Datastore Management

Volume Sizing and Count

A common question when first provisioning storage on the FlashArray is what capacity should I be using for each volume? VMware VMFS supports up to a maximum size of 64 TB. The FlashArray supports far larger than that, but for ESXi, volumes should not be made larger than 64 TB due to the filesystem limit of VMFS.

Using a smaller number of large volumes is generally a better idea today. In the past a recommendation to use a larger number of smaller volumes was made for performance limitations that no longer exist. This limit traditionally was due to two reasons: VMFS scalability issues due to locking and/or per-volume queue limitations on the underlying array. VMware resolved the first issue with the introduction of Atomic Test and Set, also called Hardware Assisted Locking.

Prior to the introduction of VAAI ATS (Atomic Test and Set), VMFS used LUN-level locking via full SCSI reservations to acquire exclusive metadata control for a VMFS volume. In a cluster with multiple nodes, all metadata operations were serialized and hosts had to wait until whichever host, currently holding a lock, released that lock. This behavior not only caused metadata lock queues but also prevented standard I/O to a volume from VMs on other ESXi hosts which were not currently holding the lock.

With VAAI ATS, the lock granularity is reduced to a much smaller level of control (specific metadata segments, not an entire volume) for the VMFS that a given host needs to access. This behavior makes the metadata change process not only very efficient, but more importantly provides a mechanism for parallel metadata access while still maintaining data integrity and availability. ATS allows for ESXi hosts to no longer have to queue metadata change requests, which consequently speeds up operations that previously had to wait for a lock. Therefore, situations with large amounts of simultaneous virtual machine provisioning operations will see the most benefit. The standard use cases benefiting the most from ATS include:

- High virtual machine to VMFS density.
- Extremely dynamic environments—numerous provisioning and de-provisioning of VMs (e.g. VDI using non-persistent linked-clones).
- High intensity virtual machine operations such as boot storms, or virtual disk growth.

The introduction of ATS removed scaling limits via the removal of lock contention; thus, moving the bottleneck down to the storage, where many traditional arrays had per-volume I/O queue limits. This limited what a single volume could do from a performance perspective as compared to what the array could do in aggregate. This is not the case with the FlashArray.

A FlashArray volume is not limited by an artificial performance limit or an individual queue. A single FlashArray volume can offer the full performance of an entire FlashArray, so provisioning ten volumes instead of one, is not going to empty the HBAs out any faster. From a FlashArray perspective, there is no immediate performance benefit to using more than one volume for your virtual machines.
The main point is that there is always a bottleneck somewhere, and when you fix that bottleneck, it is just transferred somewhere else. ESXi was once the bottleneck due to its locking mechanism, then it fixed that with ATS. This, in turn, moved the bottleneck down to the array volume queue depth limit. The FlashArray doesn’t have a volume queue depth limit, so now that bottleneck has been moved back to ESXi and its internal queues.

Altering VMware queue limits is not generally needed with the exception of extraordinarily intense workloads. For high-performance configuration, refer to the section of this document on ESXi queue configuration.

**VMFS Version Recommendations**

Pure Storage recommends using the latest supported version of VMFS that is permitted by your ESXi host.

For ESXi 5.x through 6.0, use VMFS-5. For ESXi 6.5 and later it is highly recommended to use VMFS-6. It should be noted that VMFS-6 is not the default option for ESX 6.5, so be careful to choose the correct version when creating new VMFS datastores in ESXi 6.5.

Furthermore, when upgrading to ESXi 6.5, there is no in-place upgrade path of a VMFS-5 datastore to VMFS-6. Therefore, it is recommended to create a new volume entirely, format it was VMFS-6, and then Storage vMotion all virtual machines from the old VMFS-5 datastore to the new VMFS-6 datastore and then delete and remove the VMFS-5 datastore when complete.

**BEST PRACTICE:** Use the latest supported VMFS version for the in-use ESXi host

**Datastore Performance Management**

ESXi and vCenter offer a variety of features to control the performance capabilities of a given datastore. This section will overview FlashArray support and recommendations for these features.

For a deeper-dive of ESXi queueing and the FlashArray, please read this post:

Queue Depth Limits and DSNRO

ESXi offers the ability to configure queue depth limits for devices on a HBA or iSCSI initiator. This dictates how many I/Os can be outstanding to a given device before I/Os start queuing in the ESXi kernel. If the queue depth limit is set too low, IOPS and throughput can be limited and latency can increase due to queuing. If too high, virtual machine I/O fairness can be affected and high-volume workloads can affect other workloads from other virtual machines or other hosts. The device queue depth limit is set on the initiator and the value (and setting name) varies depending on the model and type:

<table>
<thead>
<tr>
<th>Type</th>
<th>Default Value</th>
<th>Value Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>QLogic</td>
<td>64</td>
<td>qlfxmaxqdepth</td>
</tr>
<tr>
<td>Brocade</td>
<td>32</td>
<td>bfa_lun_queue_depth</td>
</tr>
<tr>
<td>Emulex</td>
<td>32</td>
<td>lpfc0_lun_queue_depth</td>
</tr>
<tr>
<td>Cisco UCS</td>
<td>32</td>
<td>fnic_max_qdepth</td>
</tr>
<tr>
<td>Software iSCSI</td>
<td>128</td>
<td>iscsivmk_LunQDepth</td>
</tr>
</tbody>
</table>

Changing these settings require a host reboot. For instructions to check and set these values, please refer to this VMware KB article:

Changing the queue depth for QLogic, Emulex, and Brocade HBAs

There is a second per-device setting called “Disk Schedule Number Requests Outstanding” often referred to as DSRNO. This is a hypervisor-level queue depth limit that provides a mechanism for managing the queue depth limit for an individual device. This value is a per-device setting that defaults to 32 and can be increased to a value of 256.
It should be noted that this value only comes into play for a volume when that volume is being accessed by two or more virtual machines on that host. If there is more than one virtual machine active on it, the lowest of the two values (DSNRO or the HBA device queue depth limit) is the value that is observed by ESXi as the actual device queue depth limit. So, in other words, if a volume has two VMs on it, and DSRNO is set to 32 and the HBA device queue depth limit is set to 64, the actual queue depth limit for that device is 32. For more information on DSRNO see the VMware KB here:

Setting the Maximum Outstanding Disk Requests for virtual machines

In general, Pure Storage does not recommend changing these values. The majority of workloads are distributed across hosts and/or not intense enough to overwhelm the default queue depths. The FlashArray is fast enough (low enough latency) that the workload has to be quite high in order to overwhelm the queue.

If the default queue depth is consistently overwhelmed, the simplest option is to provision a new datastore and distribute some virtual machines to the new datastore. If a workload from a virtual machine is too great for the default queue depth, then increasing the queue depth limit is the better option.

If a workload demands queue depths to be increased, Pure Storage recommends making both the HBA device queue depth limit and DSNRO equal. Generally, do not change these values without direction from VMware or Pure Storage support.

You can verify the values of both of these for a given device with the command:

```
esxcli storage core device list –d <naa.xxxxx>
Device Max Queue Depth: 96
No of outstanding IOs with competing worlds: 64
```

BEST PRACTICE: Leave queue depth limits at the default. Only raise them when performance requirements dictate it.

Dynamic Queue Throttling

ESXi supports the ability to dynamically throttle a device queue depth limit when an array volume has been overwhelmed. An array volume is overwhelmed when the array responds to an I/O request with a sense code of QUEUE FULL or BUSY. When a certain number of these are received, ESXi will throttle down the queue depth limit for that device and slowly increase it as conditions improve. This is controlled via two settings:

- `Disk.QFullSampleSize`—the count of QUEUE FULL or BUSY conditions it takes before ESXi will start throttling. Default is zero (feature disabled)
- `Disk.QFullThreshold`—the count of good condition responses after a QUEUE FULL or BUSY required before ESXi starts increasing the queue depth limit again

The Pure Storage FlashArray does not advertise a queue full condition for a volume. Since every volume can use the full performance and queue of the FlashArray, this limit is impractically high and this sense code essentially will never be issued. Therefore, there is no reason to set or alter these values for Pure Storage FlashArray volumes because QUEUE FULL will never occur.
Storage I/O Control

VMware vCenter offers a feature called Storage I/O Control (SIOC) that will throttle selected virtual machines when a certain average datastore latency has been reached or when a certain percentage of peak throughput has been hit. ESXi throttles virtual machines by artificially reducing the number of slots that are available to it in the device queue depth limit.

Pure Storage fully supports enabling this technology on datastores residing on the FlashArray. That being said, it may not be particularly useful for a few reasons.

First, the minimum latency that can be configured for SIOC before it will begin throttling a virtual machine is 5 ms.

When a latency threshold is entered, vCenter will aggregate a weighted average of all disk latencies seen by all hosts that see that particular datastore. This number does not include host-side queuing, it is only the time it takes for the I/O to be sent from the SAN to the array and acknowledged back.

Furthermore, SIOC uses a random-read injector to identify the capabilities of a datastore from a performance perspective. At a high-level, it runs a quick series of tests with increasing numbers of outstanding I/Os to identify the throughput maximums via high latency identification. This allows ESXi to determine what the peak throughput is, for when the “Percentage of peak throughput” is chosen.

Knowing these factors, we can make these points about SIOC and the FlashArray:

1. SIOC is not going to be particularly helpful if there is host-side queueing since it does not take host-induced additional latency into account. This (the ESXi device queue) is generally where most of the latency is introduced in a FlashArray environment.
2. The FlashArray will rarely have sustained latency above 1 ms, so this threshold will not be reached for any meaningful amount of time on a FlashArray volume so SIOC will never kick in
3. A single FlashArray volume does not have a queue limit, so it can handle quite a high number of outstanding I/O and throughput (especially reads), therefore SIOC and its random-read injector cannot identify FlashArray limits in meaningful ways.

In short, SIOC is fully supported by Pure Storage, but Pure Storage makes no specific recommendations for configuration.

Storage DRS

VMware vCenter also offers a feature called Storage DRS (SDRS). SDRS moves virtual machines from one datastore to another when a certain average latency threshold has been reach on the datastore or when a certain used capacity has been reached. For this section, let’s focus on the performance-based moves.

Storage DRS, like Storage IO Control, wait for a certain latency threshold to be reached before it acts. And, also like SIOC, the minimum is 5 ms.
While it is too high in general to be useful for FlashArray-induced latency, SDRS differs from SIOC in the latency it actually looks at. SDRS uses the “VMObservedLatency” (referred to a GAVG in esxtop) averages from the hosts accessing the datastore. Therefore, this latency includes time spent queueing in the ESXi kernel. So, theoretically, a high-IOPS workload, with a low configured device queue depth limit, an I/O could conceivably spend 5 ms or more queuing in the kernel. In this situation Storage DRS will suggest moving a virtual machine to a datastore which does not have an overwhelmed queue.

That being said, this is still an unlikely scenario because:

1. The FlashArray empties out the queue fast enough that a workload must be quite intense to fill up an ESXi queue so much that it spends 5 ms or more in it. Usually, with a workload like that, the queueing is higher up the stack (in the virtual machine)
2. Storage DRS samples for 16 hours before it makes a recommendation, so typically you will get one recommendation set per-day for a datastore. So this workload must be consistently and extremely high, for a long time, before SDRS acts.

In short, SDRS is fully supported by Pure Storage, but Pure Storage makes no specific recommendations for performance-based move configuration.

Datastore Capacity Management

Managing the capacity usage of your VMFS datastores is an important part of regular care of your virtual infrastructure. There are a variety of mechanisms inside of ESXi and vCenter to monitor capacity. Frequently, the concept of data reduction on the FlashArray is seen as a complicating factor, when in reality it is a simplifying factor, or at worse, a non-issue. Let’s overview some concepts on how to best manage VMFS datastores from a capacity perspective.

VMFS Usage vs. FlashArray Volume Capacity

VMFS reports how much is currently allocated in the filesystem on that volume. Depending on the type of virtual disk (thin or thick), dictates how much is consumed upon creation of the virtual machine (or virtual disk specifically). Thin disks only allocates what the guest has actually written to, and therefore VMFS only records what the virtual machine has written in its space usage. Thick type virtual disks allocate the full virtual disk immediately, so VMFS records much more space as being used than is actually used by the virtual machines.

This is one of the reasons thin virtual disks are preferred—you get better insight into how much space the guests are actually using.

Regardless of what type you choose, ESXi is going to take the sum total of the allocated space of your virtual disks and compare that to the total capacity of the filesystem of the volume. The used space is the sum of those virtual disks allocations. This number increases as virtual disks grow or new ones are added, and can decrease as old ones are deleted or moved, or even shrunk.

©2018 Copyright Pure Storage. All rights reserved.
Compare this to what the FlashArray reports for a capacity. What the FlashArray reports for a volume usage is NOT the amount used for that volume. What the FlashArray reports is the unique footprint of the volume on that array. Let’s look at this VMFS that is on 512 GB FlashArray volume. The VMFS is therefore 512 GB, but is using 401 GB of space of the filesystem. This means that there are 401 GB of allocated virtual disks:

Now let’s look at the FlashArray volume.

The FlashArray volume shows that 80 GB is being used. Does this mean that VMFS is incorrect? No. VMFS is always the source of truth. The “Volumes” metric represents the amount of physical capacity that has been written to the volume after data reduction that no other volume on the array shares.

This metric can go change at any time as the data set changes on that volume or any other volume on the FlashArray. If, for instance, some other host writes 2 GB to another volume (let’s call it “volume2”), and that 2 GB happens to be identical to 2 GB of that 80 GB GB on “Vol04”, then “Vol04” would no longer have 80 GB of unique space. It would drop down to 78 GB, even though nothing changed on “Vol04” itself. Instead, someone else just happened to write similar data, making the footprint of “InfrastructureDS” less unique.

For a more detailed conversation around this, refer to this blog post:


So, why doesn’t VMFS report the same used capacity as that the FlashArray reports for as used for the underlying volume? Well, because they mean different things. VMware reports what is allocated on the VMFS and the FlashArray reports what is unique on the underlying volume. The FlashArray value can change constantly. The FlashArray metric is only meant to show how reducible the data on that volume is internal to the volume and against the entire array. Conversely, VMFS capacity usage is based solely on how much capacity is allocated to it by virtual machines. The FlashArray volume space metric, on the other hand, actually relates to what is also being used on other volumes. In other words, VMFS usage is only affected by data on the VMFS volume itself. The FlashArray volume space metric is affected by the data on the volume and also on all of volumes. So the two values should not be conflated.

For capacity tracking, you should refer to the VMFS usage. How do we best track VMFS usage? What do we do when it is full?
Monitoring and Managing VMFS Capacity Usage

As virtual machines grow and as new ones are added, the VMFS volume they sit on will slowly fill up. How to respond and to manage this is a common question.

In general, using a product like vRealize Operations Manager with the FlashArray Management Pack is a great option here. But for the purposes of this document we will focus on what can be done inside of vCenter alone.

You need to decide on a few things:

• At what percentage full of my VMFS volume do I become concerned?
• When that happens what should I do?
• What capacity value should I monitor on the FlashArray?

The first question is the easiest to answer. Choose either a percentage full, or at a certain capacity free. Do you want to do something when, for example, a VMFS volume hits 75% full or when there is less than 50 GB left free? Choose what makes sense to you.

vCenter alerts are a great way to monitor VMFS capacity automatically. There is a default alert for datastore capacity, but it does not do anything other than tag the datastore object with the alarm state. Pure Storage recommends creating an additional alarm for capacity that executes some type of additional action when the alarm is triggered.

Configuring a script to run, an email to be issued, or a notification trap to be sent greatly diminishes the chance of a datastore running out of space unnoticed.
The next step is to decide what happens when a capacity warning occurs. There are a few options:

1. **Increase the capacity of the volume**
2. **Move virtual machines off of the volume**
3. **Add a new volume**

Your solution may be one of these options or a mix of all three. Let’s quickly walk through the options.

**Option 1: Increase the capacity of the volume**

This is the simplest option. If capacity has crossed the threshold you have specified, increase the volume capacity to clear the threshold. The process is:

1. **Increase the FlashArray volume capacity:**
2. **Rescan the hosts that use the datastore:**
Increase the VMFS to use the new capacity:

Choose “Use ‘Free space xxx TB’ to expand the datastore”. There should be a note that the datastore already occupies space on this volume. If this note does not appear, you have selected the wrong device to expand to. Pure Storage highly recommends that you do not create VMFS datastores that span multiple volumes—a VMFS should have a one to one relationship to a FlashArray volume.

This will clear the alarm and add additional capacity.

**Option 2: Move virtual machine off of the volume**

Another option is to move one or more virtual machines from a more-full datastore to a less-full datastore. While this can be manually achieved through case-by-case Storage vMotion, Pure Storage recommends leveraging Storage DRS to automate this. Storage DRS provides, in addition to the performance-based moves discussed earlier in this document, the ability to automatically Storage vMotion virtual machines based on capacity usage of VMFS datastores. If a datastore reaches a certain percent full, SDRS can automatically move, or make recommendations for, virtual machines to be moved to balance out space usage across volumes.

SDRS is enabled on a datastore cluster:
When a datastore cluster is created you can enable SDRS and choose capacity threshold settings, which can either be a percentage or a capacity amount:

Pure Storage has no specific recommendations for these values and can be decided upon based on your own environment. Pure Storage does have a few recommendations for datastore cluster configuration in general:

- Only include datastores on the same FlashArray in a given datastore cluster. This will allow Storage vMotion to use the VAAI XCOPY offload to accelerate the migration process of virtual machines and greatly reduce the footprint of the migration workload.
- Include datastores with similar configurations in a datastore cluster. For example, if a datastore is replicated on the FlashArray, only include datastores that are replicated in the same FlashArray protection group so that a SDRS migration does not violate required protection for a virtual machine.

**Option 3: Create a new VMFS volume**

The last option is to create an entirely new VMFS volume. You might decide to do this for a few reasons:

1. The current VMFS volumes have maxed out possible capacity (64 TB each)
2. The current VMFS volumes have overloaded the queue depth inside of every ESXi server using it. Therefore, they can be grown in capacity, but cannot provide any more performance due to ESXi limits.
In this situation follow the standard VMFS provisioning steps for a new datastore. Once the creation of volumes and hosts/host groups and the volume connection is complete, the volumes will be accessible to the ESXi host(s) [Presuming SAN zoning is completed]. Using the vSphere Web Client, initiate a “Rescan Storage…” to make the newly-connected Pure Storage volume(s) fully-visible to the ESXi servers in the cluster as shown above. One can then use the “Add Storage” wizard to format the newly added volume.

**Shrinking a Volume**

While it is possible to shrink a FlashArray volume non-disruptively, vSphere does not have the ability to shrink a VMFS partition. Therefore, **do not shrink** FlashArray volumes that contain VMFS datastores as doing so could incur **data loss**.

**Mounting a Snapshot Volume**

The Pure Storage FlashArray provides the ability to take local or remote point-in-time snapshots of volumes which can then be used for backup/restore and/or test/dev. When a snapshot is taken of a volume containing a VMFS, there are a few additional steps from both the FlashArray and vSphere sides to be able to access the snapshot point-in-time data.

When a FlashArray snapshot is taken, a new volume is not created—essentially it is a metadata point-in-time reference to a data blocks on the array that reflect that moment’s version of the data. This snapshot is immutable and cannot be directly mounted. Instead, the metadata of a snapshot has to be “copied” to an actual volume which then allows the point-in-time, which was preserved by the snapshot metadata, to be presented to a host. This behavior allows the snapshot to be re-used again and again without changing the data in that snapshot. If a snapshot is not needed more than one time an alternative option is to create a direct snap copy from one volume to another—merging the snapshot creation step with the association step.

When a volume hosting a VMFS datastore is copied via array-based snapshots, the copied VMFS datastore is now on a volume that has a different serial number than the original source volume. Therefore, the VMFS will be reported as having an invalid signature since the VMFS datastore signature is a hash partially based on the serial of the hosting device. Consequently, the device will not be automatically mounted upon rescan—instead the new datastore wizard needs to be run to find the device and resignature the VMFS datastore. Pure Storage recommends resignaturing copied volumes rather than mounting them with an existing signatures (referred to as force mounting).
For more detail on resignaturing and snapshot management, please refer to the following blog posts:

- Mounting an unresolved VMFS
- Why not force mount?
- Why might a VMFS resignature operation fail?
- How to correlate a VMFS and a FlashArray volume
- How to snapshot a VMFS on the FlashArray
- Restoring a single VM from a FlashArray snapshot

**Deleting a Datastore**

Prior to the deletion of a volume, ensure that all important data has been moved off or is no longer needed. From the vSphere Web Client (or CLI) delete or unmount the VMFS volume and then detach the underlying device from the appropriate host(s).

After a volume has been detached from the ESXi host(s) it must first be disconnected (from the FlashArray perspective) from the host within the Purity GUI before it can be destroyed (deleted) on the FlashArray.

**BEST PRACTICE:** Unmount and detach FlashArray volumes from all ESXi hosts before destroying them on the array

1. Unmount the VMFS datastore on every host that it is mounted to.
2. Detach the volume that hosted the datastore from every ESXi host that sees the volume

3. Disconnect the volume from the hosts or host groups on the FlashArray

Destroy the volume on FlashArray. It is not recommended to eradicate it. Let the FlashArray eradicate it automatically in 24 to provide for recovery if needed.

By default a volume can be recovered after deletion for 24 hours to protect against accidental removal.
This entire removal and deletion process is automated through the Pure Storage Plugin for the vSphere Web Client and its use is therefore recommended.