SDC 19

September 23-26, 2019
Santa Clara, CA

So, You Want to Build a Storage Performance Test Lab?

Best Practices and Lessons Learned

Nick Principe

iXsystems

Twitter: @nickprincipe
Github: @powernap
Email: nap@ixsystems.com

Motivations for Building a Performance Lab

- Finding bugs
- Marketing material
- Sizing guidance
- Testing and qualifying new technologies
- Creating best practices
- Competitive guidance
- Convert large quantities of money into rapidly depreciating capital assets
  - :-)

You Get What You Measure

- A multipurpose axiom
  - Equally applicable to personnel management, performance testing, etc.
- If you measure a team based solely on closed ticket count…
  - You'll get a lot of closed tickets!
- If you measure a storage array based solely on maximum IOPS
  - You'll get really good cache hit 512b reads!

- A cautionary tale to encourage thought and attention to hardware, software, and benchmark design in your performance measurement environment

You Get What You Measure

- Carefully consider:
  - Benchmark / Load Generation Tool and Benchmark Workloads
    - See my MeetBSD 2018 Presentation: *Tales of a Daemontown Performance Peddler: Why "it depends" and what you can do about it.*
      - https://www.meetbsd.com/wp-content/uploads/MeetBSD_2018_Nick_Principe-PerformancePeddler.pdf
      - https://youtu.be/CpwngKaLZrg
  - Configuration of Solution Under Test
    - Physical Infrastructure
    - Virtual Infrastructure

Virtual Load Generators

Are Virtualized Load Generators Feasible?

- From varying degrees of testing I have done:
  - For publicly disclosed results, not really
  - For internal testing, yes, but only with VMware at this time
  - Tested multiple hypervisors and found deficiencies with all with respect to storage performance testing

Note: these are my experiences from limited testing – if things have improved or I am wrong, please let me know!

Hypervisor Testing

bhyve on FreeBSD

- Main issues:
  - bridge(4)/tap(4) virtual networking is too slow (~1 Gbit/s max)
  - BAR issues trying to pass SR-IOV network adapters into Windows
- Could work using SR-IOV with FreeBSD or Linux VMs
- Have not investigated using epair(4), but suspect bridge performance will still be limiting

KVM on Debian Linux

- Very promising network performance through virtual switches
- However, when adding a second VM on same switch, one VM starves another for network traffic
- Still possible KVM could be suitable with SR-IOV
  - Literature suggests KVM on Linux can successfully pass SR-IOV devices into Windows OS
  - Investigation diverted at this point

XenServer / XCP-NG

- XCP-NG is appealing because it provides a more turnkey ESXi-like experience than KVM or bhyve
- Virtual switch networking was too slow
  - Unable to saturate 10GbE link
- Very limited SR-IOV support
  - Only a few Intel 10GbE adapters listed as supported
  - Would be promising if SR-IOV support added for more cards

VMware ESXi

- Virtual network performance is very good
  - No need for SR-IOV
  - Good for older hosts that do not support SR-IOV
- Virtualized hosts with 5.5-6.0 had very similar network performance to physical hosts
  - When oversubscription is avoided
- With 6.5, saw more degradations with physical vs. virtual
  - Intend to revisit with newer testing at some point
- However… VMware EULA is aggressively limiting for benchmarking

VMware EULA

- VMware EULA seems to require review and approval for third-party distribution of benchmark results collected with VMware products
  - https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/downloads/eula/vmware-benchmarking-approval-process.pdf

The VMware End-User License Agreement (EULA) Clause:

"2.4 Benchmarking. You may use the Software to conduct internal performance testing and benchmarking studies. You may only publish or otherwise distribute the results of such studies to third parties as follows: (a) if with respect to VMware's Workstation or Fusion products, only if You provide a copy of Your study to benchmark@vmware.com prior to distribution; (b) if with respect to any other Software, only if VMware has reviewed and approved of the methodology, assumptions and other parameters of the study (please contact VMware at benchmark@vmware.com to request such review and approval) prior to such publication and distribution."

Reference: http://www.vmware.com/download/eula.html

This process document provides step-by-step overview of the benchmark review and approval process.

Note: VMware reserves the right to refuse publication of any benchmark.

Note: I am not a lawyer, this is not legal advice, nor a legal interpretation of the referenced EULA. This is my interpretation and is simply my personal opinion.

Configuration Best Practices

Virtualized Load Generator Best Practices

1. Disable hyperthreading on the host
2. Total powered on vCPUs <= Total real CPU cores
3. Total powered on allocated vMem < Total Physical Memory
   - Leave an adequate buffer for hypervisor overhead
4. Use one uplink port per vSwitch
   - Use multiple vSwitches if required
     - Manually round-robin VMs across vSwitches
5. If testing iSCSI, use software initiators on VM Oses
6. Inconsistent performance is worse than bad performance!

General Best Practices – Network

1. Total network bandwidth of all load generators must exceed that of the storage system under test
2. Avoid LAGs on the load generators
3. Avoid switch hops
   - Ensure sufficient ISL bandwidth if you must have a hop
4. Match MTU on OSes, switch ports, physical switches, and virtual switches

General Best Practices – Memory

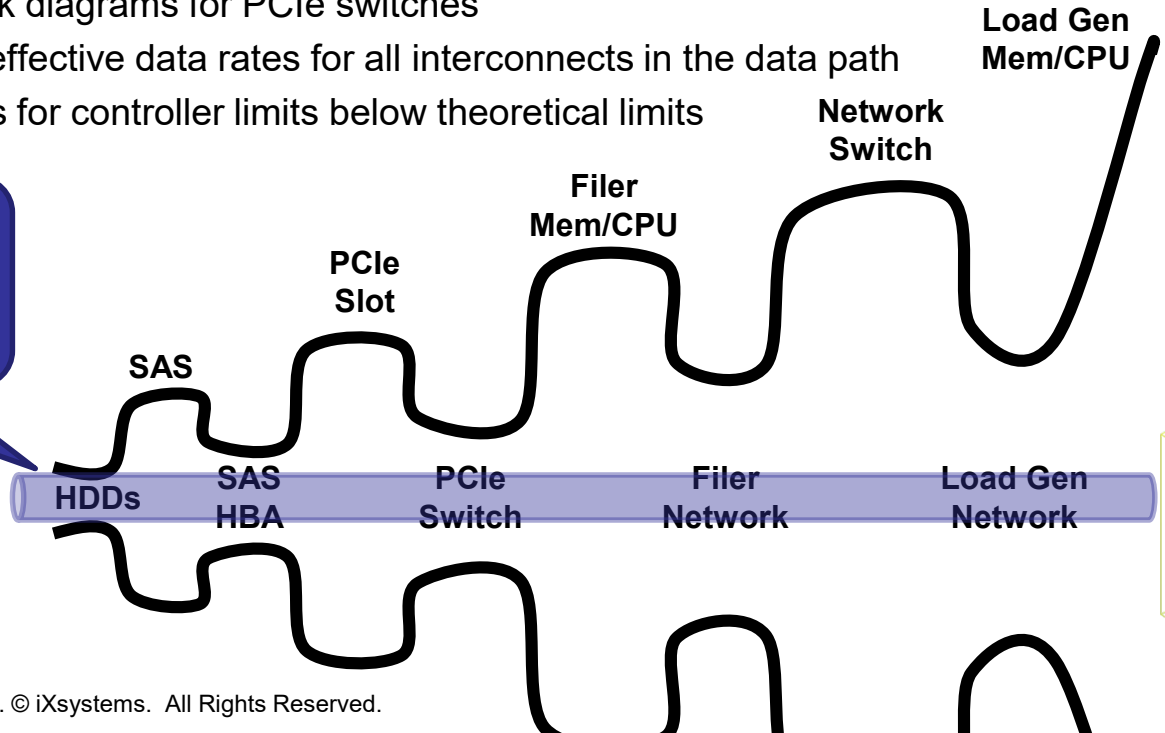
1. Minimum recommended memory
   - Windows 10: 8-16 GiB
   - Linux: 4 GiB
2. More memory may or may not be advantageous
   1. Minimal effect with workloads using direct I/O
      1. Most block testing
      2. Basic non-metadata file testing
   2. Substantial effect possible with complex file testing
      1. Metadata caching simplifies workload to storage system under test
   3. NFSv4 delegations/SMB leases
      1. Could dramatically increase load generator cache effectiveness

General Best Practices – Interconnects

1. Ensure architecture of Solution Under Test avoids unintended bottlenecks
   - Check server block diagrams for PCIe switches
   - Check maximum effective data rates for all interconnects in the data path
   - Check data sheets for controller limits below theoretical limits

This solution is designed to bottleneck on the performance of its HDDs

Storage solutions make strangely shaped bottles



Maximum Effective Data Rates

Maximum Effective Data Rates

- Rate of effective data transmission over an interconnect
  - Lower than raw signaling rate!
    - Physical encoding (8b/10b, etc.)
    - Protocol encapsulation overhead (CRC, headers, etc.)
  - Protocol overheads variable on message length
    - Usually low percentage impact if pushing high data rate
  - Only try to exclude physical encoding overheads
    - Big impact for 8b/10b, for example

Maximum Effective Data Rate Tables

PCI Express 3.0 & NVMe	Mebibytes per second (MiB/s) (1024^2 Bytes)	Megabytes per second (MB/s) (1000^2 Bytes)	Gigabits per second (Gb/s) (1000^3 Bits)	Notes
x1	939	985	7.9	
x2	1,878	1,969	15.7	NVMe M.2 (M+B Key)
x4	3,756	3,938	31.5	NVMe M.2 (M Key) and U.2
x8	7,512	7,877	63	
x16	15,024	15,754	126	

SAS-3(12 Gb/s)	Mebibytes per second (MiB/s) (1024^2 Bytes)	Megabytes per second (MB/s) (1000^2 Bytes)	Gigabits per second (Gb/s) (1000^3 Bits)
x1	1,144	1,200	9.6
x4	4,578	4,800	38.4
x8	9,152	9,600	76.8

For more, reference: https://www.ixsystems.com/blog/storage-performance-guide-interconnect-maximum-rates/

Using Maximum Effective Data Rates

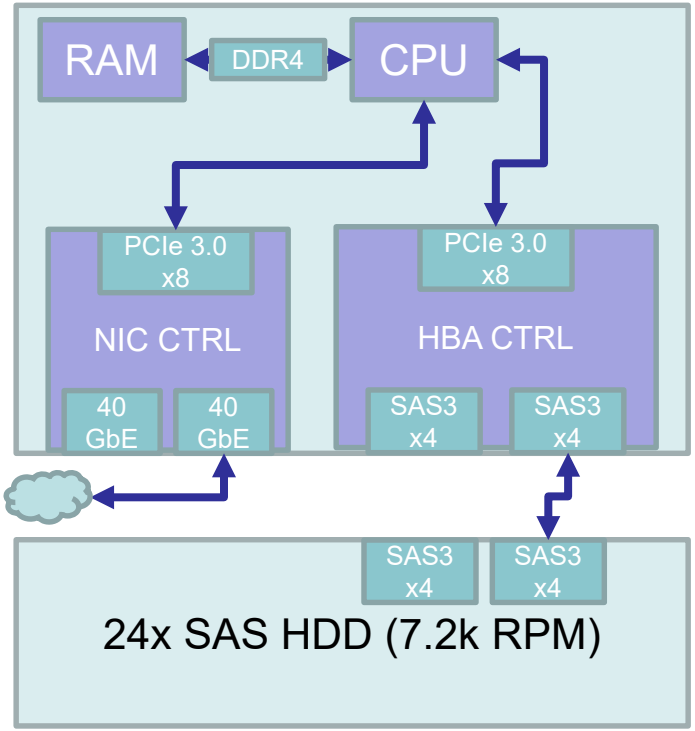
As an example, let's look at a generic storage server:

- PCIe 3.0 x8: 7,512 MiB/s
- SAS HBA chipset: 5,722 MiB/s
  - From chipset spec sheet
- 24x 7.2k SAS HDD: 4,608 MiB/s
  - Each drive (spec sheet): 192 MiB/s
- 1x SAS-3 x4 port: 4,576 MiB/s

What about the front-end?

- PCIe 3.0 x8: 7,512 MiB/s
- 1x 40GbE NIC port: 4,768 MiB/s

Theoretically, bottleneck is at connection to JBOD



Using Maximum Effective Data Rates

But wait! We're writing to a pool of mirrors!

- PCIe 3.0 x8: 7,512 MiB/s > 4,576 MiB/s
- SAS HBA chipset: 5,722 MiB/s > 4,576 MiB/s
  - From chipset spec sheet
- 24x 7.2k SAS HDD: 4,608 MiB/s > 4,576 MiB/s
  - Each drive (spec sheet): 192 MiB/s > 191 MiB/s
- 1x SAS-3 x4 port: 4,576 MiB/s / 2 = 2,288 MiB/s max effective

What about the front-end?

- 1x 40GbE NIC port: 4,768 MiB/s > 2,288 MiB/s
- PCIe 3.0 x8: 7,512 MiB/s > 2,288 MiB/s

The Search for Perfect Load Generation Hardware

Does Load Generating Hardware Matter?

- Yes; at least to some extent
  - As always, it depends!
- Matters less for synthetic testing
  - Except for single client/thread testing
- Matters more for real application testing

Does Load Generating Hardware Matter?

- For synthetic at-scale testing, they only need to be “good enough”
  - Synthetic benchmarks should be efficient at generating I/O with low resource utilization
  - Some workloads involve the client OS more than others
    - A significant metadata component or lack of O_DIRECT will put more pressure on VFS, protocol, and caching layers in client OS
  - Keep in mind all previous configuration and sizing guidance!

Does Load Generating Hardware Matter?

- For real application testing, load generator hardware matters a lot
  - Actually require compute time and cause memory pressure
  - They're still doing "real work"
- We maintain a separate lab with high-spec hardware for this testing
  - This hardware is also used for single client / single threaded maximum synthetic performance testing
  - Lower spec at-scale synthetic load generators may get artificially low numbers when not used at-scale

The Search Begins

- We wanted new load generators for at-scale synthetic testing
  - Balance between cost, flexibility, convenience, and compatibility
  - Want performance same or better than current E3-1270 v5-based systems
  - Want 25 GbE for better switch bandwidth utilization
  - Require enterprise out-of-band management/IPMI

The Candidates

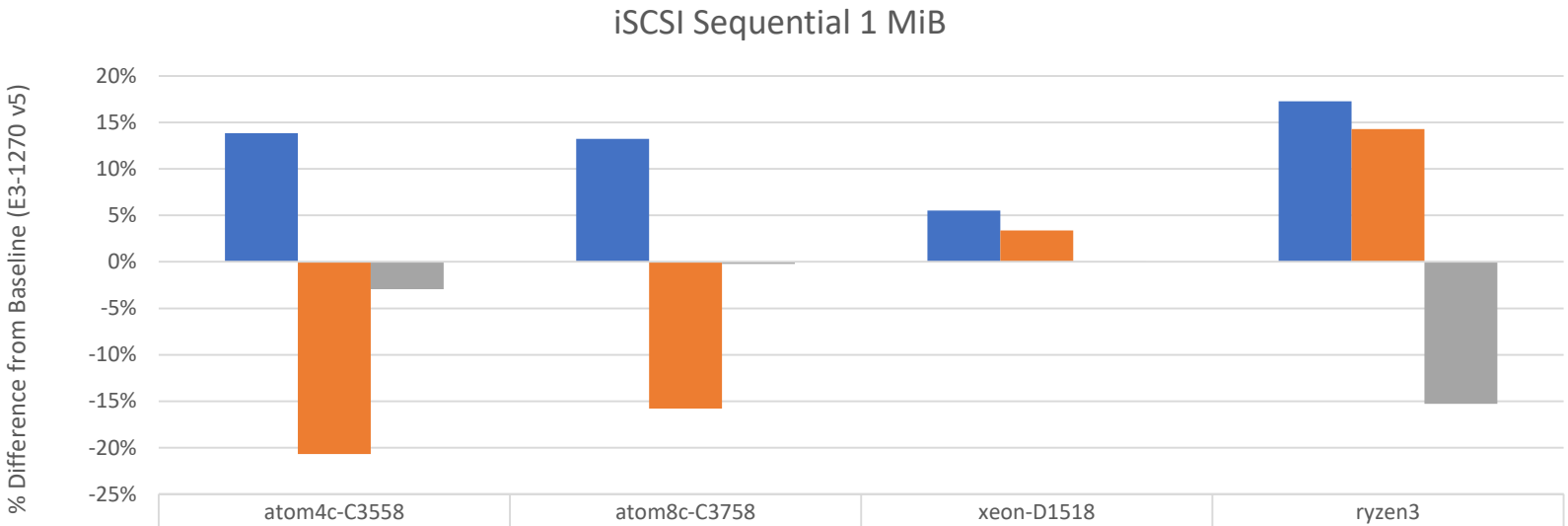
Processor	CPU Cores (Threads)	CPU Clock (Turbo)	Memory	Network
E3-1270 v5**	4 (8)	3.6 (4.0) GHz	32 GiB	Intel 10GbE
Xeon D-1518	4 (8)	2.2 (2.2) GHz	16 GiB	Intel 10GbE
Atom C3758	8 (8)	2.2 (2.2) GHz	16 GiB	Chelsio 10GbE
Atom C3558	4 (4)	2.2 (2.2) GHz	16 GiB	Chelsio 10GbE
Ryzen 5 1600X*	4 (4)	3.5 (4.0) GHz	16 GiB	Chelsio 10GbE
Ryzen 3 2200G	4 (4)	3.5 (3.7) GHz	16 GiB	Chelsio 10/25GbE

*** Ryzen 5 configured to be similar to 2200G**
**** E3-1270 v5 is the baseline CPU in our current load generators**

iSCSI Test Setup

- Tested against an older all-flash FreeNAS configured to maximize performance
  - Place bottleneck at load generator as much as possible
- Small active dataset size
  - Reads are all cached on FreeNAS
- Load generators running CentOS Linux 7

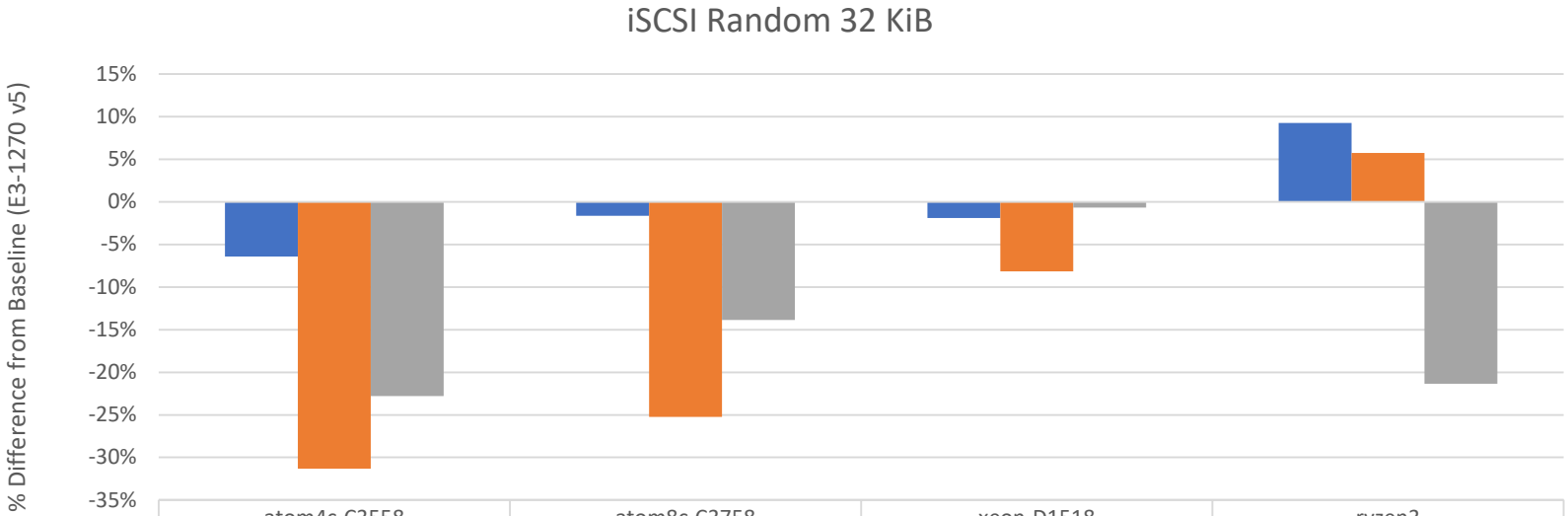
iSCSI Sequential 1 MiB



	atom4c-C3558	atom8c-C3758	xeon-D1518	ryzen3
100% Read	14%	13%	6%	17%
50% Read	-21%	-16%	3%	14%
100% Write	-3%	0%	0%	-15%

- Assume ~5% run-to-run variation; want to be > -5% off baseline load gen results
- Believe large dip on writes with Ryzen 3 to be testing artifact
- Reads are basically at 10Gb/s (line rate) for Ryzen 3 test

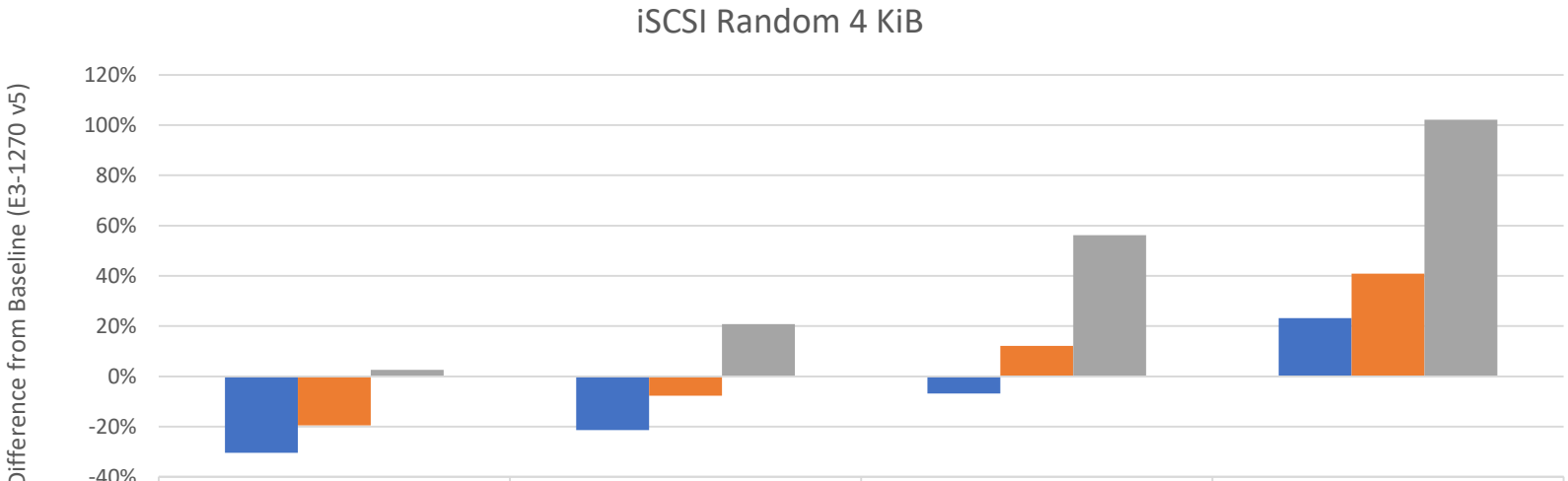
iSCSI Random 32 KiB



	atom4c-C3558	atom8c-C3758	xeon-D1518	ryzen3
100% Read	-6%	-2%	-2%	9%
50% Read	-31%	-25%	-8%	6%
100% Write	-23%	-14%	-1%	-21%

- Assume ~5% run-to-run variation; want to be > -5% off baseline load gen results
- Believe large dip on writes with Ryzen 3 to be testing artifact

iSCSI Random 4KiB



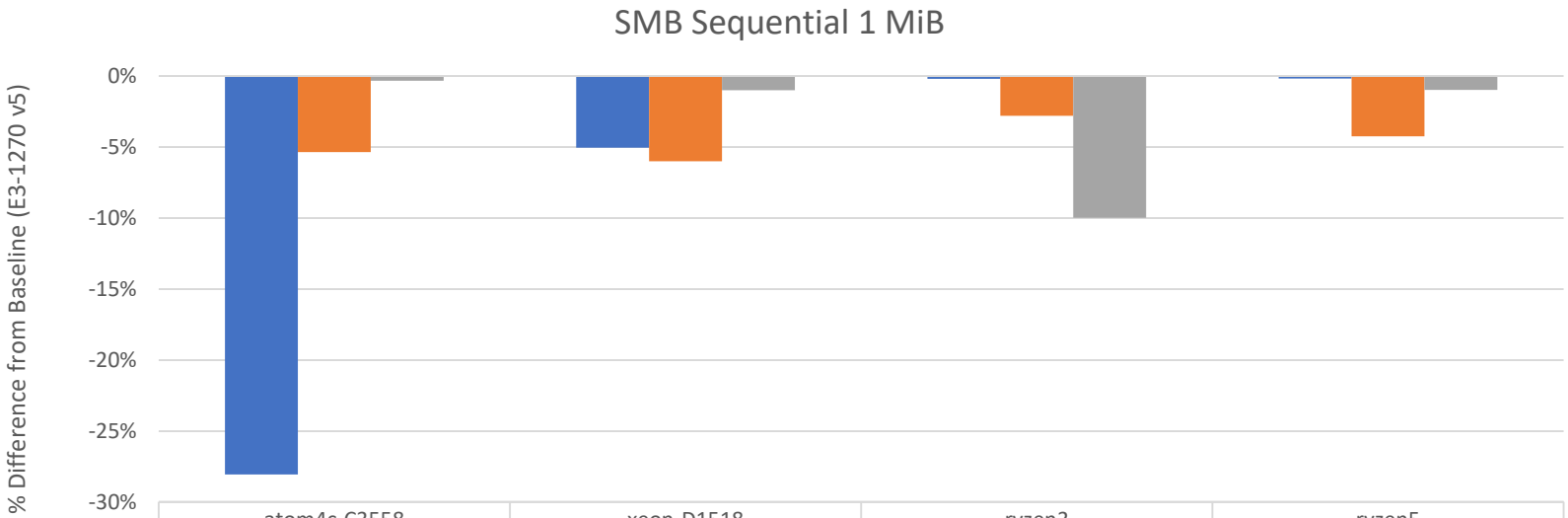
	atom4c-C3558	atom8c-C3758	xeon-D1518	ryzen3
100% Read	-30%	-21%	-7%	23%
50% Read	-20%	-8%	12%	41%
100% Write	3%	21%	56%	102%

- Assume ~5% run-to-run variation; want to be > -5% off baseline load gen results

SMB Test Setup

- Nearly identical setup for SMB testing
  - Running Windows 10 on load generators

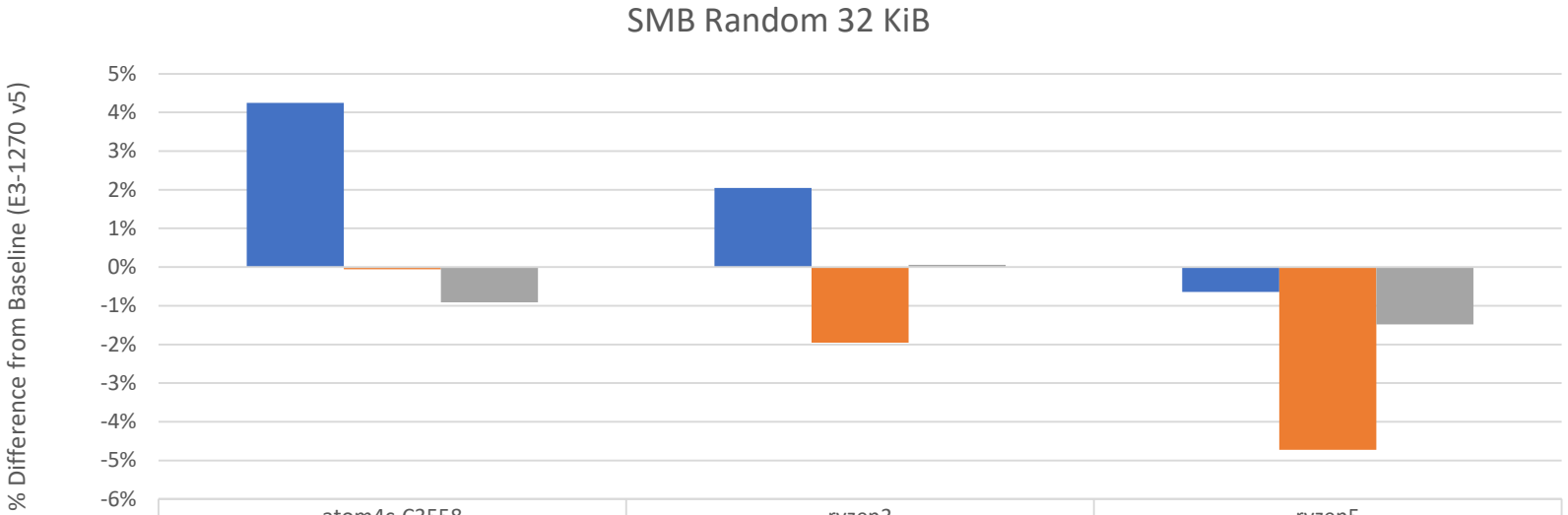
SMB Sequential 1 MiB



	atom4c-C3558	xeon-D1518	ryzen3	ryzen5
100% Read	-28%	-5%	0%	0%
50% Read	-5%	-6%	-3%	-4%
100% Write	0%	-1%	-10%	-1%

- Assume ~5% run-to-run variation; want to be > -5% off baseline load gen results
- Large dip on 100% writes on Ryzen 3 is a test artifact – faster on different array
- Atom became core-bound during 100% read; believe Xeon-D core-bound as well

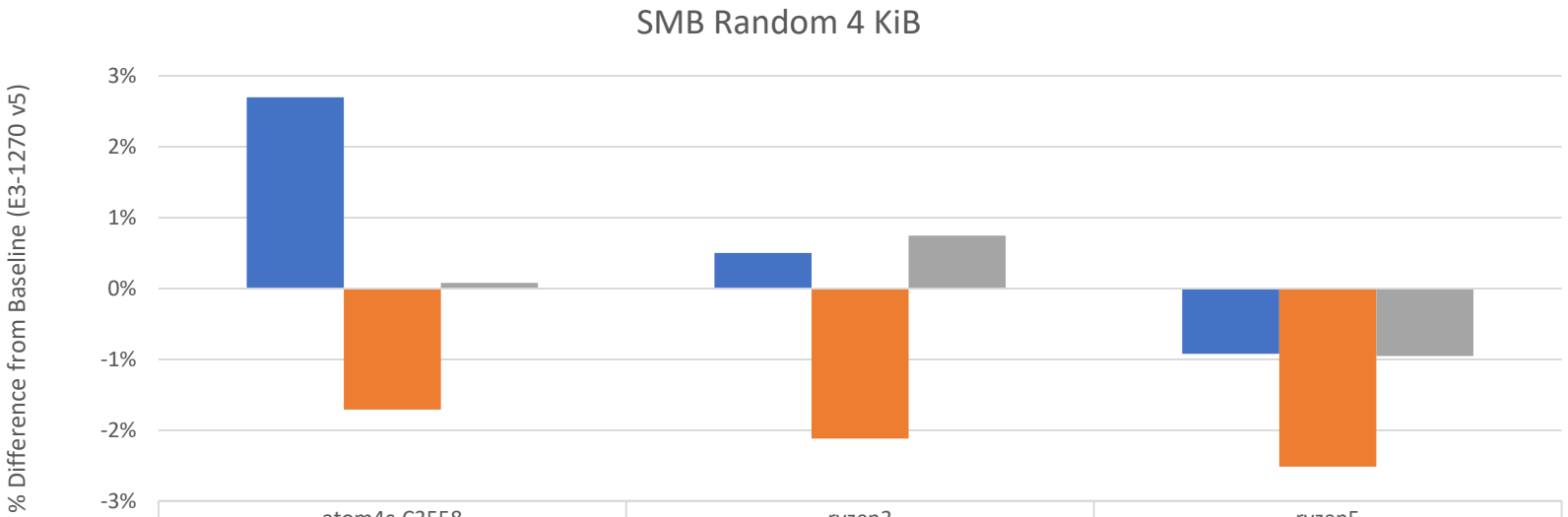
SMB Random 32 KiB



	atom4c-C3558	ryzen3	ryzen5
100% Read	4%	2%	-1%
50% Read	0%	-2%	-5%
100% Write	-1%	0%	-1%

- Assume ~5% run-to-run variation; want to be > -5% off baseline load gen results
- All platforms within +/- 5%

SMB Random 4KiB



	atom4c-C3558	ryzen3	ryzen5
100% Read	3%	1%	-1%
50% Read	-2%	-2%	-3%
100% Write	0%	1%	-1%

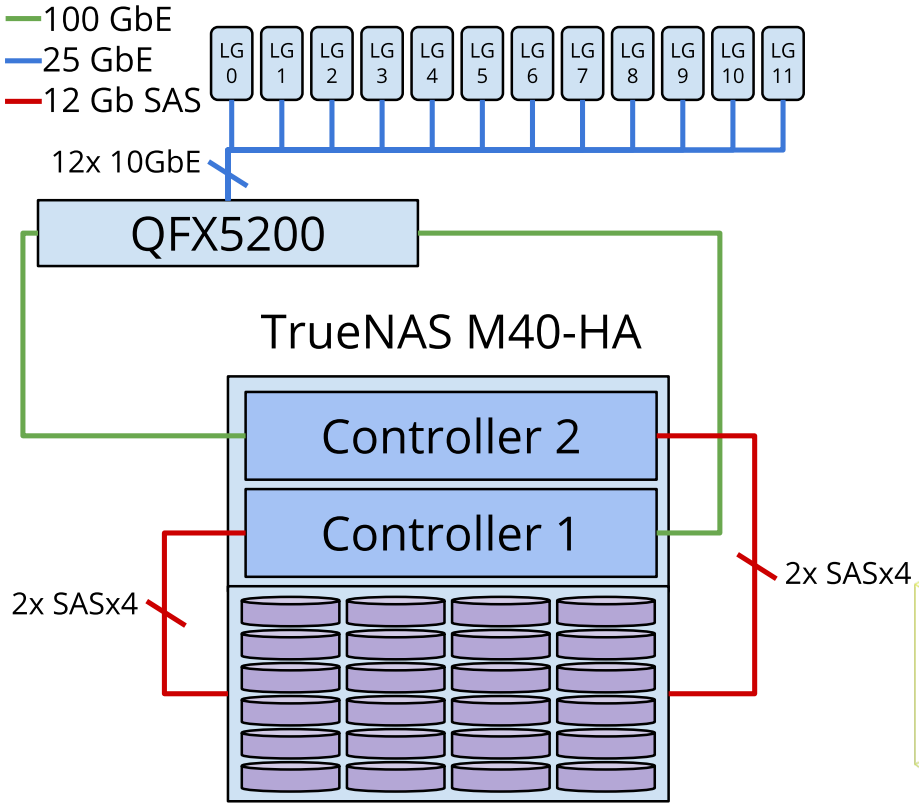
- Assume ~5% run-to-run variation; want to be > -5% off baseline load gen results
- All results within +/- 3%

The Decision

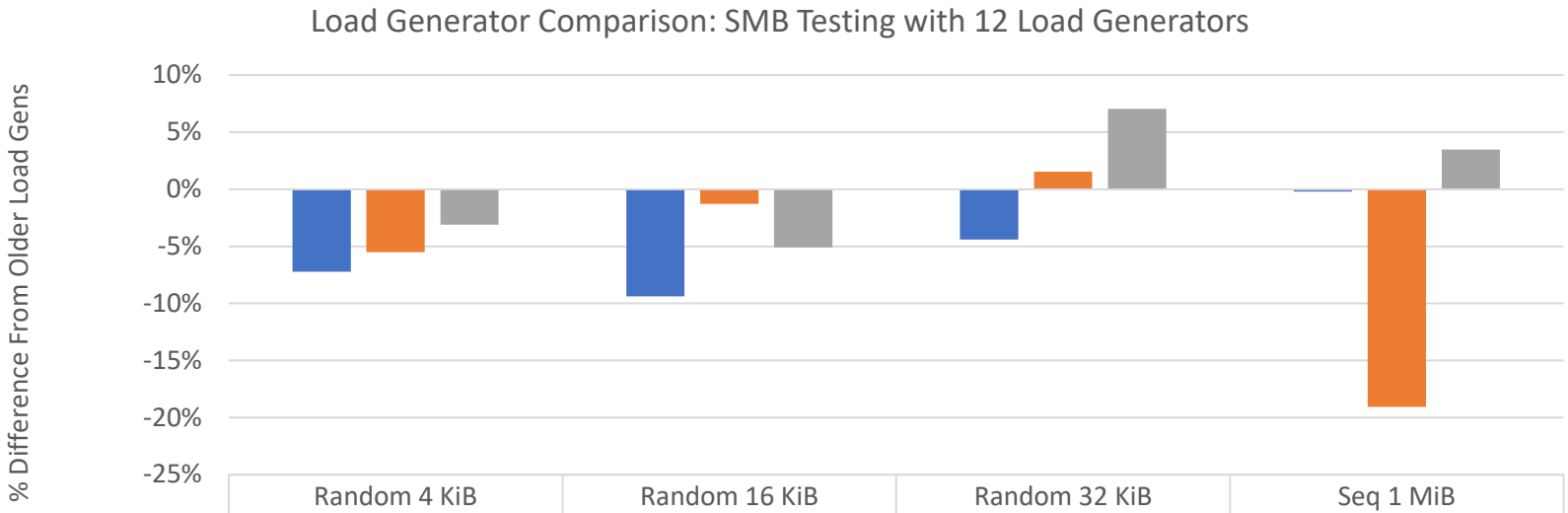
- Decided on Ryzen 3 2200G based on data
  - Wound up switching to Ryzen 3 3200G
- SMB sequential read tests important
  - Ruled out Atom due to becoming core-bound at under 900 MiB/s
  - Concerns about Xeon-D in same scenario due to similar clock rate
  - Ryzen-based load generator with 25GbE can achieve ~1,600 MiB/s in same test

Full-Scale Performance Testing

- Manufactured a full set of 12 load generators
- Setup in performance test lab
- Compared to our older load generators
  - Unfortunately not apples-to-apples
  - New load generators required new OS image
    - Newer Windows 10
    - Different, newer, Linux distribution
- Used our standard Vdbench-based performance characterization test suite



How Did We Do?



	Random 4 KiB	Random 16 KiB	Random 32 KiB	Seq 1 MiB
100% Read	-7%	-9%	-4%	0%
70%/50% Read	-6%	-1%	2%	-19%
100% Write	-3%	-5%	7%	3%

- Validating with SMB testing comparing new load gens to old load gens
  - Identical setup, only difference is load generator hardware and OS version
- New Windows 10 install, newer build

Conclusions

- Carefully choose and configure equipment to avoid unintended bottlenecks or inconsistent performance
- Pay attention to maximum effective data rates for all interconnects in the data path
- Virtualized load generators can work, but much more care is needed
- Load generator hardware matters to varying degrees
  - For synthetic testing, no need to break the bank
- Always A/B test configuration changes to avoid surprise changes to performance results!
  - Don't change more than one variable at once!

Please take a moment
to rate this session.
Your feedback matters to us.

Thank You! Questions?

Nick Principe
iXsystems

Twitter: @nickprincipe
Github: @powernap
Email: nap@ixsystems.com