

SPDK Vhost Performance Report

Release 23.05

Testing Date: July 2023

Performed by:

Michal Berger (michal.berger@intel.com)

Karol Latecki (karol.latecki@intel.com)

Jaroslav Chachulski (jaroslawx.chachulski@intel.com)

Acknowledgments:

James Harris (james.r.harris@intel.com)

Tomasz Zawadzki (tomasz.zawadzki@intel.com)

Contents

Contents	2
Audience and Purpose.....	3
Test setup	4
Hardware configuration	4
BIOS Settings	5
Virtual Machine Settings.....	5
Introduction to the SPDK Vhost target	7
SPDK Vhost target architecture	7
Test Case 1: SPDK Vhost Core Scaling	9
4KiB Random Read Results.....	11
4KiB Random Write Results	12
4KiB Random Read-Write Results	13
Logical Volumes performance impact	14
Packed Ring performance impact.....	15
Conclusions	16
Test Case 2: Rate Limiting IOPS per VM.....	17
Test Case 2 Results	19
Conclusions	21
Test Case 3: Performance per NVMe drive	22
Test Case 3 results.....	23
Conclusions	26
Summary	27
Appendix A - Vfiio-User Performance	28
List of Tables	30
List of Figures	31

Audience and Purpose


This report is intended for people who are interested in looking at SPDK Vhost scsi and blk stack performance and comparison to its Linux kernel equivalents. It provides performance and efficiency comparisons between SPDK Vhost-scsi and Linux Kernel Vhost-scsi software stacks under various test cases.

The purpose of this report is not to imply a single correct approach, but rather to provide a baseline of well-tested configurations and procedures that produce repeatable and reproducible results. This report can also be viewed as information regarding best known method when performance testing SPDK Vhost-scsi and Vhost-blk stacks.

Test setup

Hardware configuration

Table 1: Hardware setup configuration

Item	Description																		
Server Platform	Ultra SuperServer SYS-220U-TNR 																		
Motherboard	Server board X12DPU-6																		
CPU	2 CPU sockets, Intel(R) Xeon(R) Gold 6348 CPU @ 2.60GHz Number of cores 28 per socket, number of threads 56 per socket Both sockets populated. Microcode: 0xd000389																		
Memory	16 x 32GB SK Hynix DDR4 HMA84GR7DJR4N-XN; Total 512 GBs. Memory channel population: <table border="1"> <thead> <tr> <th>P1</th><th>P2</th></tr> </thead> <tbody> <tr> <td>CPU1_DIMM_A1</td><td>CPU2_DIMM_A1</td></tr> <tr> <td>CPU1_DIMM_B1</td><td>CPU2_DIMM_B1</td></tr> <tr> <td>CPU1_DIMM_C1</td><td>CPU2_DIMM_C1</td></tr> <tr> <td>CPU1_DIMM_D1</td><td>CPU2_DIMM_D1</td></tr> <tr> <td>CPU1_DIMM_E1</td><td>CPU2_DIMM_E1</td></tr> <tr> <td>CPU1_DIMM_F1</td><td>CPU2_DIMM_F1</td></tr> <tr> <td>CPU1_DIMM_G1</td><td>CPU2_DIMM_G1</td></tr> <tr> <td>CPU1_DIMM_H1</td><td>CPU2_DIMM_H1</td></tr> </tbody> </table>	P1	P2	CPU1_DIMM_A1	CPU2_DIMM_A1	CPU1_DIMM_B1	CPU2_DIMM_B1	CPU1_DIMM_C1	CPU2_DIMM_C1	CPU1_DIMM_D1	CPU2_DIMM_D1	CPU1_DIMM_E1	CPU2_DIMM_E1	CPU1_DIMM_F1	CPU2_DIMM_F1	CPU1_DIMM_G1	CPU2_DIMM_G1	CPU1_DIMM_H1	CPU2_DIMM_H1
P1	P2																		
CPU1_DIMM_A1	CPU2_DIMM_A1																		
CPU1_DIMM_B1	CPU2_DIMM_B1																		
CPU1_DIMM_C1	CPU2_DIMM_C1																		
CPU1_DIMM_D1	CPU2_DIMM_D1																		
CPU1_DIMM_E1	CPU2_DIMM_E1																		
CPU1_DIMM_F1	CPU2_DIMM_F1																		
CPU1_DIMM_G1	CPU2_DIMM_G1																		
CPU1_DIMM_H1	CPU2_DIMM_H1																		
Operating System	Fedora 37																		
BIOS	1.4b																		
Linux kernel version	6.1.6-200.fc37.x86_64 Spectre-meltdown mitigations enabled																		
SPDK version	SPDK 23.05																		
Qemu version	7.0.0 (qemu-7.0.0-12.fc37)																		
Storage	OS: 1x 250GB Crucial CT250MX500SSD1 Storage: 22x Kioxia® KCM61VUL3T20 3.2TBs (FW: 0105) (10 on CPU NUMA Node 0, 12 on CPU NUMA Node 1)																		

BIOS Settings

Table 2: Test platform BIOS settings

Item	Description
BIOS	VT-d = Enabled CPU Power and Performance Policy = <Performance> CPU C-state = No Limit CPU P-state = Enabled Enhanced Intel® Speedstep® Tech = Enabled Turbo Boost = Enabled Hyper Threading = Enabled

Table 3: Test System NVMe storage setup

Item	Description	
PCIe Riser cards	“Ultra” Riser Card: AOC-2UR68G4-i2XT <ul style="list-style-type: none">• PCIe Slot 1 – x16, CPU2• PCIe Slot 2 – x8, CPU2• PCIe Slot 3 – x8, CPU2 Right-facing riser card: RSC-WR-6 <ul style="list-style-type: none">• PCIe Slot 4 – x16, CPU1 Left-facing riser card: RSC-W2-66G4 <ul style="list-style-type: none">• PCIe Slot 5 – x16, CPU2• PCIe Slot 7 – x16, CPU1 More information can be found in SYS-220U-TNR manual document .	
	PCIe Retimer cards	
NVMe Drives distribution across the system	3 x AOC-SLG4-4E4T Installed in: <ul style="list-style-type: none">○ PCIe Retimer 1: RSC-WR-6, PCIe Slot 4 (using CPU1 PCIe Lanes)○ PCIe Retimer 2: AOC-2UR68G4-i2XT, PCIe Slot 1 (using CPU2 PCIe Lanes)○ PCIe Retimer 3: RSC-W2-66G4, PCIe Slot 5 (using CPU2 PCIe Lanes)	
	Nvme0 – 5	Motherboard ports (CPU1 PCIe Lanes)
	Nvme6 – 9	Motherboard ports (CPU2 PCIe Lanes)
	Nvme9 – 13	PCIe Retimer 1 (CPU1 PCIe Lanes)
	Nvme14 - 17	PCIe Retimer 2 (CPU2 PCIe Lanes)
	Nvme18 - 21	PCIe Retimer 3 (CPU2 PCIe Lanes)

Virtual Machine Settings

Table 4: Guest VM configuration

Item	Description
CPU	2 vCPU, pass through from physical host server. Explicit core usage enforced using “taskset –a –c” command. QEMU arguments for starting the VM: -cpu host -smp 1

Memory	<p>2 GB RAM. Memory is pre-allocated for each VM using Hugepages on host system and used from appropriate NUMA node, to match the CPU which was passed to the VM.</p> <p>QEMU arguments:</p> <pre>-m 2048 -object memory-backend-file,id=mem,size=2048M,mem-path=/dev/hugepages,share=on,prealloc=yes,host-nodes=0,policy=bind</pre>
Operating System	Fedora 35
Linux kernel version	5.15.7-200.fc35.x86_64
Additional boot options in /etc/default/grub	Multi queue enabled: scsi_mod.use_blk_mq=1

Introduction to the SPDK Vhost target

SPDK Vhost is a userspace target designed to extend the performance efficiencies of SPDK into QEMU/KVM virtualization environments. The SPDK Vhost-scsi target presents a broad range of SPDK-managed block devices into virtual machines. SPDK community has leveraged existing SPDK SCSI layer, DPDK Vhost library, QEMU Vhost-scsi and Vhost-blk functionality to create the high performance SPDK userspace Vhost target.

SPDK Vhost target architecture

QEMU sets up the Vhost target via UNIX domain socket. QEMU pre-allocates huge pages for the guest VM to enable DMA by the Vhost target. The guest VM submits I/O directly to the Vhost target via virtqueues in shared memory as shown in Figure 1. The Vhost target transfers data to/from the guest VM via shared memory. The Vhost target then completes I/O to the guest VM via virtqueues in shared memory. There is a completion interrupt sent using eventfd which requires a system call and a guest VM exit. It should be noted that there is no QEMU intervention during the I/O submission process.

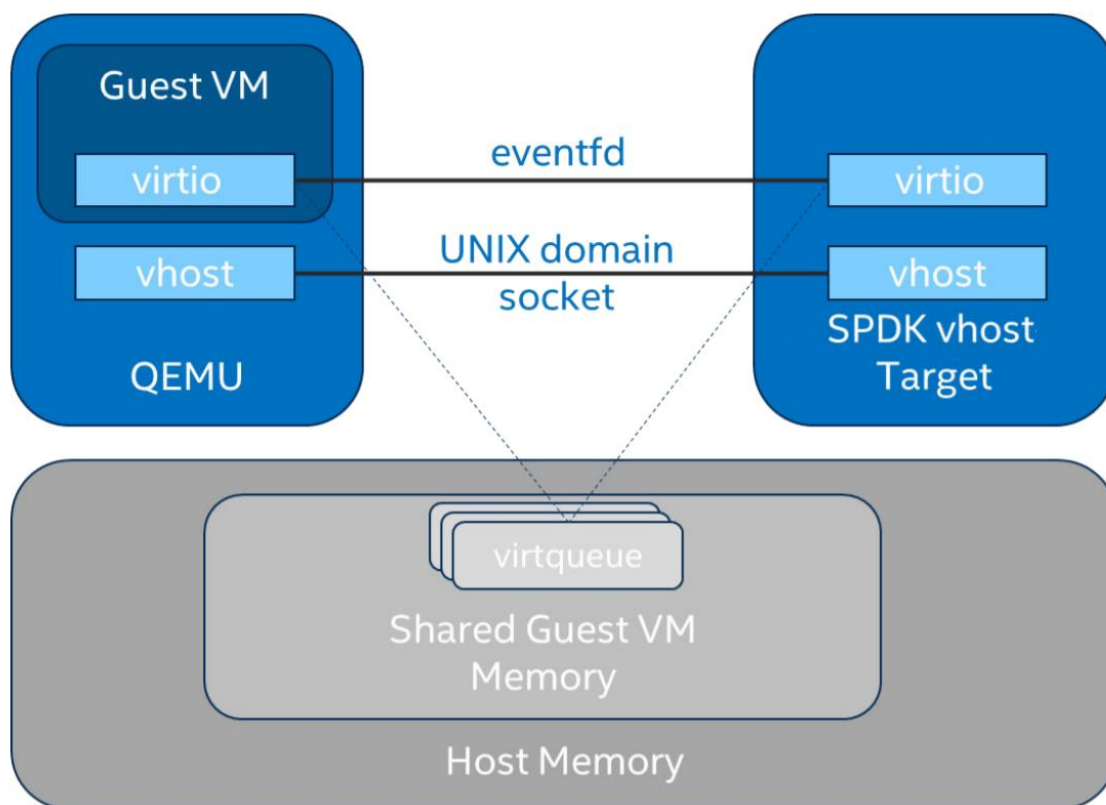


Figure 1: SPDK Vhost-scsi architecture

This report shows the performance comparisons between the traditional interrupt-driven Linux Kernel Vhost-scsi and the accelerated polled-mode SPDK Vhost-scsi under 3 different test cases using local

NVMe storage. Additionally, the SPDK Vhost-blk stack is included in the report for further comparison with the SCSI stack.

Test Case 1: SPDK Vhost Core Scaling

This test case was performed to understand aggregate VM performance with SPDK Vhost I/O core scaling. We ran up to 36 virtual machines, each running following FIO workloads:

- 4KiB 100% Random Read
- 4KiB 100% Random Write
- 4KiB Random 70% Read / 30 % Write

We increased the number of CPU cores used by SPDK Vhost target to process I/O from 1 up to 22 and measured the throughput (in IOPS) and latency. The number of VMs between test runs was not constant and was increased by 2 for each Vhost CPU added, up to a maximum of 44 VMs. VM number was not increased beyond 44 because of the platform capabilities in terms of available CPU cores.

FIO was run in client-server mode. FIO client was run on the host machine and distributed jobs to FIO servers run on each VM. This allowed us to start the FIO jobs across all VMs at the same time.

Results in the table and charts represent aggregate performance (IOPS and average latency) seen across all the VMs. The results are average of 3 runs.

Table 5: SPDK Vhost Core Scaling test configuration

Item	Description
Test case	Test SPDK Vhost target I/O core scaling performance
Test configuration	<p>FIO Version: fio-3.28</p> <p>VM Configuration:</p> <ul style="list-style-type: none">• Common settings are described in the Virtual Machine Settings chapter.• Number of VMs: variable (2 VMs per 1 Vhost CPU core, up to 44 VMs max).• Each VM has a single Vhost device as a target for the FIO workload. This is achieved by sharing SPDK NVMe bdevs by using either a Split NVMe vbdev or Logical Volume bdev configuration. <p>SPDK Vhost target configuration:</p> <ul style="list-style-type: none">• Test were run with both the Vhost-scsi and Vhost-blk stacks.• The Vhost-scsi stack was run with Split NVMe bdevs and Logical Volume bdevs.• Vhost-blk stack was run with Logical Volume bdevs.• Tests were performed with 1, 2, 6, 10, 14, 18 and 22 Vhost cores for each stack-bdev combination. <p>Kernel Vhost target configuration:</p> <ul style="list-style-type: none">• N/A
FIO configuration	[global] ioengine=libaio direct=1 thread=1

	<p> norandommap=1 time_based=1 gtod_reduce=0 ramp_time=60s runtime=240s numjobs=2 bs=4k rw=randrw rwmixread=100 (100% reads), 70 (70% reads, 30% writes), 0 (100% writes) iodepth= {1, 64, 128, 192, 384} </p>
--	---

4KiB Random Read Results

Table 6: SPDK Vhost core scaling results, 4KiB 100% Random Reads IOPS, QD=64

# of CPU cores	# of VMs	Stack / Backend	IOPS (millions)
1	2	SCSI / Split NVMe Bdev	1.69
		SCSI / Lvol Bdev	1.72
		BLK / Lvol Bdev	1.84
2	4	SCSI / Split NVMe Bdev	3.35
		SCSI / Lvol Bdev	3.43
		BLK / Lvol Bdev	3.66
6	12	SCSI / Split NVMe Bdev	9.12
		SCSI / Lvol Bdev	9.04
		BLK / Lvol Bdev	9.70
10	20	SCSI / Split NVMe Bdev	14.68
		SCSI / Lvol Bdev	13.62
		BLK / Lvol Bdev	14.52
14	28	SCSI / Split NVMe Bdev	15.43
		SCSI / Lvol Bdev	14.03
		BLK / Lvol Bdev	14.64
18	36	SCSI / Split NVMe Bdev	16.85
		SCSI / Lvol Bdev	16.39
		BLK / Lvol Bdev	17.63
22	44	SCSI / Split NVMe Bdev	17.79
		SCSI / Lvol Bdev	18.31
		BLK / Lvol Bdev	20.15

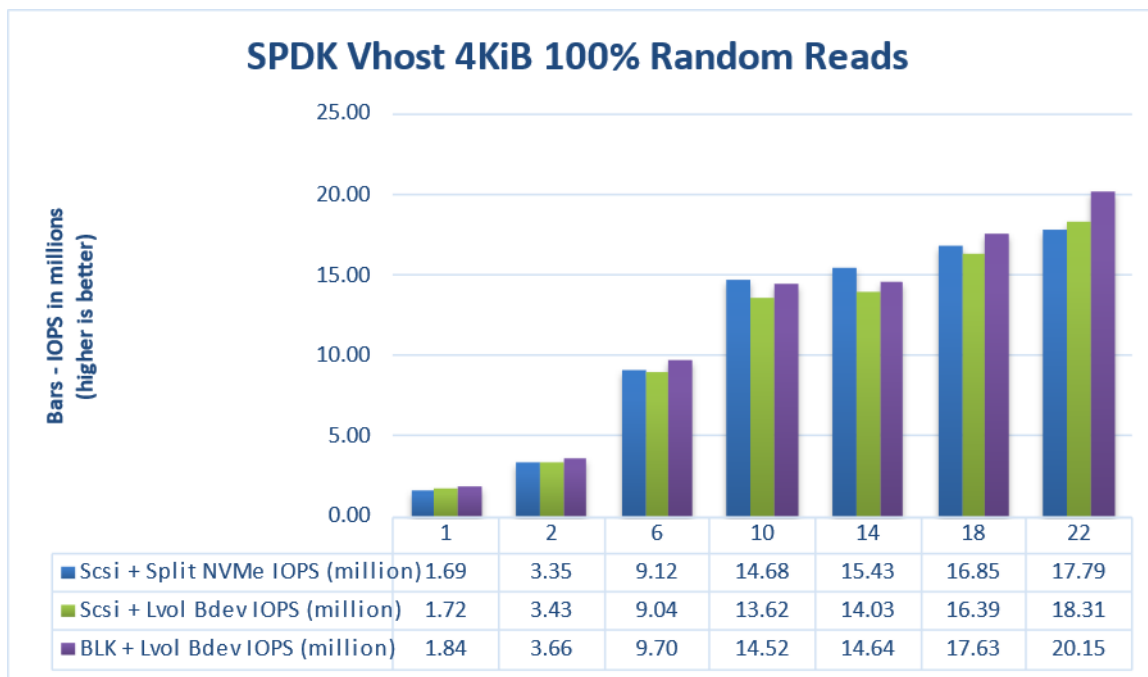


Figure 2: Comparison of performance between various SPDK Vhost stack-bdev combinations for 4KiB Random Read QD=64 workload

4KiB Random Write Results

Table 7: SPDK Vhost core scaling results, 4KiB 100% Random Write IOPS, QD=64

# of CPU cores	# of VMs	Stack / Backend	IOPS (millions)
1	2	SCSI / Split NVMe Bdev	1.09
		SCSI / Lvol Bdev	1.07
		BLK / Lvol Bdev	1.12
2	4	SCSI / Split NVMe Bdev	2.47
		SCSI / Lvol Bdev	2.47
		BLK / Lvol Bdev	2.59
6	12	SCSI / Split NVMe Bdev	6.92
		SCSI / Lvol Bdev	7.10
		BLK / Lvol Bdev	7.44
10	20	SCSI / Split NVMe Bdev	12.02
		SCSI / Lvol Bdev	12.55
		BLK / Lvol Bdev	13.21
14	28	SCSI / Split NVMe Bdev	12.86
		SCSI / Lvol Bdev	12.13
		BLK / Lvol Bdev	12.72
18	36	SCSI / Split NVMe Bdev	12.17
		SCSI / Lvol Bdev	11.90
		BLK / Lvol Bdev	13.01
22	44	SCSI / Split NVMe Bdev	12.22
		SCSI / Lvol Bdev	12.37
		BLK / Lvol Bdev	13.19

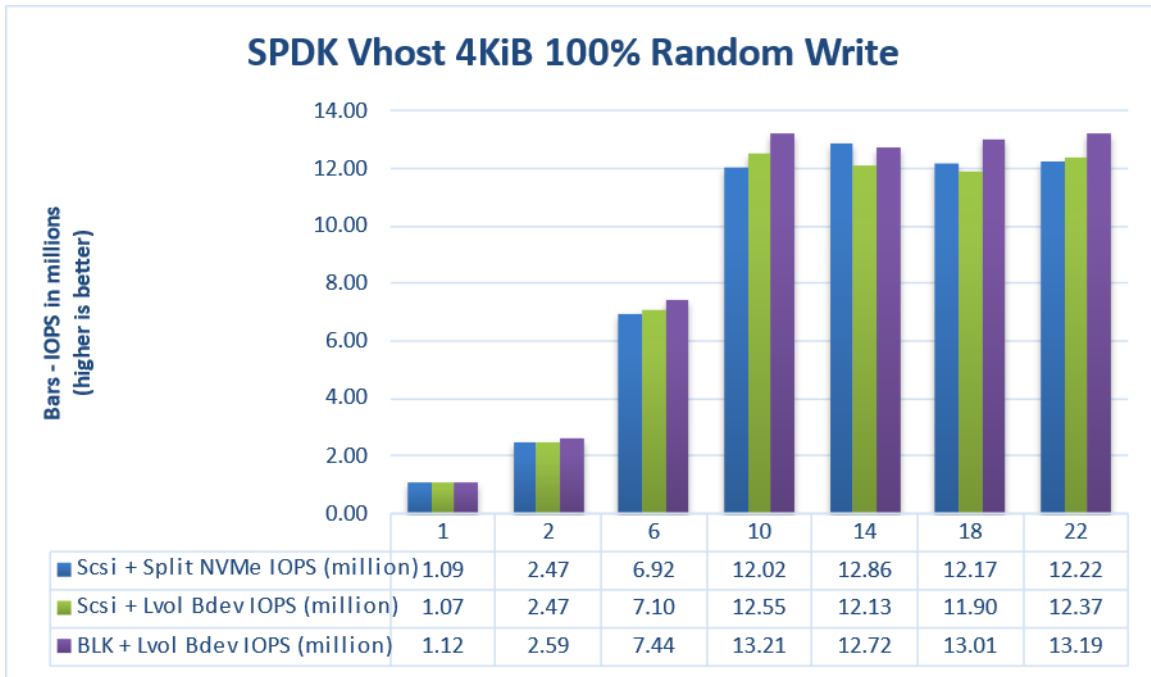


Figure 3: Comparison of performance between various SPDK Vhost stack-bdev combinations for 4KiB Random Write QD=64 workload

4KiB Random Read-Write Results

Table 8: SPDK Vhost core scaling results, 4KiB Random 70% Read 30% Write IOPS, QD=64

# of CPU cores	# of VMs	Stack / Backend	IOPS (millions)
1	2	SCSI / Split NVMe Bdev	1.32
		SCSI / Lvol Bdev	1.38
		BLK / Lvol Bdev	1.47
2	4	SCSI / Split NVMe Bdev	2.78
		SCSI / Lvol Bdev	2.90
		BLK / Lvol Bdev	3.00
6	12	SCSI / Split NVMe Bdev	7.84
		SCSI / Lvol Bdev	7.83
		BLK / Lvol Bdev	8.33
10	20	SCSI / Split NVMe Bdev	12.92
		SCSI / Lvol Bdev	12.35
		BLK / Lvol Bdev	13.21
14	28	SCSI / Split NVMe Bdev	13.61
		SCSI / Lvol Bdev	12.52
		BLK / Lvol Bdev	13.26
18	36	SCSI / Split NVMe Bdev	14.71
		SCSI / Lvol Bdev	14.30
		BLK / Lvol Bdev	15.20
22	44	SCSI / Split NVMe Bdev	15.33
		SCSI / Lvol Bdev	15.49
		BLK / Lvol Bdev	16.68

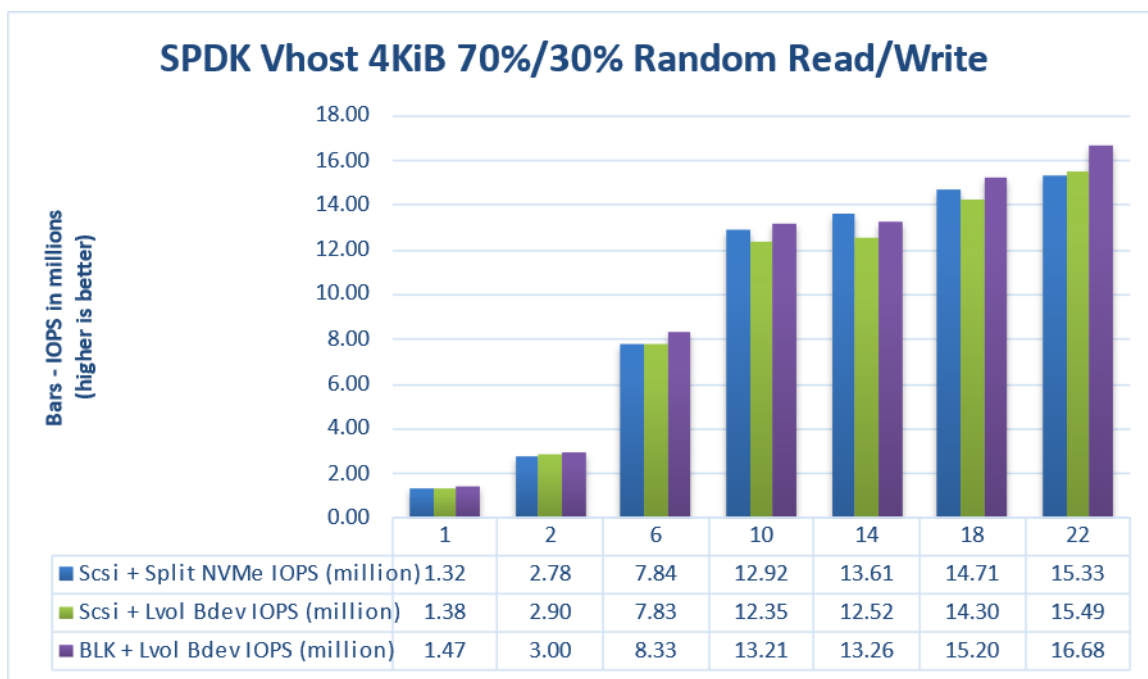


Figure 4: Comparison of performance between various SPDK Vhost stack-bdev combinations for 4KiB Random 70% Read 30% Write QD=64 workload

Logical Volumes performance impact

The SPDK Vhost SCSI tests were run using two bdev backends – Split NVMe and Logical Volumes. Both “Split NVMe Bdevs” and “Logical Volume Bdevs” allow to logically partition NVMe SSDs, the latter being more flexible in configuration. Here we measure the overhead of extra flexibility afforded by Logical Volumes.

Table 9: Logical Volumes performance impact for SPDK Vhost SCSI

Workload	# of CPU cores	# of VMs	Vhost SCSI + Split NVMe IOPS (millions)	Vhost SCSI + Lvol IOPS (millions)	Lvol Impact (%)
4KiB 100% Random Read	1	2	1.69	1.72	1.78%
	2	4	3.35	3.43	2.43%
	6	12	9.12	9.04	-0.90%
	10	20	14.68	13.62	-7.22%
	14	28	15.43	14.03	-9.10%
	18	36	16.85	16.39	-2.77%
	22	44	17.79	18.31	2.94%
4KiB 100% Random Write	1	2	1.09	1.07	-2.14%
	2	4	2.47	2.47	-0.08%
	6	12	6.92	7.10	2.65%
	10	20	12.02	12.55	4.39%
	14	28	12.86	12.13	-5.68%
	18	36	12.17	11.90	-2.22%
	22	44	12.22	12.37	1.24%
4KiB 70% Random Read 30% Random Write	1	2	1.32	1.38	3.91%
	2	4	2.78	2.90	4.30%
	6	12	7.84	7.83	-0.18%
	10	20	12.92	12.35	-4.40%
	14	28	13.61	12.52	-7.97%
	18	36	14.71	14.30	-2.83%
	22	44	15.33	15.49	1.02%

Packed Ring performance impact

Selected test cases were re-run to show benefits of using Packed Rings as an option when configuring SPDK Vhost BLK controllers. For this, an optional parameter “--packed_ring” must be used when creating a SPDK Vhost BLK controller. Packed Ring feature requires QEMU 4.2.0 or later.

Table 10: Packed Ring performance impact on SPDK Vhost BLK controllers.

Workload	# of CPU cores	# of VMs	IOPS (millions) Split Ring	IOPS (millions) Packed Ring	Avg. Latency (usec) Split Ring	Avg. Latency (usec) Packed Ring	Packed Ring IOPS impact (%)	Packed Ring Avg. Latency impact (%)
4KiB 100% Random Read QD=64	1	2	1.91	1.94	133.67	131.48	1.65%	-1.64%
	2	4	3.76	3.88	135.98	131.71	3.17%	-3.14%
	6	12	10.40	10.72	148.32	142.80	3.01%	-3.72%
	10	20	15.97	16.53	159.78	154.28	3.51%	-3.44%
	14	28	16.48	16.91	216.68	211.53	2.60%	-2.38%
	18	36	19.20	19.60	239.39	235.25	2.11%	-1.73%
	22	44	21.30	21.84	263.55	256.86	2.53%	-2.54%
4KiB 100% Random Write QD=64	1	2	1.10	1.12	247.77	238.24	1.61%	-3.85%
	2	4	2.54	2.59	203.82	210.00	1.76%	3.03%
	6	12	7.40	7.40	208.22	211.03	0.11%	1.35%
	10	20	13.00	13.23	202.45	195.15	1.71%	-3.60%
	14	28	13.67	13.87	265.05	256.26	1.49%	-3.32%
	18	36	13.24	13.42	346.76	346.10	1.36%	-0.19%
	22	44	13.07	13.34	430.68	426.14	2.05%	-1.05%
4KiB 70% Random Read 30% Random Write QD=64	1	2	1.49	1.52	175.91	169.30	1.98%	-3.76%
	2	4	3.00	3.05	169.24	166.32	1.65%	-1.73%
	6	12	8.68	8.72	177.09	175.23	0.45%	-1.05%
	10	20	14.00	14.26	182.97	179.36	1.86%	-1.97%
	14	28	14.29	14.60	251.02	245.16	2.20%	-2.33%
	18	36	16.05	16.02	289.21	285.93	-0.15%	-1.13%
	22	44	16.89	17.27	331.88	325.48	2.24%	-1.93%

Conclusions

1. For SPDK Vhost SCSI performance with split NVMe bdevs, we measured 1.69 million IOPS on one Vhost core for the 4KiB 100% Random Read workload. The single Vhost core IOPS for 4 KiB Random Write and 4KiB Random 70/30 Read/Write were 1.09 million and 1.32 million IOPS respectively. For all workloads, the IOPS scaled near linearly with addition of I/O processing cores up to 10 CPU cores. Peak performance was achieved at:

- 22 CPU cores with 17.79 million IOPS for Random Read workload
- 14 CPU cores with 12.86 million IOPS for Random Write workload
- 22 CPU cores with 15.33 million IOPS for Random Read/Write workload

2. For SPDK Vhost SCSI performance with with Logical Volume backend devices, we measured 1.72 million IOPS on one Vhost core for the 4KiB 100% Random Read workload. The single Vhost core IOPS for 4 KiB Random Write and 4KiB Random 70/30 Read/Write were 1.07 million and 1.38 million IOPS respectively. For all workloads, the IOPS scaled near linearly with addition of I/O processing cores up to 10 CPU cores.

Peak performance was achieved at:

- 22 CPU cores with 18.31 million IOPS for Random Read workload
- 10 CPU cores with 12.55 million IOPS for Random Write workload
- 22 CPU cores with 15.49 million IOPS for Random Read/Write workload

3. For SPDK Vhost BLK with Logical Volume backend devices, we measured 1.84 million IOPS on one Vhost core for the 4KiB 100% Random Read workload. The single Vhost core IOPS for 4 KiB Random Write and 4KiB Random 70/30 Read/Write were 1.12 million and 1.47 million IOPS respectively. For all workloads, the IOPS scaled near linearly with addition of I/O processing cores up to 10 CPU cores. Peak performance was achieved at:

- 22 CPU cores with 20.15 million IOPS for Random Read workload
- 10 CPU cores with 13.21 million IOPS for Random Write workload
- 22 CPU cores with 16.68 million IOPS for Random Read/Write workload

4. Using Logical Volumes has an impact of up to about 7-9% lower IOPS than when using Split NVMe block devices.
5. Using Packed Ring option instead of default Split Ring mode for SPDK Vhost BLK controllers results in minor performance improvement.

Test Case 2: Rate Limiting IOPS per VM

This test case was geared towards understanding how many VMs can be supported at a pre-defined Quality of Service of IOPS per Vhost device. Both read and write IOPS were rate limited for each Vhost device on each of the VMs and then VM density was compared between SPDK & the Linux Kernel. 25k IOPS per VM were chosen as the rate limiter using Linux cgroups2.

Each individual VM was running FIO with the following workloads:

- 4KiB 100% Random Read
- 4KiB 100% Random Write

The results in tables are average of 3 runs.

Table 11: Rate Limiting IOPS per VM test case configuration

Item	Description
Test case	Test rate limiting IOPS/VM to 25000 IOPS
Test configuration	<p>FIO Version: fio-3.28</p> <p>VM Configuration:</p> <ul style="list-style-type: none">• Common settings are described in the Virtual Machine Settings chapter.• Number of VMs: 24, 48, and 72• Each VM has a single Vhost device which is one of equal partitions of NVMe drive. Total number of partitions depends on run test case.<ul style="list-style-type: none">○ For 24 VMs: 24xNVMe * 1 partition per NVMe = 24 partitions○ For 48 VMs: 24xNVMe * 2 partitions per NVMe = 48 partitions○ For 72 VMs: 24xNVMe * 3 partitions per NVMe = 72 partitions• Devices on VMs were throttled to run at a maximum of 25k IOPS (read or write) <p>SPDK Vhost target configuration:</p> <ul style="list-style-type: none">• Test were run with both Vhost-scsi and Vhost-blk stacks.• The Vhost-scsi stack was run with Split NVMe bdevs and Logical Volume bdevs.• The Vhost-blk stack was run with Logical Volume bdevs.• Test were run with the Vhost target using 6 CPU cores (NUMA optimized). <p>Kernel Vhost-scsi configuration:</p> <ul style="list-style-type: none">• There was not CPU limitations set on Kernel Vhost threads.• NUMA optimizations were not explored.
FIO configuration run on each VM	[global] ioengine=libaio direct=1 rw=randrw rwmixread=100 (100% reads), 0 (100% writes)

	thread=1 norandommap=1 time_based=1 runtime=240s ramp_time=60s bs=4k iodepth=1 numjobs=1
--	---

Test Case 2 Results

Table 12: 4KiB 100% Random Reads QD=1 rate limiting test results, 6 Vhost CPU cores

# of VMs	Stack	Backend bdev	IOPS (k)	Avg Lat. (usec)
24 VMs	SPDK-SCSI	Split NVMe	292.81	81.70
	SPDK-SCSI	Logical Volume	291.70	82.00
	SPDK-BLK	Logical Volume	293.55	81.49
	Kernel-SCSI	Partitioned NVMe	254.82	93.99
48 VMs	SPDK-SCSI	Split NVMe	575.83	83.09
	SPDK-SCSI	Logical Volume	574.27	83.35
	SPDK-BLK	Logical Volume	577.60	82.86
	Kernel-SCSI	Partitioned NVMe	465.17	102.69
72 VMs	SPDK-SCSI	Split NVMe	793.91	90.25
	SPDK-SCSI	Logical Volume	795.22	90.13
	SPDK-BLK	Logical Volume	799.40	89.72
	Kernel-SCSI	Partitioned NVMe	667.88	107.22

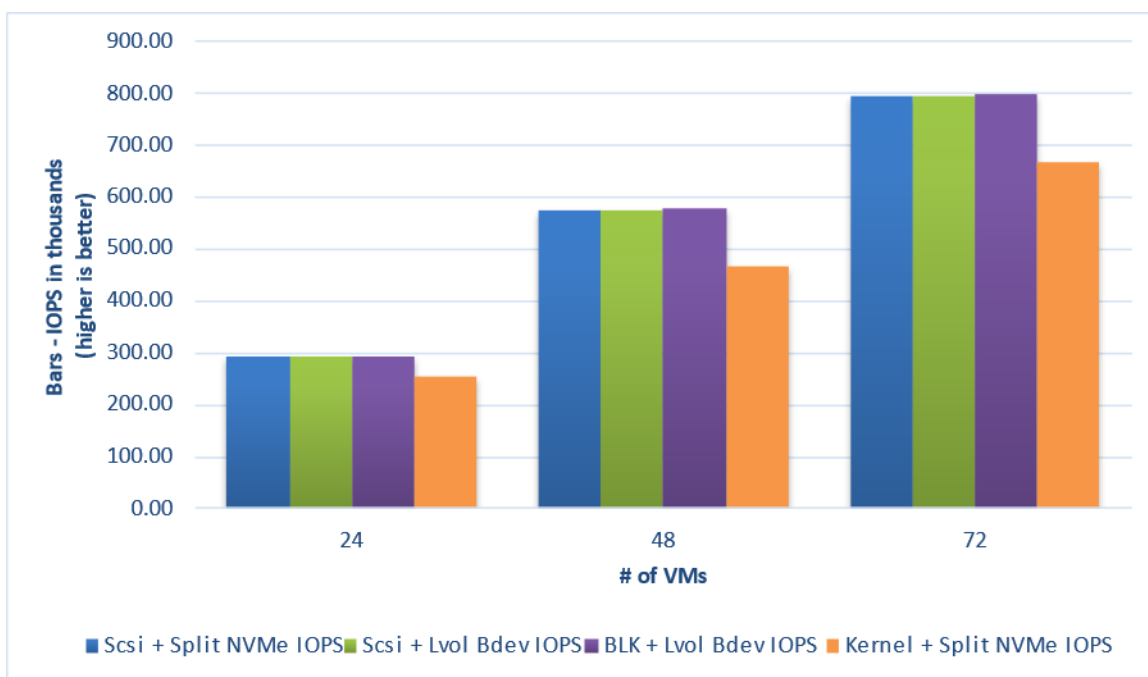


Figure 5: 4KiB 100% Random Reads IOPS, QD=1, throttling = 25k IOPS, 6 Vhost CPU cores

Table 13: 4KiB 100% Random Writes QD=1 rate limiting test results

# of VMs	Stack	Backend bdev	IOPS (k)	Avg Lat. (usec)
24 VMs	SPDK-SCSI	Split NVMe	600.00	39.74
	SPDK-SCSI	Logical Volume	600.00	39.74
	SPDK-BLK	Logical Volume	599.97	39.74
	Kernel-SCSI	Partitioned NVMe	599.96	39.73
48 VMs	SPDK-SCSI	Split NVMe	1199.97	39.74
	SPDK-SCSI	Logical Volume	1199.98	39.73
	SPDK-BLK	Logical Volume	1199.92	39.74
	Kernel-SCSI	Partitioned NVMe	1199.80	39.49
72 VMs	SPDK-SCSI	Split NVMe	1799.72	39.59
	SPDK-SCSI	Logical Volume	1799.72	39.59
	SPDK-BLK	Logical Volume	1799.74	39.61
	Kernel-SCSI	Partitioned NVMe	1524.28	46.54

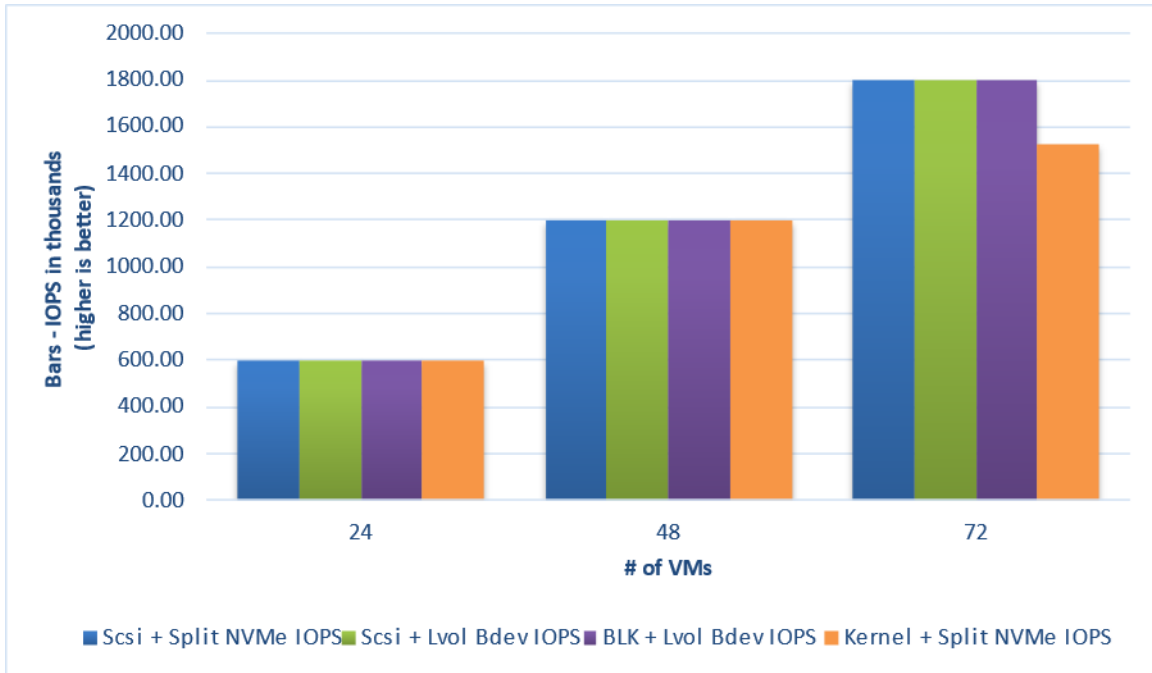


Figure 6: 4KiB 100% Random Writes IOPS, QD=1, throttling = 25k IOPS, 6 Vhost CPU cores

Conclusions

1. None of the tested Vhost solutions was able to serve 25,000 IOPS per VM for 4KiB Random Read workload.
2. Using 6 I/O processing cores, the SPDK Vhost server 25,000 IOPS per VM to up to 72 VMs for 4 KiB Random Write workload.
3. Throughput and average latencies were up to 1.23x and 1.17x times better for Random Read and Random Write workloads respectively when using SPDK Vhost as compared to Kernel Vhost.
4. Contrary to SPDK Vhost 23.01 performance report there was no CPU limitation imposed on Kernel-Vhost using Linux cgroups. This was caused by a bug in automation scripts which caused the cgroups to be ineffective and may be the reason for Kernel-Vhost being able to catch up with SPDK-Vhost performance.

Test Case 3: Performance per NVMe drive

This test case was performed to understand performance and efficiency of the Vhost scsi and blk process using SPDK vs. Linux Kernel with a single NVMe drive on 2 VMs. Each VM had a single Vhost device which is one of two equal partitions of an NVMe drive. Results in the table represent performance (IOPS, avg. latency & CPU %) seen from the VM. The VM was running FIO with the following workloads:

- 4KiB 100% Random Read
- 4KiB 100% Random Write
- 4KiB Random 70% Read 30% Write

The results in tables are average of 3 runs.

Table 14: Performance per NVMe drive test case configuration

Item	Description
Test case	Test SPDK Vhost target I/O core scaling performance
Test configuration	<p>FIO Version: fio-3.28</p> <p>VM Configuration:</p> <ul style="list-style-type: none"> • Common settings are described in the Virtual Machine Settings chapter. • 2 VMs were tested. • Each VM had a single Vhost device which was one of two equal partitions of a single NVMe drive. <p>SPDK Vhost target configuration:</p> <ul style="list-style-type: none"> • The SPDK Vhost process was run on a single, physical CPU core. • The Vhost-scsi stack was run with Split NVMe bdevs and Logical Volume bdevs. • The Vhost-blk stack was run with Logical Volume bdevs. <p>Kernel Vhost target configuration:</p> <ul style="list-style-type: none"> • There was not CPU limitations set on Kernel Vhost threads.
FIO configuration	<pre>[global] ioengine=libaio direct=1 rw=randrw rwmixread=100 (100% reads), 70 (70% reads, 30% writes), 0 (100% writes) thread=1 norandommap=1 time_based=1 runtime=240s ramp_time=60s bs=4k iodepth= {1, 8, 32, 64, 128, 192} numjobs=1</pre>

Test Case 3 results

SPDK Vhost-Scsi

Table 15: Performance per NVMe drive IOPS and latency results, SPDK SCSI stack

Access pattern	Backend	QD	Throughput (IOPS k)	Avg. latency (usec)
4KiB 100% Random Reads	Split NVMe	1	24.31	82.03
4KiB 100% Random Reads	Split NVMe	8	185.97	85.81
4KiB 100% Random Reads	Split NVMe	32	621.76	102.63
4KiB 100% Random Reads	Split NVMe	64	733.23	174.51
4KiB 100% Random Reads	Split NVMe	128	737.13	346.97
4KiB 100% Random Reads	Split NVMe	192	735.19	524.40
4KiB 100% Random Reads	Lvol	1	24.23	82.27
4KiB 100% Random Reads	Lvol	8	185.51	86.01
4KiB 100% Random Reads	Lvol	32	619.96	102.99
4KiB 100% Random Reads	Lvol	64	750.36	170.00
4KiB 100% Random Reads	Lvol	128	744.96	343.33
4KiB 100% Random Reads	Lvol	192	746.58	514.15
4KiB 100% Random Writes	Split NVMe	1	139.18	14.08
4KiB 100% Random Writes	Split NVMe	8	524.15	31.70
4KiB 100% Random Writes	Split NVMe	32	629.18	102.31
4KiB 100% Random Writes	Split NVMe	64	636.73	200.73
4KiB 100% Random Writes	Split NVMe	128	646.66	394.60
4KiB 100% Random Writes	Split NVMe	192	640.91	599.67
4KiB 100% Random Writes	Lvol	1	138.64	14.18
4KiB 100% Random Writes	Lvol	8	519.02	31.93
4KiB 100% Random Writes	Lvol	32	648.83	98.65
4KiB 100% Random Writes	Lvol	64	650.83	196.47
4KiB 100% Random Writes	Lvol	128	666.52	383.28
4KiB 100% Random Writes	Lvol	192	655.65	582.74
4KiB 70%/30% Random Read Writes	Split NVMe	1	32.47	61.95
4KiB 70%/30% Random Read Writes	Split NVMe	8	223.52	71.27
4KiB 70%/30% Random Read Writes	Split NVMe	32	573.26	111.39
4KiB 70%/30% Random Read Writes	Split NVMe	64	655.23	195.59
4KiB 70%/30% Random Read Writes	Split NVMe	128	686.13	372.63
4KiB 70%/30% Random Read Writes	Split NVMe	192	704.83	545.27
4KiB 70%/30% Random Read Writes	Lvol	1	32.33	60.50
4KiB 70%/30% Random Read Writes	Lvol	8	226.35	70.43
4KiB 70%/30% Random Read Writes	Lvol	32	570.03	111.94
4KiB 70%/30% Random Read Writes	Lvol	64	676.99	189.01
4KiB 70%/30% Random Read Writes	Lvol	128	692.50	370.12
4KiB 70%/30% Random Read Writes	Lvol	192	695.22	551.65

SPDK Vhost-Blk

Table 16: Performance per NVMe drive IOPS and latency results, SPDK BLK stack

Access pattern	Backend	QD	Throughput (IOPS k)	Avg. latency (usec)
4KiB 100% Random Reads	Lvol	1	24.34	81.88
4KiB 100% Random Reads	Lvol	8	186.52	85.49
4KiB 100% Random Reads	Lvol	32	631.38	101.37
4KiB 100% Random Reads	Lvol	64	832.06	153.93
4KiB 100% Random Reads	Lvol	128	833.69	306.35
4KiB 100% Random Reads	Lvol	192	832.18	460.35
4KiB 100% Random Writes	Lvol	1	141.55	13.91
4KiB 100% Random Writes	Lvol	8	538.79	30.86
4KiB 100% Random Writes	Lvol	32	674.38	94.71
4KiB 100% Random Writes	Lvol	64	701.22	180.96
4KiB 100% Random Writes	Lvol	128	705.45	362.40
4KiB 100% Random Writes	Lvol	192	713.67	537.30
4KiB 70%/30% Random Read Writes	Lvol	1	33.00	59.59
4KiB 70%/30% Random Read Writes	Lvol	8	222.54	71.71
4KiB 70%/30% Random Read Writes	Lvol	32	614.60	104.06
4KiB 70%/30% Random Read Writes	Lvol	64	722.94	176.96
4KiB 70%/30% Random Read Writes	Lvol	128	790.91	324.09
4KiB 70%/30% Random Read Writes	Lvol	192	816.46	470.61

Kernel Vhost-Scsi

Table 17: Performance per NVMe drive IOPS and latency results, Kernel Vhost-Scsi

Access pattern	Backend	QD	Throughput (IOPS k)	Avg. latency (usec)
4KiB 100% Random Reads	NVMe	1	20.71	96.82
4KiB 100% Random Reads	NVMe	8	152.64	104.44
4KiB 100% Random Reads	NVMe	32	434.59	146.89
4KiB 100% Random Reads	NVMe	64	570.42	224.09
4KiB 100% Random Reads	NVMe	128	656.36	389.80
4KiB 100% Random Reads	NVMe	192	652.50	590.35
4KiB 100% Random Writes	NVMe	1	70.87	28.00
4KiB 100% Random Writes	NVMe	8	303.78	52.57
4KiB 100% Random Writes	NVMe	32	541.47	118.91
4KiB 100% Random Writes	NVMe	64	629.61	201.69
4KiB 100% Random Writes	NVMe	128	640.51	400.07
4KiB 100% Random Writes	NVMe	192	655.37	585.13
4KiB 70%/30% Random Read Writes	NVMe	1	28.06	70.22
4KiB 70%/30% Random Read Writes	NVMe	8	177.54	89.82
4KiB 70%/30% Random Read Writes	NVMe	32	436.54	146.35
4KiB 70%/30% Random Read Writes	NVMe	64	556.86	229.70
4KiB 70%/30% Random Read Writes	NVMe	128	628.13	407.44
4KiB 70%/30% Random Read Writes	NVMe	192	628.12	608.91

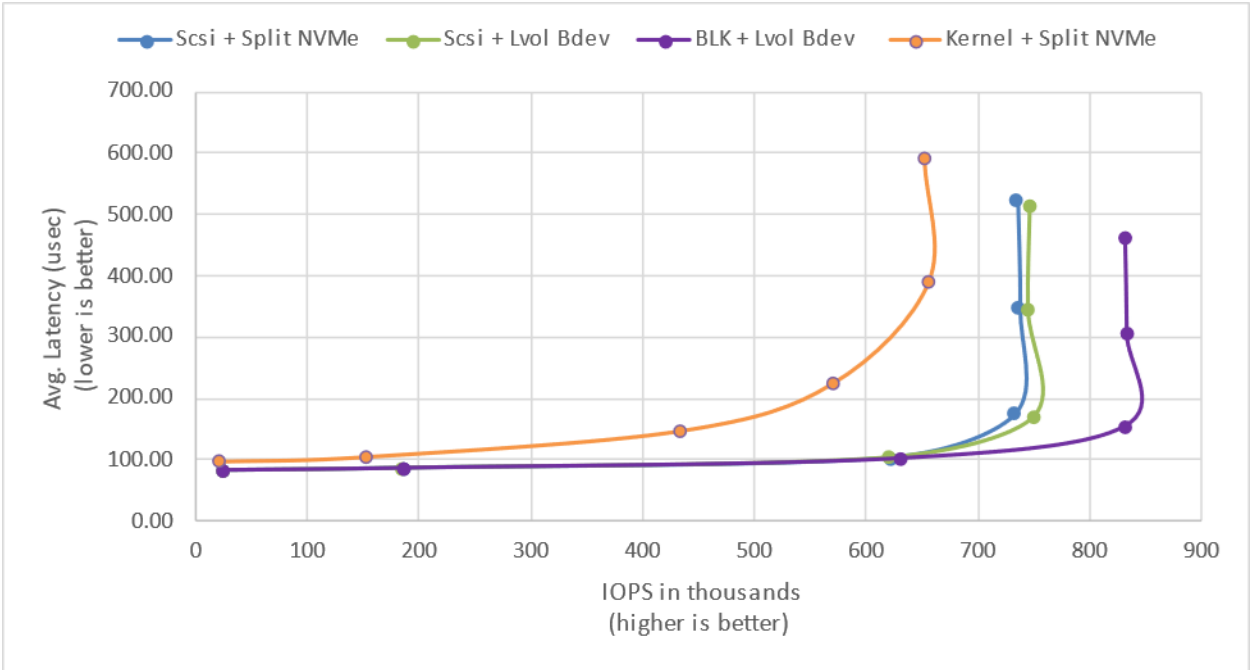


Figure 7: 4KiB 100% Random Reads IOPS and latency

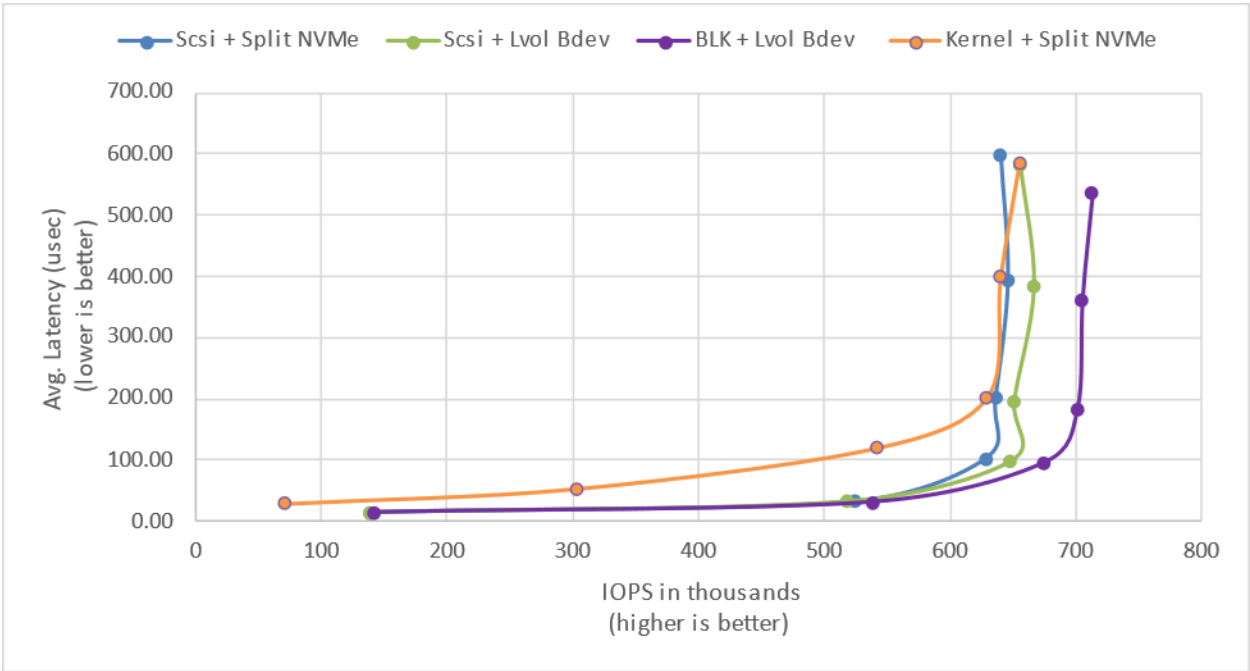


Figure 8: 4KiB 100% Random Writes IOPS and latency

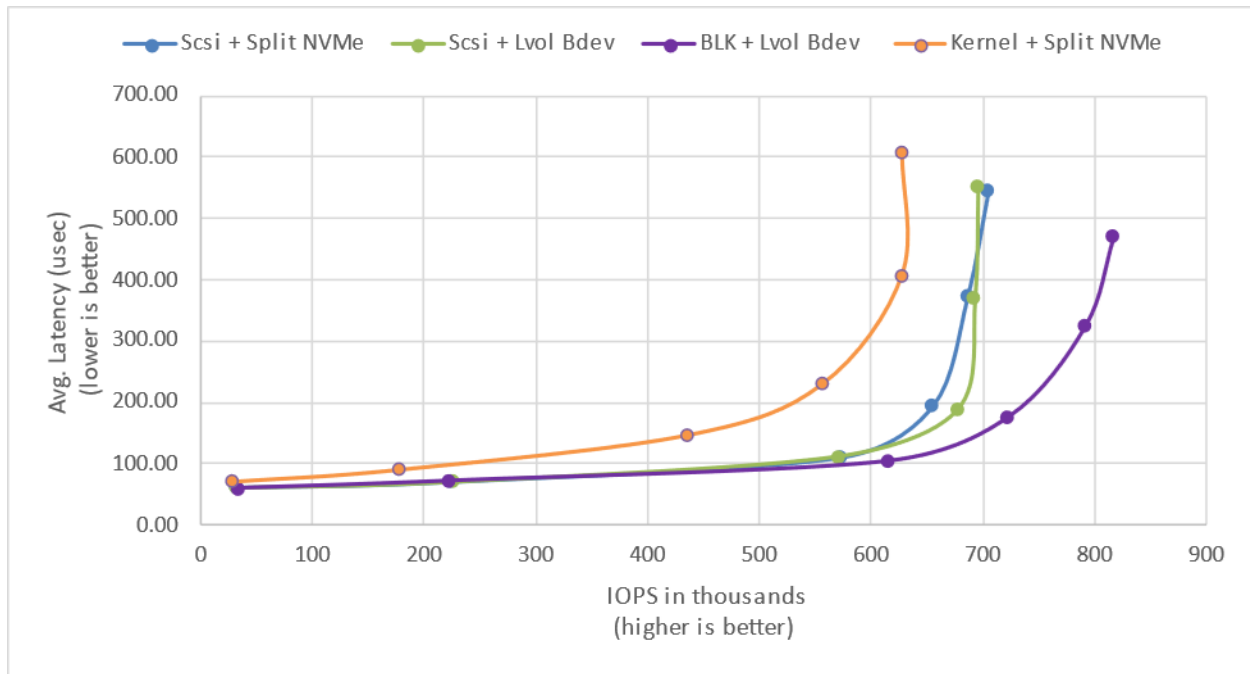


Figure 9: 4KiB 70%/30% Random Read/Write IOPS and latency

Conclusions

1. SPDK Vhost-scsi with NVMe Split bdevs has lower latency and higher throughput than Kernel Vhost-scsi in majority of workload / queue depth combinations.

Summary

This report compared performance results while running Vhost-scsi using traditional interrupt-driven kernel Vhost-scsi against the accelerated polled-mode driven SPDK implementation. Various local ephemeral configurations were demonstrated, including rate limiting IOPS, performance per VM, and maximum performance from an underlying system when comparing kernel vs. SPDK Vhost-scsi target implementations.

In addition, performance impacts of using SPDK Logical Volume Bdevs and the SPDK Vhost-blk stack were presented.

This report provided information regarding methodologies and practices while benchmarking Vhost-scsi and Vhost-blk using both SPDK and the Linux Kernel. It should be noted that the performance data showcased in this report is based on specific hardware and software configurations and that performance results may vary depending on different hardware and software configurations.

Appendix A - Vfiio-User Performance

Vfio-user is a framework that allows implementing PCI devices in userspace. Clients, such as Qemu, talk to the server over a UNIX socket using vfio-user protocol. For more information see: [libvfio-user](#).

SPDK has added a vfio-user transport to the nvme target. This allows presenting a virtual NVMe PCIe device to a client, such as a Qemu VM. This is analogous to how the SPDK vhost target presents virtual virtio PCIe devices to a Qemu VM.

The performance comparison between SPDK Vhost stacks versus vfio-user transport layer was done using [oracle/qemu fork](#) (vfio-user-p3.0), which supports running Qemu guest VMs with “vfio-user-pci” devices.

The vfio-user transport performance is roughly comparable with SPDK Vhost-BLK results with increasing number of SPDK target application CPU cores assigned.

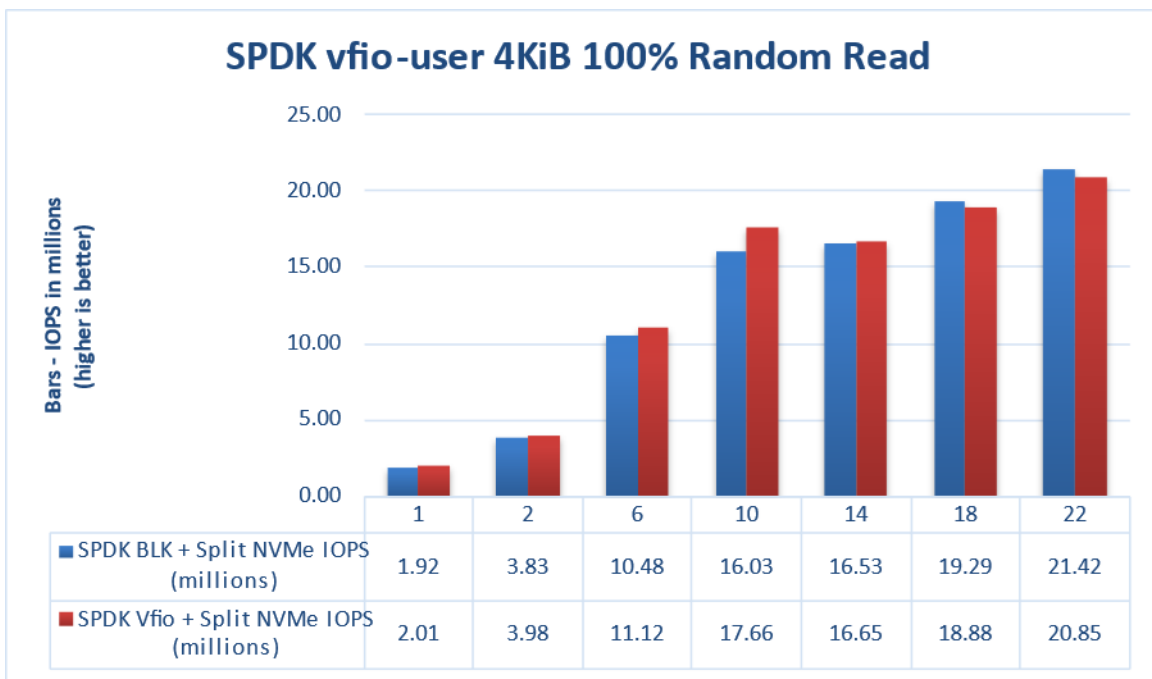


Figure 10: Comparison of performance between SPDK Vhost-BLK and Vfiio-User transport for 4KiB Random Read QD=64 workload

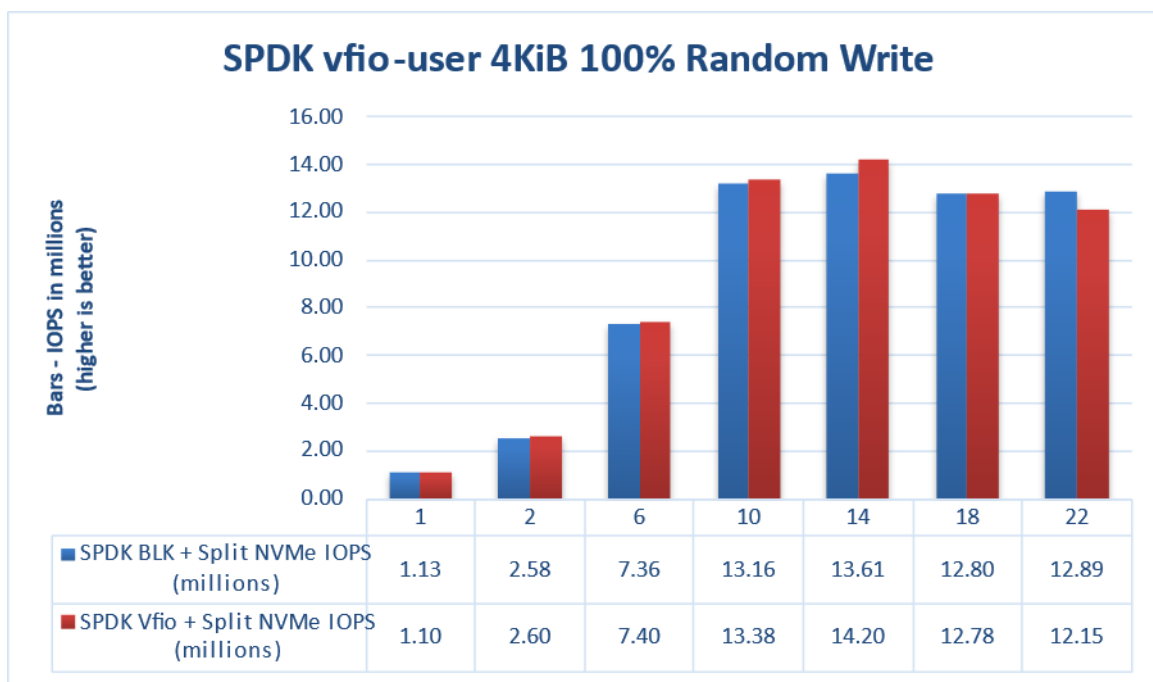


Figure 11: Comparison of performance between SPDK Vhost-BLK and Vfiio-User transport for 4KiB Random Write QD=64 workload

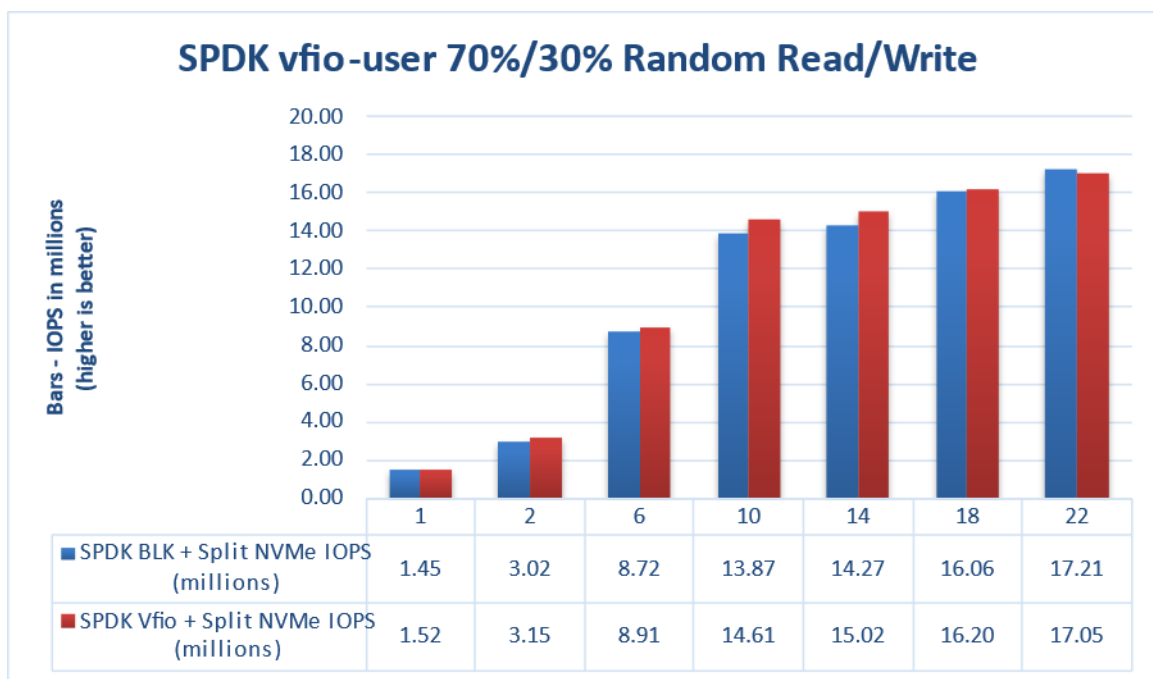


Figure 12: Comparison of performance between SPDK Vhost-BLK and Vfiio-User transport for 4KiB Random Read/Write QD=64 workload

List of Tables

Table 1: Hardware setup configuration	4
Table 2: Test platform BIOS settings	5
Table 3: Test System NVMe storage setup	5
Table 4: Guest VM configuration	5
Table 5: SPDK Vhost Core Scaling test configuration	9
Table 6: SPDK Vhost core scaling results, 4KiB 100% Random Reads IOPS, QD=64	11
Table 7: SPDK Vhost core scaling results, 4KiB 100% Random Write IOPS, QD=64	12
Table 8: SPDK Vhost core scaling results, 4KiB Random 70% Read 30% Write IOPS, QD=64	13
Table 9: Logical Volumes performance impact for SPDK Vhost SCSI	14
Table 10: Packed Ring performance impact on SPDK Vhost BLK controllers.	15
Table 11: Rate Limiting IOPS per VM test case configuration	17
Table 12: 4KiB 100% Random Reads QD=1 rate limiting test results, 6 Vhost CPU cores	19
Table 13: 4KiB 100% Random Writes QD=1 rate limiting test results	20
Table 14: Performance per NVMe drive test case configuration	22
Table 15: Performance per NVMe drive IOPS and latency results, SPDK SCSI stack	23
Table 16: Performance per NVMe drive IOPS and latency results, SPDK BLK stack	24
Table 17: Performance per NVMe drive IOPS and latency results, Kernel Vhost-Scsi	24

List of Figures

Figure 1: SPDK Vhost-scsi architecture	7
Figure 2: Comparison of performance between various SPDK Vhost stack-bdev combinations for 4KiB Random Read QD=64 workload	11
Figure 3: Comparison of performance between various SPDK Vhost stack-bdev combinations for 4KiB Random Write QD=64 workload	12
Figure 4: Comparison of performance between various SPDK Vhost stack-bdev combinations for 4KiB Random 70% Read 30% Write QD=64 workload	13
Figure 5: 4KiB 100% Random Reads IOPS, QD=1, throttling = 25k IOPS, 6 Vhost CPU cores	19
Figure 6: 4KiB 100% Random Writes IOPS, QD=1, throttling = 25k IOPS, 6 Vhost CPU cores	20
Figure 7: 4KiB 100% Random Reads IOPS and latency	25
Figure 8: 4KiB 100% Random Writes IOPS and latency	25
Figure 9: 4KiB 70%/30% Random Read/Write IOPS and latency	26
Figure 10: Comparison of performance between SPDK Vhost-BLK and Vfiio-User transport for 4KiB Random Read QD=64 workload	28
Figure 11: Comparison of performance between SPDK Vhost-BLK and Vfiio-User transport for 4KiB Random Write QD=64 workload	29
Figure 12: Comparison of performance between SPDK Vhost-BLK and Vfiio-User transport for 4KiB Random Read/Write QD=64 workload	29

Notices & Disclaimers

Performance varies by use, configuration and other factors. Learn more at [www.Intel.com/PerformanceIndex](https://www.intel.com/PerformanceIndex).

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.

Your costs and results may vary.

No product or component can be absolutely secure.

Intel technologies may require enabled hardware, software or service activation.

© Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others.