Let Latency Be Gone!

How NVMe, SCM and RDMA are tackling latency

PS C:\> $env:USERNAME
Alex Bytes

- Senior Infrastructure Specialist
  - Looks after ‘all the things’
  - Attempting to avoid going grey prematurely

- Blogger of IT
  - **ByteSizedAlex.com** - Byte sized for when a nibble just isn't enough!
  - **Twitter - @ByteSizedAlex**

Get-Agenda

- What is latency and why does it hurt performance?
- How are technologies like NVMe, SCM and RDMA addressing this issue
- Examples and study references throughout to give you an idea of the improvement you can realise through adoption

This isn't a technical deep dive, the goal is to excite your interest in these technologies and get a feel for the results they can deliver

What is latency?

- Generally: *"A measure of the time delay experienced by a system"*
- Network specific: *"A measure of the time delay required for information to travel across a network"*
- If you need to explain it to a non-technical person use the analogy of a phone call. The latency is the time taken for the other person to pick up and say hello after you start the call

- For our discussion we are concerned with two types of latency:
  - Network latency
  - Storage latency
    - Whether it be to local disk or remote storage over a fabric

Network (fabric) latency will impact storage latency when we consider remote storage

Latency Impact

Online gamers will know the affects of latency well – it really hurts your K/D ☹ but you're probably more interested in 'business' stuff

- Amazon estimate 1% loss for every 100ms of latency
- A study suggested 0.5 seconds additional latency on search page generation time dropped traffic by 20%
- One report suggested a stock broker could lose $4 million in revenue per millisecond if their electronic trading system is 5 milliseconds behind the competition

Regardless of whether these are 100% accurate it is safe to say that latency hurts business. Those with high performing, lower latency platforms have a competitive edge.

How can we reduce latency and improve performance?

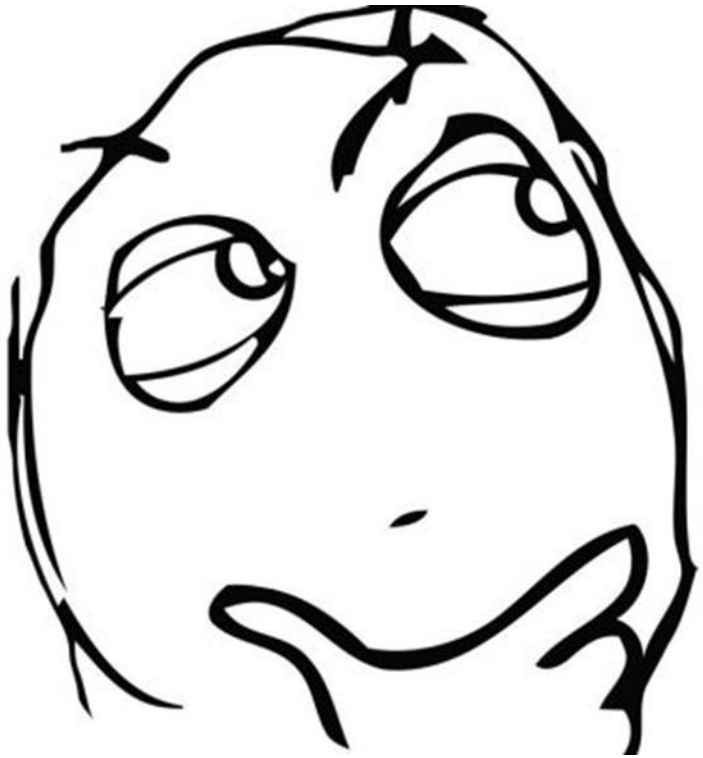
- We will look at 3 technologies which can enable drastic reductions in latency
  - NVMe – Non-Volatile Memory Express
  - SCM – Storage Class Memory
  - RDMA – Remote Direct Memory Access

- Not only is latency reduced but in the case of NVMe and RDMA so is CPU consumption

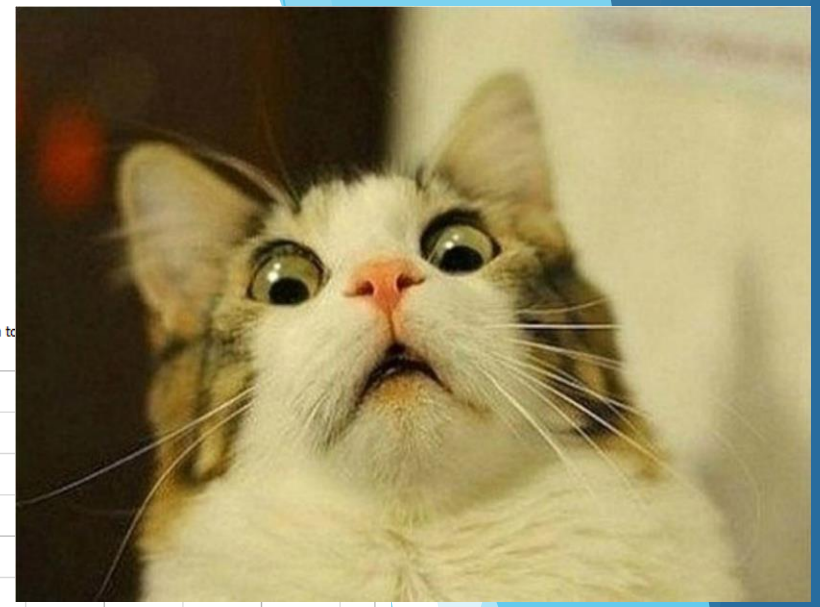
Lower latency, faster systems all with less CPU consumed, who wouldn't want that!

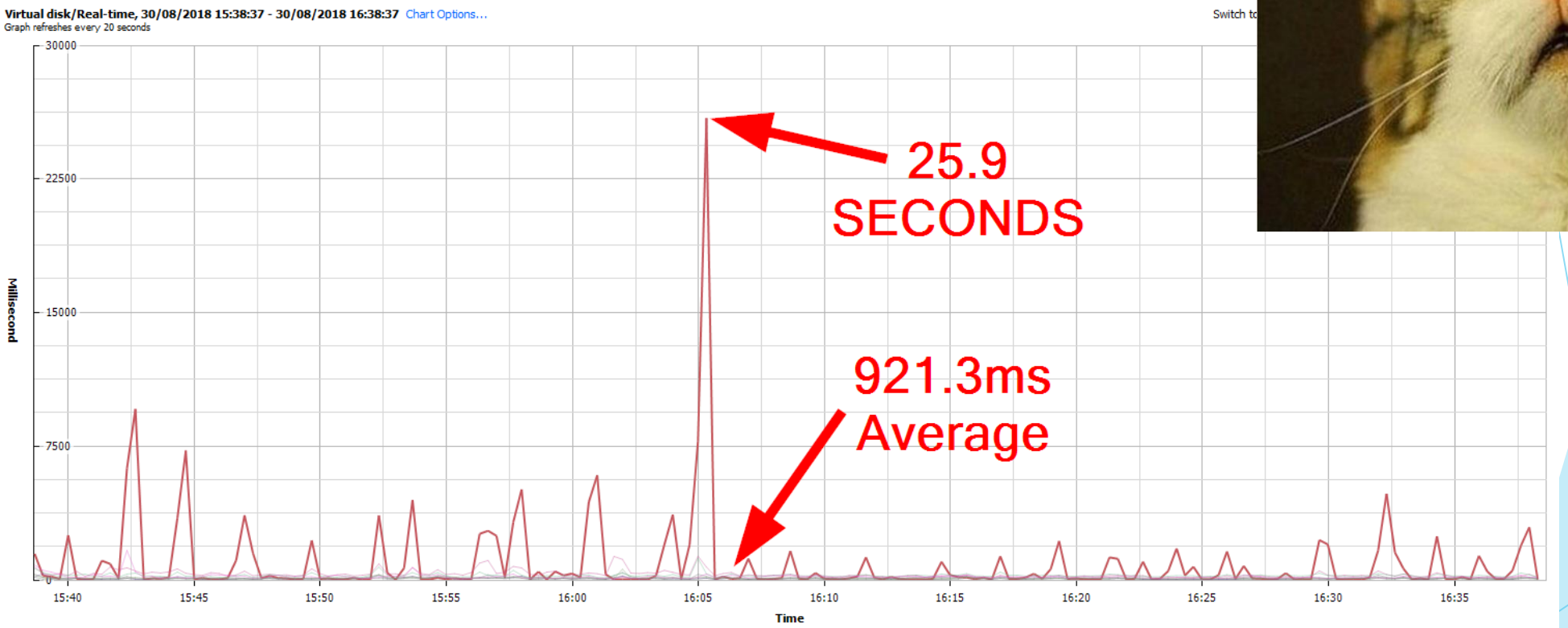
What storage latency do you see in your environment?

- > 10ms
- 5 <-> 10ms
- 1 <-> 5ms
- < 1ms



If you think your latency is bad…





Performance Chart Legend

Key	Object	Measurement	Rollup	Units	Latest	Maximum	Minimum	Average
	scsi0:0	Write latency	Average	Millisecond	27	181	2	31.361
	scsi0:3	Write latency	Average	Millisecond	2	25934	0	921.339
	scsi0:1	Write latency	Average	Millisecond	104	1313	89	287.4
	scsi0:2	Write latency	Average	Millisecond	27	395	6	64.472
	scsi0:0	Read latency	Average	Millisecond	53	324	8	73.15
	scsi0:2	Read latency	Average	Millisecond	43	579	29	78.028

NVMe – Non-Volatile Memory Express

- A modern storage protocol built around PCIe connected SSD storage
- Instruction set drastically reduced when compared to SCSI/AHCI
  - Simplified to remove a lot of the bloat in legacy protocols
- Throughput scales with PCIe generation and lane count (1 -> 16 lanes)

NVMExpress.org describes it as such –

"NVM Express is a scalable host controller interface designed to address the needs of Enterprise and Client systems that utilize PCI Express based solid state drives. The interface provides optimized command submission and completion paths. Additionally, support has been added for many Enterprise capabilities like end-to-end data protection (compatible with SCSI Protection Information, commonly known as T10 DIF, and SNIA DIX standards), enhanced error reporting, and virtualization"

NVMe vs SAS & SATA

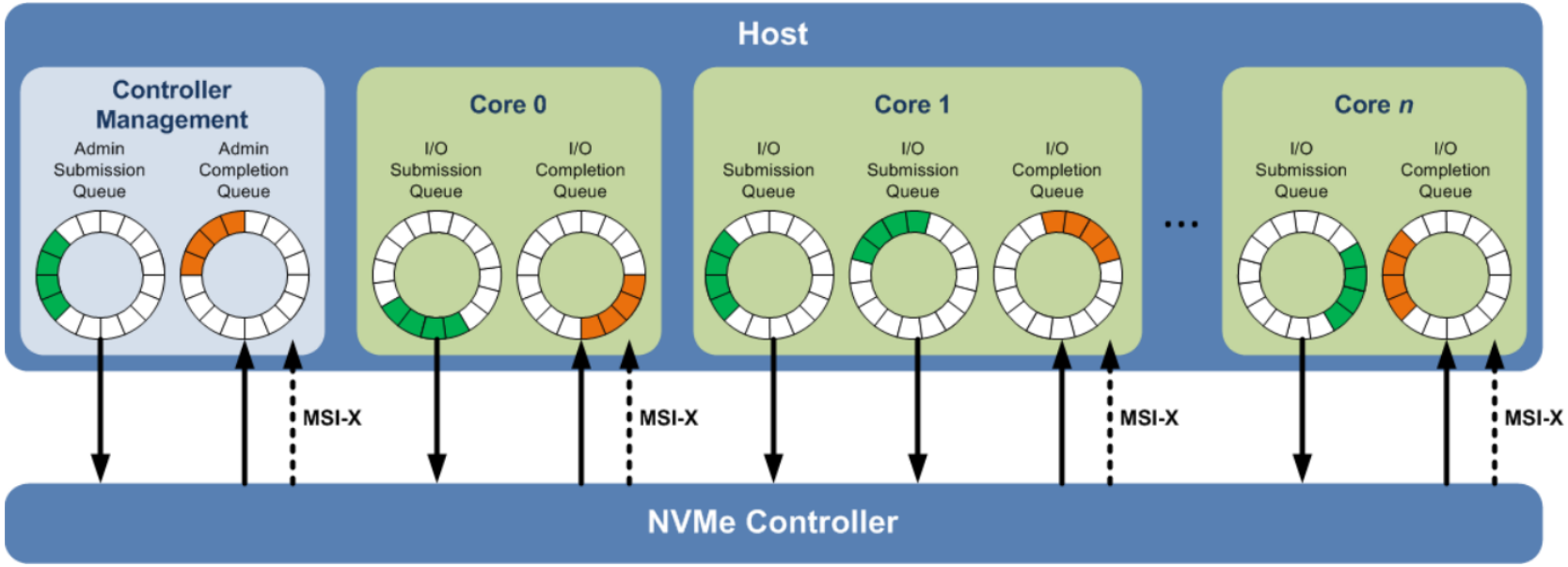
	I/O Queues	Commands Per Queue	Bandwidth
SATA/AHCI	1	Up to 32	6Gbps
SAS	1	Up to 254	6 & 12Gbps
NVMe	65,535	64,000	PCIe Gen and Lane Dependent

Generation	x1	x2	x4	x8	x16
PCIe 3	**1GB/s**	**2GB/s**	**4GB/s**	**8GB/s**	**16GB/s**
PCIe 4	**2GB/s**	**4GB/s**	**8GB/s**	**16GB/s**	**32GB/s**
PCIe 5	**4GB/s**	**8GB/s**	**16GB/s**	**32GB/s**	**64GB/s**

PCIe lanes are bi-directional; Numbers are rounded theoretical for simplicity
PCI-SIG is targeting Q1 2019 for completion of the PCIe 5.0 specification

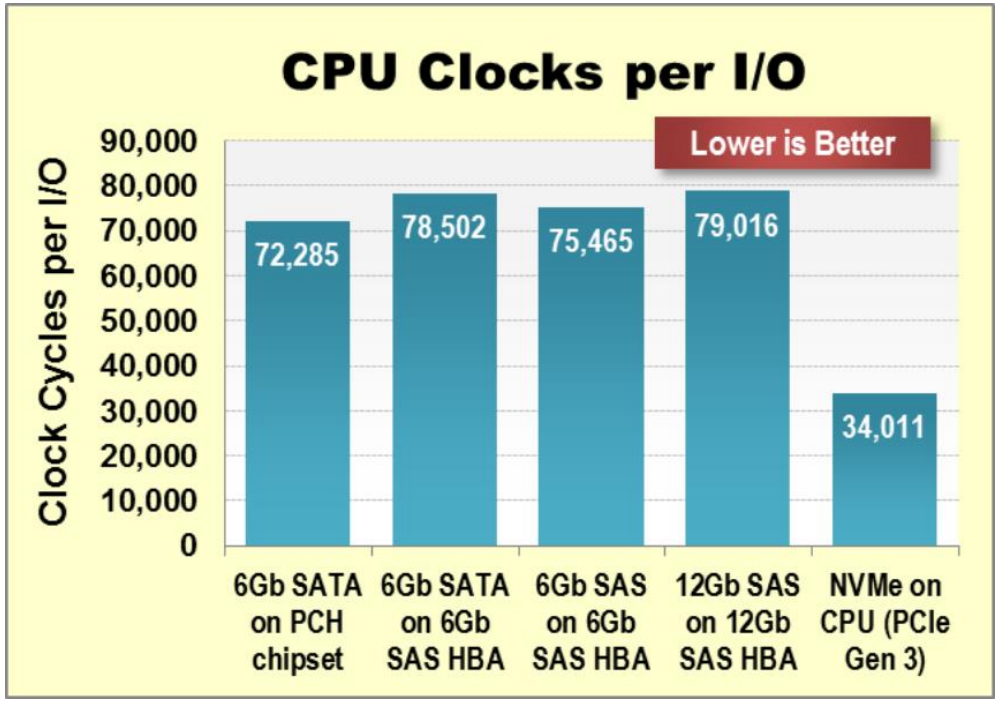
NVMe Queues

- Each CPU core can host a submission/completion queue pair for each NVMe device (queue count is controller specific)
- Message Signaled Interrupts leveraged (software interrupt as opposed to dedicated circuit/trace)



Reference: *SNIA - The Performance Impact of NVM Express and NVM Express over Fabrics*

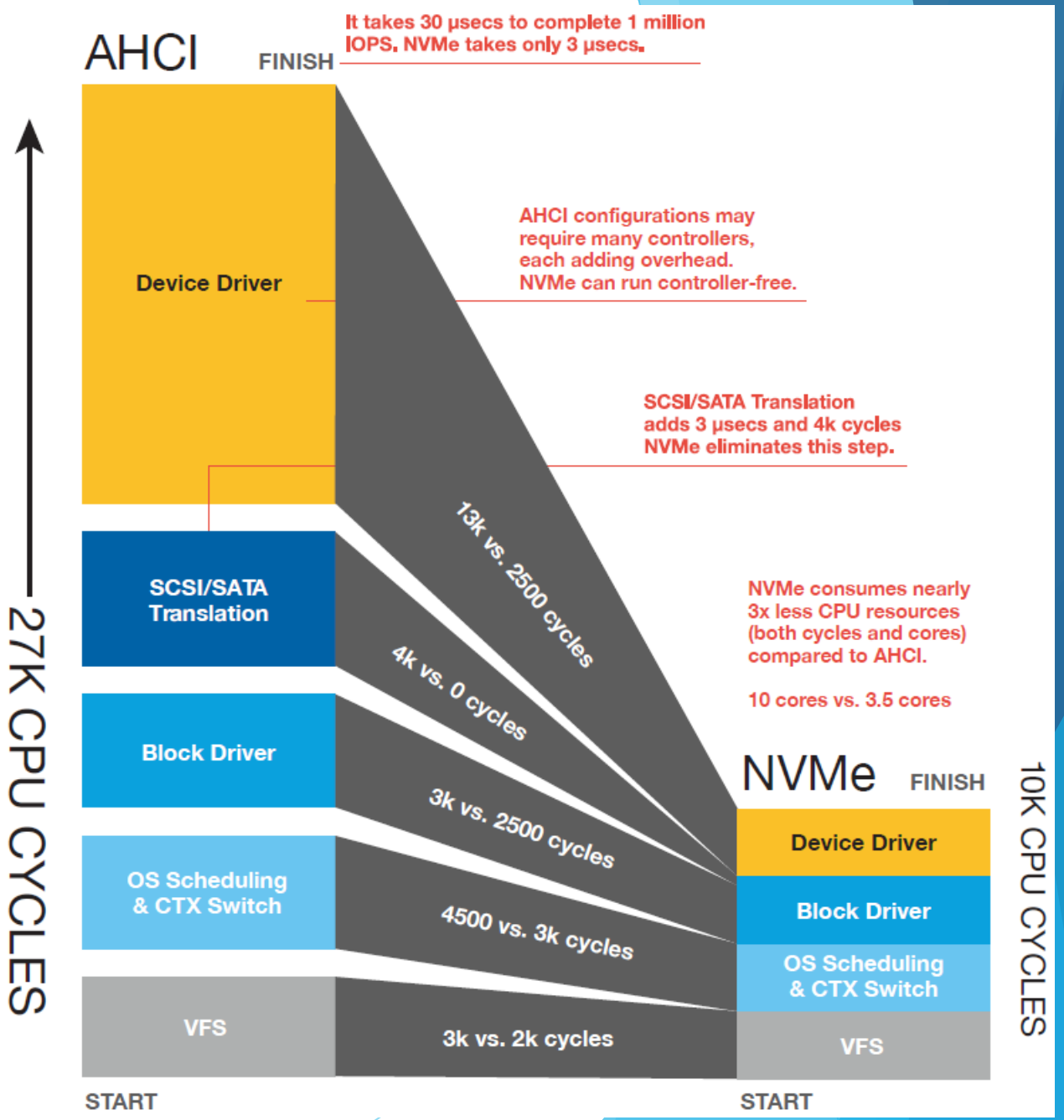
CPU Cycle Reduction



Numbers vary but typically claimed to half the cycles required to process an IO

References: *nvmexpress.org - NVMe Overview*

NVMe—Performance for the SSD Age - Seagate



SCM – Storage Class Memory

PCIe is still slow compared to the memory bus – directly attach persistent storage to the main memory bus (currently DDR4)!

- Persistent storage at close to RAM latencies (nanoseconds)
- PMEM – Persistent Memory
  - NVDIMMs
    - NVDIMM-N - DRAM backed by flash memory
    - NVDIMM-F – Flash on DIMM module
    - NVDIMM-P – Multi-function
  - Intel 3DXPoint Optane
- Devices can operate in either classical 'Block Addressable' mode or in DAX (Direct Access) mode where they are treated as 'Byte Addressable' memory devices

NVDIMM Performance

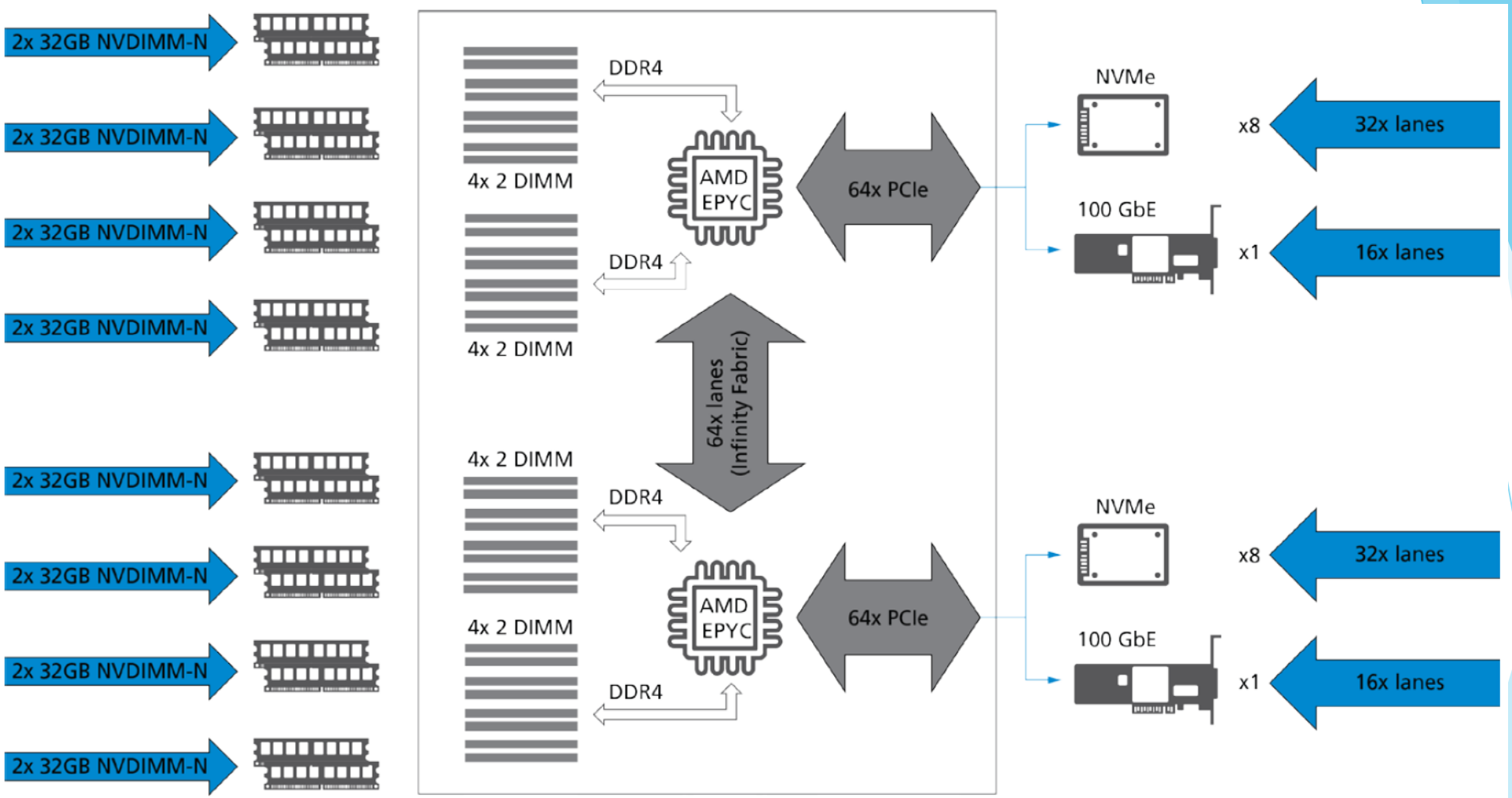
- HPE example using 8GB NVDIMM module in block addressable mode (note that DAX mode results would be even better)

Parameter	NVDIMM vs SAS SSD	NVDIMM vs PCIe Accelerator
IOPS	34x Increase	24x Increase
Bandwidth	16x Increase	6x Increase
Latency	81x Lower	73x Lower

- Micron demonstrate the difference between Block and DAX performance

Storage	IOPs	Delta (vs SSD)	MB/s	Latency (ns)
NVMe SSD	14,553	Baseline	56.85	66,632
Block NVDIMM-N	148,566	10.2X	580.34	6418
DAX NVDIMM-N	1,112,006	76.4X	4343.78	828

Reference: *HPE Persistent Memory for HPE ProLiant Servers*



"If each 32GB NVDIMM-N is capable of around 1.1 million IOPS; Using 16x 32GB NVDIMM-Ns in our example, we have the potential to reach 17 million IOPS in the NVDIMM-N layer alone. When added to the 16 NVMe SSD using 11TB Micron 9200 SSDs (800,000 read IOPS per SSD), we can produce an additional 9.6 million IOPs at the NVMe layer. The combined NVDIMM-N and NVMe IOPS exceed 26 million in 2RU! Real-world performance is, of course, application, workload and file system dependent."

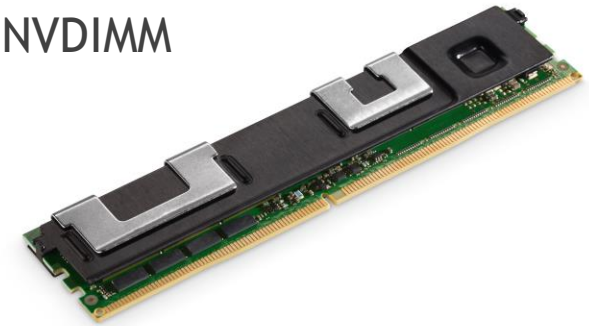
Reference: *AMD's EPYC Architecture Benefits Storage-Centric Solutions*

SCM Usage Examples

- Microsoft support the use of SCM in Storage Spaces Direct (S2D) to accelerate the write caching tier
  - Combined with SMB Direct/MultiChannel (their buzz word for RDMA) to enable low latency between nodes you can get really good performance
- SQL server can use SCM either as a block addressable device or via DAX mode for Tail of Log Writes (ToLW)
  - Dell provide an example using SCM for ToLW –
    - Increase of roughly 42% in Transactions Per-Second
    - Increase of roughly 43% in Batch Requests Per-Second
- 3PAR Storage platforms offer SCM as a caching layer – 50% latency reduction, 80% increased IOPS according to HPE blog

Intel Optane DC Persistent Memory

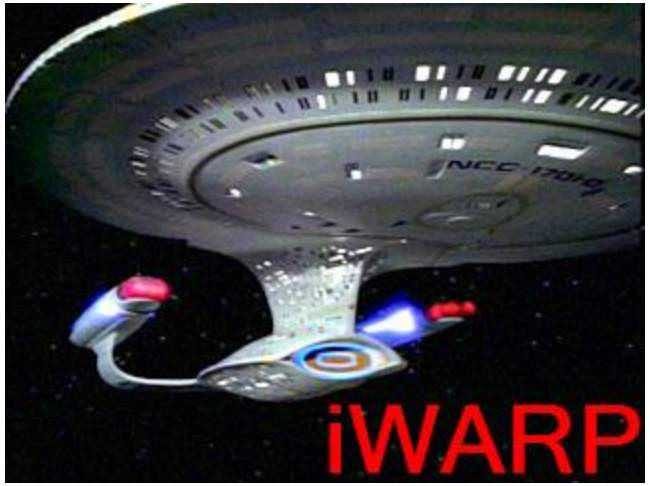
- As the name suggests, it's persistent
- Intel indicated DRAM 'like' speeds – until publically available we can't put a value to this (Intel state 'broad availability' 2019)
- No Power Loss Protection (PLP) required unlike NVDIMM
- Capacities per module –
  - 128GB
  - 256GB
  - 512GB



- With these capacities you could feasibly deploy ~6TB PMEM in a dual socket system alongside ~1.5TB RAM (3TB RAM if using M series Xeon)
  - Scalable Xeon supports 2 DPC, DRAM and Optane can be deployed in tandem
    - 6 x 512GB Optane per Socket
    - 6 x 128GB DDR4 per Socket

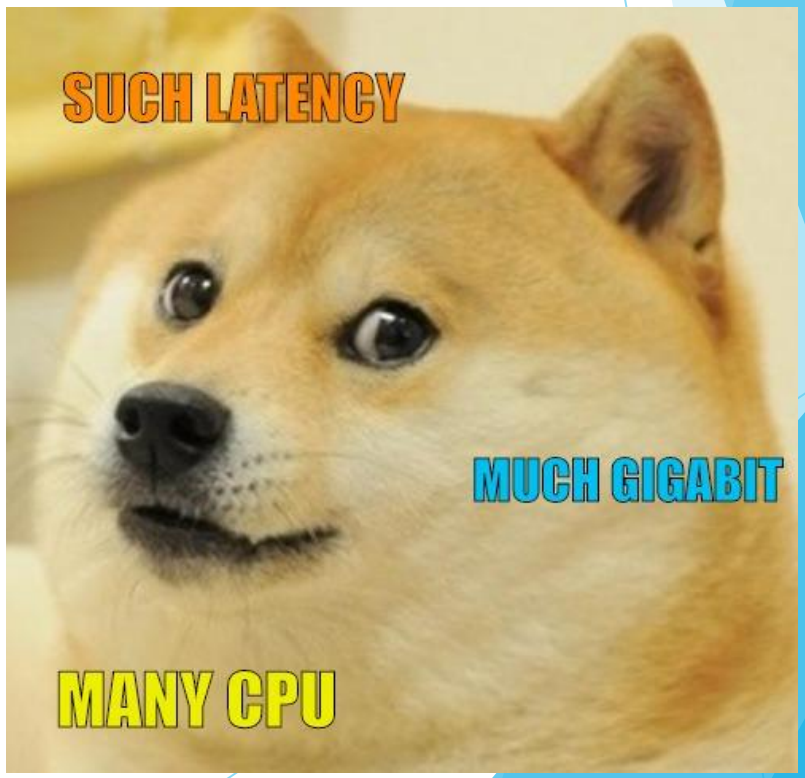
RDMA – Remote Direct Memory Access

- Not a new technology but being adopted in Enterprise. Many hyperscalers and cloud providers run Ethernet based RDMA
- Ethernet based RDMA is offered in two competing flavours
  - **iWARP** – **I**nternet **W**ide **A**rea **R**emote Direct Memory Access **P**rotocol
  - **RoCE** (v1 and v2) – **R**emote Direct Memory Access **O**ver **C**onverged **E**thernet



Why RDMA?

- Massive reduction in CPU overhead when processing network traffic
- Increased throughput (more gigabits per second)
- Huge reduction in latency

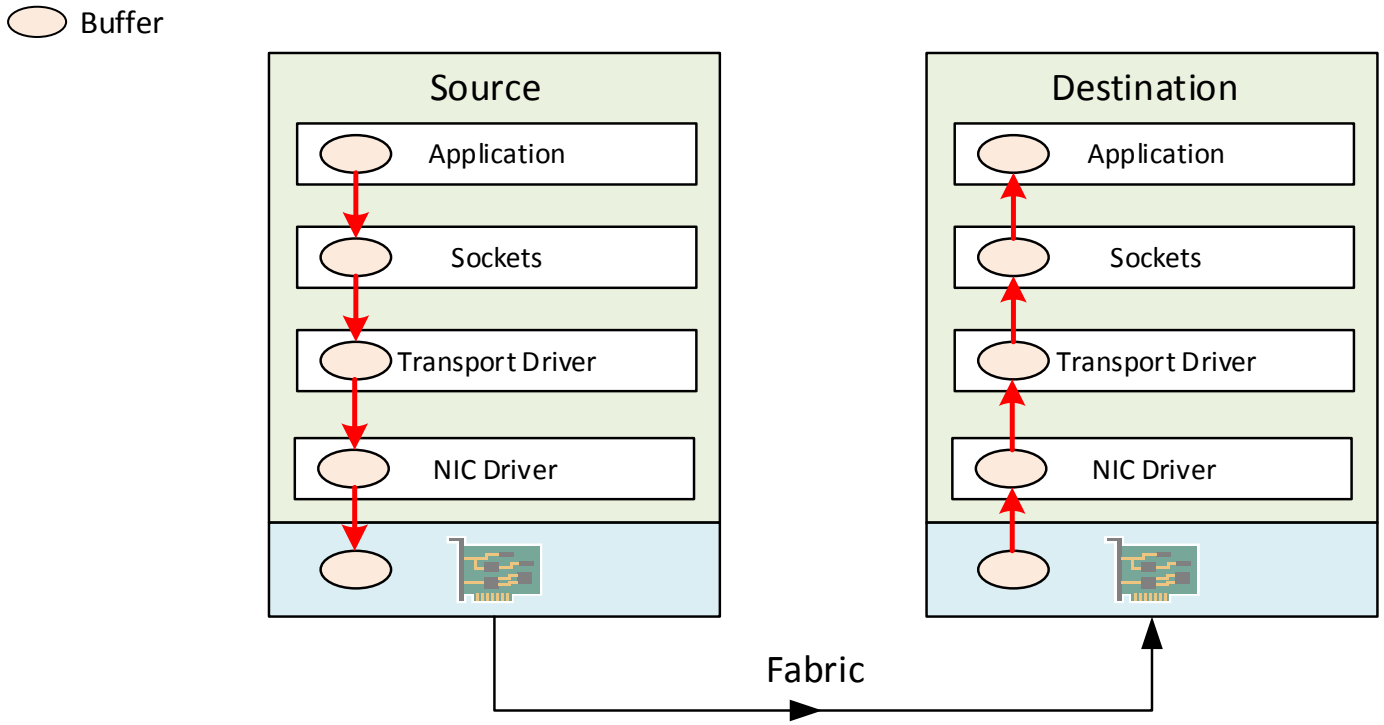


What is DMA?

- Implemented to free CPU from data movement –CPU time is lost doing data move on slower buses/context switching
- Classically a discrete microcontroller was added to a system to facilitate DMA
  - Known as a DMA Controller (DMAC)
- In time this was moved into the PCH (Platform Controller Hub) and then into individual devices which ran their own
- Types of DMA:
  - Device to Device
  - Device to Memory
  - Memory to Memory

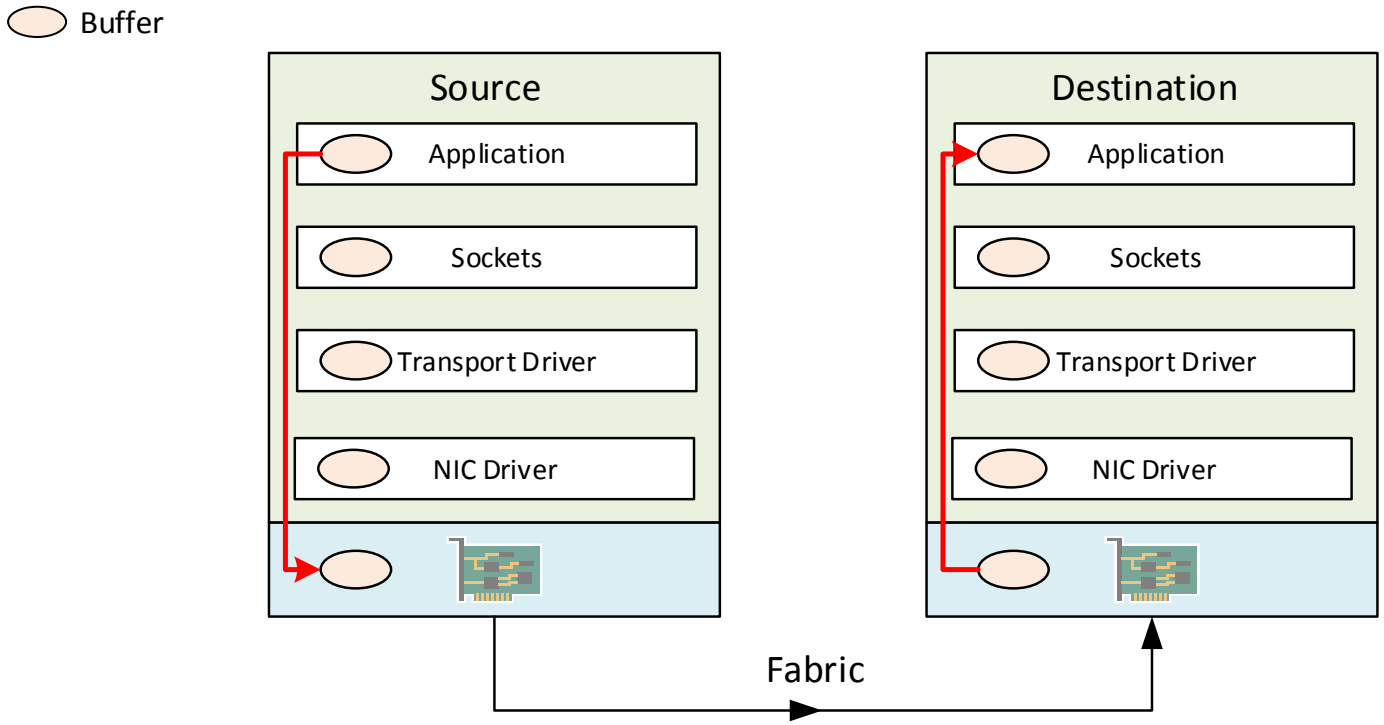
Traditional TCP Communication

- Note the many Buffer-To-Buffer copies – requires CPU processing time and incurs latency



RDMA Enabled Copy

- Buffer copying eliminated – direct transfer from Application memory space through RDMA enabled NIC to destination Application memory space



Microsoft Finding…

- Microsoft provide an example of network traffic CPU consumption:
  - 2 Cores Consumed on 10Gb Ethernet
  - 8 Cores Consumed on 40Gb Ethernet

- RDMA virtually eliminates this overhead which is vital for performance scaling and ensures CPU cores are doing actual ‘work’

- Microsoft already support 10, 40, 50 and 100Gb Ethernet
  - 400Gb and 1Tb are in testing – RDMA will be critical in ensuring CPU cores are not saturated at these speeds

How many cores do you think could be lost at those speeds?

RDMA will (I believe) in time become another ‘standard’ offload we expect on NICs

(Once the whole VHS/Betamax war between iWARP and ROCE is over)

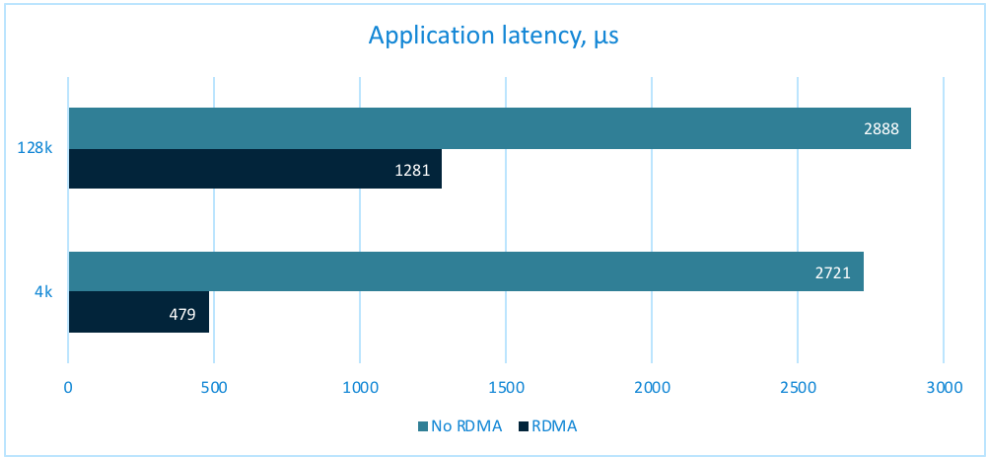
Reference: *Microsoft - Deep dive on Windows Server 2019 updates*

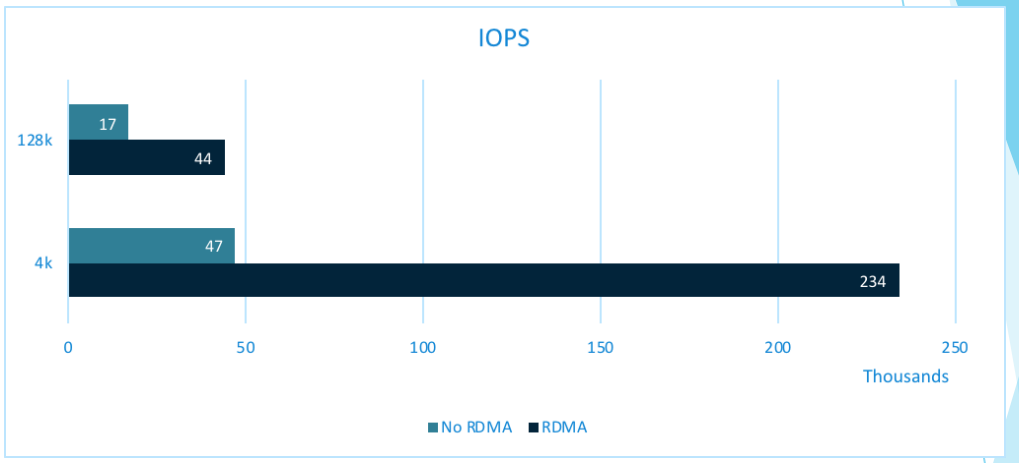
Microsoft RDMA vs TCP/IP

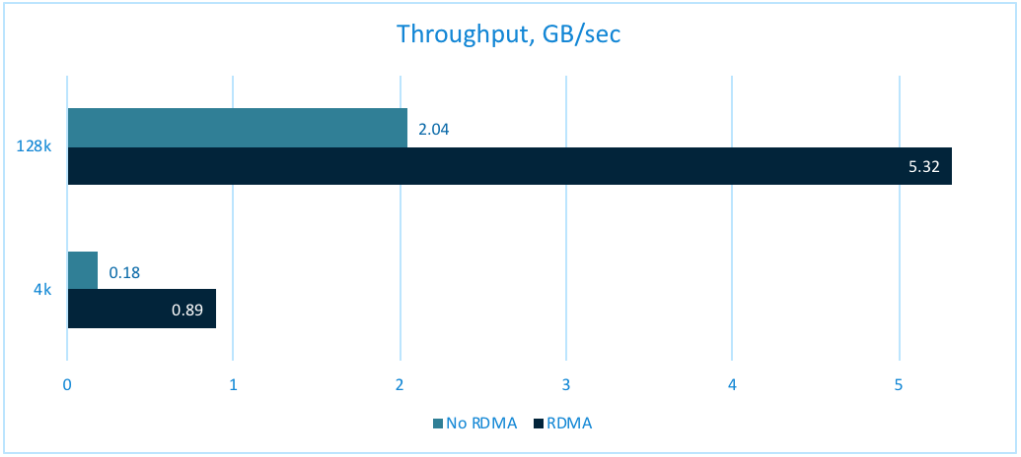
Metric	RDMA	TCP/IP	RDMA Advantage
IOPs	185,500	145,500	40,000 additional IOPS with the same workload
IOPs/%kernel CPU	16,300	12,800	3500 additional IOPs per-percent CPU consumed
90th Percentile Write Latency	250µs	390µs	140µs (~36%)
90th Percentile Read Latency	260µs	360µs	100µs (28%)

Reference: *Microsoft - To RDMA, or not to RDMA - that is the question*

Mellanox RDMA Study – 50Gb NIC NFS



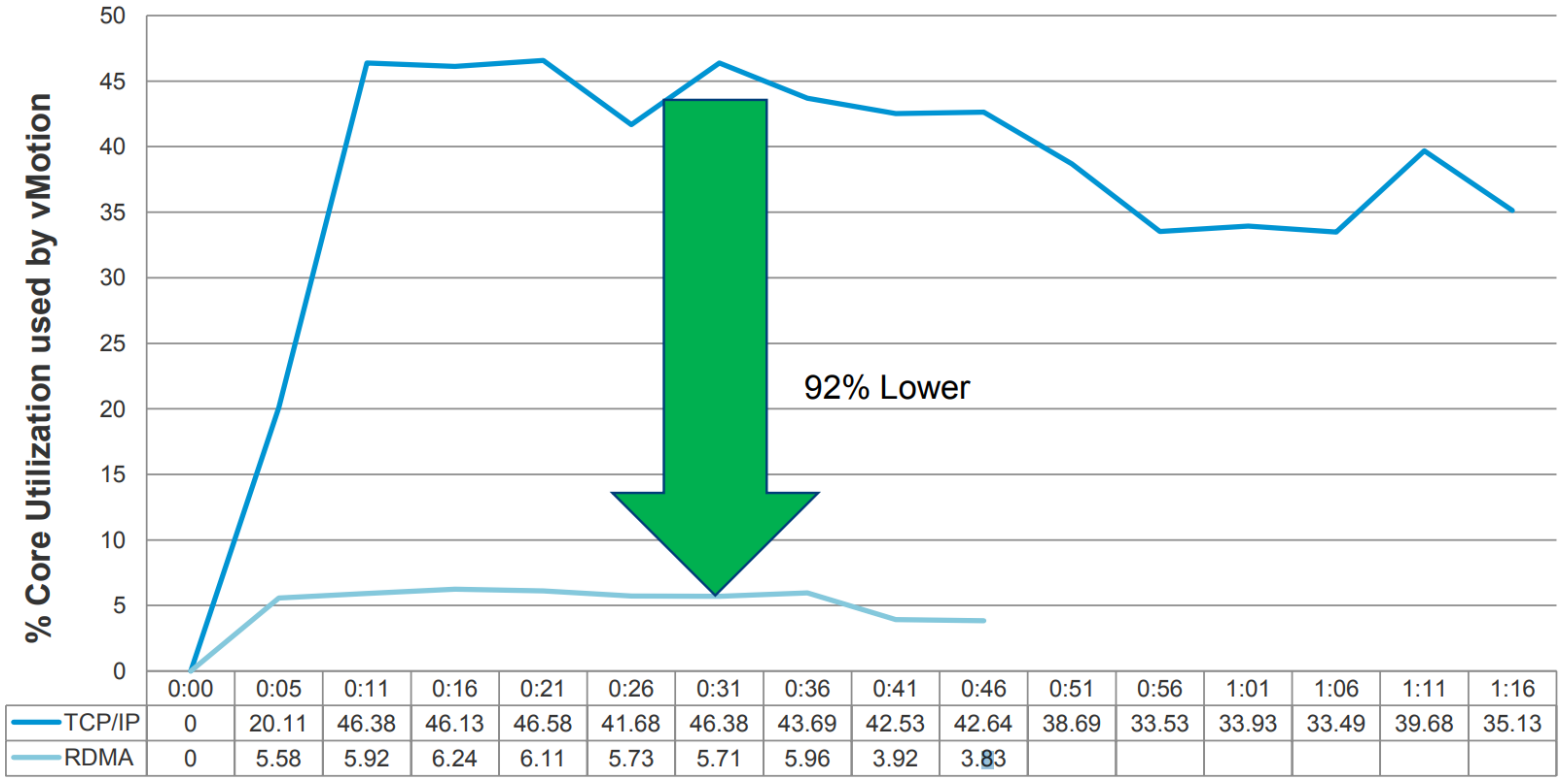




5.32GBps roughly equals 42.5Gbps

Reference: *Mellanox - Boosting Performance With RDMA - A Case Study*

VMware vMotion CPU Utilisation



	0:00	0:05	0:11	0:16	0:21	0:26	0:31	0:36	0:41	0:46	0:51	0:56	1:01	1:06	1:11	1:16
TCP/IP	0	20.11	46.38	46.13	46.58	41.68	46.38	43.69	42.53	42.64	38.69	33.53	33.93	33.49	39.68	35.13
RDMA	0	5.58	5.92	6.24	6.11	5.73	5.71	5.96	3.92	3.83						

- Source CPU Utilisation: **84%** Lower
- Destination CPU Utilisation: **92%** Lower

Reference: *Open Fabrics Alliance - RDMA on vSphere: Update and Future Directions*

NVMe & RDMA?

- To get the best of both worlds we can access NVMe storage over an RDMA fabric
  - NVMe over Fabrics (NVMeoF) – e.g either over Fiber Channel (FC) or RDMA Ethernet
  - Future fabrics expected to be supported (e.g GenZ)

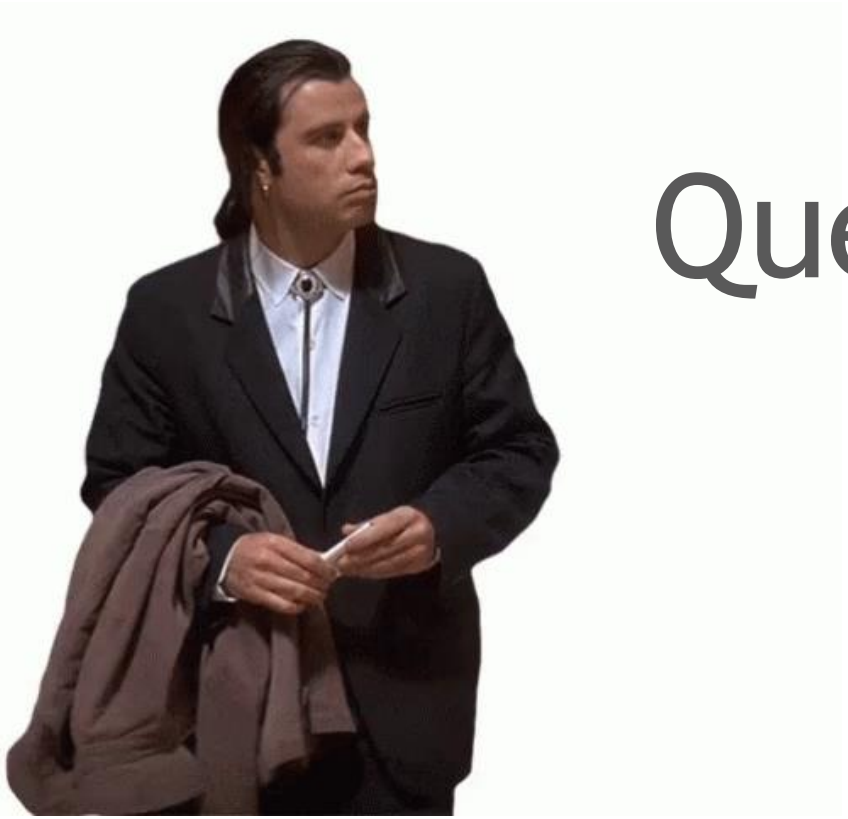
- **NVMe over Fabrics** goal is to provide connectivity to remote NVMe devices with no more than 10 microseconds additional latency over a local NVMe device

Balanced System Design

- Designing a balanced system is important to ensure each component can perform at its best without being bottlenecked too much by others
- Consider that high speed I/O (network, storage) requires large numbers of PCIe lanes
  - 100Gb Ethernet – 1 port requires PCIe Gen3 x16 lanes (12.5GB/s)
    - Dual port card would require 25GB/s which exceeds Gen3 capacity
- Evenly distribute devices across CPU sockets
  - We want to minimise socket <-> socket traffic so ensure fabric connectivity and local storage exists across both

Takeaway

- **SCM** devices bring persistent storage to the memory bus resulting in greater IOPs, throughput and **significantly** lower latency (potentially nanoseconds)
- **NVMe** enables far greater performance scaling with substantial latency reductions compared to traditional SAS/SATA
- **RDMA** enables efficient network transfer while also reducing latency to levels close to local access. CPU cycles are freed to do real work, not moving packets!



Questions

Feedback

Feedback is truly valued; either directly or via the team at TechUG

Slides available: **ByteSizedAlex.com**