SER2355BE

Best Practices for All-Flash Arrays with VMware vSphere

Vaughn Stewart, VP, Enterprise Architect, Pure Storage
Cody Hosterman, Technical Director, Pure Storage

#VMworld  #SER2355BE
AGENDA

– Settings, configuration options, etc designed for every all-flash array regardless of vendor

– Our philosophy...
  – What you need to consider
  – What you don’t need to think about
  – ...and importantly why
BEST PRACTICE BASICS

BASICS

- VMware is the expert on general vSphere storage settings
- Storage vendors are experts on vSphere with their arrays

SIMPLICITY IS THE ALWAYS THE GOAL

- Adopt VVols or Use large datastores – be sure to consider back and up restore times
- Limit use of RDMs – consider VVols when you can
- Avoid jumbo frames for iSCSI & NFS – inconsistent, marginal gains with added complexity
PERFORMANCE CONFIGURATION
VMFS, QUEUING, ETC.
MULTIPATHING—ROUND ROBIN

Set devices to use Round Robin

- Default configuration is usually Most Recently Used—which is less than ideal
- Maximizes performance for array devices by using all available paths
I/O OPERATIONS LIMIT

Set the I/O Operations Limit set to ‘1’ (or sufficiently low)—default is 1,000

- How often ESXi switches logical paths for a given device
- Some vendors have somewhat different numbers 1-20 or so. Follow their recommendation

Why change this?

- Performance (NOT a panacea though...), more important for iSCSI
- Path Failover time (reduces from 10’s of seconds to a few seconds normally)
- I/O Balance

<table>
<thead>
<tr>
<th>Most Recently Used</th>
<th>Round Robin IO=1,000</th>
<th>Round Robin IO=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% imbalance on array ports</td>
<td>~20-30% imbalance on array ports</td>
<td>~ 1% imbalance on array ports</td>
</tr>
</tbody>
</table>
SATP RULES

Best option? SATP rule

- Always set the SATP rule first thing, PRIOR to provisioning any storage to an ESXi host
- Set via SSH or PowerCLI or Host Profiles

Do this once on every host!

```
esxcli storage nmp satp rule add -s "VMW_SATP_ALUA" -V "VENDOR" -M "MODEL" -P "VMW_PSP_RR" -O "iops=1"
```
COMMON QUESTIONS

What about:

– Volume Size?
– Volume Count?
– VM/VMFS Density?

AFAs doesn’t necessarily have requirements here...

...though, there are considerations...
VMFS SCALE ENHANCEMENTS

Replaces traditional SCSI Reservations

- SCSI Reservations decreased cluster performance by locking out I/O during metadata changes

Hardware Assisted Locking

- Only lock the metadata they need
- Permits simultaneous access and does not lock out I/O
- Also known as Atomic Test and Set (ATS)
WHAT IS A QUEUE DEPTH LIMIT?

A queue is a line, and a queue depth limit is how “wide” that line is. Essentially, how many “things” can be allowed through at once.

Example:

One grocery clerk can help one customer at a time (queue depth limit of 1). So, if there are two customers, one must wait for the first to finish (added latency).

If there are two clerks (queue depth limit of 2), two customers can be helped at a time and neither has to wait (no added latency)
WHAT IS A QUEUE DEPTH LIMIT?

In terms of storage, a queue depth limit has many names:

- Outstanding I/Os
- Concurrent threads
- In-flight I/Os

If queue depth limit is 32, 32 I/Os can be processed at once. The 33rd must wait and the 33rd then has added latency because it has to wait.
QUEUE LIMITS, QUEUE LIMITS EVERYWHERE

- Storage Array Queue Depth Limit
- Device Queue Depth Limit
- vSCSI Adapter Queue Depth Limit
- Virtual Disk Queue Depth Limit
ESXI IS DESIGNED TO PROVIDE SOME LEVEL OF FAIRNESS.
This is really the first consideration.
If a volume or a target on an array has a low limit—there is no point to increase anything above it.

For storage arrays with per-volume or per-target limits, volume/target parallelization is the key. Not ESXi tuning. If the array does not have this limit, move on.
HBA DEVICE QUEUE DEPTH LIMIT

This is a HBA setting that controls how many outstanding I/Os can be queued on a particular device.

Values are configured via esxcli

Changing requires a reboot of ESXi

https://kb.vmware.com/kb/1267

Depending on your HBA vendor the default value varies:

<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>
“No of outstanding IOs with competing worlds”

- Controls the number of active I/Os issued to a device when there is more than one VM

DSRNO can be set to a maximum of:

- 6.0 and earlier: 256
- 6.5 and on: Whatever the HBA Device Queue Depth Limit is

```
[root@esxi-01:] esxcli storage core device list -d
naa.624a937076a1e05df05441ba000253a3

(naa.624a937076a1e05df05441ba000253a3)
Vendor: PURE
Model: FlashArray
Revision: 8888
SCSI Level: 6
Is Pseudo: false
Status: on

<...>
Device Max Queue Depth: 128
No of outstanding IOs with competing worlds: 128
```
The actual effective device queue limit will be the minimum of:

- HBA Device Queue Depth Limit
- DSNRO
PARAVIRTUAL SCSI ADAPTER

For high-performance workloads, Paravirtual SCSI adapter is always best

Better CPU efficiency with heavy workloads

Higher default and maximum queue depths

VMware tools includes the drivers

https://kb.vmware.com/kb/1010398

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Adapter Queue Depth</td>
<td>256</td>
</tr>
<tr>
<td>Maximum Adapter Queue Depth</td>
<td>1,024</td>
</tr>
<tr>
<td>Default Virtual Disk Queue Depth</td>
<td>64</td>
</tr>
<tr>
<td>Maximum Virtual Disk Queue Depth</td>
<td>256</td>
</tr>
</tbody>
</table>

*A few slots are reserved for the virtualization layer, so actual available queue slots are slightly lower*
PVSCSI CONTINUED...

- Simply only switching to PV SCSI virtual adapters or increasing the limits in it is unlikely to improve performance (or really at all)
- Device queue depth limit is usually 32 (min of HBA and DSNRO)

Otherwise, you are just moving the bottleneck from the guest to the ESXi kernel.
DQLEN—this is the configured queue depth limit for the volume. This value is identified by looking at the configured HBA queue depth limit, and DSNRO

ACTV—this is the number of slots currently in use by the workload going to the volume. This value will never exceed DQLEN

QUED—this value is populated if the workload exceeds what DQLEN allows. If ACTV = DQLEN, anything over and beyond that will be queued to wait to be sent to the volume.

If QUED is non-zero, latency is added and reported in KAVG
SHOULD I CHANGE QUEUE DEPTH LIMITS WITH MY AFA?

IN GENERAL: NO
**Little's Law**: The long-term average number of customers in a stable system $L$ is equal to the long-term average effective arrival rate, $\lambda$, multiplied by the average time a customer spends in the system, $W$. 

$$L = \lambda W$$
LITTLE’S LAW IN ACTION

Let’s use our grocery store analogy:

If one customer takes 1 minute to check out (get through the line) and there is one clerk, a store can serve 60 customers in an hour.

If there are two clerks, the store can serve 120 customers in an hour.
SOME QUICK MATH...

Let’s suppose the latency should be .5 ms:

• 1 second = 1,000 ms

• 1,000 ms / .5 ms per IO = 2,000 IOPS

• 2,000 IOPS * 32 outstanding I/Os = 64,000 IOPS

With this latency, we would expect 64,000 IOPS maximum per datastore, per host.
DO I NEED TO CHANGE THIS STUFF?

Depends. Usually no.

**YES IF:** you see host-introduced latency and/or you need more available throughput or IOPS, increase the queue depth limits.

**NO BECAUSE:** Most workloads are distributed across VMs, hosts and/or volumes (parallel queues)

**NO BECAUSE:** Low-latency arrays (AFAs) are less likely to need changes—they empty out the queue (i.e. service the I/Os) very fast—more IOPS with lower limits
vSPHERE PERFORMANCE FEATURES
Storage I/O Control

Storage I/O Control?

Latency checks are based on device latency (Array + SAN) so with AFAs, this is unlikely to invoke
Storage DRS performance moves?

Storage DRS includes kernel latency (ESXi queue + SAN + Array aka VM Observed Latency, aka GAVG) so it can be helpful with exceeded queue depth limits.
vSPHERE IOPS LIMITS

IOPS Limits in vSphere might be an option to look at if you are opening up queues, to give more important VMs priority.

In ESXi 6.5 can be set via policy:

In ESXi 6.0 and earlier set per-VM manually:
I/O LATENCY IMPACTS CPU & MEMORY
MYTH BUSTING: I DON’T NEED THE PERFORMANCE OF AN AFA

The common view is all-flash storage accelerates applications – this is true.

Most VMs are ‘Tier-2’ and thus most think they don’t need the performance of all-flash storage – this is a misunderstanding of the impact I/O latency has on Hypervisor CPU & Memory.

Storage I/O is exponentially slower than CPU processing.

All-Flash allows data to be processed faster resulting in more CPU cycles so you can process more data. The memory associated with the I/O is released faster.

**Bottom Line: Reducing I/O Latency allows for...**

1. More DB transactions per CPU (faster applications)
2. More VMs per CPU (reduced infrastructure costs)
STORAGE EFFICIENCY
REDUCING THE TOTAL COST PER VM
DATA REDUCTION: A COLLECTION OF TECHNOLOGIES & IMPLEMENTATIONS

DEDUPLICATION:
- Inline or Background
  - Nominal diff in final results
  - Relates to performance impact relative to architecture design
- What Matters to Results
  - Global vs Volume/Disk Grp
    - One copy vs multiple copies
    - Global is Better i.e. Inter-Vol XCOPY
  - Granularity of Block Size
    - 16KB / 8KB / 4KB / 512B
    - Smaller is Better

COMPRESSION:
- Inline or Background
  - All inline is somewhat light-weight in order to preserve performance
- What Matters to Results
  - Single vs Multiple algorithms
    - CPU resource impact
    - Align algorithm to data type

THIN PROVISIONING:
- Thin to Start is Lowest Common Denominator
- What Matters to Results
  - Override Host Thick Provisioning
  - Zero & Pattern Removal
  - VM Filesystem and Datastore UNMAP
  - Remove deleted data from storage layer
  - Found only with SCSI-Based Datastores & VVols

BOTTOM LINE:
The more technologies, the better the results
YOUR STORAGE IS STORING DELETED DATA

Dead Space accumulation occurs in two places:

- **Datastore**: On the VMFS after a VM has been deleted or moved.
- **In-VM**: Inside a virtual machine’s virtual disk when a file has been deleted or moved.
ESXi 5.5 & 6.0 MANUAL UNMAP OF VMFS DATASTORES
RUNNING MANUAL VMFS UNMAP

ESXi 5.5 and 6.0

UNMAP can be executed from a variety of ways:

SSH into ESXi:

```
esxcli storage vmfs unmap -l <VMFS> -n <block count>
```

PowerCLI:

```
$esxcli=Get-ESXcli -VMHost $esx -v2
$unmapargs = $esxcli.Storage.Vmfs.Unmap.Createargs()
$unmapargs.volumelabel = $datastore.Name
$unmapargs.reclaimunit = $blockcount
$esxcli.Storage.Vmfs.Unmap.Invoke($unmapargs)
```

vRealize Orchestrator:

Web Client Plugins:
ESXi 6.5 AUTOMATIC UNMAP OF VMFS DATASTORES
AUTOMATIC UNMAP IN VMFS 6
NEW IN VSHERE 6.5

In vSphere 6.5 with VMFS 6, VMware (re)introduces Automatic UNMAP

Per-datastore setting

All-attached ESXi hosts run a background crawler and will asynchronously issue UNMAP

• Can take 12+ hours to reclaim
• Must have powered-on VMs
ESXi 6.0 & 6.5 AUTOMATIC VM FILESYSTEM UNMAP
EnableBlockDelete Setting

ESXi can also automatically reclaim the array level dead space after the virtual disk has been shrunk.

Turn on EnableBlockDelete to complete the end-to-end UNMAP.

This is not really required in ESXi 6.5+
VM FILESYSTEM UNMAP IN VSPHERE 6.X

Requirements:
- ESXi 6.0+
- VM Hardware version 11
- Thin virtual disk
- EnableBlockDelete
VM FILESYSTEM UNMAP: SHRINK THE VMDK
Customer with 6.25 TBs of Thin VMs
Reduced to 1.25TBs
5:1 Data Reduction
Customer Enables VM Filesystem UNMAP
Reduced to 740 GBs
8:4 (unreported) Data Reduction
Two options:

1. Scheduled UNMAP via disk optimizer
   1. weekly is default
   2. Daily or Monthly

2. Automatic UNMAP—issue UNMAP when a file is deleted or moved
LINUX TIP: VM FILESYSTEM UNMAP

LINUX (EXT4, XFS, ETC....)

A few options:

1. Scheduled UNMAP via FSTRIM and CRON
2. Automatic UNMAP via filesystem mount 'discard' option

Recommended for Existing VMs
1. Easiest to implement across existing deployed VMs
2. Leverage /etc/cron.d & an automation tool (puppet)
3. Non-disruptive compared to mount options

Recommended for VM Templates

pureuser@ubuntu:/mnt$ sudo mount /dev/sdd /mnt/unmaptest -o discard
ESXI TIP: ONLY UNMAP WITH 6.5 PATCH 1
FIXES ALIGNMENT TOLERANCE ISSUE

VMFS requires all UNMAPs sent from a guest to be aligned to 1 MB boundaries

Not all UNMAPs sent from guests will be, so UNMAP was blocked and nothing was reclaimed

This changes in ESXi 6.5 P1:

- Passes along any UNMAPs that are aligned
- Mis-aligned UNMAPs are instead zeroed out

Provides:

- 100% reclamation on the FlashArray
- Much higher percentage of space returned on VMFS
- Allows fstrim in Linux
- Supports all allocation unit sizes in Windows

Not an issue for AFAs with zero removal or deduplication
## ESXi 6.0 AND 6.5 DIFFERENCES

<table>
<thead>
<tr>
<th></th>
<th>ESXi 6.0</th>
<th>ESXi 6.5</th>
<th>ESXi 6.5 P1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SCSI Version</strong></td>
<td>2</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td><strong>Works with Change Block Tracking On?</strong></td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Virtual Disk Type Supported</strong></td>
<td>Thin</td>
<td>Thin</td>
<td>Thin</td>
</tr>
</tbody>
</table>
UNMAP: THE MORE YOU KNOW

Datastore & VM filesystem UNMAP reduces storage

UNMAP is often unreported by arrays

UNMAP will only remove metadata pointers with deduplicated data. No freeing of physical capacity.

UNMAP will remove unique data that has been deleted
WAIT A MINUTE AREN'T THIN VMDKS SLOW?
THIN -VS- THICK

APPLIES TO AFAS... NOT NECESSARILY FOR HYBRID OR DISK STORAGE PLATFORMS

Historically Thin VMDKs traded off performance for efficiency

- VMFS pauses I/O as new sectors were allocated to the VMDK and formatted
- Performance differences can be observed when performance benchmarking unstructured data in a guest filesystem

Most performance sensitive applications pre allocate space (i.e. SQL Server, Oracle Database, etc) and perform optimally on Thin VMDKs
VIRTUAL VOLUMES
Virtual Volumes

Virtual volumes provide per-VMDK granularity on the underlying array

- Array-based snapshots
- Policy-driven
- No VMFS in the way
  - No more VMFS UNMAP!

Therefore:

1. Can apply array-features and/or VMware features at a per-VMDK granularity (QoS, replication, etc.)

2. Opens up data mobility: a VVol is a volume, so it can be shared with or copied to anything
Virtual Volumes and Performance

Does using virtual volumes bypass all of this queue stuff?

No more datastore = No more DSRNO. Right?

No.

VVols share a queue depth limit from the protocol endpoint they are bound to

Understand what the performance limits of a single volume (PE) on your array

The default queue depth limit of PEs is 128, but can go up to 4,096

*Quick math at 1 ms: 8,192,000 IOPS per PE per host!

Need more than one PE?
IN CONCLUSION

A **BIG THANK YOU** to you, the VMworld Community for Voting for this session

Please complete a survey for this session – SER2355BU

Follow us online:

Cody:

@CodyHosterman

http://CodyHosterman.com

Vaughn:

@vStewed

http://VaughnStewart.com
Please fill out your survey.

Take a survey and enter a draw for a VMware company store gift card.
Thank You