtual SAN believes the data on that component is lost and won't return, Virtual SAN takes immediate action and starts the rebuild at once. If the component is ABSENT, implying Virtual SAN believes that the data may come back, the following optimization kicks in.

Virtual SAN has to tradeoff two aspects: On the one hand, Virtual SAN wants to minimize the time that virtual machines are exposed to risk due to reduced redundancy. On the other hand, a full rebuild of a component takes time and resources. If the disruption is short (minutes), it's better for Virtual SAN to just wait and allow the situation to rectify itself (e.g. reboot of host). If the disruption continues for long (hours), Virtual SAN should restore full redundancy even if the component will eventually come back. Thus, Virtual SAN uses a configurable timer of 60 minutes. If a component has been ABSENT for 60 minutes, Virtual SAN will proceed to replace it with a new component. We will discuss how to change the timeout in the advanced parameters appendix.

## Object compliance status: compliant vs. not compliant

A virtual machine is said to be **non-compliant** when one or more of its objects doesn't meet all requirements of its VM Storage Policy. This could mean one mirror is down on an object, but it could also mean that all of its mirrors are down.

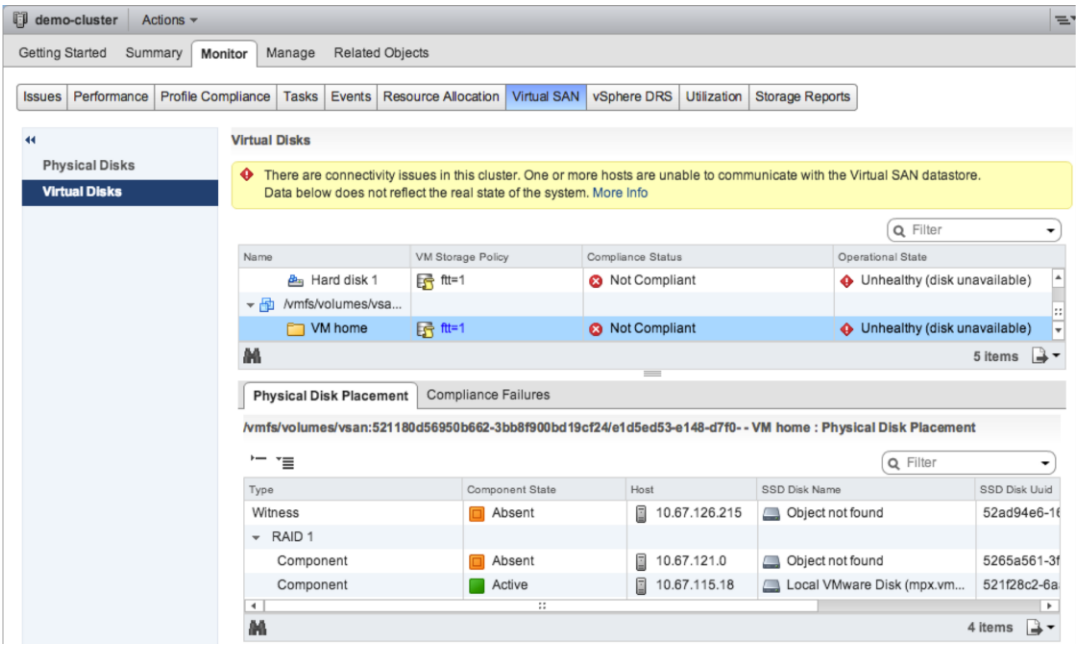
In the screenshot shown below, only one failure has occurred in the cluster. This has made a number of objects enter into a "*Not Compliant*" state. By looking closer at the objects in question in the *Physical Disk Placement* tab, it can be seen that one of the components has entered into an absent state (discussed earlier), but the object's *Operational Status* remains healthy as it still has greater than 50% of its components available (Virtual SAN 5.5) or it still has greater than 50% of its votes available (Virtual SAN 6.0) and one full mirror is still intact:



## Object operational state: healthy vs. unhealthy

The operation state of an object can be healthy or unhealthy, depending on the type of failure and number of failures in the cluster. If a full mirror is still available, and more than 50% of the object's components are available in Virtual SAN 5.5, (or 50% of the object's votes are available in Virtual SAN 6.0), the object's *Operational State* is said to be healthy. If no full mirror is available, or less than 50% of the components are available (possibly due to multiple failures in the cluster when the VM is configured to tolerate 1 failure only), then the object's operational state is said to be unhealthy.

If we take the example that there were two failures in the cluster, the *Compliance Status* remains at *Not Compliant*, the *Operation Status* changes to *Unhealthy*, and the reason for the unhealthy state is displayed. In this example, the reason for the "*Unhealthy*" Operations State is "*disk unavailable*" as shown below:



## VM Accessibility: inaccessible vs. orphaned

A VM's accessibility can also be dependent on which objects have been affected by a failure. These states will only be observed if a VM's objects have been setup to tolerate X number of failures in the cluster, and the cluster suffers greater than X number of failures.

If a VM's state is reported as *Inaccessible*, it means that at least one object of the VM is completely down (temporarily or permanently) so either there is no full mirror of the object (the failures have impacted both mirrors), or less than 50% of the components or votes are available (the failures have impacted a mirror and witnesses).

If a VM's state is reported as *Orphaned*, it may mean that neither vCenter Server nor the ESXi host can monitor/track the VM, i.e. there is no read access to VM's ".vmx" file. From a Virtual SAN perspective, this could imply that the VM Home Namespace is currently down, meaning that there is no full mirror of the object or less than 50% of the components or votes that make up the VM Home Namespace are available. There are other reasons why a VM may be *orphaned*, but a problem with the VM Home Namespace could be one of them.

However it should be understood that this state is a transient, and not a permanent state. As soon as the underlying issue has been rectified and once a full mirror copy and more than 50% of an object's components or votes become available, the virtual machine would automatically exit this inaccessible or orphaned state and become accessible once again.

## Failure handling – Virtual SAN fail safe mechanism

There have been occasions where the storage I/O controller does not tell the ESXi host that anything is wrong; rather the controller may simply stall I/O, and become 'wedged'. There have also been cases when pulling disk drives isn't handled well by the controller. For these reasons Virtual SAN has it's own fail-safe mechanism for failure handling.

Virtual SAN handles this condition by placing its own timeout on in-flight I/O, which is somewhere between 20-30 seconds. If the timeout expires (because the controller is stalled), Virtual SAN will mark the devices associated with the timed out I/O as DEGRADED.

If an I/O controller "wedged", and the I/O was destined for a capacity layer device, Virtual SAN marks this disk as DEGRADED. If the I/O was destined for a flash layer device, Virtual SAN marks the device as DEGRADED. Over time, all of the devices sitting behind this controller will also be marked as DEGRADED.

# VM behavior when multiple failures are encountered

An object remains accessible when a full mirror copy is available, and greater than 50% of the components or votes are available. The witnesses are there to assist with the latter requirement.

The behavior when there are more failures in the cluster than the *NumberOfFailuresToTolerate* setting is discussed here next.

## VM powered on and VM Home Namespace object goes inaccessible

If a running VM has its VM Home Namespace object go inaccessible due to failures in the cluster, a number of different things may happen. One of these includes the VM process crashing resulting in the VM powering off. Once the VM is powered off, it will be marked "inaccessible" in the vSphere web client UI. There can also be other side effects, such as the VM getting renamed in the UI to its vmx path rather than VM name, or the VM being marked "orphaned".

## VM powered on and disk object goes inaccessible

If a running VM has one of its disk objects go inaccessible, the VM will keep running, but its VMDK has all its I/O stalled. Typically, the Guest OS will eventually time out I/O. Some Windows versions may BSOD when this occurs. Many Linux distributions typically downgrade the filesystem on the VMDK to read-only.

The Guest OS behavior, and even the VM behavior is not Virtual SAN specific. This is simply ESXi behavior to APD (All Paths Down).

Once the VM becomes accessible again, the status should resolve, and things go back to normal. Of course, nothing ever gets corrupted.

Commands to examine the state of the network and storage will be seen shortly, and these will help to make a decision about what corrective action is needed to resolve VM accessibility issues.

# 7. Understanding expected failure behavior

When doing failure testing with Virtual SAN, or indeed, when failures occur on the cluster during production, it is important to understand the expected behavior for different failure scenarios.

## Disk is pulled unexpectedly from ESXi host

The Virtual SAN Administrators Guide provides guidelines on how to correctly decommission a device from a disk group. VMware recommends following tried and tested procedures for removing devices from Virtual SAN.

In Virtual SAN hybrid configurations, when a magnetic disk is unexpectedly pulled from an ESXi hosts (where it is being used to contribute storage to Virtual SAN) without first decommissioning the disk, all the Virtual SAN components that reside on the disk go ABSENT and are inaccessible. The same holds true for Virtual SAN all-flash configurations when a flash device in the capacity layer is pulled.

The ABSENT state is chosen over DEGRADED because Virtual SAN knows the disk is not lost, but rather just removed. If the disk gets reinserted into the server before the 60-minute timeout, no harm is done, and Virtual SAN simply syncs it back up. Virtual SAN returns to full redundancy without wasting resources on an expensive rebuild.

### Expected behaviors:

- If the VM has a policy that includes *NumberOfFailuresToTolerate=1* or greater, the VM's objects will still be accessible from another ESXi host in the Virtual SAN cluster.
- The disk state is marked as ABSENT and can be verified via vSphere web client UI.
- All in-flight I/O is halted while Virtual SAN reevaluates the availability of the object without the failed component as part of the active set of components.
- All components residing on this disk will be marked as ABSENT in the UI.
- If Virtual SAN concludes that the object is still available (based on available full mirror copy and witness/votes), in-flight I/O is restarted.
- The typical time from physical removal of the drive, Virtual SAN processing this event, marking the component ABSENT halting and restoring I/O flow is approximately 5-7 seconds.
- If the same disk is placed back on the same host within 60 minutes, no new components will be re-built.
- If 60 minutes pass, and the original disk have not been reinserted in the host, components on the removed disk will be built elsewhere in the cluster, if

capacity is available, including any newly inserted disks claimed by Virtual SAN.

- If the VM Storage Policy has *NumberOfFailuresToTolerate=0*, the VMDK will be inaccessible if one of the VMDK components (think one component of a stripe or a full mirror) exists on the pulled disk. To restore the VMDK, the same disk has to be placed back in the ESXi host. There is no option to recover the VMDK.

### Expected behaviors – UI view and log entries:

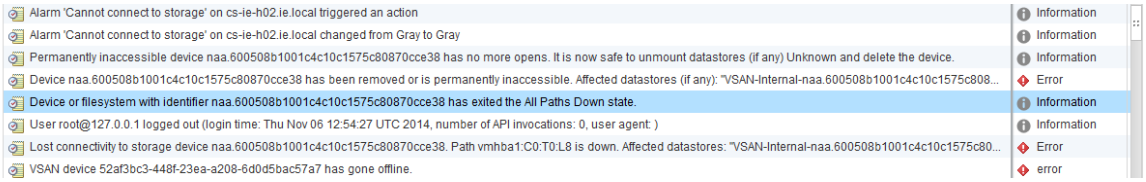
Removal of a disk will result in the following messages similar to the following appearing in the *vmkernel.log*:

```
2014-09-30T09:44:55.312Z cpu20:32849)WARNING: ScsiPath: 7028: Path lost for adapter
vmhba0 target 1 channel 0 pun 0  2014-09-30T09:44:55.312Z cpu22:33441)WARNING:
LSOMCommon: IORETRY_NotifyAPD:1647: Got APD event 1 on 0x410c0285ca90
2014-09-30T09:44:55.312Z cpu20:32849)WARNING: ScsiDevice: 8820: PDL set on VSAN device
path "vmhba0:C0:T1:L0"  [...]
2014-09-30T09:44:59.317Z cpu22:33506)WARNING: LSOM: LSOMEventNotify:4581: VSAN device
527cfbf5-ae7a-33f6-78bc-beed2f1730dd has gone offline.
```

It will also result in the following messages appearing in the *vobd.log*:

```
2014-09-30T09:44:59.317Z: [VsanCorrelator] 59865017530us: [vob.vsan.pdl.offline] VSAN
device 527cfbf5-ae7a-33f6-78bc-beed2f1730dd has gone offline.
2014-09-30T09:44:59.317Z: [VsanCorrelator] 59867627549us:
[esx.problem.vob.vsan.pdl.offline] VSAN device 527cfbf5-ae7a-33f6-78bc-beed2f1730dd has
gone offline.
```

Removal of a disk will trigger an event in the vSphere Web Client, under ESX host -> Monitor -> Events. You will find a "VSAN device XYZ has gone offline" similar to what was observed in vobd.log:



# Flash Cache SSD is pulled unexpectedly from ESXi host

When an SSD (acting as the cache tier) is pulled without decommissioning it, all the Virtual SAN components residing in that disk group go ABSENT and are inaccessible. In other words, if the flash tier SSD is removed, it will appear as a removal of the flash device as well as all associated magnetic disks (in hybrid configurations) or flash devices (in all-flash configurations) providing backing capacity.

### Expected behaviors:

- If the VM has a policy that includes *NumberOfFailuresToTolerate=1* or greater, the VM's objects will still be accessible.
- Disk group and the disks under the disk group states will be marked as ABSENT and can be verified via the vSphere web client UI.
- All in-flight I/O is halted while Virtual SAN reevaluates the availability of the objects without the failed component(s) as part of the active set of components.
- If Virtual SAN concludes that the object is still available (based on available full mirror copy and witness/votes), all in-flight I/O is restarted.
- All components residing in this disk group will be marked as ABSENT in the UI.
- The typical time from physical removal of the drive, Virtual SAN processing this event, marking the component ABSENT halting and restoring I/O flow is approximately 5-7 seconds.
- When the same SSD is placed back on the same host within 60 minutes, no new objects will be re-built.
- When the timeout expires (default 60 minutes), components on the impacted disk group will be rebuilt elsewhere in the cluster, if capacity is available.

- If the VM Storage Policy has *NumberOfFailuresToTolerate=0*, the VMDK will be inaccessible if one of the VMDK components (think one component of a stripe or a full mirror) exists on disk group whom the pulled flash tier SSD belongs to. To restore the VMDK, the same SSD has to be placed back in the ESXi host. There is no option to recover the VMDK.

## What happens when a disk fails?

Virtual SAN marks the disk as DEGRADED as the failure is permanent.

### Expected behaviors:

- If the VM has a policy that includes *NumberOfFailuresToTolerate=1* or greater, the VM's objects will still be accessible.
- The disk state is marked as DEGRADED and can be verified via vSphere web client UI.
- At this point, all in-flight I/O is halted while Virtual SAN reevaluates the availability of the object without the failed component as part of the active set of components.
- If Virtual SAN concludes that the object is still available (based on available full mirror copy and witness), all in-flight I/O is restarted.
- The typical time from physical removal of the drive, Virtual SAN processing this event, marking the component DEGRADED halting and restoring I/O flow is approximately 5-7 seconds.
- Virtual SAN now looks for any hosts and disks that can satisfy the object requirements. This includes adequate free disk space and placement rules (e.g. 2 mirrors may not share the same hosts/fault domains). If such resources are found, Virtual SAN will create new components on there and start the recovery process immediately.

- If the VM Storage Policy has *NumberOfFailuresToTolerate=0*, the VMDK will be inaccessible if one of the VMDK components (think one component of a stripe) exists on the pulled disk. This will require a restore of the VM from a known good backup.

## What happens when a cache tier SSD fails?

A cache tier flash device failure follows a similar sequence of events to that of a disk failure with one major difference; Virtual SAN will mark the entire disk group as DEGRADED. Virtual SAN marks the flash device and all the capacity devices in the disk group as DEGRADED as the failure is permanent (disk is offline, no longer visible, etc.).

### Expected behaviors:

- If the VM has a policy that includes *NumberOfFailuresToTolerate=1* or greater, the VM's objects will still be accessible from another ESXi host in the Virtual SAN cluster.
- Disk group and the disks under the disk group states will be marked as DEGRADED and can be verified via the vSphere web client UI.
- All in-flight I/O is halted while Virtual SAN reevaluates the availability of the objects without the failed component(s) as part of the active set of components.
- If Virtual SAN concludes that the object is still available (based on available full mirror copy and witness), all in-flight I/O is restarted.
- The typical time from physical removal of the drive, Virtual SAN processing this event, marking the component DEGRADED halting and restoring I/O flow is approximately 5-7 seconds.
- Virtual SAN now looks for any hosts and disks that can satisfy the object requirements. This includes adequate free SSD and disk space and placement rules (e.g. 2 mirrors may not share the same hosts/fault domains). If such resources are found, Virtual SAN will create new components on there and start the recovery process immediately.

- If the VM Storage Policy has *NumberOfFailuresToTolerate=0*, the VMDK will be inaccessible if one of the VMDK components (think one component of a stripe) exists on disk group whom the pulled SSD belongs to. There is no option to recover the VMDK. This may require a restore of the VM from a known good backup.

## New disk is placed in the ESXi host

### Expected behaviors:

- When a new partition-free disk is placed on the ESXi host, whether or not the ESXi host detects it automatically is dependent on whether the storage controller is configured for RAID-0 or pass-through. If pass-through, the device should be visible on a rescan. If a RAID-0 configuration is required, a RAID-0 volume will need to be configured on the device for it to be visible to the host. The Virtual SAN Administrators Guide will have further details on the steps involved.
- If the cluster is in automatic mode, Virtual SAN will claim the disk and add it to any of the existing disk groups.
- If the cluster is in manual mode, an administrator will have to manually claim the disk by adding it to a disk group via the vSphere web client UI.
- If the device is a flash device, and the wish is to use it as part of the capacity layer in an all-flash configuration, additional steps are needed to mark it as a capacity device. The Virtual SAN Administrators Guide will have further details on the steps involved.

## New cache tier SSD is placed in the ESXi host

### Expected behaviors:

- When replacing a cache tier flash device with a new cache tier flash device, whether or not the ESXi host detects it automatically is dependent on whether the storage controller is configured for RAID-0 or pass-through. If pass-through, the device should be visible on a rescan. If a RAID-0 configuration is required, a RAID-0 volume will need to be configured on the SSD for it to be visible to the host. The Virtual SAN Administrators Guide should have further details.
- Next, an administrator must manually decommission existing magnetic disks (hybrid) or SSDs (all-flash) in the capacity tier by removing the disks from their current disk group. Unless you do this, these disks cannot be associated to the new cache tier SSD. Note that this decommissioning process does not preserve the data on the disks. The data will be rebuilt automatically once the new disk group is created.
- If you are doing a proactive replacement of a cache tier SSD, you can use maintenance mode to do a full evacuation of the data from the disk group before replacing the SSD. In fact, Virtual SAN 6.0 allows for the evacuation of data from individual capacity devices when they are being removed from a disk group. However this is obviously not possible on a failure.

- After manual decommissioning of the magnetic disks, the new disk group with the new SSD should claim the available disks if the cluster is in automatic mode. There are situations where disks do not get claimed even when the cluster is in automatic mode. These are covered in detail in the Common storage problems and resolutions section of the troubleshooting Virtual SAN storage chapter of this manual.
- If the cluster is not in automatic mode, then the administrator will have to build the disk group by manually claiming the disks via the UI. New components will be then be built on the new disk group.

## What happens when a server fails or is rebooted?

A host failure can occur in a number of ways. It could be a crash, or it could be a network issue (which is discussed in more detail in the next section). However, it could also be something as simple as a reboot, and the host will be back online when the reboot process completed. Once again, Virtual SAN needs to be able to handle all of these events.

If there are active components of an object residing on the host that fails or is offline (due to any of the stated reasons), then those components are marked as ABSENT. I/O flow to the object is restored within 5-7 seconds by removing the ABSENT component from the active set of components in the object.

The ABSENT state is chosen rather than the DEGRADED state because in many case host failure is a temporary condition. A host might be configured to auto-reboot after a crash, or the host's power cable was inadvertently removed, but plugged back in immediately. Virtual SAN is designed to allow enough time for a host to reboot before starting rebuilds on other hosts so as not to waste resources. Because Virtual SAN cannot tell if this is a host failure, a network disconnect or a host reboot, the 60-minute timer is once again started. If the timer expires, and the host has not rejoined the cluster, a rebuild of components on the remaining hosts in the cluster commences.

If a host fails, or is rebooted, this event will trigger a "Host connection and power state" alarm, and if vSphere HA is enabled on the cluster, it will also cause a" vSphere HA host status" alarm and a "Host cannot communicate with all other nodes in the VSAN Enabled Cluster" message.

If *NumberOfFailuresToTolerate=1* or higher in the VM Storage Policy, and an ESXi host goes down, VMs not running on the failed host continue to run as normal. If any VMs with that policy were running on the failed host, they will get restarted on one of the other ESXi hosts in the cluster by vSphere HA, as long as it is configured on the cluster.

***Caution**: If VMs are configured in such a way as to not tolerate failures, (NumberOfFailuresToTolerate=0), a VM that has components on the failing host will become inaccessible through the vSphere web client UI.*

## What happens when a network link is pulled?

This is a similar situation to the host failure discussed previously. When a network is pulled from one of the hosts participating in a Virtual SAN cluster, Virtual SAN doesn't know if it is a host failure (permanent issue) or host reboot/cable pull (transient issues).

The network will partition, with the host with the network link down residing in its own partition, and the remaining hosts in their own partition. Virtual SAN will also report a "network misconfiguration detected" state in the vSphere web client UI.

Each host will elect an "owner" for the object when a network partition occurs. Then, for each component on the isolated host, all other components that are part of the object but reside on hosts not accessible from the partition will be marked as ABSENT. As all partitions are doing this, it means there are now multiple "owners" in the cluster for a given object, and they all have a different perspective on which components are ABSENT and which are ACTIVE. Thus, depending on which ESXi host is being queried during a network partition, the status of an object may look different.

However, while we have multiple "owners", at most only one owner (but possibly none in the case of a full network partition) will have quorum (which requires more than 50% of all components/votes to be ACTIVE). By requiring more than 50% Virtual SAN ensures that there can only be at most one partition with quorum. And only an object that has quorum can be "available" and hence accept any IO (read or write). This ensures that while availability may be lost during a network partition, Virtual SAN will never corrupt data or have divergence in data among the split brains.

The 60-minute timeout is once again utilized. If the timer expires, a rebuild of component commences in the cluster without the partitioned host, if resources are available. If the network is restored before the timer expires, the out-of-date components on the partitioned host simply synchronize with the newer components in the cluster. A lot of data components are likely to be stale and may need to be resynchronized. The time to resync depends on the amount of data written while the components were ABSENT.

When a network partition heals or when an isolated host can reach the remaining cluster again, Virtual SAN will establish a new cluster which includes the previously partitioned host and any ABSENT components will start to resync. If these components have already been rebuilt elsewhere, due to the network issue lasting longer than 60 minutes, Virtual SAN discards the components on the previously partitioned host and continues to use the more recently built components.

## What happens when the entire cluster network fails?

A failure of this nature will result is a complete network partition, with each host residing in its own partition. To an isolated host, it will look like all the other hosts have failed. Behavior is similar to what was described in the previous section, but in this case, since no quorum can be achieved for any object, no rebuilding takes place. When the network issue is resolved, Virtual SAN will first establish a new cluster and components will start to resync. Since the whole cluster was down, there won't have been many changes, but Virtual SAN ensures that components are synchronized against the latest, most up to date copy of a component.

## What happens when a storage I/O controller fails?

If a controller in a single controller configuration fails with a permanent failure, then every disk group on the host will be impacted. This condition should be reasonably easy to diagnose. The behavior of Virtual SAN when a storage I/O controller fails will be similar to having all-flash cache devices and all disks fail in all disk groups. As described above, components will be marked as DEGRADED in this situation (permanent error) and component rebuilding should be immediate.

It might be difficult to determine if there is a flash cache device failure or a storage I/O controller failure in a single controller configuration, where there is also only one disk group with one flash device in the host. Both failures impact the whole disk group. The VMkernel log files on the ESXi host may be able to assist with locating the root cause. Also, leveraging third party tools from the controller vendor to query the status of the hardware might be helpful.

If there are multiple disk groups on a host with a single controller, and all devices in both disk groups are impacted, then you might assume that the common controller is a root cause.

If there is a single disk group on a host with a single controller, and all devices in that disk group are impacted, additional research will be necessary to determine if the storage I/O controller is the culprit, or if it is the flash cache device that is at fault.

Lastly, if there are multiple controllers on the host, and only the devices sitting behind one controller is impacted, then you might assume that this controller is a root cause.

## Handling multiple failures

It is important when testing Virtual SAN to limit the testing to only one thing at a time. This is especially true if you have used a VM Storage Policy that configures objects with *NumberOfFailuresToTolerate=1*. These objects typically have 2 replicas (A, B) and 1 witness (W). Imagine the host containing replica 'A' crashes. As discussed above, in order to restore I/O flow, Virtual SAN will use components 'B' and 'W' in the active set. Virtual SAN allows replica A to become "STALE".

Now assume that the host holding the witness 'W' also crashes. At this point, the object will lose availability as less than 50% of the components/votes are available. This is expected. The user asked Virtual SAN to tolerate 1 concurrent failure, but 2 concurrent failures have now occurred on the cluster.

Now the host that contains replica 'A' comes back online. You might think that since there are now 2 out of 3 components available (i.e. quorum) and a full replica of the data, the object should now be accessible. However, the object would still not be accessible, because replica 'A' is STALE and is not up to date. Replica 'B' has had a number of updates not yet synced to replica 'A'.

Virtual SAN requires a quorum of up-to-date components/votes, but here we have the following situation: 'A' is present yet STALE, 'B' is present and up to date, W is missing. So Virtual SAN only has 1 out of 3 up-to-date components and so will not make the object accessible. This is a necessary precaution as otherwise we could construct a scenario where Virtual SAN would incorrectly conclude an object is up-to-date when it is actually not. So in this scenario, even though 'A' came back up, Virtual SAN would still consider this a double failure.

When the witness (W) returns, the object can now become accessible. This is because we now have two out of three up-to-date components forming a quorum. The up-to-date replica (B) can be synced to the STALE replica (A) and when complete, the component will be placed in ACTIVE state.

# 8. Getting started with RVC

## Introduction to RVC and VSAN observer

VMware vCenter Server contains a command line interface (CLI) tool for the monitoring and troubleshooting of Virtual SAN deployments. The command line tool in called the Ruby vSphere Console (RVC). This tool will give administrators a cluster-wide view of Virtual SAN instead of the host-centric view offered by ESXCLI.

While RVC is extremely useful by itself, RVC also provides another utility called VSAN Observer. Through RVC, VSAN Observer may be launched. It provides a web browser interface for monitoring Virtual SAN. In this section, we introduce you to both RVC and VSAN Observer. The first step is to show how to launch RVC, the Ruby vSphere Console.

Since VMware has two flavors of vCenter Server, the Windows version and the Linux appliance version, how to log into both versions is shown, as the steps are subtly different.

## RVC deployment suggestion/recommendation

VMware suggests that customers consider deploying a standalone vCenter Server Appliance that does not manage any of the infrastructures, but instead is used to simply manage Virtual SAN, i.e. use it as an "RVC appliance". The "RVC appliance" may be used to connect to any vCenter Server in your infrastructure, and thus this RVC appliance can be used to manage multiple Virtual SAN deployments if you so wish.

## Launching RVC from the vCenter Server Appliance

Once the vCenter Server Appliance is deployed, no additional configuration is required to begin using the Ruby vSphere Console. To begin using the Ruby vSphere Console, simply open a secure shell (ssh) to the dedicated vCenter Server Appliance and login as a privileged user. This privileged user is the exact same user that would be used for logging into the vSphere web client for administrative purposes.

Next, run the ***rvc*** script that should already be in the executable path.

```
Syntax:  rvc [options] [username[:password]@]hostname
```

Here is an example of (a) logging into the vCenter Appliance version 5.5 and (b) launching RVC:

```
login as: root
VMware vCenter Server Appliance
administrator@192.168.1.100's password: ******
Last login: Thu Jul 17 22:29:15 UTC 2014 from 10.113.230.172 on ssh  Last failed
login: Fri Jul 18 06:31:16 UTC 2014 from 192.168.1.20 on ssh:notty
There was 1 failed login attempt since the last successful login.
Last login: Fri Jul 18 06:31:35 2014 from 192.168.1.20
vcsa:~ # rvc administrator:vmware@192.168.1.100
0 /
1 192.168.1.100/
>
```

**Note**: The 6.0 vCenter Server Appliance is somewhat different to previous versions. Before launching RVC, a shell environment must first be launched. To do that, the following commands must be run:

```
login as: root

 VMware vCenter Server Appliance 6.0.0

 Type: vCenter Server with an embedded Platform Services Controller

 root@ie-vcsa-03's password:
 Last login: Thu Nov 27 13:57:35 2014 from ie-loginisite-01.ie.local
 Connected to service

     * List APIs: "help api list"
     * List Plugins: "help pi list"
     * Enable BASH access: "shell.set --enabled True"
     * Launch BASH: "shell"

 Command> shell
 Shell is disabled.
 Command> shell.set --enabled True
 Command> shell
     ---------- !!!! WARNING WARNING WARNING !!!! ----------

 Your use of "pi shell" has been logged!

 The "pi shell" is intended for advanced troubleshooting operations and while
 supported in this release, is a deprecated interface, and may be removed in a
 future version of the product.  For alternative commands, exit the "pi shell"
 and run the "help" command.

 The "pi shell" command launches a root bash shell.  Commands within the shell
 are not audited, and improper use of this command can severely harm the
 system.

 Help us improve the product!  If your scenario requires "pi shell," please
 submit a Service Request, or post your scenario to the
 communities.vmware.com/community/vmtn/server/vcenter/cloudvm forum.

 ie-vcsa-03:~ #
```

There are a number of interesting points to make about this login sequence. In the first place, the user has to login to the vCenter Server Appliance as a privileged user. In this case, the user was `root`. Once logged in successfully as root, the user next has to login to RVC to vSphere as someone with suitable credentials. We can see that the user logs into RVC as `administrator` and also supplied the password on the command line; "vmware" in this case. Also the IP address of the vCenter server was

provided (which is the same IP address as the vCenter Server Appliance where RVC is running).

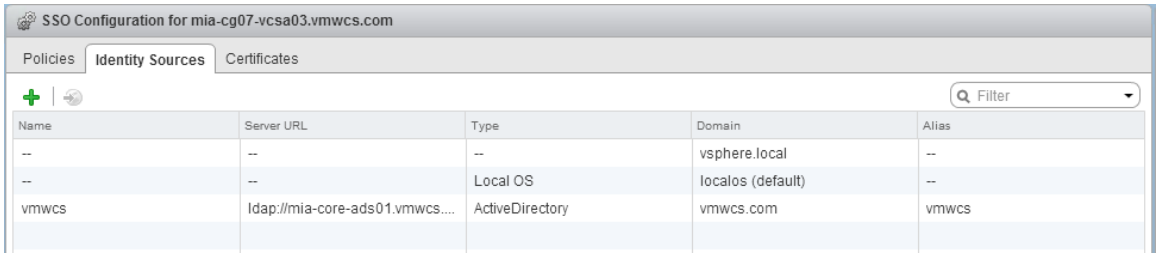
Note that you do not need to supply a username, password or hostname in the rvc command line. If you do not, RVC will simply prompt you to each of these in turn.

```
ie-vcva-01:~ # rvc
Host to connect to (user@host): administrator:vmware@192.168.1.100
password:
0 /
1 localhost/
>
```

In the above example, "`administrator`" was the privileged user that logged into RVC. This may need some further clarification as vSphere can support a number of different sources for users. For example, the user might be a local user on the vCenter Server Appliance (e.g. root), or it could be a Single Sign On (SSO) user such as administrator@vsphere.local or indeed an Active Directory (AD) user. The "default domain" setting in vCenter's Single Sign-On configuration determines default domain for the administrator user.

The default domain in SSO is important as it determines which password should you be supplying to login to not just RVC, but also the vSphere web client. As mentioned, there are multiple administrator users in vSphere environments. It could be the AD administrator, the SSO administrator or in the case of Windows vCenter Servers, the local administrator. This has caused a little confusion in the past, which is why we are taking the time to explain it here.

To check the current default domain settings, login to the vSphere web client as administrator@vsphere.local, navigate to Administration > Single Sign-On > Configuration > Identity Sources. There are entries for the vsphere.local domain, the Local OS and Active Directory if it is configured. One of these will be the default, and this has a direct bearing on which administrator is the default one when administrator is used as the username when attempting to launch RVC.




In the above example, *localos* is the default. Therefore RVC will attempt to find a local user to authenticate for RVC. If this is the vCenter Server Appliance, which runs Linux, it is unlikely that there will be a local privileged user called administrator. Therefore an attempt to launch RVC with this user will typically not work. Note that this is not a consideration specific to RVC. This will not allow a log via the vSphere

web client as simply "administrator" either; the user will either have to provide the active directory domain or the SSO domain along with the username and password to successfully login.

The options here, for the vCenter Server Appliance, are to log into RVC as the Active Directory "administrator" or as the SSO administrator. For this to succeed, the AD domain or the SSO domain must also be included in the login (when localos is chosen as the default). For SSO, this would look similar to:

```
vcsa:~ # rvc vsphere.local\administrator:vmware@192.168.1.100
0 /
1 192.168.1.100/
>
```

Here is another example of logging is as the SSO Administrator, but typing the password at the prompt:

```
ie-vcsa-02:~ # rvc
Host to connect to (user@host): administrator@vsphere.local@localhost
password:
0 /
1 localhost/
>
```

For Active Directory, where an AD domain is *vmwcs.com,* this would look similar to:

```
vcsa:~ # rvc vmwcs\administrator:vmware@192.168.1.100
0 /
1 192.168.1.100/
>
```

Note that if any of the passwords contains a special character, it will need to be escaped out with a "\". See the following example where the password is vmware123!:

```
vcsa:~ # rvc vmwcs\administrator:vmware123\!@192.168.1.100
0 /
1 192.168.1.100/
>
```

By now, the reader should have enough information to successfully login to RVC. In the next section, navigating RVC and running various RVC commands is examined.

# Launching RVC from Windows vCenter Server

RVC for Windows and RVC for Linux are practically identical. However there are some nuances when it comes to launching. The same rules apply as before, where knowing what single sign-on (SSO) has configured as the "default domain". This will determine which administrator is the default, and which password should be supplied. This might be even more confusing on the Windows version of vCenter Server since the Windows Operating system will also have a local administrator user. Now we have Windows local administrator, an Active Directory administrator (if configured) and a SSO administrator all as potential logins to RVC.

In the vCenter server 6.0, an **rvc.bat** file is used to launch RVC. You will find this file in the following path:

*C:\Program Files\VMware \VirtualCenter Server\rvc*.

In vCenter 5.5, the path is:

*C:\Program Files\VMware\Infrastructure\VirtualCenter Server\rvc*.

Running rvc.bat with the '-h' for help option, the following information is displayed:

```
Ruby vSphere Console.

 Usage:
        rvc [options] [username[:password]@]hostname

 where [options] are:
           --path <s>:   Initial directory
   --create-directory:   Create the initial directory if it doesn't exist
        --cmd, -c <s>:   command to evaluate
     --script, -s <s>:   file to execute
    --script-args <s>:   arguments to script (default: )
     --cookie, -k <s>:   authentication cookie file
          --quiet, -q:   silence unnecessary output
        --version, -v:   Print version and exit
           --help, -h:   Show this message
```

Username, password and hostname information must be provided at the command line. In 6.0, if this information is not provided at the command line, the default login for RVC has been set to the SSO administrator, administrator@vsphere.local, and the default host has been set to localhost. Here is a default rvc.bat running:

```
C:\Program Files\VMware\vCenter Server\rvc>rvc.bat
 Try "--help" or "-h" to view more information about RVC usage.
 No argument is specified, connecting to localhost as administrator@vsphere.local
 password:
```

In many cases, the **rvc.bat** file may be edited and modified so that it contains specific login and vCenter server details. Here are the contents of rvc.bat:

```
@echo off
 setlocal
 if "%1"=="" (
   echo Try "--help" or "-h" to view more information about RVC usage.
   echo No argument is specified, connecting to localhost as administrator@vsphere.local
   set RVC_ARGS="administrator@vsphere.local@localhost"
 ) else (
   set RVC_ARGS=%*
 )
 pushd "%VMWARE_RVC_HOME%"
 "%VMWARE_RUBY_BIN%"    -Ilib    -Igems\backports-3.1.1\lib    -Igems\builder-3.2.0\lib    -
Igems\highline-1.6.15\lib  -Igems\nokogiri-1.5.6-x86-mingw32\lib  -Igems\rbvmomi-1.7.0\lib
-Igems\terminal-table-1.4.5\lib        -Igems\trollop-1.16\lib        -Igems\zip-2.0.2\lib
bin\rvc %RVC_ARGS%
 popd
```

To edit **rvc.bat**, follow these steps:

1. Open `rvc.bat` with an editor such as notepad.
2. Navigate to the line containing set RVC_ARGS and modify the "`administrator@vsphere.local@localhost`" to the name of the user that will connect to the vCenter Server, and the name of the vCenter server.
3. Save the rvc.bat.

Note once again that the Single Sign-On default domain setting in vCenter Server determines which is the default domain (AD, localos or SSO). If there is a wish to connect as a user that is not in the default domain, this will require adding the domain name to the username as was seen previously.

Now when **rvc.bat** is run, it should launch the Ruby vSphere Console successfully on your Windows version of vCenter Server and connect to the desired vCenter server.

# 9. Navigating RVC

The Ruby vSphere Console represents the vCenter inventory as a virtual filesystem. Once launched, users can navigate the vCenter server inventory using typical shell command such as "cd" to change directory (navigate to another part of the inventory, for example to navigate from cluster level to host level) and "ls" to display directory contents (for example, display all the hosts in a cluster).

There are also a number of additional and very useful features, such as:

- Command completion through the use of the tab key
- Wildcards, such as *
- Numeric references to objects when navigating, such as "cd 0", "cd 1", "cd 2"

A 6.0 version of the RVC Reference Guide is available if further detail is required.

## Navigating RVC examples

In this example, we log into a vCenter Server Appliance, and connect to the localhost as user root, which is a local user (note that the password contains a special character, "!", that must be escaped out using a "\"). Once logged in, we can start to use the numeric references to objects, as well as tab complete to show a list of available commands. In these examples, a 4-node cluster is being used.

```
ie-vcva-01:~ # rvc root:VMware123\!@localhost
 0 /
 1 localhost/
 > cd 1
 /localhost> ls
 0 ie-datacenter-01 (datacenter)
 /localhost> cd 0
 /localhost/ie-datacenter-01> ls
 0 storage/
 1 computers [host]/
 2 networks [network]/
 3 datastores [datastore]/
 4 vms [vm]/
 /localhost/ie-datacenter-01> cd 1
 /localhost/ie-datacenter-01/computers> ls
 0 ie-vsan-01 (cluster): cpu 109 GHz, memory 330 GB
 /localhost/ie-datacenter-01/computers> cd 0
 /localhost/ie-datacenter-01/computers/ie-vsan-01> ls
 0 hosts/
 1 resourcePool [Resources]: cpu 109.65/109.65/normal, mem 330.46/330.46/normal
 /localhost/ie-datacenter-01/computers/ie-vsan-01> cd 0
 /localhost/ie-datacenter-01/computers/ie-vsan-01/hosts> ls
 0 cs-ie-h01.ie.local (host): cpu 2*12*2.67 GHz, memory 103.00 GB
 1 cs-ie-h02.ie.local (host): cpu 2*12*2.67 GHz, memory 103.00 GB
 2 cs-ie-h03.ie.local (host): cpu 2*12*2.67 GHz, memory 103.00 GB
 3 cs-ie-h04.ie.local (host): cpu 2*12*2.67 GHz, memory 103.00 GB
```

Note the use of numeric reference for short cuts. Also note that the first part of the vsan command can be typed, and then by hitting the <tab> key twice, command-completion occurs.

This method provides a list of all commands that begin with "vsan.". The first output is the set of commands taken from a 5.5U2 distribution.

```
> vsan.<tab><tab>
 vsan.apply_license_to_cluster    vsan.host_info
 vsan.check_limits                vsan.host_wipe_vsan_disks
 vsan.check_state                 vsan.lldpnetmap
 vsan.clear_disks_cache           vsan.obj_status_report
 vsan.cluster_change_autoclaim    vsan.object_info
 vsan.cluster_info                vsan.object_reconfigure
 vsan.cluster_set_default_policy  vsan.observer
 vsan.cmmds_find                  vsan.reapply_vsan_vmknic_config
 vsan.disable_vsan_on_cluster     vsan.recover_spbm
 vsan.disk_object_info            vsan.resync_dashboard
 vsan.disks_info                  vsan.support_information
 vsan.disks_stats                 vsan.vm_object_info
 vsan.enable_vsan_on_cluster      vsan.vm_perf_stats
 vsan.enter_maintenance_mode      vsan.vmdk_stats
 vsan.fix_renamed_vms             vsan.whatif_host_failures
 vsan.host_consume_disks
>
```

This next set of commands comes from a 6.0 distribution:

```
 vsan.apply_license_to_cluster      vsan.host_wipe_non_vsan_disk
 vsan.check_limits                  vsan.host_wipe_vsan_disks
 vsan.check_state                   vsan.lldpnetmap
 vsan.clear_disks_cache             vsan.obj_status_report
 vsan.cluster_change_autoclaim      vsan.object_info
 vsan.cluster_change_checksum       vsan.object_reconfigure
 vsan.cluster_info                  vsan.observer
 vsan.cluster_set_default_policy    vsan.observer_process_statsfile
 vsan.cmmds_find                    vsan.proactive_rebalance
 vsan.disable_vsan_on_cluster       vsan.proactive_rebalance_info
 vsan.disk_object_info              vsan.reapply_vsan_vmknic_config
 vsan.disks_info                    vsan.recover_spbm
 vsan.disks_stats                   vsan.resync_dashboard
 vsan.enable_vsan_on_cluster        vsan.scrubber_info
 vsan.enter_maintenance_mode        vsan.support_information
 vsan.fix_renamed_vms               vsan.v2_ondisk_upgrade
 vsan.host_claim_disks_differently  vsan.vm_object_info
 vsan.host_consume_disks            vsan.vm_perf_stats
 vsan.host_evacuate_data            vsan.vmdk_stats
 vsan.host_exit_evacuation          vsan.whatif_host_failures
 vsan.host_info
```

6.0 versions have a number of additional commands. These additional commands will be discussed throughout the course of this reference guide. One may also reference the *Virtual SAN and SPBM RVC Command Reference Guide* for additional information. At this point, the first few characters of the rest of the vsan command may be typed, and by hitting the <tab> key once again, the command will complete.

# Using RVC to display adapter information

In an earlier chapter, ESXi commands that were used to gather adapter information were examined. This was done with a view to using this information to check the VMware Compatibility Guide (VCG) and ensuring that the Virtual SAN components are supported. There is also a way to display useful information about adapters using RVC by using the following command.

### vsan.disks_info –show-adapters

The RVC command, `vsan.disks_info --show-adapters`, is useful for displaying information about your disks and flash devices. It also displays information about the adapter and drivers used by the ESXi hosts. Here is an output taken from a host with a PCI-E FusionIO adapter. The device shows up as an SSD in the output (this is from the 5.5 version of the command):

```
+-------------------------------------------------------------+-----+--------+-----------------------+
| Local FUSIONIO Disk (eui.48f8681115d6416c00247172ce4df168) | SSD | 1122GB | inUse                 |
| FUSIONIO IODRIVE                                            |     |        |                       |
|                                                             |     |        | Adapters:             |
|                                                             |     |        | fioiom0 (iomemory-vsl )|
|                                                             |     |        |   Fusion-io ioDrive2  |
+-------------------------------------------------------------+-----+--------+-----------------------+
```

Here is a sample output form a host with uses HP's P410i controller. The driver is "`hpsa`" and the disk device size (MD is short for magnetic disk) is 136GB that is actually a relatively small disk drive; this is from the 6.0 version of the command.

```
+-----------------------------------------+-------+---------+-----------------------------------------+
| HP Serial Attached SCSI Disk            | MD    | 136 GB  | inUse (The disk resides on a            |
| (naa.600508b1001c64816271482a56a48c3c)  |       |         | non-local storage transport:            |
| HP LOGICAL VOLUME                       |       |         |'naa.600508b1001c0cc0ba2a3866cf8e28be'.)|
|                                         |       |         |                                         |
|                                         |       |         | Adapters                                |
|                                         |       |         | vmhba1 (hpsa)                           |
|                                         |       |         |                                         |
|                                         |       |         |   Hewlett-Packard Company               |
|                                         |       |         |   Smart Array P410i                     |
|                                         |       |         |                                         |
|                                         |       |         | Checksum Enabled: false                 |
+-----------------------------------------+-------+---------+-----------------------------------------+
```

# Using RVC to verify Virtual SAN functionality

At this point, the most common RVC commands will be run to check Virtual SAN state and verify functionality. Other commands will be looked at in the later sections of this troubleshooting reference manual.

On a Virtual SAN cluster that has just been deployed, the vSphere web client may report everything functioning normally, but it is low on detail. This next set of commands will display detailed information about the state of the Virtual SAN cluster, and confirm that the cluster is fully operational.

This first command is very useful for getting an overview of the cluster state.

## vsan.cluster_info

This command produces detailed information for each node in the cluster, so for very large clusters, the amount of information produced by the commands starts to get quite large. In this environment, there is a 4-node cluster, but the output is truncated to show the first two hosts only. This output shows if Virtual SAN is enabled, whether the role is master, backup or agent, the UUIDs of the other nodes in the cluster, disk mappings and network information

```
/ie-vcsa-03.ie.local/vsan-dc/computers> vsan.cluster_info 0
2014-11-27 14:44:02 +0000: Fetching host info from cs-ie-h04.ie.local (may take a moment) ...
2014-11-27 14:44:02 +0000: Fetching host info from cs-ie-h03.ie.local (may take a moment) ...
2014-11-27 14:44:02 +0000: Fetching host info from cs-ie-h02.ie.local (may take a moment) ...
2014-11-27 14:44:02 +0000: Fetching host info from cs-ie-h01.ie.local (may take a moment) ...
Host: cs-ie-h02.ie.local
  Product: VMware ESXi 6.0.0 build-2305723
  VSAN enabled: yes
  Cluster info:
    Cluster role: agent
    Cluster UUID: 529ccbe4-81d2-89bc-7a70-a9c69bd23a19
    Node UUID: 54196e13-7f5f-cba8-5bac-001517a69c72
    Member UUIDs: ["54188e3a-84fd-9a38-23ba-001b21168828", "545ca9af-ff4b-fc84-dcee-001f29595f9f",
"5460b129-4084-7550-46e1-0010185def78", "54196e13-7f5f-cba8-5bac-001517a69c72"] (4)
  Node evacuated: no
  Storage info:
    Auto claim: no
    Checksum enforced: no
    Disk Mappings:
      SSD: HP Serial Attached SCSI Disk (naa.600508b1001c577e11dd042e142a583f) - 186 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c19335174d82278dee603) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001ca36381622ca880f3aacd) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001cb2234d6ff4f7b1144f59) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c0cc0ba2a3866cf8e28be) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c07d525259e83da9541bf) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c10548f5105fc60246b4a) - 136 GB, v1
  FaultDomainInfo:
    Not configured
  NetworkInfo:
    Adapter: vmk2 (172.32.0.2)

Host: cs-ie-h03.ie.local
  Product: VMware ESXi 6.0.0 build-2305723
  VSAN enabled: yes
  Cluster info:
    Cluster role: agent
    Cluster UUID: 529ccbe4-81d2-89bc-7a70-a9c69bd23a19
    Node UUID: 5460b129-4084-7550-46e1-0010185def78
    Member UUIDs: ["54188e3a-84fd-9a38-23ba-001b21168828", "545ca9af-ff4b-fc84-dcee-001f29595f9f",
"5460b129-4084-7550-46e1-0010185def78", "54196e13-7f5f-cba8-5bac-001517a69c72"] (4)
  Node evacuated: no
  Storage info:
    Auto claim: no
    Checksum enforced: no
    Disk Mappings:
      SSD: HP Serial Attached SCSI Disk (naa.600508b1001c9c8b5f6f0d7a2be44433) - 186 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001ceefc4213ceb9b51c4be4) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001cd259ab7ef213c87eaad7) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c2b7a3d39534ac6beb92d) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001cb11f3292fe743a0fd2e7) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c1a7f310269ccd51a4e83) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c9b93053e6dc3ea9bf3ef) - 136 GB, v1
  FaultDomainInfo:
    Not configured
  NetworkInfo:
    Adapter: vmk2 (172.32.0.3)

        <<truncated>>
```

This is a useful command to get a "big picture" of the cluster. Useful information such as the number of nodes in the cluster (4 as per Member UUIDs) is displayed.

This command in 6.0 has some additional information not in the 5.5 versions; namely information on whether the node is evacuated and whether fault domains has been configured.

Note: Although the output also reports emulated checksums in version 6.0, emulated checksums are not yet supported.

### A note of fault domains

In Virtual SAN 5.5, when deploying a virtual machine with a *NumberOfFailuresToTolerate = 1*, there were 2n + 1 hosts required (where n = NumberOfFailuresToTolerate. This meant that if we wished to tolerate 1 failure, 3 ESXi hosts were required. If we wished to tolerate 2 failures, then 5 hosts were required and if we wished the virtual machine to tolerate 3 failures (maximum), then 7 hosts were required.

| *NumberOfFailuresToTolerate* | Number of hosts required |
|---|---|
| 1 | 3 |
| 2 | 5 |
| 3 | 7 |

The same hosts true in Virtual SAN 6.0 when fault domains are not enabled. However if fault domains are enabled, this allows hosts to be grouped together to form a fault domain. What this means is that no two copies/replicas of the virtual machine's data will be placed in the same fault domain. To calculate the number of fault domains required to tolerate failures, we can use the same equation as before; when deploying a virtual machine with a *NumberOfFailuresToTolerate = 1* on a cluster with fault domains, 2n + 1 fault domains (containing 1 or more hosts is required).

| *NumberOfFailuresToTolerate* | Number of fault domains required |
|---|---|
| 1 | 3 |
| 2 | 5 |
| 3 | 7 |

This is possibly the most important of all the commands for checking the state of the cluster.

### vsan.check_state

There are 3 checks that this command does:

- Check for inaccessible Virtual SAN objects
- Check for invalid/inaccessible VMs
- Check for VMs for which VC/hostd/vmx are out of sync

Inaccessible Virtual SAN objects are an indication that there is probably a failure somewhere in the cluster, but that Virtual SAN is still able to track the virtual machine. An invalid or inaccessible object is when the VM has objects that have lost the majority of its components or votes, again due to hardware failures. Note that for a VM's object to be accessible, it must have a full, intact mirror and greater than 50% of its components/votes available.

The next check is for invalid or inaccessible VMs. These are VMs that, most likely due to the fact that the failure(s) that have occurred in the cluster, have been impacted so much that it is no longer accessible by the vCenter server or the ESXi hosts. This is likely be due to the fact that the VM Home Namespace, where the .vmx file resides, is no longer online. Common causes are clusters that have had multiple failures, but the virtual machines have been configured to tolerate only one failure, or network outages.

Finally, we ensure that the vCenter Server and the ESXi hosts are in agreement with regards to the state of the cluster.

If everything is ok, then the output should be similar to the following:

```
/localhost/ie-datacenter-01/computers> ls
0 ie-vsan-01 (cluster): cpu 109 GHz, memory 330 GB
/localhost/ie-datacenter-01/computers> vsan.check_state 0
2014-10-19 16:03:39 +0000: Step 1: Check for inaccessible VSAN objects
Detected 0 objects to be inaccessible

2014-10-19 16:03:39 +0000: Step 2: Check for invalid/inaccessible VMs

2014-10-19 16:03:39 +0000: Step 3: Check for VMs for which VC/hostd/vmx are out of sync
Did not find VMs for which VC/hostd/vmx are out of sync

/localhost/ie-datacenter-01/computers>
```

For completeness, here is an example of such an output when there are inaccessible objects:

```
/ie-vcsa-03.ie.local/vsan-dc/computers> ls
0 vsan (cluster): cpu 109 GHz, memory 329 GB
/ie-vcsa-03.ie.local/vsan-dc/computers> vsan.check_state vsan
2014-11-27 14:51:24 +0000: Step 1: Check for inaccessible VSAN objects
Detected 19 objects to be inaccessible
Detected 34723e54-7840-c72e-42a5-0010185def78 on cs-ie-h02.ie.local to be inaccessible
Detected 4a743e54-f452-4435-1d15-001f29595f9f on cs-ie-h02.ie.local to be inaccessible
```

```
Detected 3a743e54-a8c2-d13d-6d0c-001f29595f9f on cs-ie-h02.ie.local to be inaccessible
Detected 6e713e54-4819-af51-edb5-0010185def78 on cs-ie-h02.ie.local to be inaccessible
Detected 2d6d3e54-848f-3256-b7d0-001b21168828 on cs-ie-h02.ie.local to be inaccessible
Detected f0703e54-4404-c85b-0742-001f29595f9f on cs-ie-h02.ie.local to be inaccessible
Detected 76723e54-74a3-0075-c1a9-001b21168828 on cs-ie-h02.ie.local to be inaccessible
Detected e4c33b54-1824-537c-472e-0010185def78 on cs-ie-h02.ie.local to be inaccessible
Detected ef713e54-186d-d77c-bf27-001b21168828 on cs-ie-h02.ie.local to be inaccessible
Detected 77703e54-0420-3a81-dc1a-001f29595f9f on cs-ie-h02.ie.local to be inaccessible
Detected 30af3e54-24fe-4699-f300-001b21168828 on cs-ie-h02.ie.local to be inaccessible
Detected 58723e54-047e-86a0-4803-001b21168828 on cs-ie-h02.ie.local to be inaccessible
Detected 85713e54-dcbe-fea6-8205-001b21168828 on cs-ie-h02.ie.local to be inaccessible
Detected c2733e54-ac02-78ca-f0ce-001f29595f9f on cs-ie-h02.ie.local to be inaccessible
Detected 94713e54-08e1-18d3-ffd7-001b21168828 on cs-ie-h02.ie.local to be inaccessible
Detected f0723e54-18d2-79f5-be44-001b21168828 on cs-ie-h02.ie.local to be inaccessible
Detected 3b713e54-9851-31f6-2679-001f29595f9f on cs-ie-h02.ie.local to be inaccessible
Detected fd743e54-1863-c6fb-1845-001f29595f9f on cs-ie-h02.ie.local to be inaccessible
Detected 94733e54-e81c-c3fe-8bfc-001b21168828 on cs-ie-h02.ie.local to be inaccessible

2014-11-27 14:51:25 +0000: Step 2: Check for invalid/inaccessible VMs

2014-11-27 14:51:25 +0000: Step 3: Check for VMs for which VC/hostd/vmx are out of sync
Did not find VMs for which VC/hostd/vmx are out of sync

/ie-vcsa-03.ie.local/vsan-dc/computers>
```

Shortly, we will look at how to troubleshoot the "root cause" of such as issue. For the moment, the focus is ensuring that there is a correctly configured system.

## vsan.check_limits

This next command is useful for ensuring that Virtual SAN is operating within its resource limits. The command once again runs against a cluster object, as shown here. The first output is taken from a 5.5U2 cluster.

```
/localhost/ie-datacenter-01/computers> ls
 0 ie-vsan-01 (cluster): cpu 109 GHz, memory 330 GB
 /localhost/ie-datacenter-01/computers> vsan.check_limits 0
 2014-10-19 16:21:06 +0000: Gathering stats from all hosts ...
 2014-10-19 16:21:08 +0000: Gathering disks info ...
 2014-10-19 16:21:08 +0000: Fetching VSAN disk info from cs-ie-h04 (may take a moment) ...
 2014-10-19 16:21:08 +0000: Fetching VSAN disk info from cs-ie-h03 (may take a moment) ...
 2014-10-19 16:21:08 +0000: Fetching VSAN disk info from cs-ie-h01 (may take a moment) ...
 2014-10-19 16:21:08 +0000: Fetching VSAN disk info from cs-ie-h02 (may take a moment) ...
 2014-10-19 16:21:10 +0000: Done fetching VSAN disk infos
 +--------------------+--------------------+-----------------------------------------+
 | Host               | RDT                | Disks                                   |
 +--------------------+--------------------+-----------------------------------------+
 | cs-ie-h01.ie.local | Assocs: 256/20000  | Components: 75/3000                     |
 |                    | Sockets: 134/10000 | naa.600508b1001c388c92e817e43fcd5237: 7%  |
 |                    | Clients: 36        | naa.600508b1001c79748e8465571b6f4a46: 16% |
 |                    | Owners: 36         | eui.48f8681115d6416c00247172ce4df168: 0%  |
 |                    |                    | naa.600508b1001c16be6e256767284eaf88: 2%  |
 |                    |                    | naa.600508b1001c64816271482a56a48c3c: 3%  |
 |                    |                    | naa.600508b1001c3ea7838c0436dbe6d7a2: 3%  |
 |                    |                    | naa.600508b1001ccd5d506e7ed19c40a64c: 2%  |
 |                    |                    | naa.600508b1001c2ee9a6446e708105054b: 3%  |
 | cs-ie-h02.ie.local | Assocs: 76/20000   | Components: 75/3000                     |
 |                    | Sockets: 79/10000  | eui.c68e151fed8a4fcf0024712c7cc444fe: 0%  |
 |                    | Clients: 0         | naa.600508b1001c07d525259e83da9541bf: 3%  |
 |                    | Owners: 0          | naa.600508b1001cb2234d6ff4f7b1144f59: 3%  |
 |                    |                    | naa.600508b1001c10548f5105fc60246b4a: 3%  |
 |                    |                    | naa.600508b1001c4c10c1575c80870cce38: 3%  |
 |                    |                    | naa.600508b1001ca36381622ca880f3aacd: 3%  |
 |                    |                    | naa.600508b1001c0cc0ba2a3866cf8e28be: 3%  |
 |                    |                    | naa.600508b1001c19335174d82278dee603: 3%  |
 | cs-ie-h03.ie.local | Assocs: 179/20000  | Components: 75/3000                     |
 |                    | Sockets: 108/10000 | naa.600508b1001c2b7a3d39534ac6beb92d: 2%  |
 |                    | Clients: 13        | naa.600508b1001c1a7f310269ccd51a4e83: 3%  |
 |                    | Owners: 21         | eui.d1ef5a5bbe864e27002471febdec3592: 0%  |
 |                    |                    | naa.600508b1001c492d637ba41212feff13: 2%  |
 |                    |                    | naa.600508b1001c9b93053e6dc3ea9bf3ef: 2%  |
 |                    |                    | naa.600508b1001cd259ab7ef213c87eaad7: 3%  |
 |                    |                    | naa.600508b1001cb11f3292fe743a0fd2e7: 5%  |
 |                    |                    | naa.600508b1001ceefc4213ceb9b51c4be4: 22% |
 | cs-ie-h04.ie.local | Assocs: 265/20000  | Components: 75/3000                     |
 |                    | Sockets: 134/10000 | naa.600508b1001c4d41121b41182fa83be4: 3%  |
 |                    | Clients: 36        | naa.600508b1001c846c000c3d9114ed71b3: 3%  |
 |                    | Owners: 39         | eui.a15eb52c6f4043b5002471c7886acfaa: 0%  |
 |                    |                    | naa.600508b1001cadff5d80ba7665b8f09a: 5%  |
 |                    |                    | naa.600508b1001cc426a15528d121bbd880: 3%  |
 |                    |                    | naa.600508b1001c258181f0a088f6e40dab: 3%  |
 |                    |                    | naa.600508b1001c51f3a696fe0bbbcb5096: 3%  |
 |                    |                    | naa.600508b1001c4b820b4d80f9f8acfa95: 3%  |
 +--------------------+--------------------+-----------------------------------------+
 /localhost/ie-datacenter-01/computers>
```

This is an output taken from the same cluster upgraded to version 6.0.

```
/ie-vcsa-03.ie.local/vsan-dc/computers> vsan.check_limits 0
 2014-11-27 14:52:25 +0000: Querying limit stats from all hosts ...
 2014-11-27 14:52:27 +0000: Fetching VSAN disk info from cs-ie-h03 (may take a moment) ...
 2014-11-27 14:52:27 +0000: Fetching VSAN disk info from cs-ie-h02 (may take a moment) ...
 2014-11-27 14:52:27 +0000: Fetching VSAN disk info from cs-ie-h01 (may take a moment) ...
 2014-11-27 14:52:27 +0000: Fetching VSAN disk info from cs-ie-h04 (may take a moment) ...
 2014-11-27 14:52:29 +0000: Done fetching VSAN disk infos
 +--------------------+--------------------+-----------------------------------------+
 | Host               | RDT                | Disks                                   |
 +--------------------+--------------------+-----------------------------------------+
 | cs-ie-h02.ie.local | Assocs: 139/45000  | Components: 116/3000                    |
 |                    | Sockets: 102/10000 | naa.600508b1001c19335174d82278dee603: 3%  |
 |                    | Clients: 0         | naa.600508b1001c10548f5105fc60246b4a: 8%  |
 |                    | Owners: 20         | naa.600508b1001cb2234d6ff4f7b1144f59: 10% |
 |                    |                    | naa.600508b1001c577e11dd042e142a583f: 0%  |
 |                    |                    | naa.600508b1001c0cc0ba2a3866cf8e28be: 26% |
 |                    |                    | naa.600508b1001ca36381622ca880f3aacd: 12% |
 |                    |                    | naa.600508b1001c07d525259e83da9541bf: 2%  |
 | cs-ie-h03.ie.local | Assocs: 45/45000   | Components: 0/3000                      |
 |                    | Sockets: 40/10000  | naa.600508b1001c1a7f310269ccd51a4e83: 0%  |
 |                    | Clients: 4         | naa.600508b1001c9b93053e6dc3ea9bf3ef: 0%  |
 |                    | Owners: 12         | naa.600508b1001cb11f3292fe743a0fd2e7: 0%  |
 |                    |                    | naa.600508b1001c9c8b5f6f0d7a2be44433: 0%  |
 |                    |                    | naa.600508b1001ceefc4213ceb9b51c4be4: 0%  |
 |                    |                    | naa.600508b1001c2b7a3d39534ac6beb92d: 0%  |
 |                    |                    | naa.600508b1001cd259ab7ef213c87eaad7: 0%  |
 | cs-ie-h04.ie.local | Assocs: 502/45000  | Components: 97/3000                     |
 |                    | Sockets: 187/10000 | naa.600508b1001c4b820b4d80f9f8acfa95: 6%  |
 |                    | Clients: 75        | naa.600508b1001c846c000c3d9114ed71b3: 3%  |
 |                    | Owners: 84         | naa.600508b1001cadff5d80ba7665b8f09a: 4%  |
 |                    |                    | naa.600508b1001c4d41121b41182fa83be4: 3%  |
 |                    |                    | naa.600508b1001c40e393b73af79eacdcde: 0%  |
 |                    |                    | naa.600508b1001c51f3a696fe0bbbcb5096: 4%  |
 |                    |                    | naa.600508b1001c258181f0a088f6e40dab: 4%  |
 | cs-ie-h01.ie.local | Assocs: 98/45000   | Components: 97/3000                     |
 |                    | Sockets: 101/10000 | naa.600508b1001c388c92e817e43fcd5237: 4%  |
 |                    | Clients: 0         | naa.600508b1001c64816271482a56a48c3c: 2%  |
 |                    | Owners: 0          | naa.600508b1001c79748e8465571b6f4a46: 2%  |
 |                    |                    | naa.600508b1001c61cedd42b0c3fbf55132: 0%  |
 |                    |                    | naa.600508b1001c3ea7838c0436dbe6d7a2: 16% |
 |                    |                    | naa.600508b1001c2ee9a6446e708105054b: 3%  |
 |                    |                    | naa.600508b1001ccd5d506e7ed19c40a64c: 1%  |
 |                    |                    | naa.600508b1001c16be6e256767284eaf88: 11% |
 +--------------------+--------------------+-----------------------------------------+
 /ie-vcsa-03.ie.local/vsan-dc/computers>
```

There are two columns shown in the output. RDT relates to networking limits and Disks relates to storage limits. RDT is Reliable Datagram Transport and is the Virtual SAN network transport. RDT has a number of limits listed. These are Associations (Assocs) and Sockets. Additional information regarding Clients and Owners is also displayed. The number of supported associations has more than doubled between version 5.5 and 6.0.

### Brief explanation on RDT Assocs/Sockets/Client/Owners

**Assocs**: An RDT association is used to track peer-to-peer network state within Virtual SAN.

**Sockets**: Virtual SAN limits how many TCP sockets it is allowed to use.

**Clients**: A Virtual SAN client represents the state on a host to access a Virtual SAN object in the Virtual SAN cluster. There is no hard defined limit, but this metric is shown to understand balance across hosts.

**Owners**: There is always one Virtual SAN owner for a given Virtual SAN object, typically co-located with the Virtual SAN client that is accessing this object (if just one). Virtual SAN owners coordinate all access to the Virtual SAN object and implement functionality like RAID. There is no hard defined limit, but this metric is shown to understand balance across hosts.

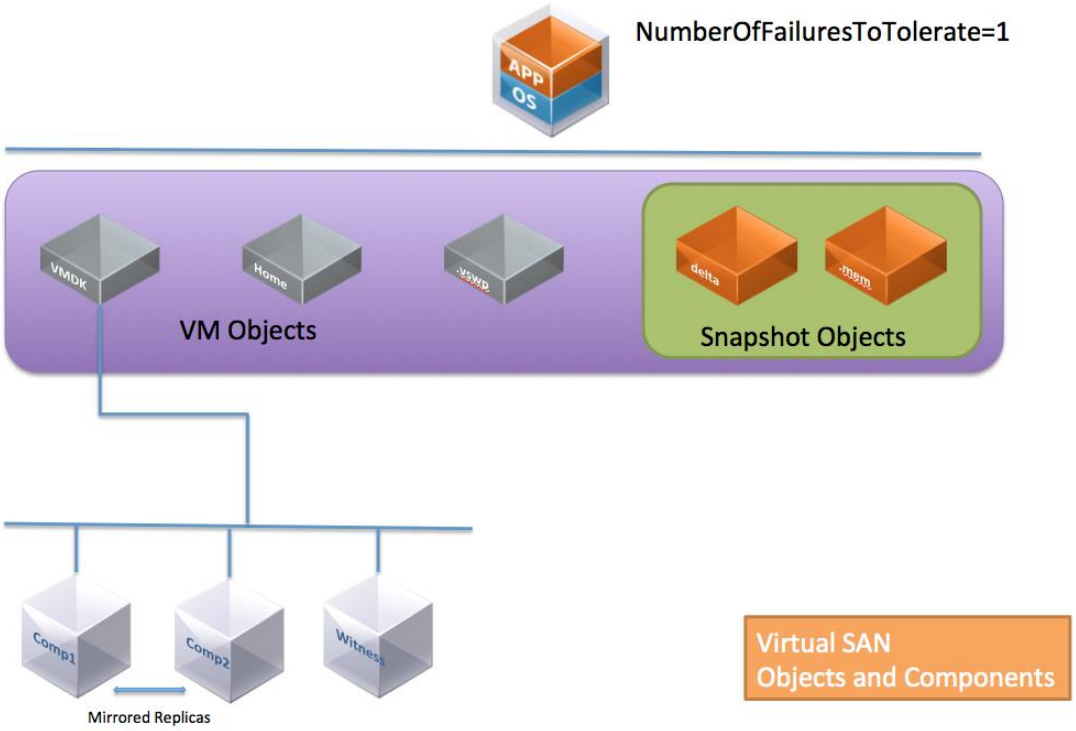
### Brief explanation of disk components revisited

For disks, in Virtual SAN 5.5, there is a limit of 3,000 components. The output displays the current component count and the percentage of disk being consumed.

In the Virtual SAN 6.0 output, the number of disk components is still shown as 3,000, although we support 9,000 components per host in 6.0. The reason it is shown as 3,000 is because the cluster has not yet been updated to the v2 on-disk format; the VSAN datastore is still running v1 on-disk format, which is the same format as version 5.5.

It is worth revisiting components at this point as we will be looking at objects and components closely in many of the upcoming command outputs.

A virtual machine deployed on a Virtual SAN datastore is comprised of a set of objects. These are the VM Home Namespace, the VMDK, VM Swap when the VM is powered on and Snapshot Delta VMDKs when snapshots are taken on the VM. New to version 6.0 is the virtual machine snapshot memory, which is now instantiated as its own object when a virtual machine snapshot is taken and memory is also selected:




Each of the objects is comprised of a set of components. For example, if *NumberOfFailuresToTolerate=1* is set in the VM Storage Policy, then some of the objects would be mirrored/replicated, with each replica (and witness) being a component.

If *NumberOfDiskStripesPerObject* is greater than 1 in the VM Storage Policy, then each stripe is also said to be a component of the object.

## Understanding components and component count

***Note**: The component limit per host was 3,000 for Virtual SAN version 5.5. Virtual SAN Version 6.0 increased this limit count to 9,000.*

The disk limits are worth noting regularly via the `vsan.check_limits` RVC command to ensure that the limits are not being reached.

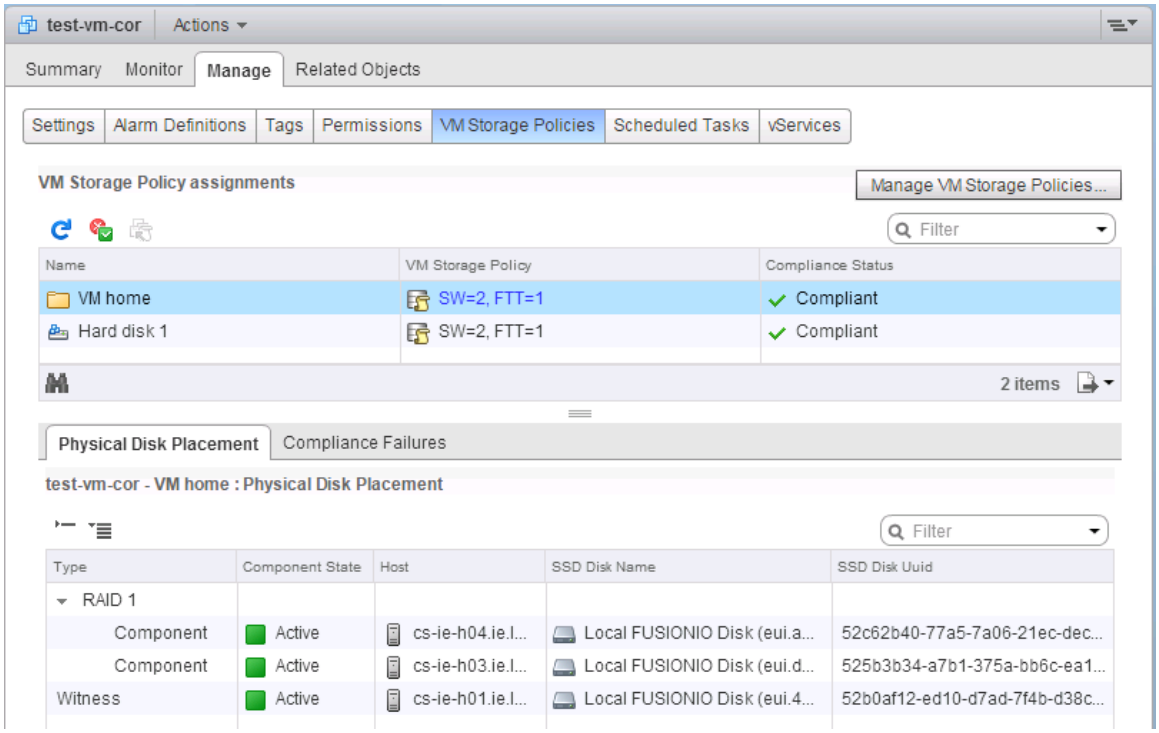
In this example, a test VM using the following VM Storage Policy settings has been deployed:

- Number of Failures to Tolerate (FTT) = 1
- Number of Disk Objects to Stripe (SW) = 2

The way this virtual machine's object are deployed and distributed across the Virtual SAN cluster is looked at next, as well as verifying that the policy has indeed taken effect.

## Examining components via the vSphere web client

One can examine some of the components via the vSphere web client, and in this example (using vSphere 5.5), they show the following layout for the VM Home Namespace and the VMDK. Here is the VM Home Namespace object and its components:

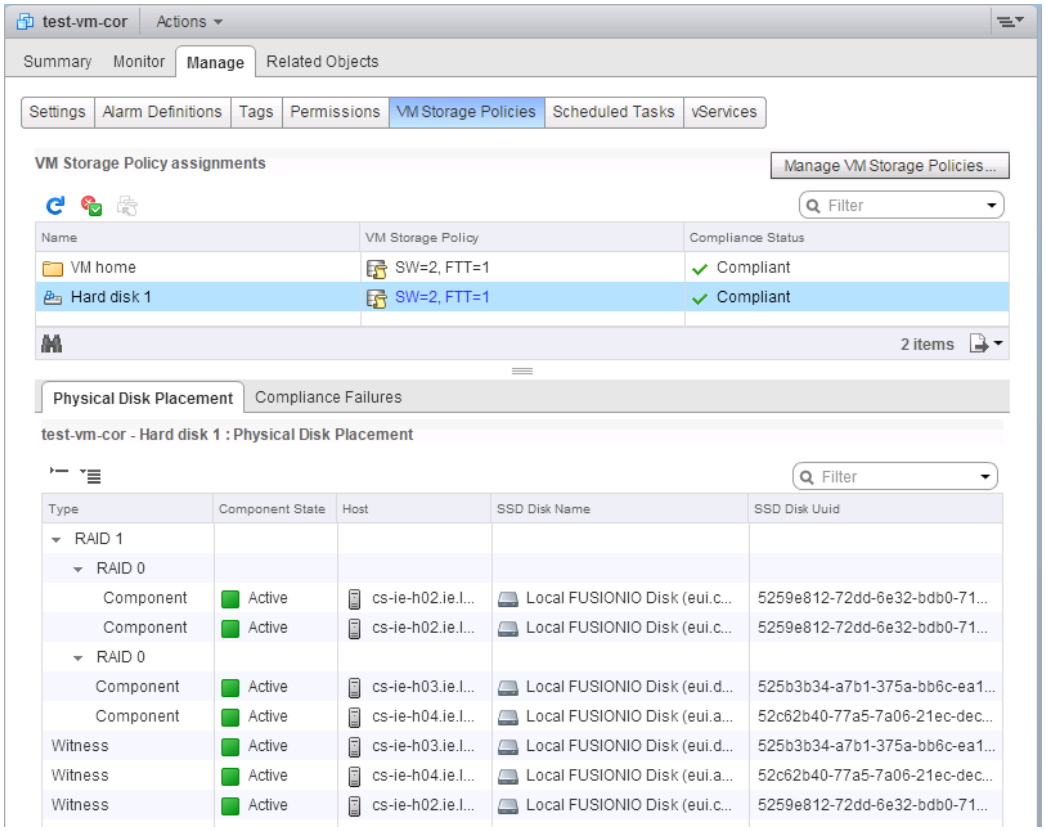


Note that the VM Home Namespace does not implement the Stripe Width policy setting, thus we only see a single component in the RAID-1 mirror/replica; RAID-1 is used for implementing the Failures To Tolerate policy setting.

There is also a single component called a Witness to help the object to remain accessible in the event of a failure. An object will remain accessible so long as:

1. There is a full replica/mirror copy without any failing components available
2. Greater than 50% of the components or votes within an object are available, which includes the witness.

This is the VMDK object and its components as it appears in version 5.5 of the vSphere web client.




The VMDK object shown here implements both the Failures to Tolerate and Stripe Width settings. The RAID-1 has got two mirrors once again, but now each side of the mirror is made up of a RAID-0 to represent the stripe. It should be noted that stripe width is a minimum value; there may be more stripes than what was placed in the VM Storage Policy.

If the VMDK storage object is too big to fit onto the free space on a single hard disk, then Virtual SAN will automatically stripe it across multiple hard disks. Also note

that since there are now more components to manage, additional witness are required to maintain a majority of components in the object in the event of a failure.

## vsan.vm_object_info

By using the `vsan.vm_object_info` command, the objects and component layout of the new VM can now be examined in great detail. Look at how RVC report the objects and components (this command and output is the same in version 5.5 and 6.0):

```
/localhost/ie-datacenter-01/vms> ls
 0 test-vm-cor: poweredOff
 /localhost/ie-datacenter-01/vms> vsan.vm_object_info 0
 VM test-vm-cor:
   Namespace directory
     DOM Object: fdeb4354-40f7-bace-a8f5-001f29595f9f (owner: cs-ie-h03.ie.local, policy:
hostFailuresToTolerate = 1, stripeWidth = 1, spbmProfileId = c1421e53-ded2-4ea1-843d-
ff6149e03525, proportionalCapacity = [0, 100], spbmProfileGenerationNumber = 0)
       Witness: feeb4354-e85e-c34b-ef73-001f29595f9f (state: ACTIVE (5), host: cs-ie-
h01.ie.local,           md:           naa.600508b1001c16be6e256767284eaf88,           ssd:
eui.48f8681115d6416c00247172ce4df168,
                                                    usage: 0.0 GB)
       RAID_1
         Component: feeb4354-58c7-c24b-ef14-001f29595f9f (state: ACTIVE (5), host: cs-ie-
h03.ie.local,           md:           naa.600508b1001cd259ab7ef213c87eaad7,           ssd:
eui.d1ef5a5bbe864e27002471febdec3592,
                                                      usage: 0.1 GB)
         Component: feeb4354-ccba-c14b-2af4-001f29595f9f (state: ACTIVE (5), host: cs-ie-
h04.ie.local,           md:           naa.600508b1001cadff5d80ba7665b8f09a,           ssd:
eui.a15eb52c6f4043b5002471c7886acfaa,
                                                      usage: 0.1 GB)
   Disk backing: [vsanDatastore] fdeb4354-40f7-bace-a8f5-001f29595f9f/test-vm-cor.vmdk
     DOM Object: 00ec4354-aca5-66fc-4615-001f29595f9f (owner: cs-ie-h03.ie.local, policy:
spbmProfileGenerationNumber = 0, stripeWidth = 2, spbmProfileId = c1421e53-ded2-4ea1-
843d-ff6149e03525, hostFailuresToTolerate = 1)
       Witness: 01ec4354-6c9c-2f29-8b11-001f29595f9f (state: ACTIVE (5), host: cs-ie-
h02.ie.local,           md:           naa.600508b1001cb2234d6ff4f7b1144f59,           ssd:
eui.c68e151fed8a4fcf0024712c7cc444fe,
                                                    usage: 0.0 GB)
       Witness: 01ec4354-2c2c-2e29-1f43-001f29595f9f (state: ACTIVE (5), host: cs-ie-
h03.ie.local,           md:           naa.600508b1001cb11f3292fe743a0fd2e7,           ssd:
eui.d1ef5a5bbe864e27002471febdec3592,
                                                    usage: 0.0 GB)
       Witness: 01ec4354-48e9-2e29-6eb3-001f29595f9f (state: ACTIVE (5), host: cs-ie-
h04.ie.local,           md:           naa.600508b1001cadff5d80ba7665b8f09a,           ssd:
eui.a15eb52c6f4043b5002471c7886acfaa,
                                                    usage: 0.0 GB)
       RAID_1
         RAID_0
           Component: 01ec4354-20fb-2c29-a6b9-001f29595f9f (state: ACTIVE (5), host: cs-
ie-h04.ie.local,          md:          naa.600508b1001c258181f0a088f6e40dab,          ssd:
eui.a15eb52c6f4043b5002471c7886acfaa,
                                                      usage: 0.0 GB)
           Component: 01ec4354-4418-2c29-0dbd-001f29595f9f (state: ACTIVE (5), host: cs-
ie-h03.ie.local,          md:          naa.600508b1001cd259ab7ef213c87eaad7,          ssd:
eui.d1ef5a5bbe864e27002471febdec3592,
                                                      usage: 0.0 GB)
         RAID_0
           Component: 01ec4354-c04e-2b29-1130-001f29595f9f (state: ACTIVE (5), host: cs-
ie-h02.ie.local,          md:          naa.600508b1001c4c10c1575c80870cce38,          ssd:
eui.c68e151fed8a4fcf0024712c7cc444fe,
                                                      usage: 0.0 GB)
           Component: 01ec4354-64ed-2929-4cae-001f29595f9f (state: ACTIVE (5), host: cs-
ie-h02.ie.local,          md:          naa.600508b1001cb2234d6ff4f7b1144f59,          ssd:
eui.c68e151fed8a4fcf0024712c7cc444fe,
                                                      usage: 0.0 GB)
```

The interesting parts are highlighted in bold. There are two objects visible, the VM Home Namespace and the VMDK (referred to as Disk backing in the above output). Again, the VM Home is using a *StripeWidth=1*, where the VMDK is using a *StripeWidth=2*. Therefore the VMDK has RAID_0 configurations in each replica, whereas the VM Home does not.

Another important point is that all components are ACTIVE. There are no components in an STALE, ABSENT or DEGRADED state.

### vsan.object_info

If you note the outputs from `vsan.vm_object_info`, a "DOM object" reference is displayed. DOM is short for Distributed Object Manager and is a core component of Virtual SAN that implements the RAID configuration. Using this "DOM object" information, one can ask Virtual SAN to display similar information about the object via a different RVC command called `vsan.object_info`. Note that this command must be run against a cluster and not against a virtual machine as was the case with the previous command.

```
/localhost/ie-datacenter-01> cd 1
/localhost/ie-datacenter-01/computers> ls
 0 ie-vsan-01 (cluster): cpu 109 GHz, memory 330 GB
/localhost/ie-d atacenter-01/computers> vsan.object_info 0 fdeb4354-40f7-bace-a8f5-
001f29595f9f
 DOM Object: fdeb4354-40f7-bace-a8f5-001f29595f9f (owner: cs-ie-h01.ie.local, policy:
hostFailuresToTolerate = 1, stripeWidth = 1, spbmProfileId = c1421e53-ded2-4ea1-843d-
ff6149e03525, proportionalCapacity = [0, 100], spbmProfileGenerationNumber = 0)

   Witness: feeb4354-e85e-c34b-ef73-001f29595f9f (state: ACTIVE (5), host: cs-ie-
h01.ie.local, md: naa.600508b1001c16be6e256767284eaf88, ssd:
eui.48f8681115d6416c00247172ce4df168, usage: 0.0 GB)

   RAID_1
     Component: feeb4354-58c7-c24b-ef14-001f29595f9f (state: ACTIVE (5), host: cs-ie-
h03.ie.local, md: naa.600508b1001cd259ab7ef213c87eaad7, ssd:
eui.d1ef5a5bbe864e27002471febdec3592, usage: 0.1 GB)

     Component: feeb4354-ccba-c14b-2af4-001f29595f9f (state: ACTIVE (5), host: cs-ie-
h04.ie.local, md: naa.600508b1001cadff5d80ba7665b8f09a, ssd:
eui.a15eb52c6f4043b5002471c7886acfaa, usage: 0.1 GB)

   Extended attributes:
     Address space: 273804165120B (255.00 GB)
     Object class: vmnamespace
     Object path: /vmfs/volumes/vsan:52dc5a95d04bcbb9-9d90f486c2f14d1d/

 /localhost/ie-datacenter-01/computers>
```

### vsan.whatif_host_failures

This is a very useful RVC command for determining if there are enough resources remaining in the cluster to rebuild the missing components in the event of a failure. The command and output is the same in versions 5.5 and 6.0. The HDD capacity reported below refers to the capacity layer, both for all-flash and hybrid. RC reservations refers to read cache reservations, an option that allows an administrator to dedicate a certain amount of read cache to a virtual machine through VM storage policy settings, but it is only relevant to hybrid configurations as there is no read cache reservation setting in all-flash configurations.

There are no 'read cache reservations' in this example.

```
/localhost/ie-datacenter-01/computers> vsan.whatif_host_failures 0
 Simulating 1 host failures:

 +-----------------+-----------------------------+-----------------------------------+
 | Resource        | Usage right now             | Usage after failure/re-protection |
 +-----------------+-----------------------------+-----------------------------------+
 | HDD capacity    |   5% used (3635.62 GB free) |   7% used (2680.12 GB free)       |
 | Components      |   3% used (11687 available) |   3% used (8687 available)        |
 | RC reservations |   0% used (3142.27 GB free) |   0% used (2356.71 GB free)       |
 +-----------------+-----------------------------+-----------------------------------+
 /localhost/ie-datacenter-01/computers>
```

At this point, not only is the Virtual SAN cluster functioning, but it is also deploying virtual machines correctly, according to the policy settings.

After running just a few RVC commands (`vsan.cluster_info`, `vsan.check_state`, `vsan.check_limits`, `vsan.vm_object_info`, `vsan.object_info`), it is possible to confirm at a low level that the Virtual SAN cluster is operational.

More RVC commands will be examined when we look at troubleshooting failures scenarios later in this reference manual, and it should become clear how RVC commands can help you to locate the root cause of errors within the cluster.

# 10. Troubleshooting the Virtual SAN Network

In this section, the Virtual SAN network is introduced, and the types of issues that can arise from a misconfigured Virtual SAN network. This chapter shows how to troubleshoot these issues.

Virtual SAN is entirely dependent on the network: its configuration, reliability, performance, etc. One of the most frequent causes of requesting support is either an incorrect network configuration, or the network not performing as expected.

**Note**: There is no support for IPv6 in Virtual SAN 5.5 or 6.0.

As highlighted in the introduction, VMware recommends using the Virtual SAN Health Services to do initial triage of network issues. The Virtual SAN Health Services carry out a range of network health checks, and direct administrators to an appropriate knowledge base article depending on the results of the health check. The knowledge base article will provide administrators with step-by-step instruction to solve the network problem at hand.

Please refer to the *Virtual SAN Health Services Guide* for further details on how to get the Health Services components, how to install them and how to use the feature for troubleshooting common Virtual SAN issues.

# Introduction to Virtual SAN Networking

Before getting into network in detail, it is important to understand the roles that nodes/hosts can play in Virtual SAN. There are three roles in Virtual SAN: master, agent and backup. There is one master that is responsible for getting CMMDS (clustering service) updates from all nodes, and distributing these updates to agents. Roles are applied during cluster discovery, when all nodes participating in Virtual SAN elect a master. A vSphere administrator has no control over roles.

A backup node will assume the master role should the master fail. This will avoid a complete rediscovery of every node, object and component in the cluster, as the backup role will already have a full copy of the directory contents, and can seamlessly assume the role of master, speeding up recovery in the event of a master failure.

There are a number of distinct parts to Virtual SAN networking. First there is the communication that takes place between all of the ESXi hosts in the Virtual SAN cluster, indicating that they are still actively participating in Virtual SAN. This is done via multicast traffic, and a heartbeat is sent from the master to all hosts once every second to ensure they are still active.

There is also communication between the master and the backup and agent nodes, where the master keeps the other nodes up to date with regards to the configuration of the cluster. This is also multicast traffic, but is relatively light from a traffic volume perspective.

Lastly, there is virtual machine disk I/O. This makes up the majority of the traffic on the Virtual SAN network. VMs on the Virtual SAN datastore are made up of a set of objects. These objects may be made up of one or more components, for example a number of RAID-1 stripes or a number of RAID-0 mirrors.

Invariably, a VMs compute and a VM's storage will be located on different ESXi hosts in the cluster. It may also transpire that if a VM has been configured to tolerate one or more failures, the compute may be on node 1, the first RAID-0 mirror may be on host 2 and the second RAID-0 mirror could be on host 3. In this case, disk reads and writes for this virtual machine will have to traverse the Virtual SAN network. This is unicast traffic, and forms the bulk of the Virtual SAN network traffic.

# Virtual SAN network requirements

In this section, the network requirements for Virtual SAN are discussed.

## Physical Network Interface Card (NIC) requirements

VMware will support both 1Gb Network Interface Cards (NICs) and 10Gb Network Interface Cards for Virtual SAN traffic. If 1Gb NICs are planned, VMware recommends that the NIC be dedicated to Virtual SAN traffic. If 10Gb NICs are planned, VMware supports sharing that NIC with other VMware network traffic types (management, vMotion, etc.). Virtual SAN also supports NIC teaming for availability purposes, but be aware that Virtual SAN does not balance Virtual SAN traffic across NICs. NIC teaming is only supported in active/standby mode on the Virtual SAN network.

## Virtual SAN traffic – vmknic requirement

Each ESXi host that wishes to participate in a Virtual SAN Cluster must have a VMkernel interface created [vmknic], and this interface must be configured for Virtual SAN Traffic. This can easily be done and checked via the vSphere web client.

For version 5.5, VMware recommends that the Virtual SAN traffic be isolated to a layer 2 non-routable VLAN. Verify that each of the VMkernel ports used for Virtual SAN traffic are on the same subnet, have the same subnet mask and are on the same VLAN segment.

In version 6.0, VMware supports Virtual SAN traffic over a layer 3, routable network.

A good test to verify that the hosts are able to communicate to each other over this network is to use the `vmkping` command and have each of the hosts in the cluster ping each other. If this fails to work, the network configuration must be revisited and rechecked on all hosts, and there should be no attempt to enable Virtual SAN until the networking is configured and working correctly.

## Virtual switch requirement

Virtual SAN works with both the original standard switch (VSS or vSwitch) and also with the distributed virtual switch (DVS). Virtual SAN also includes a license for distributed switches, which normally requires an Enterprise+ license edition. This means that customers can take advantage of simplified network management provided by the vSphere Distributed Switch for their Virtual SAN storage regardless of the underlying vSphere edition they use.

### MTU & jumbo frames

Virtual SAN fully supports the use of jumbo frames, though it is not thought to improve Virtual SAN network performance to any great extent. However recent performance tests using Virtual SAN for VDI suggests that the use of jumbo frames does assist with reduced CPU utilization. If jumbo frames is required, ensure that the VMkernel interfaces and the virtual switches to which they connect are configured to handle this new size. The physical switches to which the physical NICs are connected also need to be configured appropriately.

A good test to verify that the hosts are able to communicate to each other over this network with jumbo frames is to use the `vmkping` command but with the `-s 9000` option (for the larger packet size) and have each of the hosts in the cluster ping each other. Needless to say that if doesn't work, no attempt should be made to enable Virtual SAN until the network is configured correctly for jumbo frames.

### Multicast traffic requirement

As stated in the introduction, multicast traffic is used for certain Virtual SAN cluster functions. It is imperative that the Virtual SAN network facilitates multicast traffic across all hosts participating in the Virtual SAN cluster. In version 5.5, this must be over a layer-2 network. In 6.0, VMware supports multicast traffic routed over a layer-3 network.

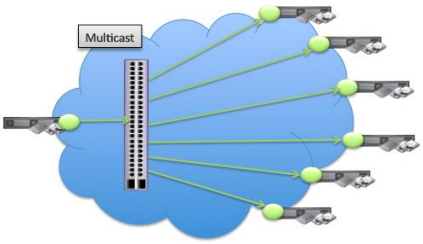
Virtual SAN uses a clustered metadata database and monitoring service (CMMDS) to make particular metadata available to each host in the cluster. The CMMDS is designed to be a highly available, performant and network efficient service that shares information regarding host, network, disks, objects, components, etc. among all of the hosts within the Virtual SAN cluster.

This distribution of metadata amongst all of the ESXi hosts in the cluster, and keeping each host synchronized, could potentially consume a considerable amount of compute resources and network bandwidth. Each ESXi host is intended to contain an identical copy of this metadata which means that if general unicast forwarding was used for this traffic, there would be duplicate traffic sent to all of the ESXi hosts in the cluster.

Virtual SAN leverages multicast forwarding for the discovery of hosts and to optimize network bandwidth consumption for the metadata updates from the CMMDS service. This eliminates the computing resource and network bandwidth penalties that unicast would impose in order to send identical data to multiple recipients.

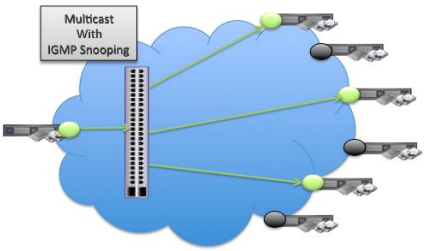
### IGMP snooping and IGMP querier for multicast traffic

Multicast forwarding is a one-to-many or many-to-many distribution of network traffic (as opposed to unicast's one-to-one forwarding scheme). Rather than using the network address of the intended recipient for its destination address, multicast uses a special destination address to logically identify a group of receivers.




Since multicasting allows the data to be sent only once, this has the benefit of freeing up computing resources from the source ESXi host. As was just mentioned, without multicast, that host would need to send individual streams of duplicate data to all of the other ESXi hosts in the cluster. Leveraging switching technology to repeat the message to the members of the multicast group is far more efficient for the host than it is sending each individually.

Many network administrators do not want to enable multicast across network switches, for good reason. This can generate a large volume of traffic on the network, since each network device attached to an active network port would receive the multicast network traffic. To avoid this, many network administrators use IGMP, the Internet Group Management Protocol. IGMP is a communications protocol used to establish multicast group memberships. IGMP Snooping and an IGMP Querier can be leveraged to send multicast traffic to a subset of switch ports that have devices attached that request it. This will avoid causing unnecessary load on other network devices as they will not have to process packets that they have not solicited.




VMware highly recommends the use of IGMP snooping with IGMP Querier which should be available on most physical switches. This will avoid the multicast traffic flooding across all ports of the switch. Multicast traffic can be limited to specific

group of ports using IGMP snooping. The default multicast group addresses for Virtual SAN are:

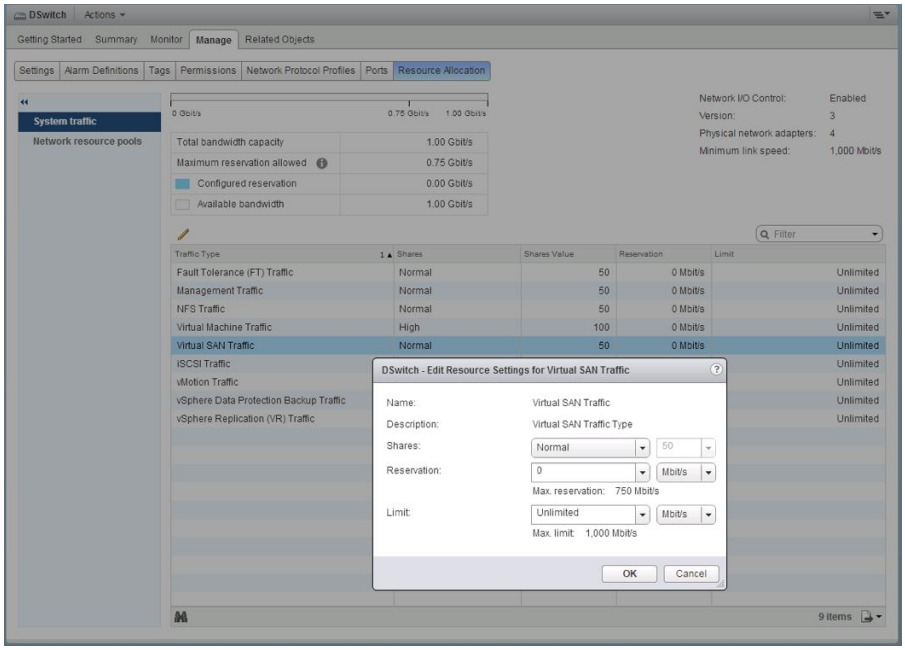
**224.1.2.3** Port: **12345**
**224.2.3.4** Port: **23451**

Port 23451 is used by the master for sending a heartbeat to each host in the cluster every second. Port 12345 is used for the CMMDS updates. Shortly, we will look at how you can test and verify your Virtual SAN networking configuration, including the multicast configuration.

## Using NIOC and VDS to set Quality of Service on Virtual SAN traffic

Network IO Control (NIOC) can be used to guarantee bandwidth for Virtual SAN cluster communication & I/O. While it may not be necessary in some Virtual SAN cluster environments, it is a good feature to have available if Virtual SAN traffic appears to be impacted by other traffic types sharing the same 10GbE network interface card. Setting up NIOC is quite straightforward, and once configured, will guarantee a certain bandwidth for the Virtual SAN traffic between all hosts. Details on how to configure distributed switches and NIOC can be found in the relevant vSphere networking documentation and VSAN Networking Design Guide.

VMware does not recommend setting a limit on the VSAN traffic. By default its unlimited, so it can use all available bandwidth.



# Virtual SAN and vSphere HA network dependency

vSphere HA is fully supported on Virtual SAN cluster to provide additional availability to virtual machines deployed in the cluster. If a host fails, vSphere HA will take responsibility for restarting any VMs that had their compute running on the failed host. Virtual SAN will ensure that the storage objects residing on the failed host are reconfigured elsewhere in the cluster, if resources are available to do so.

There have been a number of changes made to vSphere HA to ensure correct interoperability with Virtual SAN. Notably, vSphere HA agents use the Virtual SAN network for communication when Virtual SAN is also enabled on the cluster. vSphere HA & Virtual SAN must be partitioned in the same way if a network failure occurs. This avoid issues arising if vSphere HA & Virtual SAN are partitioned differently and the different partitions try to take ownership of the same object.

To enable both Virtual SAN and vSphere HA on a cluster, Virtual SAN must be enabled first, followed by vSphere HA. You cannot enable Virtual SAN if vSphere HA is already enabled.

To disable Virtual SAN on a cluster with vSphere HA also enabled, one must first of all disable vSphere HA. Only then can Virtual SAN be disabled.

## Changing the vSphere HA network

If both Virtual SAN & vSphere HA are enabled on a cluster, and the administrator wishes to make changes to the Virtual SAN networks, note that these changes are not automatically detected by vSphere HA. Therefore a vSphere HA cluster reconfiguration must be initiated by the administrator so that vSphere HA can learn about these new changes.

# Checking the Virtual SAN network is operational

When the Virtual SAN network has been configured, these commands will check its state. Using the following ESXCLI commands, an administrator can check which VMkernel Adapter is used for Virtual SAN, and what attributes it contains.

First, various ESXCLI and RVC commands verify that the network is indeed fully functional. Then various ESXCLI and RVC commands demonstrate to an administrator how to troubleshoot any network related issues with Virtual SAN.

This will involve verifying that the vmknic used for the Virtual SAN network is uniformly configured correctly across all hosts, checking that multicast is functional and verifying that each host participating in the Virtual SAN cluster can successfully communicate to one another.

## esxcli vsan network list

This is a very useful command as it tells you which VMkernel interface is being used for the Virtual SAN network. In the output below (command and output identical in ESXi 5.5 and 6.0), we see that the Virtual SAN network is using vmk2. This command continues to work even if Virtual SAN has been disabled on the cluster and the hosts no longer participate in Virtual SAN.

There are some additional useful pieces of information such as Agent Group Multicast and Master Group Multicast.

```
~ # esxcli vsan network list
 Interface
    VmkNic Name: vmk2
    IP Protocol: IPv4
    Interface UUID: ccf01954-68f4-3e5f-cb3a-001b21168828
    Agent Group Multicast Address: 224.2.3.4
    Agent Group Multicast Port: 23451
    Master Group Multicast Address: 224.1.2.3
    Master Group Multicast Port: 12345
    Multicast TTL: 5
```

This provides useful information such as which VMkernel interface is being used for Virtual SAN traffic. In this case it is **vmk2**. However, also shown are the multicast addresses. There is the group multicast address and port. Port 23451 is used for the heartbeat, sent every second by the master, and should be visible on every other host in the cluster. Port 12345 is used for the CMMDS updates between the master and backup.

Once we know which VMkernel port Virtual SAN is using for network traffic, we can use some additional commands to check on its status.

### esxcli network ip interface list

Now that we know the VMkernel adapter, we can use this command to check items like which vSwitch or distributed switch that it is attached to, as well as the MTU size, which can be useful if jumbo frames have been configured in the environment. In this case, MTU is at the default of 1500.

```
~ # esxcli network ip interface list
vmk0
   <<truncated>>
vmk1
   <<truncated>>
vmk2
   Name: vmk2
   MAC Address: 00:50:56:61:3d:e1
   Enabled: true
   Portset: DvsPortset-0
   Portgroup: N/A
   Netstack Instance: defaultTcpipStack
   VDS Name: vds01
   VDS UUID: e6 98 23 50 11 e3 dd 59-e9 8c a6 99 bb 0b 2f c8
   VDS Port: 1308
   VDS Connection: 1384090857
   MTU: 1500
   TSO MSS: 65535
   Port ID: 50331661
~ #
```

This output is almost the same on ESXi 5.5 and 6.0, although 6.0 does include some additional "Opaque Network" information. The Maximum Transmission Unit size is shown as 1500, so this VMkernel port is not configured for jumbo frames, which require an MTU of somewhere in the region on 9,000. VMware does not make any recommendation around the use of jumbo frames. Testing to date has revealed no noticeable improvement in Virtual SAN performance by using jumbo frames. However, jumbo frames are supported for use with Virtual SAN should there be a requirement to use them.

### esxcli network ip interface ipv4 get –i vmk2

This is another useful command as it displays information such as IP address and netmask of the VMkernel interface used for Virtual SAN. The command and output are identical between ESXi 5.5 and 6.0. With this information, an administrator can now begin to use other commands available at the command line to check that the Virtual SAN network is working correctly.

```
~ # esxcli network ip interface ipv4 get -i vmk2
 Name  IPv4 Address  IPv4 Netmask   IPv4 Broadcast  Address Type  DHCP DNS
 ----  ------------  -------------  --------------  ------------  --------
 vmk2  172.32.0.4    255.255.255.0  172.32.0.255    STATIC           false
 ~ #
```

### vmkping

The `vmkping` is a simple command that will verify if all of the other ESXi hosts on the network are responding to your ping requests. The command and output are identical on both ESXi 5.5 and 6.0.

```
~ # vmkping -I vmk2 172.32.0.3
 PING 172.32.0.3 (172.32.0.3): 56 data bytes
 64 bytes from 172.32.0.3: icmp_seq=0 ttl=64 time=0.186 ms
 64 bytes from 172.32.0.3: icmp_seq=1 ttl=64 time=2.690 ms
 64 bytes from 172.32.0.3: icmp_seq=2 ttl=64 time=0.139 ms

 --- 172.32.0.3 ping statistics ---
 3 packets transmitted, 3 packets received, 0% packet loss
 round-trip min/avg/max = 0.139/1.005/2.690 ms
```

While it does not verify multicast functionality, it can help with isolating a rogue ESXi host that has network issues. You can also examine the response times to see if there is any abnormal latency on the Virtual SAN network. One thing to note – if jumbo frames are configured, this command will not find any issues if the jumbo frame MTU size is incorrect. This command by default uses an MTU size of 1500. If there is a need to verify if jumbo frames are successfully working end-to-end, use vmkping with a larger packet size option as follows:

```
 ~ # vmkping -I vmk2 172.32.0.3 -s 9000
 PING 172.32.0.3 (172.32.0.3): 9000 data bytes
 9008 bytes from 172.32.0.3: icmp_seq=0 ttl=64 time=0.554 ms
 9008 bytes from 172.32.0.3: icmp_seq=1 ttl=64 time=0.638 ms
 9008 bytes from 172.32.0.3: icmp_seq=2 ttl=64 time=0.533 ms

 --- 172.32.0.3 ping statistics ---
 3 packets transmitted, 3 packets received, 0% packet loss
 round-trip min/avg/max = 0.533/0.575/0.638 ms
 ~ #
```

Consider adding -d to the vmkping command to test if packets can be sent without fragmentation.

### vsan.cluster_info

Of course the difficulty with using many of the above commands is that they need to be run on a per ESXi node basis. There are some commands in RVC to help get a more cluster centric view of the network configuration. One of these is vsan.cluster_info.

```
/localhost/ie-datacenter-01/computers> vsan.cluster_info 0
 Host: cs-ie-h01.ie.local
   Product: VMware ESXi 6.0.0 build-2305723
   VSAN enabled: yes
   Cluster info:
     Cluster role: backup
     Cluster UUID: 529ccbe4-81d2-89bc-7a70-a9c69bd23a19
     Node UUID: 545ca9af-ff4b-fc84-dcee-001f29595f9f
     Member  UUIDs:  ["54188e3a-84fd-9a38-23ba-001b21168828",  "545ca9af-ff4b-fc84-dcee-001f29595f9f",
"5460b129-4084-7550-46e1-0010185def78", "54196e13-7f5f-cba8-5bac-001517a69c72"] (4)
   Node evacuated: no
   Storage info:
     Auto claim: no
     Checksum enforced: no
```

```
    Disk Mappings:
      SSD: HP Serial Attached SCSI Disk (naa.600508b1001c61cedd42b0c3fbf55132) - 186 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c16be6e256767284eaf88) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c64816271482a56a48c3c) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c388c92e817e43fcd5237) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001ccd5d506e7ed19c40a64c) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c79748e8465571b6f4a46) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c2ee9a6446e708105054b) - 136 GB, v1
      MD: HP Serial Attached SCSI Disk (naa.600508b1001c3ea7838c0436dbe6d7a2) - 136 GB, v1
  FaultDomainInfo:
    Not configured
  NetworkInfo:
    Adapter: vmk2 (172.32.0.1)

<<truncated>>
```

The last piece of information from this command is the network information, which tells you the VMkernel adapter and the IP address. This single command displays this information for every host in the Virtual SAN cluster.

### esxcli network ip neighbor list

Switching back to ESXCLI, this next command is a very quick way of checking to see if all Virtual SAN hosts are actually on the same network segment. In this configuration, we have a 4-node cluster, and this command returns the ARP (Address Resolution Protocol) entries of the other 3 nodes, including their IP addresses and their vmknic (Virtual SAN is configured to use vmk2 on all hosts in this cluster). The command and output are identical in ESXi 5.5 and 6.0.

```
~ # esxcli network ip neighbor list -i vmk2
 Neighbor    Mac Address        Vmknic    Expiry  State  Type
 ----------  -----------------  ------  --------  -----  -------
 172.32.0.1  00:50:56:68:63:fa  vmk2    1081 sec         Unknown
 172.32.0.2  00:50:56:6d:cb:3b  vmk2    1081 sec         Unknown
 172.32.0.3  00:50:56:6c:45:63  vmk2    1081 sec         Unknown
 ~ #
```

### esxcli network diag ping

To get even more detail regarding the Virtual SAN network connectivity between the various hosts, ESXCLI provides a powerful network diagnostic command in both ESXi 5.5 & 6.0. This command checks for duplicates on the network, as well as round trip times. Here is an example of one such output, where the VMkernel interface is on vmk2 and the remote Virtual SAN network IP of another host on the network is 172.32.0.3:

```
~ # esxcli network diag ping -I vmk2 -H 172.32.0.3
    Summary:
          Duplicated: 0
          Host Addr: 172.32.0.3
          Packet Lost: 0
          Recieved: 3
          Roundtrip Avg MS: 263
          Roundtrip Max MS: 371
          Roundtrip Min MS: 185
          Transmitted: 3
    Trace:
          Detail:
          Dup: false
```

```
Host: 172.32.0.3
ICMPSeq: 0
Received Bytes: 64
Roundtrip Time MS: 372
TTL: 64

Detail:
Dup: false
Host: 172.32.0.3
ICMPSeq: 1
Received Bytes: 64
Roundtrip Time MS: 186
TTL: 64

Detail:
Dup: false
Host: 172.32.0.3
ICMPSeq: 2
Received Bytes: 64
Roundtrip Time MS: 232
```

## Checking multicast settings

Multicast configurations have been one of the most problematic issues for initial Virtual SAN deployment. One of the simplest ways to verify if multicast is working correctly in your Virtual SAN environment is by using the `tcpdump-uw` command. This command is available from the command line of the ESXi hosts.

### tcpdump-uw –i vmk2 udp port 23451 –v

This tcpdump command will show if the master is correctly sending multicast packets (port and IP info) and if all other hosts in the cluster are receiving them. On the master, this command will show the packets being sent out to the multicast address. On all other hosts, the same exact packets should be seen (from the master to the multicast address). If they are not seen, multicast is not working correctly. Run the `tcpdump-uw` command shown here on any host in the cluster and the heartbeats should be seen coming from the master, which in this case is on IP address 172.32.0.4. The "-v" for verbosity is optional.

```
~# tcpdump-uw -i vmk2 udp port 23451 -v
 tcpdump-uw: listening on vmk2, link-type EN10MB (Ethernet), capture size 96 bytes
 14:17:19.327940 IP truncated-ip - 218 bytes missing! (tos 0x0, ttl 5, id 21137, offset 0,
flags [none], proto UDP (17), length 300)
     172.32.0.3.30254 > 224.2.3.4.23451: UDP, length 272
 14:17:19.791813 IP truncated-ip - 130 bytes missing! (tos 0x0, ttl 5, id 8788, offset 0,
flags [none], proto UDP (17), length 212)
     172.32.0.4.42569 > 224.2.3.4.23451: UDP, length 184
 14:17:20.327813 IP truncated-ip - 218 bytes missing! (tos 0x0, ttl 5, id 28287, offset 0,
flags [none], proto UDP (17), length 300)
     172.32.0.3.30254 > 224.2.3.4.23451: UDP, length 272
 14:17:20.492136 IP truncated-ip - 266 bytes missing! (tos 0x0, ttl 5, id 29439, offset 0,
flags [none], proto UDP (17), length 348)
     172.32.0.3.30254 > 224.2.3.4.23451: UDP, length 320
 14:17:20.493143 IP truncated-ip - 442 bytes missing! (tos 0x0, ttl 5, id 29459, offset 0,
flags [none], proto UDP (17), length 524)
     172.32.0.3.30254 > 224.2.3.4.23451: UDP, length 496
 14:17:20.791810 IP truncated-ip - 130 bytes missing! (tos 0x0, ttl 5, id 26444, offset 0,
flags [none], proto UDP (17), length 212)
     172.32.0.4.42569 > 224.2.3.4.23451: UDP, length 184
```

While this output might seem a little confusing, suffice to say that the output shown here indicates that this host is indeed receiving a heartbeat from the master. This tcpdump-uw command would have to be run on every host to verify that they are all receiving the heartbeat. This will verify that the master is sending the heartbeats, and every other host in the cluster is receiving them, indicating multicast is working.

To get rid of the annoying "IP truncated-ip - XX byes missing" messages, one can simply add a –s0 at the end. –s0 will not truncate the packet.

```
# tcpdump-uw -i vmk2 udp port 23451 -v -s0
tcpdump-uw: listening on vmk2, link-type EN10MB (Ethernet), capture size 65535 bytes
21:14:09.093549 IP (tos 0x0, ttl 5, id 61778, offset 0, flags [none], proto UDP (17),
length 228)
    172.32.0.3.20522 > 224.2.3.4.23451: UDP, length 200
21:14:09.617485 IP (tos 0x0, ttl 5, id 46668, offset 0, flags [none], proto UDP (17),
length 316)
    172.32.0.4.16431 > 224.2.3.4.23451: UDP, length 288
21:14:10.093543 IP (tos 0x0, ttl 5, id 61797, offset 0, flags [none], proto UDP (17),
length 228)
```

If some of the Virtual SAN hosts are not able to pick up the one-second heartbeats from the master, the network admin needs to check the multicast configuration of their switches.

The next multicast checking command is related to IGMP (Internet Group Management Protocol) membership. Hosts (and network devices) use IGMP to establish multicast group membership.

### tcpdump-uw –i vmk2 igmp

Each ESXi node in the Virtual SAN cluster will send out IGMP "membership reports" (aka 'join') every 90-300 seconds. This tcpdump command will show igmp member reports from a host:

```
~ # tcpdump-uw -i vmk2 igmp
tcpdump-uw: verbose output suppressed, use -v or -vv for full protocol decode
listening on vmk2, link-type EN10MB (Ethernet), capture size 96 bytes
14:35:32.846558 IP 0.0.0.0 > all-systems.mcast.net: igmp query v2 [max resp time 1]
14:35:32.926337 IP 172.32.0.3 > 224.1.2.3: igmp v2 report 224.1.2.3
14:35:32.928899 IP 172.32.0.3 > 224.2.3.4: igmp v2 report 224.2.3.4
14:35:45.757769 IP 172.32.1.253 > all-systems.mcast.net: igmp query v2
14:35:47.528132 IP 172.32.0.3 > 224.2.3.4: igmp v2 report 224.2.3.4
14:35:47.738076 IP 172.32.0.3 > 224.1.2.3: igmp v2 report 224.1.2.3
14:36:45.762795 IP 172.32.1.253 > all-systems.mcast.net: igmp query v2
14:36:51.887170 IP 172.32.0.3 > 224.2.3.4: igmp v2 report 224.2.3.4
14:36:56.207235 IP 172.32.0.3 > 224.1.2.3: igmp v2 report 224.1.2.3
14:37:02.846211 IP 0.0.0.0 > all-systems.mcast.net: igmp query v2 [max resp time 1]
```

The output shows "igmp v2 reports" are taking place, indicating that the ESXi host is regularly updating its membership. If a network administrator has any doubts whether or not Virtual SAN ESXi nodes are doing IGMP correctly, running this command on each ESXi host in the cluster and showing this trace can be used to verify that this is indeed the case.

# Changing multicast settings when multiple Virtual SAN clusters are present

When multiple Virtual SAN clusters are configured on the same network, it is recommended that the different Virtual SAN clusters are given distinct multicast ranges. This avoids any hosts from different clusters processing each other's multicast traffic.

To change the multicast address on an ESXi host configured for Virtual SAN, identify the VMkernel interface configured for Virtual SAN by running this command:

### esxcli vsan network list

You should see an output similar to:

```
Interface
   VmkNic Name: vmk2
   IP Protocol: IPv4
   Interface UUID: 6a836354-bf24-f157-dda7-001517a69c72
   Agent Group Multicast Address: 224.2.3.4
   Agent Group Multicast Port: 23451
   Master Group Multicast Address: 224.1.2.3
   Master Group Multicast Port: 12345
   Multicast TTL: 5
```

### esxcli vsan network ipv4 set

Now change the multicast addresses used by Virtual SAN using this command:

```
esxcli vsan network ipv4 set -i <vmkernel interface> -d <multicast agent group
address> -u <multicast master group address>
```

The Master Group Multicast address is used for updating CMMDS between the master and the backup nodes via port 12345. The Agent Group Multicast Address is used for sending heartbeats to all the nodes in the cluster via port 23451. For example, to set the Master Group Multicast address (also called upstream) to 224.2.3.5, and the Agent Group Multicast Address (also called downstream) to 224.2.3.6, use the following ESXCLI command on each ESXi host for this particular Virtual SAN cluster:

```
esxcli vsan network ipv4 set -i vmk2 -d 224.2.3.6 -u 224.2.3.5
```

This command will have to be repeated on every ESXi host that is in the same Virtual SAN Cluster.

### vsan.lldpnetmap

If there are non-Cisco switches with Link Layer Discovery Protocol (LLDP) enabled in the environment, there is an RVC command to display uplink <-> switch <-> switch port information.

This is extremely useful for determining which hosts are attached to which switches when the Virtual SAN Cluster is spanning multiple switches. It may help to isolate a problem to a particular switch when only a subset of the hosts in the cluster is impacted.

```
vsan.lldpnetmap 0
2013-08-15 19:34:18 -0700: This operation will take 30-60 seconds ...
+---------------+---------------------------+
| Host          | LLDP info                 |
+---------------+---------------------------+
| 10.143.188.54 | w2r13-vsan-x650-2: vmnic7 |
|               | w2r13-vsan-x650-1: vmnic5 |
+---------------+---------------------------+
```

This is only available with non-Cisco switches that support LLDP. For Cisco switches, which do not support LLDP but which use their own CDP (Cisco Discovery Protocol), there is no RVC command.

## Network ports and ESXi firewall

These are the network ports used by Virtual SAN.

| Name | Port | Protocol | Traffic Type |
|---|---|---|---|
| CMMDS – Cluster Directory Service | 12345, 23451 | UDP | Multicast |
| RDT – Reliable Datagram Transport | 2233 | TCP | Unicast |
| VSANVP – VASA Provider | 8080 | TCP | Unicast |
| VSAN Observer | 8010 | TCP | Unicast |

This is important to know in case there are firewalls between the vCenter server and the ESXi hosts, as you will need to open port 8080 in that case for the VASA Provider.

It is also important to know in case there are fault domains implemented, and the ESXi hosts are located in different datacenters (although fault domains are really designed for rack awareness in Virtual SAN 6.0). In the case where hosts are located in different datacenters, it is important that port 2233 (for RDT) as well as ports 12345 and 23451 (for CMMDS) is opened between the data centers.

# Checking performance of Virtual SAN network

One of the most important aspects of networking is making sure that there is sufficient bandwidth between your ESXi hosts. This next tool will assist you in testing that your Virtual SAN network is performing optimally.

## iperf for Virtual SAN 5.5

To check the performance of the Virtual SAN network, a commonly used tool is `iperf`. Iperf is a tool to measure maximum TCP bandwidth and latency. This tool is available from http://sourceforge.net/projects/iperf/. However, in version 5.5, it can only be run in a virtual machine and cannot be run directly on the ESXi hosts.

In order to use this tool with Virtual SAN 5.5, the physical NIC or NICs used for the Virtual SAN traffic will have to be used (temporarily) for virtual machine traffic. If using 10Gb NICs, then this is most likely how the network is already configured (sharing the NIC with all traffic types).

If you are using 1Gb NICs, you will need to create a temporary virtual machine network on those NICs dedicated to Virtual SAN traffic, just for the purposes of testing with `iperf`.

***Caution**: If you create a temporary virtual machine network for the purposes of testing network performance, be sure to delete it after testing to avoid any confusion with production virtual machine networks.*

To measure the network performance between two ESXi nodes participating in a Virtual SAN 5.5 cluster, deploy a temporary VM on both ESXi hosts, and ensure that the VM network to which they are attached is consuming the same NICs and physical segments as the Virtual SAN traffic. On one VM, run `iperf` in server mode and on the other VM, run iperf in client mode. VMware Knowledgebase (KB) Article 2001003 has information on how to correctly run `iperf`.

This utility can provide many more options for network tuning which are beyond the scope of this troubleshooting reference manual. This reference manual is looking at the tool solely from the bandwidth and latency measurement, and is a good tool to give you confidence that there is decent network performance between hosts participating in a Virtual SAN 5.5 cluster.

## iperf for Virtual SAN 6.0

For Virtual SAN 6.0, there is a version of iperf available on the ESXi 6.0 hosts. It can be found in `/usr/lib/vmware/vsan/bin/iperf`. Run it with the `--help` option to see how to use the various options. Once again, this tool can be used to check network bandwidth and latency between ESXi hosts participating in a VSAN cluster.

KB article 2001003 referenced previously can assist with setup and testing. Needless to say, this is most useful when run when a Virtual SAN cluster is being commissioned. Running iperf tests on the Virtual SAN network when the cluster is already in production may impact the performance of the virtual machines running on the cluster.

## Checking Virtual SAN network limits

### vsan.check_limits

This command, as seen earlier in the manual, verifies that none of the in-built maximum resource thresholds are being breached.

```
/ie-vcsa-03.ie.local/vsan-dc/computers> ls
0 vsan (cluster): cpu 109 GHz, memory 329 GB
/ie-vcsa-03.ie.local/vsan-dc/computers> vsan.check_limits 0
.
.
.
+--------------------+--------------------+-----------------------------------------+
| Host               | RDT                | Disks                                   |
+--------------------+--------------------+-----------------------------------------+
| cs-ie-h02.ie.local | Assocs: 139/45000  | Components: 116/3000                    |
|                    | Sockets: 102/10000 | naa.600508b1001c19335174d82278dee603: 3%  |
|                    | Clients: 0         | naa.600508b1001c10548f5105fc60246b4a: 8%  |
|                    | Owners: 20         | naa.600508b1001cb2234d6ff4f7b1144f59: 10% |
|                    |                    | naa.600508b1001c577e11dd042e142a583f: 0%  |
|                    |                    | naa.600508b1001c0cc0ba2a3866cf8e28be: 26% |
|                    |                    | naa.600508b1001ca36381622ca880f3aacd: 12% |
|                    |                    | naa.600508b1001c07d525259e83da9541bf: 2%  |
| cs-ie-h03.ie.local | Assocs: 45/45000   | Components: 0/3000                      |
|                    | Sockets: 40/10000  | naa.600508b1001c1a7f310269ccd51a4e83: 0%  |
|                    | Clients: 4         | naa.600508b1001c9b93053e6dc3ea9bf3ef: 0%  |
|                    | Owners: 12         | naa.600508b1001cb11f3292fe743a0fd2e7: 0%  |
|                    |                    | naa.600508b1001c9c8b5f6f0d7a2be44433: 0%  |
|                    |                    | naa.600508b1001ceefc4213ceb9b51c4be4: 0%  |
|                    |                    | naa.600508b1001c2b7a3d39534ac6beb92d: 0%  |
|                    |                    | naa.600508b1001cd259ab7ef213c87eaad7: 0%  |
| cs-ie-h04.ie.local | Assocs: 502/45000  | Components: 97/3000                     |
|                    | Sockets: 187/10000 | naa.600508b1001c4b820b4d80f9f8acfa95: 6%  |
|                    | Clients: 75        | naa.600508b1001c846c000c3d9114ed71b3: 3%  |
|                    | Owners: 84         | naa.600508b1001cadff5d80ba7665b8f09a: 4%  |
|                    |                    | naa.600508b1001c4d41121b41182fa83be4: 3%  |
|                    |                    | naa.600508b1001c40e393b73af79eacdcde: 0%  |
|                    |                    | naa.600508b1001c51f3a696fe0bbbcb5096: 4%  |
|                    |                    | naa.600508b1001c258181f0a088f6e40dab: 4%  |
| cs-ie-h01.ie.local | Assocs: 98/45000   | Components: 97/3000                     |
|                    | Sockets: 101/10000 | naa.600508b1001c388c92e817e43fcd5237: 4%  |
|                    | Clients: 0         | naa.600508b1001c64816271482a56a48c3c: 2%  |
|                    | Owners: 0          | naa.600508b1001c79748e8465571b6f4a46: 2%  |
|                    |                    | naa.600508b1001c61cedd42b0c3fbf55132: 0%  |
|                    |                    | naa.600508b1001c3ea7838c0436dbe6d7a2: 16% |
|                    |                    | naa.600508b1001c2ee9a6446e708105054b: 3%  |
|                    |                    | naa.600508b1001ccd5d506e7ed19c40a64c: 1%  |
|                    |                    | naa.600508b1001c16be6e256767284eaf88: 11% |
+--------------------+--------------------+-----------------------------------------+
```

From a network perspective, it is the RDT associations ("Assocs") and sockets count that most interests us. There are 20,000 associations per host in Virtual SAN 5.5 and 45,000 associations per host in Virtual SAN 6.0. An RDT association is used to track

peer-to-peer network state within Virtual SAN. Virtual SAN is sized such that it should never runs out of RDT associations. Here we can see that there are 45,000 such associations available per host.

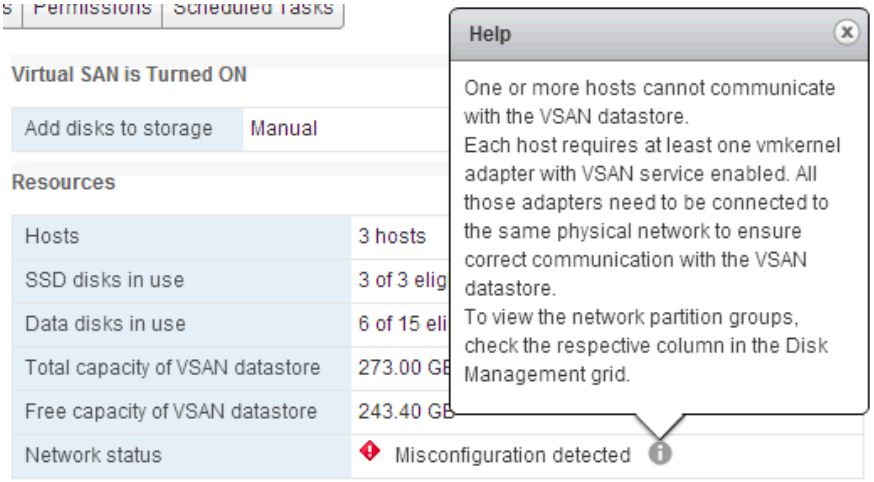
Virtual SAN also limits how many TCP sockets it is allowed to use, and Virtual SAN is sized such that it should never runs out of its allocation of TCP sockets. As can be seen, there is a limit of 10,000 sockets per host.

A Virtual SAN **client** represents object access in the Virtual SAN cluster. Most often than not, the client will typically represent a virtual machine running on a host. Note that the client and the object may not be on the same host. There is no hard defined limit, but this metric is shown to help understand how "clients" balance across hosts.

There is always one and only one Virtual SAN **owner** for a given Virtual SAN object, typically co-located with the Virtual SAN client accessing this object. Virtual SAN owners coordinate all access to the Virtual SAN object and implement functionality like mirroring and striping. There is no hard defined limit, but this metric is once again shown to help understand how "owners" balance across hosts.

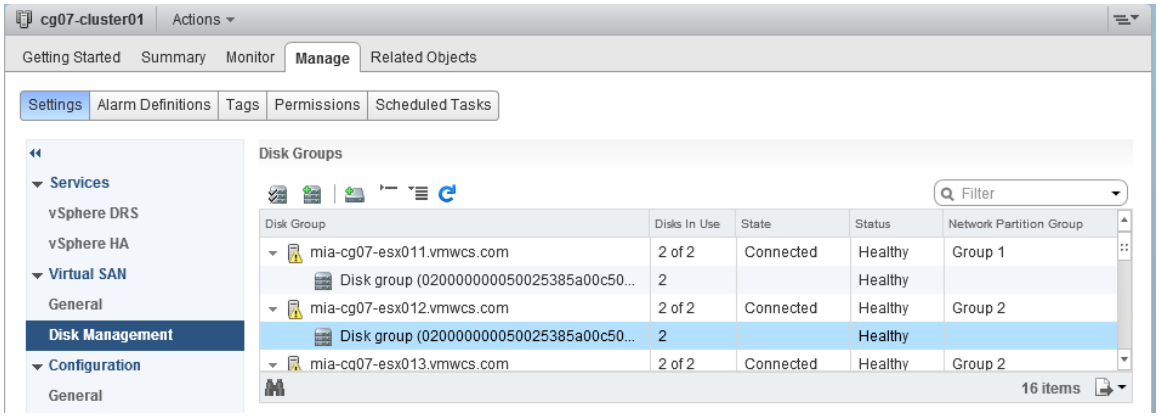
## Network status: Misconfiguration detected

Network configuration issues on Virtual SAN normally show up as a *Misconfiguration detected* on the main Virtual SAN screen in the vSphere Web Client. Clicking on the (i) information icon usually gives some additional detail. This can be caused by a number of issues, the most common of which are covered here.




## Identifying a Partitioned Cluster

Consider a cluster experiencing a network partition issue. The easiest way to identify how exactly the cluster is partitioned is to navigate to the Disk Management view of Virtual SAN shown below (taken from a 5.5 environment), and look at the Network Partition Group column. If all hosts are part of the same group, then there is no partitioning. However, if hosts are in different groups, partitioning has occurred.




Because some hosts appear in Group 1 and others appear in Group 2, there is a network partition. There are other ways of telling whether a cluster has been partitioned. One method is to use the `esxcli vsan cluster get` command on the

various hosts in the cluster. If no host can communicate to any other host, each host will be in its own partition, and will be designated its own "master" node. When multiple hosts are reporting themselves as "master", there is a partition.

If there is only one host in its own partition, and all other hosts are in their own partition, then the scope of the issue would appear to be just that host. If all hosts were in their own unique partition, then it would suggest a network-wide problem.

### esxcli vsan cluster get

Here is an example of a partition with a single node. It is designated "master" but there is no backup node, or are there any additional members in the cluster.

```
~ # esxcli vsan cluster get

Cluster Information
   Enabled: true
   Current Local Time: 2012-12-18T10:35:19Z
   Local Node UUID: 507e7bd5-ad2f-6424-66cb-1cc1de253de4
   Local Node State: MASTER
   Local Node Health State: HEALTHY
   Sub-Cluster Master UUID: 507e7bd5-ad2f-6424-66cb-1cc1de253de4
   Sub-Cluster Backup UUID:
   Sub-Cluster UUID: 52e4fbe6-7fe4-9e44-f9eb-c2fc1da77631
   Sub-Cluster Membership Entry Revision: 7
   Sub-Cluster Member UUIDs: 507e7bd5-ad2f-6424-66cb-1cc1de253de4
   Sub-Cluster Membership UUID: ba45d050-2e84-c490-845f-1cc1de253de4
~ #
```

Compare the above output to one taken from a 4-node cluster that has formed correctly and is not partitioned in any way. Note the Sub-Cluster Member UUIDs in this list.

```
 ~ # esxcli vsan cluster get
 Cluster Information
    Enabled: true
    Current Local Time: 2014-11-28T09:05:29Z
    Local Node UUID: 54188e3a-84fd-9a38-23ba-001b21168828
    Local Node State: MASTER
    Local Node Health State: HEALTHY
    Sub-Cluster Master UUID: 54188e3a-84fd-9a38-23ba-001b21168828
    Sub-Cluster Backup UUID: 545ca9af-ff4b-fc84-dcee-001f29595f9f
    Sub-Cluster UUID: 529ccbe4-81d2-89bc-7a70-a9c69bd23a19
    Sub-Cluster Membership Entry Revision: 3
    Sub-Cluster Member UUIDs: 54188e3a-84fd-9a38-23ba-001b21168828, 545ca9af-ff4b-
fc84-dcee-001f29595f9f, 5460b129-4084-7550-46e1-0010185def78, 54196e13-7f5f-cba8-
5bac-001517a69c72
    Sub-Cluster Membership UUID: 80757454-2e11-d20f-3fb0-001b21168828
```

One additional way of checking for partitions is to use RVC commands. Some examples are show here.

### vsan.cluster_info

This command, seen already, gives an overview of the cluster state. Once again, numerical shortcuts are used – the 0 here represents the cluster name. If this command is returning multiple clusters with a master in each and a subset of hosts in the member UUID list (in this case 2 & 1 for a 3-node cluster), then this is also indicative of a cluster partition.

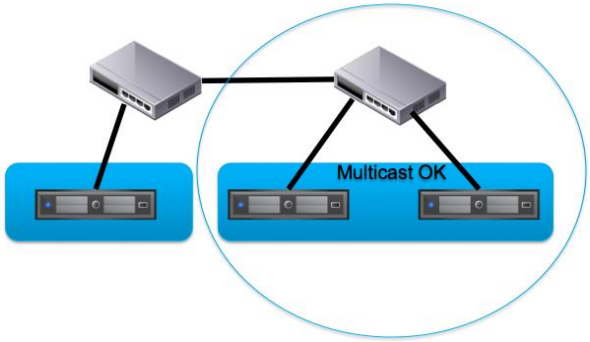
```
/localhost/ie-datacenter-01/computers> vsan.cluster_info 0
 Host: cs-ie-h01.ie.local
   VSAN enabled: yes
   Cluster info:
     Cluster role: master
     Cluster UUID: 52dc5a95-d04b-cbb9-9d90-f486c2f14d1d
     Node UUID: 54184636-badc-c416-1699-001f29595f9f
     Member UUIDs: ["54196e13-7f5f-cba8-5bac-001517a69c72", "54184636-badc-c416-1699-
001f29595f9f"] (2)
   Storage info:
     Auto claim: no
     ...
       <<truncated>>

Host: cs-ie-h02.ie.local
   VSAN enabled: yes
   Cluster info:
     Cluster role: master
     Cluster UUID: 52dc5a95-d04b-cbb9-9d90-f486c2f14d1d
     Node UUID: 54184636-badc-c416-1699-001f29595f9f
     Member UUIDs: ["54196e13-aabb-cba8-6734-001517bdef76"] (1)
   Storage info:
     Auto claim: no
     ...
       <<truncated>>
```

One of the most common network misconfiguration issues with Virtual SAN that leads to partitioning is the multicast misconfiguration. Either multicast has not been enabled correctly, or the IGMP Querier hasn't been setup properly to include all of the hosts in the Virtual SAN cluster.

Let's look at that scenario in more detail.

# Troubleshooting a multicast configuration issue

As mentioned, a common issue is that the Virtual SAN Cluster is configured across multiple physical switches, and while multicast has been enabled on one switch, it has not been enabled across switches. In this case, the cluster forms with two ESXi hosts in one partition, and another ESXi host (connected to the other switch) is unable to join this cluster. Instead it forms its own Virtual SAN cluster in another partition. Remember that the `vsan.lldpnetmap` command seen earlier can assist in determining network configuration, and which hosts are attached to which switch.




## Symptoms of a multicast misconfiguration issue

Other than the fact that the Virtual SAN Cluster status displays a network misconfiguration issue, there are some other telltale signs when trying to form a Virtual SAN Cluster that multicast may be the issue. Assume at this point that the checklist for subnet, VLAN, MTU has been followed and each host in the cluster can `vmkping` every other host in the cluster.

If there is get a network misconfiguration detected when the cluster is formed, the next symptom you will observe is that each ESXi host forms its own Virtual SAN cluster, with itself as the master. It will also have a unique network partition id. This symptom suggests that there is no multicast between any of the hosts.

However if there is a situation where a subset of the ESXi hosts form a cluster, and another subset form another cluster, and each have unique partitions with their own master, backup and perhaps even agent nodes, this may very well be a situation where multicast is enabled in-switch, but not across switches. In situations like this, Virtual SAN shows hosts on the first physical switch forming their own cluster partition, and hosts on the second physical switch forming their own cluster partition too, each with its own "master". If you can verify which switches the hosts in the cluster connect to, and hosts in a cluster are connected to the same switch, then this may well be the issue.

# Troubleshooting a MTU/jumbo frames mismatch

An MTU mismatch can be hard to locate. In Virtual SAN, the cluster will not report any misconfiguration issue if only jumbo frames are misconfigured. That is because the packets used to check that the cluster is working from a network perspective do not utilize jumbo frames to do their checks. In fact, Virtual SAN simply sends a heartbeat, the size of which depends on the number of hosts in the cluster. This means that the heartbeat may be much less than 1500 MTU. Therefore, a situation may arise where the cluster reports that network status is normal, but any attempt to deploy a virtual machine on Virtual SAN will fail.

The check that the MTU size is set correctly, use the following ESXCLI commands:

## esxcli network ip interface list

```
~ # esxcli network ip interface list
 vmk0
    <<truncated>>
 vmk1
    <<truncated>>
 vmk2
    vmk2
    Name: vmk2
    MAC Address: 00:50:56:61:3d:e1
    Enabled: true
    Portset: DvsPortset-1
    Portgroup: N/A
    Netstack Instance: defaultTcpipStack
    VDS Name: ie-vds-04
    VDS UUID: 83 d4 3e 50 ae 27 78 5d-1d 27 2d ff 5a 34 64 85
    VDS Port: 18
    VDS Connection: 1525799189
    Opaque Network ID: N/A
    Opaque Network Type: N/A
    External ID: N/A
    MTU: 1500
    TSO MSS: 65535
    Port ID: 50331661
```

Here is how to check the MTU setting on a standard switch:

## esxcli network vswitch standard list

```
~ # esxcli network vswitch standard list
 vSwitch0
    Name: vSwitch0
    Class: etherswitch
    Num Ports: 4352
    Used Ports: 4
    Configured Ports: 128
    MTU: 1500
    CDP Status: listen
    Beacon Enabled: false
    Beacon Interval: 1
    Beacon Threshold: 3
    Beacon Required By:
    Uplinks: vmnic0
    Portgroups: VM Network, Management Network
```

To find out the MTU value set on a distributed switch, use the command `/bin/net-dvs –l`: and grep from the MTU value. It is in the global properties section on the output. Use the –B option to "grep" to display some leading lines for reference:

```
~ # /bin/net-dvs -l|grep -i MTU -B 15
         global properties:
                 com.vmware.common.version = 0x 3. 0. 0. 0
                         propType = CONFIG
                 com.vmware.common.opaqueDvs = false ,   propType = CONFIG
                 com.vmware.common.alias = vds01 ,       propType = CONFIG
                 com.vmware.common.uplinkPorts:
                         mgmt, vmotion, vm, vsan1, vsan2, vsan3
                         propType = CONFIG
                 com.vmware.etherswitch.ipfix:
                         idle timeout = 15 seconds
                         active timeout = 60 seconds
                         sampling rate = 0
                         collector = 0.0.0.0:0
                         internal flows only = false
                         propType = CONFIG
                 com.vmware.etherswitch.mtu = 1500 ,     propType = CONFIG
 ~ #
```

The final line of this output in the global properties section of the distributed switch shows that the MTU size is indeed set to 1500.

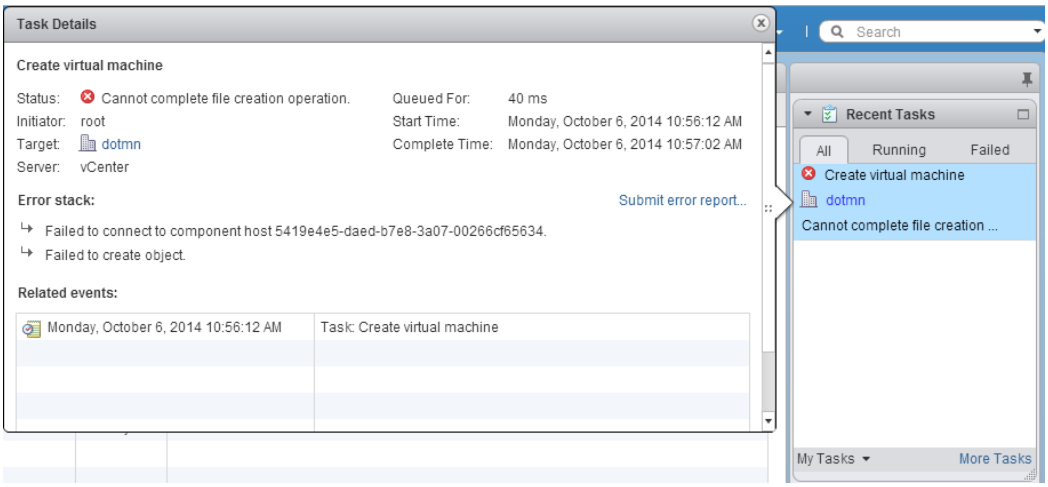
***Caution**: This command appears in a number VMware KB articles to display distributed switch information. However VMware strongly advises against using this command to do anything other than display distributed switch information.*

Finally, the MTU size on the physical switch must be checked. As this can vary from switch to switch, it is outside the scope of this document, but physical switch configurations should be kept in mind when troubleshooting Virtual SAN networking issues.

## Symptoms of MTU misconfiguration: Cannot complete file creation

This is one of the possible symptoms of having mismatched MTU sizes in your environment. The cluster formed successfully and reported that the network status was normal. However on attempting to deploy a virtual machine on the Virtual SAN datastore, an error reported that it "*cannot complete file creation operation*".

This is the error that popped up when the VM was being provisioned on the Virtual SAN datastore:




Note also the '*Failed to connect to component host*" message.

In this case, the customer wanted to use jumbo frames with Virtual SAN in their environment. An MTU of 9000 (jumbo frames) was configured on the physical switch. However in this setup, it seems that an MTU of 9000 on the physical switch (DELL PowerConnect) wasn't large enough to match the MTU of 9000 required on the ESXi configuration due to additional overhead requirements on the switch. The switch actually required an MTU of 9124 (9 * 1024) to allow successful communication using jumbo frames on the Virtual SAN network.

Once this change was made on the physical switch ports used by Virtual SAN, virtual machines could be successfully provisioned on the Virtual SAN datastore.

Again, with jumbo frames, ensure that all nodes in the cluster can be *pinged* with the larger packet size of 9000 as seen earlier.

# Verifying subnets/VLAN settings

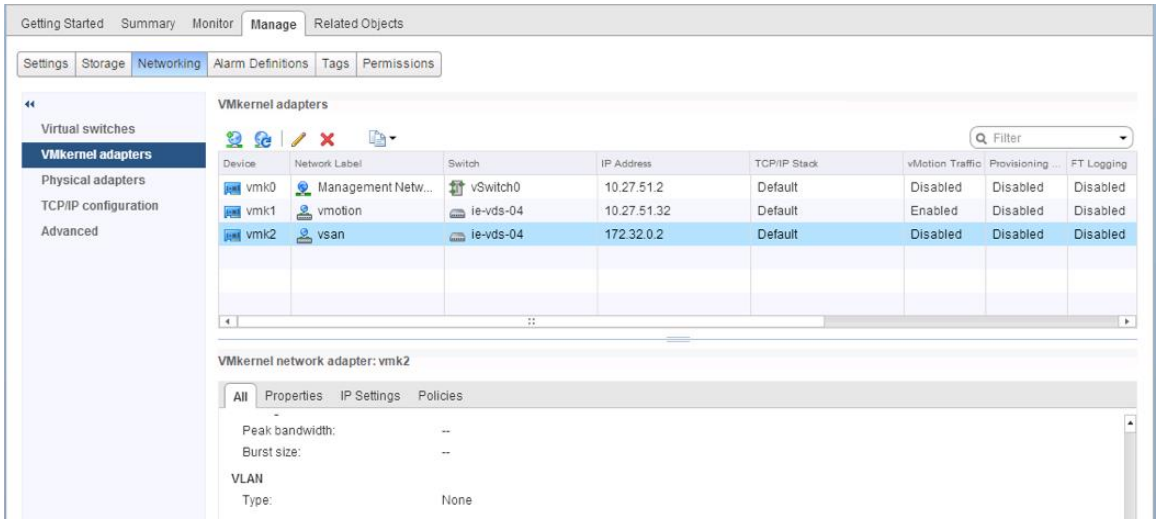
To view correct subnet masks, the following ESXCLI command may be used on each ESXi host in the cluster:

### esxcli network ip interface ipv4 get –i vmk2

```
~ # esxcli network ip interface ipv4 get -i vmk2
 Name  IPv4 Address  IPv4 Netmask   IPv4 Broadcast  Address Type  DHCP DNS
 ----  ------------  -------------  --------------  ------------  --------
 vmk2  172.32.0.4    255.255.255.0  172.32.0.255    STATIC           false
 ~ #
```

One thing missing from the above output is the VLAN id associated with the Virtual SAN network. This is important as the vmknic for the Virtual SAN network on some hosts may be tagged with the VLAN IDs and others may not be tagged. This will again lead to network misconfiguration and cluster partitioning.

To check which VLAN ID, if any, is associated with the Virtual SAN network, the easiest way is to use the vSphere web client, and navigate to the VMkernel adapter on each host, select the Virtual SAN network adapter, and check its properties as shown below. This screenshot is taken from a Virtual SAN 5.5 deployment:




In this case, the Virtual SAN network is not using VLANs that is why the VLAN type in the lower part of the screen is set to "None". If VLANs were being used, this information would be populated. This would then have to be checked on all hosts in the Virtual SAN Cluster.

# Refreshing network configuration

There may be instances where network issues were encountered, and then resolved, but Virtual SAN does not learn about the updated network changes. In this situation, the RVC command `vsan.reapply_vsan_vmknic_config` can help by unbinding Virtual SAN from the VMkernel port, rebinding the Virtual SAN VMkernel port and reapply the Virtual SAN networking configuration.

### vsan.reapply_vsan_vmknic_config

```
> vsan.reapply_vsan_vmknic_config cs-ie-h02.ie.local
Host: cs-ie-h02.ie.local
   Reapplying config of vmk2:
     AgentGroupMulticastAddress: 224.2.3.4
     AgentGroupMulticastPort: 23451
     IPProtocol: IPv4
     InterfaceUUID: 6a836354-bf24-f157-dda7-001517a69c72
     MasterGroupMulticastAddress: 224.1.2.3
     MasterGroupMulticastPort: 12345
     MulticastTTL: 5
   Unbinding VSAN from vmknic vmk2 ...
   Rebinding VSAN to vmknic vmk2 ...
  >
```

# Considerations when using LACP for VSAN network

One option for providing network redundancy for the Virtual SAN network traffic is to use LACP or Link Aggregation. This enables multiple VMkernel ports to be bound together in a NIC team using a "Route based on IP hash" policy. However caution should be exercised when using this feature and administrators should document each uplink used for the Virtual SAN traffic. A single misconfigured uplink on one host can prevent the Virtual SAN cluster from forming.

If a network misconfiguration involving LACP is rectified but the cluster remains partitioned, administrators may run the RVC command `vsan.reapply_vsan_vmknic_config` seen previously. This step may be needed before the network status changes from misconfiguration detected to OK.

# Routing the Virtual SAN traffic over layer 3 networks

In Virtual SAN 5.5, routing Virtual SAN traffic over layer 3 networks is <u>not</u> supported. Therefore a layer 2 leaf-spine architecture must be used if Virtual SAN version 5.5 cluster nodes are connected to different top-of-rack (ToR) switches and where inter-node communication needs to travel through the spine.

Layer 3 (routable) networks are supported in Virtual SAN 6.0. Virtual SAN network documentation planned for the 6.0 timeframe will discuss considerations for layer 3 support.

# Physical network switch configurations and flow control

There have been situations where misbehaving network switches have led to Virtual SAN network outages. Symptoms include hosts unable to communicate, and exceedingly high latency reported for virtual machine I/O in VSAN Observer. However when latency is examined at the VSAN Disks layer, there is no latency, which immediately points to latency being incurred at the network layer.

In one case, it was observed that the physical network switch in question was sending excessive amounts of Pause frames. Pause frames are a flow control mechanism that is designed to stop or reduce the transmission of packets for an interval of time. This behavior negatively impacted the Virtual SAN network performance.

### ethtool

There is a command on the ESXi host called ethtool to check for flow control. Here is an example output:

```
~ # ethtool -a vmnic4
 Pause parameters for vmnic4:
 Autonegotiate:  on
 RX:             off
 TX:             off
```

This output shows that auto-negotiate is set to on, which is recommended for ESXi host NICs, but that there is no flow control enabled on the switch (RX and TX are both off).

In the example outage discussed earlier, there were excessive amounts of pause frames in the RX field, with values in the millions. In this case, one troubleshooting step might be to disable the flow control mechanisms on the switch while further investigation into the root cause takes place.

# Physical network switch feature interoperability

There have been situations where certain features, when enabled on a physical network switch, did not interoperate correctly. In one example, a customer attempted to use multicast with jumbo frames, and because of the inability of the network switch to handle both these features, it impacted the whole of the Virtual SAN network. Note that many other physical switches handled this perfectly; this was an issue with one switch vendor only.

Pay due diligence to whether or not the physical network switch has the ability to support multiple network features enabled concurrently.

## Checklist summary for Virtual SAN networking

1. Shared 10Gb NIC or dedicated 1Gb NIC?
2. Redundant NIC teaming connections set up?
3. VMkernel port for Virtual SAN network traffic configured on each host?
4. Identical VLAN, MTU and subnet across all interfaces?
5. Run vmkping successfully between all hosts?
6. If jumbo frames in use, run vmkping successfully with 9000 packet size between all hosts?
7. Network layout check – single physical switch or multiple physical switches?
8. Multicast enabled on the network?
9. Multiple Virtual SAN clusters on same network? Modify multicast configurations so that unique multicast addresses are being used.
10. If Virtual SAN spans multiple switches, is multicast configured across switches?
11. If IGMP disabled, is IGMP querier configured correctly?
12. Can all other ESXI hosts in the cluster see master's heartbeat over multicast using tcpdump-uw?
13. Verify that ESXi hosts are reporting IGMP membership if IGMP used?
14. Is flow control enabled on the ESXi host NICs?
15. Test Virtual SAN network performance with iperf - meets expectations?
16. Check network limits – within acceptable margins?
17. Ensure that the physical switch can meet VSAN requirements (multicast, flow control, feature interop)

# 11. Troubleshooting Virtual SAN storage

In this section, the Virtual SAN storage is introduced, and the types of issues that can arise from storage issues on your Virtual SAN cluster, as well as how to troubleshoot them.

As highlighted in the introduction, VMware recommends using the Virtual SAN Health Services to do initial triage of storage issues. The Virtual SAN Health Services carry out a range of storage health checks, and direct administrators to an appropriate knowledge base article depending on the results of the health check. The knowledge base article will provide administrators with step-by-step instruction to solve the storage problem at hand.

Please refer to the *Virtual SAN Health Services Guide* for further details on how to get the Health Services components, how to install them and how to use the feature for troubleshooting common Virtual SAN issues.

## Virtual SAN objects and components revisited

At this stage, the concepts of Virtual SAN should now be familiar, such as how hybrid configurations utilize flash devices (such as solid-state disks) for read caching and write buffering, and how data on flash devices is regularly destaged to magnetic disks. All-flash configurations are a little different of course. Each host in a hybrid configuration that is contributing storage to the Virtual SAN datastore needs at least one flash device and one magnetic disk in Virtual SAN 5.5. This grouping of magnetic disks and in Virtual SAN is called a disk group. In Virtual SAN 6.0, flash devices may also be used for the capacity layer (all-flash configuration).

As previously detailed, Virtual SAN also introduces the concept of a virtual machine being made up as a group of objects, rather than a set of files which is what most administrators would be familiar with if they had experience with VMFS and NFS in virtual infrastructures. However, in the context of examining the Virtual SAN datastore and virtual machines deployed on the Virtual SAN datastore, it is worthwhile discussing this critical concept once more.

When a virtual machines is deployed on a Virtual SAN datastore, it may have a number different kinds of storage objects associated with it:

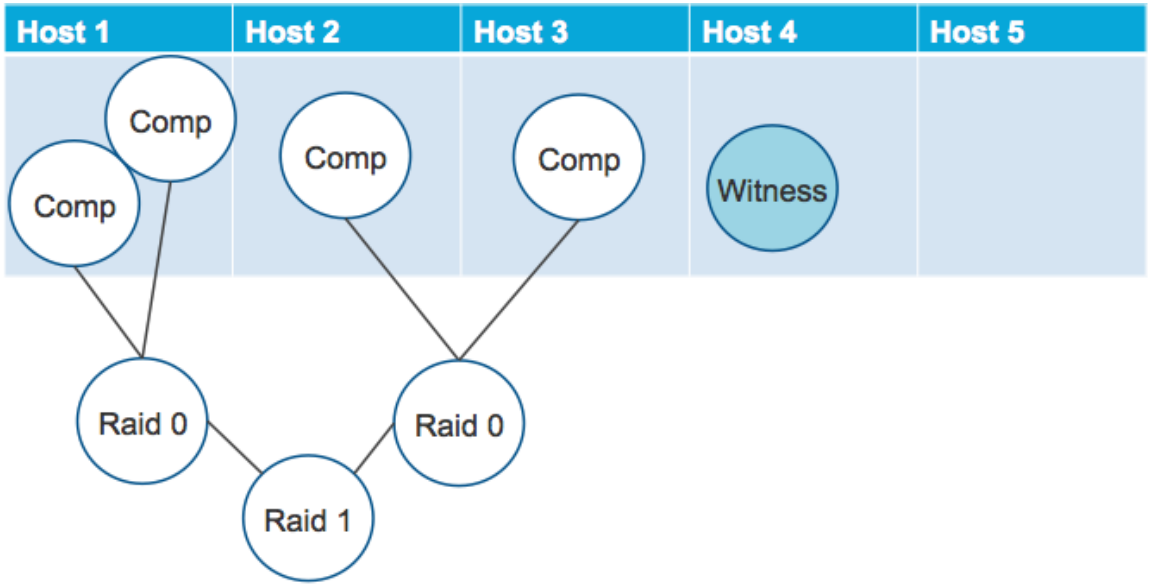
1. The Virtual Machine home or “namespace directory”
2. A swap object (if the virtual machine is powered on)
3. Virtual disks/VMDKs
4. Delta-disks created for snapshots. Each delta-disk is an object.
5. Virtual machine memory object created for snapshots.

## Object layout and RAID trees

Each storage object is deployed on Virtual SAN as a RAID tree and each leaf of the tree is said to be a component. This RAID tree is very much dependent on the VM storage policy associated with the virtual machine. Technically, a storage object may consume more than one component. For instance, if a VMDK is deployed with a stripe width of two, then a RAID-0 stripe would be configured across two disks for this virtual machine disk. The VMDK would be the object, and each of the stripes would be a component of that object.

Similarly, if a VMDK needs to tolerate at least one failure in the cluster (host, disk or network), then a RAID-1 mirror of the VMDK object would be setup, with one replica component on one host and another replica component setup on another host in the Virtual SAN cluster. Finally, if both striping and availability are requested, then the striped components would be mirrored across hosts, giving a RAID 0+1 configuration.

This diagram of a RAID tree should assist in understanding the relationship between objects and components. The VMDK in this example has a VM Storage Policy that has a *NumberOfDiskStripesPerObject*=2 and a *NumberOfFailuresToTolerate*=1. This implies that the VMDK is striped across 2 separate disks, but also that it is mirrored across hosts. Therefore a total of four disks are involved in this configuration.




Understanding the layout of a virtual machine's object on the Virtual SAN datastore is critical to understanding Virtual SAN Storage, checking if the VM's storage is functioning correctly, and if not, how to troubleshoot it.

Note that striping does not guarantee that components get striped across hosts, it only ensure that components are striped across physical disks. However those disks could be in different hosts, depending on how Virtual SAN decides to implement the configuration.

Delta-disks are created when a snapshot is taken of the VM. A delta disk inherits the same policy as the parent disk, e.g. stripe width, replicas, etc.

The swap object is created only when the virtual machine is powered on.

The memory snapshot object is created when a snapshot is taken of the VM, and a memory snapshot is also requested.

There are two limits in relation to components in Virtual SAN:

- Maximum number of components per host limit in Virtual SAN 5.5: 3000
- Maximum number of components per host limit in Virtual SAN 6.0: 9000*
- Maximum number of components per object: 64 (this includes stripe width and replica copies)

* Note that the component count only increases in Virtual SAN 6.0 when the on-disk format is upgraded to v2.

Components per host includes components from powered off virtual machines.

Virtual SAN distributes components across the various hosts in the cluster and will always try to achieve an even distribution of components for balance. However some hosts may have more components than others, which is why VMware recommends, as a best practice, that hosts participating in a Virtual SAN cluster be similarly or identically configured. Components are a significant sizing consideration when designing and deploying a Virtual SAN cluster in 5.5 but not so much really in 6.0, since the maximums are considerably larger.

# Virtual SAN storage requirements

This has already been covered in great detail when we looked at the VMware Compatibility Guide. At this point, the importance of ensuring that the storage I/O controller model chosen for a Virtual SAN is on the VMware Compatibility Guide (VCG) should be clear, and that the driver and firmware versions of the storage I/O controller should be at a supported level.

Another important factor is the class of flash device. This information is also found on the VCG, and the higher the grade or class of flash device that is used, the better the Virtual SAN performance.

However there are some points related to storage requirements that need to be elaborated on.

## Pass-thru mode versus RAID-0 mode

Virtual SAN prefers to be able to consume storage natively, and for that reason we recommend devices be presented to ESXi from the storage controller in pass-through mode. This means that the disks are visible to ESXi without the need of the controller to place a volume (RAID 0) on them.

However not all of the storage controllers on the VCG allow devices to be presented to the host in pass-through mode. In those cases, each individual disk device has to be put into its own RAID 0 volume, and at the point, the ESXi host should be able to discover it. This is not to be confused with the RAID 0 configuration implemented by Virtual SAN stripe width via the policies. The RAID 0 referred to here is a configuration place on the physical device by the controller, either through the BIOS of the card, or via third party tools.

The reason for recommending pass-through over RAID 0 is that the replacement process becomes much easier. New or replacement disks that can be plugged into a host, and detected by ESXi, make disk management a lot simpler.

By examining the VCG output from earlier, when the HP storage controllers on the VCG were listed, we can see that some require RAID 0 configurations, and others support the pass-through mechanism (as listed in the Feature column):

Click on the 'Model' to view more details and to subscribe to RSS feeds.

Bookmark | Print | Export to CSV

**Search Results:** Your search for " Virtual SAN IO Controller " returned **7 results.** Back to Top Turn Off Auto Scroll Display: 10

| Brand Name | Model | Feature | Product Description | Queue Depth | Supported Releases |
|---|---|---|---|---|---|
| HP | Smart Array P822 | Virtual SAN RAID 0 | **Device Type:** SAS **VID:** 103c **SVID:** 103c **DID:** 323b **SSID:** 3353 | 1020 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | Smart Array P420i | Virtual SAN RAID 0 | **Device Type:** SAS **VID:** 103C **SVID:** 103c **DID:** 323B **SSID:** 3354 | 1020 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | Smart Array P420 | Virtual SAN RAID 0 | **Device Type:** SAS **VID:** 103C **SVID:** 103c **DID:** 323B **SSID:** 3351 | 1020 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | Smart Array P220i | Virtual SAN RAID 0 | **Device Type:** SAS **VID:** 103C **SVID:** 103c **DID:** 323B **SSID:** 3355 | 1020 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | H222 | Virtual SAN Pass-Through | **Device Type:** SAS **VID:** 1000 **SVID:** 1590 **DID:** 0087 **SSID:** 0043 | 600 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | H220i | Virtual SAN Pass-Through | **Device Type:** SAS **VID:** 1000 **SVID:** 1590 **DID:** 0087 **SSID:** 0044 | 600 | ESXi 5.5 U2 ESXi 5.5 U1 |
| HP | H220 | Virtual SAN Pass-Through | **Device Type:** SAS **VID:** 1000 **SVID:** 1590 **DID:** 0087 **SSID:** 0041 | 600 | ESXi 5.5 U2 ESXi 5.5 U1 |

## Checking your storage I/O controller queue depth

A good practice is, from time to time, to re-check that everything is behaving correctly from an ESXi perspective. One of the most critical items related to storage is the queue depth, and we have seen the negative impact that small queue depths can have on Virtual SAN performance. If, after upgrading a storage controller's firmware, verify that it has not reduced the queue depth in any significant way (this is an issue we have seen in the past with new controller firmware versions).

## Esxtop for controller queue depth

The easiest way to check the controller queue depth is to use the ESXi command `esxtop`. When esxtop is launched, hit the (d) key to switch to disk view. Next hit the (f) key to add more fields and the field that you need to add is "D" for Queue Stats. When this is enabled, hit return to revert to the disk view screen and now a new column is display, AQLEN that is short for "adapter queue length". Here is a sample output of one such configuration:

```
 2:57:29am up 27 days  6:06, 795 worlds, 9 VMs, 9 vCPUs; CPU load average: 0.12, 0.13, 0.14

 ADAPTR PATH                    NPTH AQLEN    CMDS/s   READS/s WRITES/s MBREAD/s MBWRTN/s DAVG/cmd KAVG/cmd GAVG/cmd QAVG/cmd
fioiom0 -                          1  5000 12801.55   3766.82  9034.73    55.06   147.68      0.81      0.00      0.81      0.00
 vmhba0 -                          1     1     0.00      0.00     0.00     0.00     0.00      0.00      0.00      0.00      0.00
 vmhba1 -                          8  1011     0.95      0.57     0.38     0.00     0.00      0.38      0.01      0.38      0.00
vmhba32 -                          1     1     0.00      0.00     0.00     0.00     0.00      0.00      0.00      0.00      0.00
vmhba33 -                          0  1024     0.00      0.00     0.00     0.00     0.00      0.00      0.00      0.00      0.00
vmhba34 -                          0  1024     0.00      0.00     0.00     0.00     0.00      0.00      0.00      0.00      0.00
vmhba35 -                          0  1024     0.00      0.00     0.00     0.00     0.00      0.00      0.00      0.00      0.00
vmhba36 -                          0  1024     0.00      0.00     0.00     0.00     0.00      0.00      0.00      0.00      0.00
vmhba37 -                          0     1     0.00      0.00     0.00     0.00     0.00      0.00      0.00      0.00      0.00
```

Note this will have to be repeated on all ESXi hosts in the cluster. At the time of writing this manual, VMware is currently only supporting adapters with a queue

depth greater than 256. If the AQLEN is less than this, then verify that this is an adapter supported on the VCG, as well as the version of driver and firmware. Also verify that new upgrades to the driver or firmware of the adapter did not reduce the queue depth. There are other ways of checking the queue depth of a storage I/O controller.

### esxcfg-info –s | grep "==+SCSI Interface" –A 18

Although this command appears a little convoluted, it is a quick way of retrieving a lot of device information, including the adapter queue depth, rather than navigating through `esxtop` as shown in the previous example. Below are some outputs from an HP 410i storage controller shown as adapter **vmhba1**, and a FusionIO PCI-E flash device shown as adapter **fioiom0**.

```
          \==+SCSI Interface :
             |----Name.....................................................vmhba1
             |----
UID.....................................................sas.5001438013ebe0d0
             |----Driver...................................................hpsa
             |----Queue Depth.............................................1011
             |----Is Virtual..............................................false
             \==+PCI Device :
                |----Segment.............................................0x0000
                |----Bus.................................................0x05
                |----Slot................................................0x00
                |----Function............................................0x00
                |----Runtime Owner.......................................vmkernel
                |----Has Configured Owner................................false
                |----Configured Owner....................................
                |----Vendor Id...........................................0x103c
                |----Device Id...........................................0x323a
                |----Sub-Vendor Id.......................................0x103c
                |----Sub-Device Id.......................................0x3245
                |----Vendor Name.........................................Hewlett-
Packard Company

          \==+SCSI Interface :
             |----Name.....................................................fioiom0
             |----
UID.....................................................unknown.fioiom0
             |----Driver...................................................iomemory-
vsl
             |----Queue Depth.............................................5000
             |----Is Virtual..............................................false
             \==+PCI Device :
                |----Segment.............................................0x0000
                |----Bus.................................................0x0b
                |----Slot................................................0x00
                |----Function............................................0x00
                |----Runtime Owner.......................................vmkernel
                |----Has Configured Owner................................false
                |----Configured Owner....................................
                |----Vendor Id...........................................0x1aed
                |----Device Id...........................................0x2001
                |----Sub-Vendor Id.......................................0x1aed
                |----Sub-Device Id.......................................0x2001
                |----Vendor Name.........................................Fusion-io
```

Here is another example, this time from LSI Logic controller that has a queue depth of 600:

```
\==+SCSI Interface :
 |----Name.....................................vmhba1
 |----UID......................................sas.500605b007460710
 |----Driver...................................mpt2sas
 |----Queue Depth..............................600
 |----Is Virtual...............................false
  \==+Data Integrity Information :
  |----Protection Mask.........................0x00000000
  |----Guard Type..............................NO GUARD SUPPORT
  \==+PCI Device :
   |----Segment.................................0x0000
   |----Bus.....................................0x03
   |----Slot....................................0x00
   |----Function................................0x00
   |----Runtime Owner...........................vmkernel
   |----Has Configured Owner....................false
   |----Configured Owner........................
   |----Vendor Id...............................0x1000
   |----Device Id...............................0x0087
   |----Sub-Vendor Id...........................0x1000
   |----Sub-Device Id...........................0x3020
   |----Vendor Name.............................LSI Logic / Symbios Logic
   |----Device Name.............................LSI2308_2
   |----Device Class............................263
   |----Device Class Name.......................Serial Attached SCSI controller
```

# Configuring Virtual SAN storage

Some consideration needs to be given to how the hardware is configured. The importance of ensuring that the storage controller, the flash devices, and the driver and firmware versions are supported has already been covered. There are some additional considerations too.

## Storage I/O controller cache

Different vendors make different recommendations when it comes to the read and write cache on the storage I/O controller. Some vendors ship with both the read and write cache enabled and require that they be disabled. There may even be a recommendation to disable write cache but leave read cache enabled.

VMware recommend disabling the controller cache when it is used with Virtual SAN.

If the read and write cache of a particular storage controller cannot be fully disabled, set the cache to be 100% read, thereby effectively disabling the write cache on the storage I/O controller.

Some of the storage I/O controller vendors supply command line tools to allow for the configuration of their storage controllers, including the cache. Check the appendices of this manual for some examples. However in some cases, this will have to be configured through the BIOS of the storage controller, which may mean scheduling a maintenance slot to bring the host down to do this task.

## A note about HP SSD Smart Path observations

HP describes their SSD Smart Path mechanism as improving the performance of select HP Smart Array Controllers in SSD-based HP servers. HP SSD Smart Path technology allows certain I/O requests to bypass the normal I/O path involving firmware layers, and instead use an accelerated I/O operation called HP SSD Smart. VMware requires that this feature be **disabled** on all HP controllers if they are being used for Virtual SAN.

When HP Smart Path is enabled, you may encounter VSAN datastore accessibility issues, with lots of messages similar to these in the VMkernel log:

```
2014-09-24T13:32:47.184Z cpu12:33081)ScsiDeviceIO: 2324: Cmd(0x412e803f0e00) 0x28, CmdSN
0x1617cd from world 0 to dev "naa.600508b1001c9dc52f1be65fc447d5ca" failed H:0xc D:0x0
P:0x0 Possible sense data: 0x0 0x0 0x0.

2014-09-24T13:32:47.184Z cpu12:33081)NMP: nmp_ThrottleLogForDevice:2321: Cmd 0x28
(0x412e88d04b80, 0) to dev "naa.600508b1001c9dc52f1be65fc447d5ca" on path
"vmhba0:C0:T0:L2" Failed: H:0xc D:0x0 P:0x0 Possible sense data: 0x0 0x0 0x0. Act:NONE
```

A SCSI Read Command (Cmd 0x28) is failing to the device identified by the NAA ID "naa.600508b1001c9dc52f1be65fc447d5ca". It transpires that this issue was caused by having HP Smart Path enabled on the HP 420i controller. Once the feature was disabled on the controllers, the errors stopped.

Unfortunately there are no ESXCLI tools for determining if this feature is enabled or not. The BIOS on the storage controller may have to be accessed, or third party tools from HP used, to determine this. See the appendices for some *hpssacli* commands to query of Smart Path is enabled or not on the storage controller.

Although the manual is calling out HP Smart Path explicitly here, the same advice should be taken for features on controllers from DELL or LSI or any other vendor. Keep the controller configuration as simple as possible. Unless otherwise instructed by VMware support staff, refrain from changing any of the other settings on the storage I/O controllers. Changing from the default settings could adversely impact the performance of the Virtual SAN Cluster.

## A note about the all-flash capacity layer

Virtual SAN 6.0 introduced support for both hybrid and all-flash configurations. With all-flash configurations, flash devices are used not just for the caching layer, but also for the capacity layer. There are some considerations when using flash devices for the capacity layer in all-flash configurations:

1. Flash device must be tagged as capacity devices. This cannot be done via the UI, but must be done at the CLI. The commands used to tag flash devices as capacity are as follows:

   ```
   esxcli vsan storage tag add -d naa.XYZ -t capacityFlash
   esxcli vsan storage tag remove -d naa.XYZ -t capacityFlash
   ```

   See the vSphere Administrators Guide for further details.

2. Flash devices tagged as capacity devices show up as HDDs in the vSphere UI. This is expected in Virtual SAN 6.0.

The **vdq** command in 6.0 has a new **IsCapacityFlash** field that can be used to check if a flash device is tagged as a capacity device. There is an example later on in this manual.

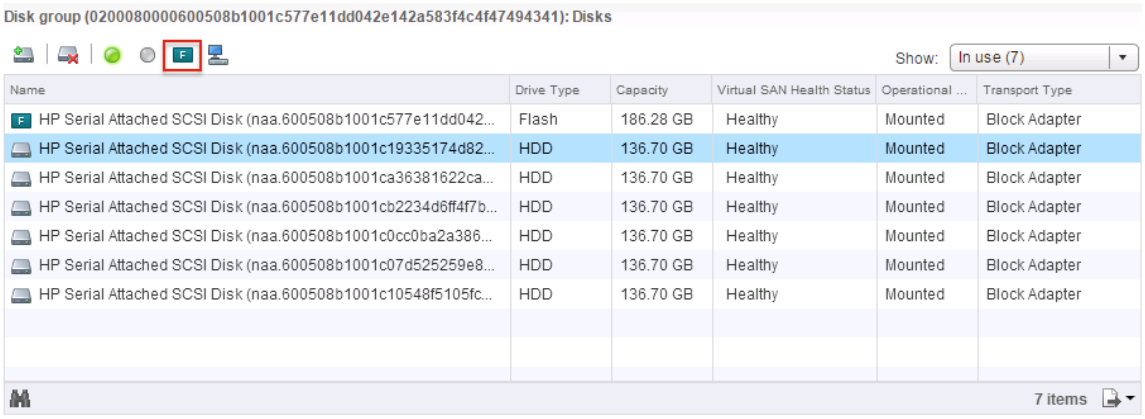
# Identify an SSD which is a RAID-0 volume

Certain devices require a RAID 0 configuration to be identified by the ESXi host, and are not visible to the host when they are passed directly through to the ESXi host.

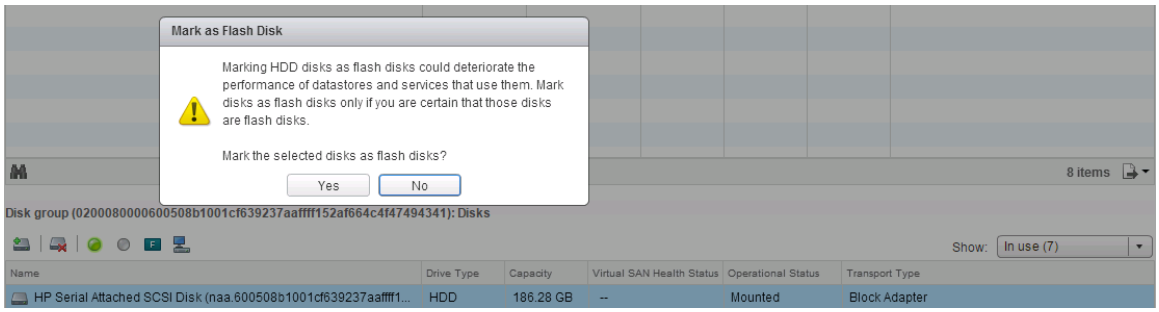
One other nuance of using a RAID 0 configuration to present devices to the ESXi host is that solid state disks do not appear as SSDs to the ESXi host – the *Is_SSD* flag is set to false. Because of this, disk groups cannot be built as these require at least one SSD device. To address this, in Virtual SAN version 5.5 the administrator will need to setup a claim rule in the ESXi Pluggable Storage Architecture (PSA) that sets the option *enable_ssd* on the SSD. This then allows the ESXi host to see the device correctly as an SSD.

VMware Knowledgebase Article 2013188 describes the process in detail for Virtual SAN 5.5. The process is a little convoluted as the administrator must create a "claim rule" that includes the SSD device, unclaim the SSD device from the host, load the new claim rule which marks the device as an SSD, run the rules and finally reclaim the device. It now shows up as an SSD.

In Virtual SAN 6.0, new functionality has been added to the vSphere web client that allows an administrator to mark a device as a flash device via the UI. Navigate to the Disk Management section of Virtual SAN, and if you select one of the disks in the disk group that you wish to designate as a flash device, you will be able to do this by clicking on the "F" icon highlighted below:




When the disk is select and the "F" icon is clicked, the following popup is displayed:




Therefore if using RAID 0 configurations to present disk devices to the ESXi host, and Virtual SAN is not able to detect the SSD as an actual flash device, then it is more than likely the above procedures for marking the flash device as an SSD is needed. The procedure used depends on the version of Virtual SAN.

# Virtual SAN storage limits

This command displays some storage limits, which were briefly touched on.

### vsan.check_limits

The command runs against a cluster object, as shown here.

```
/ie-vcsa-03.ie.local/vsan-dc/computers> ls
 0 vsan (cluster): cpu 109 GHz, memory 329 GB
 /ie-vcsa-03.ie.local/vsan-dc/computers> vsan.check_limits 0
 2014-11-28 10:13:56 +0000: Querying limit stats from all hosts ...
 +--------------------+--------------------+-------------------------------------------+
 | Host               | RDT                | Disks                                     |
 +--------------------+--------------------+-------------------------------------------+
 | cs-ie-h02.ie.local | Assocs: 136/45000  | Components: 116/3000                      |
 |                    | Sockets: 101/10000 | naa.600508b1001c19335174d82278dee603: 3%  |
 |                    | Clients: 0         | naa.600508b1001c10548f5105fc60246b4a: 8%  |
 |                    | Owners: 19         | naa.600508b1001cb2234d6ff4f7b1144f59: 10% |
 |                    |                    | naa.600508b1001c577e11dd042e142a583f: 0%  |
 |                    |                    | naa.600508b1001c0cc0ba2a3866cf8e28be: 26% |
 |                    |                    | naa.600508b1001ca36381622ca880f3aacd: 12% |
 |                    |                    | naa.600508b1001c07d525259e83da9541bf: 2%  |
 | cs-ie-h03.ie.local | Assocs: 45/45000   | Components: 0/3000                        |
 |                    | Sockets: 40/10000  | naa.600508b1001c1a7f310269ccd51a4e83: 0%  |
 |                    | Clients: 4         | naa.600508b1001c9b93053e6dc3ea9bf3ef: 0%  |
 |                    | Owners: 12         | naa.600508b1001cb11f3292fe743a0fd2e7: 0%  |
 |                    |                    | naa.600508b1001c9c8b5f6f0d7a2be44433: 0%  |
 |                    |                    | naa.600508b1001ceefc4213ceb9b51c4be4: 0%  |
 |                    |                    | naa.600508b1001c2b7a3d39534ac6beb92d: 0%  |
 |                    |                    | naa.600508b1001cd259ab7ef213c87eaad7: 0%  |
 | cs-ie-h04.ie.local | Assocs: 505/45000  | Components: 97/3000                       |
 |                    | Sockets: 188/10000 | naa.600508b1001c4b820b4d80f9f8acfa95: 6%  |
 |                    | Clients: 75        | naa.600508b1001c846c000c3d9114ed71b3: 3%  |
 |                    | Owners: 85         | naa.600508b1001cadff5d80ba7665b8f09a: 4%  |
 |                    |                    | naa.600508b1001c4d41121b41182fa83be4: 3%  |
 |                    |                    | naa.600508b1001c40e393b73af79eacdcde: 0%  |
 |                    |                    | naa.600508b1001c51f3a696fe0bbbcb5096: 4%  |
 |                    |                    | naa.600508b1001c258181f0a088f6e40dab: 4%  |
 | cs-ie-h01.ie.local | Assocs: 98/45000   | Components: 97/3000                       |
 |                    | Sockets: 101/10000 | naa.600508b1001c388c92e817e43fcd5237: 4%  |
 |                    | Clients: 0         | naa.600508b1001c64816271482a56a48c3c: 2%  |
 |                    | Owners: 0          | naa.600508b1001c79748e8465571b6f4a46: 2%  |
 |                    |                    | naa.600508b1001c61cedd42b0c3fbf55132: 0%  |
```

```
|                    |                    | naa.600508b1001c3ea7838c0436dbe6d7a2: 16% |
|                    |                    | naa.600508b1001c2ee9a6446e708105054b: 3%  |
|                    |                    | naa.600508b1001ccd5d506e7ed19c40a64c: 1%  |
|                    |                    | naa.600508b1001c16be6e256767284eaf88: 11% |
+--------------------+--------------------+-------------------------------------------+
```

The RDT (Reliable Datagram Transport) limits were discussed in the network troubleshooting section.

For disks, each host has a limit of 3,000 components, and the current component count is also displayed. Also visible, percentage wise, is how much of the disk is being consumed.

***Note**: The component limit per host was 3,000 for Virtual SAN version 5.5. Virtual SAN Version 6.0 increased this limit count to 9,000. However, the on-disk format needs to be updated to v2 for the new component limit to take effect.*

Both of these disk limits in the `vsan.check_limits` are worth noting regularly to ensure that the limits are not being reached.

# Verifying Virtual SAN storage operation – ESX CLI

This section looks at how to examine the storage in the Virtual SAN cluster. First make sure that the device is visible on the ESXi host. The command `esxcli core storage device list` can be used for that:

## esxcli core storage device list

The output of this command differs significantly between ESXi versions 5.5 and 6.0, with a number of extra fields around VVOL and Emulated DIX/DIF.

The first output displays a HP volume that has been configured as a RAID 0 as the model field states that this is a Logical Volume. Also of interest is the **Is SSD**: field set to **false**, and **Is Local**: also set to **false**.

```
# esxcli storage core device list -d naa.600508b1001c626dcb42716218d73319
 naa.600508b1001c626dcb42716218d73319
    Display Name: HP Serial Attached SCSI Disk
(naa.600508b1001c626dcb42716218d73319)
    Has Settable Display Name: true
    Size: 139979
    Device Type: Direct-Access
    Multipath Plugin: NMP
    Devfs Path: /vmfs/devices/disks/naa.600508b1001c626dcb42716218d73319
    Vendor: HP
    Model: LOGICAL VOLUME
    Revision: 3.66
    SCSI Level: 5
    Is Pseudo: false
    Status: degraded
    Is RDM Capable: true
    Is Local: false
    Is Removable: false
    Is SSD: false
    Is VVOL PE: false
    Is Offline: false
    Is Perennially Reserved: false
    Queue Full Sample Size: 0
    Queue Full Threshold: 0
    Thin Provisioning Status: unknown
    Attached Filters:
    VAAI Status: unknown
    Other UIDs: vml.0200060000600508b1001c626dcb42716218d733194c4f47494341
    Is Shared Clusterwide: true
    Is Local SAS Device: false
    Is SAS: true
    Is USB: false
    Is Boot USB Device: false
    Is Boot Device: false
    Device Max Queue Depth: 256
    No of outstanding IOs with competing worlds: 32
    Drive Type: unknown
    RAID Level: unknown
    Number of Physical Drives: unknown
    Protection Enabled: false
    PI Activated: false
    PI Type: 0
    PI Protection Mask: NO PROTECTION
    Supported Guard Types: NO GUARD SUPPORT
    DIX Enabled: false
    DIX Guard Type: NO GUARD SUPPORT
    Emulated DIX/DIF Enabled: false
```

### Is SSD and Is Local

This device is not an SSD but a magnetic disk. This is why "Is SSD" is set to false. However the device is also local, but this commands states "Is Local" as false. The reason for this is because of SAS controllers. Since some SAS controllers allow their disks to be shared between two hosts, Virtual SAN is cautious about reporting the device as local. This can lead to some auto-claim issues, which will be discussed in the common storage issues section shortly.

You might also notice that the **Status** of the magnetic disk is shown as ***degraded*** in the above output. The reason for this is because there is only a single path to the device. If there were multiple paths to the device, then the degraded state goes away. This is nothing to worry about as in most all local disk configurations, there will only be one path to the devices.

This next output is from a FusionIO PCI-E flash device which, when configured, presents a disk device back to an ESXi 5.5 host. There are a number of missing fields when compared to the 6.0 outputs, but a number are still similar. Note that the **Is SSD:** field is set to **true**, as is the **Is Local:** field in this case.

```
eui.a15eb52c6f4043b5002471c7886acfaa
    Display Name: Local FUSIONIO Disk (eui.a15eb52c6f4043b5002471c7886acfaa)
    Has Settable Display Name: true
    Size: 1149177
    Device Type: Direct-Access
    Multipath Plugin: NMP
    Devfs Path: /vmfs/devices/disks/eui.a15eb52c6f4043b5002471c7886acfaa
    Vendor: FUSIONIO
    Model: IODRIVE
    Revision: v1.0
    SCSI Level: 5
    Is Pseudo: false
    Status: on
    Is RDM Capable: false
    Is Local: true
    Is Removable: false
    Is SSD: true
    Is Offline: false
    Is Perennially Reserved: false
    Queue Full Sample Size: 0
    Queue Full Threshold: 0
    Thin Provisioning Status: yes
    Attached Filters:
    VAAI Status: unknown
    Other UIDs: vml.0100000000313231304430393235494f44524956
    Is Local SAS Device: false
    Is USB: false
    Is Boot USB Device: false
    No of outstanding IOs with competing worlds: 32
```

### esxcli vsan storage list

This is a very useful command for looking at the storage claimed by Virtual SAN on a host. The most interesting detail in this output is "`In CMMDS: true`" that implies that the cluster membership and directory services know about this disk, and that the capacity of this disk is contributing to the capacity of the VSAN datastore.

```
~ # esxcli vsan storage list
 naa.600508b1001cb11f3292fe743a0fd2e7
    Device: naa.600508b1001cb11f3292fe743a0fd2e7
    Display Name: naa.600508b1001cb11f3292fe743a0fd2e7
    Is SSD: false
    VSAN UUID: 529ca1d7-5b66-b2aa-d025-2f9a36717030
    VSAN Disk Group UUID: 52349cd9-aacc-3af8-a8d9-b45ea9b3b8bd
    VSAN Disk Group Name: naa.600508b1001c9c8b5f6f0d7a2be44433
    Used by this host: true
    In CMMDS: true
    Checksum: 1624780165303985407
    Checksum OK: true
    Emulated DIX/DIF Enabled: false
<<truncated>>
```

### vdq

This is a very useful ESXi command to look at disk group mappings. Is shows which devices from a particular host are in which disk groups.

In this output, there is a single disk group with one flash device and 7 magnetic disks. This is useful if you have error messages pertaining to a particular SCSI identifier (such as the NAA ID) and you need to figure out which disk group the device resides in.

```
~ # vdq -iH
 Mappings:
    DiskMapping[0]:
            SSD:  eui.a15eb52c6f4043b5002471c7886acfaa
             MD:  naa.600508b1001c4b820b4d80f9f8acfa95
             MD:  naa.600508b1001c4d41121b41182fa83be4
             MD:  naa.600508b1001c846c000c3d9114ed71b3
             MD:  naa.600508b1001c258181f0a088f6e40dab
             MD:  naa.600508b1001cc426a15528d121bbd880
             MD:  naa.600508b1001c51f3a696fe0bbbcb5096
             MD:  naa.600508b1001cadff5d80ba7665b8f09a
```

This next command details whether or not a disk is use by Virtual SAN, and if it is not, what is the reason why. Usually this is because there is an already existing partition table on the disk. This can be very useful for figuring out why Virtual SAN won't claim a particular disk for inclusion in a disk group. Here is a sample output taken from a host in a Virtual SAN 5.5 cluster.

```
~ # vdq -qH
 DiskResults:
    DiskResult[0]:
          Name:  naa.600508b1001c4b820b4d80f9f8acfa95
      VSANUUID:  52c1b588-81f5-cdc7-f4a4-771fbc6f78de
         State:  In-use for VSAN
        Reason:  Non-local disk
```

```
           IsSSD?:  0
           IsPDL?:  0

<<truncated>>

       DiskResult[10]:
             Name:  mpx.vmhba32:C0:T0:L0
         VSANUUID:
            State:  Ineligible for use by VSAN
           Reason:  Has partitions
           IsSSD?:  0
           IsPDL?:  0

       DiskResult[12]:
             Name:  eui.a15eb52c6f4043b5002471c7886acfaa
         VSANUUID:  52c62b40-77a5-7a06-21ec-decd04e21fef
            State:  In-use for VSAN
           Reason:  None
           IsSSD?:  1
           IsPDL?:  0
```

In this truncated output, there is one disk "`Ineligible for use by VSAN`" because it "`Has partitions`". In fact, this is the boot device for this ESXi host, which is why it shows up like this. But this is also a good way to see if there is an existing partition tables on other disks what you feel Virtual SAN should in fact claim for itself.

### vdq - IsCapacityFlash

There are some notable enhancements to this command in Virtual SAN 6.0. It now includes information about checksum support, and if this device is an SSD, is it being used for flash capacity rather than caching.

```
~ # vdq -qH
 DiskResults:

<<truncated>>

       DiskResult[14]:
             Name:  naa.50015178f36363ca
         VSANUUID:  52c97eb4-125e-7b56-d358-7cf3e6c8c6a1
            State:  In-use for VSAN
  ChecksumSupport:  0
           Reason:  None
           IsSSD?:  1
 IsCapacityFlash?:  1
           IsPDL?:  0
```

Here, the second device is an SSD/flash device, but it is being used as a capacity device in an all-flash configuration.

### esxcli storage core device stats get

This next command is very useful to see if there are failures on the disks that are in the Virtual SAN cluster. Block read and written, as well as read and write operations are of most interest and should be at 0, as shown below. Anything other than 0 may indicate a media issue. If errors and warnings appear in the VMkernel logs related to read and or write problems, this is a useful command for checking the actual state of the physical disks (it is same command in 5.5 & 6.0):

```
~ # esxcli storage core device stats get
 naa.600508b1001c4b820b4d80f9f8acfa95
    Device: naa.600508b1001c4b820b4d80f9f8acfa95
    Successful Commands: 37289717
    Blocks Read: 55103809
    Blocks Written: 1108650774
    Read Operations: 2469633
    Write Operations: 34805069
    Reserve Operations: 1
    Reservation Conflicts: 0
    Failed Commands: 14621
    Failed Blocks Read: 0
    Failed Blocks Written: 0
    Failed Read Operations: 0
    Failed Write Operations: 0
    Failed Reserve Operations: 0

 naa.600508b1001c4d41121b41182fa83be4
    Device: naa.600508b1001c4d41121b41182fa83be4
    Successful Commands: 36336511
    Blocks Read: 25348281
    Blocks Written: 1122279506
    Read Operations: 1104872
    Write Operations: 35216624
    Reserve Operations: 1
    Reservation Conflicts: 0
    Failed Commands: 14621
    Failed Blocks Read: 0
    Failed Blocks Written: 0
    Failed Read Operations: 0
    Failed Write Operations: 0
    Failed Reserve Operations: 0
```

# Verifying Virtual SAN storage operation – RVC

In this section, some RVC commands that can help to verify the state of the storage (both physical disks, and objects and components) are examined.

## vsan.check_state

The `vsan.check_state` (seen many times already) will check for inaccessible Virtual SAN object, invalid/inaccessible VMs and it will also check if there are any VMs for which VC/hostd/vmx are out of sync. This is an excellent command for making sure that there are no objects or indeed whole VMs with underlying storage issues on the Virtual SAN cluster. When this command reports inaccessible objects and VMs, some object specific commands will be needed to locate the underlying storage and check its status. This will be done shortly.

## vsan.disks_stats

This is a very useful command to determine the following information about disks:

- Number of components on a disk (for SSDs, this is always 0)
- Total disk capacity
- The percentage of disk that is being consumed
- Health Status of the disk
- The version of the on-disk format

```
/ie-vcsa-03.ie.local/vsan-dc/computers> vsan.disks_stats 0
+-----------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
|                                         |                    |       | Num  | Capacity  |      |          | Status  |
| DisplayName                             | Host               | isSSD | Comp | Total     | Used | Reserved | Health  |
+-----------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c61cedd42b0c3fbf55132 | cs-ie-h01.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v1) |
| naa.600508b1001c2ee9a6446e708105054b | cs-ie-h01.ie.local | MD    | 12   | 136.50 GB | 3 %  | 2 %      | OK (v1) |
| naa.600508b1001c16be6e256767284eaf88 | cs-ie-h01.ie.local | MD    | 14   | 136.50 GB | 12 % | 11 %     | OK (v1) |
| naa.600508b1001ccd5d506e7ed19c40a64c | cs-ie-h01.ie.local | MD    | 14   | 136.50 GB | 2 %  | 1 %      | OK (v1) |
| naa.600508b1001c64816271482a56a48c3c | cs-ie-h01.ie.local | MD    | 15   | 136.50 GB | 3 %  | 2 %      | OK (v1) |
| naa.600508b1001c388c92e817e43fcd5237 | cs-ie-h01.ie.local | MD    | 14   | 136.50 GB | 4 %  | 4 %      | OK (v1) |
| naa.600508b1001c3ea7838c0436dbe6d7a2 | cs-ie-h01.ie.local | MD    | 14   | 136.50 GB | 17 % | 16 %     | OK (v1) |
| naa.600508b1001c79748e8465571b6f4a46 | cs-ie-h01.ie.local | MD    | 14   | 136.50 GB | 3 %  | 2 %      | OK (v1) |
+-----------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c577e11dd042e142a583f | cs-ie-h02.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v1) |
| naa.600508b1001ca36381622ca880f3aacd | cs-ie-h02.ie.local | MD    | 28   | 136.50 GB | 13 % | 12 %     | OK (v1) |
| naa.600508b1001cb2234d6ff4f7b1144f59 | cs-ie-h02.ie.local | MD    | 18   | 136.50 GB | 10 % | 10 %     | OK (v1) |
| naa.600508b1001c19335174d82278dee603 | cs-ie-h02.ie.local | MD    | 11   | 136.50 GB | 3 %  | 3 %      | OK (v1) |
| naa.600508b1001c0cc0ba2a3866cf8e28be | cs-ie-h02.ie.local | MD    | 20   | 136.50 GB | 26 % | 26 %     | OK (v1) |
| naa.600508b1001c10548f5105fc60246b4a | cs-ie-h02.ie.local | MD    | 27   | 136.50 GB | 9 %  | 8 %      | OK (v1) |
| naa.600508b1001c07d525259e83da9541bf | cs-ie-h02.ie.local | MD    | 12   | 136.50 GB | 3 %  | 2 %      | OK (v1) |
+-----------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c9c8b5f6f0d7a2be44433 | cs-ie-h03.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v1) |
| naa.600508b1001ceefc4213ceb9b51c4be4 | cs-ie-h03.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001cd259ab7ef213c87eaad7 | cs-ie-h03.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001c1a7f310269ccd51a4e83 | cs-ie-h03.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001c9b93053e6dc3ea9bf3ef | cs-ie-h03.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001cb11f3292fe743a0fd2e7 | cs-ie-h03.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001c2b7a3d39534ac6beb92d | cs-ie-h03.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
+-----------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c40e393b73af79eacdcde | cs-ie-h04.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v1) |
| naa.600508b1001c51f3a696fe0bbbcb5096 | cs-ie-h04.ie.local | MD    | 17   | 136.50 GB | 5 %  | 4 %      | OK (v1) |
| naa.600508b1001c4b820b4d80f9f8acfa95 | cs-ie-h04.ie.local | MD    | 15   | 136.50 GB | 6 %  | 6 %      | OK (v1) |
| naa.600508b1001cadff5d80ba7665b8f09a | cs-ie-h04.ie.local | MD    | 17   | 136.50 GB | 4 %  | 4 %      | OK (v1) |
| naa.600508b1001c846c000c3d9114ed71b3 | cs-ie-h04.ie.local | MD    | 16   | 136.50 GB | 3 %  | 3 %      | OK (v1) |
| naa.600508b1001c258181f0a088f6e40dab | cs-ie-h04.ie.local | MD    | 16   | 136.50 GB | 4 %  | 4 %      | OK (v1) |
| naa.600508b1001c4d41121b41182fa83be4 | cs-ie-h04.ie.local | MD    | 16   | 136.50 GB | 4 %  | 3 %      | OK (v1) |
+-----------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
/ie-vcsa-03.ie.local/vsan-dc/computers>
```

If `vsan.check_state` reports that there are inaccessible objects or virtual machines, this `vsan.disks_stats` command is a good one to run, as it should be able to tell if all disks are currently OK, or if some are absent or in a failed state.

# Virtual SAN datastore space management

This section will explain why certain operations such as maintenance mode, or provisioning a virtual machine with certain policies, cannot be done due to space constraints on the Virtual SAN datastore.

## Maintenance Mode

When doing remedial operations on a Virtual SAN Cluster, it may be necessary from time to time to place the ESXi host into maintenance mode. Maintenance Mode offers you various options, one of which is a full data migration. There are a few items to consider with this approach:

1. Are there enough hosts in the cluster to meet the number of failures to tolerate policy requirements?
2. Are there enough magnetic disk spindles in the remaining hosts to handle stripe width policy requirement, if set?
3. Is there enough disk space on the remaining hosts to handle the amount of data that must be migrated off of the host being placed into maintenance mode?
4. Is there enough flash capacity on the remaining hosts to handle any flash read cache reservations?

With this in mind, some preliminary examination of the cluster might be necessary to see if it can feasibly handle a maintenance mode operation. The RVC command `vsan.whatif_host_failures`, can be extremely useful for determining whether there is enough capacity to handle a maintenance mode operation (which can be considered along the same lines as a host failure).

```
/localhost/ie-datacenter-01/computers> vsan.whatif_host_failures 0
 Simulating 1 host failures:

 +-----------------+-----------------------------+-----------------------------------+
 | Resource        | Usage right now             | Usage after failure/re-protection |
 +-----------------+-----------------------------+-----------------------------------+
 | HDD capacity    |   5% used (3635.62 GB free) |   7% used (2680.12 GB free)       |
 | Components      |   3% used (11687 available) |   3% used (8687 available)        |
 | RC reservations |   0% used (3142.27 GB free) |   0% used (2356.71 GB free)       |
 +-----------------+-----------------------------+-----------------------------------+
 /localhost/ie-datacenter-01/computers>
```

If it transpires that there is not enough space, consider one of the other maintenance mode options, in the knowledge that the virtual machines may be running in an unprotected state for the duration of the maintenance operation.

### SSD, magnetic disk or host failure

When an SSD or magnetic disk fails, Virtual SAN will immediately begin to rebuild the components from these failed disks on other disks in the cluster. In the event of a magnetic disk failure or a flash capacity device failure, components may get rebuilt on the remaining capacity devices in the same disk group, or on a different disk group (in the knowledge that Virtual SAN always tries to achieve a balance of components across the cluster).

In the case of a flash cache device failure, since this impacts the whole of the disk group, Virtual SAN is going to have to find a lot of space capacity in the cluster to rebuild all the components of that disk group. If there are other disk groups on the same host, it may try to use these, but it may also use disk groups on other hosts in the cluster.

Bottom line – if a disk group fails which has a lot of virtual machines, a lot of spare capacity needs to be found in order to rebuild the components to meet the VM Storage Policy Requirements.

Host failures are slightly different. Because Virtual SAN does not know if a host failure is transient or permanent, it waits for up to 60 minutes, as per the `vsan.clomrepairdelay` advanced setting. When this timer expires, and the host has not come back online, Virtual SAN will start to rebuild the components that were on that host elsewhere in the cluster to meet the VM Storage Policy Requirements. If there were multiple virtual machines across many disk groups and disks, then a considerable amount of space might be required to accommodate this.

Once again the RVC command `vsan.whatif_host_failures` can be really helpful in determining if there is enough capacity.

### Small disk drive capacity considerations

If Virtual SAN is using lots of small drives, and the virtual machines is deployed with large VMDKs, the VMDK may be "split" across two or more disks to accommodate the large VMDK size. This is not necessarily an issue, and Virtual SAN handles this quite well. However it may cause some confusion since the VMDK "split" is shown as a RAID 0 configuration. Note however that this is not a true stripe, as this split will allow different "chunks" of the VMDK to reside on the same physical disk, something that is not allowed when a stripe width requirement is placed in the policy.

### Very large VMDK considerations

With Virtual SAN 6.0, virtual machine disk sizes of 62TB are now supported. However, consideration should be given as to whether an application actually requires this size of a VMDK, and serious consideration should be given to the size of the VMDK chosen for a virtual machine and application.

Once again, although Virtual SAN might have the aggregate space available on the cluster to accommodate this large size, it will depend on where this space is available and whether or not this space can be used to meet the requirements in the VM storage policy. For example, in a 3 node cluster which has 200TB of free space, one could conceivably believe that this should accommodate a VMDK with 62TB that has a *NumberOfFailuresToTolerate*=1 (2 x 62TB = 124TB). However if one host has 100TB free, host two has 50TB free and host three has 50TB free, then this Virtual SAN is not going to be accommodate this request, i.e. it will not move components around to ensure that there are two hosts with greater than 62TB free.

This is another reason for recommending uniformly configured ESXi hosts. This will allow Virtual SAN to balance hosts from a usage perspective, and allow an even distribution of components across all hosts in the cluster. If for some reason, there is an imbalance in the usage, an administrator can try to use the proactive rebalance mechanism introduced in version 6.0 through RVC.

## Changing a VM Storage Policy dynamically

It is important for Virtual SAN administrators to be aware of how Virtual SAN changes a VM Storage Policy dynamically, especially when it comes to sizing. Administrators need to be aware that changing policies dynamically may lead to a *temporary* increase in the amount of space consumed on the Virtual SAN datastore.

When administrators make a change to a VM Storage Policy and then apply this to a virtual machine to make the change, Virtual SAN will attempt to find a new placement for a replica with the new configuration. If Virtual SAN fails to find a new placement, the reconfiguration will fail. In some cases existing parts of the current configuration can be reused and the configuration just needs to be updated or extended. For example, if an object currently uses *NumberOfFailuresToTolerate*=1, and the user asks for *NumberOfFailuresToTolerate* =2, if there are additional hosts to use, Virtual SAN can simply add another mirror (and witnesses).

In other cases, such as changing the *StripeWidth* from 1 to 2, Virtual SAN cannot reuse the existing replicas and will create a brand new replica or replicas without impacting the existing objects. This means that applying this policy change will increase the amount of space that is being consumed by the virtual machine, albeit temporarily, and the amount of space consumed will be determined by the

requirements placed in the policy. When the reconfiguration is completed, Virtual SAN then discards the old replicas.

## Provisioning with a policy that cannot be implemented

Another consideration related to VM Storage Policy requirements is that even though there may appear to be enough space in the Virtual SAN cluster, a virtual machine will not provision with certain policy settings.

While it might be obvious that a certain number of spindles is needed to satisfy a stripe width requirement, and that the number of spindles required increases as a *NumberOfFailuresToTolerate* requirement is added to the policy, Virtual SAN does not consolidate current configurations to accommodate newly deployed virtual machines.

For example, Virtual SAN will not move components around hosts or disks groups to allow for the provisioning of a new replica, even though this might free enough space to allow the new virtual machine to be provisioned. Therefore, even though there may be enough free space overall in the cluster, most of the free space may be on one node, and there may not be enough space on the remaining nodes to satisfy the replica copies for *NumberOfFailuresToTolerate.*

A well balanced cluster, with unform storage and flash configurations, will mitiage this issue significantly.

### What happens when a threshold is reached?

There is only one capacity threshold to consider with Virtual SAN and that is the physical disk drive threshold. This is set at 80%, and if any single disk drive reaches this threshold (i.e. more than 80% of the drive is consumed), Virtual SAN attempts an automatic rebalance of components across the cluster to bring the amount of space consumed below this threshold.

In version 5.5, components will be moved to disks whose capacity is below 80% used. VSAN will not move components to a disk that is already 80% full. If all of the disks go above the 80% threshold, rebalance will stop since there are no target disks that can be moved to.

In version 6.0, Virtual SAN will continue to do rebalancing to achieve better resource utilization, even if all the disks are above the 80% capacity mark. Also in version 6.0, there is a new proactive rebalance utility that can be run from RVC.

There are no thresholds set on the Virtual SAN datastore. Note however that the capacity usage shown against the Virtual SAN datastore is the average fullness of all disks in the Virtual SAN cluster.

# Component distribution on Virtual SAN

While Virtual SAN always tried to deploy components in a balanced way across the cluster, something missing in the 5.5 releases of Virtual SAN was the ability to rebalance components. This was critical in the case of full evacuation maintenance mode, introducing new hosts/disks or indeed host failures. In these scenarios, all of the components would reside in the original hosts in the case of a newly introduced host or on the remaining hosts in the event of a maintenance mode operation or host failure. The cluster would be left in an imbalanced state from a components perspective. In Virtual SAN 6.0, a new balancing mechanism has been introduced via RVC. Commands for checking component distribution via RVC as shown here:

## Checking disk usage distribution with RVC – vsan.disks_stats

```
/ie-vcsa-06.ie.local/IE-VSAN-DC/computers> vsan.disks_stats 0
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
|                                      |                    |       | Num  | Capacity  |      |          | Status  |
| DisplayName                          | Host               | isSSD | Comp | Total     | Used | Reserved | Health  |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c61cedd42b0c3fbf55132 | cs-ie-h01.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c3ea7838c0436dbe6d7a2 | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c388c92e817e43fcd5237 | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001ccd5d506e7ed19c40a64c | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c2ee9a6446e708105054b | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c16be6e256767284eaf88 | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c79748e8465571b6f4a46 | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c64816271482a56a48c3c | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c64b76c8ceb56e816a89d | cs-ie-h02.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c0cc0ba2a3866cf8e28be | cs-ie-h02.ie.local | MD    | 17   | 136.69 GB | 10 % | 10 %     | OK (v2) |
| naa.600508b1001cb2234d6ff4f7b1144f59 | cs-ie-h02.ie.local | MD    | 15   | 136.69 GB | 4 %  | 4 %      | OK (v2) |
| naa.600508b1001c19335174d82278dee603 | cs-ie-h02.ie.local | MD    | 17   | 136.69 GB | 4 %  | 4 %      | OK (v2) |
| naa.600508b1001c10548f5105fc60246b4a | cs-ie-h02.ie.local | MD    | 17   | 136.69 GB | 3 %  | 3 %      | OK (v2) |
| naa.600508b1001ca36381622ca880f3aacd | cs-ie-h02.ie.local | MD    | 14   | 136.69 GB | 3 %  | 3 %      | OK (v2) |
| naa.600508b1001c07d525259e83da9541bf | cs-ie-h02.ie.local | MD    | 17   | 136.69 GB | 2 %  | 2 %      | OK (v2) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c9c8b5f6f0d7a2be44433 | cs-ie-h03.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c2b7a3d39534ac6beb92d | cs-ie-h03.ie.local | MD    | 19   | 136.69 GB | 3 %  | 3 %      | OK (v2) |
| naa.600508b1001ceefc4213ceb9b51c4be4 | cs-ie-h03.ie.local | MD    | 15   | 136.69 GB | 3 %  | 3 %      | OK (v2) |
| naa.600508b1001cd259ab7ef213c87eaad7 | cs-ie-h03.ie.local | MD    | 15   | 136.69 GB | 4 %  | 4 %      | OK (v2) |
| naa.600508b1001cb11f3292fe743a0fd2e7 | cs-ie-h03.ie.local | MD    | 13   | 136.69 GB | 3 %  | 3 %      | OK (v2) |
| naa.600508b1001c9b93053e6dc3ea9bf3ef | cs-ie-h03.ie.local | MD    | 17   | 136.69 GB | 8 %  | 8 %      | OK (v2) |
| naa.600508b1001c1a7f310269ccd51a4e83 | cs-ie-h03.ie.local | MD    | 18   | 136.69 GB | 18 % | 18 %     | OK (v2) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001cf639237aaffff152af66 | cs-ie-h04.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001cadff5d80ba7665b8f09a | cs-ie-h04.ie.local | MD    | 18   | 136.69 GB | 16 % | 11 %     | OK (v2) |
| naa.600508b1001c51f3a696fe0bbbcb5096 | cs-ie-h04.ie.local | MD    | 20   | 136.69 GB | 8 %  | 2 %      | OK (v2) |
| naa.600508b1001c4b820b4d80f9f8acfa95 | cs-ie-h04.ie.local | MD    | 14   | 136.69 GB | 21 % | 1 %      | OK (v2) |
| naa.600508b1001c6a664d5d576299cec941 | cs-ie-h04.ie.local | MD    | 13   | 136.69 GB | 5 %  | 1 %      | OK (v2) |
| naa.600508b1001c258181f0a088f6e40dab | cs-ie-h04.ie.local | MD    | 16   | 136.69 GB | 7 %  | 2 %      | OK (v2) |
| naa.600508b1001c846c000c3d9114ed71b3 | cs-ie-h04.ie.local | MD    | 16   | 136.69 GB | 6 %  | 1 %      | OK (v2) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
```

## Checking component distribution with RVC – vsan.disks_limits

```
/ie-vcsa-06.ie.local/IE-VSAN-DC/computers> vsan.check_limits 0
2014-12-11 14:11:32 +0000: Querying limit stats from all hosts ...
2014-12-11 14:11:36 +0000: Fetching VSAN disk info from cs-ie-h01.ie.local (may take a moment) ...
2014-12-11 14:11:36 +0000: Fetching VSAN disk info from cs-ie-h02.ie.local (may take a moment) ...
2014-12-11 14:11:36 +0000: Fetching VSAN disk info from cs-ie-h04.ie.local (may take a moment) ...
2014-12-11 14:11:36 +0000: Fetching VSAN disk info from cs-ie-h03.ie.local (may take a moment) ...
2014-12-11 14:11:40 +0000: Done fetching VSAN disk infos
+--------------------+-------------------+-----------------------------------------+
| Host               | RDT               | Disks                                   |
+--------------------+-------------------+-----------------------------------------+
| cs-ie-h01.ie.local | Assocs: 1/45000   | Components: 0/9000                      |
|                    | Sockets: 4/10000  | naa.600508b1001c16be6e256767284eaf88: 0%  |
|                    | Clients: 0        | naa.600508b1001c3ea7838c0436dbe6d7a2: 0%  |
|                    | Owners: 0         | naa.600508b1001c388c92e817e43fcd5237: 0%  |
|                    |                   | naa.600508b1001ccd5d506e7ed19c40a64c: 0%  |
|                    |                   | naa.600508b1001c61cedd42b0c3fbf55132: 0%  |
|                    |                   | naa.600508b1001c64816271482a56a48c3c: 0%  |
|                    |                   | naa.600508b1001c79748e8465571b6f4a46: 0%  |
|                    |                   | naa.600508b1001c2ee9a6446e708105054b: 0%  |
```

```
| cs-ie-h02.ie.local | Assocs: 156/45000 | Components: 97/9000                      |
|                    | Sockets: 26/10000 | naa.600508b1001c0cc0ba2a3866cf8e28be: 9%  |
|                    | Clients: 11       | naa.600508b1001c19335174d82278dee603: 3%  |
|                    | Owners: 12        | naa.600508b1001c10548f5105fc60246b4a: 3%  |
|                    |                   | naa.600508b1001c07d525259e83da9541bf: 1%  |
|                    |                   | naa.600508b1001c64b76c8ceb56e816a89d: 0%  |
|                    |                   | naa.600508b1001ca36381622ca880f3aacd: 2%  |
|                    |                   | naa.600508b1001cb2234d6ff4f7b1144f59: 3%  |
| cs-ie-h03.ie.local | Assocs: 151/45000 | Components: 97/9000                      |
|                    | Sockets: 26/10000 | naa.600508b1001c9c8b5f6f0d7a2be44433: 0%  |
|                    | Clients: 0        | naa.600508b1001cd259ab7ef213c87eaad7: 4%  |
|                    | Owners: 17        | naa.600508b1001c1a7f310269ccd51a4e83: 17% |
|                    |                   | naa.600508b1001c9b93053e6dc3ea9bf3ef: 7%  |
|                    |                   | naa.600508b1001ceefc4213ceb9b51c4be4: 2%  |
|                    |                   | naa.600508b1001c2b7a3d39534ac6beb92d: 2%  |
|                    |                   | naa.600508b1001cb11f3292fe743a0fd2e7: 3%  |
| cs-ie-h04.ie.local | Assocs: 438/45000 | Components: 97/9000                      |
|                    | Sockets: 42/10000 | naa.600508b1001c846c000c3d9114ed71b3: 6%  |
|                    | Clients: 68       | naa.600508b1001c258181f0a088f6e40dab: 6%  |
|                    | Owners: 68        | naa.600508b1001cf639237aaffff152af66: 0%  |
|                    |                   | naa.600508b1001cadff5d80ba7665b8f09a: 15% |
|                    |                   | naa.600508b1001c51f3a696fe0bbbcb5096: 8%  |
|                    |                   | naa.600508b1001c4b820b4d80f9f8acfa95: 20% |
|                    |                   | naa.600508b1001c6a664d5d576299cec941: 4%  |
+--------------------+-------------------+-------------------------------------------+
```

## Proactively balancing component distribution with RVC

There is a new manual rebalance command in RVC as part of 6.0. This will look at the distribution of components around the cluster, and will proactively begin to balance the distribution of components around the cluster.

The option is not running by default. An administrator will have to initiate the proactive balancing of components with the `--start` option.

Other options allow an administrator to specify how long the rebalance operation should run for, and how much data should be moved per hour for each node.

### vsan.proactive_rebalance

```
usage: proactive_rebalance [opts] cluster

Configure proactive rebalance for Virtual SAN
   cluster: Path to ClusterComputeResource
                     --start, -s:   Start proactive rebalance
              --time-span, -t <i>:   Determine how long this proactive rebalance
                                     lasts in seconds, only be valid when option
                                     'start' is specified
   --variance-threshold, -v <f>:   Configure the threshold, that only if disk's
                                     used_capacity/disk_capacity exceeds this
                                     threshold, disk is qualified for proactive
                                     rebalance, only be valid when option 'start'
                                     is specified
        --time-threshold, -i <i>:   Threashold in seconds, that only when
                                     variance threshold continuously exceeds this
                                     threshold, corresponding disk will be
                                     involved to proactive rebalance, only be
                                     valid when option 'start' is specified
        --rate-threshold, -r <i>:   Determine how many data in MB could be moved
                                     per hour for each node, only be valid when
                                     option 'start' is specified
                      --stop, -o:   Stop proactive rebalance
                      --help, -h:   Show this message
```

Some clarity might be needed for the start parameter "--variance-threshold". The description states "Configure the threshold, that only if disk's used capacity divided by disk capacity exceeds this threshold... "

In fact, the trigger condition is only when the following calculation is greater than the <variance_threshold>:

```
(<used_capacity_of_this_disk> / <this_disk_capacity>) -
(<used_capacity_of_least_full_disk_in_cluster> / <least_full_disk_capacity>)
```

In other words, a disk is qualified for proactive rebalancing only if its fullness (used_capacity/disk_capacity) exceeds the fullness of the "least-full" disk in the vsan cluster by the threshold. The rebalancing process also needs to wait until the <time_threshold> is met under this situation, and then start to try rebalancing.

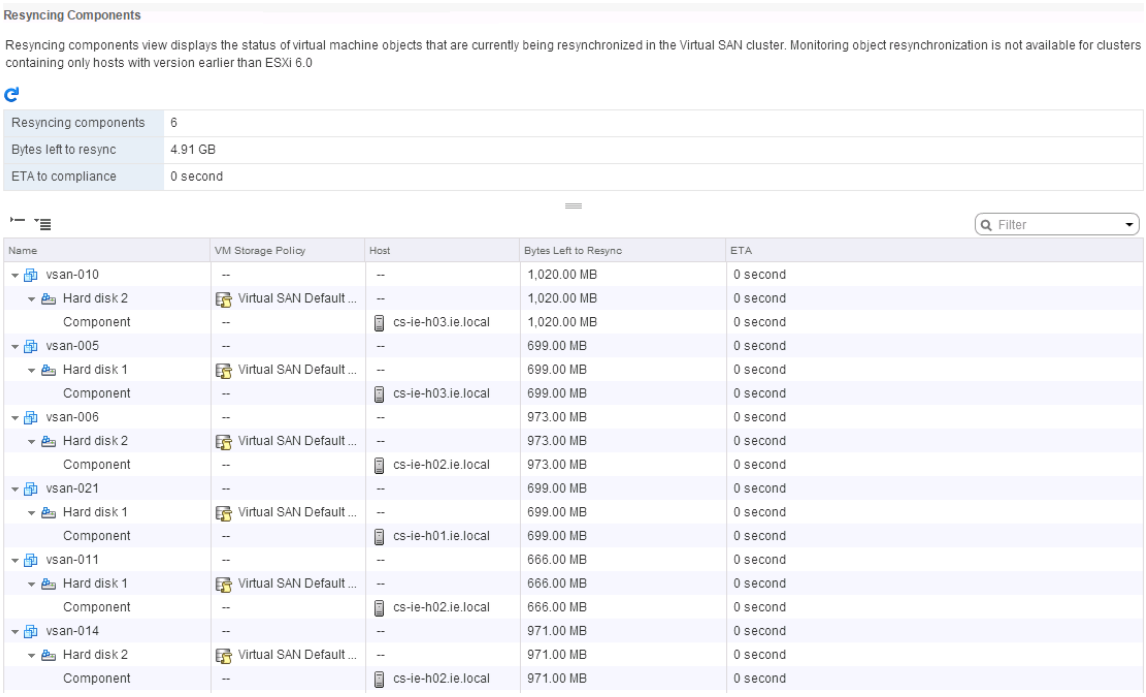
Here is an example of how to start and stop proactive component balancing via RVC:

```
/ie-vcsa-06.ie.local/IE-VSAN-DC/computers> vsan.proactive_rebalance_info 0
 2014-12-11 14:14:27 +0000: Retrieving proactive rebalance information from host cs-ie-h02.ie.local ...
 2014-12-11 14:14:27 +0000: Retrieving proactive rebalance information from host cs-ie-h04.ie.local ...
 2014-12-11 14:14:27 +0000: Retrieving proactive rebalance information from host cs-ie-h01.ie.local ...
 2014-12-11 14:14:27 +0000: Retrieving proactive rebalance information from host cs-ie-h03.ie.local ...

 Proactive rebalance is not running!
 Max usage difference triggering rebalancing: 30.00%
 Average disk usage: 5.00%
 Maximum disk usage: 26.00% (21.00% above mean)
 Imbalance index: 5.00%
 No disk detected to be rebalanced

/ie-vcsa-06.ie.local/IE-VSAN-DC/computers> vsan.proactive_rebalance -s 0
 2014-12-11 14:15:05 +0000: Processing Virtual SAN proactive rebalance on host cs-ie-h02.ie.local ...
 2014-12-11 14:15:05 +0000: Processing Virtual SAN proactive rebalance on host cs-ie-h04.ie.local ...
 2014-12-11 14:15:05 +0000: Processing Virtual SAN proactive rebalance on host cs-ie-h01.ie.local ...
 2014-12-11 14:15:05 +0000: Processing Virtual SAN proactive rebalance on host cs-ie-h03.ie.local ...

 Proactive rebalance has been started!

/ie-vcsa-06.ie.local/IE-VSAN-DC/computers> vsan.proactive_rebalance_info 0
 2014-12-11 14:15:11 +0000: Retrieving proactive rebalance information from host cs-ie-h02.ie.local ...
 2014-12-11 14:15:11 +0000: Retrieving proactive rebalance information from host cs-ie-h01.ie.local ...
 2014-12-11 14:15:11 +0000: Retrieving proactive rebalance information from host cs-ie-h04.ie.local ...
 2014-12-11 14:15:11 +0000: Retrieving proactive rebalance information from host cs-ie-h03.ie.local ...

 Proactive rebalance start: 2014-12-11 14:13:10 UTC
 Proactive rebalance stop: 2014-12-12 14:16:17 UTC
 Max usage difference triggering rebalancing: 30.00%
 Average disk usage: 5.00%
 Maximum disk usage: 26.00% (21.00% above mean)
 Imbalance index: 5.00%
 No disk detected to be rebalanced

/ie-vcsa-06.ie.local/IE-VSAN-DC/computers> vsan.proactive_rebalance -o 0
 2014-12-11 14:15:45 +0000: Processing Virtual SAN proactive rebalance on host cs-ie-h01.ie.local ...
 2014-12-11 14:15:45 +0000: Processing Virtual SAN proactive rebalance on host cs-ie-h02.ie.local ...
 2014-12-11 14:15:45 +0000: Processing Virtual SAN proactive rebalance on host cs-ie-h04.ie.local ...
 2014-12-11 14:15:45 +0000: Processing Virtual SAN proactive rebalance on host cs-ie-h03.ie.local ...

 Proactive rebalance has been stopped!

 /ie-vcsa-06.ie.local/IE-VSAN-DC/computers>
```

# Virtual SAN failure remediation – rebuilding components

In the event of a failure that impacts storage, Virtual SAN will (if there is enough available disk space elsewhere in the cluster) rebuild the components that reside on the failed storage.

In this first example, the object is still accessible. If there are enough resources in the cluster, Virtual SAN will rebuild components automatically. This status can be observed in a number of different ways. First, it can be seen in the vSphere Web Client in version 6.0. This view is not available in 5.5. Select the Virtual SAN cluster object in the inventory, then Monitor, Virtual SAN and then Resyncing Components. It will display something similar to the following:



The same information can be retrieved from the `vsan.resync_dashboard` RVC command. This is available in both 5.5 and 6.0 versions:

### vsan.resync_dashboard

```
> vsan.resync_dashboard 0
 2014-11-28 15:26:42 +0000: Querying all VMs on VSAN ...
 2014-11-28 15:26:42 +0000: Querying all objects in the system from cs-ie-h02.ie.local ...
 2014-11-28 15:26:43 +0000: Got all the info, computing table ...
 +-----------------------------------------------------------------------------+-----------------+---------------+
 | VM/Object                                                                   | Syncing objects | Bytes to sync |
 +-----------------------------------------------------------------------------+-----------------+---------------+
 | vsan-022                                                                    | 1               |               |
 |    [vsanDatastore] 89ae3e54-6495-85f6-9b6b-001b21168828/vsan-022_1.vmdk     |                 | 1.00 GB       |
 | vsan-013                                                                    | 1               |               |
 |    [vsanDatastore] f0713e54-2071-7be0-d570-001b21168828/vsan-013.vmdk       |                 | 0.68 GB       |
 | vsan-009                                                                    | 1               |               |
 |    [vsanDatastore] 1d713e54-9c80-4c2c-03de-001b21168828/vsan-009_1.vmdk     |                 | 0.97 GB       |
 | vsan-008                                                                    | 1               |               |
 |    [vsanDatastore] 1d713e54-b8d5-dd1d-b144-001b21168828/vsan-008.vmdk       |                 | 0.68 GB       |
 | vsan-007                                                                    | 2               |               |
 |    [vsanDatastore] d3703e54-24be-0d86-06e0-001f29595f9f/vsan-007.vmdk       |                 | 0.68 GB       |
 |    [vsanDatastore] d3703e54-24be-0d86-06e0-001f29595f9f/vsan-007_1.vmdk     |                 | 0.95 GB       |
 | vsan-010                                                                    | 1               |               |
 |    [vsanDatastore] 77713e54-04b3-7214-f997-001b21168828/vsan-010.vmdk       |                 | 0.65 GB       |
 +-----------------------------------------------------------------------------+-----------------+---------------+
 | Total                                                                       | 7               | 5.61 GB       |
 +-----------------------------------------------------------------------------+-----------------+---------------+
 >
```

However, there are also some useful RVC commands to get a more granular state of the recovery/resyncing process.

### vsan.vm_object_info

This command displays the state of individual VM objects. If you look closely at this VM Namespace object (referred to as the Namespace directory below), the Witness component is shown as ABSENT and STALE in the last line of the output.

```
> vsan.vm_object_info 1
 VM vsan-002:
   Namespace directory
     DOM Object: 10703e54-40ab-2706-b9cd-001f29595f9f (v1, owner: cs-ie-
h04.ie.local, policy: forceProvisioning = 0, hostFailuresToTolerate = 1,
spbmProfileId = VSANDefaultProfileId, proportionalCapacity = [0, 100],
spbmProfileGenerationNumber = 0, cacheReservation = 0, stripeWidth = 1)
       RAID_1
         Component: 7e8c7454-d7b2-6ec6-b44b-001f29595f9f (state: ACTIVE (5), host:
cs-ie-h02.ie.local, md: naa.600508b1001c07d525259e83da9541bf, ssd:
naa.600508b1001c577e11dd042e142a583f,
                                                          usage: 0.2 GB)
         Component: be937454-22a0-e950-fb7d-001b21168828 (state: ACTIVE (5), host:
cs-ie-h01.ie.local, md: naa.600508b1001c79748e8465571b6f4a46, ssd:
naa.600508b1001c61cedd42b0c3fbf55132,
                                                          usage: 0.2 GB)
       Witness: c3567854-4aee-aba7-6733-001b21168828 (state: ABSENT (6), csn:
STALE (78!=79), host: LSOM object not found)
   Disk backing: [vsanDatastore] 10703e54-40ab-2706-b9cd-001f29595f9f/vsan-
002.vmdk
```

The STALE state is reflective of the fact that changes have occurred in the configuration since the underlying device failed. As you can see, there is no reference to a host, magnetic disk (md) or solid-state disk (ssd). Instead, we see LSOM (Local Storage Object Manager) object not found. This is not a permanent failure or else the status would be DEGRADED. Since it absent, Virtual SAN believes

this device might come back, so this situation could be a result of a host reboot or a disk that was offlined or ejected from the host. Virtual SAN will wait 60 minutes before rebuilding any of these components by default.

**Note**: The above output is taken from version 5.5. **vsan.vm_object_info** is different in Virtual SAN version 6.0. In the 6.0 version, a new field is included which includes the vote count of the component, as shown here:

```
Component: b91dd354-a3ed-a07c-c799-b8ca3a70ba70 (state: ACTIVE (5), host:
10.155.60.18, md: naa.5000c500581adacb, ssd: naa.55cd2e404b4d06e4, votes: 1,
usage: 0.4 GB)
```

## vsan.resync_dashboard

If the previous situation where a device is ABSENT continues for 60 minutes (by default, but tunable), Virtual SAN will rebuild the affected components of the ABSENT device. The command `vsan.resync_dashboard` will display the re-syncing of the components that are being rebuilt elsewhere in the cluster. Using this command, it is possible to tell how many bytes are left to sync for that particular VM/Object.

| VM/Object | Syncing objects | Bytes to sync |
|---|---|---|
| io-232-vsanDatastore-rhel6-64-vmwpv-lc-0011 | 1 | |
| [vsanDatastore] 67853352-a080-c5e0-af3b-d4ae52659eeb/io-232-vsanDatastore-rhel6-64-vmwpv-lc-0011.vmdk | | 0.07 GB |
| io-51-vsanDatastore-rhel6-64-vmwpv-p-0009 | 1 | |
| [vsanDatastore] 329a3652-a42b-93c6-c71a-d4ae52659f0f/io-51-vsanDatastore-rhel6-64-vmwpv-p-0009-000001.vmdk | | 0.00 GB |
| io-232-vsanDatastore-rhel6-64-vmwpv-lc-0010 | 1 | |
| [vsanDatastore] b1843352-c04b-70c6-571b-d4ae52659eeb/io-232-vsanDatastore-rhel6-64-vmwpv-lc-0010.vmdk | | 0.05 GB |
| io-147-vsanDatastore-rhel6-64-vmwpv-p-0006 | 1 | |
| [vsanDatastore] 8b8b3352-e422-60d2-5ec5-d4ae526536c3/io-147-vsanDatastore-rhel6-64-vmwpv-p-0006-000001.vmdk | | 0.01 GB |
| io-147-vsanDatastore-rhel6-64-vmwpv-p-0003 | 1 | |
| [vsanDatastore] 8a8b3352-bccb-6be0-bfff-d4ae526536c3/io-147-vsanDatastore-rhel6-64-vmwpv-p-0003-000001.vmdk | | 0.01 GB |
| io-232-vsanDatastore-rhel6-64-vmwpv-lc-0057 | 1 | |
| [vsanDatastore] 97883352-80fb-d706-ec01-d4ae52659eeb/io-232-vsanDatastore-rhel6-64-vmwpv-lc-0057.vmx | | 0.01 GB |
| io-245-vsanDatastore-rhel6-64-vmwpv-lc-0055 | 1 | |
| [vsanDatastore] f3883352-fcfa-5e2d-a0a9-d4ae526548f8/io-245-vsanDatastore-rhel6-64-vmwpv-lc-0055.vmx | | 0.00 GB |
| io-188-vsanDatastore-rhel6-64-vmwpv-lc-0011 | 1 | |
| [vsanDatastore] ed853352-8a5f-9544-c2f3-d4ae52652342/io-188-vsanDatastore-rhel6-64-vmwpv-lc-0011.vmdk | | 0.00 GB |
| io-188-vsanDatastore-rhel6-64-vmwpv-lc-0046 | 1 | |
| [vsanDatastore] 48883352-a631-2d8f-d7d2-d4ae52652342/io-188-vsanDatastore-rhel6-64-vmwpv-lc-0046.vmx | | 0.00 GB |
| io-36-vsanDatastore-rhel6-64-vmwpv-np-0013 | 1 | |
| [vsanDatastore] 318b3352-d8d8-7727-aa08-d4ae5265363f/io-36-vsanDatastore-rhel6-64-vmwpv-np-0013.vmx | | 0.00 GB |
| io-188-vsanDatastore-rhel6-64-vmwpv-np-0013 | 1 | |
| [vsanDatastore] 288b3352-d2bd-146c-4be0-d4ae52652342/io-188-vsanDatastore-rhel6-64-vmwpv-np-0013.vmx | | 0.01 GB |
| io-36-vsanDatastore-rhel6-64-vmwpv-lc-0067 | 1 | |
| [vsanDatastore] ad893352-59b2-4f90-1ee3-d4ae5265363f/io-36-vsanDatastore-rhel6-64-vmwpv-lc-0067.vmdk | | 0.00 GB |
| io-51-vsanDatastore-rhel6-64-vmwpv-lc-0010 | 1 | |
| [vsanDatastore] af923652-a082-84d0-0217-d4ae52659f0f/io-51-vsanDatastore-rhel6-64-vmwpv-lc-0010.vmdk | | 0.00 GB |
| io-51-vsanDatastore-rhel6-64-vmwpv-lc-0048 | 1 | |
| [vsanDatastore] 63943652-e8c5-ae6e-2a6a-d4ae52659f0f/io-51-vsanDatastore-rhel6-64-vmwpv-lc-0048.vmdk | | 0.00 GB |
| io-51-vsanDatastore-rhel6-64-vmwpv-lc-0041 | 1 | |
| [vsanDatastore] 35943652-d052-6a1e-5296-d4ae52659f0f/io-51-vsanDatastore-rhel6-64-vmwpv-lc-0041.vmdk | | 0.01 GB |
| io-51-vsanDatastore-rhel6-64-vmwpv-lc-0044 | 1 | |
| [vsanDatastore] 37943652-c8ff-a18a-83a6-d4ae52659f0f/io-51-vsanDatastore-rhel6-64-vmwpv-lc-0044.vmdk | | 0.00 GB |
| Total | 16 | 0.20 GB |

# Testing Virtual SAN functionality - deploying VMs

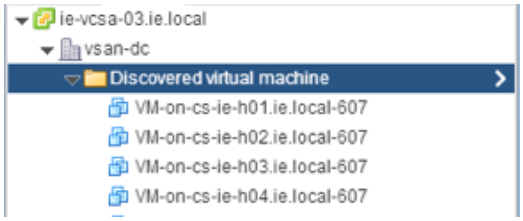
One strong recommendation from VMware is to verify that virtual machines can be successfully deployed on the Virtual SAN datastore once the cluster has been created.

## diagnostics.vm_create

RVC provides a command to do this test. It attempts to create a virtual machine on each host participating in the Virtual SAN cluster. A successful VM creation suggests that the cluster is functioning correctly. If the command fails, useful diagnostic information is displayed which might help determine the root cause. Here is a successful output from the command:

```
/localhost/vsan-dc/computers> diagnostics.vm_create --
datastore ../datastores/vsanDatastore --vm-folder ../vms/Discovered\ virtual\ machine 0
 Creating one VM per host ... (timeout = 180 sec)
 Success
/localhost/vsan-dc/computers>
```

In the vSphere web client, the newly created VMs will be temporarily visible in the folder selected, with the name VM-on-<ESXi-host-name>-XXX:




The VMs are removed automatically once the operation completes.

## diagnostics.vm_create failure – clomd not running

Here is the output from an unsuccessful attempt at creating a VM, with an informative error message. In this example, **clomd,** one of the Virtual SAN daemons**,** was not running on the host in question, and thus the command could not create a VM on that host.

```
/localhost/vsan-dc/computers> diagnostics.vm_create --
datastore ../datastores/vsanDatastore --vm-folder ../vms/Discovered\ virtual\ machine 0
 Creating one VM per host ... (timeout = 180 sec)
 Failed to create VM on host cs-ie-h03.ie.local (in cluster vsan): CannotCreateFile:
Cannot complete file creation operation.
   vob.vsanprovider.object.creation.failed: Failed to create object.
   vob.vsan.dom.noclomattached: A CLOM is not attached. This could indicate that the
clomd daemon is not running.
 /localhost/vsan-dc/computers>
```

# Common storage problems and resolutions

The following is a list of some of the most common storage issues that customers reported with Virtual SAN.

## Virtual SAN claiming disks but capacity not correct

There have been occasions where customers report that the capacity of the VSAN datastore is not shown correctly. We have seen this situation when a customer replaced or changed components on a host that also changed the way the local SCSI disks were presented to the ESXi host. This caused the local VMFS-L volumes on those disks to show up as snapshots and an ESXi host does not mount snapshot volumes by design. The disks are displayed as *In CMMDS: false* in the output of the `esxcli vsan storage list` as shown below:

```
naa.600605b008b04b90ff0000a60a119dd3:
  Device: naa.600605b008b04b90ff0000a60a119dd3
  Display Name: naa.600605b008b04b90ff0000a60a119dd3
  Is SSD: false
  VSAN UUID: 520954bd-c07c-423c-8e42-ff33ca5c0a81
  VSAN Disk Group UUID: 52564730-8bc6-e442-2ab9-6de5b0043d87
  VSAN Disk Group Name: naa.600605b008b04b90ff0000a80a26f73f
  Used by this host: true
  In CMMDS: false
  Checksum: 15088448381607538692
  Checksum OK: true
```

Because the volumes were not mounted, they could not be used to add capacity to the VSAN datastore. In this case, the volumes were identified and resignatured/remounted for them to be added back into CMMDS and thus added to the capacity of the VSAN datastores. Customers with this issue should reference KB article 1011387 or speak to Global Support Services for advice.

## Virtual SAN not claiming disks - existing partition information

A common question is how to repurpose disks that were once used by Virtual SAN but you now wish to use these disks for other purposes? In Virtual SAN 5.5, when you place the host into maintenance mode and remove the disk group from the host, this will automatically remove the partitions from the disks and these disks can now be used for some other purpose.

In Virtual SAN, individual disks can be removed from a disk group, also removing the disk information including partition information.

However, if ESXi is reinstalled on the host that was running Virtual SAN but the appropriate Virtual SAN clean up steps were not first followed, then there may still be Virtual SAN partition information on the disks. This next section will detail how to go about cleaning up these disks.

### esxcli vsan storage remove

ESXCLI contains a nice command to remove physical disks from Virtual SAN disk groups. This is the preferred command from removing disks from Virtual SAN.

```
Usage: esxcli vsan storage remove [cmd options]
```

The command options are -d (for magnetic disks), -s (for SSDs) and -u (for UUID) of Virtual SAN disks.

***Caution**: The -s option for SSD will also remove the magnetic disks from the disk group too, so use this with caution. Removing an SSD from a disk group will invalidate the whole of the disk group.*

### partedUtil

If the disk wasn't previously used by Virtual SAN, but had some other partition information on it, such as VMFS partitions, there is no way via the vSphere Web Client to delete partition information currently. Instead, you the ESXI command `partedUtil` can be used to remove the partitions from the disk and allow Virtual SAN to claim it. You can use `partedUtil` to display the current partitions before deleting them from the disk drive. Using the `getptbl` option rather than the get option displays a more human readable format:

```
 ~ # partedUtil get  /dev/disks/naa.500xxxxxx
15566 255 63 250069680
1 2048 6143 0 0
2 6144 250069646 0 0

~ # partedUtil getptbl /dev/disks/naa.500xxxxxx gpt
15566 255 63 250069680
1 2048 250069646 AA31E02A400F11DB9590000C2911D1B8 vmfs 0


~ # partedUtil delete /dev/disks/naa.500xxxxxx 1
```

Using partedUtil on a disk used by Virtual SAN should only be attempted when Virtual SAN is disabled on the cluster.

### Virtual SAN not claiming disks - Is Local: false

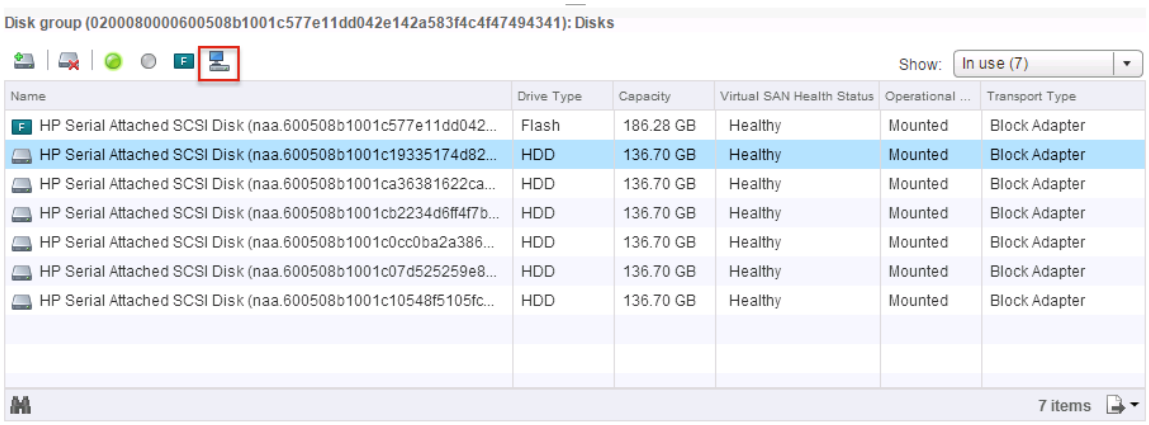
The interest in the "**Is Local"** field relates to whether or not Virtual SAN can automatically claims disks. If the Virtual SAN cluster is configured in *Automatic* mode, one would expect it to claim all of the local disks. This is true, but some SAS controllers report their disks to ESXi as non-local because some SAS controllers allow their disks to be accessed by more than once host.

The symptoms are that in the Virtual SAN General tab, the disks in use will report 0 of X eligible (where X is the number of disks in the hosts participating in the cluster)

and the Total and Free capacity of the Virtual SAN datastore will also be 0 (assuming all the hosts are configured similarly, which they should be if adhering to best practices).

If Virtual SAN is not automatically claiming the disks, the disks will have to be claimed manually by going the Disk Management section of the UI and clicking on 'Select all eligible disks'.

In Virtual SAN 6.0, a new feature has been introduced that allows disks to be marked as local in the UI, if they are not reported as local. This new functionality is available via the vSphere web client. Navigate to the Disk Management section of Virtual SAN, and if you select one of the disks in the disk group, you will be able mark that disk as local by clicking on the icon highlighted below:



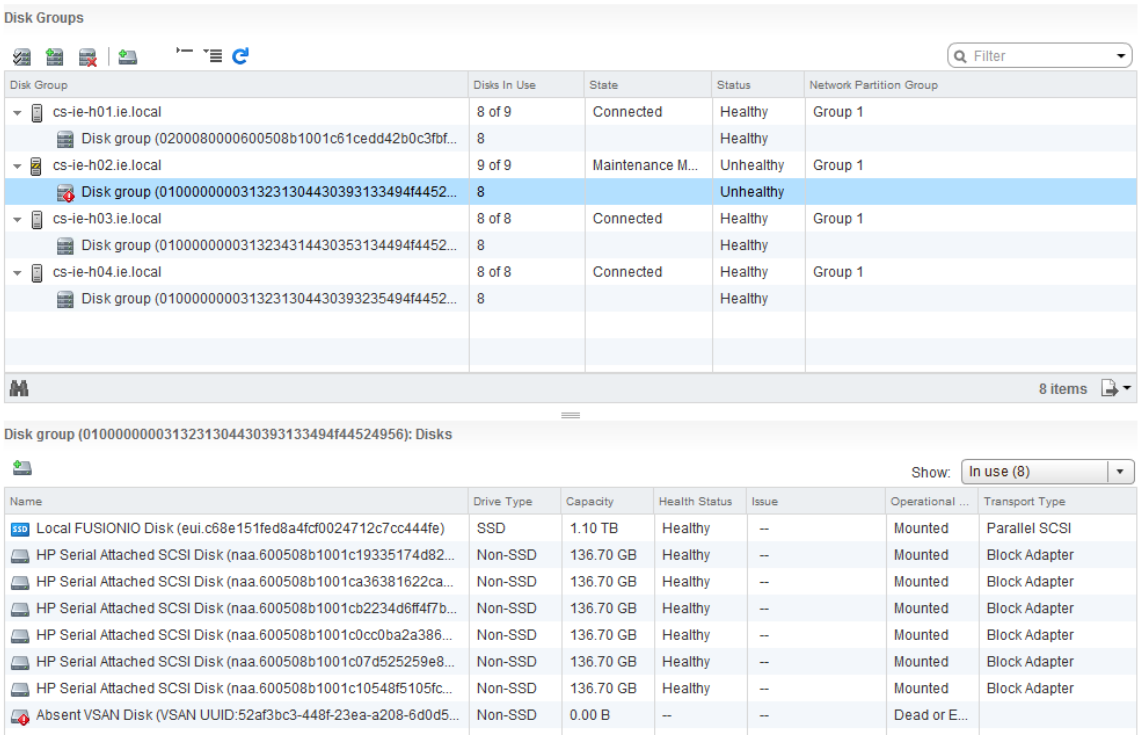
# Virtual SAN storage device failure observations

In this section, events from when disk devices are removed from a host participating in a Virtual SAN cluster are examined, and also what happens when a device actually fails. Both have different behaviors, and need to be understood by anyone who is going to troubleshoot storage issues on Virtual SAN.

## Observations when a disk is failed/removed in a controlled manner

In this example, to observe the behavior of Virtual SAN when a disk is removed, storage I/O controller CLI commands were used to remove a disk from the host. HP CLI commands were used to offline the drive, and observed what the ESXi host and Virtual SAN reported when this event occurred. Virtual SAN marked the disk as ABSENT, giving an administrator a 60-minute window to put the drive back online.

The disk was in a logical RAID 0 volume, as the storage controller in question did not support pass-through. The first thing observed was the Unhealthy disk group state and the drive ABSENT state via the vSphere web client, when the disk groups were examined:



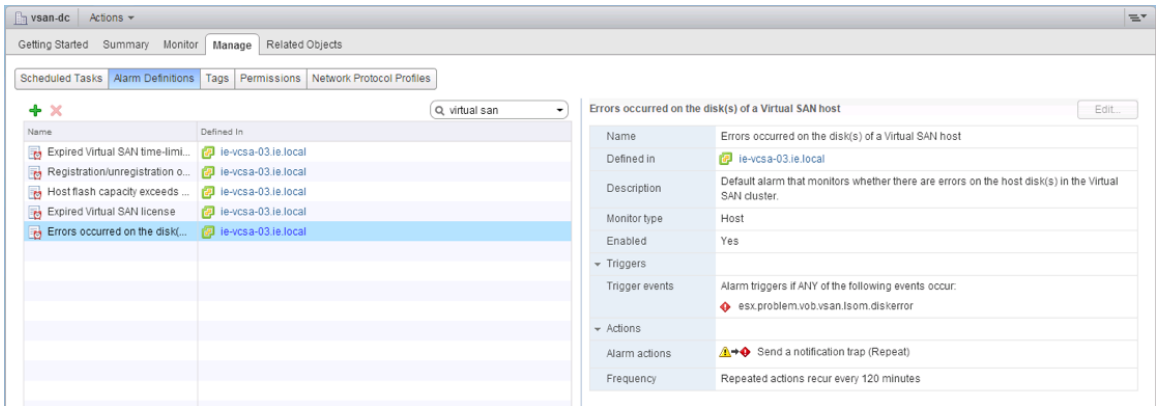

The disk flagged as **Absent** shows an operational state of **Dead or Error**.

The next step is to look at the event logs, as they show some interesting detail:

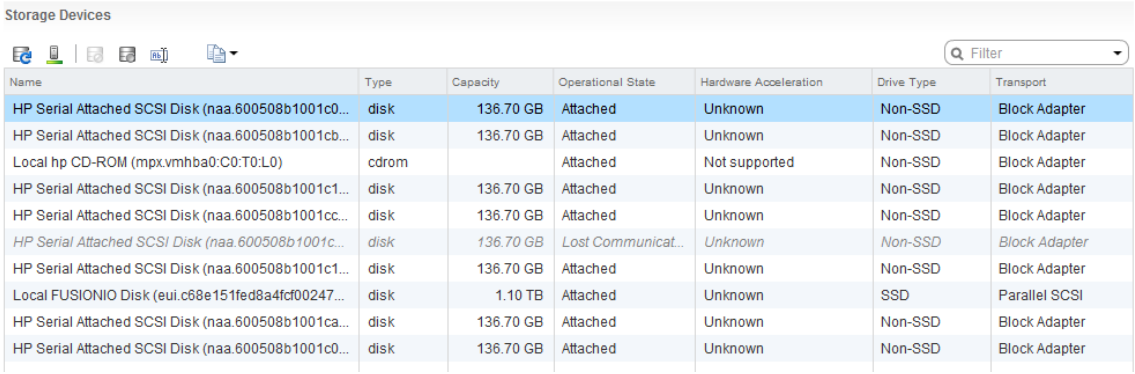
| Description | Type | Date Time |
|---|---|---|
| Permanently inaccessible device naa.600508b1001c4d41121b41182fa83be4 has no more opens. It is now safe to unmount datastores (if any) U... | Information | 28/11/2014 11:16:20 |
| Alarm 'Cannot connect to storage': an SNMP trap for entity cs-ie-h04.ie.local was sent | Information | 28/11/2014 11:16:18 |
| Alarm 'Cannot connect to storage' on cs-ie-h04.ie.local triggered by event 3348 'Lost connectivity to storage device naa.600508b1001c4d41121b41... | Error | 28/11/2014 11:16:18 |
| Alarm 'Cannot connect to storage' on cs-ie-h04.ie.local triggered an action | Information | 28/11/2014 11:16:18 |
| Alarm 'Errors occurred on the disk(s) of a Virtual SAN host': an SNMP trap for entity cs-ie-h04.ie.local was sent | Information | 28/11/2014 11:16:18 |
| Alarm 'Errors occurred on the disk(s) of a Virtual SAN host' on cs-ie-h04.ie.local changed from Gray to Red | Information | 28/11/2014 11:16:18 |
| Alarm 'Errors occurred on the disk(s) of a Virtual SAN host' on cs-ie-h04.ie.local triggered an action | Information | 28/11/2014 11:16:18 |
| Device naa.600508b1001c4d41121b41182fa83be4 has been removed or is permanently inaccessible. Affected datastores (if any): "VSAN-Internal... | Error | 28/11/2014 11:16:08 |
| Device or filesystem with identifier naa.600508b1001c4d41121b41182fa83be4 has exited the All Paths Down state. | Information | 28/11/2014 11:16:08 |
| Lost connectivity to storage device naa.600508b1001c4d41121b41182fa83be4. Path vmhba1:C0:T0:L4 is down. Affected datastores: "VSAN-Intern... | Error | 28/11/2014 11:16:08 |
| User root@127.0.0.1 logged out (login time: Fri Nov 28 11:14:42 UTC 2014, number of API invocations: 0, user agent: ) | Information | 28/11/2014 11:16:08 |
| Virtual SAN device 5227c17e-ec64-de76-c10e-c272102beba7 has gone offline. | Error | 28/11/2014 11:16:08 |
| Device or filesystem with identifier naa.600508b1001c4d41121b41182fa83be4 has entered the All Paths Down state. | Warning | 28/11/2014 11:16:08 |
| Virtual SAN device 5227c17e-ec64-de76-c10e-c272102beba7 is under permanent failure. | Error | 28/11/2014 11:16:08 |
| Virtual SAN device 5227c17e-ec64-de76-c10e-c272102beba7 is under permanent failure. | Error | 28/11/2014 11:16:08 |

The sequence of events starts with Virtual SAN determining that the device has a permanent failure, then the disk going offline, then lost connectivity, then the APD or All Paths Down (APD) to the device being permanently inaccessible.

In version 5.5, there were no alarms for Virtual SAN. These had to be configured manually. In 6.0, a number of alarms have been added and will trigger automatically. By selecting the Virtual SAN cluster object in the vCenter inventory and navigating to the Manage section, the alarm definitions can be viewed. Here is the alarm generated when a disk error occurs. You can also see that it is configured to send an SNMP trap when triggered.



Taking one last view from the vSphere client, this time of the physical disks, a Lost Communication operation state against the disk in question is displayed.




| Name | Type | Capacity | Operational State | Hardware Acceleration | Drive Type | Transport |
|---|---|---|---|---|---|---|
| HP Serial Attached SCSI Disk (naa.600508b1001c0... | disk | 136.70 GB | Attached | Unknown | Non-SSD | Block Adapter |
| HP Serial Attached SCSI Disk (naa.600508b1001cb... | disk | 136.70 GB | Attached | Unknown | Non-SSD | Block Adapter |
| Local hp CD-ROM (mpx.vmhba0:C0:T0:L0) | cdrom | | Attached | Not supported | Non-SSD | Block Adapter |
| HP Serial Attached SCSI Disk (naa.600508b1001c1... | disk | 136.70 GB | Attached | Unknown | Non-SSD | Block Adapter |
| HP Serial Attached SCSI Disk (naa.600508b1001cc... | disk | 136.70 GB | Attached | Unknown | Non-SSD | Block Adapter |
| *HP Serial Attached SCSI Disk (naa.600508b1001c...* | *disk* | *136.70 GB* | *Lost Communicat...* | *Unknown* | *Non-SSD* | *Block Adapter* |
| HP Serial Attached SCSI Disk (naa.600508b1001c1... | disk | 136.70 GB | Attached | Unknown | Non-SSD | Block Adapter |
| Local FUSIONIO Disk (eui.c68e151fed8a4fcf00247... | disk | 1.10 TB | Attached | Unknown | SSD | Parallel SCSI |
| HP Serial Attached SCSI Disk (naa.600508b1001ca... | disk | 136.70 GB | Attached | Unknown | Non-SSD | Block Adapter |
| HP Serial Attached SCSI Disk (naa.600508b1001c0... | disk | 136.70 GB | Attached | Unknown | Non-SSD | Block Adapter |

It is also beneficial to use the CLI in these situations, as additional information can be gleaned.

## esxcli vsan storage list - unknown

This first ESXCLI command lists the storage on the host but the last entry in this truncated list displays a lot of *unknowns*.

```
# esxcli vsan storage list
 naa.600508b1001cadff5d80ba7665b8f09a
    Device: naa.600508b1001cadff5d80ba7665b8f09a
    Display Name: naa.600508b1001cadff5d80ba7665b8f09a
    Is SSD: false
    VSAN UUID: 5209c0c3-2143-7031-b404-060642590295
    VSAN Disk Group UUID: 52e4dd37-f670-2125-2a4b-14eb7a0a6140
    VSAN Disk Group Name: naa.600508b1001c40e393b73af79eacdcde
    Used by this host: true
    In CMMDS: true
    Checksum: 108245589703025155
    Checksum OK: true
    Emulated DIX/DIF Enabled: false

 Unknown
    Device: Unknown
    Display Name: Unknown
    Is SSD: false
    VSAN UUID: 5227c17e-ec64-de76-c10e-c272102beba7
    VSAN Disk Group UUID:
    VSAN Disk Group Name:
    Used by this host: false
    In CMMDS: false
    Checksum:
    Checksum OK: false
    Emulated DIX/DIF Enabled: false

.
.
<<truncated>>
```

Also note that the disk is no longer claimed by Virtual SAN as the `In CMMDS: false` indicates. This is the cluster monitoring and membership service, and Virtual SAN is no longer claiming it.

### vdq –qH: IsPDL

One other CLI command is the `vdq` command. This command has one very useful field in situations like this, and that is "***IsPDL?"***. It basically highlights if the device is in a state of PDL, or permanent device loss, which it is.

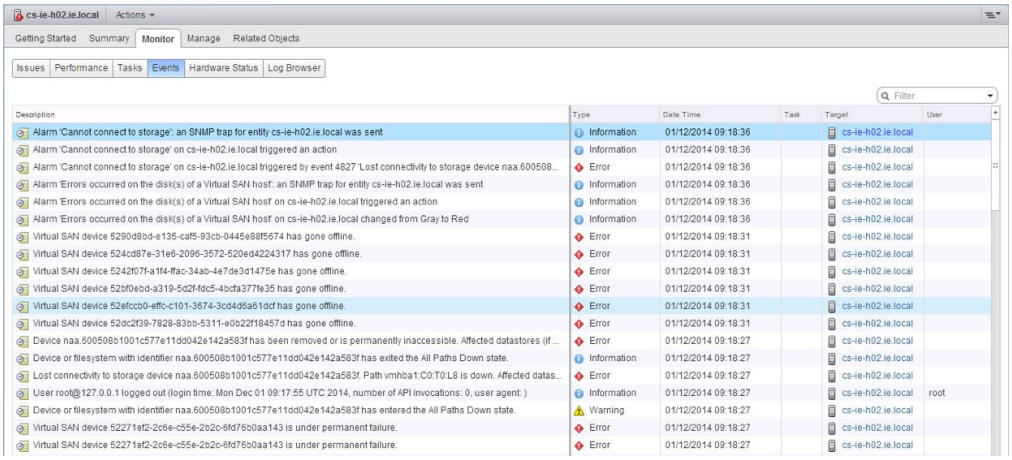
```
[root@cs-ie-h04:~] vdq -qH
 DiskResults:
       DiskResult[0]:
             Name:  naa.600508b1001c4b820b4d80f9f8acfa95
        VSANUUID:  5294bbd8-67c4-c545-3952-7711e365f7fa
            State:  In-use for VSAN
  ChecksumSupport:  0
           Reason:  Non-local disk
           IsSSD?:  0
 IsCapacityFlash?:  0
           IsPDL?:  0

       <<truncated>>

       DiskResult[18]:
             Name:
        VSANUUID:  5227c17e-ec64-de76-c10e-c272102beba7
            State:  In-use for VSAN
  ChecksumSupport:  0
           Reason:  None
           IsSSD?:  0
 IsCapacityFlash?:  0
           IsPDL?:  1                <<< Device is in PDL state
```

These are some of the observations one might expect to see when a device is removed from an ESXi host participating in a Virtual SAN cluster, and that device is part of a Virtual SAN disk group. The Virtual SAN Administrators Guide should be referenced from the correct procedures to remove and replace disk devices from a Virtual SAN cluster.

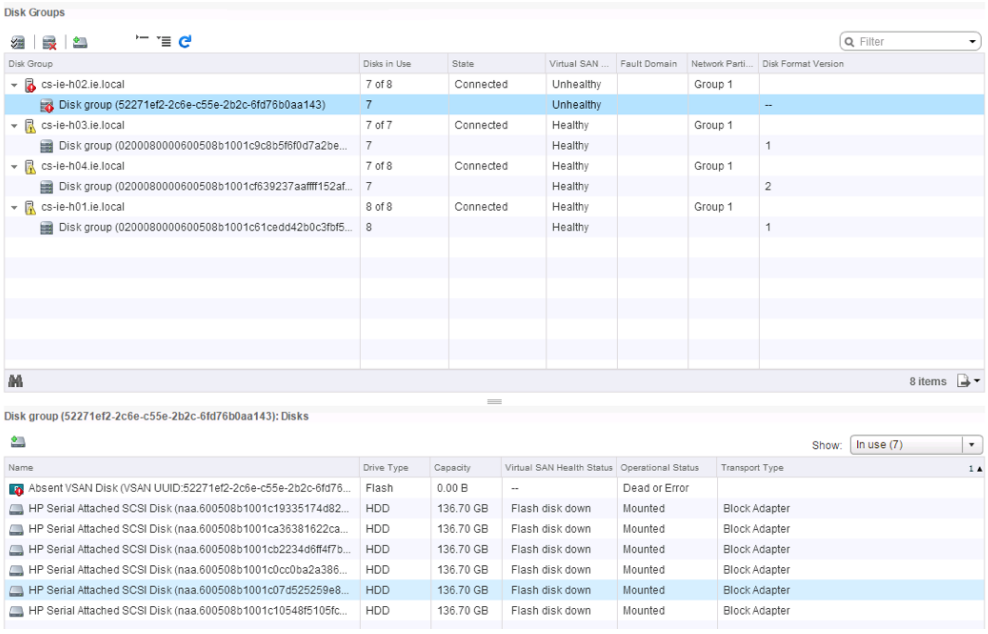
Finally, taking a drive offline using third party tools, or ejecting it form the host, may not induce an error condition, and Virtual SAN will typically treat this device as ABSENT, not DEGRADED. This means that rebuilding of components will not happen immediately if the state is ABSENT. Instead, Virtual SAN waits the perquisite 60 minutes before beginning the resyncing and reconfiguring of components. It is important to keep this in mind when failure testing, especially drive failure testing, on Virtual SAN.

## Observations when a flash device fails

In this example, the flash cache device (in this case an SSD) was removed from the disk group. The events were similar to what was observed with a magnetic disk removal. Virtual SAN determines that the device has a permanent failure, enters APD state, and the device being marked as permanently inaccessible. Note however that all the magnetic disks in the disk group also go offline.




When a flash cache device in a disk group is impacted by a failure, the whole of the disk group is impacted. The disk group status in the vSphere web client shows the overall disk group is now "Unhealthy". The status of the magnetic disks in the same disk group shows "Flash disk down".




If the timeout expires before the original device has been reinserted in the host, the rebuilding of components in the disk group takes place.

### Observations when a storage controller fails

In this particular case, an LSI MegaRAID storage controller had issues when it was using old driver/firmware in Virtual SAN. The following are `vmkernel.log` samples taken from each of the hosts in the Virtual SAN cluster.

Controller resets:

```
2014-08-24T17:00:25.940Z cpu29:33542)<4>megasas: Found FW in FAULT state, will reset
adapter.
2014-08-24T17:00:25.940Z cpu29:33542)<4>megaraid_sas: resetting fusion adapter.
```

I/Os fail due to controller issue (SCSI write is Cmd 0x2a):

```
2014-08-24T17:00:25.940Z cpu34:9429858)NMP: nmp_ThrottleLogForDevice:2321: Cmd 0x2a
(0x4136803d32c0, 0) to dev "naa.50015178f3636429" on path "vmhba0:C0:T4:L0" Failed:
H:0x8 D:0x0 P:0x0 Possible sense data: 0x0 0x0 0x0. Act:EVAL

2014-08-24T17:00:25.940Z cpu34:9429858)WARNING: NMP:
nmp_DeviceRequestFastDeviceProbe:237: NMP device "naa.50015178f3636429" state in
doubt; requested fast path state update...

2014-08-24T17:00:25.940Z cpu34:9429858)ScsiDeviceIO: 2324: Cmd(0x41368093bd80) 0x2a,
CmdSN 0x648c4f3b from world 0 to dev "naa.50015178f3636429" failed H:0x8 D:0x0 P:0x0
Possible sense data: 0x0 0x0 0x0.

2014-08-24T17:00:25.940Z cpu34:9429858)ScsiDeviceIO: 2324: Cmd(0x4136e17d15c0) 0x2a,
CmdSN 0x648c4ee8 from world 0 to dev "naa.50015178f3636429" failed H:0x8 D:0x0 P:0x0
Possible sense data: 0x0 0x0 0x0.

2014-08-24T17:00:25.940Z cpu34:9429858)NMP: nmp_ThrottleLogForDevice:2321: Cmd 0x2a
(0x4136e2370d40, 0) to dev "naa.50015178f3636429" on path "vmhba0:C0:T4:L0" Failed:
H:0x8 D:0x0 P:0x0 Possible sense data: 0x0 0x0 0x0. Act:EVAL

2014-08-24T17:00:25.940Z cpu34:9429858)ScsiDeviceIO: 2324: Cmd(0x41370c3043c0) 0x2a,
CmdSN 0x648c4f3a from world 0 to dev "naa.50015178f3636429" failed H:0x8 D:0x0 P:0x0
Possible sense data: 0x0 0x0 0x0.

2014-08-24T17:00:25.940Z cpu34:9429858)ScsiDeviceIO: 2324: Cmd(0x4136e17d4680) 0x2a,
CmdSN 0x648c4eeb from world 0 to dev "naa.50015178f3636429" failed H:0x8 D:0x0 P:0x0
Possible sense data: 0x0 0x0 0x0.

2014-08-24T17:00:25.940Z cpu34:9429858)NMP: nmp_ThrottleLogForDevice:2321: Cmd 0x2a
(0x4136e07e1700, 0) to dev "naa.50015178f3636429" on path "vmhba0:C0:T4:L0" Failed:
H:0x8 D:0x0 P:0x0 Possible sense data: 0x0 0x0 0x0. Act:EVAL

2014-08-24T17:00:25.940Z cpu34:9429858)NMP: nmp_ThrottleLogForDevice:2321: Cmd 0x28
(0x4136e884c500, 0) to dev "naa.5000c500583c4b1f" on path "vmhba0:C0:T6:L0" Failed:
H:0x8 D:0x0 P:0x0 Possible sense data: 0x0 0x0 0x0. Act:EVAL 3)
```

When the MegaRAID firmware is resetting, it simply takes too long for Virtual SAN to hold off, and eventually fails because it simply took to long for the controller to come back online. The retries are visible in the logs, as well as the maximum number of kernel-level retries exceeded:

```
2014-08-24T17:00:30.845Z cpu38:33542)<6>megasas: Waiting for FW to come to ready state
[ … ]

2014-08-24T17:00:30.912Z cpu20:33167)LSOMCommon: IORETRYCompleteIO:389: Throttled:
0x413701b8cc40 IO type 265 (READ) isOdered:NO since 30001 msec status Maximum kernel-
level retries exceeded
```

```
2014-08-24T17:00:30.912Z cpu33:7027198)WARNING: LSOM: LSOMEventNotify:4570: VSAN
device 52378176-a9da-7bce-0526-cdf1d863b3b5 is under permanent error.

2014-08-24T17:00:30.912Z cpu33:7027198)WARNING: LSOM: RCVmfsIoCompletion:99:
Throttled: VMFS IO failed. Wake up 0x4136af9a69c0 with status Maximum kernel-level
retries exceeded

2014-08-24T17:00:30.912Z cpu33:7027198)WARNING: LSOM: RCDrainAfterBERead:5070:
Changing the status of child state from Success to Maximum kernel-level retries
exceeded
```

Eventually, the firmware reset on the controller completed, but by this time, it is too late and Virtual SAN had already marked the disks as failed:

```
2014-08-24T17:00:49.279Z cpu21:33542)megasas: FW now in Ready state

2014-08-24T17:00:49.299Z cpu21:33542)<3>megasas:IOC Init cmd success

2014-08-24T17:00:49.320Z cpu36:33542)<4>megaraid_sas: Reset successful.
```

When a controller 'wedges' like this, Virtual SAN will retry I/O for a finite amount of time. In this case, it took a full 24 seconds (2400ms) for the adapter to come back online after resetting. This was too long for Virtual SAN, which meant that the maximum retries threshold had been exceeded. This in turn led to Virtual SAN marking the disks as DEGRADED.

Virtual SAN is responding as designed here. The problem is that the firmware crashed. This particular issue was resolved by using recommended versions of MegaRAID driver and firmware as per the VMware Compatibility Guide.

### Storage controller replacement

In general, controller replacement should be for the same make and model and administrators should not be swapping a pass-through controller with a RAID 0 controller or vice-versa.

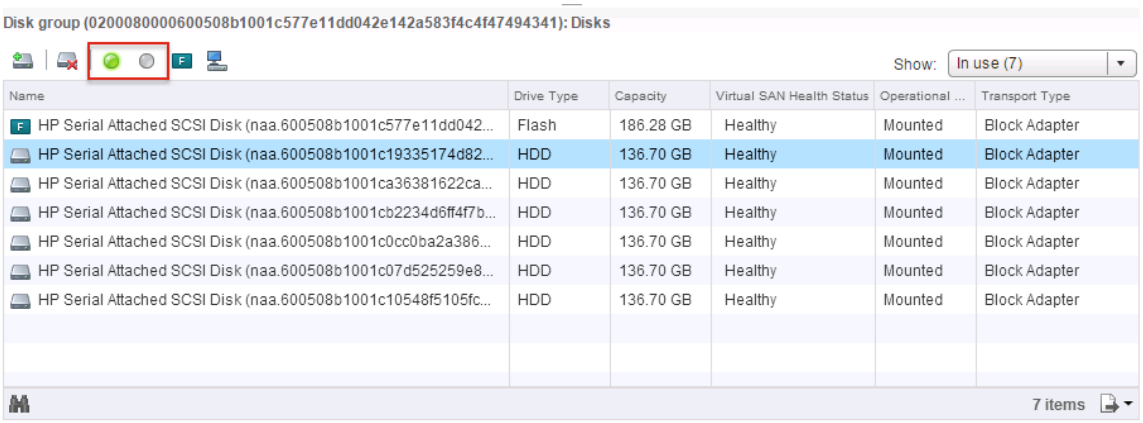
### Expectations when a drive is reporting errors

In this scenario, a disk drive is reporting errors due to bad blocks. If a read I/O accessing component data on behalf of a virtual machine fails in such a manner, Virtual SAN will check other replicas of the component to satisfy the read. If another mirror can satisfy the read, Virtual SAN will attempt to write the good data onto the disk that reported the bad read. If this procedure succeeds, the disk does not enter a DEGRADED state. Note that this is not the behavior if we get a read error when accessing Virtual SAN metadata. If a read of metadata, or any write fails, Virtual SAN will mark all components of the disk as DEGRADED. It treats the disk as failed and the data is no longer usable. Upon entering the DEGRADED state, Virtual SAN will restore I/O flow immediately (by taking the bad component out of the active set of the effected object) and try to re-protect the object by creating a new replica of the component somewhere else in the cluster.

# Blinking LEDs on drives

Virtual SAN 6.0 introduces new functionality to make it easier to identify the location of disk drives within a datacenter. This new functionality is available via the vSphere web client. Note that this is supported only with specific controller/enclosure combinations. The user should refer to the VCG for details.

Navigate to the Disk Management section of Virtual SAN, and select one of the disks in the disk group. Administrators will be able to turn on and off the locator LED of a particular disk using the icons highlighted below:



# Predictive reporting - smartd

In ESXi 5.1, VMware introduced support for SMART, Self-Monitoring, Analysis and Reporting Technology for disk drives. If a drive supports SMART, some useful fields can be extracted from the disk drive, especially solid-state disks, which can help with proactive reporting. For example, it can help determine if there are any read or write errors counting, or the drive overheating, and so on.

Unfortunately, not every drive supports every smart parameter, and the output may contain some fields displayed as N/A as shown below. Some drives that do not support smart will show every field as N/A. However, for drives that do support smart, some really useful information can be extracted using the command shown here:

## esxcli storage core device smart get

```
[root@cs-tse-d01:~] esxcli storage core device smart get -d
t10.ATA_____DELL_P320h2DMTFDGAL175SAH_____
 ____________00000000012270202CF28
 Parameter                     Value  Threshold  Worst
 ----------------------------  -----  ---------  -----
 Health Status                 OK     N/A        N/A
 Media Wearout Indicator       N/A    N/A        N/A
 Write Error Count             0      0          0
 Read Error Count              0      0          100
 Power-on Hours                5890   0          100
 Power Cycle Count             21     0          100
 Reallocated Sector Count      N/A    N/A        N/A
 Raw Read Error Rate           N/A    N/A        N/A
 Drive Temperature             37     87         47
 Driver Rated Max Temperature  102    N/A        N/A
 Write Sectors TOT Count       N/A    N/A        N/A
 Read Sectors TOT Count        N/A    N/A        N/A
 Initial Bad Block Count       1      0          100
 [root@cs-tse-d01:~]
```

Note that when RAID 0 is used on the controller to present disk devices to the ESXi host, on many occasions the controller blocks the SMART (Self-Monitoring, Analysis and Reporting Technology) information and disk type information. This is another reason to use controllers that support pass-through mode where possible.

## Considerations when cloning on Virtual SAN

Cloning is a very sequential I/O intensive operation.

Consider a hybrid Virtual SAN environment where you wish to clone a number of VMs to the Virtual SAN datastore. Virtual SAN may be able to write into the flash cache write buffer at approximately 200-300 MB per second. A single magnetic disk can maybe do 100MB per second. So assuming no read operations are taking place at the same time, we would need 2-3 magnetic disks to match the flash cache speed for destaging purposes.

Similarly, if one has a full disk group with 7 disks, the total bandwidth available on the magnetic disk is 700 MB/s. However the flash cache device can only do half of this and thus the bottleneck maybe with the flash device, not the magnetic disk.

To summarize, sustained sequential write workloads (such as VM cloning operations) run on hybrid Virtual SAN configurations may simply fill the cache and future writes will need to wait for the cache to be destaged to the spinning magnetic disk layer before that can be written to cache, so performance from these types of operations will be a reflection of the spinning disk(s) and not of flash.

Conversely, a clone operation that is utilizing all magnetic disks in a disk group may be impacted by the performance of the flash device.

## A note about vsanSparse virtual disk format

A new disk format called vsanSparse is introduced in Virtual SAN 6.0. This replaces the older vmfsSparse (redo log) format used by snapshots for the delta disk. vsanSparse uses the new (VirstoFS) sparseness and caching features of the newer v2 on-disk format. The new vsanSparse format is expected to have much improved performance comparable to the older vmfsSparse.

Snapshots of virtual machines running on the older v1 on-disk format will continue to use the vmfsSparse format. This includes Virtual SAN 5.5, and Virtual SAN 6.0 that has not had its on-disk format upgraded to v2 and continues to use v1. Snapshots taken of virtual machines running on the new v2 on-disk format will use the new vsanSparse format.

There are no customer visible functional changes when using the new vsanSparse format when compared to the earlier vmfsSparse format. All snapshots activities and operations remain the same.

## Summary checklist for Virtual SAN storage

1. Are all hosts uniformly configured from a storage perspective?
2. Is the SSD on the VCG, and is it a class suitable for your workloads?
3. Is the storage controller on the VCG?
4. Does it have the correct driver and firmware versions?
5. Does it support Pass-thru or RAID-0? If the latter, is RAID-0 configured?
6. Read and write cache disabled on storage controller?
7. If cache cannot be disabled on storage controller, is it set to 100% read?
8. If HP controllers, is HP SSD Smart Path disabled?
9. Are all disk & flash devices visible on ESXi host?
10. Is there any existing partition information on the disks? Might explain why disk not visible.
11. Is cluster in Auto or Manual mode? Manual mode means you have to create disk groups manually.
12. Is RAID-0 configured on SSD, you may need to add new claim rules to add SSD attribute to device. Might explain why SSD is not showing up as SSD.
13. Is Local set to false due to SAS controller? If false, automatic mode won't claim disks and you will have to create disk groups manually.
14. Are the disk drives and flash devices claimed by CCMDS?
15. Can Virtual SAN successfully build a diagnostic VM?
16. How much spare capacity is in the Virtual SAN cluster? Will objects rebuild on spare capacity if there is a failure?
17. How much spare capacity is in the Virtual SAN cluster? Is the spare capacity available across all hosts to allow my VM to be deployed?

# 12. Troubleshooting Virtual SAN Upgrades

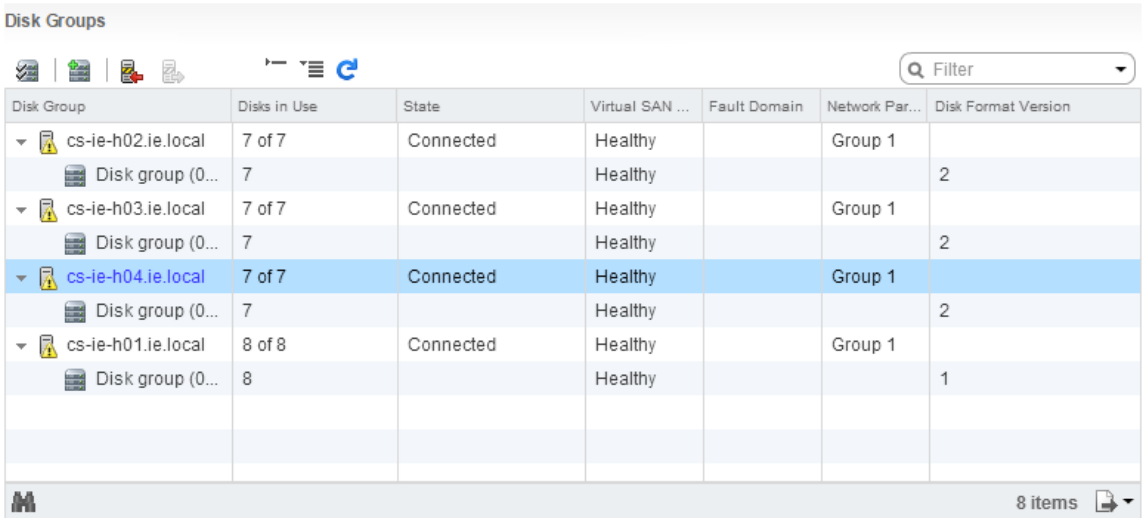
The complete Virtual SAN upgrade procedure is documented in the Virtual SAN 6.0 Administrators Guide. This section of the troubleshooting reference manual is simply to demonstrate observations during a Virtual SAN On-Disk format upgrade, and provide insight into when things may not proceed according to plan.

## Virtual SAN upgrade - on-disk format v2

The upgrade process is 2-phase:

- Software upgrade: Upgrade software from vSphere 5.5 to 6.0.
- Format conversion: Reformat disk groups to on-disk format v2.

With the release of Virtual SAN 6.0, there now exists two different disk formats for Virtual SAN. To check which version you are using, from the vSphere web client UI, navigate to the Virtual SAN view, under disk group list and look for disk format version column.




| Disk Group | Disks in Use | State | Virtual SAN ... | Fault Domain | Network Par... | Disk Format Version |
|---|---|---|---|---|---|---|
| cs-ie-h02.ie.local | 7 of 7 | Connected | Healthy | | Group 1 | |
| Disk group (0... | 7 | | Healthy | | | 2 |
| cs-ie-h03.ie.local | 7 of 7 | Connected | Healthy | | Group 1 | |
| Disk group (0... | 7 | | Healthy | | | 2 |
| cs-ie-h04.ie.local | 7 of 7 | Connected | Healthy | | Group 1 | |
| Disk group (0... | 7 | | Healthy | | | 2 |
| cs-ie-h01.ie.local | 8 of 8 | Connected | Healthy | | Group 1 | |
| Disk group (0... | 8 | | Healthy | | | 1 |

In this example, 3 out of 4 hosts are at v2. One is still at v1, and still needs to be upgraded. Running Virtual SAN with different on-disk formats is **not supported**.

### Before you start upgrading the on-disk format

Before doing an upgrade, ensure that there are no unhealthy disks in your Virtual SAN Cluster. To view the disk status, run the RVC `vsan.disks_stats` command. The command lists the names of all disks and hosts in the Virtual SAN. Use this command to verify the current format and the health status of the disk. The status appears as **Unhealthy** in the Virtual SAN **Health Status** column (in the **Disk Management** page) for the hosts or disk groups with a failed disk.

## On-disk format upgrade pre-check: vsan.disks_stats

The `vsan.disks_stats` in version 6.0 will also display the version number of the on-disk format displayed in the right hand column (e.g. v1 or v2). Here is a cluster upgraded to vSphere version 6.0 but still using the original on-disk format (v1). Note the (v1) in the Status Health column on the right hand side:

```
vsan.disks_stats /localhost/ie-datacenter-04/computers/ie-vsan/
2014-11-10 15:08:46 +0000: Fetching VSAN disk info from cs-ie-h04.ie.local (may take a moment) ...
2014-11-10 15:08:46 +0000: Fetching VSAN disk info from cs-ie-h01.ie.local (may take a moment) ...
2014-11-10 15:08:46 +0000: Fetching VSAN disk info from cs-ie-h02.ie.local (may take a moment) ...
2014-11-10 15:08:46 +0000: Fetching VSAN disk info from cs-ie-h03.ie.local (may take a moment) ...
2014-11-10 15:08:48 +0000: Done fetching VSAN disk infos
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
|                                      |                    |       | Num  | Capacity  |      |          | Status  |
| DisplayName                          | Host               | isSSD | Comp | Total     | Used | Reserved | Health  |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c61cedd42b0c3fbf55132 | cs-ie-h01.ie.local | SSD   | 0    | 130.39 GB | 0 %  | 0 %      | OK (v1) |
| naa.600508b1001c2ee9a6446e708105054b | cs-ie-h01.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001c16be6e256767284eaf88 | cs-ie-h01.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001ccd5d506e7ed19c40a64c | cs-ie-h01.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001c64816271482a56a48c3c | cs-ie-h01.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001c388c92e817e43fcd5237 | cs-ie-h01.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001c3ea7838c0436dbe6d7a2 | cs-ie-h01.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001c79748e8465571b6f4a46 | cs-ie-h01.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c577e11dd042e142a583f | cs-ie-h02.ie.local | SSD   | 0    | 130.39 GB | 0 %  | 0 %      | OK (v1) |
| naa.600508b1001ca36381622ca880f3aacd | cs-ie-h02.ie.local | MD    | 25   | 136.50 GB | 10 % | 9 %      | OK (v1) |
| naa.600508b1001cb2234d6ff4f7b1144f59 | cs-ie-h02.ie.local | MD    | 18   | 136.50 GB | 18 % | 17 %     | OK (v1) |
| naa.600508b1001c19335174d82278dee603 | cs-ie-h02.ie.local | MD    | 1    | 136.50 GB | 92 % | 92 %     | OK (v1) |
| naa.600508b1001c0cc0ba2a3866cf8e28be | cs-ie-h02.ie.local | MD    | 21   | 136.50 GB | 26 % | 25 %     | OK (v1) |
| naa.600508b1001c10548f5105fc60246b4a | cs-ie-h02.ie.local | MD    | 14   | 136.50 GB | 10 % | 10 %     | OK (v1) |
| naa.600508b1001c07d525259e83da9541bf | cs-ie-h02.ie.local | MD    | 24   | 136.50 GB | 92 % | 92 %     | OK (v1) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c9c8b5f6f0d7a2be44433 | cs-ie-h03.ie.local | SSD   | 0    | 130.39 GB | 0 %  | 0 %      | OK (v1) |
| naa.600508b1001ceefc4213ceb9b51c4be4 | cs-ie-h03.ie.local | MD    | 52   | 136.50 GB | 8 %  | 7 %      | OK (v1) |
| naa.600508b1001cd259ab7ef213c87eaad7 | cs-ie-h03.ie.local | MD    | 1    | 136.50 GB | 92 % | 92 %     | OK (v1) |
| naa.600508b1001c1a7f310269ccd51a4e83 | cs-ie-h03.ie.local | MD    | 2    | 136.50 GB | 92 % | 92 %     | OK (v1) |
| naa.600508b1001c9b93053e6dc3ea9bf3ef | cs-ie-h03.ie.local | MD    | 23   | 136.50 GB | 31 % | 13 %     | OK (v1) |
| naa.600508b1001cb11f3292fe743a0fd2e7 | cs-ie-h03.ie.local | MD    | 13   | 136.50 GB | 7 %  | 7 %      | OK (v1) |
| naa.600508b1001c2b7a3d39534ac6beb92d | cs-ie-h03.ie.local | MD    | 14   | 136.50 GB | 9 %  | 8 %      | OK (v1) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c40e393b73af79eacdcde | cs-ie-h04.ie.local | SSD   | 0    | 130.39 GB | 0 %  | 0 %      | OK (v1) |
| naa.600508b1001c51f3a696fe0bbbcb5096 | cs-ie-h04.ie.local | MD    | 9    | 136.50 GB | 4 %  | 4 %      | OK (v1) |
| naa.600508b1001c4b820b4d80f9f8acfa95 | cs-ie-h04.ie.local | MD    | 9    | 136.50 GB | 5 %  | 4 %      | OK (v1) |
| naa.600508b1001cadff5d80ba7665b8f09a | cs-ie-h04.ie.local | MD    | 53   | 136.50 GB | 5 %  | 5 %      | OK (v1) |
| naa.600508b1001c846c000c3d9114ed71b3 | cs-ie-h04.ie.local | MD    | 10   | 136.50 GB | 5 %  | 5 %      | OK (v1) |
| naa.600508b1001c258181f0a088f6e40dab | cs-ie-h04.ie.local | MD    | 12   | 136.50 GB | 13 % | 12 %     | OK (v1) |
| naa.600508b1001c4d41121b41182fa83be4 | cs-ie-h04.ie.local | MD    | 10   | 136.50 GB | 5 %  | 4 %      | OK (v1) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
```

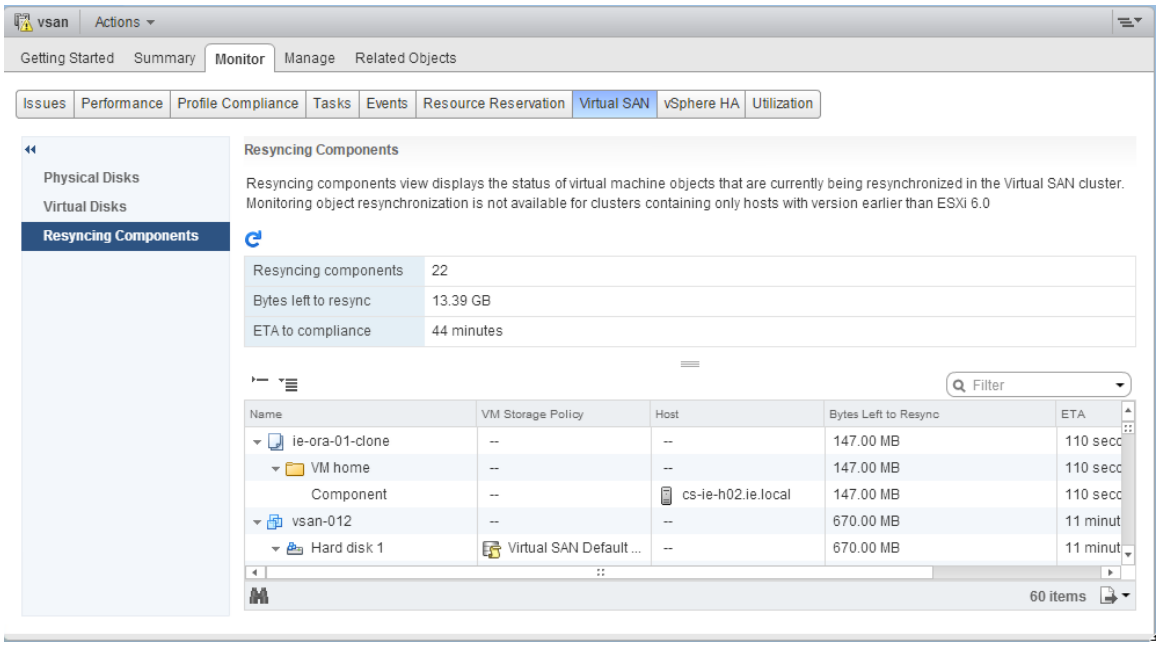
If everything is healthy, there is a new Virtual SAN 6.0 command, `vsan.v2_ondisk_upgrade`, to upgrade the on-disk format.

## On-disk format upgrade: vsan.v2_ondisk_upgrade

This command will rotate through each of the hosts in the Virtual SAN cluster (rolling upgrade), doing a number of verification checks on the state of the host and cluster before evacuating components from each of the disk groups and rebuilding them elsewhere in the cluster.

It then upgrades the on-disk format from v1 to v2. Even though the command is an RVC command, you will be able to monitor the tasks associated with the on-disk upgrade via the vSphere web client UI. In particular, the task *Remove disks from use by Virtual SAN* followed by *Add disks to Virtual SAN* will be seen.


Using the Monitor > Virtual SAN > Resyncing components view, you can also see how much data is left to resync, and an ETA to compliance:




Note that there is no need to manually place the host in maintenance mode or evacuate any data. The RVC script will handle all of these tasks.

Here is an example of how to run the command, showing a snippet from one on-disk format upgrade from v1 to v2:

```
/ie-vcsa-03.ie.local/vsan-dc/computers> vsan.v2_ondisk_upgrade 0
+--------------------+-----------+-------------+----------------+----------------+
| Host               | State     | ESX version | v1 Disk-Groups | v2 Disk-Groups |
+--------------------+-----------+-------------+----------------+----------------+
| cs-ie-h02.ie.local | connected | 6.0.0       | 1              | 0              |
| cs-ie-h03.ie.local | connected | 6.0.0       | 1              | 0              |
| cs-ie-h04.ie.local | connected | 6.0.0       | 1              | 0              |
| cs-ie-h01.ie.local | connected | 6.0.0       | 1              | 0              |
+--------------------+-----------+-------------+----------------+----------------+

2014-12-10 14:49:16 +0000: Running precondition checks ...
2014-12-10 14:49:19 +0000: Passed precondition checks
2014-12-10 14:49:19 +0000:
2014-12-10 14:49:19 +0000: Target file system version: v2
2014-12-10 14:49:19 +0000: Disk mapping decommission mode: evacuateAllData
2014-12-10 14:49:28 +0000: Cluster is still in good state, proceeding ...
2014-12-10 14:49:28 +0000: Enabled v2 filesystem as default on host cs-ie-h02.ie.local
2014-12-10 14:49:28 +0000: Removing VSAN disk group on cs-ie-h02.ie.local:
2014-12-10 14:49:28 +0000:   SSD: HP Serial Attached SCSI Disk (naa.600508b1001c64b76c8ceb56e816a89d)
2014-12-10 14:49:28 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c19335174d82278dee603)
2014-12-10 14:49:28 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001ca36381622ca880f3aacd)
2014-12-10 14:49:28 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001cb2234d6ff4f7b1144f59)
2014-12-10 14:49:28 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c0cc0ba2a3866cf8e28be)
2014-12-10 14:49:28 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c07d525259e83da9541bf)
2014-12-10 14:49:28 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c10548f5105fc60246b4a)
RemoveDiskMapping cs-ie-h02.ie.local: success
2014-12-10 15:20:40 +0000: Re-adding disks to VSAN on cs-ie-h02.ie.local:
2014-12-10 15:20:40 +0000:   SSD: HP Serial Attached SCSI Disk (naa.600508b1001c64b76c8ceb56e816a89d)
2014-12-10 15:20:40 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c19335174d82278dee603)
2014-12-10 15:20:40 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001ca36381622ca880f3aacd)
2014-12-10 15:20:40 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001cb2234d6ff4f7b1144f59)
2014-12-10 15:20:40 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c0cc0ba2a3866cf8e28be)
2014-12-10 15:20:40 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c07d525259e83da9541bf)
2014-12-10 15:20:40 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c10548f5105fc60246b4a)
AddDisks cs-ie-h02.ie.local: success
2014-12-10 15:21:13 +0000: Done upgrade host cs-ie-h02.ie.local
2014-12-10 15:21:16 +0000:
2014-12-10 15:21:16 +0000: Cluster is still in good state, proceeding ...
2014-12-10 15:21:16 +0000: Enabled v2 filesystem as default on host cs-ie-h03.ie.local
2014-12-10 15:21:16 +0000: Removing VSAN disk group on cs-ie-h03.ie.local:
2014-12-10 15:21:16 +0000:   SSD: HP Serial Attached SCSI Disk (naa.600508b1001c9c8b5f6f0d7a2be44433)
2014-12-10 15:21:16 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001ceefc4213ceb9b51c4be4)
2014-12-10 15:21:16 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001cd259ab7ef213c87eaad7)
2014-12-10 15:21:16 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c2b7a3d39534ac6beb92d)
2014-12-10 15:21:16 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001cb11f3292fe743a0fd2e7)
2014-12-10 15:21:16 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c1a7f310269ccd51a4e83)
2014-12-10 15:21:16 +0000:   HDD: HP Serial Attached SCSI Disk
(naa.600508b1001c9b93053e6dc3ea9bf3ef)

RemoveDiskMapping cs-ie-h03.ie.local: running
[================================================== <<truncated>>
```

The overall progress of the command can be monitored via RVC, as shown here. Notice that RVC upgrades one disk group at a time. For each disk group upgrade, a disk is first removed from Virtual SAN cluster by evacuating all data from the disk. The format is updated and the disk is then added back to Virtual SAN with the new v2 on-disk format.

Once the upgrade is completed successfully, the following message appears:

```
<<<truncated>>>
2014-12-10 16:27:26 +0000: Cluster is still in good state, proceeding ...
2014-12-10 16:27:29 +0000: Enabled v2 filesystem as default on host cs-ie-h01.ie.local
2014-12-10 16:27:29 +0000: Removing VSAN disk group on cs-ie-h01.ie.local:
2014-12-10 16:27:29 +0000:   SSD: HP Serial Attached SCSI Disk (naa.600508b1001c61cedd42b0c3fbf55132)
2014-12-10 16:27:29 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c16be6e256767284eaf88)
2014-12-10 16:27:29 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c64816271482a56a48c3c)
2014-12-10 16:27:29 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c388c92e817e43fcd5237)
2014-12-10 16:27:29 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001ccd5d506e7ed19c40a64c)
2014-12-10 16:27:29 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c79748e8465571b6f4a46)
2014-12-10 16:27:29 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c2ee9a6446e708105054b)
2014-12-10 16:27:29 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c3ea7838c0436dbe6d7a2)
RemoveDiskMapping cs-ie-h01.ie.local: success
2014-12-10 16:52:17 +0000: Re-adding disks to VSAN on cs-ie-h01.ie.local:
2014-12-10 16:52:17 +0000:   SSD: HP Serial Attached SCSI Disk (naa.600508b1001c61cedd42b0c3fbf55132)
2014-12-10 16:52:17 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c16be6e256767284eaf88)
2014-12-10 16:52:17 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c64816271482a56a48c3c)
2014-12-10 16:52:17 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c388c92e817e43fcd5237)
2014-12-10 16:52:17 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001ccd5d506e7ed19c40a64c)
2014-12-10 16:52:17 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c79748e8465571b6f4a46)
2014-12-10 16:52:17 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c2ee9a6446e708105054b)
2014-12-10 16:52:17 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c3ea7838c0436dbe6d7a2)
AddDisks cs-ie-h01.ie.local: success
2014-12-10 16:52:58 +0000: Done upgrade host cs-ie-h01.ie.local
2014-12-10 16:52:58 +0000:
2014-12-10 16:52:58 +0000: Done with disk format upgrade phase
2014-12-10 16:52:58 +0000: There are 97 v1 objects that require upgrade
2014-12-10 16:53:04 +0000: Object upgrade progress: 97 upgraded, 0 left
2014-12-10 16:53:04 +0000: Object upgrade completed: 97 upgraded
2014-12-10 16:53:04 +0000: Done VSAN upgrade
/ie-vcsa-03.ie.local/vsan-dc>
```

## vsan.v2_ondisk_upgrade pre-checks

The following are the pre-checks that the upgrade command runs through before proceeding with an upgrade.

1. Detect disconnected hosts in a given cluster; an administrator should (a) bring disconnected host back to connected or (b) remove these hosts from current cluster.

2. Detect network partition in given Virtual SAN cluster; an administrator should resolve the network partition before proceeding with the upgrade.

3. Detect if hosts in a vCenter cluster are not participating in the Virtual SAN cluster; this might happen if an administrator was using ESXLI to manage the cluster and makes one node rejoin an other Virtual SAN cluster. Administrator will have to make the host join the correct cluster before upgrading.

4. Detect if hosts are part of the VSAN cluster, but not participating in the vCenter cluster; the administrator will have reconcile the clusters and make sure Virtual SAN cluster aligns with vCenter cluster.

5. ESXi version check, all hosts should be above 6.0; the administrator should ensure that all hosts are upgraded to version 6.0.

6. Disk auto-claim should be disabled prior to upgrading because with auto-claim enabled, we cannot remove disk groups to evacuate data; the administrator may disable this configuration through vSphere web client UI, or via the RVC command:

```
vsan.cluster_change_autoclaim -d
```

7. Detect if inaccessible objects exist in the VSAN cluster; if inaccessible swap object exists, the administrator may use the following RVC command to cleanup:

```
vsan.purge_inaccessible_vswp_objects
```

8. Detect unhealthy disks used by VSAN; the administrator may use the RVC command "`vsan.disks_stats`" to check all in-use disks, and manually fix the issue, such as replacing broken disk with a new one;

9. Detect if there are v2 object in given VSAN cluster if the administrator specifies "downgrade" mode; v2 objects should be removed if customer really needs a v1 VSAN cluster. Downgrade is for rollback, if a user encounters an issue and needs to go back to v1 on disk format.

## On-disk format post upgrade check: vsan.disks_limits

After upgrading, the number of supported component will be 9,000 per host. This can be checked with `vsan.check_limits`.

```
/ie-vcsa-03.ie.local/vsan-dc> vsan.check_limits computers/vsan/
2014-12-10 16:54:14 +0000: Querying limit stats from all hosts ...
2014-12-10 16:54:18 +0000: Fetching VSAN disk info from cs-ie-h03.ie.local (may take a moment) ...
2014-12-10 16:54:18 +0000: Fetching VSAN disk info from cs-ie-h01.ie.local (may take a moment) ...
2014-12-10 16:54:18 +0000: Fetching VSAN disk info from cs-ie-h04.ie.local (may take a moment) ...
2014-12-10 16:54:18 +0000: Fetching VSAN disk info from cs-ie-h02.ie.local (may take a moment) ...
2014-12-10 16:54:20 +0000: Done fetching VSAN disk infos
+--------------------+------------------+-----------------------------------------+
| Host               | RDT              | Disks                                   |
+--------------------+------------------+-----------------------------------------+
| cs-ie-h02.ie.local | Assocs: 156/45000 | Components: 97/9000                    |
|                    | Sockets: 26/10000 | naa.600508b1001c0cc0ba2a3866cf8e28be: 9% |
|                    | Clients: 11      | naa.600508b1001c19335174d82278dee603: 3% |
|                    | Owners: 12       | naa.600508b1001c10548f5105fc60246b4a: 3% |
|                    |                  | naa.600508b1001c07d525259e83da9541bf: 1% |
|                    |                  | naa.600508b1001c64b76c8ceb56e816a89d: 0% |
|                    |                  | naa.600508b1001ca36381622ca880f3aacd: 2% |
|                    |                  | naa.600508b1001cb2234d6ff4f7b1144f59: 3% |
| cs-ie-h03.ie.local | Assocs: 151/45000 | Components: 97/9000                    |
|                    | Sockets: 26/10000 | naa.600508b1001c9c8b5f6f0d7a2be44433: 0% |
|                    | Clients: 0       | naa.600508b1001cd259ab7ef213c87eaad7: 4% |
|                    | Owners: 17       | naa.600508b1001c1a7f310269ccd51a4e83: 17% |
|                    |                  | naa.600508b1001c9b93053e6dc3ea9bf3ef: 7% |
|                    |                  | naa.600508b1001ceefc4213ceb9b51c4be4: 2% |
|                    |                  | naa.600508b1001c2b7a3d39534ac6beb92d: 2% |
|                    |                  | naa.600508b1001cb11f3292fe743a0fd2e7: 3% |
| cs-ie-h04.ie.local | Assocs: 438/45000 | Components: 97/9000                    |
|                    | Sockets: 42/10000 | naa.600508b1001c846c000c3d9114ed71b3: 6% |
|                    | Clients: 68      | naa.600508b1001c258181f0a088f6e40dab: 6% |
|                    | Owners: 68       | naa.600508b1001cf639237aaffff152af66: 0% |
|                    |                  | naa.600508b1001cadff5d80ba7665b8f09a: 15% |
|                    |                  | naa.600508b1001c51f3a696fe0bbbcb5096: 8% |
|                    |                  | naa.600508b1001c4b820b4d80f9f8acfa95: 20% |
|                    |                  | naa.600508b1001c6a664d5d576299cec941: 4% |
| cs-ie-h01.ie.local | Assocs: 1/45000  | Components: 0/9000                     |
|                    | Sockets: 4/10000 | naa.600508b1001c16be6e256767284eaf88: 0% |
|                    | Clients: 0       | naa.600508b1001c3ea7838c0436dbe6d7a2: 0% |
|                    | Owners: 0        | naa.600508b1001c388c92e817e43fcd5237: 0% |
|                    |                  | naa.600508b1001ccd5d506e7ed19c40a64c: 0% |
|                    |                  | naa.600508b1001c61cedd42b0c3fbf55132: 0% |
|                    |                  | naa.600508b1001c64816271482a56a48c3c: 0% |
|                    |                  | naa.600508b1001c79748e8465571b6f4a46: 0% |
|                    |                  | naa.600508b1001c2ee9a6446e708105054b: 0% |
+--------------------+------------------+-----------------------------------------+
```

## On-disk format post upgrade check: vsan.disks_stats

Finally, the v2 on-disk format can be checked once again with `vsan.disks_stats`. The version of the on-disk format is shown in the Status Health column on the right hand side:

```
> vsan.disks_stats computers/vsan/
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
|                                      |                    |       | Num  | Capacity  |      |          | Status  |
| DisplayName                          | Host               | isSSD | Comp | Total     | Used | Reserved | Health  |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c61cedd42b0c3fbf55132 | cs-ie-h01.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c3ea7838c0436dbe6d7a2 | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c388c92e817e43fcd5237 | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001ccd5d506e7ed19c40a64c | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c2ee9a6446e708105054b | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c16be6e256767284eaf88 | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c79748e8465571b6f4a46 | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c64816271482a56a48c3c | cs-ie-h01.ie.local | MD    | 0    | 136.69 GB | 0 %  | 0 %      | OK (v2) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c64b76c8ceb56e816a89d | cs-ie-h02.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c0cc0ba2a3866cf8e28be | cs-ie-h02.ie.local | MD    | 17   | 136.69 GB | 10 % | 10 %     | OK (v2) |
| naa.600508b1001cb2234d6ff4f7b1144f59 | cs-ie-h02.ie.local | MD    | 15   | 136.69 GB | 4 %  | 4 %      | OK (v2) |
| naa.600508b1001c19335174d82278dee603 | cs-ie-h02.ie.local | MD    | 17   | 136.69 GB | 4 %  | 4 %      | OK (v2) |
| naa.600508b1001c10548f5105fc60246b4a | cs-ie-h02.ie.local | MD    | 17   | 136.69 GB | 3 %  | 3 %      | OK (v2) |
| naa.600508b1001ca36381622ca880f3aacd | cs-ie-h02.ie.local | MD    | 14   | 136.69 GB | 3 %  | 3 %      | OK (v2) |
| naa.600508b1001c07d525259e83da9541bf | cs-ie-h02.ie.local | MD    | 17   | 136.69 GB | 2 %  | 2 %      | OK (v2) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c9c8b5f6f0d7a2be44433 | cs-ie-h03.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001c2b7a3d39534ac6beb92d | cs-ie-h03.ie.local | MD    | 19   | 136.69 GB | 3 %  | 3 %      | OK (v2) |
| naa.600508b1001ceefc4213ceb9b51c4be4 | cs-ie-h03.ie.local | MD    | 15   | 136.69 GB | 3 %  | 3 %      | OK (v2) |
| naa.600508b1001cd259ab7ef213c87eaad7 | cs-ie-h03.ie.local | MD    | 15   | 136.69 GB | 4 %  | 4 %      | OK (v2) |
| naa.600508b1001cb11f3292fe743a0fd2e7 | cs-ie-h03.ie.local | MD    | 13   | 136.69 GB | 3 %  | 3 %      | OK (v2) |
| naa.600508b1001c9b93053e6dc3ea9bf3ef | cs-ie-h03.ie.local | MD    | 17   | 136.69 GB | 8 %  | 8 %      | OK (v2) |
| naa.600508b1001c1a7f310269ccd51a4e83 | cs-ie-h03.ie.local | MD    | 18   | 136.69 GB | 18 % | 18 %     | OK (v2) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001cf639237aaffff152af66 | cs-ie-h04.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v2) |
| naa.600508b1001cadff5d80ba7665b8f09a | cs-ie-h04.ie.local | MD    | 18   | 136.69 GB | 16 % | 11 %     | OK (v2) |
| naa.600508b1001c51f3a696fe0bbbcb5096 | cs-ie-h04.ie.local | MD    | 20   | 136.69 GB | 8 %  | 2 %      | OK (v2) |
| naa.600508b1001c4b820b4d80f9f8acfa95 | cs-ie-h04.ie.local | MD    | 14   | 136.69 GB | 21 % | 1 %      | OK (v2) |
| naa.600508b1001c6a664d5d576299cec941 | cs-ie-h04.ie.local | MD    | 13   | 136.69 GB | 5 %  | 1 %      | OK (v2) |
| naa.600508b1001c258181f0a088f6e40dab | cs-ie-h04.ie.local | MD    | 16   | 136.69 GB | 7 %  | 2 %      | OK (v2) |
| naa.600508b1001c846c000c3d9114ed71b3 | cs-ie-h04.ie.local | MD    | 16   | 136.69 GB | 6 %  | 1 %      | OK (v2) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
```

## On-disk upgrade concerns – inaccessible swap objects

During an upgrade process, if there are any inaccessible vswp (swap) objects on a disk, this may prevent a full data migration from a host in the Virtual SAN cluster. This will prevent an upgrade from v1 to v2 on-disk format.

```
> vsan.v2_ondisk_upgrade  ie-vcsa-03.ie.local/vsan-dc/computers/vsan/
+--------------------+-----------+-------------+----------------+----------------+
| Host               | State     | ESX version | v1 Disk-Groups | v2 Disk-Groups |
+--------------------+-----------+-------------+----------------+----------------+
| cs-ie-h02.ie.local | connected | 6.0.0       | 1              | 0              |
| cs-ie-h03.ie.local | connected | 6.0.0       | 1              | 0              |
| cs-ie-h04.ie.local | connected | 6.0.0       | 1              | 0              |
| cs-ie-h01.ie.local | connected | 6.0.0       | 1              | 0              |
+--------------------+-----------+-------------+----------------+----------------+

2014-12-10 14:19:00 +0000: Running precondition checks ...
2014-12-10 14:19:01 +0000: Detected inaccessible objects in VSAN. Upgrade has been
2014-12-10 14:19:01 +0000: halted. Please fix or remove them and try again. Following
2014-12-10 14:19:01 +0000: inaccessible objects were detected:
2014-12-10 14:19:01 +0000: 34723e54-7840-c72e-42a5-0010185def78\n4a743e54-f452-4435-1d15-
001f29595f9f\n3a743e54-a8c2-d13d-6d0c-001f29595f9f\n6e713e54-4819-af51-edb5-
0010185def78\n2d6d3e54-848f-3256-b7d0-001b21168828\nf0703e54-4404-c85b-0742-
001f29595f9f\n76723e54-74a3-0075-c1a9-001b21168828\ne4c33b54-1824-537c-472e-
0010185def78\nef713e54-186d-d77c-bf27-001b21168828\n77703e54-0420-3a81-dc1a-
001f29595f9f\n30af3e54-24fe-4699-f300-001b21168828\n58723e54-047e-86a0-4803-
001b21168828\n85713e54-dcbe-fea6-8205-001b21168828\nc2733e54-ac02-78ca-f0ce-
001f29595f9f\n94713e54-08e1-18d3-ffd7-001b21168828\nf0723e54-18d2-79f5-be44-
001b21168828\n3b713e54-9851-31f6-2679-001f29595f9f\nfd743e54-1863-c6fb-1845-
001f29595f9f\n94733e54-e81c-c3fe-8bfc-001b21168828
>
```

The vswp file (virtual machine swap) is used for swapping memory to disk for VMs that have memory resource issues. The ESXi host handles this. In Virtual SAN, a vswp file is stored as a separate Virtual SAN object.

Due to a known issue in vSphere 5.5, it is possible for Virtual SAN to have done incomplete deletions of vswp objects. For example, if one of the hosts that contained a component of the object was (temporarily) down at the time of deletion. In such cases, the majority of components of the vswp object were deleted/removed, but a minority of components was left behind on the hosts that were down. When the host or hosts are brought back online, the minority of the components on those hosts, which were not deleted, resurfaces. These now present themselves as an inaccessible object because a minority of components can never gain quorum. Such objects waste space and cause issues for any operations involving full data migrations, such as maintenance mode, from these hosts or disks.

This means that administrators will be unable to completely evacuate all of the components from the disks on a certain host, which will mean you will not be able to upgrade the on-disk format from v1 to v2. Fortunately, RVC has the tools available to clean up the stranded objects and complete the full data migration from the disks that will allow the on-disk upgrade to continue.

## Removing orphaned vswp objects from the Virtual SAN datastore

As mentioned, in Virtual SAN 6.0, VMware has provided an RVC tool to clean up stranded vswp objects that are now inaccessible. This tool will purge all inaccessible vswp objects from the Virtual SAN datastore and allow a full data evacuation to take place.

To ensure that the object is a vswp object and not some other object, the tool finds the active data components and reads the extended attributes that will tell us whether or not it is a vswp object.

Note that if the inaccessible object only contains witness and there are no active data components, Virtual SAN cannot read the extended attributes, so it cannot determine if the stranded object is vswp object or some other object. However, assuming that the virtual machine was deployed with a *NumberOfFailuresToTolerate=1* attribute, then there is 2 in 3 chance that the remaining component for the vswp object is an active data component and not the witness.

In the case where the remaining component of the object is indeed the witness, the command will allow a user to force delete those witness-only objects.

***Caution****: Extreme caution needs to exercise here because this command will also allow you to force delete non-vswp objects (which may cause a real data loss). If you are not completely sure that this is indeed a vswp object, please contact GSS for support with the upgrade.*

### vsan.purge_inaccessible_vswp_objects

```
> vsan.purge_inaccessible_vswp_objects -h
usage: purge_inaccessible_vswp_objects [opts] cluster_or_host
Search and delete inaccessible vswp objects on a virtual SAN cluster.
  cluster_or_host: Path to a ClusterComputeResource or HostSystem
   --force, -f:   Force to delete the inaccessible vswp objects quietly (no
interactive confirmations)
   --help, -h:   Show this message
```

If a vswp object goes inaccessible, this virtual machine will be unable to do any swapping. If the ESXi tries to swap this virtual machines pages while the vswp file is inaccessible, this may cause the virtual machines to crash. By deleting the inaccessible vswp object, things are not any worse for the virtual machine. However, it does remove all possibility of the object regaining accessibility in future time if this inaccessibility is due just a temporary issue on the cluster (e.g. due to network failure or planned maintenance). The command to purge inaccessible swap objects will not cause data loss by deleting the vswp object. The vswp object will be regenerated when the virtual machine is next powered on.

# On-disk upgrade – out of resources to complete operation

Because the upgrade procedure needs to evacuate disk groups before they can be upgraded to v2, you need to ensure that there are enough resources available in the cluster. If you do not have enough resources in the cluster, the upgrade will report the following:

```
/ie-vcsa-03.ie.local/vsan-dc/computers> vsan.v2_ondisk_upgrade 0
+--------------------+-----------+-------------+----------------+----------------+
| Host               | State     | ESX version | v1 Disk-Groups | v2 Disk-Groups |
+--------------------+-----------+-------------+----------------+----------------+
| cs-ie-h02.ie.local | connected | 6.0.0       | 1              | 0              |
| cs-ie-h03.ie.local | connected | 6.0.0       | 1              | 0              |
| cs-ie-h04.ie.local | connected | 6.0.0       | 1              | 0              |
| cs-ie-h01.ie.local | connected | 6.0.0       | 1              | 0              |
+--------------------+-----------+-------------+----------------+----------------+

2014-12-10 14:42:29 +0000: Running precondition checks ...
2014-12-10 14:42:32 +0000: Passed precondition checks
2014-12-10 14:42:32 +0000:
2014-12-10 14:42:32 +0000: Target file system version: v2
2014-12-10 14:42:32 +0000: Disk mapping decommission mode: evacuateAllData
2014-12-10 14:42:38 +0000: Cluster is still in good state, proceeding ...
2014-12-10 14:42:41 +0000: Enabled v2 filesystem as default on host cs-ie-h02.ie.local
2014-12-10 14:42:41 +0000: Removing VSAN disk group on cs-ie-h02.ie.local:
2014-12-10 14:42:41 +0000:   SSD: HP Serial Attached SCSI Disk (naa.600508b1001c64b76c8ceb56e816a89d)
2014-12-10 14:42:41 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c19335174d82278dee603)
2014-12-10 14:42:41 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001ca36381622ca880f3aacd)
2014-12-10 14:42:41 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001cb2234d6ff4f7b1144f59)
2014-12-10 14:42:41 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c0cc0ba2a3866cf8e28be)
2014-12-10 14:42:41 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c07d525259e83da9541bf)
2014-12-10 14:42:41 +0000:   HDD: HP Serial Attached SCSI Disk (naa.600508b1001c10548f5105fc60246b4a)
RemoveDiskMapping cs-ie-h02.ie.local: SystemError: A general system error occurred: Failed to evacuate
data for disk uuid 52271ef2-2c6e-c55e-2b2c-6fd76b0aa143 with error: Out of resources to complete the
operation
2014-12-10 14:42:45 +0000: Failed to remove this disk group from VSAN
2014-12-10 14:42:45 +0000: A general system error occurred: Failed to evacuate data for disk uuid
52271ef2-2c6e-c55e-2b2c-6fd76b0aa143 with error: Out of resources to complete the operation
2014-12-10 14:42:48 +0000: Failed to remove disk group from VSAN, aborting.
2014-12-10 14:42:48 +0000: Upgrade tool stopped due to errors.
/ie-vcsa-03.ie.local/vsan-dc/computers>
```

Ensure there are enough resources available in the cluster to accommodate an upgrade. Note that fault domains also play a role here. If you have configured fault domains, ensure that there are enough resources in each domain to accommodate full evacuations of disk groups from other hosts/domains.

### Upgrade path when not enough resources in the cluster

A common question is what is the recommended upgrade path for customers who have overcommitted and do not have enough free space to do a full evacuation, for example 3 node clusters?

The `vsan.v2_ondisk_upgrade` has an option called *allow-reduced-redundancy*. It should be noted that there are risks associated with this approach but unfortunately there is no other way to do the upgrade. For a portion of the upgrade, virtual machines will be running without replica copies of the data, so any failure during the upgrade can lead to virtual machine downtime.

When this option is used, the upgrade deletes and creating disk groups one at a time, on each host, and then allows the components rebuild once the on-disk format is at v2. When the operation has completed on the first host, it is repeat for the next host and so on until all hosts in the cluster are running on-disk format v2. However administrators need to be aware that their virtual machines could be running unprotected for a period during this upgrade.

# 13. Troubleshooting the VASA Provider

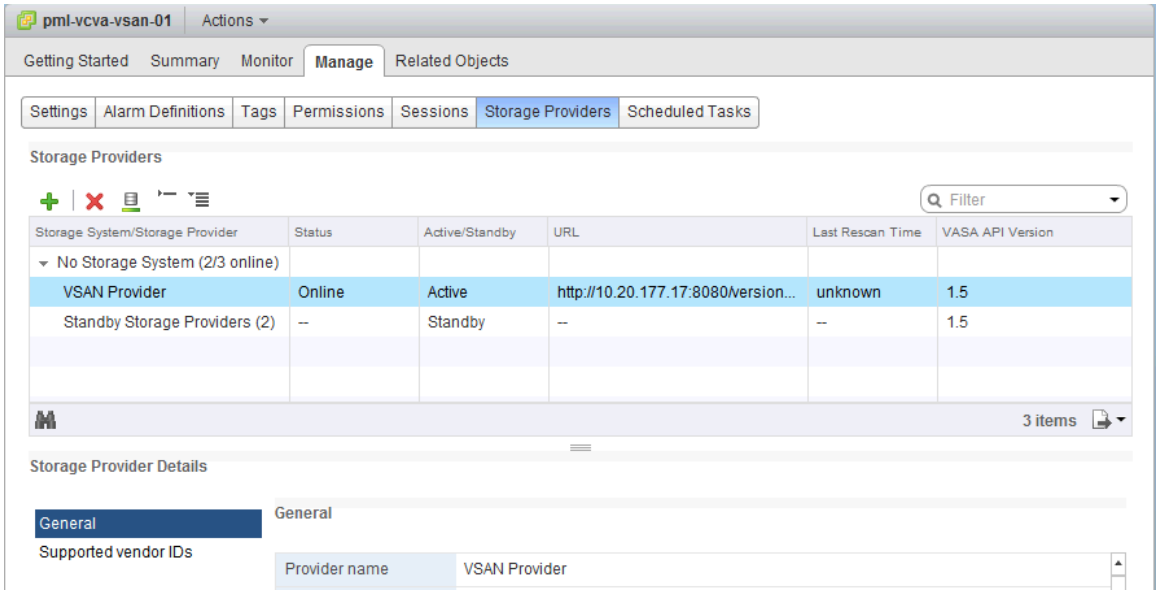
## Introduction to VASA Provider

VASA is shorthand for vSphere Storage APIs for Storage Awareness. It was initially designed to allow storage arrays to integrate with vCenter for management functionality via server-side plug-ins called a storage providers (or vendor provider). In the case of Virtual SAN, the storage provider is placed on the ESXi host.

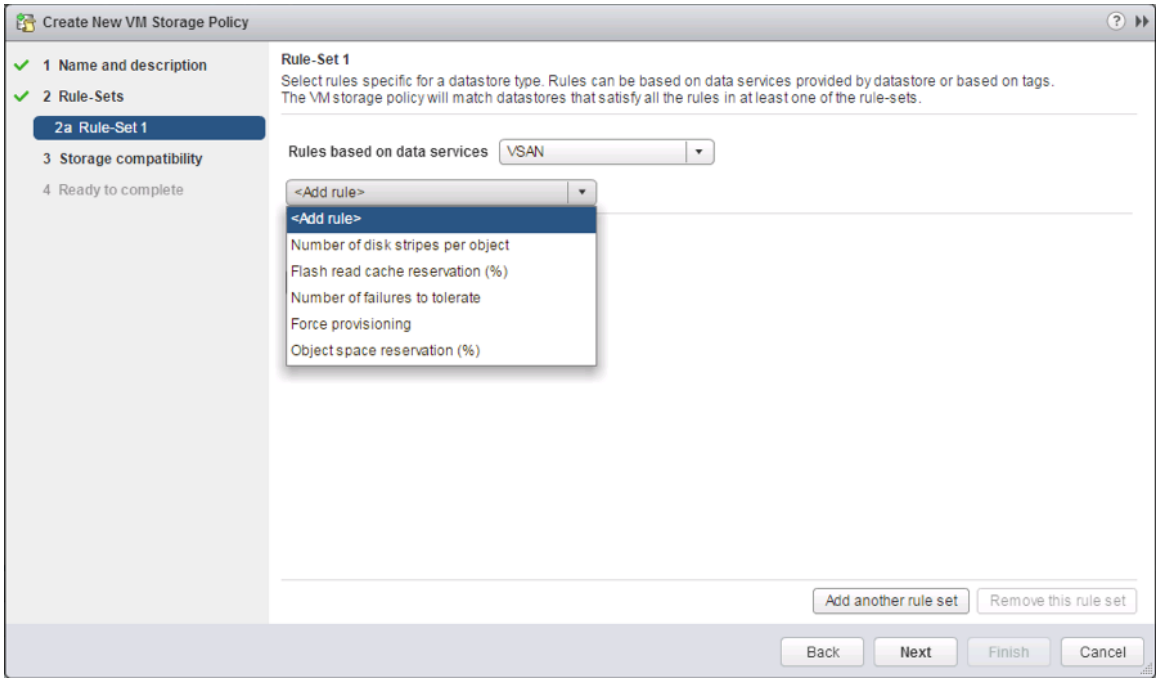
Initially the storage capabilities handled by VASA in vSphere 5.0 were very simplistic. There was a single capability associated with a LUN/datastore. In Virtual SAN, there are multiple capabilities associated with the Virtual SAN datastore. The storage provider is responsible for surfacing up a set of predefined capabilities. In the case of Virtual SAN, each ESXi host can provide this information. In Virtual SAN, a subset of capabilities around availability, provisioning and performance are surfaced up to vCenter Server. The capabilities can be viewed in various places in the vSphere web client, including on the Virtual SAN datastore view.

The storage provider, one on each ESXi host participating in the Virtual SAN Cluster, provide out-of-band information about the underlying storage system, in this case Virtual SAN. If there isn't at least one of these providers on the ESXi hosts communicating to the SMS (Storage Monitoring Service) on vCenter Server, then vCenter will not be able to display any of the capabilities of the Virtual SAN datastore, which means that administrators will be unable to build any further storage policies for virtual machine deployments However, VMs already deployed using VM Storage Policies are unaffected if the storage provider has issues.

To verify that the storage provider is online, navigate to the vCenter server, select the Manage tab, and then select the storage provider. The Status should show Online and the Active/Standby state should show one Virtual SAN Provider as Active. In the case of a host failure, a standby storage provide will assume the active state to enable the continuous surfacing up of the storage capabilities to vCenter. The following screenshot shows a Virtual SAN cluster with 3 ESXi hosts. All hosts have a storage provider, but only one is active. The other two are in standby. Should the host that currently has the online storage provider fail, another host will bring its provider online.




As mentioned, the Virtual SAN Provider is responsible for surfacing up a set of underlying storage capabilities to vCenter from Virtual SAN. There are currently 5 capabilities surfaced, as shown below.



# Analysis of VASA Provider operations

This is the sequence of steps that takes places during VASA Provider registration. It may be useful for future troubleshooting activities.

When the host is added to a Virtual SAN enabled cluster in vCenter Server, vCenter Server will trigger auto registration of Virtual SAN Providers running on the host with SMS, the Storage Management Service running on vCenter Server.

If this operation is successful, the new provider shows up in Storage Providers view in the vSphere web client.

Regardless of whether the hosts are added to vCenter using DNS name or IP address, vCenter always constructs an IP address based URL for Virtual SAN Provider.

The Resync Virtual SAN Providers workflow in SMS is invoked either when services in vCenter server are restarted or users click on the resync button on Storage Providers view in the vSphere web client.

This invokes the following steps to take place:

1. SMS finds which hosts are part of Virtual SAN enabled clusters. This SMS query returns hostnames. If the hosts were added using a DNS name, SMS query will return DNS names of these hosts. These DNS names are then used by SMS to construct Virtual SAN provider URLs.

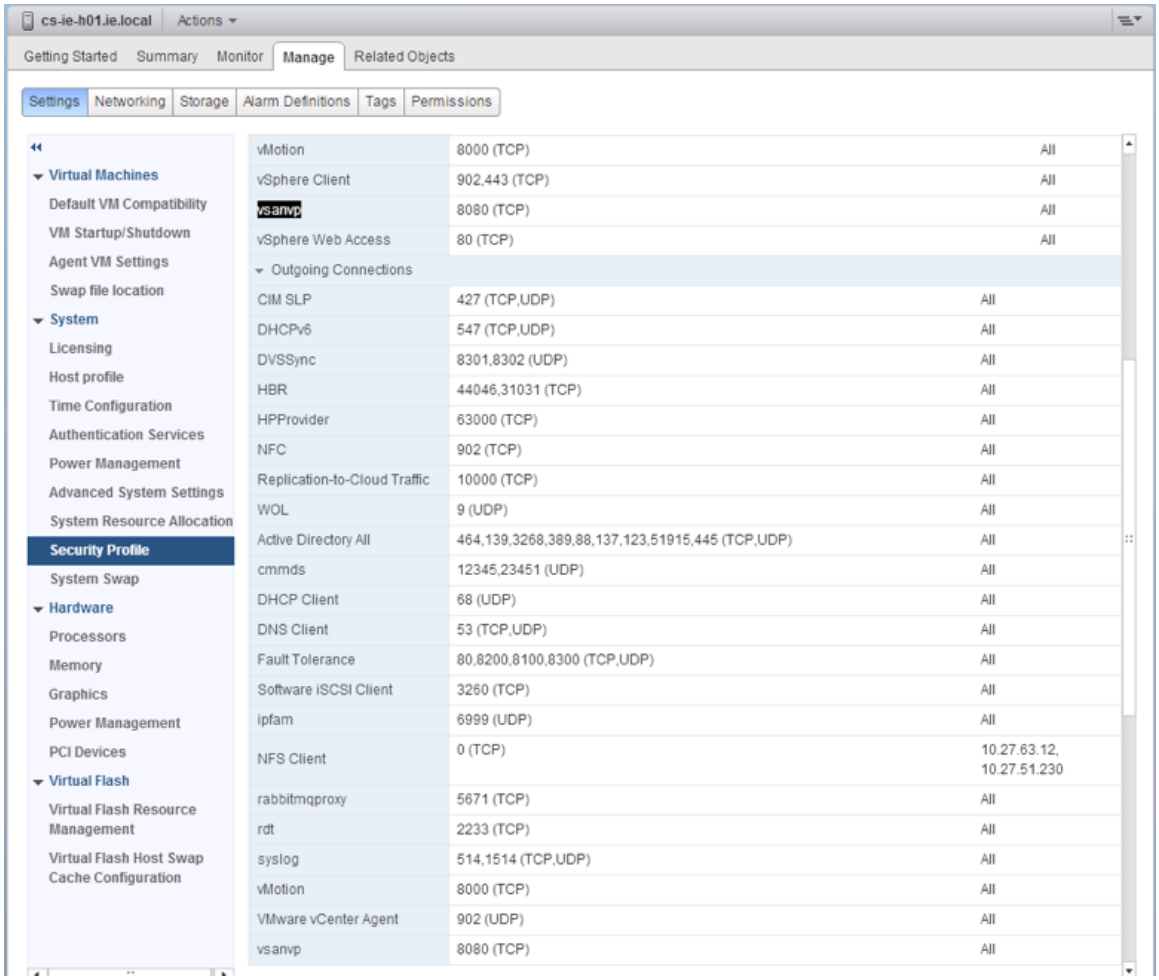
2. SMS queries already registered Virtual SAN provider. This SMS query returns IP address based URLs for Virtual SAN providers as vCenter Server auto registration always uses IP address based URLs.

3. To detect which providers need to be added or removed, SMS compares URLs obtained in step (a) and (b). This then decides which newly discovered providers, if any, needs to be registered.

# Virtual SAN Provider network port requirements

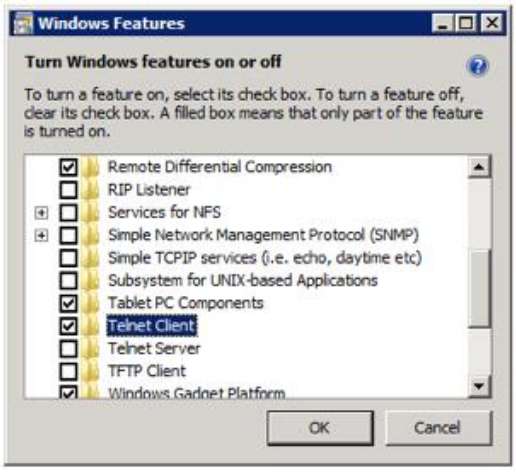
For the Virtual SAN storage providers to be visible on the vSphere client, port 8080 needs to be opened over TCP in both directions between the vCenter Server and the ESXi hosts.

If there is a firewall between the ESXi hosts and vCenter Server, it will need to accommodate this requirement. Examine the security profile of any ESXi host participating in a Virtual SAN Cluster to see the provider service. It appears as **vsanvp** in both the incoming and outgoing connections, as shown below:

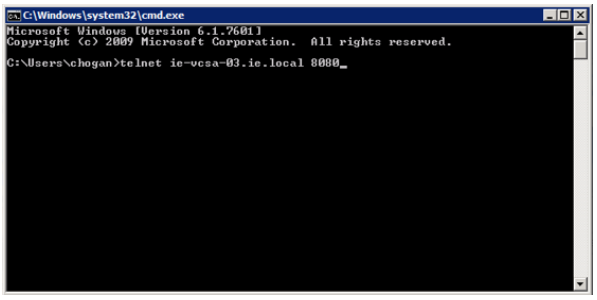


## Testing if port 8080 is open between vCenter and ESXi

The following is a quick test to verify that port 8080, used by the Virtual SAN storage provider, is open between the vCenter server and ESXi hosts. It uses the utility `telnet` that is available on most Linux and Windows distributions. However, on Windows, you may have to enable it as it is turned off by default in the Windows Features. You only need the Telnet Client as shown below:




Next step is to simply open a command window and telnet to the IP address of the ESXi host and provide the port as an argument, i.e. **telnet <hostname> 8080**




If the screen goes blank, this indicates a successful connection and the host is listening on that port for connections. If nothing happens and the connection simply sits there trying to connect, then the host is not listening on that port, or there is something sitting between the vCenter and the ESXi hosts that are blocking the connection (e.g. firewall).

This is a quick and simple test to ensure that the ports are open and vCenter can connect to them.

# Known issue with VASA Providers in version 5.5

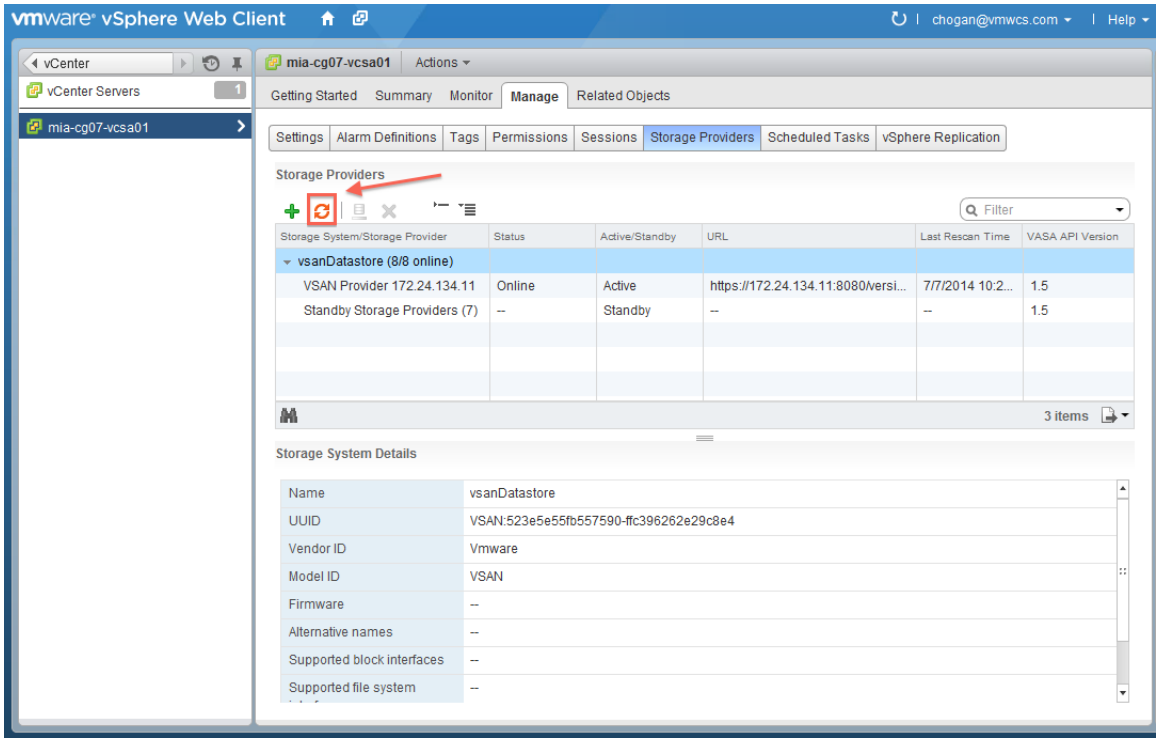
There was an issue with VASA storage providers disconnecting from vCenter Server, resulting in no Virtual SAN capabilities being visible when you try to create a VM Storage Policy. Even a resynchronization operation fails to reconnect the storage providers to vCenter. This seems to be predominantly related to vCenter servers which were upgraded to vCenter 5.5U1 and not newly installed vCenter servers.

The symptoms are described in the *vCenter 5.5 U1b release notes* as follows:

***Under certain conditions, Virtual SAN storage providers might not be created automatically after you enable Virtual SAN on a cluster***
*When you enable Virtual SAN on a cluster, Virtual SAN might fail to automatically configure and register storage providers for the hosts in the cluster, even after you perform a resynchronization operation.*

vCenter 5.5 U1b, released in June 2014, resolves the issue.

If the Virtual SAN storage providers become disconnected, with the latest releases of Virtual SAN one simply needs to do a resynchronization operation. To resynchronize, click the synchronize icon in the **Storage Providers** tab, found under vCenter Servers > vCenter > Manage, as shown below.



# 14. vCenter Server & Cluster Considerations

This chapter is designed to cover some additional considerations related to the cluster in general, as well as vCenter server.

As highlighted in the introduction, VMware recommends using the Virtual SAN Health Services to do initial triage of cluster issues. The Virtual SAN Health Services carry out a range of cluster health checks, such as verifying that all hosts that are part of the vSphere cluster are part of the VSAN cluster, and vice-versa. The Health Services directs administrators to an appropriate knowledge base article depending on the results of the health check. The knowledge base article will provide administrators with step-by-step instruction to solve the cluster problem at hand.

Please refer to the *Virtual SAN Health Services Guide* for further details on how to get the Health Services components, how to install them and how to use the feature for troubleshooting common Virtual SAN issues.

## Alarms and Events

Virtual SAN 6.0 has some built in default alarms. However Virtual SAN 5.5 does not have any such mechanism enabled by default and requires customers to create their own alarms based on VMkernel Observations (VOBs).

### Triggering Alarms based on Virtual SAN VOBs

vCenter Server 5.5 does not contain any prepopulated alarms for Virtual SAN 5.5. In Virtual SAN 6.0, only a small number of alarms are prepopulated. The VMware ESXi Observation Log (VOBD) does contain system events (termed "*observations*") observed by the VMkernel, many of which are Virtual SAN specific. By leveraging VOBs, one can quickly and easily create vCenter Alert mechanisms for Virtual SAN implementations.

### VOB IDs for Virtual SAN

This is a list of VOB IDs for Virtual SAN in version 5.5.

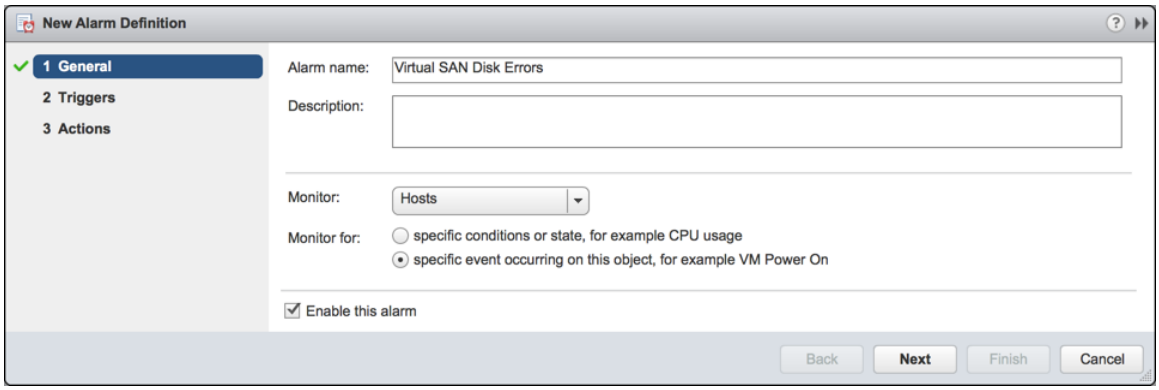
| VMkernel Observation ID | Descriptions |
|---|---|
| esx.audit.vsan.clustering.enabled | Virtual SAN clustering service had been enabled |
| esx.clear.vob.vsan.pdl.online | Virtual SAN device has come online. |
| esx.clear.vsan.clustering.enabled | Virtual SAN clustering services have now been enabled. |
| esx.clear.vsan.vsan.network.available | Virtual SAN now has at least one active network configuration. |

| | |
|---|---|
| esx.clear.vsan.vsan.vmknic.ready | A previously reported vmknic now has a valid IP. |
| esx.problem.vob.vsan.lsom.componentthreshold | Virtual SAN Node: Near node component count limit. |
| esx.problem.vob.vsan.lsom.diskerror | Virtual SAN device is under permanent error. |
| esx.problem.vob.vsan.lsom.diskgrouplimit | Failed to create a new disk group. |
| esx.problem.vob.vsan.lsom.disklimit | Failed to add disk to disk group. |
| esx.problem.vob.vsan.pdl.offline | Virtual SAN device has gone offline. |
| esx.problem.vsan.clustering.disabled | Virtual SAN clustering services have been disabled. |
| esx.problem.vsan.lsom.congestionthreshold | Virtual SAN device Memory/SSD congestion has changed. |
| esx.problem.vsan.net.not.ready | A vmknic added to Virtual SAN network configuration doesn't have valid IP. Network is not ready. |
| esx.problem.vsan.net.redundancy.lost | Virtual SAN doesn't haven any redundancy in its network configuration. |
| esx.problem.vsan.net.redundancy.reduced | Virtual SAN is operating on reduced network redundancy. |
| esx.problem.vsan.no.network.connectivity | Virtual SAN doesn't have any networking configuration for use. |
| esx.problem.vsan.vmknic.not.ready | A vmknic added to Virtual SAN network configuration doesn't have valid IP. It will not be in us |

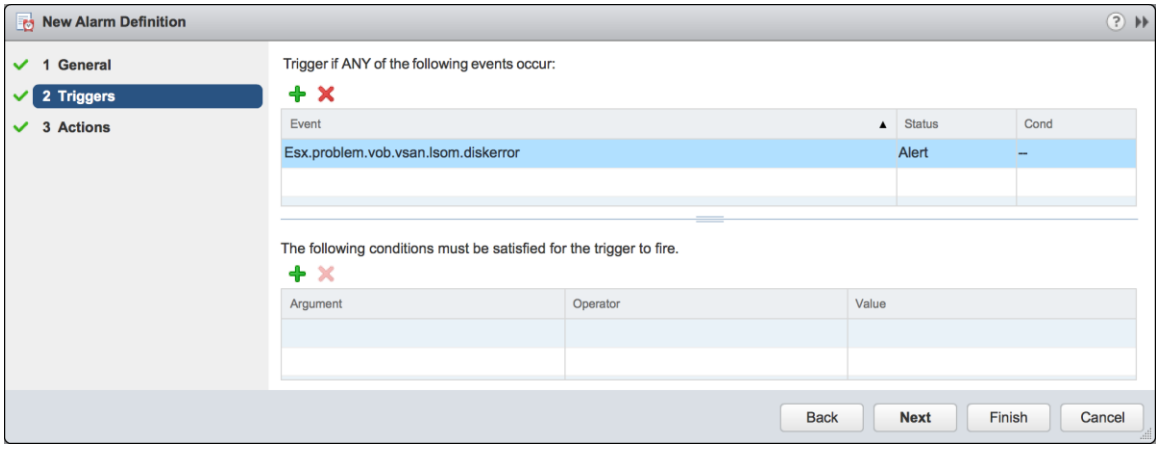
## Creating a vCenter Server Alarm for a Virtual SAN Event

To create an alarm for a Virtual SAN VOB, use the vSphere web client UI, select the vCenter Server object in the inventory, select the Manage tab, then Alarm Definitions, and then click on the (+) sign to create a new vCenter Server Alarm.

Next, provide a name for the Alarm. In this example it is called Virtual SAN Disk Error:



In this example, "Hosts" has been chosen as the object to monitor, and "specific event occurring on this object …" has also been selected. After proceeding to the Triggers window in the create alarm wizard, the particular VOB to trigger on is added.




This alarm will be triggered if the VOB "*esx.problem.vob.vsan.lsom.diskerror*" occurs. As the name suggests, this will monitor for disks errors. By referring to the table of VOBs previously, this is an observation when Virtual SAN has a permanent disk failure.

In the next window of the wizard, a particular action to take when the alarm is triggered can be selected. For example, send an email or an SNMP trap, etc.

This process can be repeated for the other VOBs as well, so as to provide notifications when other Virtual SAN event occurs.

## Maintenance mode and 3 node clusters

Putting one node of a 3-node cluster into maintenance mode with full data migration will show the selected host "Enter Maintenance Mode" stuck at 2%. This is expected behavior.

If there is a 3-node cluster, and all hosts are providing storage, this will meet the *NumberOfFailuresToTolerate*=1 policy, as the formula for the number of hosts required to provide storage is 2n+1 where "n" is the number of FTT. In order to satisfy the FTT=1 there needs to be three hosts providing storage.

If one host in 3-node cluster is placed into maintenance mode, this leaves two hosts in the cluster providing storage. Now keep in mind that components and witnesses cannot reside on the same host as the following example is examined:

Host #1 has a disk component mirror for VM1
Host #2: has a second disk component mirror for VM1
Host #3: has a witness for disk object for VM1

If host 1 is placed into maintenance mode with the option "Full Data Migration" chosen, Virtual SAN will attempt to make a new replica mirror. Unfortunately there are not enough hosts in the cluster to accommodate this request, so the task halts as described.

If there was a 4th host in the cluster also providing storage, this maintenance mode would be successful as the new mirror component could be created on host 4.

Choosing "Ensure Accessibility" would be the only option in this case, because Virtual SAN will first of all check there is an online copy of the data somewhere else within the cluster, even though the VM Disk will report as "Out of Compliance" whilst that single host is in maintenance mode, without a 4th host, this is the correct choice.

## Multiple disk groups and 3 node clusters

Disk groups are containers that contain a flash cache device and up to 7 magnetic disks in hybrid configurations. In all-flash configuration, there is also a flash cache device and up to 7 additional flash devices for capacity. However hosts can have multiple disk groups. A common question relates to the difference between having a single disk group and multiple disk groups.

The main difference is the failure domain. If the flash cache fails, it impacts the whole of the disk group and all components will need to be evacuated. If this is a 3-node cluster and a single disk group, then there is nowhere for the data in the failed disk group to be resynced/reconfigured to. If there were multiple disk groups on each host, then data in the failed disk group could be synced to the working disk group(s) on the same host, providing additional resilience.

## Support for compute only nodes

Earlier, we discussed how VMware's best practice for Virtual SAN is to have uniformly configured ESXi hosts across the cluster. Although it is possible for hosts that do not contributed storage to the Virtual SAN datastore to still deploy virtual machines to the datastore, it is not recommended by VMware. The main reasons for this are to offer a uniform performance across the cluster, as well as a balance of objects and components across all ESXi hosts. VMware's best practice is to avoid the use of compute only nodes.

# Known issue: CLOM experienced an unexpected error. Try restarting clomd

**CLOM**, the Cluster Level Object Manager, ensures that an object has a configuration that matches its policy, i.e. stripe width or failures to tolerate, to meet the requirements of the virtual machine. Each ESXi host in a Virtual SAN cluster runs an instance of ***clomd***, which is responsible for the policy compliance of the objects. However, occasionally due to a known resource issue in version 5.5, clomd might not be able to run, which results in virtual machine provisioning and cloning issues.

```
2014-08-28T05:55:50.506Z cpu4:12724417)WARNING: VSAN: VsanIoctlCtrlNode:1612: 65c4fe53-
1864-7f03-bdaf-c81f66f86313: RPC to DOM returned: Out of memory

2014-08-28T05:55:50.648Z cpu19:12724380)Vol3: 661: Unable to register file system
779ce153-82f3-08e8-d69f-c81f66f84d89 for APD timeout notifications: Inappropriate ioctl
for device

2014-08-28T05:56:13.546Z cpu6:12724417)WARNING: VSAN: VsanIoctlCtrlNode:1612: 7cc4fe53-
5c41-9bcf-4d8f-c81f66f86313: RPC to DOM returned: Out of memory

2014-08-28T05:56:13.723Z cpu23:12724411)swapobj: SwapObjDumpVmklinkResponse:1235:
{ID:0x1120; type:CREATE; ret:{status:bad0001; open:{lock:s/volumes/vsan:52c5242d4c2912fd-
9104b170781f24dd/dfe6e253-b671-9db8-4542-c81f66f84d89/ph3-f67f9dd0.vswp; dev:}}}

2014-08-28T05:56:13.723Z cpu23:12724411)swapobj: SwapObjCreate:527: Error returned from
vmklink call: Failure
```

Here is another sample vmkernel.log output, again showing when clomd is constrained by memory issues. This was an issue that impacted early 5.5 versions only:

```
2014-08-19T19:19:14Z clomd[15986764]: W110: Cannot create SSD  database: Out of memory

2014-08-19T19:19:14Z clomd[15986764]: W110: Couldn't generate base tree: Out of memory

2014-08-19T19:19:14Z clomd[15986764]: I120: CLOMVobConfigError:CLOM experienced an
unexpected error.  Try restarting clomd.

2014-08-19T19:19:14Z clomd[15986764]: CLOMGenerateNewConfig:Failed to generate a
configuration: Out of memory

2014-08-19T19:19:14Z clomd[15986764]: W110: Failed to generate configuration: Out of
memory
```

In this case, a simple restart of clomd on each host in the cluster is enough to resolve the issue at hand. To do this, simply restart the clomd services using the command:

```
/etc/init.d/clomd restart
```

The clomd module is assigned with a limited memory in Virtual SAN. The module may run out of memory on occasion, if a large number of virtual machines are deployed on  the Virtual SAN datastore.

# Handling a vCenter Server Failure

When moving hosts from one vCenter server to a new vCenter server, particularly if the original vCenter server has failed, there are 3 main considerations:

1. Order of adding hosts to new Virtual SAN cluster object on new vCenter server
2. Missing policies on new vCenter server
3. Handling distributed switch issues (if these are used in the environment)

My colleague, William Lam, in this blog post, eloquently describes the procedure. http://www.virtuallyghetto.com/2014/09/how-to-move-a-vsan-cluster-from-one-vcenter-server-to-another.html. Let's discuss these in more detail.

The first point relates to trying to add all of the ESXi hosts from the existing Virtual SAN Cluster to the new Virtual SAN Cluster at once. This method results in an error regarding UUID mismatch. To avoid this, add one host first to the cluster and once that task has been done, add the remaining ESXi hosts to the cluster. This avoids this issue.

The second point relates to the VM Storage Policies associated with the virtual machines. The VMs will continue to run with these policies, but unfortunately the new vCenter server will not know about them. There is an RVC command that can help to create these policies on the new vCenter Server. The command is `vsan.recover_spbm`. Not only will it detect VMs that are missing policy settings, but it will also provide the option to recreate policies on the new vCenter server, as shown here:

### vsan.recover_spbm

```
/localhost/vsan-dc/computers> vsan.recover_spbm 0
2014-12-02 14:54:02 +0000: Fetching Host info
2014-12-02 14:54:02 +0000: Fetching Datastore info
2014-12-02 14:54:02 +0000: Fetching VM properties
2014-12-02 14:54:02 +0000: Fetching policies used on VSAN from CMMDS
2014-12-02 14:54:03 +0000: Fetching SPBM profiles
2014-12-02 14:54:04 +0000: Fetching VM <-> SPBM profile association
2014-12-02 14:54:04 +0000: Computing which VMs do not have a SPBM Profile ...
2014-12-02 14:54:04 +0000: Fetching additional info about some VMs
2014-12-02 14:54:04 +0000: Got all info, computing after 1.92 sec
2014-12-02 14:54:04 +0000: Done computing
SPBM Profiles used by VSAN:
+-----------------------------------------+----------------------------+
| SPBM ID                                 | policy                     |
+-----------------------------------------+----------------------------+
| Existing SPBM Profile:                  | stripeWidth: 1             |
| Virtual SAN Default Storage Policy      | cacheReservation: 0        |
|                                         | proportionalCapacity: 0    |
|                                         | hostFailuresToTolerate: 1  |
|                                         | forceProvisioning: 0       |
+-----------------------------------------+----------------------------+
| Existing SPBM Profile:                  | stripeWidth: 1             |
| Virtual SAN Default Storage Policy      | cacheReservation: 0        |
```

```
 |                                          | proportionalCapacity: 0   |
 |                                          | hostFailuresToTolerate: 1 |
 |                                          | forceProvisioning: 0      |
 +------------------------------------------+---------------------------+
 | Unknown SPBM Profile. UUID:              | hostFailuresToTolerate: 1 |
 | 5810fe86-6f0f-4718-835d-ce30ff4e0975-gen0 |                           |
 +------------------------------------------+---------------------------+

 Recreate missing SPBM Profiles using following RVC commands:
 spbm.profile_create --rule VSAN.hostFailuresToTolerate=1 5810fe86-6f0f-4718-835d-
ce30ff4e0975-gen0

 Do you want to create SPBM Profiles now? [Y/N]
 Y
 Running: spbm.profile_create --rule VSAN.hostFailuresToTolerate=1 5810fe86-6f0f-4718-
835d-ce30ff4e0975-gen0

 Please rerun the command to fix up any missing VM <-> SPBM Profile associations
 /localhost/vsan-dc/computers>
```

The final issue relates to the use of a distributed switch. If there is a distributed switch in the configuration, then information about the switch is now lost since this is saved on the vCenter server. If the original vCenter Server is still available, you can export the distributed switch configuration and then import it on the new vCenter server. If it is not available, then distributed switch, distributed portgroups, and all of the mappings will need to be created and the host's networking needs to be moved to the new distributed switch.

### Preserving Storage Policies during vCenter Backups & Restores

VMware Knowledgebase Article 2072307 describes how to preserve VM Storage Policies during the back and restore of a vCenter server.

# Known issue: Migration complete but Maintenance Mode not entered

Note that the `vsan.resync_dashboard` command may not display the full information about a resynchronization activity. Therefore if a host in the Virtual SAN cluster is placed into maintenance mode, and full data migration is requested, but is still in progress when the `vsan.resync_dashboard` reports no bytes to sync, it could well mean that other objects, for example templates, are still resync'ing.

```
> vsan.resync_dashboard ie-vsan-01
2014-11-06 12:07:45 +0000: Querying all VMs on VSAN ...
2014-11-06 12:07:45 +0000: Querying all objects in the system from cs-ie-h01.ie.local ...
2014-11-06 12:07:45 +0000: Got all the info, computing table ...
+-----------+-----------------+---------------+
| VM/Object | Syncing objects | Bytes to sync |
+-----------+-----------------+---------------+
+-----------+-----------------+---------------+
| Total     | 0               | 0.00 GB       |
+-----------+-----------------+---------------+
```

The following commands will help check if this is the case:

## vsan.disks_stats

First, use `vsan.disks_stats` on the host that is being placed into maintenance mode. Check for any disks that still have a significant amount of used space and a number of components greater than 0. Note that display name of the disk. In the example below, cs-ie-03.ie.local is the host placed into maintenance mode.

```
> vsan.disks_stats 0
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
|                                      |                    |       | Num  | Capacity  |      |          | Status  |
| DisplayName                          | Host               | isSSD | Comp | Total     | Used | Reserved | Health  |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------+
| naa.600508b1001c9c8b5f6f0d7a2be44433 | cs-ie-h03.ie.local | SSD   | 0    | 186.27 GB | 0 %  | 0 %      | OK (v1) |
| naa.600508b1001cd259ab7ef213c87eaad7 | cs-ie-h03.ie.local | MD    | 1    | 136.50 GB | 16 % | 0 %      | OK (v1) |
| naa.600508b1001ceefc4213ceb9b51c4be4 | cs-ie-h03.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001cb11f3292fe743a0fd2e7 | cs-ie-h03.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001c9b93053e6dc3ea9bf3ef | cs-ie-h03.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001c2b7a3d39534ac6beb92d | cs-ie-h03.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
| naa.600508b1001c1a7f310269ccd51a4e83 | cs-ie-h03.ie.local | MD    | 0    | 136.50 GB | 1 %  | 0 %      | OK (v1) |
+--------------------------------------+--------------------+-------+------+-----------+------+----------+---------
<<<truncated>>>
```

## vsan.disk_object_info

In the output above, there is only one component left on the disk in question. All other disks are empty on that host. Next, use the `vsan.disk_object_info` to see what objects and components are still on the disk, using the display name from the previous command as an argument.

```
> vsan.disk_object_info ie-vsan-01 naa.600508b1001c79748e8465571b6f4a46

Physical disk naa.600508b1001c79748e8465571b6f4a46 (52191bcb-7ea5-95ff-78af-
2b14f72d95e4):
```

```
DOM Object: 8e802154-7ccc-2191-0b4e-001517a69c72 (owner: cs-ie-h03.ie.local, policy:
hostFailuresToTolerate = 1)
    Context: Can't attribute object to any VM, may be swap?
    Witness: 5f635b54-04f5-8c08-e5f6-0010185def78 (state: ACTIVE (5), host: cs-ie-
h03.ie.local, md: naa.600508b1001ceefc4213ceb9b51c4be4, ssd:
eui.d1ef5a5bbe864e27002471febdec3592, usage: 0.0 GB)
    Witness: 5f635b54-a4d2-8a08-9efc-0010185def78 (state: ACTIVE (5), host: cs-ie-
h02.ie.local, md: naa.600508b1001c19335174d82278dee603, ssd:
eui.c68e151fed8a4fcf0024712c7cc444fe, usage: 0.0 GB)
    RAID_1
      Component: 5f635b54-14f7-8608-a05a-0010185def78 (state: RECONFIGURING (10), host:
cs-ie-h04.ie.local, md: naa.600508b1001c4b820b4d80f9f8acfa95, ssd:
eui.a15eb52c6f4043b5002471c7886acfaa,
                                dataToSync: 1.79 GB, usage: 21.5 GB)
      Component: ad153854-c4c4-c4d8-b7e0-001f29595f9f (state: ACTIVE (5), host: cs-ie-
h03.ie.local, md: naa.600508b1001ceefc4213ceb9b51c4be4, ssd:
eui.d1ef5a5bbe864e27002471febdec3592, usage: 21.5 GB)
      Component: 8e802154-cc97-ffc1-4c85-001517a69c72 (state: ACTIVE (5), host: cs-ie-
h01.ie.local, md: **naa.600508b1001c79748e8465571b6f4a46**, ssd:
eui.48f8681115d6416c00247172ce4df168, usage: 21.5 GB)
```

At this point, we are able to see a state of RECONFIGURING against one component, and also a **dataToSync** field displaying how much data is left to sync. This will need to complete before the host can be successfully placed into maintenance mode.

### vsan.object_info

If more information about the object is need, e.g. the name, the command `vsan.object_info` can be used against the DOM Object ID from the previous command output. Checking the resulting displayed object path might help in ascertaining what object it is, for example, a template.

```
> vsan.object_info ie-vsan-01 8e802154-7ccc-2191-0b4e-001517a69c72
DOM Object: 8e802154-7ccc-2191-0b4e-001517a69c72 (owner: cs-ie-h03.ie.local, policy:
hostFailuresToTolerate = 1)
  Witness: 3f675b54-3043-9fc5-bdb9-0010185def78 (state: ACTIVE (5), host: cs-ie-
h02.ie.local, md: naa.600508b1001c19335174d82278dee603, ssd:
eui.c68e151fed8a4fcf0024712c7cc444fe, usage: 0.0 GB)
  RAID_1
    Component: 5f635b54-14f7-8608-a05a-0010185def78 (state: ACTIVE (5), host: cs-ie-
h04.ie.local, md: naa.600508b1001c4b820b4d80f9f8acfa95, ssd:
eui.a15eb52c6f4043b5002471c7886acfaa, usage: 21.5 GB)
    Component: ad153854-c4c4-c4d8-b7e0-001f29595f9f (state: ACTIVE (5), host: cs-ie-
h03.ie.local, md: naa.600508b1001ceefc4213ceb9b51c4be4, ssd:
eui.d1ef5a5bbe864e27002471febdec3592, usage: 21.5 GB)
  Extended attributes:
    Address space: 53687091200B (50.00 GB)
    Object class: vdisk
    Object path: /vmfs/volumes/vsan:52dc5a95d04bcbb9-9d90f486c2f14d1d/89802154-5c14-e025-
333b-001517a69c72/ie-ora-01-clone.vmdk
```

In this example, using the object path at the end of the output, it was determined that this was a template. By converting this template to a virtual machine temporarily, the evacuation then succeeded. Note that this is an issue that is well understood internally at VMware and something that will be addressed going forward.

Here are some other RVC commands that can help to remediate storage related issues on Virtual SAN.

### vsan.object_status_report

This command verifies the health of the Virtual SAN cluster. When all objects are in a known good state, it is expected that this command return no issues. With absent components however, this command provides details on the missing components.

The way to read the output below is that there are 96 objects in this Virtual SAN cluster that only have 3 out of 3 components currently in a healthy state. The second line can be read as the Virtual SAN cluster having 19 object where only 1 of its 3 components are currently healthy. Therefore they are listed as orphans.

```
> vsan.obj_status_report 0
 2014-11-28 12:47:04 +0000: Querying all VMs on VSAN ...
 2014-11-28 12:47:04 +0000: Querying all objects in the system from cs-ie-h02.ie.local ...
 2014-11-28 12:47:04 +0000: Querying all disks in the system from cs-ie-h02.ie.local ...
 2014-11-28 12:47:05 +0000: Querying all components in the system from cs-ie-
h02.ie.local ...
 2014-11-28 12:47:05 +0000: Querying all object versions in the system ...
 2014-11-28 12:47:06 +0000: Got all the info, computing table ...

 Histogram of component health for non-orphaned objects

 +-------------------------------------+------------------------------+
 | Num Healthy Comps / Total Num Comps | Num objects with such status |
 +-------------------------------------+------------------------------+
 | 3/3 (OK)                            |  96                          |
 +-------------------------------------+------------------------------+
 Total non-orphans: 96


 Histogram of component health for possibly orphaned objects

 +-------------------------------------+------------------------------+
 | Num Healthy Comps / Total Num Comps | Num objects with such status |
 +-------------------------------------+------------------------------+
 | 1/3 (Unavailable)                   |  19                          |
 +-------------------------------------+------------------------------+
 Total orphans: 19

 Total v1 objects: 115
 Total v2 objects: 0
```

If Virtual SAN doesn't start to remediate automatically, i.e. doesn't start to rebuild the missing components, there are a couple of possible causes. First, ensure that if the components are declared ABSENT, that 60 minutes have passed. Virtual SAN will only begin remediating ABSENT components after 60 minutes by default.

Another likely issue is that there is a resource issue. If Virtual SAN does not have enough resources to auto-remediate, consider adding additional resources to the cluster, or resolving the initial problem. Once the new resources become available, Virtual SAN will remediate automatically.

### vsan.check_state --refresh-state

One final scenario might be that all the objects are accessible, but VM is still shown as inaccessible or orphaned. This is one of those scenarios where vCenter/ESXi may be out of sync with Virtual SAN. The RVC command `vsan.check_state -r` (for refresh state) followed by the name of the cluster will synchronize the views that vCenter/ESXi has with Virtual SAN, and might be enough to resolve this disparity and commence remediation.

# 15. Getting started with VSAN Observer

In this section a web based performance-monitoring tool called VSAN Observer is introduced. The objective of this section is to give you a brief overview of the tool.

***Caution**: VSAN Observer was originally an engineering tool, and continues to be used by VMware engineering for debug purposes. Therefore there are a number of graphs and metrics that will not be useful for troubleshooting performance issues. Having said that, there are a lot of graphs and metrics that are useful for troubleshooting performance issues. The purpose of this section of the manual is to highlight which ones are useful, and what they might mean.*

## What is VSAN Observer?

The VMware Virtual SAN Observer is a monitoring and troubleshooting tool for Virtual SAN. The tool is launched from RVC and can be utilized for monitoring performance statistics for Virtual SAN live mode or offline. When running in live mode, a web browser can be pointed at vCenter Server to see live graphs related to the performance of Virtual SAN.

The utility can be used to understand Virtual SAN performance characteristics. The utility is intended to provide deeper insights of Virtual SAN performance characteristics and analytics. The VSAN Observer's user interface displays performance information of the following items:

- Statistics of the physical disk layer
- Deep dive physical disks group details
- CPU Usage Statistics
- Consumption of Virtual SAN memory pools
- Physical and In-memory object distribution across Virtual SAN clusters

The Virtual SAN Observer UI depends on some JavaScript and CSS libraries (JQuery, d3, angular, bootstrap, font-awesome) in order to successfully display the performance statistics and other information. These library files are accessed and loaded over the Internet at runtime when the Virtual SAN Observer page is rendered. The tool requires access to the libraries mentioned above in order to work correctly. This means that the vCenter Server requires access to the Internet. However with a little work beforehand, VSAN Observer can be configured to work in an environment that does not have Internet access.

# Launching VSAN Observer without internet access

In order to configure VSAN Observer to work without internet connectivity (offline mode) a number of files need to be modified. The html files which need to be modified are located in the following direcory on the vCenter Server Appliance:

```
/opt/vmware/rvc/lib/rvc/observer/
```

The files are:

- graphs.html
- history.erb.html
- login.erb.html
- stats.erb.html

This procedure will now show you how to modify the files so that VSAN Observer can work without accessing the Internet.

## JavaScript and CSS files downloads

In each of the html files (graphs.html, stats.erb.html, history.erb.html and login.erb.html) there are external references to JavaScript and CSS libraries. The CSS libraries have reference to external fonts. The objective is to download these files locally so that RVC doesn't need to pull them down from the internet.

Here is an example of the files which have internet references and will need to be downloaded locally:

```
# head -15 graphs.html.orig
 <!DOCTYPE html>
 <html>
 <head>
       <link href="observer.css" rel="stylesheet">

       <script src="https://code.jquery.com/jquery-1.9.1.min.js"></script>
       <script src="https://code.jquery.com/ui/1.9.1/jquery-ui.min.js"></script>
       <script
       src="https://ajax.googleapis.com/ajax/libs/angularjs/1.1.5/angular.min.js"></scrip
       t>
       <link href="https://netdna.bootstrapcdn.com/twitter-bootstrap/2.3.1/css/bootstrap-
       combined.no-icons.min.css" rel="stylesheet">
       <link href="https://netdna.bootstrapcdn.com/font-awesome/3.1.1/css/font-
       awesome.css" rel="stylesheet">
       <script src="https://cdnjs.cloudflare.com/ajax/libs/d3/3.4.6/d3.min.js"></script>
       <script src="graphs.js"></script>

 <style>
```

There are a number of javascript (js) files and CSS files to be downlaoded. In this example, a folder called "`externallibs`" is created under the directory `/opt/vmware/rvc/lib/rvc/observer/` and is used to place the respective JavaScript, CSS, and font files in each subdirectory.

A very useful command for downloading the files is `wget`, which is shipped with the vCenter Server Appliance. Another option is to use `cURL`. Obviously this will mean that your appliance will need Internet connectivity in order to download the files. Simply change directory to the folder where you wish the file to be downloaded, and run as follows:

```
# wget https://code.jquery.com/ui/1.9.1/jquery-ui.min.js
 --2014-12-05 10:31:57--  https://code.jquery.com/ui/1.9.1/jquery-ui.min.js
 Resolving code.jquery.com... 94.31.29.230, 94.31.29.53
 Connecting to code.jquery.com|94.31.29.230|:443... connected.
 HTTP request sent, awaiting response... 200 OK
 Length: 237108 (232K) [application/x-javascript]
 Saving to: `jquery-ui.min.js'

 100%[======================================================>] 237,108      481K/s   in 0.5s

 2014-12-05 10:31:58 (481 KB/s) - `jquery-ui.min.js' saved [237108/237108]
#
```

If it is not possible to provide, even temporarily, Internet access to the vCenter Server Appliance, then an alternative method will have to be used to download the files and later transfer them to the vCenter Server appliance that is running RVC.

Determining which fonts to download is a little bit more difficult. The fonts are found in the font CSS file. Look for the "url" keyword in the file to see which ones are need. The full list is in the structure below.

### VSAN Observer folder structure

Here is the list of files, and where they need to be placed in the new folder structure:

**Top folder -** externallibs
Sub folders – js, css, font

**Sub folder -** js
- angular.min.js
- bootstrap.min.js
- d3.v3.min.js
- jquery-1.9.1.min.js
- jquery-ui.min.js
- jquery.js

**Sub folder** – css
- bootstrap-combined.no-icons.min.css
- font-awesome.css

**Sub folder** – font
- FontAwesome.otf
- fontawesome-webfont.eot

fontawesome-webfont.svg
fontawesome-webfont.ttf
fontawesome-webfont.woff

## Downloading Fonts

The necessary fonts can be found in http://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.2.0/fonts. These should be stored in the font directory, as hightlighted earlier. Here is an example of downloading a font:

```
# wget http://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.2.0/fonts/fontawesome-
webfont.svg
 --2014-12-05 11:56:37--  http://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.2.0/fonts/fontawesome-webfont.svg
 Resolving cdnjs.cloudflare.com... 198.41.215.186, 198.41.214.183, 198.41.214.184, ...
 Connecting to cdnjs.cloudflare.com|198.41.215.186|:80... connected.
 HTTP request sent, awaiting response... 200 OK
 Length: 287526 (281K) [image/svg+xml]
 Saving to: `fontawesome-webfont.svg'

 100%[==============================================>] 287,526     --.-K/s   in 0.1s

 2014-12-05 11:56:39 (2.01 MB/s) - `fontawesome-webfont.svg' saved [287526/287526]
```

## HTML files modifications

Once all of the necessary files are downlaoded, the next step is to remove the internet references in each of the files.

***Caution**: Make backup copies of these files before proceeding.*

As you can see, a number of the files contain URLs to obtain the library files. Modify the <script> and <link> tags and replace them with the path to the library files downloaded earlier, instead of the URL. To make it easier, the changes are provided here:

```
# diff graphs.html graphs.html.orig
 6,11c6,11
 <   <script src="/externallibs/js/jquery-1.9.1.min.js"></script>
 <   <script src="/externallibs/js/jquery-ui.min.js"></script>
 <   <script src="/externallibs/js/angular.min.js"></script>
 <   <link href="/externallibs/css/bootstrap-combined.no-icons.min.css" rel="stylesheet">
 <   <link href="/externallibs/css/font-awesome.css" rel="stylesheet">
 <   <script src="/externallibs/js/d3.min.js"></script>
 ---
 >   <script src="https://code.jquery.com/jquery-1.9.1.min.js"></script>
 >   <script src="https://code.jquery.com/ui/1.9.1/jquery-ui.min.js"></script>
 >   <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.1.5/angular.min.js"></script>
 >   <link href="https://netdna.bootstrapcdn.com/twitter-bootstrap/2.3.1/css/bootstrap-
combined.no-icons.min.css" rel="stylesheet">
 >   <link href="https://netdna.bootstrapcdn.com/font-awesome/3.1.1/css/font-awesome.css"
rel="stylesheet">
 >   <script src="https://cdnjs.cloudflare.com/ajax/libs/d3/3.4.6/d3.min.js"></script>
```

```
# diff history.erb.html history.erb.html.orig
 14,20c14,20
 <   <script src="/externallibs/js/jquery.js"></script>
 <   <script src="/externallibs/js/bootstrap.min.js"></script>
 <   <script src="/externallibs/js/jquery-1.9.1.min.js"></script>
 <   <script src="/externallibs/js/jquery-ui.min.js"></script>
 <   <script src="/externallibs/js/d3.min.js"></script>
 <   <script src="/externallibs/js/angular.min.js"></script>
 <   <link href="/externallibs/css/font-awesome.css" rel="stylesheet">
 ---
 >   <script src="https://code.jquery.com/jquery.js"></script>
 >   <script src="https://netdna.bootstrapcdn.com/twitter-
bootstrap/2.3.1/js/bootstrap.min.js"></script>
 >   <script src="https://code.jquery.com/jquery-1.9.1.min.js"></script>
 >   <script src="https://code.jquery.com/ui/1.9.1/jquery-ui.min.js"></script>
 >   <script src="https://cdnjs.cloudflare.com/ajax/libs/d3/3.4.6/d3.min.js"></script>
 >   <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.1.5/angular.min.js"></script>
 >   <link href="https://netdna.bootstrapcdn.com/font-awesome/3.1.1/css/font-awesome.css"
rel="stylesheet">


# diff login.erb.html login.erb.html.orig
 14,20c14,20
 <   <script src="/externallibs/js/jquery.js"></script>
 <   <script src="/external/js/bootstrap.min.js"></script>
 <   <script src="/externallibs/js/jquery-1.9.1.min.js"></script>
 <   <script src="/externallibs/js/jquery-ui.min.js"></script>
 <   <script src="/externallibs/js/d3.min.js"></script>
 <   <script src="/externallibs/js/angular.min.js"></script>
 <   <link href="/externallibs/css/font-awesome.css" rel="stylesheet">
 ---
 >   <script src="https://code.jquery.com/jquery.js"></script>
 >   <script src="https://netdna.bootstrapcdn.com/twitter-
bootstrap/2.3.1/js/bootstrap.min.js"></script>
 >   <script src="https://code.jquery.com/jquery-1.9.1.min.js"></script>
 >   <script src="https://code.jquery.com/ui/1.9.1/jquery-ui.min.js"></script>
 >   <script src="https://cdnjs.cloudflare.com/ajax/libs/d3/3.4.6/d3.min.js"></script>
 >   <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.1.5/angular.min.js"></script>
 >   <link href="https://netdna.bootstrapcdn.com/font-awesome/3.1.1/css/font-awesome.css"
rel="stylesheet">


# diff stats.erb.html stats.erb.html.orig
 19,24c19,24
 <   <script src="/externallibs/js/jquery-1.9.1.min.js"></script>
 <   <script src="/externallibs/js/bootstrap.min.js"></script>
 <   <script src="/externallibs/js/jquery-ui.min.js"></script>
 <   <script src="/externallibs/js/d3.min.js"></script>
 <   <script src="/externallibs/js/angular.min.js"></script>
 <   <link href="/externallibs/css/font-awesome.css" rel="stylesheet">
 ---
 >   <script src="https://code.jquery.com/jquery-1.9.1.min.js"></script>
 >   <script src="https://netdna.bootstrapcdn.com/twitter-
bootstrap/2.3.1/js/bootstrap.min.js"></script>
 >   <script src="https://code.jquery.com/ui/1.9.1/jquery-ui.min.js"></script>
 >   <script src="https://cdnjs.cloudflare.com/ajax/libs/d3/3.4.6/d3.min.js"></script>
 >   <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.1.5/angular.min.js"></script>
 >   <link href="https://netdna.bootstrapcdn.com/font-awesome/3.1.1/css/font-awesome.css"
rel="stylesheet">
 ie-vcsa-06:/opt/vmware/rvc/lib/rvc/observer #
```

Once all the changes have been made, log onto the vCenter Server Appliance and launch RVC. Start Virtual SAN Observer, then access the Virtual SAN Observer web page by pointing a browser to the vCenter Server appliance address on port 8010.

Verify that everything is functioning in VSAN Observer without the need access to the Internet to download any JavaScript, CSS, or font files. If assistance is needed with launching VSAN Observer, refer to the next section of the manual.

My colleague, William Lam, wrote an automated way of setting up offline mode for version 5.5 here: http://www.virtuallyghetto.com/2014/10/automate-vsan-observer-offline-mode-configurations.html. While there are some differences in 6.0, this script is still useful if you want to automate the whole process.

## Launching VSAN Observer

VSAN Observer is launched from the RVC command line using the command `vsan.observer`. It needs a number of arguments supplied at the command line, and can be run in both live monitoring mode or offline/log gathering mode. Here is the list of options available in version 6.0:

```
                --filename, -f <s>:   Output file path
                    --port, -p <i>:   Port on which to run webserver (default: 8010)
               --run-webserver, -r:   Run a webserver to view live stats
                     --force, -o:   Apply force
  --keep-observation-in-memory, -k:   Keep observed stats in memory even when commands ends.
                                      Allows to resume later
    --generate-html-bundle, -g <s>:   Generates an HTML bundle after completion. Pass a location
                --interval, -i <i>:   Interval (in sec) in which to collect stats (default: 60)
             --max-runtime, -m <i>:   Maximum number of hours to collect stats. Caps memory usage.
                                      (Default: 2)
                 --forever, -e <s>:   Runs until stopped. Every --max-runtime intervals retires
                                      snapshot to disk. Pass a location
                    --no-https, -n:   Don't use HTTPS and don't require login. Warning: Insecure
       --max-diskspace-gb, -a <i>:   Maximum disk space (in GB) to use in forever mode. Deletes
                                      old data periodically (default: 5)
                      --help, -h:   Show this message
```

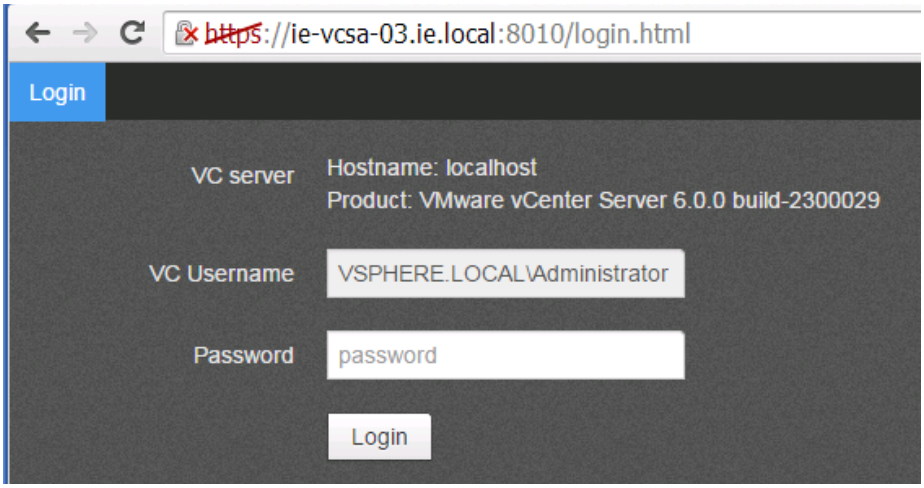
To enable live monitoring for a cluster, run the following command. Note that the cluster in this example is called Virtual SAN.

```
vsan.observer ~/computers/Virtual SAN --run-webserver --force
```

On a web browser, you can now navigate to the vCenter Server hostname or IP Address and add the port number specified in the output from the previous step. By default the port used by Virtual SAN is 8010.

```
https://vCenterServer_hostname_or_IP_Address:8010
```

This will bring you initially to the login page. Provide privileged credentials to login:




Note that in the vSphere 5.5U1 version of VSAN Observer, there was no login page, and http rather than https was used to access the tool. With the 5.5U2 release, the login page was introduced and https is used by default to access the VSAN Observer web browser.

## Launching VSAN Observer with a non-default port

If port 8010 is already in use, or you wish to use a different port for VSAN observer, this procedure will show you how. Keep in mind is that the port that is chosen instead may not be open in the firewall. When RVC is installed on vCenter Server, it automatically adds a firewall rule to open port 8010. But if you wish to use a different port, you will have to create your own firewall rule.

Here is an example of creating such a rule on the vCenter Server Appliance. Note that this does not persist through a reboot. Note that initially there is no rule for port 8011:

```
# iptables -list-rules
 <<snip>>
 -A port_filter -p tcp -m tcp -dport 22 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 902 -j ACCEPT
 -A port_filter -p udp -m udp -dport 902 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 9443 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 88 -j ACCEPT
 -A port_filter -p udp -m udp -dport 88 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 2012 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 2020 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 7444 -j ACCEPT
 -A port_filter -p udp -m udp -dport 6500 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 2014 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 514 -j ACCEPT
 -A port_filter -p udp -m udp -dport 514 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 1514 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 6501 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 6502 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 8010 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 80 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 443 -j ACCEPT
#
```

```
# iptables -A port_filter -p tcp -m tcp -dport 8011 -j ACCEPT

# iptables -list-rules
 <<snip>>
 -A port_filter -p tcp -m tcp -dport 389 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 636 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 22 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 902 -j ACCEPT
 -A port_filter -p udp -m udp -dport 902 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 9443 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 88 -j ACCEPT
 -A port_filter -p udp -m udp -dport 88 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 2012 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 2020 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 7444 -j ACCEPT
 -A port_filter -p udp -m udp -dport 6500 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 2014 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 514 -j ACCEPT
 -A port_filter -p udp -m udp -dport 514 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 1514 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 6501 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 6502 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 8010 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 80 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 443 -j ACCEPT
 -A port_filter -p tcp -m tcp -dport 8011 -j ACCEPT
#
```

Launch RVC and start vsan.observer with the new port as follows:

```
vsan.observer -o -run-webserver -port 8011
```

Note the startup message reporting the port:

```
[2014-12-04 15:42:09] INFO  WEBrick::HTTPServer#start: pid=10771 port=8011
```

Finally open a web browser to the new port as follows:

https://<vcenter-server-address>:8011

### OpenSSL::X509::CertificateError: error getting time

If an issue with VSAN Observer failing to start is experienced, and it reports an OpenSSL error 'getting time message', this is a reflection of VSAN Observer unable to retrieve the relevant vSphere certificates.

This can be addressed by running the following prior to launching VSAN Observer, which allows VSAN Observer to locate the vSphere certificates on a Windows version of vCenter server:

```
/$rvc_https_pkey = File.join(ENV['VMWARE_CFG_DIR'], 'vmware-vpx', 'ssl', 'rui.key')
/$rvc_https_cert = File.join(ENV['VMWARE_CFG_DIR'], 'vmware-vpx', 'ssl', 'rui.crt')
```

Another option is to consider the –n [--no-https] option. This will bypass https and does not prompt for a login. This method is unsecure however.

```
> vsan.observer ~/computers/Virtual SAN –run-webserver –force –no-https
```

With this latter –no-https option, open a web browser using http rather than https:

```
http://<vcenter-server-address>:8010
```

## What to look for in VSAN Observer

As we navigate through VSAN Observer, look out for statistics that have exceeded their thresholds. This is easy to observe, as these charts have a red line underneath.

Metrics that are within their boundaries are shown with a green underline in their charts.

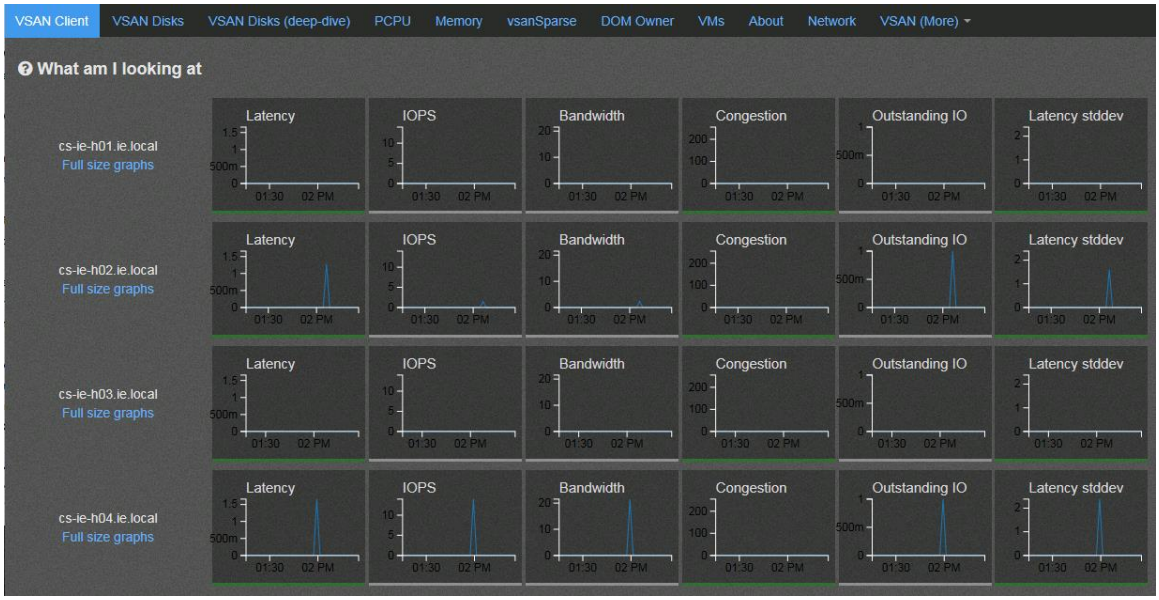
Finally, some graphs will be underscored in grey, which means that what they display is derived from other counters.

Data is refreshed every 60 seconds to give the "Average" for that 60 second period

# Navigating VSAN Observer – VSAN Client

When VSAN Observer is started, the browser is launched, and you have successfully logged in, the following is an example of the initial landing page. This first page is the **VSAN Client** page**.** Here there is a set of graphs for each host in the Virtual SAN cluster. This is the client that provides access to Virtual SAN storage. It runs on all hosts that are running VMs and performs I/O on behalf of the VMs.

The VSAN client is the top- most layer of Virtual SAN and the graphs displayed on this page provide performance statistics as observed by the Virtual Machines running in the respective hosts.




At this point, it is probably opportune to provide a brief introduction to some of the most commonly used performance counters in VSAN Observer. It is important to appreciate the relationship between these metrics, and how changes in one of these counters can impact the other.

### What is latency?

Latency gives a measure of how long it takes to complete one I/O operation from an application's viewpoint, in other words, response time measured in milliseconds. I/O sizes can vary from a few bytes to several megabytes. It therefore follows that latency can vary based on the size of the I/O. In general, both throughput and latency will increase in an almost linear fashion as the size of your I/O increases, while at the same time the total number of I/O (IOPS) will decrease. Being able to generate a very high number of IOPS is meaningless if the latency impacts the application response time in a negative manner. Smaller numbers are better.

### What is I/O per second (IOPS)?

IOPS gives a measure of number of Input/Output Operations Per Second of a storage system. An I/O operation is typically a read or a write operation and a size. I/O size can vary between a few bytes and several megabytes.

If high IOPS is observed in Virtual SAN, it does not necessarily mean that we have a problem. It could simply mean that we are using the storage to its maximum. For example, a disk clone operation may try to use all available IOPS to complete the operation in least possible time. Similarly, a low IOPS value does not mean that there is an immediate problem either; it could simply be that the I/O sizes are very large, and because of the large I/O sizes, it implies that we do fewer IOPS. Typically large numbers for IOPS is better.

### What is bandwidth?

Bandwidth (or thoroughput) measures the data rate or throughput that a storage device is capable of, or to put it another way, how much data is being transferred. IOPS and bandwidth are related. When small I/O sizes (e.g. 4KB) are involved, a disk device may hit the maximum IOPS ceiling before exhausting the available bandwidth provided by the device, or controller, or the underlying physical link. Conversely, for large I/O sizes (e.g. 1MB) the bandwidth may become a limiting factor before the maximum IOPS of a device or controller is reached. Larger numbers for bandwidth are better.

When troubleshooting storage performance, look at IOPS, I/O sizes, outstanding I/O, and latency to get a complete picture. Throughput and latency increase in an almost linear fashion as the size of your I/O increases, while at the same time the total number of I/O (IOPS) will decrease.

### What is congestion?

Congestion in Virtual SAN happens when the lower layers fail to keep up with the I/O rate of higher layers.

For example, if virtual machines are performing a lot of write operations, it could lead to filling up of write buffers on the flash cache device. These buffers have to be destaged to magnetic disks in hybrid configurations. However, this de-staging can only be done at a rate at which the magnetic disks in a hybrid configuration can handle, much slower typically than flash device performance. This can causes Virtual SAN to artificially introduce latencies in the virtual machines in order to slow down writes to the flash device so that write buffers can be freed up.

Other reasons for congestion could be related to faulty hardware, bad or misbehaving drivers/firmware or insufficient I/O controller queue depths.

Obviously, if congestion begins to occur, higher latencies will also be observed. Sustained congestion is not normal and in most cases congestion should be close to zero. Smaller values for congestion are better.

### What is outstanding I/O (OIO)?

When a virtual machine requests for certain I/O to be performed (reads or writes), these requests are sent to storage devices. Until these requests are complete they are termed outstanding I/O. This metric explains how many operations are waiting on Virtual SAN, SSD cache, I/O controller and disks. Large amounts of outstanding I/O can have an adverse effect on the device latency. Storage controllers that have a large queue depth can handle higher outstanding I/Os. For this reason, VMware only qualifies storage controllers with a significantly high queue depth, and does not support controllers with low queue depth values. Smaller numbers for OIO are better.

### What is latency standard deviation (stddev)?

The latency stddev (standard deviation) graph gives an idea of how wide the latency spread is at any given point. It is a measurement of the standard deviation of all latency times, which shows the variability of latency across all I/O requests. Lower standard deviation implies that the latency is predictable and well bounded (workload is running in a steady state). Higher values indicate that latency is fluctuating heavily and possibly warrants further investigation.

### What to look for in the VSAN Client view?

If this view shows any unexpected performance, in other words, any of the charts are underlined in red, the next step is to drill down further to understand where performance issues may be coming from.

It is important to understand that due to the distributed nature of VSAN, any host may access data from disks on any of the other hosts in the cluster. Therefore any host in the VSAN cluster may cause any performance issue seen in the VSAN Client view. The next steps will be:

- To check the 'VSAN disks' view to learn more about any bottlenecks that might be attributed to the disks (either magnetic disk or flash device/SSD).

- Check the 'VMs' centric view to see how individual disks and hosts are contributing to a virtual machines performance.

Now that we have looked at the major metrics in the VSAN client view, lets look at the full sized graphs to see what additional detail might be gleaned from VSAN Observer from a client perspective.

## VSAN Client – Full size graph

When the full size graphs link is clicked on for each client in the VSAN client page, it will display more granular information. The metrics are now split into *Reads*, *Writes* and *RecovWrites*.

The `RecovWrite` metric are specific to rebuild operations, so these metrics can tell how many writes are being used for component rebuilds. However note that this metric will be 0 from a VSAN Client perspective as clients do not have visibility into rebuild I/O. The DOM Owner graphs, seen shortly, should be checked for RecovWrite operations.

Click on any point in the chart to have the values displayed on the right hand side. Below, a very large spike can be seen, which coresponds to starting a large number of virtual machines at the same time. Because Virtual SAN uses ditributed caching, it will take a little time for the cache to warm and I/O metrics to normalize. Be aware of spikes like this when workloads are first started.

# Navigating VSAN Observer – VSAN Disks

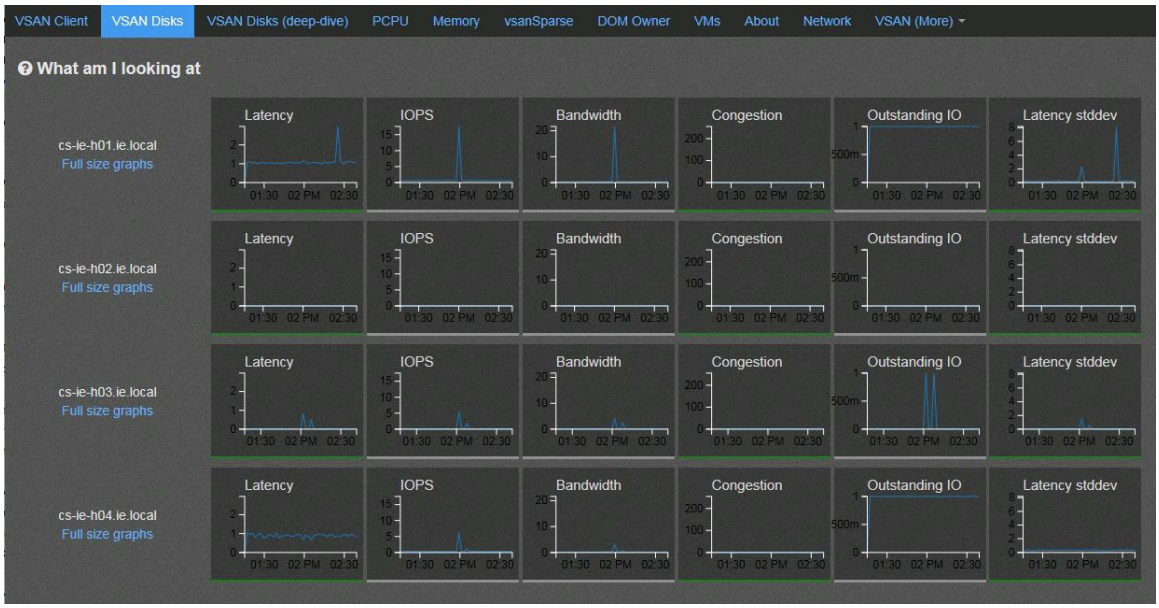
The next tab is the **VSAN Disks** view. As mentioned previously, this is where disk bottlenecks can be observed, which in turn might be impacting VM performance. All the same counters and metric observed in the VSAN client view are shown here. However, where the VSAN Client gives the client side view (nodes with VMs requesting information), VSAN Disks can be thought of as the server side view (nodes where requested information resides).

The top-most layer in Virtual SAN is the VSAN Client layer. Beneath the VSAN Client layer is the VSAN Disks layer. This layer is responsible for all of the physical disks in each host. The VSAN Disks view shows the VSAN statistics of the physical disk layer of each host. This is the layer that actually serves I/O from local disks to different nodes in the cluster.

Due to the distributed nature of VSAN the physical disks of one host may be accessed by VMs on all hosts in the VSAN cluster. Because a VM may not reside on the same node as it's storage on Virtual SAN, the Virtual SAN disks layer is shown from the node where the data resides, and it serves the data over the network to the requesting node (client).

To correlate the client view with a disk view, you would need to use the VMs view, which will be discussed shortly.

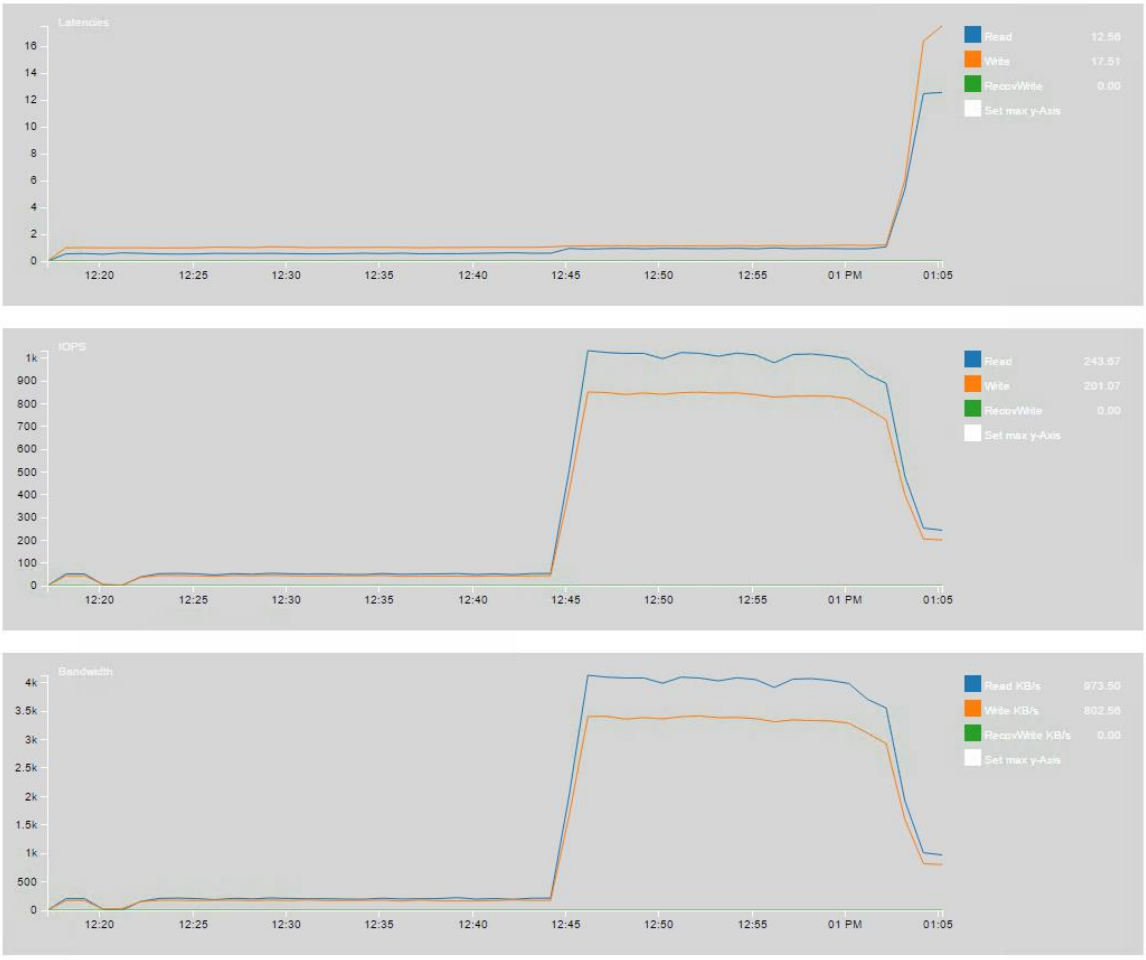
The VSAN Disks tab shows statistics related to physical disks in each host, so it includes all disk group performance from that host in a single view. This tab provides a reasonable insight into the performance of the physical disks in every host in the cluster.



This view also allows administrators to determine if there is any contention on any of the physical disks (and by extrapolation, any disk groups) that make up the overall Virtual SAN cluster. Note that the statistics shown are purely the physical disk layer (including the caching layer) but do not include any other Virtual SAN overhead (e.g. networking or Virtual SAN RAID). If virtual machine (VSAN Client) latency is an issue, but there is no latency visible in this VSAN Disk view, then the high latency may be due to network issues, which can be further investigated via the "Network" view.

## VSAN Disks – Full size graphs

Click on any point in the chart to have the values displayed on the right hand side.




If the VSAN Disks view shows physical disk contention across a majority of hosts, then this may indicates that the workload run by VMs is collectively higher than the VSAN cluster can handle. In that case, either reduce the storage workload, or check

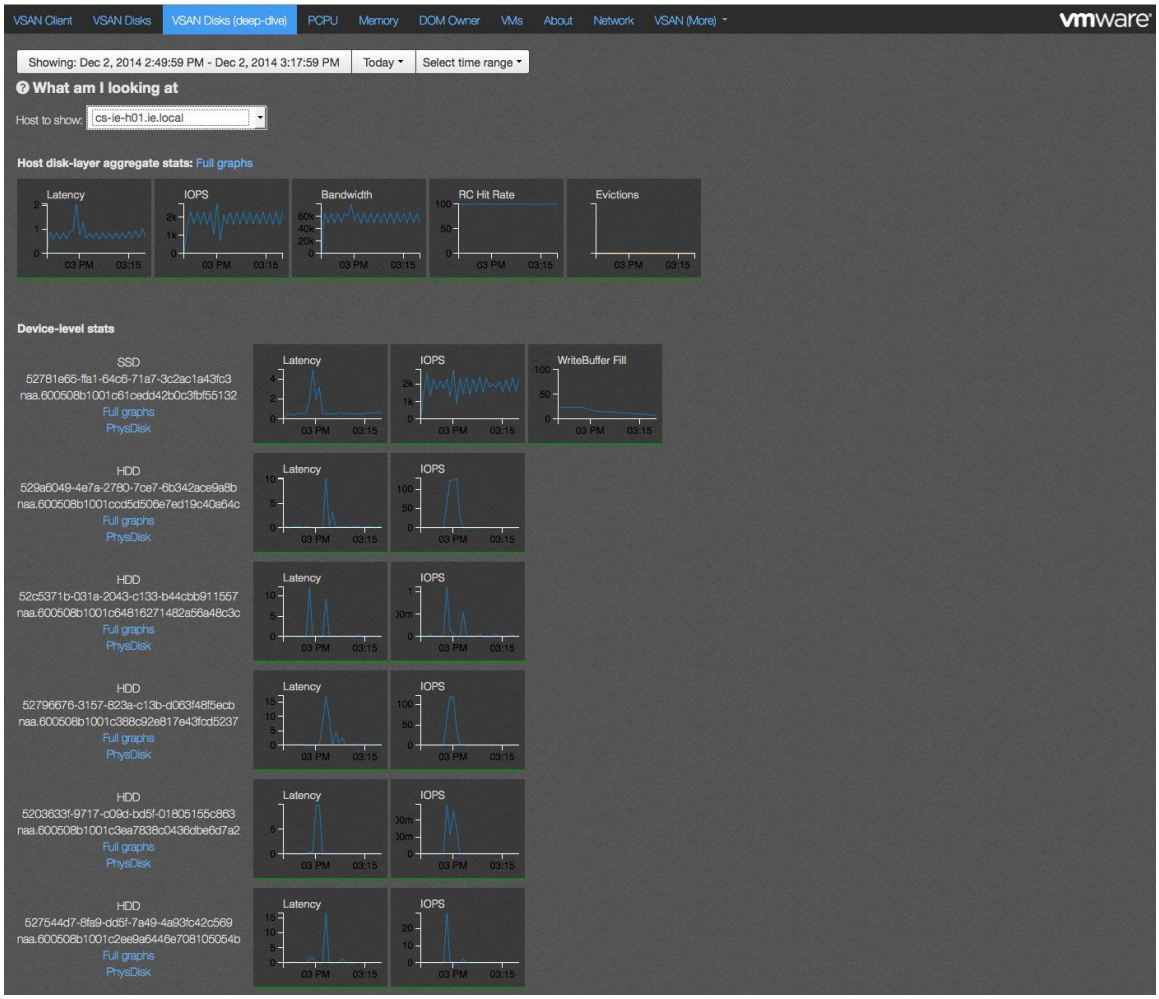
the detailed physical device view to determine if you need more magnetic disks or flash devices.

If however only a single host's physical disks are showing contention, while other hosts are operating fine, then you may have an imbalance, possibly due to a set of particularly noisy VMs.

## Navigating VSAN Observer – VSAN Disks (deep-dive)

Next to the VSAN Disks tab is **VSAN Disks (deep-dive)**. This tab provides a wealth of information broken down to each individual physical disks (both flash devices and magnetic disks) on a per host basis. This is the layer that actually does the I/O to SSD and/or magnetic disks. Using this view one can get insight into how Virtual SAN is splitting the I/O work between the flash devices/SSDs and magnetic disks.

To see the statistics, select the host that you are interested in. This view provides read cache hit rates, cache evictions and write buffer fill for SSD. Also, latency and IOPS stats are provided for all disks in each ESXi host.



This view introduces two new graphs, Evictions and WriteBuffer Fill. Both of these are references to the caching layer/SSD. Since Virtual SAN configurations use a write buffer for I/O accelerations, these graphs give insight into how the write buffer is behaving.

### WriteBuffer Fill

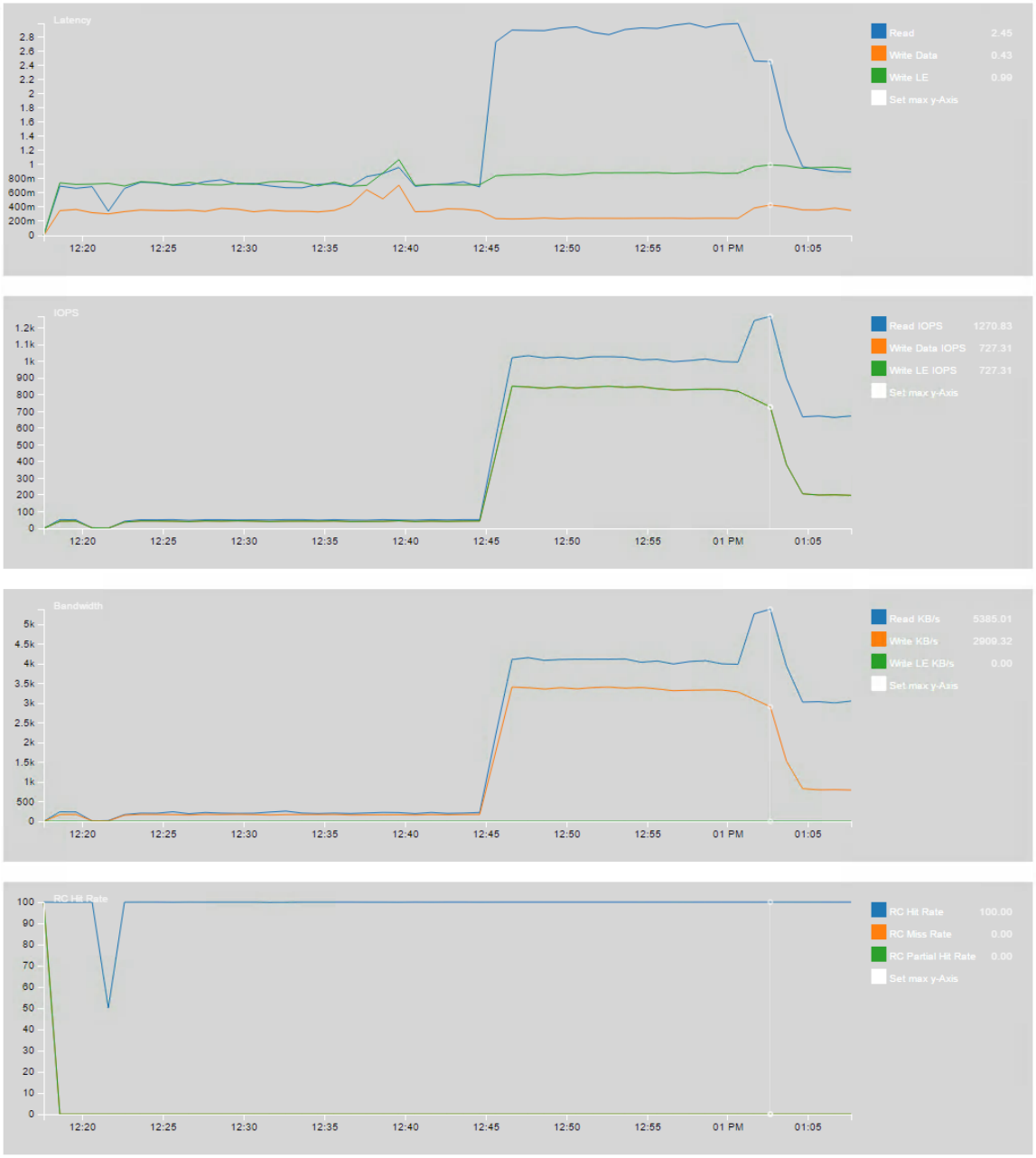
This metric tells us how much the write buffer is being consumed. One would expect that on a reasonably balanced system that a significant amount of write buffer is consumed. This is a good metric to check if you are planning to place more workloads on Virtual SAN. It can tell you whether you have enough flash capacity to cater for the additional workloads.

### Evictions

This metric tells us how often Virtual SAN has to evict data blocks from the write buffer to make room for new data blocks. In an optomized system, we would expect the working set of an application running in a virtual machine on Virtual SAN to mostly reside fully in cache. Therefore we should not expect to see too many evictions on an optomized Virtual SAN system. Excessive evictions could mean that there are workloads running that are not suitable for a caching storage system like Virtual SAN (sustained sequential write operations), or that the flash cache has been undersized for the workload requirements.

# VSAN Disks (deep-dive) – Host disk-layer aggregate stats: Full graphs

This view is essentially an LSOM view, the Local Storage Object Manager. LSOM works at the physical disk level, both flash devices and magnetic disks The first thing to notice in the Host disk-layer aggregate stats full graphs view is that there are five additional metrics shown, which are not shown in the default landing page. Here are the graphs in full:



## Latency, IOPS and Bandwidth

Latency, IOPS and bandwidth have been covered already when the VSAN Client view was discussed. The amount of data read and write, measured in KB, is displayed. One thing to note is that the IOPS of the SSD and magnetic disks will be higher than the aggregate IOPS shown. This is because of Virtual SAN consumes some IOPS for internal purposes in order to provide the caching and availability features.

You may also notice the Write LE entry in these graphs. Write LE is *Write LogEntry*. For every write I/O we actually do two I/Os to the SSD: The actual data and a Log Entry. This metric pertains to the Latency, IOPS and bandwidth of the LogEntry.

Full graphs displays additional metrics such as RC (read cache) Hit Rate, RC IOPS breakdown and evictions.


## RC Hit Rate

Read Cache (RC) is used in hybrid configurations. The RC hit rate is expected to be greater than 90% for optimal Virtual SAN performance.

A low RC hit ratio means that the size of the flash cache is not large enough to keep the 'working set' of the workload in cache. Typically that in turn leads to the magnetic disks in hybrid configurations seeing a lot of IOPS and high I/O latencies. In such cases, a cache sizing analysis should be done to determine how much cache would be sufficient for the workload. Additional flash cache devices may have to be acquired for the hosts in the cluster. Alternatively, additional and/or faster magnetic disks in each disk group may be able to absorb the higher cache miss load in hybrid Virtual SAN solutions.

If the RC hit rate is high and the magnetic disks have low IOPS and latency, but the flash cache device is seeing very high IOPS (beyond its capabilities) or high latency, then there could be a number of reasons for this behaviour. Notably, Virtual SAN may not be able to keep up with the virtual machine workload. Using additional flash devices, and using multiple disk groups, may improve the performance in this case.

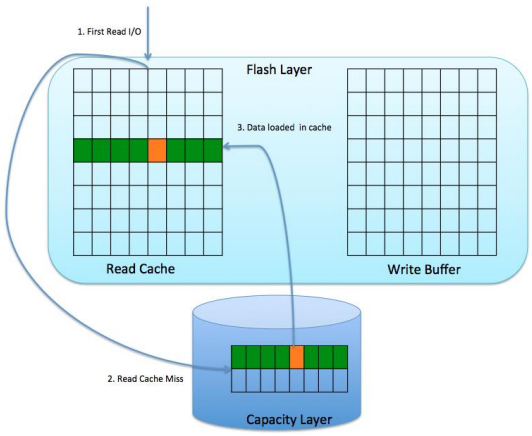
**Read cache partial hit rate**

Of interest is one of the fields displayed on the right hand side of the RC Hit Rate graph is Read Cache Partial Hit Rate.

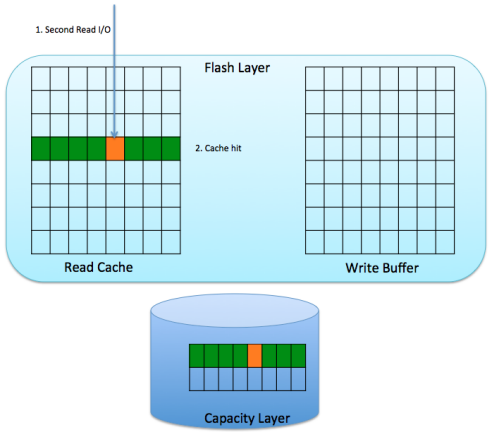
The partial hit rate graph is valuable only for Virtual SAN 5.5 because in that version there is a possibility of a read cache hit actually fetching data from the magnetic disk. This sounds unintuitive (a read cache *hit* fetching data from disk) and we have had a number of questions around partial hit rate, so let's take a moment to explain it here.

Lets walk through a simplistic workload:

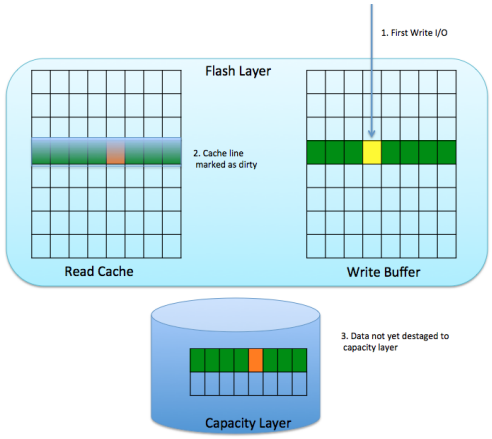
1. On the first read I/O, the data is brought into the read cache from magnetic disk. This is a miss.



2. On the second read I/O, the data is returned from the read cache. This is a cache hit.

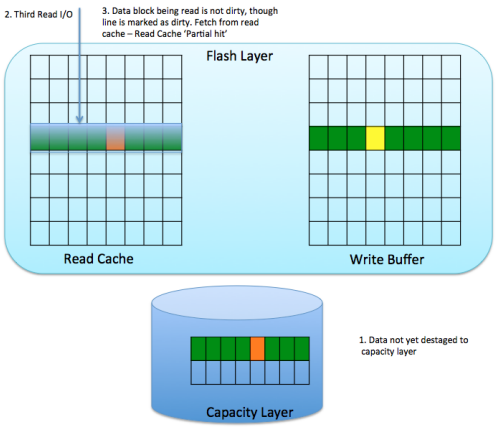



3. The third I/O is a write. This now marks that range in the read cache line as invalid/dirty. A line in cache is a range of data blocks. When one block in the range changes, the whole range (or line) is marked as dirty. At this point, the data has not yet made it from the write buffer to the magnetic disk.




4. The fourth I/O is another read, and it attempts to read a different block on the same cache line. This will be satisifed by the read cache. If the data block being read is not dirty, it can still be read from the read cache. This is considered a *Read Cache Partial Hit* as Virtual SAN does not know whether the data block it just read came from flash or magnetic disk.

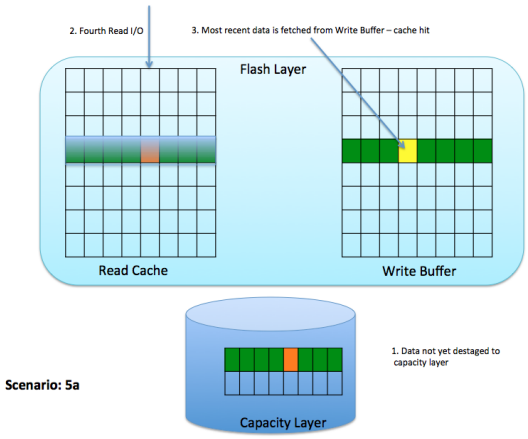
   This will become clearer when we discuss a read of the dirty/invalid block.




5. Next consider the scenario where there is an attempt to read the newly written data block. This block is no longer valid in the read cache. Now there are a number of scenarios that can occur at this point:

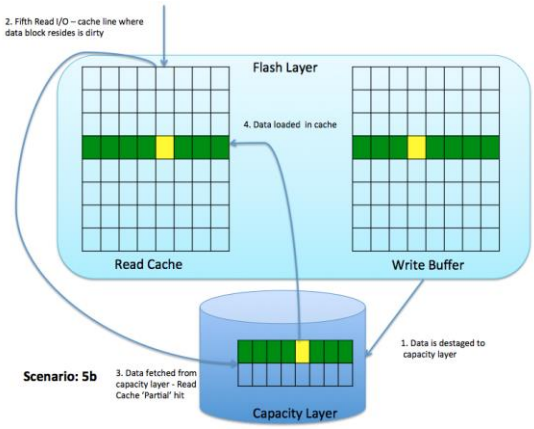
    a. Either the data has not been destaged and it is still in the write buffer
    b. Or it has been destaged to the magnetic disk.

If it is still in the write buffer, the read cache can just fetch the latest data from the flash device. This is counted as a read cache hit (not a partial hit).




Let's take the other scenario where the data has indeed destaged down to the capacity layer. The issue is that once again the read cache has no visibility into whether or not the data has been destaged.

Virtual SAN does not know if all of the data blocks in the cache line are dirty, or only some of them. If the data block being read is not dirty, it is read from flash like we saw earlier; if the data block being read is dirty/invalidated, it needs to be read from the capacity layer. However, Virtual SAN cannot tell the difference.




So potentially the data block on a invalidated line can come from cache or it can come from the magnetic disk. This is why it is called a "partial hit". In this last example, because we are attempting to read an invalid/dirty data block from read cache, the block has to be fetched from the capacity layer.

In the 6.0 version of Virtual SAN, there is new read cache code which ensures that there are no invalidated read cache lines that have their latest data on magnetic disk. This means reads from invalidated cache lines will always get their data from the flash.

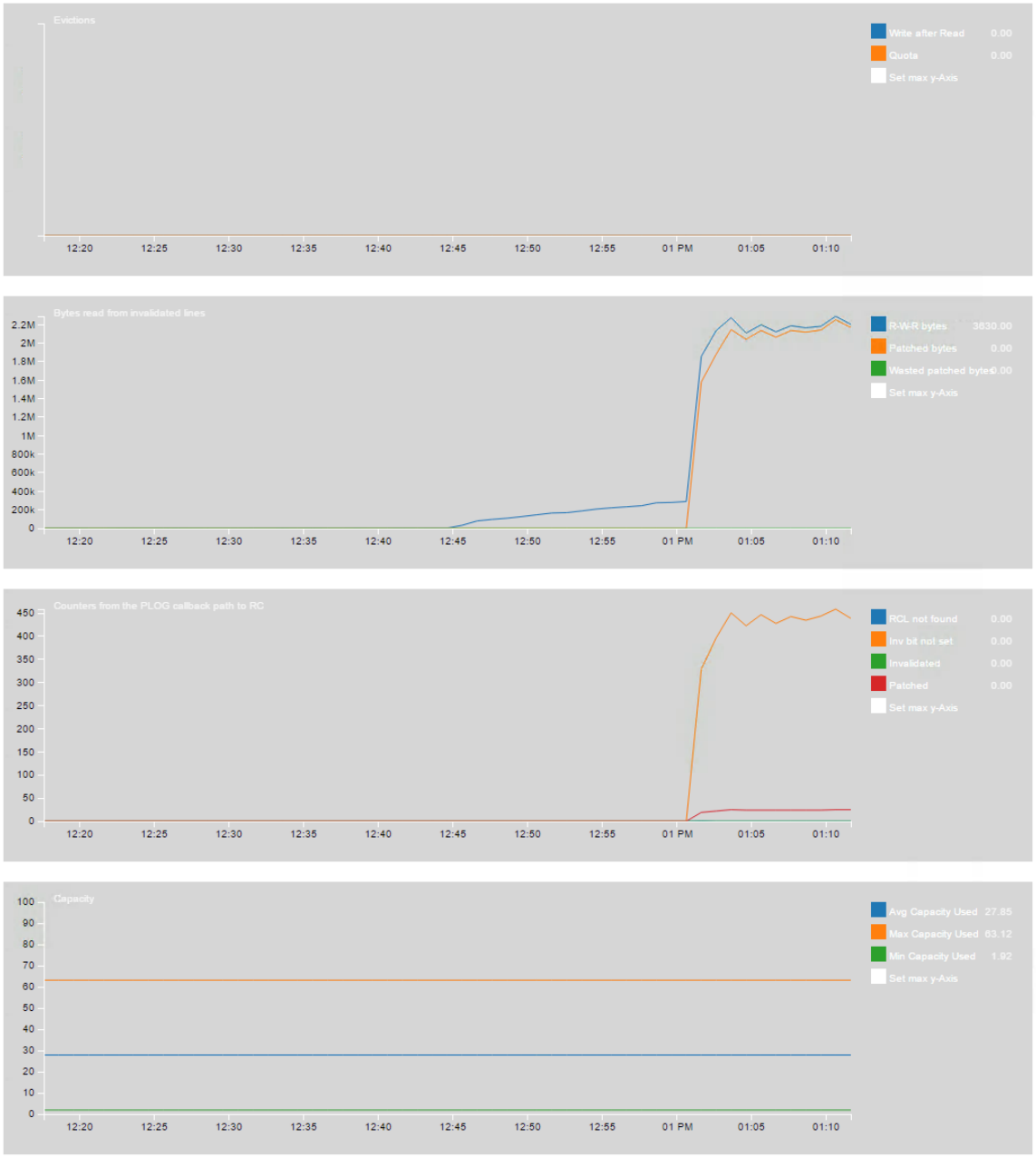
Returning to the original graphs, there are still two that need explaining.

### RC IOPS breakdown

This graphs basically details read cache I/O operations. This level of detail gives you a pretty good idea of how read cache is being utilized on this host. The metrics shown can be described as follows:

- **Total Reads** – Total number of read cache reads per second
- **RC/Mem Reads** – Number of read I/Os were served from the In-Memory ReadCache
- **RC/SSD Reads** – Number of read I/Os were served from the SSD ReadCache
- **Total RC hits** – Total number of read cache hits
- **R-a-W-a-R** – This is mostly for internal use. Normal workloads don't have this pattern, but unfortunately synthetic workload generating tools like IOMeter does. This metric is to help detect such workloads. Workloads that exhibit this behavior do not really benefiting from cache, because it is doing writes after reads, invalidating cache lines.

This brings us to the final metrics displayed in the full graphs view of host aggregrate stats.




## Evictions

The Evictions graph has two metrics, **Write after Read** and **Quota**. These both relate to invalidating cache lines. Only once there are enough data blocks dirty would a whole cache line get invalidated. Quota relates to the number of cache lines in flash.

### Bytes Read from invalidated cache lines/PLOG callback path to RC

The graphs displaying counters from **Bytes read from invalidated cache lines** and **PLOG callback path to RC** are only meant for VMware's engineering team and do not provide meaningful customer facing information that can be utilized for performance troubleshooting. They are shown here simply for completeness.

### Capacity

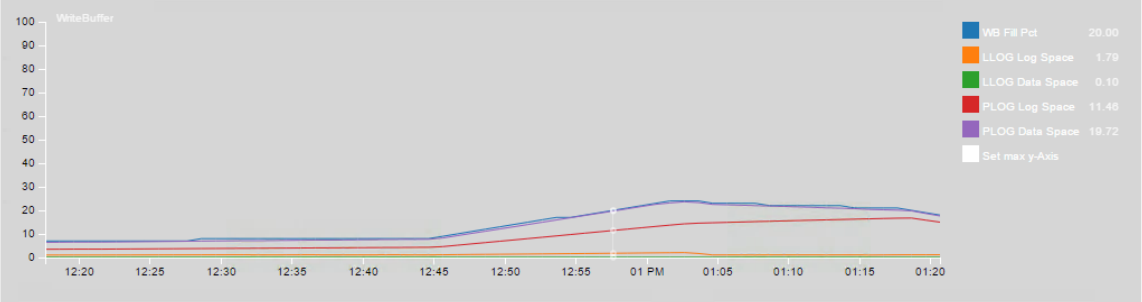
This graph's metrics are pretty self explanatory, but the give the administrator the physical or raw view of capacity, and not the vCenter view. This is always useful for reference.

# VSAN Disks (deep-dive) – Device-level stats: full graphs

As well as the Host disk-layer aggregate stats, there are device-level stats: full graph views. Let's see that this can tell us next.

### WriteBuffer

If the device is a flash device used for caching, then this chart related to write buffer is displayed. Flash devices are partitioned into a WriteBuffer (WB) and ReadCache (RC) for Virtual SAN hybrid configurations. By default, the split is 30% WB and 70% RC.




- **WB Fill Pct** – signifies the percentage on the write buffer that is currently filled.

### A note on LLOG and PLOG

The logical LOG (LLOG) and the physical LOG (PLOG) share the write buffer. The internal behavior of LLOG and PLOG are not necessarily useful for troubleshooting, but for completeness a description of their behavior and how they interoperate in include here.

When a data block arrives in the write buffer, a corresponding entry for it is kept in LLOG, the logical log. This is used by LSOM for log recovery on reboot.

However, once the data block is in the write buffer, Virtual SAN needs to calculate where to place this block of data on the magnetic disk when destaging in hybrid configurations. To do this, it consults the filesystem on the magnetic disk. This placement process could cause the filesystem to generate its own metadata updates (e.g. logical block address to physical location mapping, etc). This I/O is intercepted and buffered on the SSD and a record in kept PLOG. Once we get the physical locations for the data blocks from the filesystem, Virtual SAN stores this in PLOG. At this point, we no longer need to keep the LLOG entry.

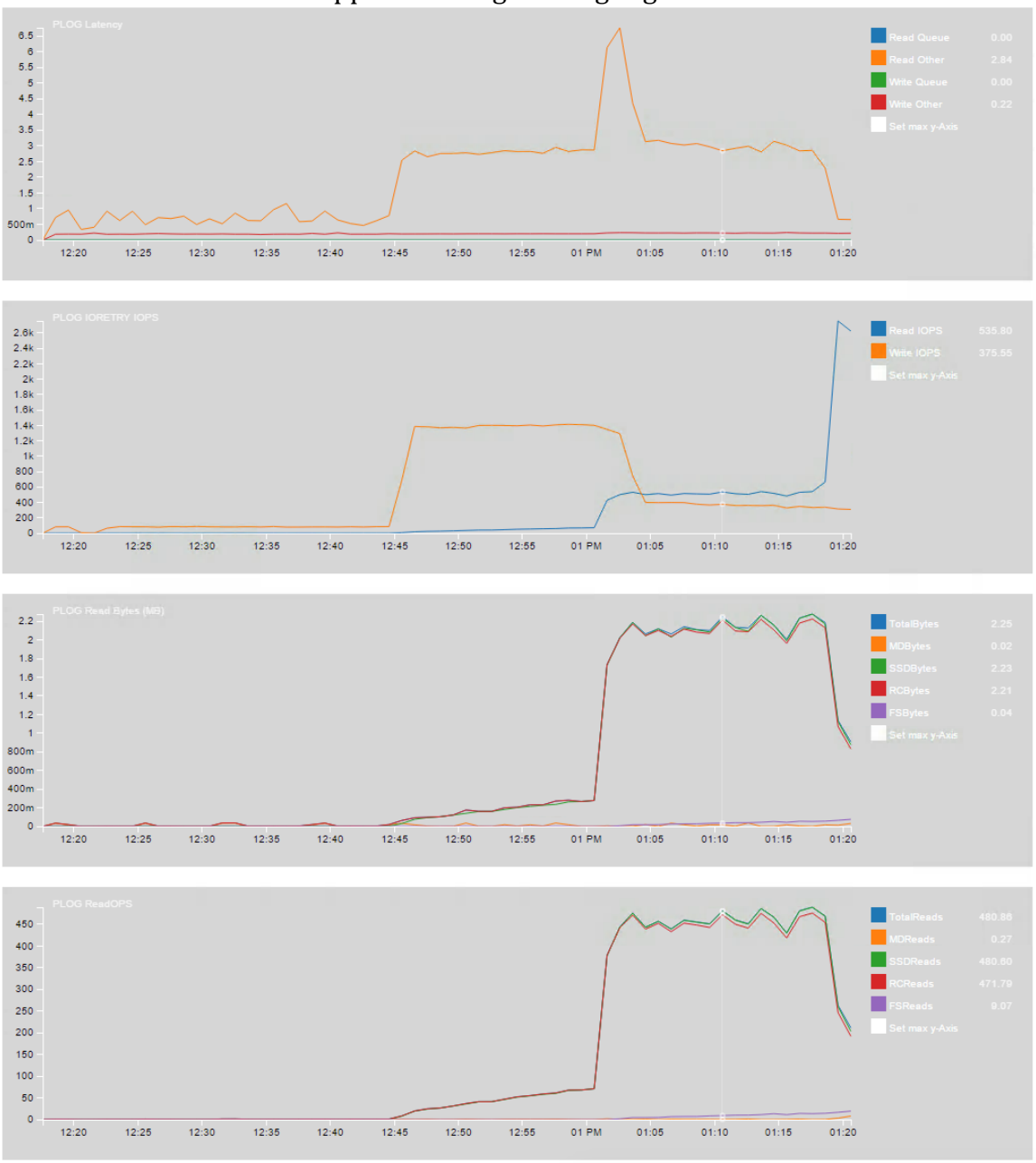
Later the elevator flushes all log entries (metadata and user data) from PLOG and destage both user and internal filesystem data blocks from SSD to magnetic disk.

- **LLOG Log & Data Space** – normally should be only using a fraction of write buffer space

- **PLOG Log & Data Space** – should maximize write buffer space. Destaging starts when WB is 30% full.

The remaining graphs require a deeper understanding on PLOG and LLOG internal operations, including elevator algorithm explanations, which are beyond the scope of this troubleshooting manual. Suffice to say that there is various metrics displaying information about the PLOG read queue latency and PLOG write queue latency. These will highlight if there is any significant latency incurred with flushing cache to disk. There are also charts displaying PLOG read retry IOPS and PLOG write retry IOPS and if there is any significant I/O retries.

The full graphs in this view go deep into Virtual SAN internals, and are typically only of use to VMware Global Support and engineering organizations.



Similarly, detailed information about the LLOG internals would be needed to understand the metrics displayed in the LLOG charts, shown here.

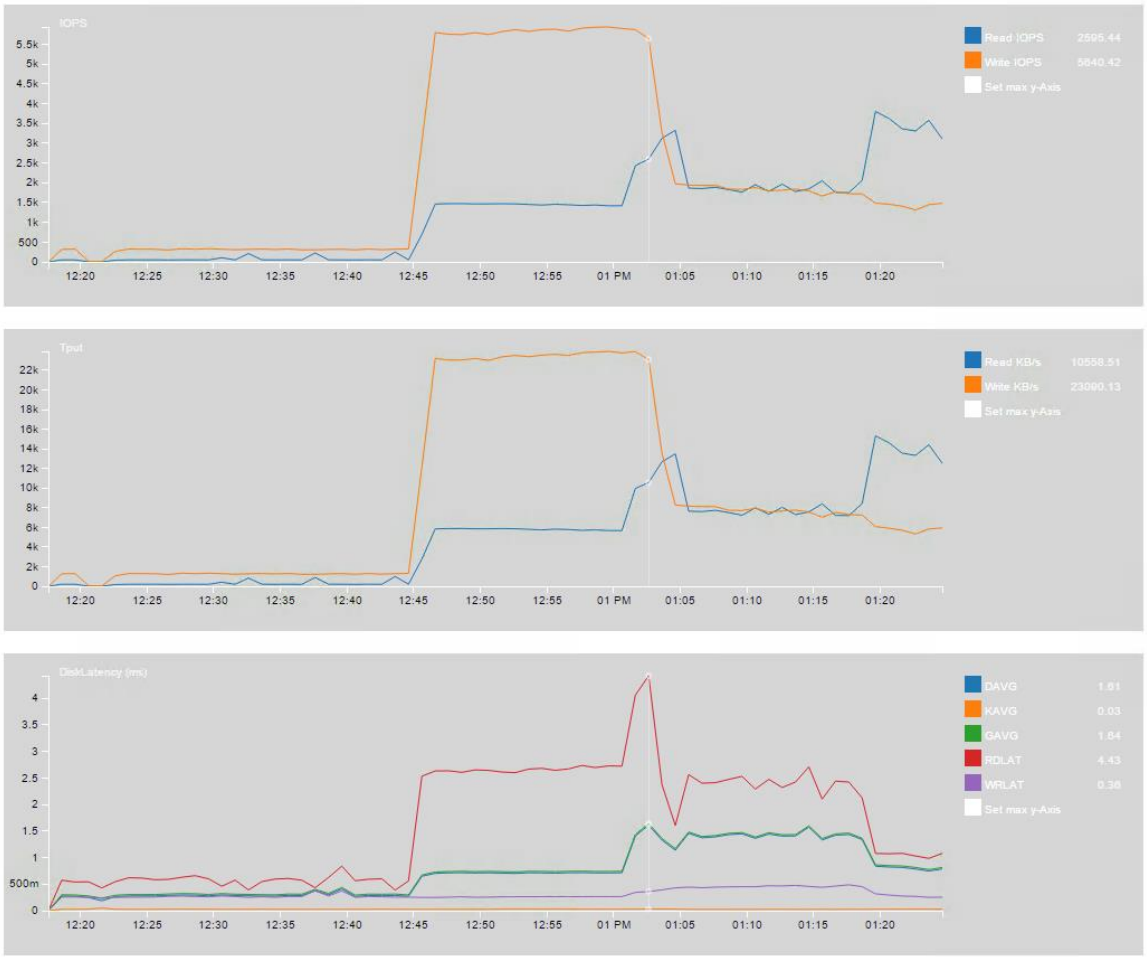

If the device is an SSD, and these full graphs of device level stats are chosen, then additional charts related to LLOG (Logical LOG) are displayed. LLOG is a property of SSD, and will not be visible in a physical view of a magnetic disk.

One important point to note about the Logical LOG is that upon every ESXi host boot, LSOM needs to perform an SSD log recovery. This is why rebooting ESXi hosts can take significantly longer with VSAN enabled. The entire log needs to be read in order to make sure in-memory state is up to date.

Once again, the full graphs in this view go deep into Virtual SAN internals, and are typically only of use to VMware Global Support and engineering organizations.

# VSAN Disks (deep-dive) – Device level stats: PhysDisk

One final additional set of graphs available in the VSAN Disks (deep-dive) views is the PhysDisk charts.




This view is giving physical disk statistics, and you might have guessed from the name of the graphs. **IOPS** and **Tput** (throughput) should be well understood at this stage. Both reads and writes are shown separately.

### DiskLatency (ms)

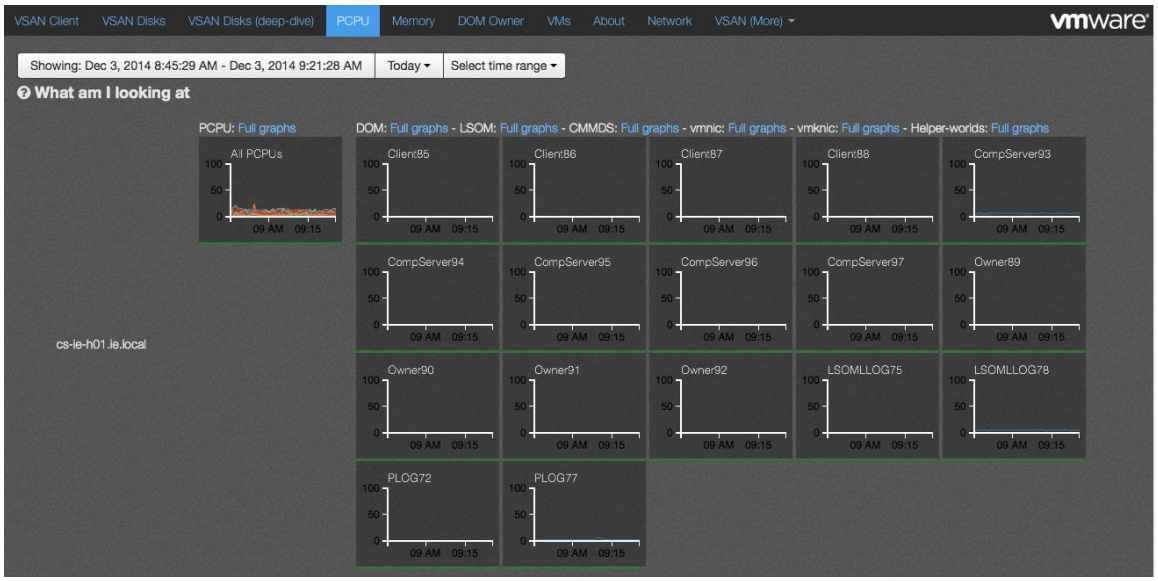
- **DAVG** – This is the average response time (latency) in milliseconds per command being sent to the device
- **KAVG** – This is the amount of time the command spends in the VMkernel.
- **GAVG** – This is the response time, as it is perceived by the guest operating system. This is calculated with the formula: `DAVG + KAVG = GAVG`
- **RDLAT** – Read Latency
- **WRLAT** – Write Latency

# Navigating VSAN Observer – PCPU

The **PCPU** tab shows overall CPU usage and also per component CPU usage statistics of Virtual SAN. This view shows CPU usage from an overall host perspective, as well as specifically from the view of Virtual SAN wordlets and networking worldlets.

A worldlet can be considered a process that does some work on behalf of Virtual SAN. A single worldlet can at most occupy a single physical CPU core. The 100% mark in the graphs refers to a percentage of that limit.

If the utilization of a Virtual SAN worldlet is getting close to 100% then CPU is likely a bottleneck for Virtual SAN performance. However, even when the utilization is not yet maxed out, a significant utilization (>10%) of readyTime indicates CPU contention between Virtual SAN and other uses of physical CPUs on the host (e.g. running VMs). Looking at overall CPU consumption of the host should confirm this picture.




Some of the graphs visible on this view are:

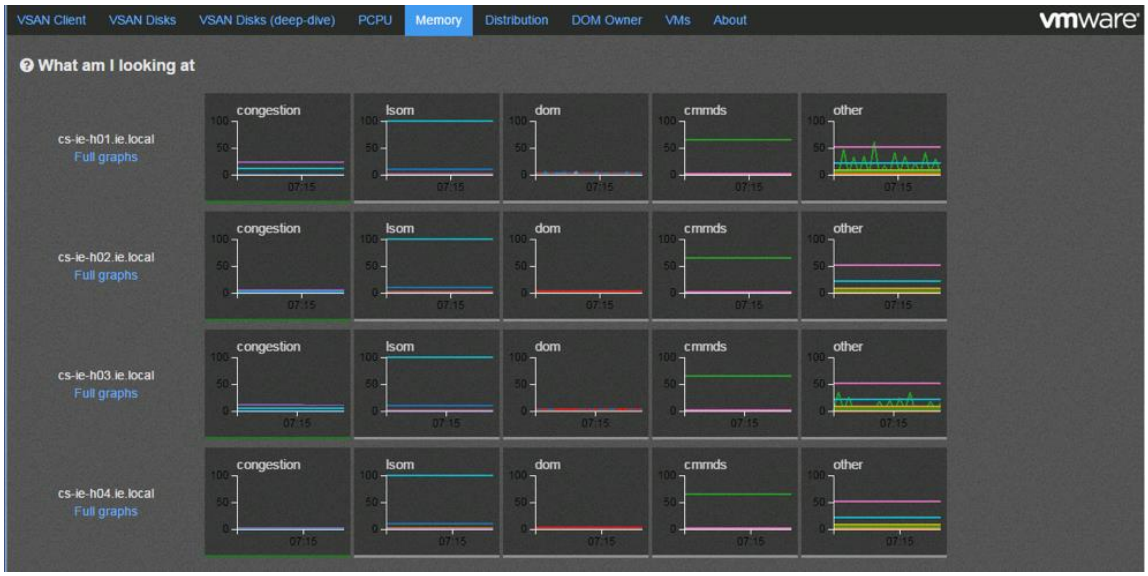
- **Client** – amount of CPU consumed by client worldlets; client is the component which accesses an object on behalf of a VM.
- **CompServer** – amount of CPU consumed by component server worldlets. This provides the network front-end for accessing an object.
- **Owner** – amount of CPU consumed by owner worldlets. The owner is the storage head and co-ordinates that can do I/O to an object. Owner communicates to the component manager to access a leaf on the object's RAID tree.
- **LSOMLOG** – amount of CPU consumed by LSOMLOG worldlets.
- **PLOG** – amount of CPU consumed by PLOG worldlets

This PCPU chart should provide enough detail for monitoring CPU usage. The full graphs in this view go deep into Virtual SAN internals, and are only of use to VMware Global Support and engineering organizations.

## Navigating VSAN Observer – Memory

The **Memory** tab shows memory consumption of various Virtual SAN components.

***Caution**: This view is primarily meant for VMware Support. Users may or may not be able to spot problems in the shown values. But VMware Support can use these views of memory pools to determine if a problem exists.*




This view shows consumption of various VSAN memory pools. The pools tracked under congestion directly impact I/O performance if they are above ~75% utilization. A high utilization triggers Virtual SAN's congestion mechanism, which imposes I/O delays (additional latencies) at the VSAN client. Congestion values can be seen in the various I/O views in VSAN Observer.

# Memory – Full size graphs

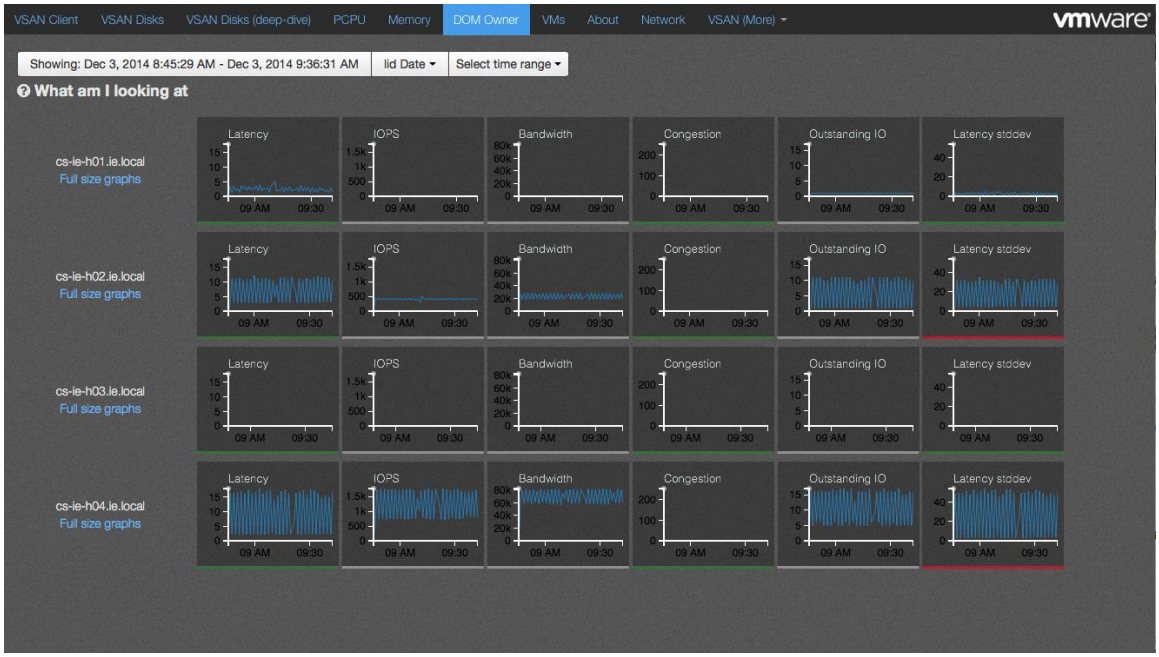
By clicking on the full graphs in the memory view, detailed memory consumption details from each host are displayed. It is broken out into separate sections, as per the overview screen seen earlier, but it should be easy to observe the main consumers of memory.


Describing each and every field is beyond the scope of this document. Suffice to say that if Virtual SAN is suffering due to memory resource shortage, these views can identify the root cause. Once again, this full graphs view go deep into Virtual SAN internals, and are only of use to VMware Global Support and engineering organizations.

# Navigating VSAN Observer – DOM Owner

This now brings us to **DOM Owner**. The DOM owner is a VSAN internal layer. DOM, short for Distributed Object Manager, is the layer which implements RAID configurations of objects (RAID-0 stripes, RAID-1 mirror) and is responsible for originating resync and recovery operations. For every VSAN object VSAN, one host is elected to be the 'owner'. The owner performance all RAID functionality and ensures correctness, even under concurrent access by multiple clients.

The metrics shown here are observed from the DOM Owner perspective.




The metrics shown have been seen a number of times already but this time we are seeing them from the DOM owner perspective.

Of interest in the VSAN Observer graph above is that on two hosts, the Latency standard deviation is underscored in red, meaning that it outside the norm. However Latency itself is still withing acceptable limits on both hosts, as can be seen by the green underscore.

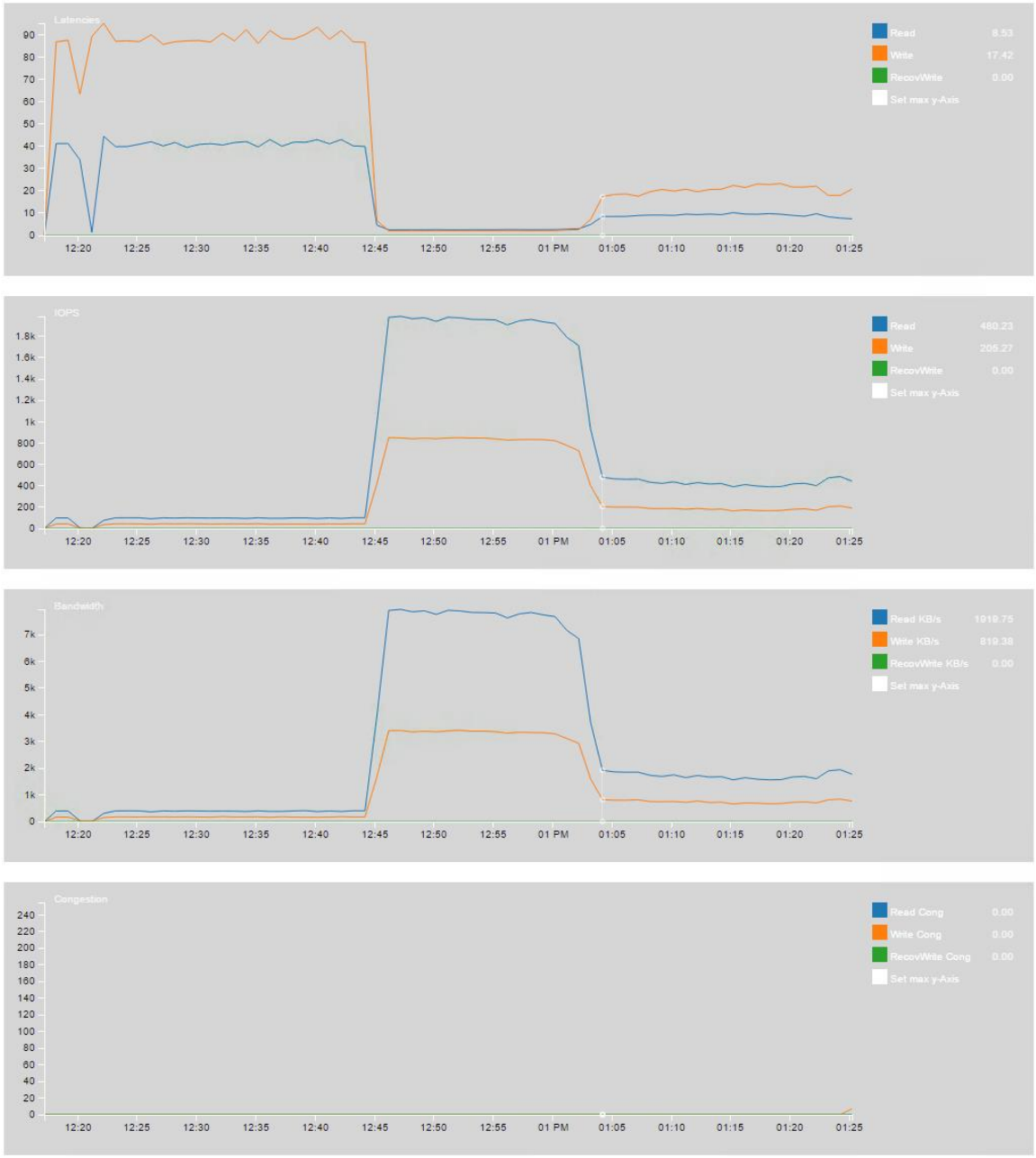
All I/O flows from the VSAN Client to the DOM owner and then to the disk layer. Virtual SAN tries to co-locate the owner and the client on the same host so as to not incur an additional network hop. However, the user can't influence where the owner for a given object is located, so reading the DOM Owner graphs and correlating them with the VSAN Client and VSAN Disks graphs can be take a little bit of effort.

While Client and DOM Owner might be on the same physical host (Virtual SAN tries to do this as much as possible), they may also be located on different hosts. And of course, if *NumberOfFailuresToTolerate* is greater than 0, at least one replica copy of the data will be on disks that are not on the same host. In fact, both copies may be on different hosts to the Client and DOM Owner.

Let's take a brief look at the DOM Owner – full size graphs next, and see the additional metric and performance information that these grapsh display.

# DOM Owner – Full size graphs

The Dom Owner full graphs display the same information we have previously seen with VSAN Client and VSAN Disk views. In this example, we can see detailed information for IOPS, Bandwidth, Congestion, Outstanding I/O and Latency.



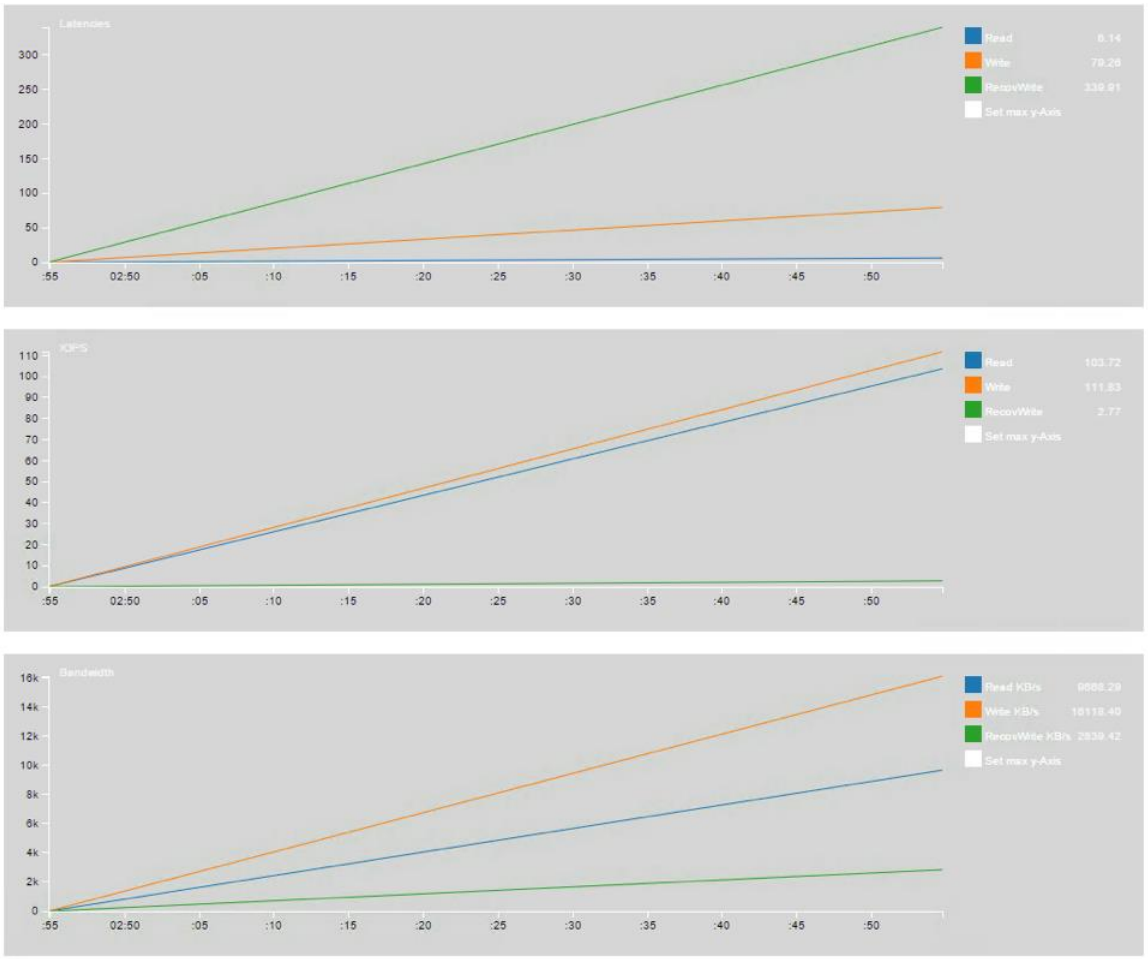

With this view, we can see if the latency is incurred with reads, writes or recoverWrites, and also does it coincide with a rise or fall in IOPS, Bandwith or

Outstanding IO. Full graphs will help you to look for patters across multiple metrics when troubleshooting performance issues with VSAN Observer.

### RecovWrites

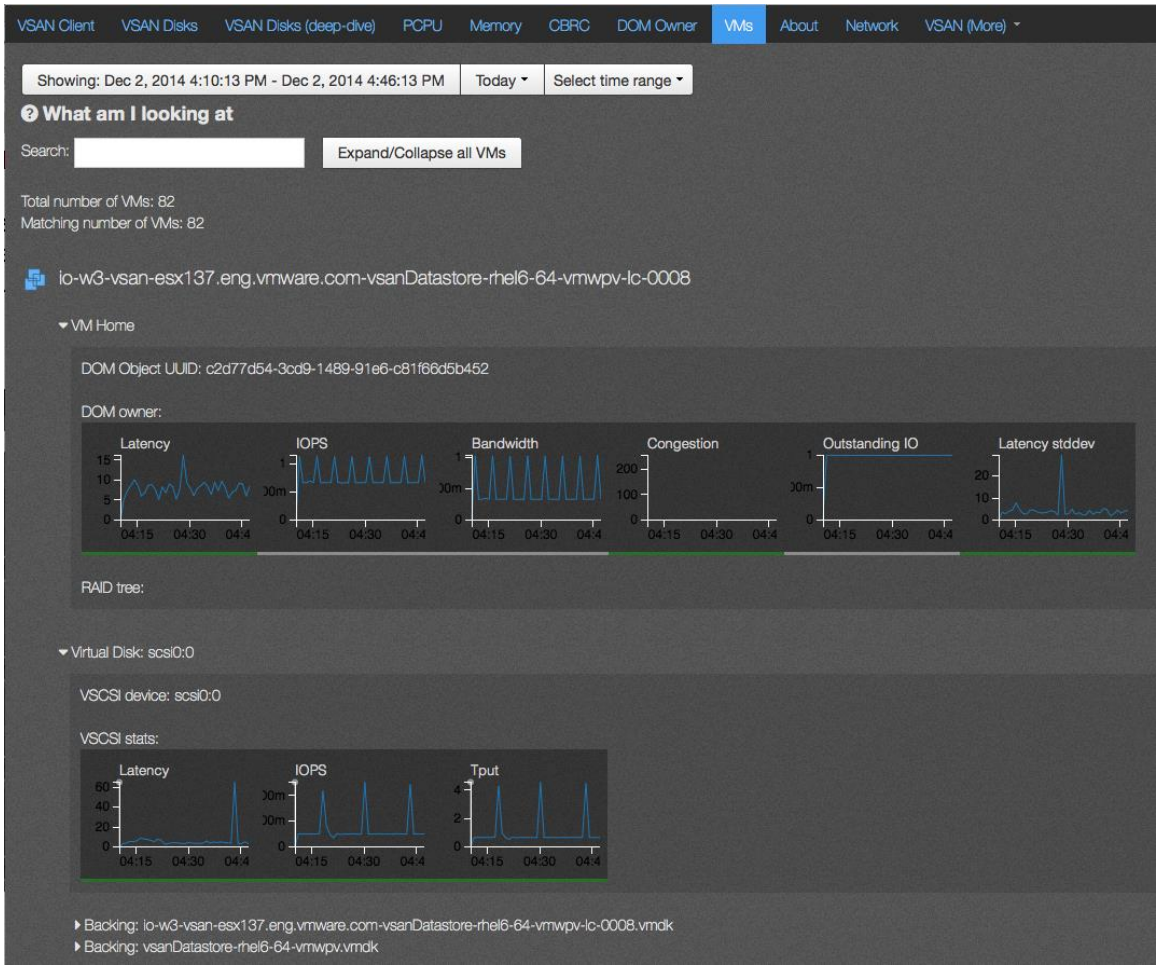
As was mentioned in the VSAN Client view, many graphs include a RecovWrite metric. This metric will only be visible when a rebuild operation is taking place, when a host is placed into maintenance mode and a full evacuation is requested for example. *DOM Owner – full graphs* view is the place to view this metric.

In this example, a new VM storage policy was applied to a number of VMs simultaneously. This is a screenshot taken from the DOM Owner – full graphs views showing how much I/O the rebuild tasks are creating.



# Navigating VSAN Observer – VM

The **VM** tab provides storage performance statistics as seen by each VM. This view allows a VM centric drill down, all the way down to individual VSAN "components" stored on VSAN disk groups. Latency, IOPS, Bandwidth, etc, are provided at the VM directory, virtual disk and backing disks level. The ability to drill down at a VM level is conveniently provided by this tab. Each VM has a VM Home Namespace that holds files like the .vmx configuration file and VM log files. In addition to this, each VM can have one or more virtual disks. Each of these virtual disks represents a single entity.




By selecting a full graphs view of the backing device, you get a screen identical to the DOM Object – full graphs, as seen previously.

## Navigating VSAN Observer – Network

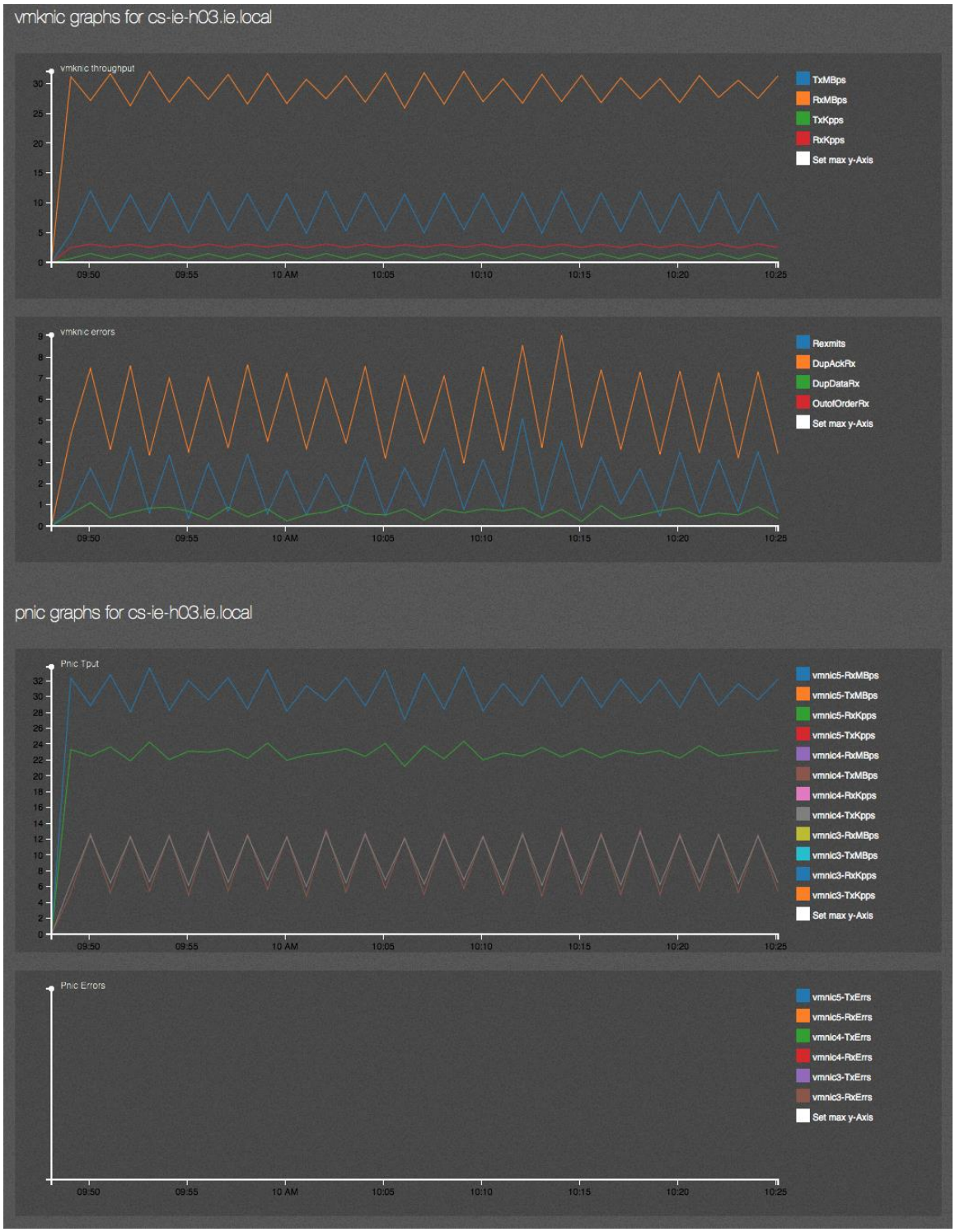
The VSAN Client view of Virtual SAN is essentially the virtual machine view of performance. The VSAN Disks view is essentially the physical disk layer. Since Virtual SAN is distributed in nature, there is a high likelihood of a virtual machine's compute residing on one node, and its storage objects residing on completely different nodes. For this reason, VSAN observer includes a network view, which will be essential for locating problems on the network infrastructure.


Most of the metric relate to error counters, similar to what you might find with CLI commands and tools. Each node provides an overview of physical NIC transmit and receive information, as well as detail on Tcp Errors.

A full graphs view provides more detailed information, including Virtual SAN VMkernel adapter information.

## Network – full graphs

For an even greater level detail, use the full graphs.



### vmknic throughput

This graph looks at the VMkernel adapter used by VSAN, with some good metrics for troubleshooting and performance monitoring.

- **TxMBps** – Amount of MB transmitted via the Virtual SAN VMkernel adapter by this host in MB per second.
- **RxMbps** - Amount of MB received via the Virtual SAN VMkernel adapter by this host in MB per second.
- **TxKpps** – Number of packets per second transmitted via the Virtual SAN VMkernel adapter by this host, measured in the 1,000s.
- **RxKpps** – Number of packets per second received via the Virtual SAN VMkernel adapter by this host, measured in the 1,000s.

### vmknic errors

This graph looks at the VMkernel adapter used by VSAN, and shows various metrics related to errors:

- **Rexmits** – Number of retransmits via the Virtual SAN VMkernel adapter
- **DupAckRx** – Number of duplicate acknowledgements received on the Virtual SAN VMkernel adapter
- **DupDataRx**- Number of times duplicate data was received on the Virtual SAN VMkernel adapter
- **OutOfOrderRx**- Number of times out of order frames were received on the Virtual SAN VMkernel adapter

### Pnic Tput

This is looking at the physical adapters (NICs) on the hosts:

- **vmnic-TxMBps** – Amount of MB transmitted via this physical network adapter by this host in MB per second.
- **vmnic-RxMbps** - Amount of MB received via this physical network adapter by this host in MB per second.
- **vmnic-TxKpps** – Number of packets per second transmitted via this physical network adapter by this host, measured in the 1,000s.
- **vmnic-RxKpps** – Number of packets per second received via this physical network adapter by this host, measured in the 1,000s.

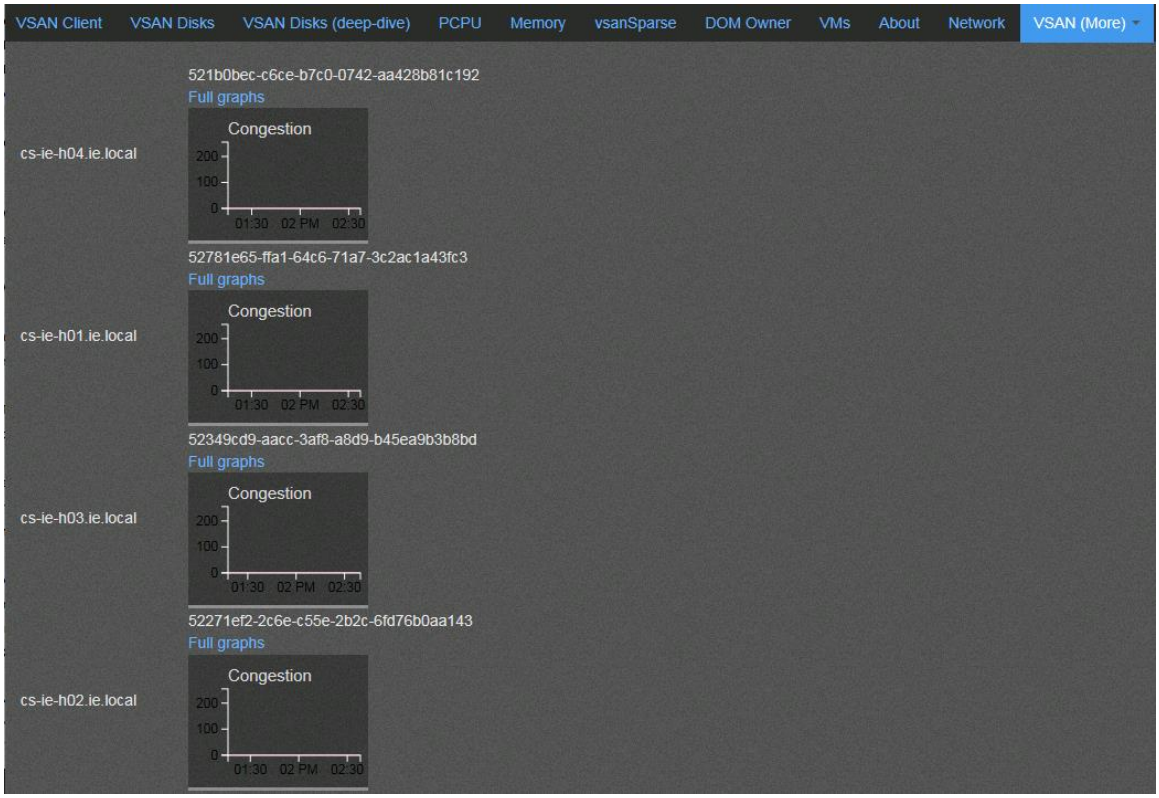
### Pnic Errors

Displays any errors on the physical adapter:

- **vmnic-TxErrs** – Physical NIC transmit errors
- **vmnic-RxErrs** – Physical NIC receive errors

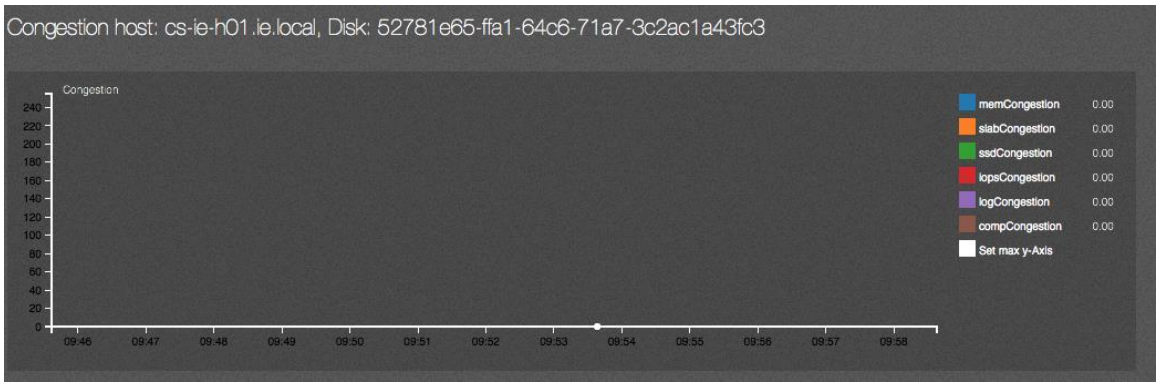
## Navigating VSAN Observer - VSAN (More) – Congestion

Virtual SAN might artificially introduce latencies to slow down writes to the flash device so that write buffers can be freed up. Obviously, if congestion begins to occur, higher latencies will also be observed. Under normal circumstances, congestion will be at zero.



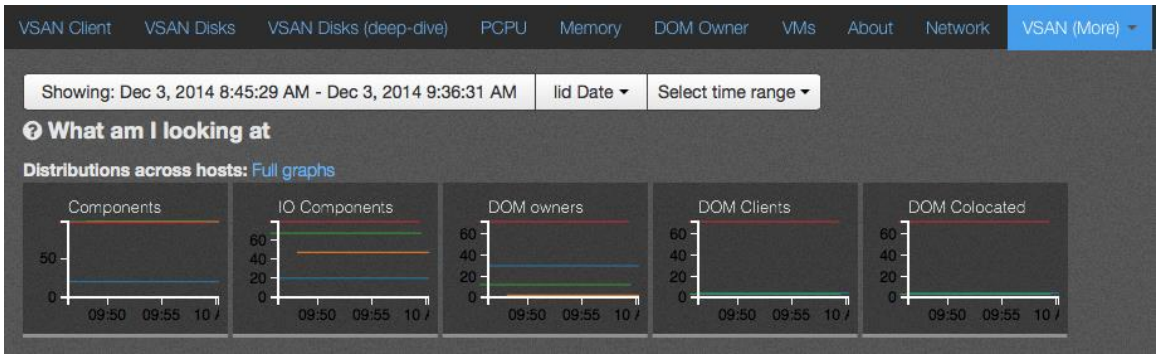

## VSAN (More) – Congestion – Full graphs

A full graph of congestion shows where it is being incurred. Of benefit to VMware support staff only, but shown here for completeness. No congestion in this scenario, which is what one would expect on a normal Virtual SAN cluster.



# Navigating VSAN Observer - VSAN (More) – Distribution

The **Distribution** tab shows how well Virtual SAN is balancing the objects (VM Home Namespaces, VMDKs, delta disks) and components (stripes, replicas, witnesses) across hosts in the cluster. The components are distributed across hosts in the cluster. Each host in version 5.5 has a 3000 component limit. In Virtual SAN 6.0, hosts with a v1 on-disk version continue to have the 3000 component limit, but those with a v2 on-disk format have a 9000 component limit per host. Balancing components across the cluster is an important side effect of balancing for performance and space.

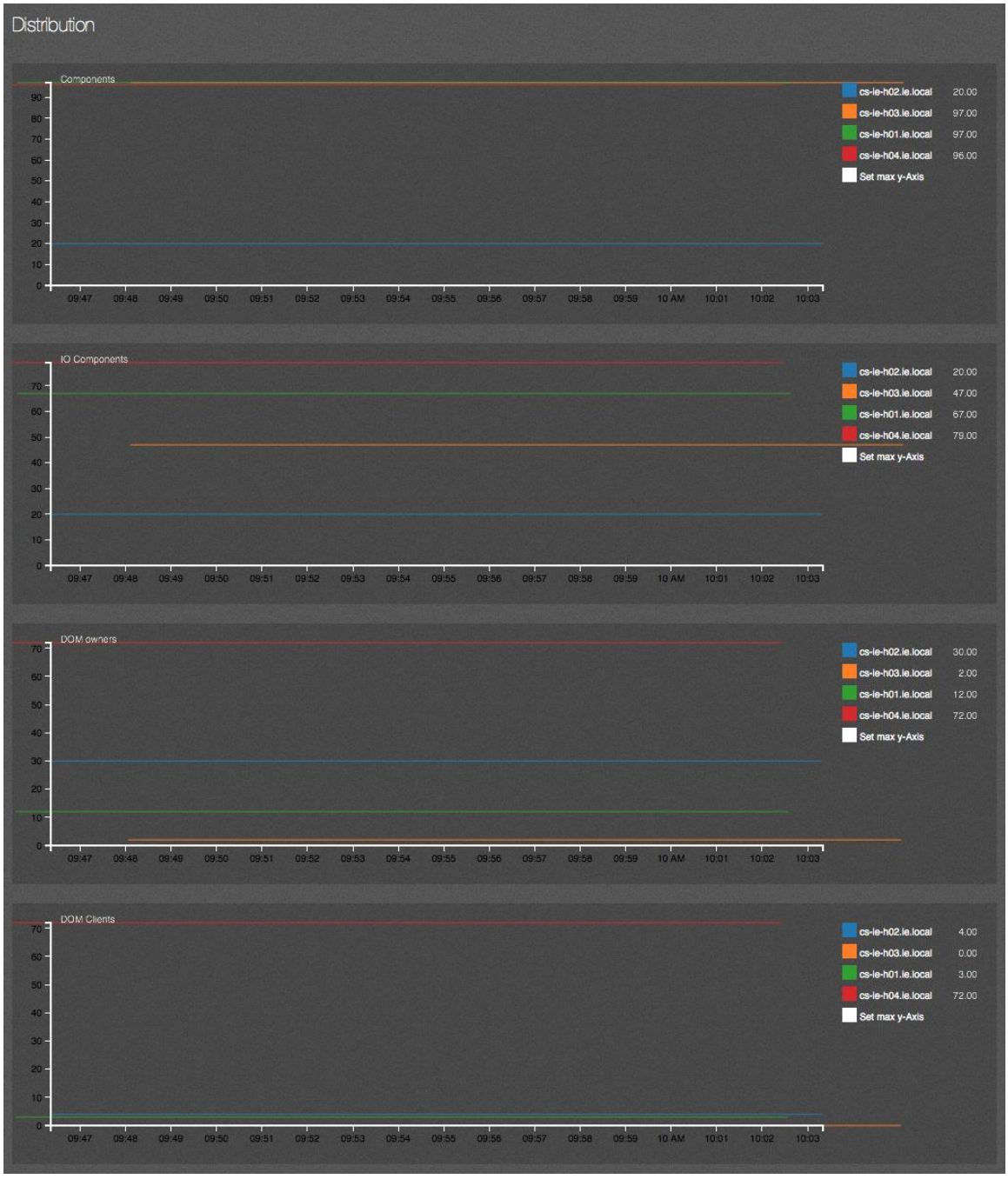



- **Components** includes both I/O components as well as witnesses.
- **IO Components** excludes witnesses and just shows how I/O components are balanced.
- **DOM owners** are in-memory states inside VSAN, showing the distribution of object owners around the cluster. This is something that the user can't control.
- **DOM Clients** are in-memory states inside VSAN, showing the distribution of object clients around the cluster. This is something that the user can't control.
- **DOM Colocated** – are in-memory states inside VSAN, showing where both the client and owner are co-located on the same host. This is something that the user can't control.

Distribution metrics are not tuneable. The information is shown here is primarily for the benefit of VMware Support.

## VSAN (More) – Distribution – Full graphs

Distribution also provides full graphs.




If we look at these charts, you will see that host cs-ie-h02.ie.local has far less components than the other host. This was because this host was recently evacuated and full data migration was the option used, which evacuated all of the components from this host to the remaining hosts in the cluster.

## Generate a VSAN Observer log bundle

VSAN Observer has the ability to generate a log bundle that can later be examined offline. This can be very useful for sending to a third party such as VMware Technical Support for troubleshooting. The option to generate a log bundle is `--generate-html-bundle`. For example, to generate a performance statistics bundle over a one hour period at 30 second intervals for a Virtual SAN cluster named `VSAN` and save the generated statistics bundle to the `/tmp` folder, run the command:

```
vsan.observer <cluster> --run-webserver --force --generate-html-bundle /tmp --
interval 30 --max-runtime 1
```

This command creates the entire set of required HTML files and then stores them in a tar.gz offline bundle in the /tmp directory (in this example). The name will be similar to `/tmp/vsan-observer-<date-time-stamp>.tar.gz`.

## Examine the VSAN Observer offline log bundle

To review the offline bundle, extract the tar.gz in an appropriate location that can be navigated to from a web browser.

## Generate a full raw stats bundle

One final mechanism to gather stats is to generate a full raw statistics bundle. This is only useful for VMware's support and engineering organizations. It generates a very large JSON file with all of the observed statistics for the period of the command run.

```
vsan.observer <cluster> --filename /tmp/vsan_observer_out.json
```

The vCenter Server retains the entire history of the observer session in memory until it is stopped with Ctrl+C.

## VSAN Observer command line options

| | |
|---|---|
| `--filename`<br>`-f` | Output file path, for example, `/tmp` |
| `--port`<br>`-p` | Port number for live statistics.<br>**Note**: The default port is 8010. |
| `--run-webserver`<br>`-r` | Run the live statistics. |
| `--force`<br>`-o` | Apply force. |
| `--keep-observation-in-memory`<br>`-k` | Keep observed statistics in memory even when commands ends. This command allows you to resume later. |
| `--generate-html-bundle`<br>`-g` | Generates an HTML bundle after completion. Pass a location. |
| `--interval`<br>`-i` | Interval value (in seconds) in which to collect statistics.<br>**Note**: The default value is 60. |
| `--max-runtime`<br>`-m` | Maximum number of hours to collect statistics. This command caps memory usage.<br>**Note**: The default value is 2. |

# 16. Troubleshooting Virtual SAN performance

As highlighted in the introduction, VMware recommends using the Virtual SAN Health Services to do initial triage of performance issues. The Virtual SAN Health Services carry out a range of health checks, and may detect issues that can directly impact performance. The Health Services directs administrators to an appropriate knowledge base article depending on the results of the health check. The knowledge base article will provide administrators with step-by-step instruction to solve the problem at hand.

Please refer to the *Virtual SAN Health Services Guide* for further details on how to get the Health Services components, how to install them and how to use the feature for troubleshooting common Virtual SAN issues.

## Virtual SAN performance expectations

Virtual SAN is a fully distributed, scale-out storage system, designed to support the aggregate needs of multiple VMs in a cluster. As such, performance profiles can be different than what is frequently seen with traditional monolithic storage arrays.

Virtual SAN is designed to maximize the number of application IOPS in a cluster, while minimizing CPU and memory consumption. As such, it uses a very different design point from arrays that have dedicated CPU and memory.

While testing a single application in a VSAN cluster may be reasonable, a full performance picture will not emerge until multiple applications across the cluster are tested.

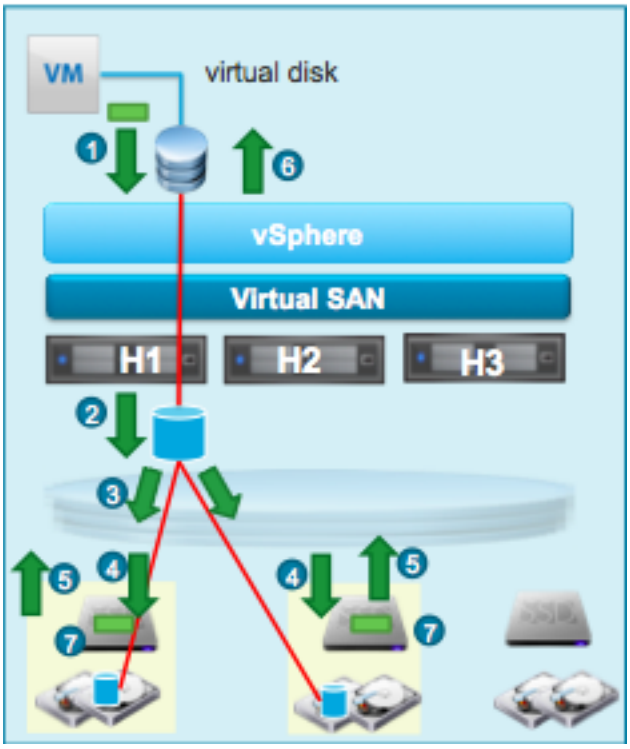
As there is wide variability in supported VSAN configurations (flash devices such as SSDs, IO controllers, magnetic disks for hybrid configurations, etc.) and application workloads (cluster size, working set, read/write mix, IO size) the best advice is to test a proposed production configuration with a workload that is as close to expected as possible.

## I/O flow

In order to successfully troubleshooting performance issues, a certain level of understanding of Virtual SAN internals is required. The intention is not to take a deep-dive into the all of the internal workings of Virtual SAN, but instead give you an appreciation of some of the basic operations. Let's start with I/O flow.

### Anatomy of a write in hybrid configurations

When a VM issue a write, and that write is acknowledged as having completed successfully to the VM, the user is guaranteed that the data just written is persisted according to the policy specified. For example, if "failures to tolerate" was set to 1 and there are 2 mirror copies of the VMDK on different hosts, Virtual SAN guarantees that the data has been written to both mirrors.



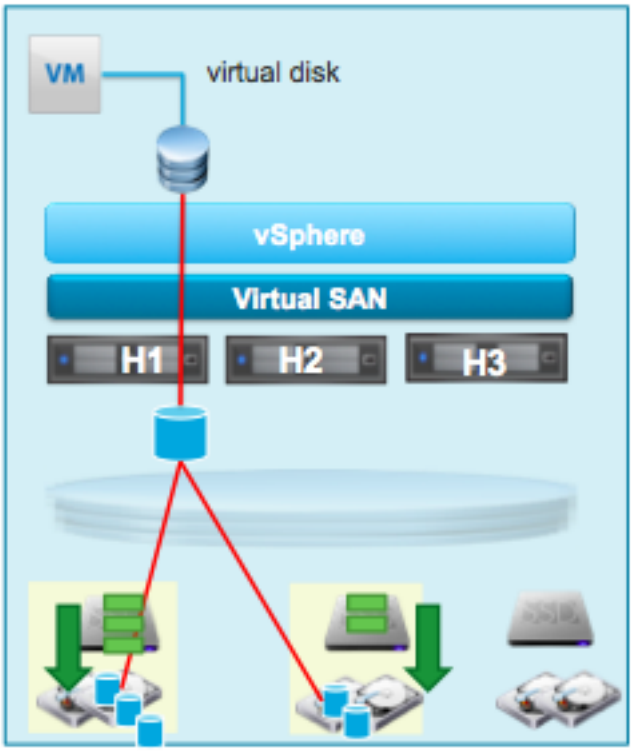

Lets take the example where there is a 3 node Virtual SAN cluster. There is a VM running on host **H1** and it is using a VM Storage Policy which has the capability "Number Of Failures To Tolerate = 1". This means that the disk object has two mirrors, one on host H1 and the other on host H2. Let's now examine the steps involved in a write operation:

1. Guest OS issues write operation to virtual disk
2. Owner clones write operation
3. In parallel: sends "prepare" operation to H1 (locally) and H2
4. H1, H2 persist write operation to Flash (log)
5. H1, H2 Acknowledge prepare operation to owner
6. Owner waits for ACK from both 'prepares' and completes I/O.
7. Later, the owner commits a batch of writes to hard disk or flash used as capacity.

### Destaging writes from flash to magnetic disk in hybrid configurations

After the previous write operation, we are now left the newly written data residing on flash on two different ESXi hosts. How is this data now destaged to the capacity layer, which is version 5.5, is always made up of magnetic disks? Remember, magnetic disks are good with sequential workloads. So Virtual SAN tries to make things optimal for these magnetic disks. Virtual SAN uses a self-tuning algorithm that decides how often writes on the flash cache device (e.g. SSD) de-stage to magnetic disk. The heuristics used for this are sophisticated and take into account many parameters such as rate of incoming I/O, queues, disk utilization, and optimal batching.

As we have seen, Virtual SAN accumulates written data in the write buffer of the flash device. It has an *elevator algorithm* that runs independently on each disk group and decides (locally) whether to move any data to the magnetic disk and when should it do it. It juggles multiple criteria to reach this decision; most importantly of all, it batches together big chunks of data that are physically proximal on a magnetic disk and retires them altogether. Therefore, writes to the magnetic disk are largely sequential. But it is also conservative. It does not rush to move data if there is plenty of space in the Write Buffer because data that is written tends to be overwritten multiple times within a short period of time. So Virtual SAN avoids writing the same block over and over to the magnetic disk.



The block size that Virtual SAN uses to write to the flash cache device is 4KB. The block size that Virtual SAN uses when de-staging to magnetic disk is 1MB, which is also the stripe size used on magnetic disks and flash devices when they are used for capacity in version 6.0.

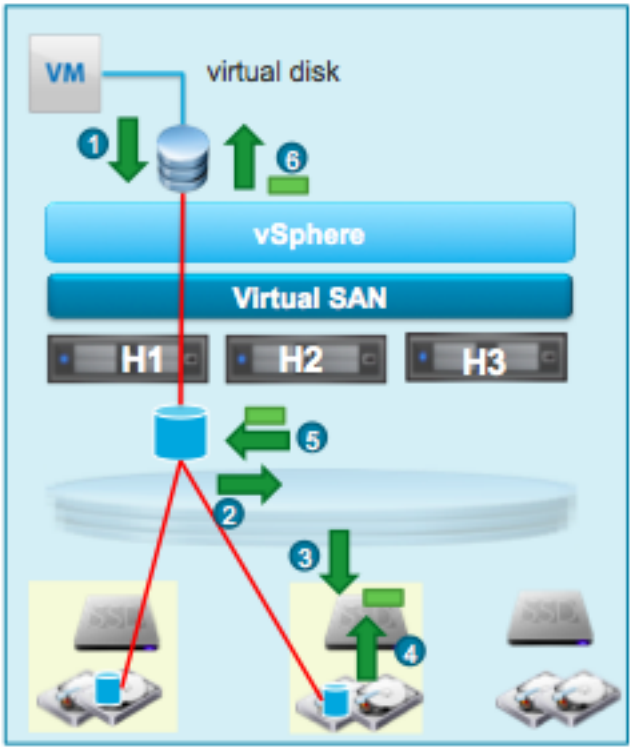
The Virtual SAN flash is split into a read buffer and a write cache, 70% read and 30% write. One common question that gets asked about the anatomy of a write operation is whether or not there is a subsequent write to the read cache so we don't need to re-read the currently written data from the magnetic disk.

In Virtual SAN 5.5, the data in the read cache that may be affected by a new write is invalidated, and if the VM issues a read for that data block before it has been destaged to magnetic disk, we will read that data from the write buffer. A subsequent read after destaging may then incur a read cache miss.

In Virtual SAN 6.0, this behavior has been changed. When the data is to be destaged to magnetic disk from the write buffer, the corresponding read cache entry (if it exists) is updated with the most recent data. Cache entries are not invalidated unless the cache decides that they're no longer useful. This should mean less magnetic disk I/O in Virtual SAN 6.0 when compared to Virtual SAN 5.5.

### Anatomy of a read in a hybrid configuration

Now that we saw how writes work, let's take a look at reads. We will use the same example as before: a VM with a VMDK object has two mirrors, one on ESXi host H1 and the other on ESXi host H2.




1. The Guest OS issues a read request from disk
2. Owner chooses which mirror copy to read from. The owner of the storage object will try load balance reads across replicas and may not necessarily read from the local replica (if one exists). On Virtual SAN, a block of data is always read from same mirror which means that the data block is cached on at most on one flash device (SSD); this maximize effectiveness of Virtual SAN's caching
3. At chosen replica (H2): read data from read cache, if it exists.
4. Otherwise, we incur a read cache miss so we must read from magnetic disk and placed in the read cache
5. Return data to owner
6. Owner completes read operation and returns data to VM

### Anatomy of a read in all-flash configurations

The major difference between a read in a hybrid configuration and a read in an all-flash configuration is that in an all-flash configuration, the flash cache is not used for caching reads. It is dedicated as a write cache only. If the read operation does not find the block in the flash cache in an all-flash configuration, then it is read directly from the 'capacity' flash device. The block does not get placed in the tier1 flash cache, like a hybrid configuration. The performance of the all-flash capacity tier is more than enough for reads.

### Anatomy of a write in all-flash configurations

In the all-flash configuration, the tier1 flash cache is now used for write caching only, what can be considered a write back cache. When the working set is bigger than write cache, old data blocks are evicted from the tier1 write cache to the flash capacity devices. If the working set fits of the virtual machine fits completely in the tier1 write cache, there are no data blocks written to the flash capacity device at all.

### Virtual SAN caching algorithms

Virtual SAN implements a distributed persistent cache on flash devices across the Virtual SAN cluster. In Virtual SAN 5.5 and 6.0 hybrid configurations, caching is done in front of the magnetic disks where the data lives. It is not done on the client side, i.e. where the virtual machine compute resides. A common question is why we took this approach to caching?

The reason for this is because such distributed caching results in better overall utilization of flash, which is the most valuable storage resource in our cluster. Also, with DRS and vMotion, virtual machines move around hosts in a cluster. You don't want to be moving GBs of data around or rewarming caches every time a VM migrates. Indeed, in Virtual SAN you will see no performance degradation after a VM migration.

### Enhancements to caching algorithms in 6.0

Virtual SAN 6.0 uses a tier-1 flash device as a write cache in both hybrid and all-flash disk groups.

However, the caching algorithm optimizes for very different goals in each case. In hybrid groups, the caching algorithm aims at accumulating large proximal chunks of data for each magnetic disk. The priority is to maximize the write performance obtained from the magnetic disks by applying a nearly sequential workload to them when destaging from the flash cache (elevator algorithm) to magnetic disks.

In the case of all-flash disk groups, the goal is to keep as many hot blocks (written often) in the flash cache device. Thus, a flash cache device that has a higher endurance will be able to satisfy a higher rate of write operations. There is no attempt made to optimize all-flash configurations for proximal destaging, as flash devices handle random workloads very well.

### Latency consideration with distributed cache

A common question related to the latency the network introduces, if you need to access data that resides on another host in the cluster.

Typical latencies in 10GbE networks are in the range of 5 – 50 microseconds. When you see the specs of flash devices you hear about latencies in the range of 50 – 100 microseconds. But these are latencies values when you send one I/O at a time. When you operate those flash devices at thousands of IOPS, then latency values start to grow to 1 millisecond or even higher in some cases. Adding something like 10 microseconds on top of 1 millisecond of latency doesn't make a huge difference to the overall performance.

## How VM Storage Policies impact performance

### Where stripe width helps and where it may not

While setting disk striping values can sometimes increase performance, that isn't always the case. As an example, if a given test is cache-friendly (e.g. most of the data is in cache), striping won't impact performance significantly. As another example, if a given VMDK is striped across disks that are busy doing other things, not much performance is gained, and may actually become worse.

### A word of caution on flash read cache reservation

The policy setting of Flash Read Cache Reservation (applicable to hybrid only) should be used only in the event of an identifiable performance improvement. If you set a read cache reservation in a policy, and you associate it with multiple virtual machines, you could consume a significant amount of flash on the host or hosts in question, thus starving the other virtual machines of valuable cache resources. Handle with care!

Since all virtual machines will share the read cache and write buffer equally, it is strongly recommended that you do not assign a dedicated amount of read cache to virtual machines unless there is a pressing need to solve an underlying performance issue for a specific set of virtual machines.

# An overview of different workload types

Another common question is whether certain workloads are suitable for Virtual SAN, and how these different workload types impact on performance. The best way to approach this subject is to discuss the different workload types. There is a very good whitepaper on this topic called "Storage Workload Characterization and Consolidation in Virtualized Environments" if you wish to do further reading: https://labs.vmware.com/download/10/

## Single workloads versus multiple workloads in a VM

When it comes to performance tuning, the easiest approach is to run a single workload in a virtual machine. In this way, any additional resource (CPU, memory) that is given to this VM is consumed by the single application running in the virtual machine. Any tuning that is done is also going to benefit that workload. When there are multiple workloads running in a VM, it becomes more difficult to ensure that any performance tweaks that are done for one workload do not impact the other workload(s). And also that any additional resources added to the virtual machine are consumed by the workload that needs it.

A test environment should reflect the production environment, e.g. if you plan on running multiple workloads in a single VM, that's what you should test, otherwise stick to testing a single workload in a single VM.

## Single VM versus multiple VMs

To get the best performance overview of Virtual SAN, multiple VMs across multiple hosts in the cluster should be deployed. Since Virtual SAN is distributed, a single VM can only consume the resources on a single ESXi host, even though it's storage may be located on other hosts. Therefore benchmarking Virtual SAN performance should not be done in the context of a single VM, which may get bottlenecked by the compute resources, storage controller resources or indeed flash resources on that single ESXi host.

## Steady state versus bursty (averages versus peaks)

Many performance-testing tools generate steady-state workloads using a few I/O block sizes. Many real-world applications are quite different: they are bursty and may generate multiple I/O profiles. This is important, as application performances during peaks are what users notice.

Let's look at an example.

Microsoft Exchange Server 2007 is a popular commercial mail server that can

support thousands of users simultaneously. It is also an application that is regularly virtualized and run in virtual machines.

`Loadgen` is an MS Exchange workload generator, used to simulate Exchange workloads. Typically, administrators separate out the Exchange data and log onto different VMDKs. A data disk typically sees a lot of random reads. Writes are mostly sequential. Data reads are 8KB while writes vary from 8KB to 32KB, with almost 75% at 8KB. Reads and writes show significant "burstiness" in this workload. Writes undergo more bursts presumably due to periodic flushing of buffers. As expected, a log disk doesn't receive any reads. For writes, log I/O is completely sequential. Log writes sizes are predominantly 512 Bytes.

When designing for virtual machine workloads of this nature, it is important to ensure that "bursts" do not negatively impact the performance. Administrators will need to ensure that the underlying storage can handle "bursty" workloads. Since all writes go to the flash layer on Virtual SAN, we expect to be able to handle this sort of "bursty workload" relatively well. However, long periods of "burstiness" could fill up the cache and/or backing disks if it hasn't been sized correctly.

### Random versus sequential

Virtual SAN uses flash as a caching layer. Therefore sustained sequential write workloads that do not do any reads of the data do not benefit from the performance of having a caching layer as much as random workloads with a mixture of writes and reads.

Most real-world workloads are combinations of reads and writes. For example, the classic "OLTP (Online Transaction Processing) mix" is 70% reads, 30% writes using 4K blocks. Best results are obtained when your performance mix reflects your application mix.

Virtual SAN provides certain benefits for write intensive workloads, such as coalescing of writes and the overwriting of data in cache that has not yet been destaged to the capacity layer. However sustained sequential write workloads simply fill the cache. Future writes will have to wait for a destage to take place from flash to magnetic disk. The performances of workloads of this nature will reflect the ability of the magnetic disks to handle the destaging operations from the flash layer. In this case, a very large stripe width may improve performance the destaging.

The same is true for sustained sequential read workflows. If the block is not found in cache, it will have to be fetched from magnetic disk in Virtual SAN 5.5 (a cache miss).

Some examples of sequential workloads include backup-to-disk type operations, and writing SQL transaction log files.

Some examples of random workloads include Online Transaction Processing (OLTP) database access, such as Oracle. A common way of testing random I/O is using DVDStore, which is an online e-commerce test application with a backend database component, and a client program to generate workload.

### Read versus write

We already mentioned that hybrid Virtual SAN configurations use flash for read caching and write buffering. Sizing cache so that the workload can be largely contained in cache will produce the most desirable performance results from Virtual SAN. All of the writes will go to the write buffer in flash, but it is not guaranteed that all of the reads will come from the read cache, also in flash. However, that is what you would like to achieve, and avoid any read cache misses.

Read cache sizing is very important; write cache sizing somewhat less so. Insufficient read cache causes cache misses, which are very expensive operations. Writes are different; they always go to cache and are eventually destaged. More write cache means that this destaging happens less often and more efficiently, but the effect of having less write cache is much less dramatic when compared to read cache.

In the current release, there is no control over the read/write cache ratios. It is hard-coded at 70:30 write:read. However, there is the ability via the VM storage policy to set a read cache reservation for your particular virtual machine, if you think that it will benefit from having its own dedicated read cache, rather than sharing it with other virtual machine storage objects. But do so carefully!

### Where cache helps and where it may not

Flash as cache helps performance in two important ways. First, frequently read blocks end up in cache, dramatically improving performance. Second, in hybrid configurations, all writes are committed to cache first, before being efficiently destaged to disks – again, dramatically improving performance.

However, data still has to move back and forth between disks and cache. A sustained sequential write test ends up being a test of the back end disks, as does a sustained sequential read test. We also mentioned that long periods of "burstiness" could fill up the cache, and I/O may need to be satisfied from the magnetic disk layer.

Also, you need to be aware that most real-world application workloads take a while for cache to "warm up" before achieving steady-state performance.

### Working set size

When working data set is 100% of the VMDK, the Virtual SAN read cache and write buffer gets filled up quickly. This will lead to steady state performance that is primarily supported by the back-end magnetic disks in a hybrid configuration. By using a smaller, more realistic working data set, Virtual SAN cache layer will give great performance.

### Guest file systems can matter

There have been reports of significant differences in performance between different guest file systems and their settings, e.g. Windows NTFS and Linux. If you aren't getting the performance you expect, consider investigating whether it could be a file system issue in the Guest OS. In many cases, consider using raw volumes rather than formatted file systems for benchmarking.

## What operations can impact performance?

While no means complete, the following list discusses operations on Virtual SAN that might impact performance. These operations should be considered in detail, and perhaps scheduled during maintenance or out-of hour periods by a vSphere/Virtual SAN administrator.

### Rebuilding/resyncing operations

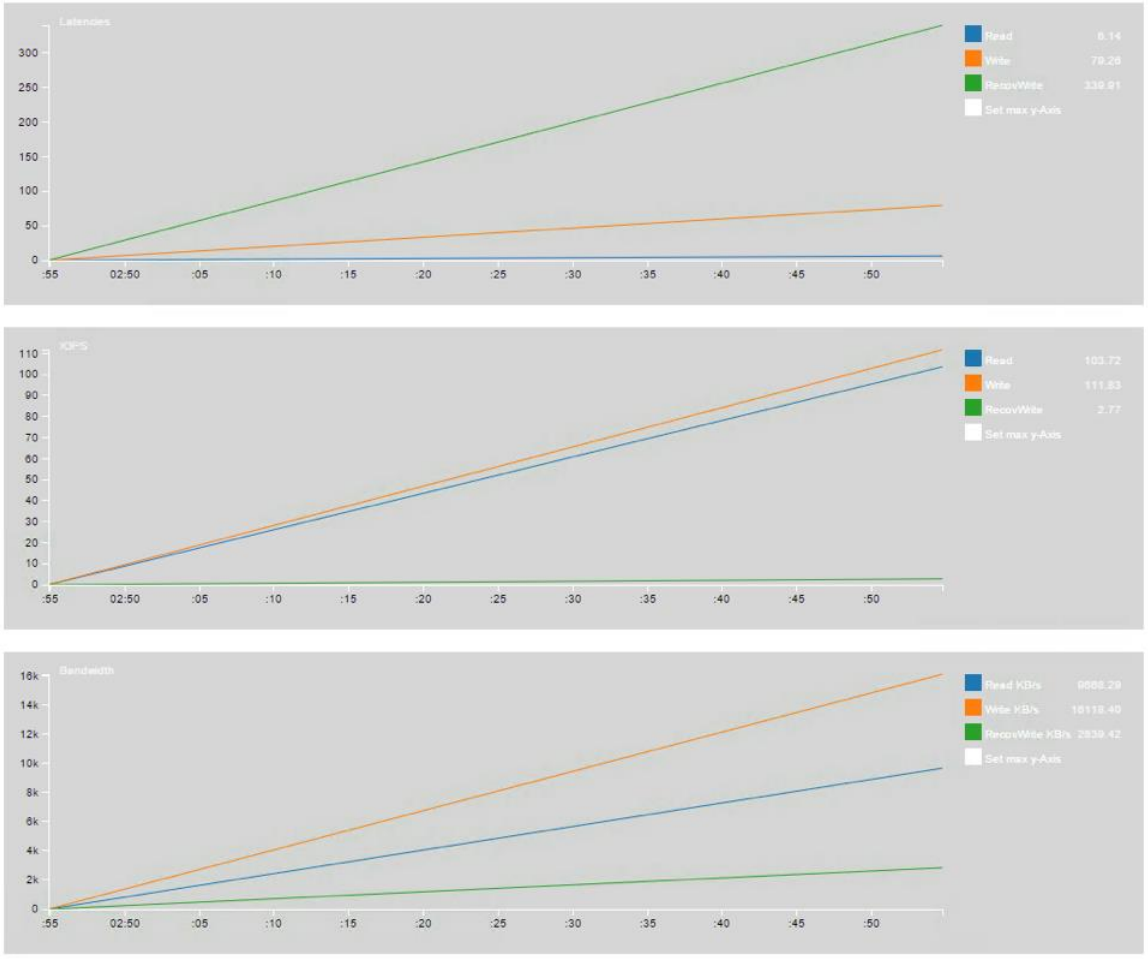
There are a number of occasions when Virtual SAN will need to rebuild or resync objects and/or components:

1) When a host is placed in maintenance mode, and the administrator wishes to do a full data migration. The objects and components on the disks in that host will need to be located elsewhere.
2) When a user changes policy on an object (or set of objects), there is a good probability that Virtual SAN might get busy for a time by essentially recreating all objects affected by the policy change.
3) If there is failure in the cluster, and components need to be recreated elsewhere is the cluster.

During these operations, resync I/O and virtual machine I/O will have to share the network and other resources, which may in turn affect overall virtual machine I/O performance.

## Observing rebuilding/resyncing operations

When examining VM I/O traffic versus resync/recovery traffic after a failure, use the "DOM Owner" tab in VSAN observer. The "full graphs" view displays the I/O split into 3 categories: read, write, recovWrite (short for recovery write). That displays how many IOPS Virtual SAN is doing for virtual machine read and write operations versus component rebuild recoveryWrite operations, and also how they compare on bandwidth.




When a disk or host fails, many objects will have components on that disk or host, so many new mirror objects may be created. This allows Virtual SAN to further optimize the rebuild process. As the entire cluster acts as a "hot spare", Virtual SAN will spread the replacement component across the cluster, which means the write I/Os to creating the new mirror will be spread among as many physical disks as possible.

Similarly, as each object is placed individually and as objects are distributed in a balanced way, the remaining copies of data for the effected objects are spread

around the remaining cluster. This means also read I/Os for creating the new mirror are spread among as many physical disks as possible.

Together this means that Virtual SAN can use a small portion of the I/O capacity of each physical disk in the cluster to facilitate a distributed rebuild process which is highly parallel and hence rebuild speed is highly optimized.

Another very significant optimization for resync/move of data on Virtual SAN is the fact that Virtual SAN is fully aware of thin provisioning when syncing/moving data. So if there are only 800GB of actual data on a 2TB drive, the total amount of data that needs to be rebuilt is just 800GB, as Virtual SAN is fully aware of actually allocated blocks on a per-component basis at all times.

This optimization also applies when the user has requested, "thick provisioning" (i.e. set the "object space reservation" policy to 100%), as Virtual SAN still tracks which blocks have actually been written to, versus just being "reserved" for future use by the object.

If a component that was previously failed comes back up, it is usually STALE, i.e. it doesn't have the latest data. Virtual SAN knows this by comparing the sequence numbers of the components contributing to the object. In this case, Virtual SAN will utilize bitmaps that keep track of which data has been written since the component went away and only those regions of data are synced up. For the duration of syncing, the component is not considered ACTIVE. Instead, it will be shown as "Absent - resynching" in the vSphere Web Client.

Also consider that a previously failed component that comes back online may not, in fact, be STALE and have a current copy of the data. This might happen when the component was part of the last active working set before accessibility was lost due to too many failures. In such a case, Virtual SAN will be able to just bring the component online.

### Backup/restore operations

There are two distinctions to be made in this section. Is Virtual SAN being used as a backup target, or is it being used as a destination? VMware supports vSphere Data Protection (VDP) running on Virtual SAN, so it could be that it may not only be a source for VMs that are being backed up, but also a destination for the backups.

When backing up virtual machines on a Virtual SAN datastore, one should expect that the initial backup would involve sequential reads for the initial backup. If using VDP on Virtual SAN as a backup store, one should expect there to be a significant amount of sequential writes for the first backup. Subsequent backup operations are incremental, so will only read and write blocks of data that have changed since the last backup.

Doing backups and restores during production hours, when other virtual machines are still running, may have an impact on overall performance, depending on how many VMs are being backed up, and/or restored. A recommendation would be to do backup and restore activities like this out-of-hours so as to lessen the impact on the production VMs, although incremental backups have less of an impact when compared to full backups.

One additional note of backup and restores; keep in mind is that the VM Storage Policy is not backed up with the VM. This isn't a problem if you are restoring to the original location and overwriting the existing VM; in this case you maintain the VM Storage Policy when you restore. However, if you restore to an original location and the original VM no longer exists, or if you restore to a new location, the VM will be restored with a default VM Storage Policy (which has *NumberOfFailuresToTolerate = 1*). Therefore you will need to reapply the VM Storage Policy to this virtual machine after it has been restored. This will lead to a new reconfiguration and resync, and this operation may once again have an impact on production virtual machine I/O.

### vMotion

vMotion operations consume a significant amount of CPU resources and network bandwidth to complete the operation in the shortest time possible. This can inadvertently affect the performance of Virtual SAN if not planned correctly.

Guidelines take from a vSphere 5 vMotion Performance Study (http://www.vmware.com/files/pdf/vmotion-perf-vsphere5.pdf) recommend the use of 10Gb NICs, Network I/O Control (NIOC) and setting aside a certain amount of CPU just for vMotion operations. Failure to follow these guidelines could inadvertently impact the performance of Virtual SAN during vMotion operations. The good thing is that Virtual SAN entitles you to the Distributed Switch functionality, no matter which vSphere edition you use with Virtual SAN. This means

that you have NIOC that will allow you to place Quality Of Service (QoS) on vMotion, Virtual SAN and other network traffic types.

With this in place, vMotion can run during production hours, and won't inadvertently impact Virtual SAN operations.

### Virus Scans

From the VMware paper "Antivirus Best Practices for VMware Horizon View 5.x" which can be found here http://www.vmware.com/files/pdf/VMware-View-AntiVirusPractices-TN-EN.pdf, the challenge with Virus Scans is most virus scanning model involves agents executing on every desktop performing antivirus scanning. During these operations, system resource usage can spike or become overly committed. Performance is severely impacted by these "antivirus storms."

Traditional antivirus agents are resource-intensive and antivirus storms can cause 100 percent saturation in the compute and storage (including a Virtual SAN datastore). In addition, the memory footprint is significant when antivirus software is installed on each virtual machine.

VMware has solutions to address the "antivirus storm" issue by consolidating and offloading all antivirus operations into one centralized security virtual appliance. The whitepaper referenced above has further information.

## Notes on using IOmeter with Virtual SAN

Many administrators rely on a tool called IOmeter to do storage performance testing. This is also true for performance testing on Virtual SAN. When the results of this testing are not correctly understood, or do not meet expectations, the assumption is that there is an underlying performance issue with Virtual SAN. This is not necessarily the case, and more often than not it is how IOmeter has been configured to run. If the plan is to use IOmeter to test Virtual SAN performance, there are a number of factors that need consideration.

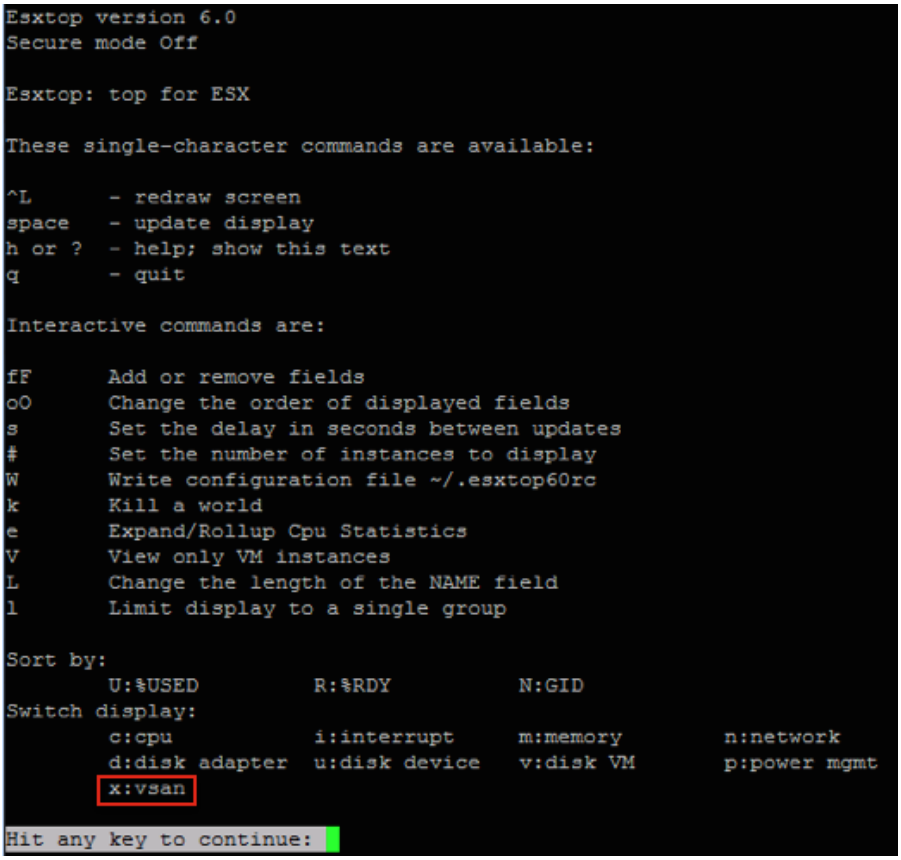
1. Iometer creates synthetic workloads, not reflective of real-life workloads. Real life virtual machine workloads do not behave like synthetic IOmeter workloads. IOmeter workloads tend to be steady state, repeating the same patterns over and over again. Production workloads are typically not like that, with peaks and troughs, as well as bursts of traffic.
2. The working set of a VM running IOmeter can be configured to be much, much larger than a working set of a production VM. IOmeter can consume a whole VMDK as its working set, which is usually not reflective of production workloads.
3. Your cache is limited. When using IOmeter consider the working set and if it fits in cache. Consider a small number of VMs with a large working set

(VMDK) versus a large number of VMs with a small working set (VMDK). A VMDK size of 1GB for IOmeter should be more than sufficient. However you might be able to tune the IOmeter working set by choosing a specific number of sectors to work on, if I remember correctly.

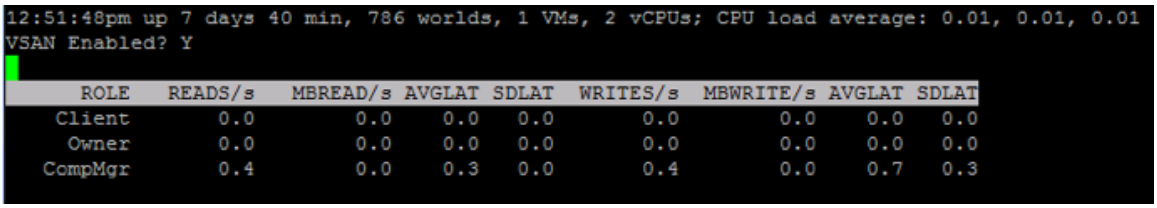
4. Be aware of creating a large working set; you can fill cache and then your performance will be based on magnetic disk performance due to cache misses.
5. Be aware of using sustained sequential workloads – these will not benefit from cache. Use random workloads or a mix of workloads to make full use of Virtual SAN's cache layer. Pure sequential workloads are rare, but important. Virtual SAN may not be best suited to these workload types.
6. Use VSAN Observer to check read cache hit rate and write buffer evictions/destaging.
7. How many vCPUs do your VMs running IOmeter have versus number of pCPUs? Scheduling these VMs could be a problem if there is contention, so check if these VMs are counting up "ready to run" (%READY) indicating that there is no pCPU resources for them to run on. However you will also need to balance this with a number of vCPUs that can drive adequate I/O. Don't overdo it on number of VMs or number of vCPUs per VM.
8. Do not set very large Outstanding I/O (OIO) when running lots of VMs and worker threads. OIO settings between 2 & 8 per VM should be sufficient.
9. Read/Write mix – start with the traditional "transactional mix" is 70% read, 30% write. You can also try a 50/50 configuration, which is considered write intensive. You can later change to a different mix: e.g. 35% read, 65% write and 65% read, 35% write.
10. Set your I/O Size to at least 4KB. However, depending on workload, you might want something bigger.
11. The observed average latency should be below 5ms, and the max latency should also be below 50ms. Your numbers may actually be substantially better, but the conservative numbers are given to cover a wider range of devices.

## Using esxtop for performance monitoring

In version 6.0, `esxtop` now has a new, limited view for Virtual SAN. A new option has been added (x) to `esxtop`:




By default, this view will display the 3 different roles of the Distributed Object Manager (DOM) that exist on the host. The roles are client, owner and component server. The statistics that are displayed by default show reads and writes and include IOPS, bandwidth, average latency and latency standard deviation.




There is one additional field that is not displayed by default. This is the "recovery write statistics" and displays how much writing on behalf of a remediation task that is doing a rebuild of components in the Virtual SAN cluster.

To display this field, hit the 'f' (for fields) key, and a list of all the fields that can be monitored is displayed. Next, click on the (d) to enable the monitoring of the recovery write stats.

```
Current Field order: ABCd

* A:  ROLE                 = DOM role name
* B:  READ STATS           = IOPS, bandwidth, average and standard deviation latency (ms) for READ
* C:  WRITE STATS          = IOPS, bandwidth, average and standard deviation latency (ms) for WRITE
  D:  RECOVERY WRITE STATS = IOPS, bandwidth, average and standard deviation latency (ms) for RECOVERY WRITE

Toggle fields with a-d, any other key to return:
```

While this information is also available in the VSAN Observer, this does allow administrators a quick way of monitoring I/O activity on a per host basis. It is not available in version 5.5.

# Using RVC for performance monitoring

There are some RVC commands that can give administrators an overview of Virtual SAN performance. Here is one such command.

### vsan.vm_perf_stats

```
vsan.vm_perf_stats ~/vms/W2k12-SQL2k12 --interval 10 --show-objects
output:
2014-10-31 15:19:33 +0000: Got all data, computing table
+--------------------------+-------------+--------------+--------------+
| VM/Object                | IOPS        | Tput (KB/s)  | Latency (ms) |
+--------------------------+-------------+--------------+--------------+
| W2k12-SQL2k12            |             |              |              |
|    /W2k12-SQL2k12.vmx    | 0.3r/0.3w   | 0.2r/0.2w    | 0.5r/1.2w    |
|    /W2k12-SQL2k12.vmdk   | 1.2r/6.1w   | 7.7r/46.5w   | 0.4r/1.8w    |
|    /W2k12-SQL2k12_1.vmdk | 0.0r/7.7w   | 0.4r/1236.7w | 0.8r/1.8w    |
|    /W2k12-SQL2k12_2.vmdk | 0.4r/647.6w | 1.6r/4603.3w | 1.3r/1.8w    |
+--------------------------+-------------+--------------+--------------+
```

By way of understanding the relationship between the metrics, the following calculations may help:

- IOPS = (MBps Throughput / KB per IO) * 1024
- MBps = (IOPS * KB per IO) / 1024

# Using VSAN Observer for Performance Monitoring

One should take the time to familiarize oneself with VSAN Observer, even when there are no anomalies occurring on the Virtual SAN cluster. Check what is a normal value for metrics like latency, IOPS, Outstanding IO, bandwidth, etc. It may even be worth gathering offline bundles from time to time to use as reference against possible future issues.

As mentioned in the VSAN Observer section, any metrics that exceed boundary thresholds will be displayed with a red underline against the chart. This can then be used as a point to start investigating the root cause. Various reasons why some of these charts may be outside their respective thresholds have been discussed, but anything from an overloaded system to poor performing devices to physical switch issues impacting the network could be a root cause. Fortunately, VSAN Observer can assist with all of this.

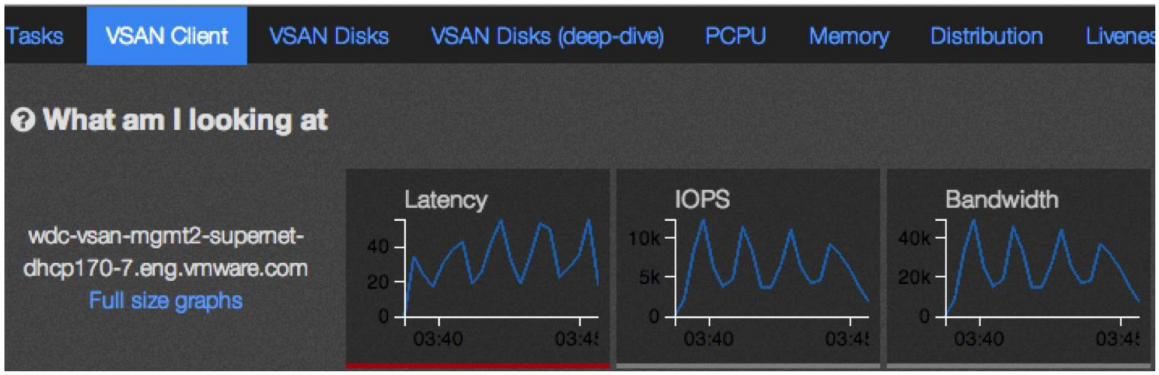
Next some VSAN Observer case studies are examined, where there are details on what to expect from VSAN Observer when some common scenarios are encountered.

# 17. VSAN Observer case studies

## I – Using VSAN Observer to troubleshoot high latency

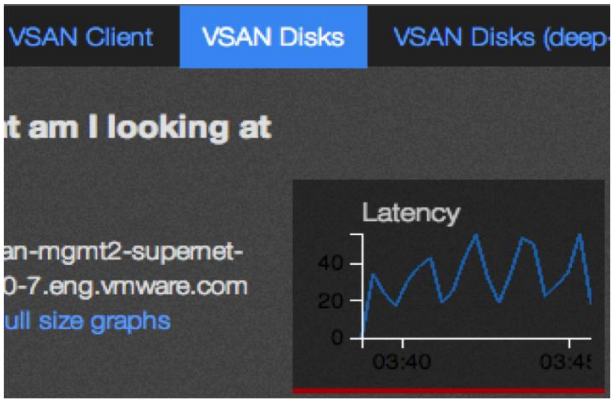
In this case study, a user is complaining of slow response times on the virtual machines running on Virtual SAN. Let's see if we can determine the root cause by using VSAN Observer.

On launching VSAN Observer, the first thing noticed is that the VSAN Client view shows that the "Latency" is highlighted as an issue as it has a red underline:




A close look shows that there is high latency of 20-40ms in the VSAN Client. Latency shown in this view is basically the latency seen by the VM accessing Virtual SAN storage.

The next step is to find where this high latency is being incurred. Remember that a virtual machine's storage does not need to be on the same host as the compute, and with *NumerOfFailuresToTolerate* = 1 or more, there are definitely remote mirror copies of the VM's disk on different hosts. So latency could possible be incurred on the network. Let's rule that out next by looking at the 'VSAN Disks' view:

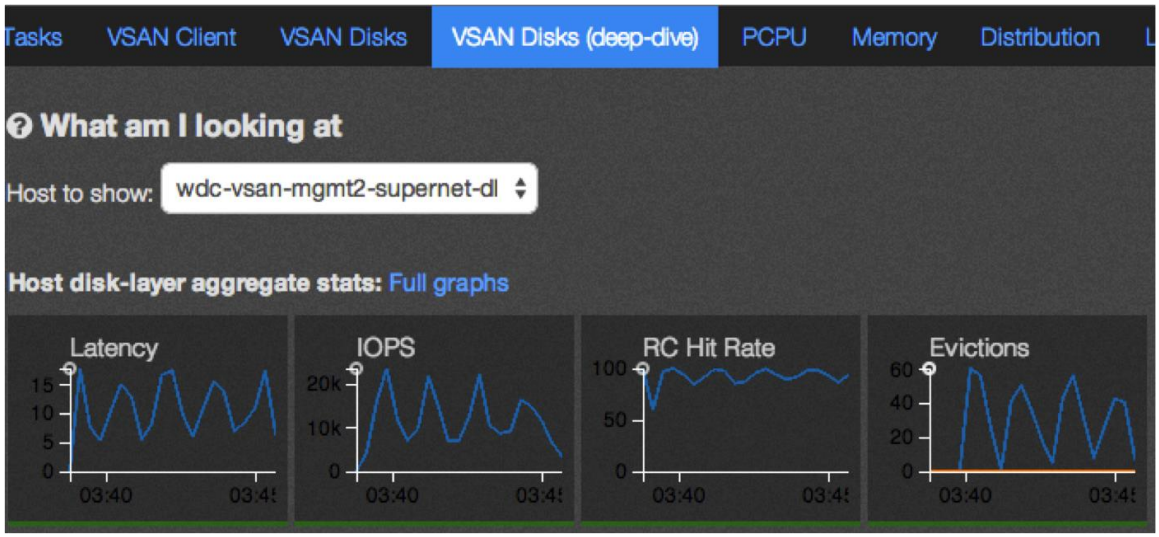


As we can see, the VSAN Disks view also displays the same high latency observed in the VSAN Client view. This view is the physical disks view, and is basically representative of the latency seen by Virtual SAN accessing physical disks. In a nutshell, this rules out the Virtual SAN network as the cause of the latency issue.

Another cause of latency could be CPU. Perhaps the CPUs are maxed out, and not able to process I/O quickly enough. We can check the CPU utilization via 'PCPU' view. It shows the following:


The ALL PCUs chart is green, and overall the CPU utilization is not very high. Also, all of the Virtual SAN "worldlets" are showing green and none of them are maxed out. The impression from this view is that there are no CPU issues, so this also rules out CPU as being the root cause of the latency issue.

This is now beginning to appear like a disk issue. Let's go back and look at the disks in more detail by visiting the 'VSAN Disks (deep dive)' view in VSAN Observer. One possibility is that we are getting a lot of flash read cache misses when requesting a block of data, resulting in the data blocks having to be fetched from magnetic disk. The VSAN Disks (deep dive) view can help us check that. It shows us the following:

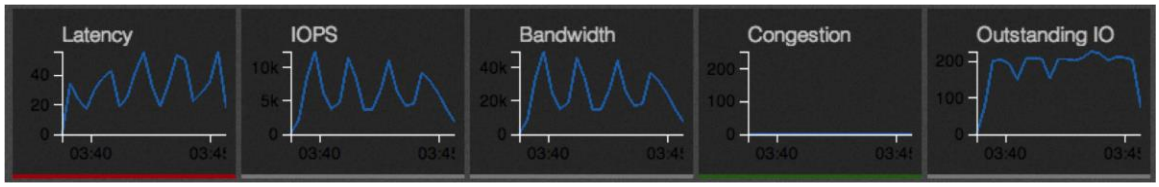



Well, it appears that we have reasonable latency at physical disk layer, between 5-15ms. This means that the magnetic disks are behaving reasonably well, and within acceptable limits. Note that the latency here is green. This implies that any destaging operations from flash to magnetic disk, as well as any read cache misses that need to fetch data blocks from magnetic disks, are not unduly impacting performance.

There is also a reasonably high read cache hit rate, which is near 100%. In other words, the flash layer is servicing pretty much all of the read requests. This basically rules out read cache misses causing latency issues.

This view basically shows us that physical magnetic disks are behaving absolutely 'fine'.

At this point, we have ruled out the network, the CPUs and the physical magnetic disk layer. The only remaining item to check is the flash layer. Let's go back to 'VSAN Disks' for a closer look:



On closer inspection, it seems that Outstanding I/O is very high with values of around 200. Let's remind ourselves again what Outstanding I/O actually is. When a virtual machine requests for certain I/O to be performed (reads or writes), these requests are sent to storage devices. Until these requests are complete they are termed outstanding I/O.

Let's recap: There are virtual machines sending I/Os that are not being acknowledged promptly by the flash layer. Large amounts of outstanding I/O could have an adverse effect on the device latency. When we look at the Outstanding I/O, we can see that this is the cause of the latency. The question now is why is it taking so long to acknowledge the I/Os? What is causing this outstanding I/O – that is the real question?

To be able to prioritize traffic classes (e.g. VM I/O, resynchronization I/O for component rebuilds, etc.), Virtual SAN runs a "scheduler" at VSAN Disks level. This scheduler means that the physical devices should only ever see a 'reasonable' amount of outstanding I/O to keep latency low. So why are we seeing such high outstanding I/O numbers here resulting in high latencies?

The root cause of this performance issue is that the SSD used as the flash cache device in this example was NOT on the VMware Compatibility Guide. It was an older, slower generation solid-state disk.

When we examined the specification sheet for this SSD, it tells us that it can do around 4-6k IOPS, but at 4 Outstanding I/O.

- 4K IOPS – 1 x I/O will take .25ms to complete
- 6K IOPS – 1 x I/O will take .175ms to complete

By extrapolating this out, what it basically means is that a batch of 4 x I/O takes between 0.7ms and 1.0ms to complete. Therefore if there are 200 Outstanding I/O in the VSAN disks view of VSAN observer, some further extrapolation reveals the following:

- 200 Outstanding I/O = 50 x (4 Outstanding I/O)
    - If 4 x I/O produces 0.7ms to 1.0ms latency
    - Then 50 times (4 x I/O) produces 50 x (0.7ms to 1.0ms) latency
- This gives a latency value of somewhere between 35ms to 50ms

This is pretty much the latency we are observing.

**Root cause**: The virtual machine workload currently running on this Virtual SAN deployment has pushed the SSD to its operational limits!

**Lesson**: Ensure you correctly size your flash device for your Virtual SAN workload.

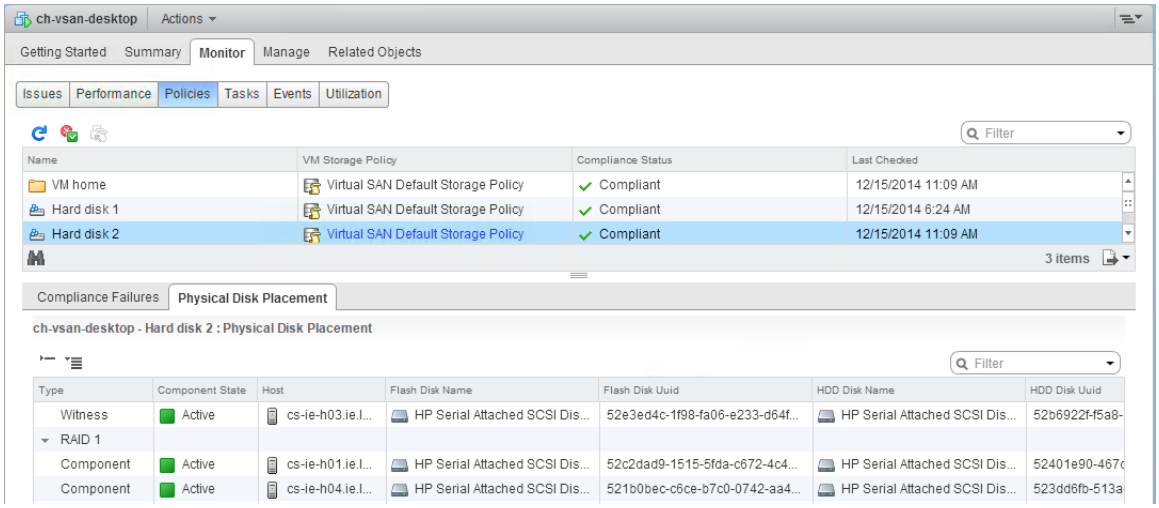
# II – Using VSAN Observer to troubleshoot a network performance issue

As Virtual SAN is a scale-out storage solution, performance of the network interconnects is also a key factor in validating performance. There have been a number of situations where customers have observed high latency during a test run. The next step is obviously to identify and then isolate the source of the latency.

An easy way to figure out whether the source of latency in a performance test run is potentially the network is to compare latency figures seen at the VSAN Client tab and the VSAN disk tab. If the latency at the client tab is much higher than what is seen at the disk tab, the network could potentially be a source of latency, and you should validate that your network configuration is appropriate.

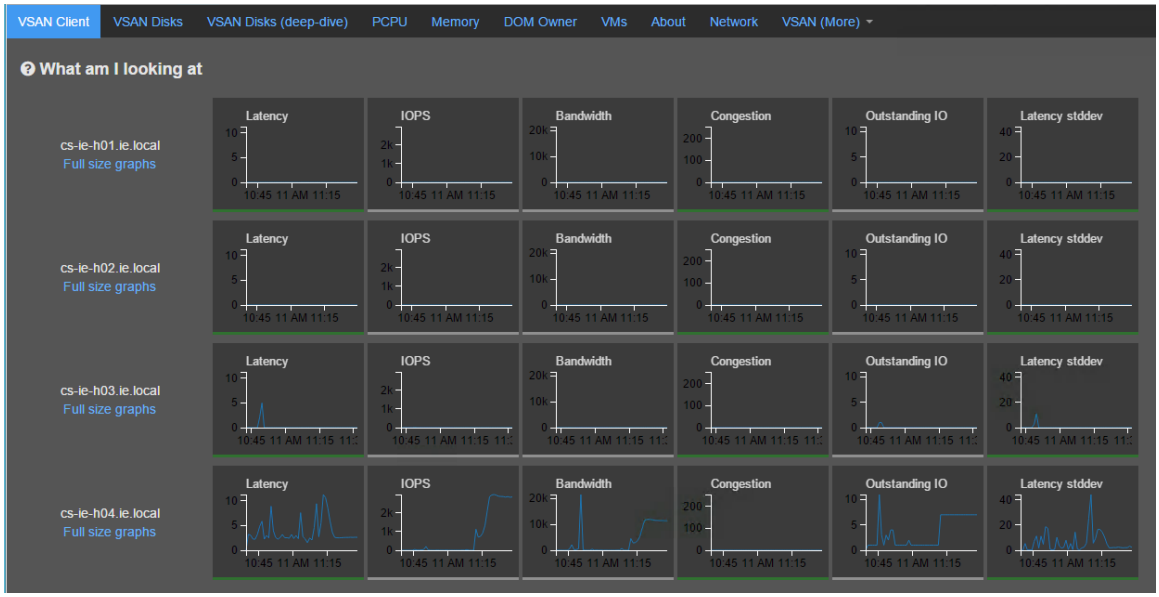
In this example, there is a virtual machine running a particular workload. In this case, it is IOmeter, running a 70% read workload, 70% random. The Outstanding IO value is set to 8, and there is only one worker thread. The *NumberOfFailuresToTolerate* has been set to 1. The virtual machine is producing a nice steady 2,000 IOPS giving us a throughput value of 8MB per second.

The host is deployed on host cs-ie-h04. The VMDK object that is the destination of the IOmeter workload is mirrored across host's cs-ie-h01 and cs-ie-h04.

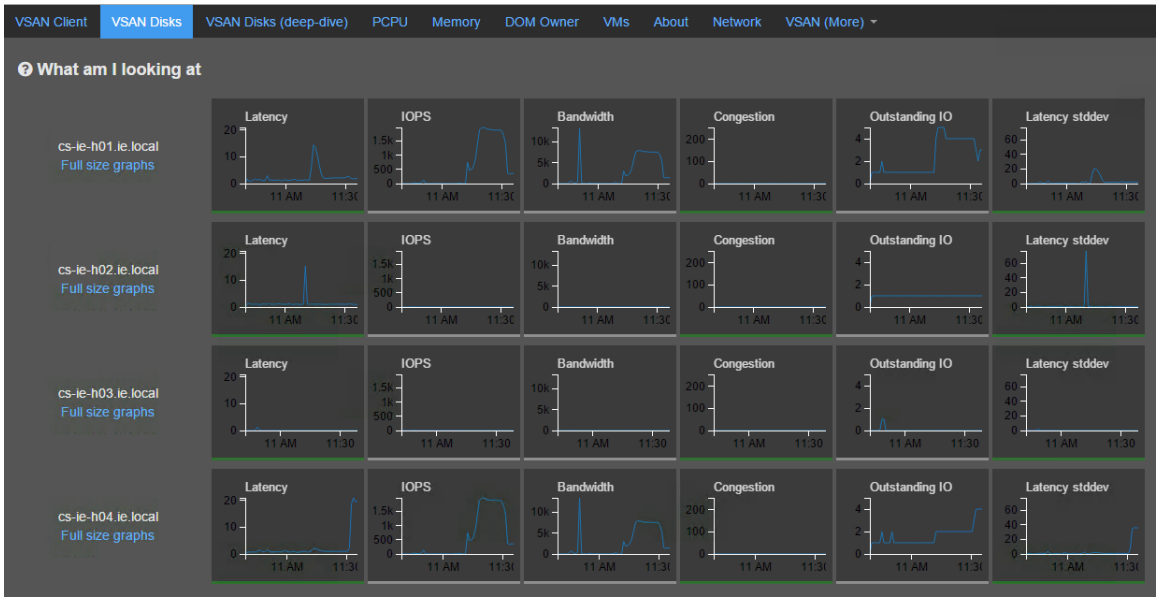



Let's begin by taking a look at performance in steady state.

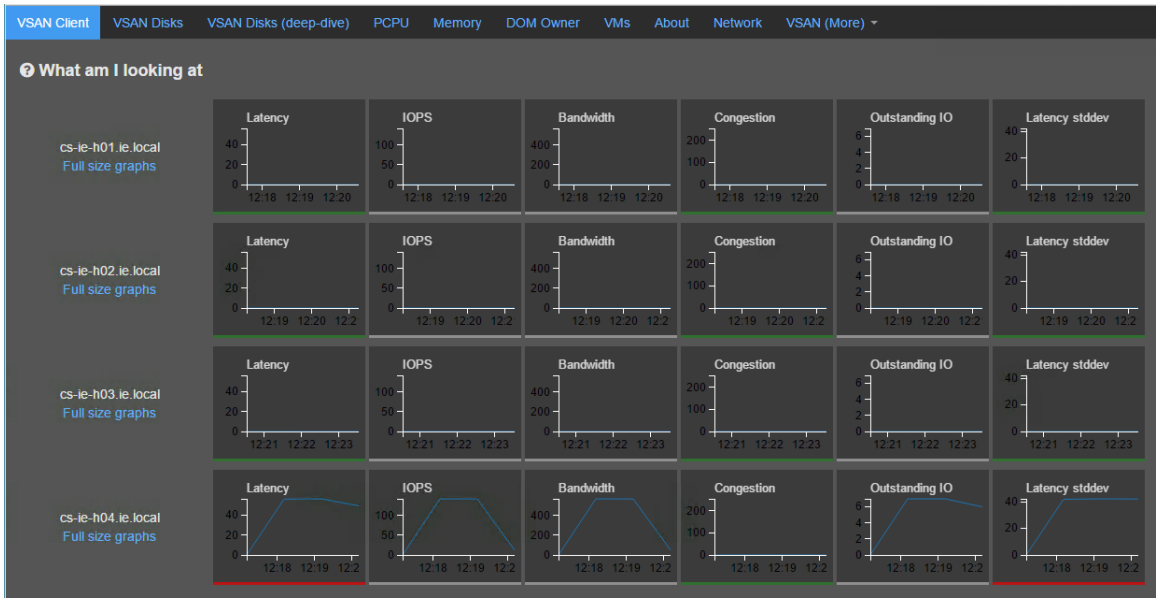
First, the VSAN Client view is showing activity for host cs-ie-04 only, as this is the host where the virtual machine with the workload is running. In this environment, there is no other activity currently. As you can see, all metrics are green. No anomalies have been detected.



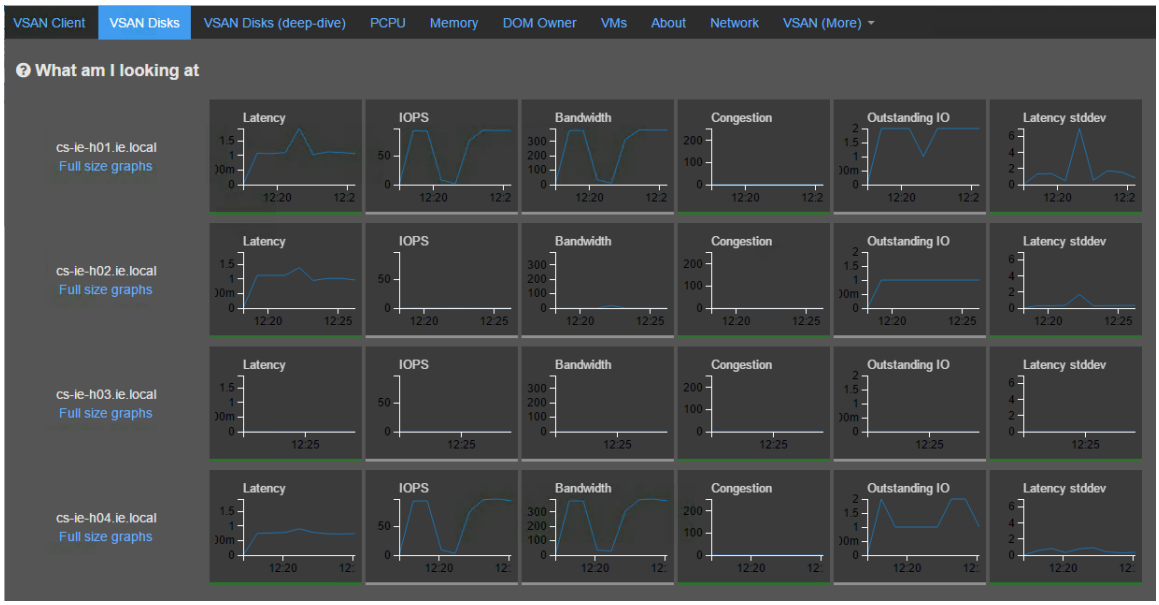

Next, let's take a look at the VSAN Disks view. This shows that most of the activity is on host cs-ie-h01 and cs-ie-h04, which is to be expected. This is where the disk components of the VMDK that are used for workload testing reside. Latency is well within limits, and IOPS and bandwidth match what IOmeter is delivering. Also of note is the fact that congestion is normal, as the metric is shown in green.



Now the problem manifests itself. The customer notices that IOPS and throughput have begun to reduce significantly, while latency has risen. Notice that latency has now been flagged as red on cs-ie-h04.




The next place to check is the VSAN Disks view, and see whether latency is occurring at the magnetic disk or flash layers.



There are clearly no latency issues here. However note again the significant drop in IOPS from approximately 2,000 in steady state to around 300 now. Perhaps another clue as to what is happening here? Why would Virtual SAN be driving only 20% of the number of IOPS that it was previously able to drive with this workload?

One final place to check is the DOM Owner view. This will tell us whether VSAN is seeing any latency at this layer:

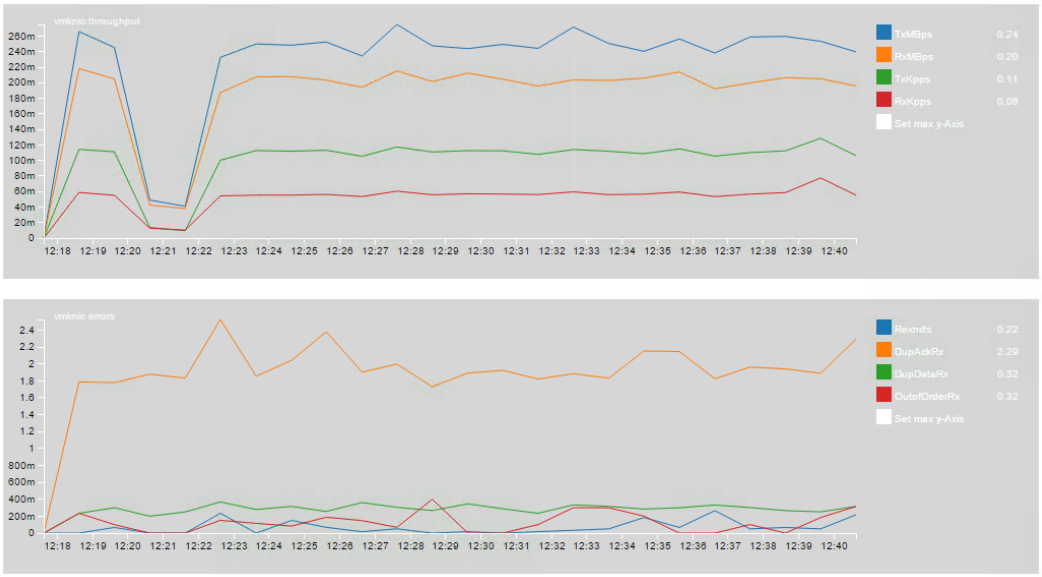

The answer is yes; we are also seeing latency at the DOM owner layer, the layer that is responsible for managing RAID configurations.

Let's recap on what we have observed. The client, in essence the virtual machine, is seeing high latency. The owner, responsible for providing network access to the storage, is seeing latency across the distributed components of the object. However the physical disk layer is no seeing any latency issues. This might now be a network issue, and it is now worth visiting the network view to see if there are any errors or issues with the network.

The default view doesn't offer much information other than telling us how much the physical NIC has transmitted and received and that there are no serious errors with the physical NICs.

If we go to the full graph view in networking, and especially the vmknic view (this is the Virtual SAN network), we see something interesting. Again, this virtual machine has a VM Storage policy of *NumberOfFailuresToTolerate* set to 1, meaning that its storage components are replicated across two separate hosts in the Virtual SAN cluster.

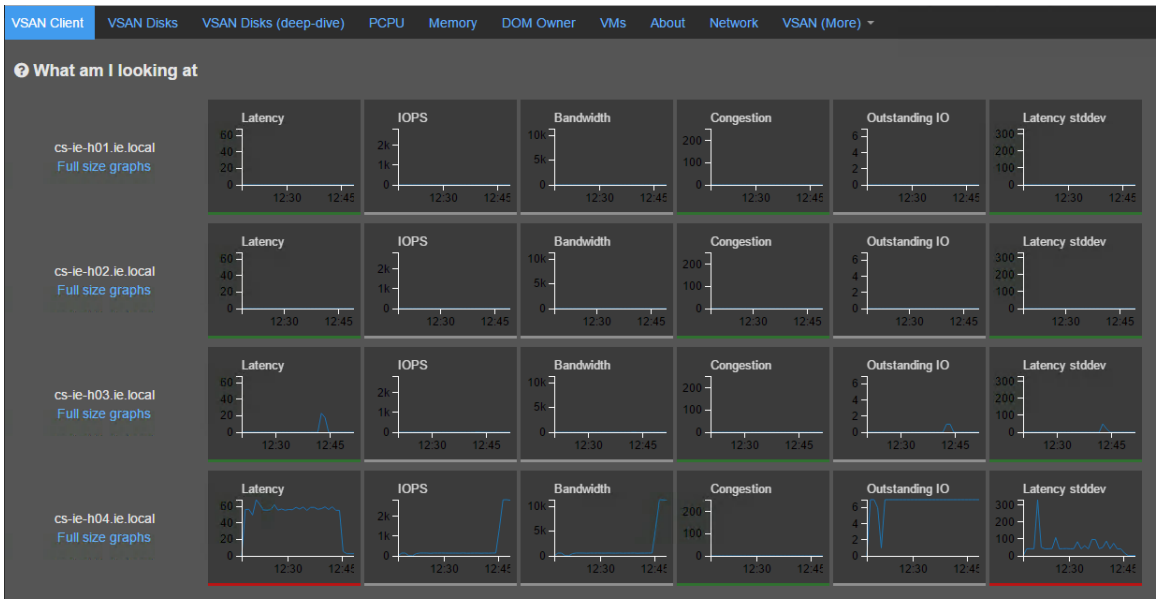
Once the full charts view is expanded, what is notable is that the amount of data transmitted and received on the Virtual SAN network is rather small:




This is taken from the host where the virtual machine resides. These hosts use 1Gb NICs, which are fully supported by Virtual SAN, but should be dedicated to Virtual SAN traffic. So why is the TxMBps and RxMBps so low when we are clearly wishing to drive far more IOPS than we are achieving?

In this scenario, using the Network I/O Control feature of a distributed switch to limit the Virtual SAN network traffic produced this situation. Although somewhat contrived, it is an accurate reflection of the behavior that may be observed in Virtual SAN if the network is suffering from issues. In fact, we had a customer situation where the switch was sending excessive pause frames (in the millions) impacting Virtual SAN traffic and leading to extremely poor latency for the virtual machine workloads (observed via the client and DOM owner views) but no latency issues on the disks view.

When the NIOC limit is lifted from the Virtual SAN traffic, we very quickly begin to see latency, IOPS and bandwidth return to expected values:




**Root cause**: The Virtual SAN network was limited in the number of MB per second that it could send, impacting IOPS and latency.

**Lesson**: Virtual SAN is a distributed storage system that relies heavily on the network between each of the participating hosts. Ensure your network is fully functional. Issues on the network will significantly impact performance.

# 18. Engaging with Global Support Services

If you are unable to root-cause an issue with a Virtual SAN deployment, be it a failure of some components or a performance issue, always consider reaching out to VMware's Global Support Services organization for additional assistance. This section of the troubleshooting reference manual will give some details on how to be prepared when logging a Service Request. The more information that can be provided to the technical support engineer, the sooner a resolution will be found.

## Ensure scratch and log files are on persistent media

Virtual SAN supports ESXi booting from USB, SD and, in Virtual SAN 6.0, SATADOM. However when booting from these devices, the scratch partition that is used for storing logs is placed on RAM disk which means that they are not persisted if the host is rebooted. The *scratch* partition is used for storing log files, such as `/var/log/vmkernel`. A best practice recommendation is to place these logs on persistent storage. VMware KB article 1033696 has details on how to redirect scratch to a persistent datastore.

It is recommended that you do not use the VSAN datastore for persisting logs. If you have configured your ESXi hosts to use the Virtual SAN datastore for the *scratch* partition, then any issues that impact the Virtual SAN datastore will also impact the ESXi hosts, as the host will be unable to save logs. It also means that previous logs might not be accessible. This will make troubleshooting and diagnosis of the underlying issue extremely difficult.

Do not use the Virtual SAN datastore as a scratch partition for ESXi hosts. Ideally use a persistent datastore, such as a VMFS volume or NFS volume, for persisting log files. Another best practice is to use a syslog server, which is discussed shortly.

## Ensure VSAN Trace files are on persistent media

VSAN trace files should be stored on persistent storage. If your VSAN trace files are not currently on persistent storage (e.g. when using USB/SD/SATADOM boot media, which places these traces on a RAM Disk), you can use following command to redirect the traces to persistent media.

```
# esxcli vsan trace set  --help
Usage: esxcli vsan trace set [cmd options]
Description:
 set                   Configure VSAN trace. Please note: This command is not
thread safe.
Cmd options:
 -f|--numfiles=<long>  Log file rotation for VSAN trace files.
 -p|--path=<str>       Path to store VSAN trace files.
 -r|--reset            When set to true, reset defaults for VSAN trace files.
 -s|--size=<long>      Maximum size of VSAN trace files in MB.
```

## Netdump

The VMware ESXi Dump Collector service is used to define a directory on the vCenter server that can be used for storing received core dumps over the network from ESXi hosts. In many deployments, the only storage available to ESXi hosts in a VSAN Cluster may be the VSAN datastore. Configuring a netdump service to store dump files on the vCenter server rather than the VSAN datastore can assist troubleshooting efforts should the VSAN datastore become inaccessible.

Options such as setting a directory, the amount of disk space used, and the listening port number can be set during installation. However, all options can be changed later by editing the configuration file. VMware KB article 2002954 has further details on how to set up the netdump service.

## Syslog server

ESXi hosts run a syslog service (`vmsyslogd`) that provides a standard mechanism for logging messages from the VMkernel and other system components. By default in ESXi, these logs are placed on a local scratch volume or a ramdisk. To preserve the logs further, ESXi can be configured to place these logs to an alternate storage location on disk, and to send the logs across the network to a syslog server. This is a recommended best practice for ESXi hosts, and VMware KB article 2003322 has further details on how to configure it. Configuring a syslog server to store log files on a remote syslog server rather than a local scratch volume or ramdisk can assist troubleshooting efforts, especially root-cause-analysis.

## Filing a support request with VMware

To file a support request with VMware, the following knowledge base article provides instructions on how to do so: http://kb.vmware.com/kb/2006985

## Which logs should be gathered?

VMware's technical support engineers will look for a certain set of files in order to be able to investigate your issue. This knowledgebase articles gives details on which logs to gather: http://kb.vmware.com/kb/2072796

Since Virtual SAN comprises of multiple ESXi hosts, you will need to capture **vm-support** output from all ESXi hosts in the cluster. Certain Virtual SAN logs are captured as trace files by the vm-support scripts. The utility *vsanTraceReader* is also included in the vm-support, which allows the trace files to be read. You will find it in `/usr/lib/vmware/vsan/bin/vsanTraceReader`.

To dump a VSAN trace into a human readable format, you can use a command similar to the following:

```
/usr/lib/vmware/vsan/bin/vsanTraceReader vsantraces--2014-12-04T03h07m38s822.gz > trace
```

However note that this information is only useful to VMware's support organization. Also considering capturing vCenter Server logs via **vc-support**. These will be critical for examining Storage Policy Based Management (SPBM) type issues and also Virtual SAN storage provider (VASA) registration issues. Finally, we may need to capture some RVC command outputs from your RVC server (typically your vCenter server). Here is a useful RVC command to capture the `vsan.support_information` to a file:

```
/usr/bin/rvc -c "vsan.support_information 1" -c "quit"
administrator@vsphere.local:password@hostname > log_location
```

If the host and password are not provided on the command line, you will be prompted for it.

VMware KB Article 2064240 provides useful VSAN Observer log gathering information.

## How to transfer log files to VMware?

Once the log bundles have been generated, they need to be sent to VMware Technical Support. The following KB article provides a variety of ways of getting the logs to VMware: http://kb.vmware.com/kb/1008525

# Appendix A: Useful HP Utilities

## hpssacli utility

While this utility does not come by default on ESXi hosts, it does come pre-bundled if you deploy your ESXi hosts with the ESXi 5.5 U2 HP OEM image that contains, among other things, the `hpssacli` utility. This tool can be extremely useful for monitoring and troubleshooting issues with your HP storage controller and disk devices. While this appendix is not all-inclusive, it does contain some very useful commands that a Virtual SAN and vSphere administrator could leverage.

For a complete list of commands and their usage, please refer to the official HP documentation on the topic:

http://h10032.www1.hp.com/ctg/Manual/c04123772.pdf

Let's begin by checking to see if the appropriate software is installed. Here is a list of all HP VIBs that come pre-installed on their OEM image:

```
~ # esxcli software vib list | grep -i Hewlett
char-hpcru                     5.5.6.6-1OEM.550.0.0.1198610          Hewlett-Packard  PartnerSupported  2014-09-16
char-hpilo                     550.9.0.2.3-1OEM.550.0.0.1198610      Hewlett-Packard  PartnerSupported  2014-09-16
hp-ams                         550.10.0.0-18.1198610                 Hewlett-Packard  PartnerSupported  2014-09-16
hp-build                       5.76.36-1198610                       Hewlett-Packard  PartnerSupported  2014-09-16
hp-conrep                      5.5.0.1-0.0.7.1198610                 Hewlett-Packard  PartnerSupported  2014-09-16
hp-esxi-fc-enablement          550.2.1.8-1198610                     Hewlett-Packard  PartnerSupported  2014-09-16
hp-smx-provider                550.03.06.00.23-1198610               Hewlett-Packard  VMwareAccepted    2014-09-16
hpbootcfg                      5.5.0.02-00.00.11.1198610             Hewlett-Packard  PartnerSupported  2014-09-16
hpnmi                          550.2.3.5-1198610                     Hewlett-Packard  PartnerSupported  2014-09-16
hponcfg                        5.5.0.4.4-0.3.1198610                 Hewlett-Packard  PartnerSupported  2014-09-16
hpssacli                       2.0.23.0-5.5.0.1198610                Hewlett-Packard  PartnerSupported  2014-09-16
hptestevent                    5.5.0.01-00.01.4.1198610              Hewlett-Packard  PartnerSupported  2014-09-16
scsi-hpdsa                     5.5.0.12-1OEM.550.0.0.1331820         Hewlett-Packard  PartnerSupported  2014-09-16
scsi-hpsa                      5.5.0.74-1OEM.550.0.0.1331820         Hewlett-Packard  VMwareCertified   2014-09-16
scsi-hpvsa                     5.5.0-88OEM.550.0.0.1331820           Hewlett-Packard  PartnerSupported  2014-09-16
```

So there are quite a few VIBs installed as one can clearly see. The VIBs that we are interested in are the ones that add additional namespaces to ESXCLI on the ESXi hosts. By simply running the esxcli command, we can see some new namespaces that we would typically not find on an ESXi host built from the standard VMware images.

```
~ # esxcli
Usage: esxcli [options] {namespace}+ {cmd} [cmd options]

Options:
  --formatter=FORMATTER
                        Override the formatter to use for a given command. ...
  --debug               Enable debug or internal use options
  --version             Display version information for the script
  -?, --help            Display usage information for the script

Available Namespaces:
  device                Device manager commands
  esxcli                Commands that operate on the esxcli system ...
  fcoe                  VMware FCOE commands.
  fio                   Fusion-io utility CLI
  graphics              VMware graphics commands.
  hardware              VMKernel hardware properties and commands for configuring ...
  hpbootcfg             HP Boot Configuration Utility
  hpssacli              HP Smart Storage Administrator CLI
  hptestevent           HP Test Event Utility
  iscsi                 VMware iSCSI commands.
  network               Operations that pertain to the maintenance of networking on ...
  sched                 VMKernel system properties and commands for configuring ...
  software              Manage the ESXi software image and packages
  storage               VMware storage commands.
  system                VMKernel system properties and commands for configuring ...
  vm                    A small number of operations that allow a user to Control ...
  vsan                  VMware VSAN commands.
```

The hpssacli, Smart Storage Administrator CLI, is the one we are particularly interested in:

```
~ # esxcli hpssacli
Usage: esxcli hpssacli {cmd} [cmd options]

Available Commands:
  cmd                   hpssacli command with options parameter.
```

```
~ # esxcli hpssacli cmd -q help

CLI Syntax
   A typical HPSSACLI command line consists of three parts: a target device,
   a command, and a parameter with values if necessary. Using angle brackets to
   denote a required variable and plain brackets to denote an optional
   variable, the structure of a typical HPSSACLI command line is as follows:

      <target> <command> [parameter=value]

   <target> is of format:
      [controller all|slot=#|serialnumber=#]
      [array all|<id>]
      [physicaldrive all|allunassigned|[#:]#:#|[#:]#:#-[#:]#:#]
      [ssdphysicaldrive all|allunassigned|[#:]#:#|[#:]#:#-[#:]#:#]
      [logicaldrive all|#]
      [enclosure all|#:#|serialnumber=#]
      [licensekey all|<key>]
      [ssdinfo]
      Note 1: The #:#:# syntax is only needed for systems that
              specify port:box:bay. Other physical drive targeting
              schemes are box:bay and port:id.

   Example targets:
      controller slot=5
      controller serialnumber=P21DA2322S
      controller slot=7 array A
      controller slot=5 logicaldrive 5
      controller slot=5 physicaldrive 1:5
```

```
      controller slot=5 physicaldrive 1E:2:3
      controller slot=5 ssdphysicaldrive all
      controller slot=5 enclosure 4E:1 show
      controller slot=5 licensekey XXXXX-XXXXX-XXXXX-XXXXX-XXXXX

For detailed command information type any of the following:
   help add
   help create
   help delete
   help diag
   help modify
   help remove
   help shorthand
   help show
   help target
   help rescan
   help version

Encryption related commands:
   help ctrlpasswdstate
   help clearencryptionconfig
   help encode
   help encrypt
   help enableencryption
   help encryption
   help eula
   help fwlock
   help import
   help instantsecureerase
   help localkeymanagermode
   help login
   help logout
   help recoverpassword
   help rekey
   help remotekey
   help removectrlpasswd
   help rescankeys
   help setmasterkeycache
   help setctrlpasswd
   help setpasswd
   help setrecoveryparams
   help unlockvolumes

Help also accepts commonly used CLI parameters and HPSSA keywords. Adding
additional keywords will further filter the help output. Examples:
   help migrate
   help expand
   help extend
   help <keyword> <keyword> ... <keyword>
```

**~ # esxcli hpssacli cmd -q "help add"**

```
<target> add [drives=[#:]#:#,[#:]#:#,[#:]#:#-[#:]#:#],...|allunassigned]
             [spares=[#:]#:#,[#:]#:#,[#:]#:#-[#:]#:#],...|allunassigned]
             [sparetype=dedicated|autoreplace]
             [modifyparitygroups=yes|no][forced]
   The add command adds a physical drive or spare to the specified array or
   logical drive. Adding physical drives is the same as expanding an array. The
   forced parameter represses warning message prompts.
   The sparetype keyword allows the user to specify the spare type as dedicated
   or autoreplace. Defaults to dedicated. A dedicated spare temporarily takes
   over for a failed drive and can be shared between arrays. An autoreplace
   spare replaces a failed drive and cannot be shared between arrays.
    The target can be any valid individual array or logical drive target.

<target> add [licensekey=XXXXX-XXXXX-XXXXX-XXXXX-XXXXX]
   The add command adds a license key to the specified array controller. The
   target can be any valid controller.
```

```
Examples:
   controller slot=5 array A add drives=1:1-1:5
   controller slot=5 array A add drives=1I:1:1,1I:1:2,1I:1:3
   controller slot=5 array A add spares=allunassigned sparetype=autoreplace
   controller slot=5 array A add spares=2e:1:5 spareactivationmode=predictive
   ctrl slot=5 logicaldrive 1 add drives=1:8
   controller slot=5 array A add drives=1I:1:1,1I:1:2,1I:1:3
   modifyparitygroups=yes
controller slot=1 array A add spares=1e:1:5
   sparetype=dedicated
controller slot=1 array A add spares=1e:1:5
   sparetype=autoreplace
   controller slot=5 add licensekey=XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
   controller slot=5 add lk=XXXXXXXXXXXXXXXXXXXXXXXXX
```

Now, this is not altogether intuitive to use so I recommend referring to **HP Configuring Arrays on HP Smart Array Controllers Reference Guide** to find the correct command syntax. The information provided from these commands may help you to determine if there are any underlying storage issues, and whether or not your controller has been configured optimally for Virtual SAN.

# Controller & Disk Drive Information via hpssacli

Let's begin by getting a little bit more detail from the controllers. The first command here will just give us a brief description of the controller.

### esxcli hpssacli cmd -q "controller all show"

```
~ # esxcli hpssacli cmd -q "controller all show"

Smart Array P410i in Slot 0 (Embedded)    (sn: 5001438013072F30)
```

Let's run one more command to get the status of the controller only.

### esxcli hpssacli cmd -q "controller all show status"

```
~ # esxcli hpssacli cmd -q "controller all show status"

Smart Array P410i in Slot 0 (Embedded)
   Controller Status: OK
   Cache Status: OK
   Battery/Capacitor Status: OK
```

This next command provides a lot more detail. In particular, is the driver write cache disabled on the controller? As we can clearly seem it is.

### esxcli hpssacli cmd -q "controller slot=0 show"

```
~ # esxcli hpssacli cmd -q "controller slot=0 show"

Smart Array P410i in Slot 0 (Embedded)
   Bus Interface: PCI
   Slot: 0
   Serial Number: 5001438013072F30
   Cache Serial Number: PBCDF0CRH0EIUW
   RAID 6 (ADG) Status: Disabled
   Controller Status: OK
   Hardware Revision: C
   Firmware Version: 3.66
   Rebuild Priority: Medium
   Expand Priority: Medium
   Surface Scan Delay: 15 secs
   Surface Scan Mode: Idle
   Queue Depth: Automatic
   Monitor and Performance Delay: 60  min
   Elevator Sort: Enabled
   Degraded Performance Optimization: Disabled
   Inconsistency Repair Policy: Disabled
   Wait for Cache Room: Disabled
   Surface Analysis Inconsistency Notification: Disabled
   Post Prompt Timeout: 0 secs
   Cache Board Present: True
   Cache Status: OK
```

```
   Cache Ratio: 25% Read / 75% Write
   Drive Write Cache: Disabled
   Total Cache Size: 1024 MB
   Total Cache Memory Available: 912 MB
   No-Battery Write Cache: Disabled
   Cache Backup Power Source: Capacitors
   Battery/Capacitor Count: 1
   Battery/Capacitor Status: OK
   SATA NCQ Supported: True
   Number of Ports: 2 Internal only
   Encryption Supported: False
   Driver Name: HP HPSA
   Driver Version: 5.5.0
   Driver Supports HP SSD Smart Path: False
```

Next we can begin to get some additional information about the physical drives that are installed. Indeed, one can also retrieve information like firmware version from the controller too, and there is also information about a HP feature called SSD Smart Path. This is disabled in this setup as there are no SSDs attached to this controller. However in the case where there are SSDs attached to the controller and in use by Virtual SAN, this feature should also be left **disabled**. This is true for other acceleration mechanisms from other controller vendors also, e.g. LSI Fast Path.

### esxcli hpssacli cmd -q "controller slot=0 show config detail"

```
~ # esxcli hpssacli cmd -q "controller slot=0 show config detail"

Smart Array P410i in Slot 0 (Embedded)
   Bus Interface: PCI
   Slot: 0
   Serial Number: 5001438013072F30
   Cache Serial Number: PBCDF0CRH0EIUW
   RAID 6 (ADG) Status: Disabled
   Controller Status: OK
   Hardware Revision: C
   Firmware Version: 3.66
   Rebuild Priority: Medium
   Expand Priority: Medium
   Surface Scan Delay: 15 secs
   Surface Scan Mode: Idle
   Queue Depth: Automatic
   Monitor and Performance Delay: 60  min
   Elevator Sort: Enabled
   Degraded Performance Optimization: Disabled
   Inconsistency Repair Policy: Disabled
   Wait for Cache Room: Disabled
   Surface Analysis Inconsistency Notification: Disabled
   Post Prompt Timeout: 0 secs
   Cache Board Present: True
   Cache Status: OK
   Cache Ratio: 25% Read / 75% Write
   Drive Write Cache: Disabled
   Total Cache Size: 1024 MB
   Total Cache Memory Available: 912 MB
   No-Battery Write Cache: Disabled
   Cache Backup Power Source: Capacitors
   Battery/Capacitor Count: 1
   Battery/Capacitor Status: OK
   SATA NCQ Supported: True
   Number of Ports: 2 Internal only
   Encryption Supported: False
   Driver Name: HP HPSA
   Driver Version: 5.5.0
   Driver Supports HP SSD Smart Path: False
```

```
Internal Drive Cage at Port 1I, Box 1, OK
   Power Supply Status: Not Redundant
   Serial Number:
   Drive Bays: 4
   Port: 1I
   Box: 1
   Location: Internal

Physical Drives
   physicaldrive 1I:1:1 (port 1I:box 1:bay 1, SAS, 146 GB, OK)
   physicaldrive 1I:1:2 (port 1I:box 1:bay 2, SAS, 146 GB, OK)
   physicaldrive 1I:1:3 (port 1I:box 1:bay 3, SAS, 146 GB, OK)
   physicaldrive 1I:1:4 (port 1I:box 1:bay 4, SAS, 146 GB, OK)


Internal Drive Cage at Port 2I, Box 1, OK
   Power Supply Status: Not Redundant
   Serial Number:
   Drive Bays: 4
   Port: 2I
   Box: 1
   Location: Internal

Physical Drives
   physicaldrive 2I:1:5 (port 2I:box 1:bay 5, SAS, 146 GB, OK)
   physicaldrive 2I:1:6 (port 2I:box 1:bay 6, SAS, 146 GB, OK)
   physicaldrive 2I:1:7 (port 2I:box 1:bay 7, SAS, 146 GB, OK)
   physicaldrive 2I:1:8 (port 2I:box 1:bay 8, SAS, 146 GB, OK)

Array: A
   Interface Type: SAS
   Unused Space: 0  MB
   Status: OK
   Array Type: Data


   Logical Drive: 1
      Size: 136.7 GB
      Fault Tolerance: 0
      Heads: 255
      Sectors Per Track: 32
      Cylinders: 35132
      Strip Size: 256 KB
      Full Stripe Size: 256 KB
      Status: OK
      Caching:  Enabled
      Unique Identifier: 600508B1001C3EA7838C0436DBE6D7A2
      Logical Drive Label: A00D1A285001438013072F305019
      Drive Type: Data
      LD Acceleration Method: Controller Cache

   physicaldrive 1I:1:1
      Port: 1I
      Box: 1
      Bay: 1
      Status: OK
      Drive Type: Data Drive
      Interface Type: SAS
      Size: 146 GB
      Native Block Size: 512
      Rotational Speed: 10000
      Firmware Revision: HPD8
      Serial Number:         PEW6N8HF
      Model: HP      DG0146FARVU
      Current Temperature (C): 30
      Maximum Temperature (C): 38
      PHY Count: 2
      PHY Transfer Rate: 6.0Gbps, Unknown
```

```
Array: B
   Interface Type: SAS
   Unused Space: 0  MB
   Status: OK
   Array Type: Data


   Logical Drive: 2
      Size: 136.7 GB
      Fault Tolerance: 0
      Heads: 255
      Sectors Per Track: 32
      Cylinders: 35132
      Strip Size: 256 KB
      Full Stripe Size: 256 KB
      Status: OK
      Caching:  Enabled
      Unique Identifier: 600508B1001C2EE9A6446E708105054B
      Logical Drive Label: A00D0E295001438013072F302E92
      Drive Type: Data
      LD Acceleration Method: Controller Cache

   physicaldrive 1I:1:2
      Port: 1I
      Box: 1
      Bay: 2
      Status: OK
      Drive Type: Data Drive
      Interface Type: SAS
      Size: 146 GB
      Native Block Size: 512
      Rotational Speed: 10000
      Firmware Revision: HPD8
      Serial Number:         PEW6129E
      Model: HP      DG0146FARVU
      Current Temperature (C): 30
      Maximum Temperature (C): 38
      PHY Count: 2
      PHY Transfer Rate: 6.0Gbps, Unknown


Array: C
   Interface Type: SAS
   Unused Space: 0  MB
   Status: OK
   Array Type: Data


   Logical Drive: 3
      Size: 136.7 GB
      Fault Tolerance: 0
      Heads: 255
      Sectors Per Track: 32
      Cylinders: 35132
      Strip Size: 256 KB
      Full Stripe Size: 256 KB
      Status: OK
      Caching:  Enabled
      Unique Identifier: 600508B1001CCD5D506E7ED19C40A64C
      Logical Drive Label: A00D040150014380130 72F30A2DD
      Drive Type: Data
      LD Acceleration Method: Controller Cache

   physicaldrive 1I:1:3
      Port: 1I
      Box: 1
      Bay: 3
      Status: OK
      Drive Type: Data Drive
      Interface Type: SAS
      Size: 146 GB
```

```
      Native Block Size: 512
      Rotational Speed: 10000
      Firmware Revision: HPD8
      Serial Number:         PEW60NUE
      Model: HP      DG0146FARVU
      Current Temperature (C): 30
      Maximum Temperature (C): 39
      PHY Count: 2
      PHY Transfer Rate: 6.0Gbps, Unknown

<<<truncated>>>


unassigned

   physicaldrive 2I:1:8
      Port: 2I
      Box: 1
      Bay: 8
      Status: OK
      Drive Type: Unassigned Drive
      Interface Type: SAS
      Size: 146 GB
      Native Block Size: 512
      Rotational Speed: 15000
      Firmware Revision: HPD9
      Serial Number:         PLX47B1E
      Model: HP      EH0146FARWD
      Current Temperature (C): 31
      Maximum Temperature (C): 46
      PHY Count: 2
      PHY Transfer Rate: 6.0Gbps, Unknown


SEP (Vendor ID PMCSIERA, Model  SRC 8x6G) 250
   Device Number: 250
   Firmware Version: RevC
   WWID: 5001438013072F3F
   Vendor ID: PMCSIERA
   Model:  SRC 8x6G


~ #
```

The unassigned device at the end of the list is a device that has not yet had a raid configuration created on it, and is thus not consumed by the ESXi host.

Now we can look at a few additional commands to check the disk configuration. This is quite useful to figure out whether disks are being passed thru to the ESXi host, or whether or not there is a RAID-0 volume configured on them, as is necessary with some of the controllers on the VCG. In this example, we can see that each individual disk has got a RAID-0 configuration on it when we look at the logical drives with this command:

### esxcli hpssacli cmd -q "controller slot=0 show config"

```
~ # esxcli hpssacli cmd -q "controller slot=0 show config"

Smart Array P410i in Slot 0 (Embedded)    (sn: 5001438013072F30)

   Internal Drive Cage at Port 1I, Box 1, OK

   Internal Drive Cage at Port 2I, Box 1, OK

   array A (SAS, Unused Space: 0  MB)

      logicaldrive 1 (136.7 GB, RAID 0, OK)

      physicaldrive 1I:1:1 (port 1I:box 1:bay 1, SAS, 146 GB, OK)

   array B (SAS, Unused Space: 0  MB)

      logicaldrive 2 (136.7 GB, RAID 0, OK)

      physicaldrive 1I:1:2 (port 1I:box 1:bay 2, SAS, 146 GB, OK)

   array C (SAS, Unused Space: 0  MB)

      logicaldrive 3 (136.7 GB, RAID 0, OK)

      physicaldrive 1I:1:3 (port 1I:box 1:bay 3, SAS, 146 GB, OK)

   array D (SAS, Unused Space: 0  MB)

      logicaldrive 4 (136.7 GB, RAID 0, OK)

      physicaldrive 1I:1:4 (port 1I:box 1:bay 4, SAS, 146 GB, OK)

   <<truncated>>

   unassigned

      physicaldrive 2I:1:8 (port 2I:box 1:bay 8, SAS, 146 GB, OK)

   SEP (Vendor ID PMCSIERA, Model  SRC 8x6G) 250  (WWID: 5001438013072F3F)
```

This creation of logical drives is fully supported for certain storage controllers, but one should certainly check the VCG to make sure it is a valid configuration option for your controller. If you want to get a more granular view of the individual logical drives, the following command can be used.

### esxcli hpssacli cmd -q "controller slot=0 Array A logicaldrive 1 show"

```
~ # esxcli hpssacli cmd -q "controller slot=0 Array A logicaldrive 1 show"

Smart Array P410i in Slot 0 (Embedded)

   array A

      Logical Drive: 1
         Size: 136.7 GB
         Fault Tolerance: 0
         Heads: 255
         Sectors Per Track: 32
         Cylinders: 35132
         Strip Size: 256 KB
         Full Stripe Size: 256 KB
         Status: OK
         Caching:  Enabled
         Unique Identifier: 600508B1001C3EA7838C0436DBE6D7A2
         Logical Drive Label: A00D1A285001438013072F305019
         Drive Type: Data
         LD Acceleration Method: Controller Cache


~ # esxcli hpssacli cmd -q "controller slot=0 Array B logicaldrive 2 show"

Smart Array P410i in Slot 0 (Embedded)

   array B

      Logical Drive: 2
         Size: 136.7 GB
         Fault Tolerance: 0
         Heads: 255
         Sectors Per Track: 32
         Cylinders: 35132
         Strip Size: 256 KB
         Full Stripe Size: 256 KB
         Status: OK
         Caching:  Enabled
         Unique Identifier: 600508B1001C2EE9A6446E708105054B
         Logical Drive Label: A00D0E295001438013072F302E92
         Drive Type: Data
         LD Acceleration Method: Controller Cache


~ #
```

### esxcli hpssacli cmd -q "controller slot=0 Array G modify led=on"

A very useful command for toggling the LED on a drive to verify that, in the event of replacing a drive, this is indeed the correct drive. Use the option "modify led=off" to turn it off again.

### esxcli hpssacli cmd -q "rescan"

If the controller does not detect a new/replaced disk drive, you can use this command to force a rescan.

### esxcli hpssacli cmd -q "controller slot=0 create type type=ld drives=2I:1:8 raid=0"

This command will build a RAID-0 logical driver (ld) on the drive found in slot 2I:1:8. Port is 2I, Box is 1 and Bay is 5. Once this command is run, you should soon start to see the device appear in ESXi (although if it takes a little while because the auto-scan for devices on ESXi has not run, you can also run the `esxcfg-rescan` command against the adapter). The easiest way to do this is to monitor the vmkernel.log file on ESXi and observe the message related to the device. Here is an example of a RAID-0 logical volume created on a HP Smart Array controller being detected by the ESXi host:

```
2014-11-06T13:25:33.076Z cpu22:8671255)ScsiScan: 976: Path 'vmhba1:C0:T0:L8': Vendor:
'HP      '  Model: 'LOGICAL VOLUME  '  Rev: '3.66'
2014-11-06T13:25:33.076Z cpu22:8671255)ScsiScan: 979: Path 'vmhba1:C0:T0:L8': Type: 0x0,
ANSI rev: 5, TPGS: 0 (none)
2014-11-06T13:25:33.076Z cpu22:8671255)ScsiScan: 1503: Add path: vmhba1:C0:T0:L8
2014-11-06T13:25:33.077Z cpu22:8671255)ScsiPath: 4695: Plugin 'NMP' claimed path
'vmhba1:C0:T0:L8'
2014-11-06T13:25:33.077Z cpu22:8671255)vmw_psp_fixed:
psp_fixedSelectPathToActivateInt:479: Changing active path from NONE to vmhba1:C0:T0:L8
for device "Unregistered Device".
2014-11-06T13:25:33.077Z cpu22:8671255)StorageApdHandler: 698: APD Handle  Created with
lock[StorageApd0x410910]
2014-11-06T13:25:33.077Z cpu22:8671255)ScsiEvents: 501: Event Subsystem: Device Events,
Created!
2014-11-06T13:25:33.077Z cpu22:8671255)VMWARE SCSI Id: Id for vmhba1:C0:T0:L8
0x60 0x05 0x08 0xb1 0x00 0x1c 0x57 0x7e 0x11 0xdd 0x04 0x2e 0x14 0x2a 0x58 0x3f 0x4c 0x4f
0x47 0x49 0x43 0x41
2014-11-06T13:25:33.077Z cpu22:8671255)ScsiDeviceIO: 7456: Get VPD 86 Inquiry for device
"naa.600508b1001c577e11dd042e142a583f" from Plugin "NMP" failed. Not supported
2014-11-06T13:25:33.077Z cpu22:8671255)ScsiDeviceIO: 6187: QErr is correctly set to 0x0
for device naa.600508b1001c577e11dd042e142a583f.
2014-11-06T13:25:33.077Z cpu22:8671255)ScsiDeviceIO: 6684: Could not detect setting of
sitpua for device naa.600508b1001c577e11dd042e142a583f. Error Not supported.
2014-11-06T13:25:33.082Z cpu7:151751 opID=71297a98)World: 14299: VC opID hostd-aa04 maps
to vmkernel opID 71297a98
2014-11-06T13:25:33.143Z  cpu22:8671255)ScsiDevice:  3443:  Successfully  registered  device
"naa.600508b1001c577e11dd042e142a583f" from plugin "NMP" of type 0
```

### esxcli hpssacli cmd -q "controller slot=0 modify cacheratio 100/0”

On HP storage controllers, the configuration advice is to disable cache, or if you cannot disable it completely, set it to 100% read. This command will set the cache ratio to 100% read and 0% write. Read is the first field and write is the second field – 100/0.

### esxcli hpssacli cmd –q “controller slot=0 array H delete forced”

This command will delete disks from the controller. This will remove the disk from the Virtual SAN disk group, so it is advisable that the host is already in maintenance mode, and if possible, the data already evacuated if this is a maintenance operation.

The operational status of the disk is Dead or Error (which basically means an All Paths Down (APD) condition from an ESXi perspective). The disk device also shows up as absent in the disk group. For additional details on observed behaviors when a device is removed in this way, refer to the storage section which discusses failed or removed devices.

If you simply want to test the behavior of Virtual SAN when a drive is offlined, then this command will place either a magnetic disk or SSD offline. Virtual SAN will then mark this disk as absent, and unless the original disk is reinserted before the rebuild timeout has expired, rebuilding of components will begin.

### esxcli hpssacli cmd -q "controller slot=0 create type=ld drives=1I:1:4 raid=0"

This command adds a new logical device with a RAID-0 configuration to the host using physical drive located at 1I:1:4.

If you previously force deleted a disk from the host (either magnetic disk or SSD), then this command will add it back in with a RAID-0 configuration, a necessary step for Virtual SAN to recognize devices from some HP controllers.

# Appendix B – Useful DELL Utilities

Dell has a number of useful utilities that can provide additional insight into storage controller and disk behavior when Dell servers are used for Virtual SAN. There is Dell OpenManager for vCenter plugin and iDRAC service module VIB for ESXi:

## OpenManage Integration for VMware vCenter

http://www.dell.com/support/home/ie/en/iebsdt1/Drivers/DriversDetails?driverId=7Y37P

The OpenManage Integration for VMware vCenter is a virtual appliance that streamlines tools and tasks associated with management and deployment of Dell servers in the virtual environment.

## Dell's OpenManage Server Administrator (OMSA)

http://en.community.dell.com/techcenter/systems-management/w/wiki/1760.openmanage-server-administrator-omsa

OMSA is a software agent that provides a comprehensive, one-to-one systems management solution in two ways: from an integrated, Web browser-based graphical user interface (GUI) and from a command line interface (CLI) through the operating system.

One useful feature is the ability to replace a failed RAID-0 drive without rebooting the host, when you use a controller that does not support passthru.

## Dell iDRAC Service Module (VIB) for ESXi 5.5, v2.0

http://www.dell.com/support/home/ie/en/iebsdt1/Drivers/DriversDetails?driverId=DRKGR

The Integrated Dell Remote Access Controller (iDRAC) Service Module is a lightweight optional software application that can be installed on Dell 12th Generation Server or greater with iDRAC7. It complements iDRAC interfaces – Graphical User Interface (GUI), Remote Access Controller Admin (RACADM) CLI and Web Service Management (WSMAN) with additional monitoring data.

# Appendix C – Useful LSI MegaCLI Utility

LSI provide a MegaCli utility for their storage controller. It provides a lot of useful extensions for troubleshooting. The following steps explain on setting it up on the ESX host. You can download the ESXi/VMware version of MegaCLI from http://www.lsi.com/support/pages/download-results.aspx?keyword=MegaCli.

To install the utility:

- Copy the VIB file to the ESXi host.
- Install the MegaCli vib on the host by running "esxcli software vib install –v <full path to the vib>".
- Once the command completes successfully, verify that the utility is installed successfully.
    - "cd" to the directory "/opt/lsi/MegaCLI".
    - Run command "./MegaCli -adpcount". The command should complete successfully and return with the controller count on the host.

### MegaCli -adpallinfo –aall

Determine the adapter number of the HBA:

```
Adapter #1
==============================================================
                    Versions
                ================
Product Name    : LSI MegaRAID SAS 9271-8i
Serial No       : SV40318544
FW Package Build: 23.28.0-0015
```

From the sample output above we see that the LSI 9271 is adapter 1. This is the value being referred to when using the <adapter number> in the commands below.

### MegaCli -encinfo –a<adapter number>

Determine the enclosure id to which the HBA and the disks are connected.

```
/opt/lsi/MegaCLI # ./MegaCli -encinfo -a1

    Number of enclosures on adapter 1 -- 1

    Enclosure 0:
    Device ID                     : 252
    Number of Slots               : 8
    Number of Power Supplies      : 0
    Number of Fans                : 0
    Number of Temperature Sensors : 0
    Number of Alarms              : 0
    Number of SIM Modules         : 1
    Number of Physical Drives     : 7
    Status                        : Normal
    Position                      : 1
    Connector Name                : Unavailable
    Enclosure type                : SGPIO
    FRU Part Number               : N/A
```

```
Enclosure Serial Number       : N/A
ESM Serial Number             : N/A
Enclosure Zoning Mode         : N/A
Partner Device Id             : Unavailable

Inquiry data                  :
    Vendor Identification     : LSI
    Product Identification    : SGPIO
    Product Revision Level    : N/A
    Vendor Specific           :
```

The enclosure ID is the value of the "Device ID" parameter that in the sample output is 252. You will need this value for disk replacement procedures where the drive needs to be taken offline before removal. This is covered in detail in the Virtual SAN Administrators Guide and various knowledgebase articles.

### MegaCli -Cfgdsply –a<adapter number>

Display the adapter configuration.

```
# ./MegaCli -Cfgdsply -a1
```

Here are some sample outputs when this command has been run against various storage controllers. It includes the details from both a magnetic disk and an SSD. Note that this is a very useful command to verify whether or not cache is disabled on the controllers, and if it is not, has it been set to 100% read.

### MegaCli -Cfgdsply : LSI 9271 : Magnetic Disk

```
Adapter 1 -- Virtual Drive Information:

Virtual Drive: 1 (Target Id: 1)
Name                :HGST-S2
RAID Level          : Primary-0, Secondary-0, RAID Level Qualifier-0
Size                : 418.656 GB
Sector Size         : 512
Is VD emulated      : No
Parity Size         : 0
State               : Optimal
Strip Size          : 256 KB
Number Of Drives    : 1
Span Depth          : 1
Default Cache Policy: WriteThrough, ReadAhead, Direct, No Write Cache if
Bad BBU
Current Cache Policy: WriteThrough, ReadAhead, Direct, No Write Cache if
Bad BBU
Default Access Policy: Read/Write
Current Access Policy: Read/Write
Disk Cache Policy   : Disk's Default
Encryption Type     : None
PI type:  No PI
Is VD Cached: No
```

## MegaCli -Cfgdsply : LSI 9271 : SSD

```
Adapter 1 -- Virtual Drive Information:

Virtual Drive: 5 (Target Id: 5)
Name                :SSD-S6
RAID Level          : Primary-0, Secondary-0, RAID Level Qualifier-0
Size                : 185.781 GB
Sector Size         : 512
Is VD emulated      : Yes
Parity Size         : 0
State               : Optimal
Strip Size          : 256 KB
Number Of Drives    : 1
Span Depth          : 1
Default Cache Policy: WriteThrough, ReadAhead, Direct, No Write Cache if
Bad BBU
Current Cache Policy: WriteThrough, ReadAhead, Direct, No Write Cache if
Bad BBU
Default Access Policy: Read/Write
Current Access Policy: Read/Write
Disk Cache Policy   : Enabled
Encryption Type     : None
PI type: No PI
Is VD Cached: No
```

## MegaCli -Cfgdsply : Dell H710 Magnetic Disk

```
Adapter 0 -- Virtual Drive Information:

Virtual Drive: 1 (Target Id: 1)
Name                :
RAID Level          : Primary-0, Secondary-0, RAID Level Qualifier-0
Size                : 931.0 GB
Sector Size         : 512
Parity Size         : 0
State               : Optimal
Strip Size          : 64 KB
Number Of Drives    : 1
Span Depth          : 1
Default Cache Policy: WriteThrough, ReadAhead, Direct, No Write Cache if Bad BBU
Current Cache Policy: WriteThrough, ReadAhead, Direct, No Write Cache if Bad BBU
Default Access Policy: Read/Write
Current Access Policy: Read/Write
Disk Cache Policy   : Disk's Default
Encryption Type     : None
Default Power Savings Policy: Controller Defined
Current Power Savings Policy: None
Can spin up in 1 minute: Yes
LD has drives that support T10 power conditions: Yes
LD's IO profile supports MAX power savings with cached writes: No
Bad Blocks Exist: No
Is VD Cached: Yes
Cache Cade Type : Read Only
```

## MegaCli -Cfgdsply : Dell H710 : SSD

```
Adapter 0 -- Virtual Drive Information:

Virtual Drive: 6 (Target Id: 6)
Name                :
RAID Level          : Primary-0, Secondary-0, RAID Level Qualifier-0
Size                : 185.75 GB
Sector Size         : 512
Parity Size         : 0
State               : Optimal
Strip Size          : 256 KB
Number Of Drives    : 1
Span Depth          : 1
Default Cache Policy: WriteThrough, ReadAhead, Direct, No Write Cache if Bad BBU
Current Cache Policy: WriteThrough, ReadAhead, Direct, No Write Cache if Bad BBU
Default Access Policy: Read/Write
Current Access Policy: Read/Write
Disk Cache Policy   : Disk's Default
Encryption Type     : None
Default Power Savings Policy: Controller Defined
Current Power Savings Policy: None
Can spin up in 1 minute: No
LD has drives that support T10 power conditions: No
LD's IO profile supports MAX power savings with cached writes: No
Bad Blocks Exist: No
Is VD Cached: No
```

# Appendix D - Advanced Settings

There are only two advanced settings that can be considered tunable at this point in time.

***Caution**: While VMware recommends against changing any of these advanced settings, you need to ensure that all hosts in the cluster have the same settings if you modify any of the advanced settings for Virtual SAN.*

## VSAN.ClomRepairDelay

The `vsan.clomrepairdelay` setting specifies the amount of time Virtual SAN waits before rebuilding a disk object. By default, the repair delay value is set to 60 minutes; this means that in the event of a failure that renders components ABSENT, Virtual SAN waits 60 minutes before rebuilding any disk objects. This is because Virtual SAN is not certain if the failure is transient or permanent.

Changing this setting requires a clomd restart.

**Note**: If a failure in a physical hardware component is detected, such as an Solid State Disk (SSD) or Magnetic Disk (MD), this results in components being marked as DEGRADED and Virtual SAN immediately responds by rebuilding the impacted components. KB article 2075456 has instructions on how to change this setting.

## VSAN.ClomMaxComponentsSizeGB

By default `VSAN.ClomMaxComponentSizeGB` is set to 255GB. When Virtual SAN stores virtual machine objects, it creates components whose default size does not exceed 255 GB. If you use physical disks that are smaller than 255GB, then you might see errors similar to the following when you try to deploy a virtual machine:

```
There is no more space for virtual disk XX. You might be able to continue this session by
freeing disk space on the relevant volume and clicking retry.
```

In situations like this, you may need to modify `VSAN.ClomMaxComponentSizeGB` to s size that is approximately 80% of the physical disk size. KB article 2080503 has instructions on how to change this setting.

## VSAN.goto11

For VSAN 5.5, this advanced setting was required to go to 32 nodes. For VSAN 6.0, it is needed for 64-node support. In Virtual SAN 5.5, to check the existing configuration:

```
esxcli system settings advanced list -o /CMMDS/goto11
```

To set the advanced option in version 5.5:

```
esxcli system settings advanced set -o /CMMDS/goto11 -i 1
```

Note that in Virtual SAN 6.0, the location of the advanced setting has changed. In 6.0, it is **VSAN.goto11**.

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com