A Day in the Life of a Database I/O

PRIMA DONNA
CONSULTING

Joe Woodhouse

isug tech

Contents

About the author

- Sybase Australia 1996 – 2003; youngest ever Principal Consultant
- After working with Sybase tech for 20+ years… is no longer young
- Principal Database Engineer at Prima Donna Consulting
- Based in London, UK, and Melbourne, Australia
- International Sybase User Group Board of Directors since 2010
- UK Sybase User Group Board of Directors since 2019
- Not a lawyer – no charge for emails!
- Improves client bottom lines by ~USD$10M/month every month

Why this presentation

- Seen many projects where upgraded hardware = worse performance
- This creates stress; causes doubt; threatens future hardware investment
- The work I enjoy the most is when I save projects, budgets, and teams
- This means I do more than just solve problems:
    - I figure out puzzles, uncover secrets, unravel mysteries, and bring order out of chaos
- Now you can unblock your projects, increase your budgets, exceed your targets, impress your bosses – and keep your jobs

Why *this* presentation

- A European bank upgraded to an expensive new storage array
- ASE performance regressed, a lot
- They fixed what they could, which undid the regression, but…
- The business case for the storage upgrade was based on performance
- DBAs, SAs, Hardware, VM, Network, Storage; all teams going nowhere
- Multiple cases raised with multiple vendors; expectations still unmet
- Challenge #1: "we won't give you access to the systems"
- Challenge #2: "fix it in eight days"

Why I/O? Because I/O hurts…

- Even the fastest NVMe storage is still slower than RAM
  - Similar bandwidth (this surprised me)
  - Latency is where storage really hurts
    - RAM (DDR4) = 8-24ns
    - NVMe = 60µs = 60,000ns = 2,500x slower
    - SATA SSD = 175µs = 175,000ns = 7,291x slower
    - SSD on SAN = 2-20ms = 2,000,000ns = 83,333x slower

      20,000,000ns = 833,333x slower

… and (almost) everything ASE does involves I/O

- Every database read or write means at least one I/O
  - Usually more than one (data + indexes + log)
- Yes, even if it is already in data cache
  - How did it get there? One or more I/Os
  - Once written, it must be flushed out of cache else data loss
- Only exception is IMDB
  - And even IMDB is initialised from a template database

The fastest I/O is no I/O

- Those relative latency numbers were compelling
- If you have a database I/O problem, the single best thing to do?
    - Refactor code to issue fewer I/Os!
- Next best thing?
    - Throw memory at it and stay away from storage!
    - Pound for pound, dollar for dollar, nothing helps performance more
- That might not be feasible so let's look at everything else

Benchmarking is always the right move

- There will always be more than one bottleneck
  - But we only ever feel the effects of, and detect, the worst
- We must fix the worst before we can know the next-worst
  - Design a repeatable benchmark for apples-to-apples comparisons
  - Change/tune only one thing at a time
  - Measure! ("Data! Data! Data! I cannot make bricks without clay!")
  - Did it help? Did it hurt? Did it make no difference?
  - Repeat until goals are met

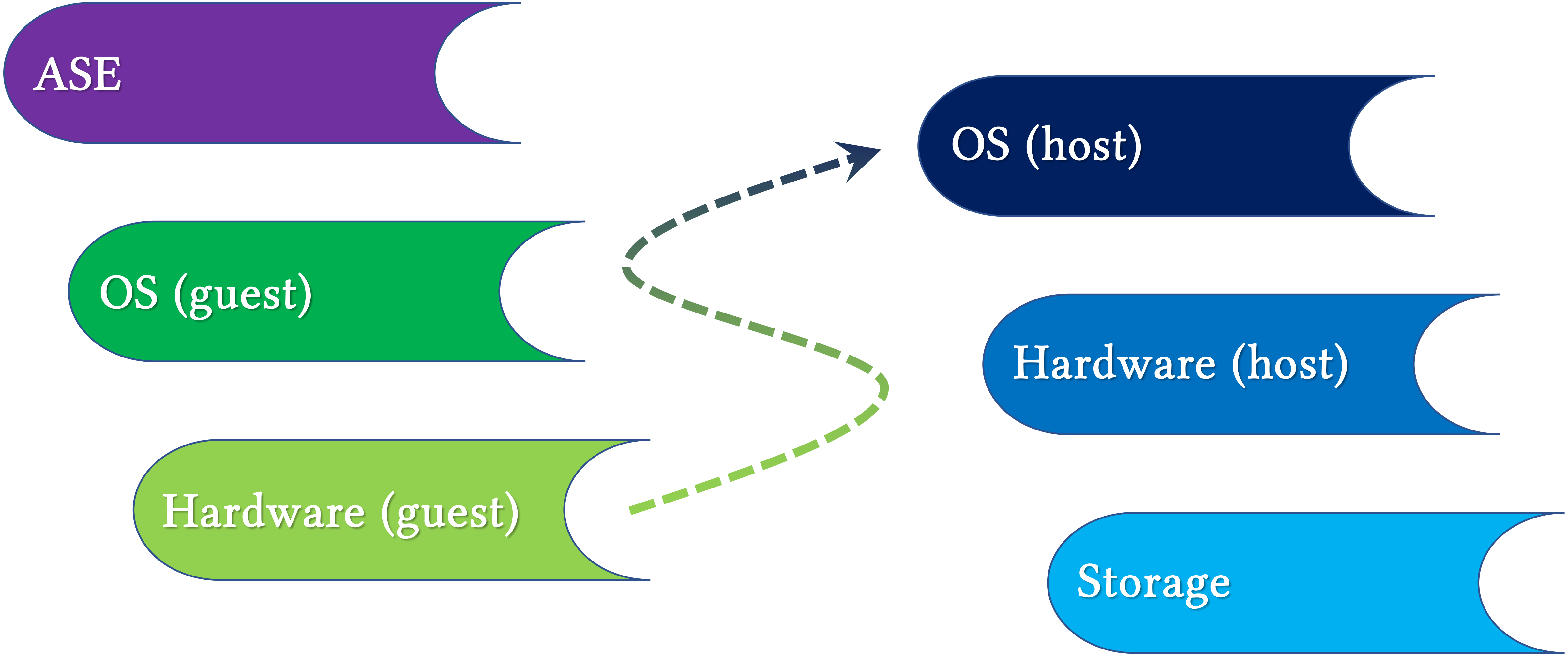
Benchmarking is not always possible

- As in this case study
    - There wasn't time
    - I didn't have access
    - Client wasn't interested in benchmarking
        - (foreshadowing...)
    - Client didn't have much monitoring infrastructure set up
    - They wanted a review only

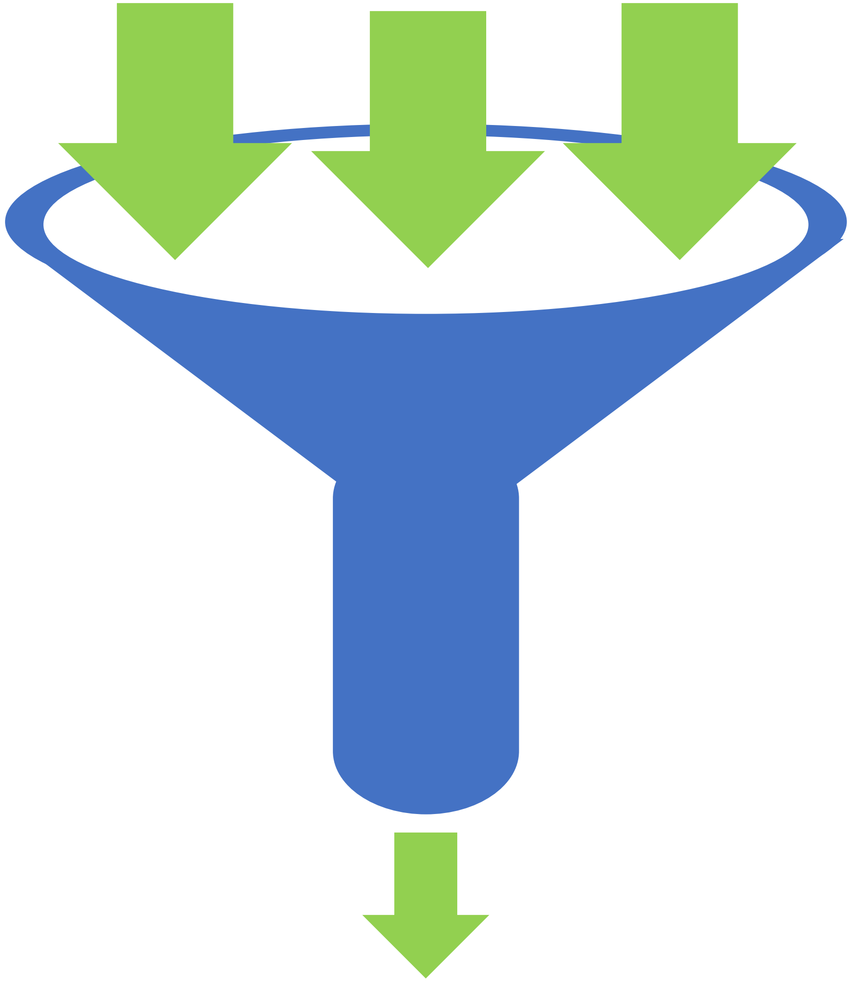
The hammer of best practices

- If we can't benchmark to find the right bottlenecks in the right order…
  - … treat everything as a bottleneck!
- All we have is the hammer of best practices
  - So hit everything, and hit it hard
- In other words survey, review, look for every clue, tune everything

The (database) I/O stack



We think of bottlenecks like this

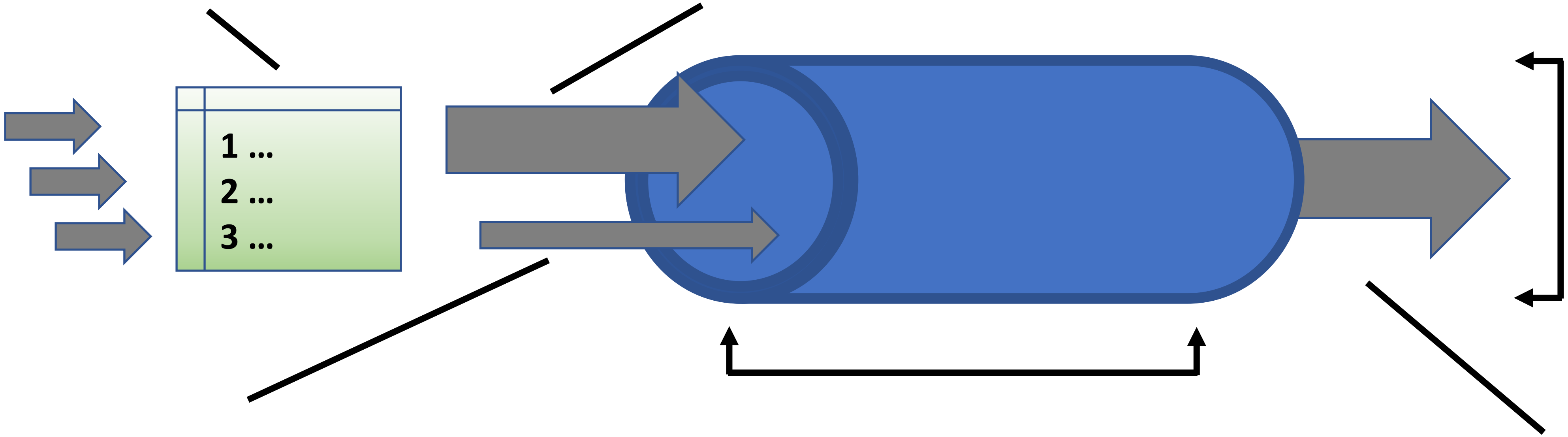


But they're usually more like this

Different ways to measure I/O performance

Queue Depth
Maximum pending I/Os

I/O Size
Amount of I/O per trip



Bandwidth
How wide is the pipe

Concurrency
How many simultaneous inputs fit in the pipe

Latency
How long does it take to reach the end of the pipe

Throughput
How much flows through the pipe

More on queue depths

- Queue depths are at the heart of I/O performance and latency stats
- We can keep adding async I/Os as long as there is room in the queue
- Once queues fill up, no more async I/Os
- All further I/Os wait…
  - **… and this doesn't appear in latency stats in any lower layer**
  - From their point of view, the I/O hasn't yet entered the system
- Queues exist at **every** layer

The (database) I/O stack, revisited

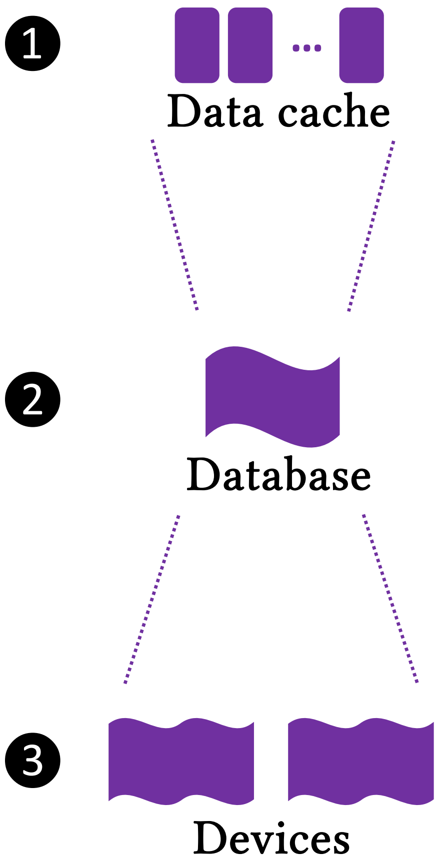
- Each layer can only see:
  - Its own level: queues, concurrency, latency, throughput, bandwidth
  - Levels below: latency (sum only), throughput (min only)
  - Levels above: ***nothing***
- Latency at each layer is counted in every layer above it
  - But latencies (and queues!) above it are not visible at that layer
- So storage can – truthfully – report great performance
- While layers above it can – truthfully – report terrible performance

Follow the I/O: case study details

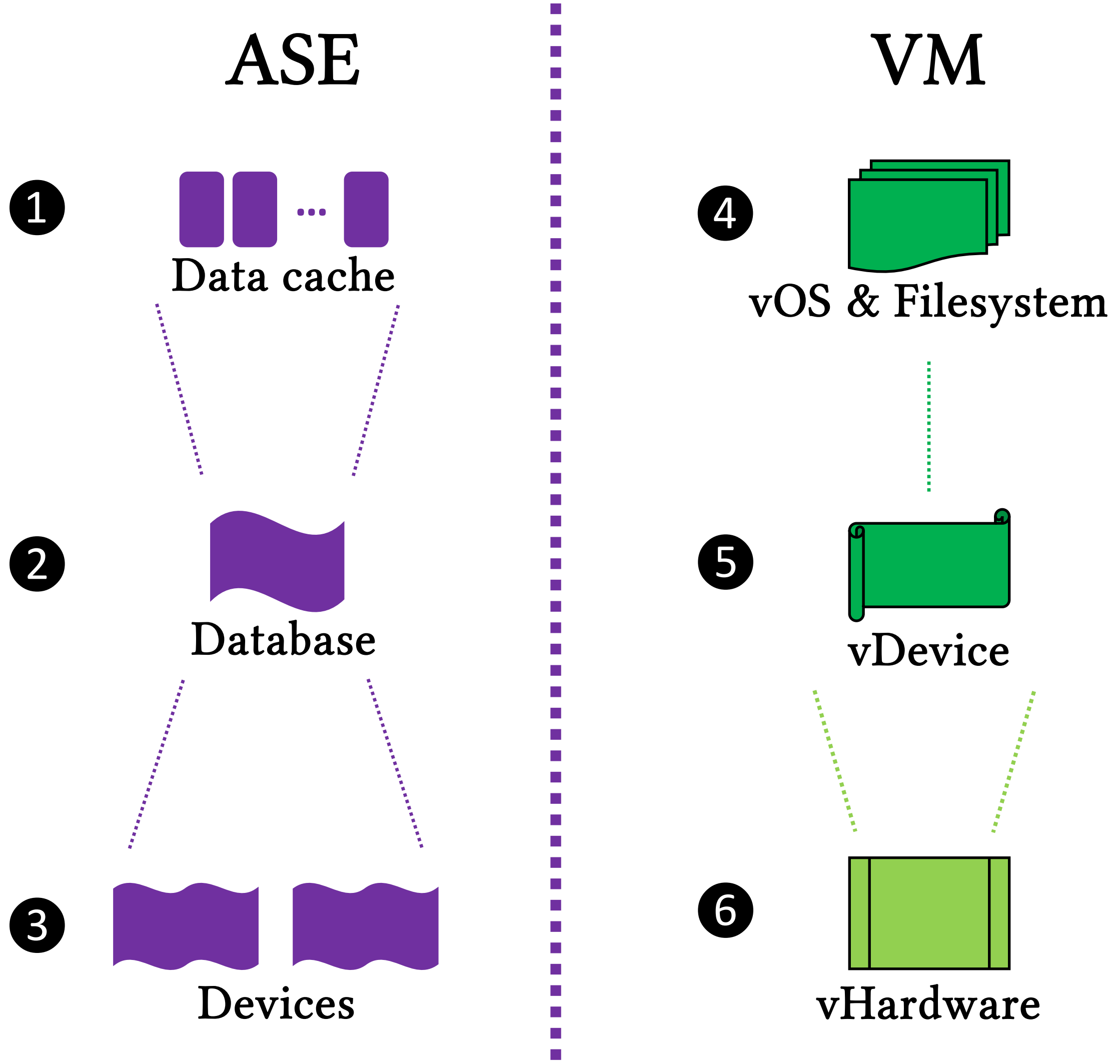
- ASE 16.0 SP04 (16.0 SP04 PL03 HF1 released Dec 22)
- RHEL 7.9
- VMware ESXi 7.0 U2
- Multiple VMs on the parent physical host
- FC multipathing to the storage array
- Pure Storage array

Follow the I/O

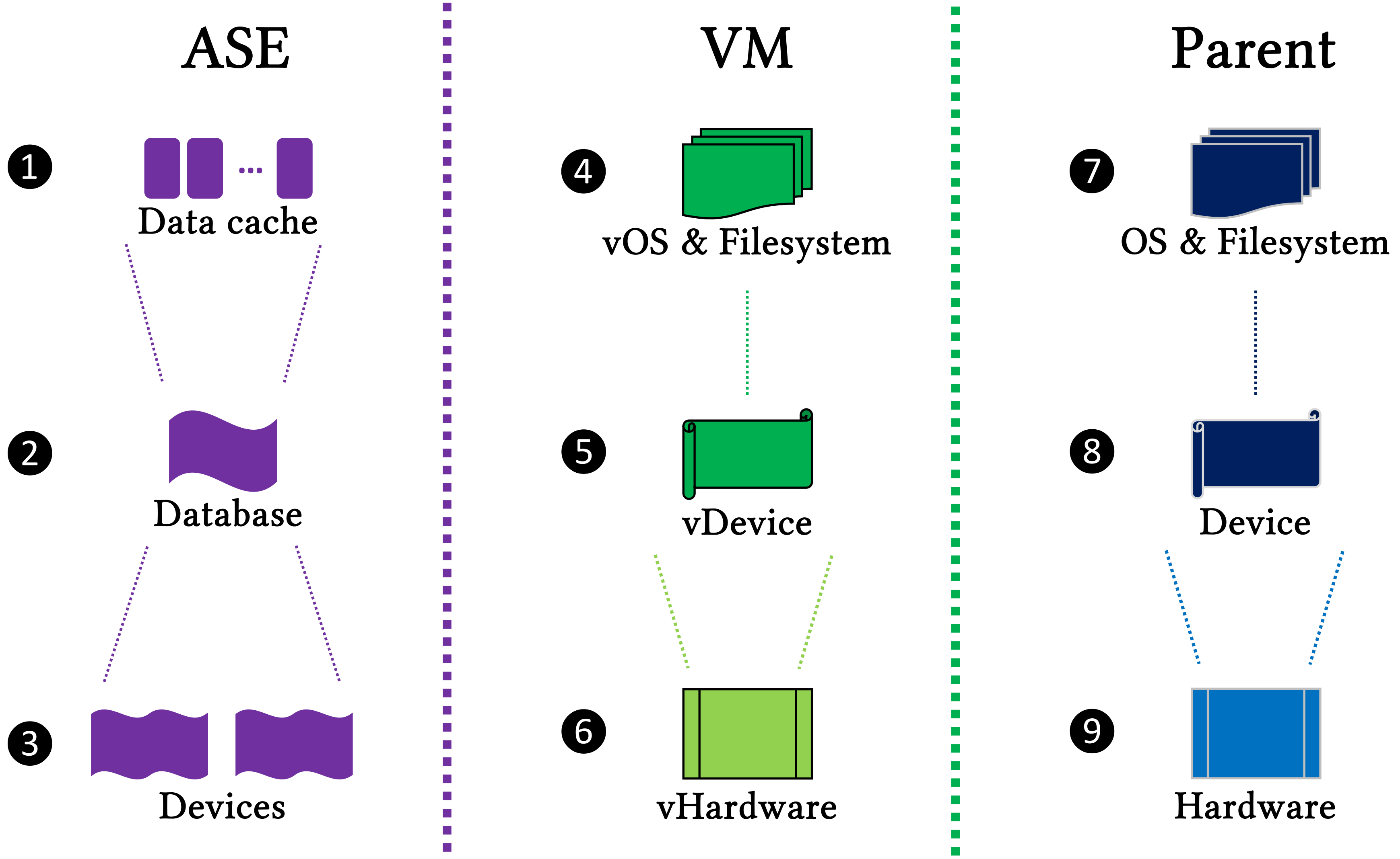
ASE



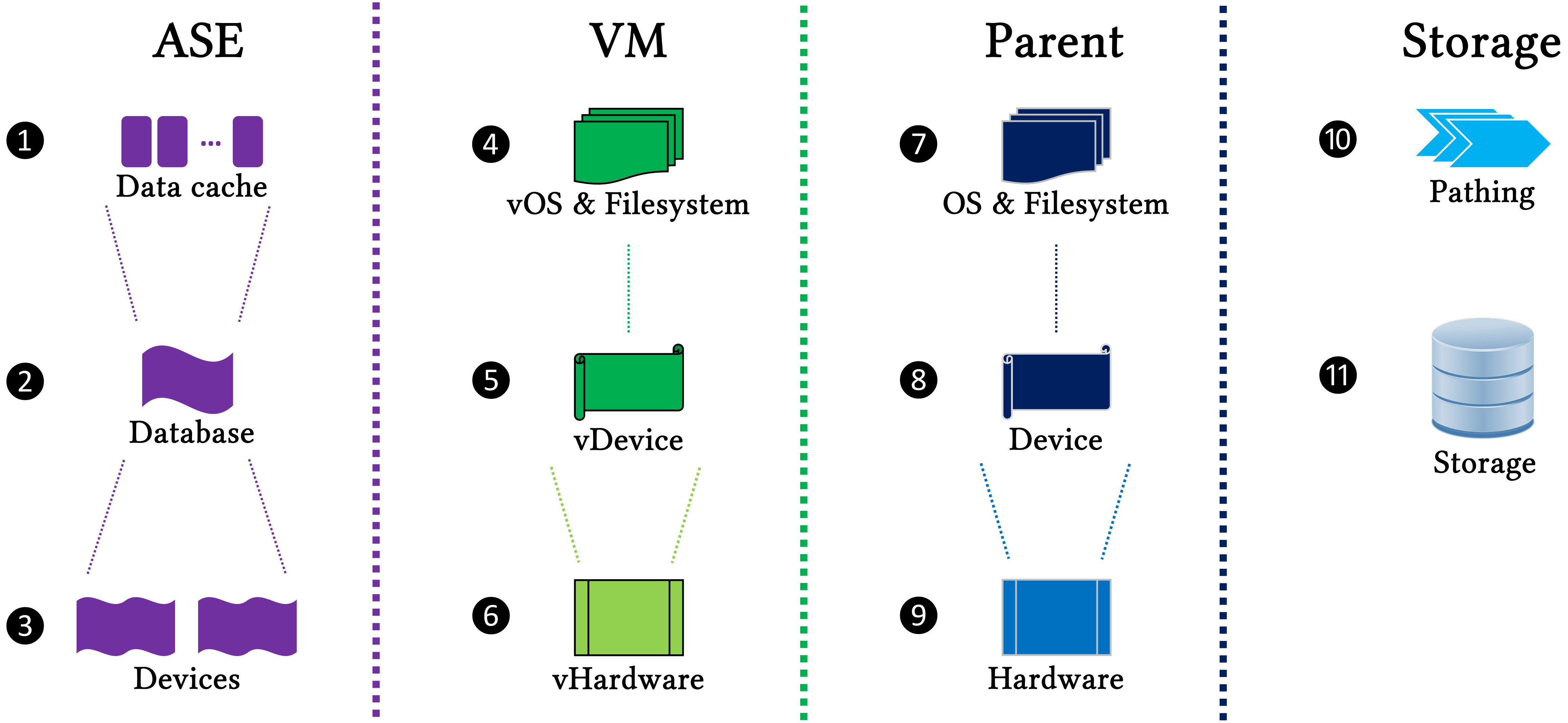
Follow the I/O



Follow the I/O



Follow the I/O



Where we focus our efforts

- The case study and this presentation mostly focus on ASE and (VM) OS
- Not just because these are what I (and DBAs) know best
- Mainly because these layers can be changed without affecting others
- Change to VM physical host affects all VMs on that host
- Change to storage array affects potentially hundreds of systems
- Technical difficulty: one size doesn't fit all for tuning measures
- Political difficulty: more approvals required

Before we get into the technical weeds

- Normally database discussions of concurrency are about locking
  - We are not talking about database locking anywhere today
  - Our focus is entirely on I/O concurrency

❶ ASE: data caches: I/O bandwidth, I/O size

- Large buffer pools will load cache with fewer trips
  - Never have four sizes in the same cache
  - Data can only ever use two sizes: 1-page, and largest
  - Log can use a third size but this must also be set using sp_logiosize
- If using data cache(s) for tempdb database(s), disable housekeeper
  - Can only be done in .CFG file

❶ ASE: data caches: I/O concurrency

- There is one spinlock per partition per cache
- More named caches = more spinlocks
  - Reduces memory available for any one object (one cache per object)
  - Better first answer: cache partitions
- Cache partitions = more spinlocks
  - Doesn't divide memory between objects
- Lockless data cache (ASE 16.0 SP02+)
  - If the cache meets the requirements for relaxed status
    - High hit rate, low volatility/replacements

❶ ASE: data caches: I/O latency

- No direct latency effects at the ASE data cache layer
- Indirect latency caused by cache misses
    - Cache miss = physical I/O = subject to all I/O stack latency
- Moral of the story: more memory
- Special note on tempdb in ASE data cache
    - Main benefit is to speed up **reads**
    - Writes still go to disk… eventually… we can tune this…

❶ ASE: data caches: I/O queues

- No direct queuing at the ASE data cache layer
- Indirect queuing effect caused by cache misses
  - Cache miss = physical I/O = subject to all I/O stack queuing
- Moral of the story: more memory

❶ ASE: data caches: I/O throughput

- Transactional memory (ASE 16.0 SP02+, Memscale, hardware)
  - Requires premium license
  - Requires hardware support on chip
- When available and enabled = +5% memory throughput
- Separate data caches for data and log
  - Allows pipelining: log write-ahead in one cache while data in other
  - Tempdb log cache too, even if not using tempdb data cache
    - Can’t directly bind tempdb system tables; indirectly via model

❶ ASE: data caches

- How would we know if there was an issue at this layer?
- ASE commands:

  sp_sysmon

  MDA tables

❷ ASE: databases: I/O bandwidth, I/O size

- Logiosize for dedicated buffer pool for logs
- User database: logiosize = 2 x @@maxpagesize
- Tempdb database: logiosize = 8 x @@maxpagesize
- Needs data cache buffer pool of same size as logiosize
  - ASE automatically sets 2 x @@maxpagesize if it finds 2-page buffer
- (Foreshadowing: **relationship between ASE & disk page sizes**)

❷ ASE: databases: I/O concurrency

- Some ASE optimiser decisions based on # of devices in segment
    - e.g. default parallel query, parallel index sort
    - Use more numerous and smaller devices per DB
    - Limit used to be 256 devices per database
    - Raised to 1269 in ASE 15.0
    - Limit is for unique devices; can still have multiple fragments each
- Multiple tempdb databases

❷ ASE: databases: I/O latency

- Few direct latency effects at the ASE database level
- Many indirect latency effects based on underlying devices

❷ ASE: databases: I/O queues

- Do staging / scratch work in dedicated databases
  - One transaction log per database
  - Beware cross-database transactions

❷ ASE: databases: I/O throughput

- Tempdb: separate data & log
- "global async prefetch limit" (save memory for regular I/Os)
- "i/o polling process count" (if set high, process kernel only)
- Good data space management = same data, fewer I/Os
  - exp_row_size
  - max_rows_per_page
  - Regular reorgs usually a sign of failure to do the above

❷ ASE: databases

- How would we know if there was an issue at this layer?
- ASE commands:

  sp_sysmon

  MDA tables

❸ ASE: devices: I/O bandwidth, I/O size

- Almost no direct bandwidth effects at the ASE device layer
  - Maybe deferred commit counts as bandwidth?
- Many indirect bandwidth effects based on ASE device attributes
  - Number
  - Size
  - Dsync, directio
  - File vs. raw
- Note: ASE writes are always one page at a time

❸ ASE: devices: I/O concurrency

- One spinlock per device
  - More numerous & smaller devices = more spinlocks
- "number of disk tasks" (threaded kernel only)
  - Commonly misunderstood
  - These are ***not*** the number of tasks performing I/Os
  - These are the number of tasks polling for completed I/Os
    - Frequently over-configured
    - Rule of thumb: 1 for every 16 engines/threads

❸ ASE: devices: I/O latency

- Ensure async I/O, platform-dependent
  - "allow sql server async i/o" (needed for any async including raw)
  - "enable hp posix async i/o" (HP, needed for file; makes raw **worse**)
  - "enable solaris async i/o mode" (Solaris, requires Solaris patch)
- ASE user database devices: dsync = false, directio = true
- ASE tempdb devices: dsync = false, directio = false
  - Definitely for tempdb log devices; maybe for tempdb data devices
- "i/o polling process count" (if set low; process kernel only)

❸ ASE: devices: I/O queues

- Direct I/O queues within ASE:
  - "disk i/o structures"
  - "max async i/os per [engine | server]"
  - Whichever is lower is the limit
  - Per-engine limit only relevant in process kernel
  - All require O/S config to support
- "housekeeper free write percent"
- "i/o batch size"

❸ ASE: devices: I/O throughput

- “global async prefetch limit”
  - Seldom helpful; save memory for regular I/Os instead
- “i/o polling process count” (if set high; process kernel only)

❸ ASE: devices

- How would we know if there was an issue at this layer?
- ASE commands:

  sp_sysmon

  MDA tables

• One last word about sp_sysmon •

```
Engine Utilization (Tick %)   User Busy   System Busy    I/O Busy        Idle
-------------------------  ------------  ------------  ----------  ----------
ThreadPool : syb_default_pool
Engine 0                       52.9 %         0.0 %      15.7 %      31.4 %
Engine 3                       43.1 %         0.0 %      19.6 %      37.3 %
Engine 6                       47.1 %         0.0 %      23.5 %      29.4 %
Engine 8                       23.5 %         0.0 %      33.3 %      43.1 %
Engine 9                       41.2 %         0.0 %      21.6 %      37.3 %
Engine 11                      60.8 %         0.0 %      21.6 %      17.6 %
Engine 12                      43.1 %         0.0 %      29.4 %      27.5 %
Engine 14                      33.3 %         0.0 %      29.4 %      37.3 %
```

- In a threaded kernel this output is meaningless
- "**At least one** I/O was outstanding during this % of clock ticks"

(Updated after the presentation; my thanks to Kevin Sherlock)

❹ VM: OS & filesystems: I/O bandwidth, I/O size

- Few direct RHEL OS disk bandwidth effects
- RHEL disk i/o scheduler *can* be set globally, but shouldn't be
  - One size does not fit all
  - Set in the next section, per "disk"

❹ VM: OS & filesystems: I/O concurrency

- ext4 forces single-threaded access to each inode; one inode per extent
- Cannot be disabled unless ext4 journalling disabled; discussed soon
- Can partially disable when ext4 journalling disabled:
  - Mount option `dioread_nolock`
  - Allows parallel reads of inode
  - Still single-threaded writers though
  - Known issue when combined with nodelalloc mount option
- Possibly not needed anymore in very recent RHEL kernels?

❹ VM: OS & filesystems: I/O latency

- ext4 journaling imposes 400% performance slowdown
- No-one disables it! Why not?!
- Filesystem journalling **is not** needed for ASE devices (and only those)
  - ASE fully preallocates; file sizes and metadata will never change
  - **Fully documented and supported by SAP and RHEL**
- `tune2fs -O ^has_journal /dev/sd[xyz]`
- This is *not* the same as disabling barriers, enabling writeback, etc.
  - Those tune ext4 journalling; we want to fully disable it

❹ VM: OS & filesystems: I/O queues

- fs.aio-max-nr = max async I/Os per process
  - Remember ASE threaded kernel = one process
  - Set very high, suggest 12,096,000
- fs.file-max = max file descriptions per process and globally
  - Also set higher than the default, suggest 6,291,456

❹ VM: OS & filesystems: I/O throughput

- ext4 updates timestamps for every file and directory access
  - This overhead isn't needed for ASE
- Set these mount point options:
  - ASE devices: noatime, nodiratime
  - ASE binaries, dump files: relatime, nodiratime
- Bad things happen when ASE page size is larger than LUN page size
  - Mostly only a problem for ASE 2K
  - Make sure ASE page size is an integer multiple of LUN page size

❹ VM: OS & filesystems

- How would we know if there was an issue at this layer?
- OS commands:
  iostat
  sar
  perf
  df
  mount
  dumpe2fs

❺ VM: OS devices: I/O bandwidth, I/O size

- /sys/block/sd[xyz]/queue/rotational
  - RHEL device geometry, 1 = mechanical, 0 = SSD/NVMe/flash
  - Most non-mechanical drives do not correctly report this
  - Not persistent, must be reset after every reboot
  - Required to enable some internal RHEL tuning
- Relevant for both raw and file

❺ VM: OS devices: I/O concurrency

- As with ASE devices, more numerous and smaller devices help
- Engineering teams don't like this
  - Usually limits of how many LUNs can be attached
- Relevant for both raw and file

❺ VM: OS devices: I/O latency

- /sys/block/sd[ccc]/queue/scheduler
  - RHEL disk i/o scheduler *can* be set globally, but shouldn't be
  - Reorders disk I/Os, intended to minimise mechanical seek time
  - RHEL default = **`deadline`**; ASE does much better with **`noop`**
  - Not persistent, must be reset after every reboot
- Relevant for both raw and file

❺ VM: OS devices: I/O queues

- /sys/block/sd[ccc]/queue/nr_requests
  - Per-device limit on maximum outstanding I/O requests
  - Default is only 128! Set instead to 1024
  - Not persistent, must be reset after every reboot
- Relevant for both raw and file

❺ VM: OS devices: I/O throughput (1)

- /sys/block/sd[ccc]/queue/add_random
  - Modern O/S collects entropy from as many sources as possible
  - RHEL collects entropy from low-level I/O events
  - In very high I/O loads this becomes measurable overhead
  - Suggest disabling it for all LUNs used by ASE
  - Not persistent, must be reset after every reboot
- Relevant for both raw and file

❺ VM: OS devices: I/O throughput (2)

- Prefetch helps sequential reads and hurts random reads if not careful
- `blockdev --setra` *`value`* `/dev/sd[xyz]`
  - RHEL is smart enough to only trigger this for sequential reads
  - Value is in 512-byte disk sectors; 2 = 1Kb, 512 = 256Kb, 4096 = 2Mb
  - Benefit is greater for multiple readers than single-threaded
  - Not persistent, must be reset after every reboot
- Relevant for both raw and file
  - Excellent idea to tune higher for load database dump files

❺ VM: OS devices

- How would we know if there was an issue at this layer?
- OS commands
  iostat
  sar
  perf
  df
  mount
  dumpe2fs
  blockdev --report
- Also by inspection of the /sys/block/sd[xyz]/queue/* files

Now we're into the technical weeds

- This is usually as far as I've taken it prior to this case study
- Normally we can't make any changes further down the stack
- This client wanted the lot
- I don't have all the answers but I'll share what I do have
- I've stopped answering "How would we know if there's a problem here"

❻ VM: disk controllers

- Every O/S has drivers for its hardware
- The disk controllers in a VM are themselves virtual hardware
- Queue depths are tuneable; defaults are too low
  - This is pointless unless queues at all other layers are also tuned
  - Otherwise latency is just moved to a different layer
- Depending on OS, VM, and controller, other settings may be tuneable

❻ VM: disk controllers

- RHEL in VMware is almost certainly using PVSCSI controllers
  - If not, it should be! Check!
- Two options to tune PVSCSI controllers in RHEL
  - Create or edit /etc/modprobe.d/vmw_pvscsi.conf

    options vmw_pvscsi cmd_per_lun=254 ring_pages=32
  - Add to /etc/grub.conf (and regenerate boot file!!!)

    vmw_pvscsi.cmd_per_lun=254

    ring_pages=32

❼ Physical host: OS

- This is the OS on the physical parent host of a VM
- For VMware it is ESXi
- Many tuneable options here! Seldom tuned much, why?
- Database servers work differently to appservers and file servers
- If I were CIO, I would not combine these on physical servers
  - i.e. one VM farm each for database, app, and file servers
- Everything done to a host has to be done here too
  - Patching, tuning, sizing, hugepages

❽ Physical host: storage layout & devices

- Many possible options for how VMware allocates and presents storage
- Best for database servers
  - Each LUN presented to guest OS = one VMDK
  - Each VMDK = one file on one datastore
  - VMFS should be VMFS6+

❾ Physical host: disk provisioning

- VMware disks can be provisioned differently per VMDK

Provisioning type	Preallocated?	When zeroed?
Thin	N	Run-time
Zeroed Thick (default)	Y	Run-time
Eager Zeroed Thick	T	Create-time

- Why do this? To allow disk over-provisioning
- Not really an issue for ASE because ASE fully preallocates storage
  - Could be an issue for database backups

❽ Physical host: storage layout & devices

- VMware sets disk controller I/O queues per VM, and per datastore
- If best practice followed (one datastore per VM) these are the same
  - Still, explicitly tune both
- Check your HBA hardware documentation, defaults are pitifully low
  - e.g.: Emulex LightPulse LPe32000 default queue depth = 30 (!)

esxcli system module parameters set –p lpfc[n]_lun_queue_depth=254 –m lpfc

esxcli system core device set –O | --sched-num-req-outstanding 254 –d device_id

❽ Physical host: storage layout & devices

- VMware can present up to four virtual disk controllers to a VM
- Usually only one
- Each controller has its own queue and latency
- More controllers = more queues = more throughput
- Care needed to load balance the mount points between controllers

❾ Physical host: hardware

- DO NOT OVERCOMMIT MEMORY
  - Test very carefully if overcommitting CPU
- You can get away with this with fileservers, and with some appservers
  - Definitely not memory with database servers
- Better to fully reserve/preallocate memory (and ideally CPU)
- Disable transparent page sharing (shares memory between VMs)
- VMware Latency Sensitivity – counterintuitive
  - SAP and all other database vendors recommend “Normal” (default)

9 Physical host: hardware

- Firmware and BIOS need to be patched too, but seldom are
- Make sure HBAs and NICs are installed in the proper slots!
  - Dual channel 16Gb/s cards require 32Gb/s slots
  - PCIe Gen2 x8, or PCIe Gen3 x4
- HP ProLiant DL380 Gen 9 has some Gen 3 x16, the rest are Gen3 x8
  - This means 64Gb/s HBAs would be choked in some slots
- This site was OK but some have not been

❾ Physical host: hardware

- If physical CPUs support pdpe1gb flag for 1Gb “extrahuge” hugepages
  - Present the most recent vCPUs to guest OS
  - Configure CPU/MMU Virtualization to Automatic
  - Advanced Options, Edit Settings

    sched.mem.lpage.enable1GPage = “TRUE”

    Mem.AllocGuestLargePage = 1

    LPage.LPageDefragEnable = 1

    LPage.LPageAlwaysTryForNPT = 1

⑩ Paths to storage

- HBAs are rated for bandwidth and # of channels
- Emulex LPe32000 = 2 channels @ 16Gb/s bandwidth = 32Gb/s
- Standard practice is to have multiple cards with multiple channels
- This is multipathing
- Check the multipathing policy!
  - Many defaults treat additional paths as failover, not load balancing
- Check every FC switch between host and storage
  - If even one is not consistent in bandwidth the whole path suffers

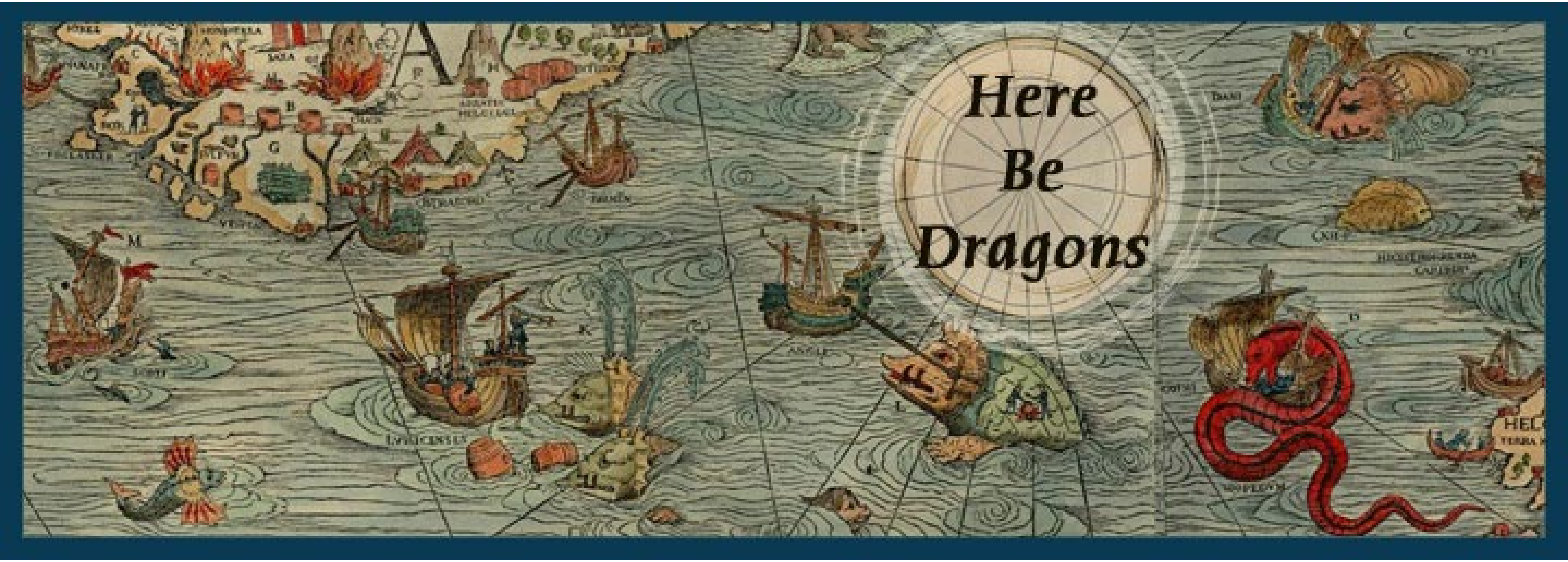
❿ Paths to storage

- VMware multipathing plugins are apparently controversial
  - VMware best practice = HPP (high performance plugin)
  - Pure Storage best practice = NMP (native multipathing plugin)
    - Older, intended for mechanical spinning disk

❿ Paths to storage

- VMware path selection policy = which path to use for a given I/O
- roundrobin policy = alternates between paths
  - When to switch is configurable
  - Best practice c. 2017 was to set policy=rr,iops=1
  - Strictly alternate each I/O… even if one path is much slower
- Best practice c. April 2022 is to measure latency and use the best path
  - set policy=latency
- Secret practice: policy=rr,iops=0 (switch I/O by least queue depth)

⓫ Storage array



⓫ Storage array

- Pure Storage claims nothing needs to be tuned
- It's just one giant slab of disk
- It still has firmware that needs updating
- The "don't tune it" best practice needs to be regularly reviewed
- … just in case something turns out to need to be tuned after all
- Probably we aren't allowed to tune anything here anyway

References

- Broadcom Emulex Drivers for VMware ESXi, User Guide v14.0, as of 21/09/2021, https://docs.broadcom.com/doc/elx_DRVVM-UG140-100.pdf
- Pure Storage Linux Recommended Settings, as of 08/11/2021, https://support.purestorage.com/@api/deki/pages/1979/pdf/Linux%2bRecommended%2bSettings.pdf?stylesheet=default
- Pure Storage Understanding VMware ESXi Queueing, as of 14/02/2022, https://blog.purestorage.com/purely-technical/understanding-vmware-esxi-queuing-and-the-flasharray-2/

References

- Red Hat Enterprise Linux 7 Performance and Tuning Guide, as of 18/12/2021, https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/performance_tuning_guide/sect-red_hat_enterprise_linux-performance_tuning_guide-storage_and_file_systems-configuration_tools
- Red Hat Enterprise Linux 7 – Storage Administration Guide, as of 20/12/2021, https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/storage_administration_guide/ch-ext4

References

- SAP KBA 2987324 – SAP ASE 16.0 SP04 Supported Operating Systems and Versions, as of 17/11/2021, https://launchpad.support.sap.com/#/notes/2987324 (requires SAP support login)
- SAP KBA 2489781 – SAP ASE 16.0 SP03 Supported Operating Systems and Versions, as of 10/12/2021, https://launchpad.support.sap.com/#/notes/2489781 (requires SAP support login)
- SAP Sybase Adaptive Server Enterprise on VMware vSphere: Essential Deployment Tips, as of 20/12/2021, https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/sap-sybase-adaptive-server-enterprise-on-vmware-vsphere.pdf, (this is written for SAP ASE 15.7 on VMware vSphere 5.0 but all advice is still sound for ASE 16.0 and vSphere 7.0.x, confirmed in the vSphere 7.0 Performance Best Practices document below)

References

- VMware vSphere 7.0 Performance Best Practices, as of 28/01/2021, https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/techpaper/performance/vsphere-esxi-vcenter-server-70-performance-best-practices.pdf
- (VMware) Large-scale workloads with intensive I/O patterns might require queue depths significantly greater than Paravirtual SCSI default values, as of 14/02/2022, https://kb.vmware.com/s/article/2053145
- (VMware) Changing the queue depths for QLogic, Emulex, and Brocade HBAs, as of 14/02/2022, https://kb.vmware.com/s/article/1267

References

- (VMware) Best Practices for Performance Tuning of Latency-Sensitive Workloads in vSphere VMs, as of 14/02/2022, https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/techpaper/vmw-tuning-latency-sensitive-workloads-white-paper.pdf (this dates to 2013 but none of it is superseded or incorrect today)
- (VMware) Thin vs thick provisioning and performance impact, as of 13/03/2023, https://www.vmware.com/pdf/vsp_4_thinprov_perf.pdf (this dates back to 2009 but none of it is superseded or incorrect today)

Q & A, and thank you

Joe Woodhouse
joe.woodhouse@primadonnaconsulting.com

Too busy putting out fires to reduce your toil?
Answering the on-call phone too often?
Drowning in technical debt?

Joe is a freelance consultant available through Prima Donna Consulting and can be engaged flexibly.

Not a lawyer – no charge for emails!