SPDK vhost performance report
Release 19.10

Testing Date: November 2019

Performed by: Karol Latecki

Acknowledgments:

John Kariuki (john.k.kariuki@intel.com)

Vishal Verma (vishal4.verma@intel.com)

Revision History

Date	Revision	Comment
25/11/19	1.0	Test runs finished
27/11/19	1.0	Draft version of the document created

Contents

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 information between SPDK vhost-scsi and Linux Kernel vhost-scsi software stacks under various test cases.

The purpose of 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>
<tr><th>Item</th><th>Description</th></tr>
<tr><td>Server Platform</td><td>Intel WolfPass
R2224WFTZS
</td></tr>
<tr><td>Motherboard</td><td>S2600WFT</td></tr>
<tr><td>CPU</td><td>Intel(R) Xeon(R) Gold 6230N CPU @ 2.30GHz
Number of cores 20, number of threads 40</td></tr>
<tr><td>Memory</td><td>10 x 32GB Micron DDR4 36ASF4G72PZ-2G6H1R
Total 320 GBs
Memory channel population:
<table>
<tr><th>P1</th><th>P2</th></tr>
<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>
</table></td></tr>
<tr><td>Operating System</td><td>Fedora 29</td></tr>
<tr><td>BIOS</td><td>02.01.0008 (02.04.2019)</td></tr>
<tr><td>Linux kernel version</td><td>5.1.20-200.fc29.x86_64</td></tr>
<tr><td>SPDK version</td><td>SPDK 19.10</td></tr>
<tr><td>Qemu version</td><td>QEMU emulator version 3.0.1 (qemu-3.0.1-4.fc29)</td></tr>
<tr><td>Storage</td><td>OS: 1x 120GB Intel SSDSC2BB120G4
Storage:
24x Intel® P4610™ 1.6TBs (FW: VDV10140) (6 on CPU NUMA Node 0, 18 on CPU NUMA Node 1)</td></tr>
</table>

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

Virtual Machine Settings

Common settings used for all VMs used in tests.

Item	Description
CPU	2vCPU, pass through from physical host server. Explicit core usage on enforced using "taskset –a –c" command on host. Related QEMU arguments used for starting the VM: -cpu host -smp 1
Memory	4 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. Related QEMU arguments: -m 4096 -object memory-backend-file,id=mem,size=4096M,mem-path=/dev/hugepages,share=on,prealloc=yes,host-nodes=0,policy=bind
Operating System	Fedora 29
Linux kernel version	5.1.20-200.fc29.x86_64
Additional boot options in /etc/default/grub	• Multi queue enabled: scsi_mod.use_blk_mq=1 • Spectre-meltdown patches disabled: spectre_v2=off nopti

Kernel & BIOS spectre-meltdown information

Host server system uses 5.1.20 kernel version available from DNF repository with default patches for spectre meltdown issue enabled.

Guest VM systems use 5.1.20 kernel version available from DNF repository, but with spectre-meltdown patches disabled. Following options are added to GRUB options in /etc/default/grub:

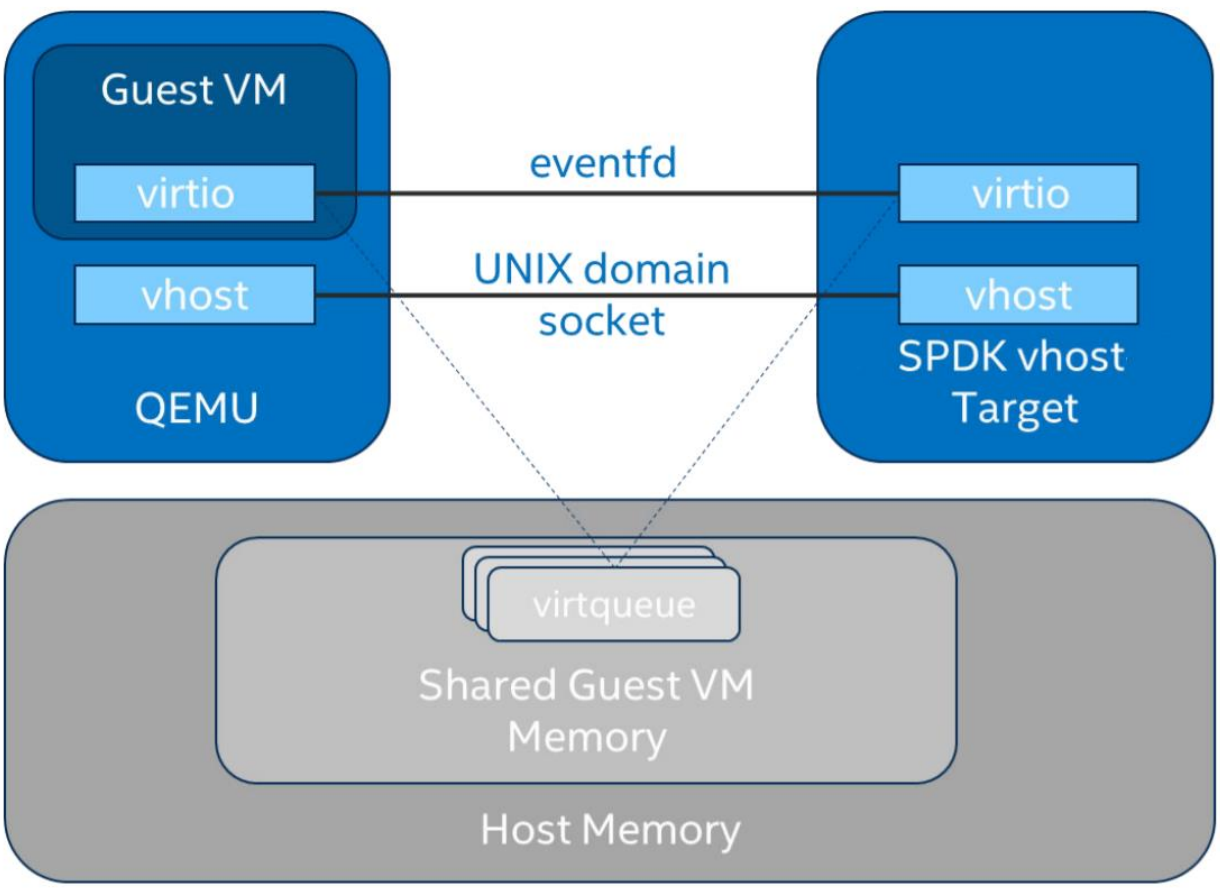
spectre_v2=off nopti

Introduction to SPDK vhost target

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

SPDK vhost target working

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



This report is prepared to show the performance comparisons between traditional interrupt-driven kernel vhost-scsi vs. accelerated polled-mode driven SPDK vhost-scsi under 4 different test cases using local NVMe storage. In addition, SPDK vhost-blk stack is also included in the report for further comparison with scsi stack.

Test Case 1: SPDK vhost core scaling

This test case was performed in order to understand aggregate VM performance with SPDK vhost I/O core scaling. We ran 48 virtual machines, each running following FIO workloads:

- 4KB 100% Random Read
- 4KB 100% Random Write
- 4KB 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 12 and measured the throughput (in IOPS) and latency. The number of VMs between test runs was not constant and was increased by 6 for each Vhost CPU added, up to maximum 36 VMs. VM number was not increased beyond 36 because of the platform capabilities in terms of available CPU cores.

FIO was run in a client-server mode. Each VM was running a FIO server and the host server distributed jobs as a client. This allowed us to start FIO jobs across all VMs at the same time. Gtod_reduce=1 option was used to disable FIO latency measurements which allowed better IOPS and bandwidth results.

Results in the table and chart represent aggregate performance (IOPS and average latency) seen across all the VMs.

Item	Description
Test case	Test SPDK vhost target I/O core scaling performance
Test configuration	**FIO Version**: fio-3.3 **VM Configuration**: • Common settings described in **Virtual Machine Settings** chapter • Number of VMs: variable (6 VMs per 1 Vhost CPU core, up to 36 VMs max) • Each VM has a single vhost device as target for FIO workload. This is achieved by splitting SPDK NVMe bdevs by using either split vbdevs or lvol bdevs in configuration. **SPDK vhost target configuration:** • Test run with vhost-scsi and vhost-blk stacks • Vhost-scsi stack run with Split NVMe bdevs and Logical Volume bdevs • Vhost-blk stack run with Logical Volume bdevs • Test run with 1,2,3,4,5,6,8,10 and 12 cores for each stack-bdev combination **Kernel vhost target configuration:** - N/A
FIO configuration	[global] ioengine=libaio direct=1

	thread=1 norandommap=1 time_based=1 gtod_reduce=1 ramp_time=60s runtime=240s numjobs=1 bs=4k rw=randrw rwmixread=100 (100% reads), 70 (70% reads, 30% writes), 0 (100% writes) iodepth={1, 8, 32, 64}

4k Random Read Results

Table 1: 4k 100% Random Reads IOPS, QD=32

# of CPU cores	# of VMs	Stack / Backend	IOPS (millions)
1	6	SCSI / Split NVMe Bdev	0.80
		SCSI / Lvol Bdev	0.88
		BLK / Lvol Bdev	0.90
2	12	SCSI / Split NVMe Bdev	2.23
		SCSI / Lvol Bdev	1.63
		BLK / Lvol Bdev	1.52
3	18	SCSI / Split NVMe Bdev	2.52
		SCSI / Lvol Bdev	1.97
		BLK / Lvol Bdev	2.26
4	24	SCSI / Split NVMe Bdev	3.47
		SCSI / Lvol Bdev	2.74
		BLK / Lvol Bdev	2.87
5	30	SCSI / Split NVMe Bdev	3.89
		SCSI / Lvol Bdev	3.27
		BLK / Lvol Bdev	3.16
6	36	SCSI / Split NVMe Bdev	4.40
		SCSI / Lvol Bdev	4.11
		BLK / Lvol Bdev	3.56
8	36	SCSI / Split NVMe Bdev	4.52
		SCSI / Lvol Bdev	4.33
		BLK / Lvol Bdev	4.60
10	36	SCSI / Split NVMe Bdev	5.26
		SCSI / Lvol Bdev	4.52
		BLK / Lvol Bdev	4.57
12	36	SCSI / Split NVMe Bdev	5.39
		SCSI / Lvol Bdev	5.63
		BLK / Lvol Bdev	5.88

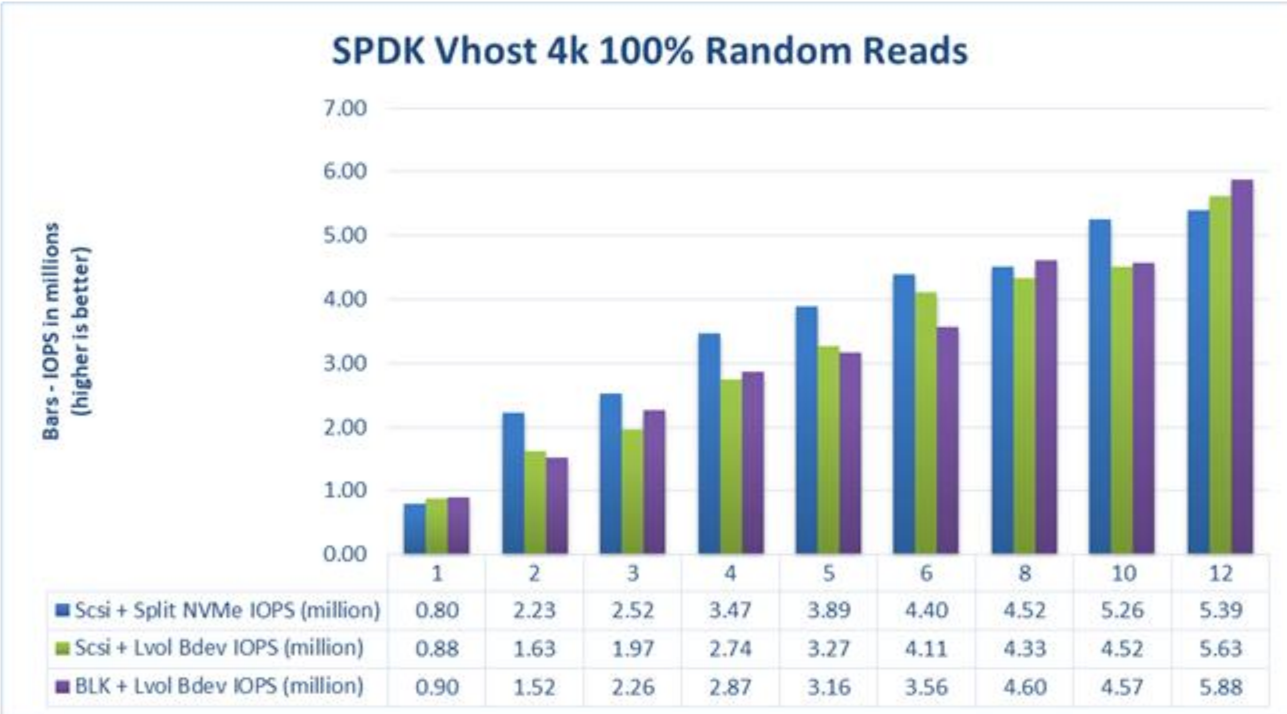


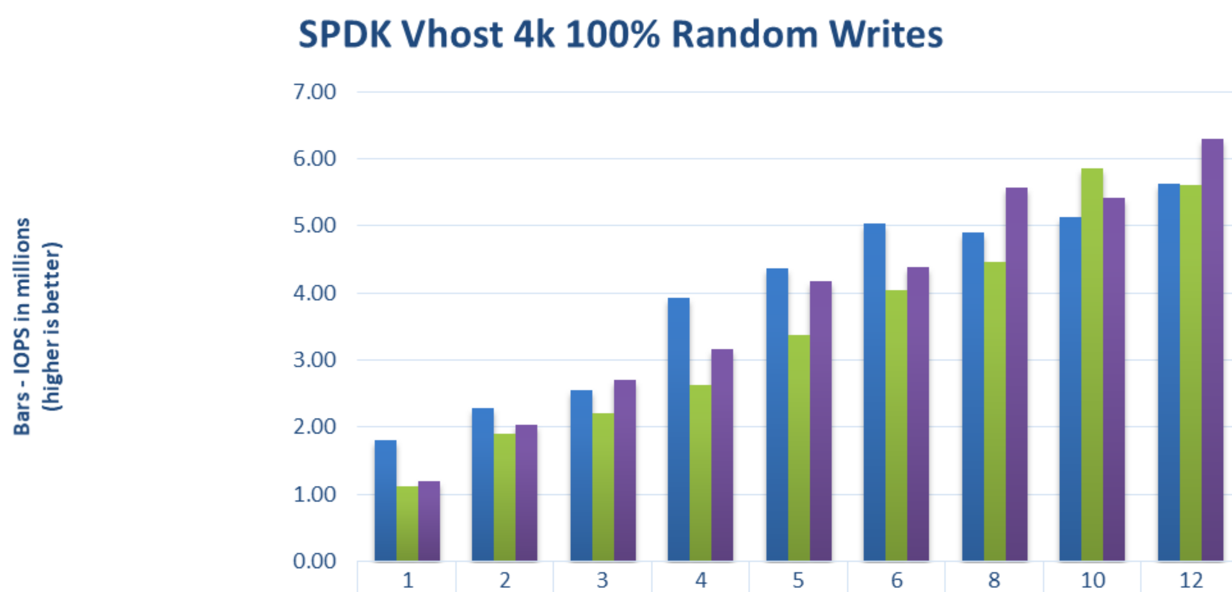
Figure 1: Comparison of performance between various SPDK Vhost stack-bdev combinations for 4k Random Read QD=32 workload

4k Random Write Results

Table 2: 4k 100% Random Write IOPS, QD=32

# of CPU cores	# of VMs	Stack / Backend	IOPS (millions)
1	6	SCSI / Split NVMe Bdev	1.80
		SCSI / Lvol Bdev	1.11
		BLK / Lvol Bdev	1.20
2	12	SCSI / Split NVMe Bdev	2.29
		SCSI / Lvol Bdev	1.90
		BLK / Lvol Bdev	2.03
3	18	SCSI / Split NVMe Bdev	2.55
		SCSI / Lvol Bdev	2.20
		BLK / Lvol Bdev	2.70
4	24	SCSI / Split NVMe Bdev	3.93
		SCSI / Lvol Bdev	2.63
		BLK / Lvol Bdev	3.17
5	30	SCSI / Split NVMe Bdev	4.36
		SCSI / Lvol Bdev	3.38
		BLK / Lvol Bdev	4.18

6	36	SCSI / Split NVMe Bdev	5.04
		SCSI / Lvol Bdev	4.04
		BLK / Lvol Bdev	4.38
8	36	SCSI / Split NVMe Bdev	4.89
		SCSI / Lvol Bdev	4.46
		BLK / Lvol Bdev	5.57
10	36	SCSI / Split NVMe Bdev	5.14
		SCSI / Lvol Bdev	5.85
		BLK / Lvol Bdev	5.42
12	36	SCSI / Split NVMe Bdev	5.63
		SCSI / Lvol Bdev	5.61
		BLK / Lvol Bdev	6.29



	1	2	3	4	5	6	8	10	12
Scsi + Split NVMe IOPS (million)	1.80	2.29	2.55	3.93	4.36	5.04	4.89	5.14	5.63
Scsi + Lvol Bdev IOPS (million)	1.11	1.90	2.20	2.63	3.38	4.04	4.46	5.85	5.61
BLK + Lvol Bdev IOPS (million)	1.20	2.03	2.70	3.17	4.18	4.38	5.57	5.42	6.29

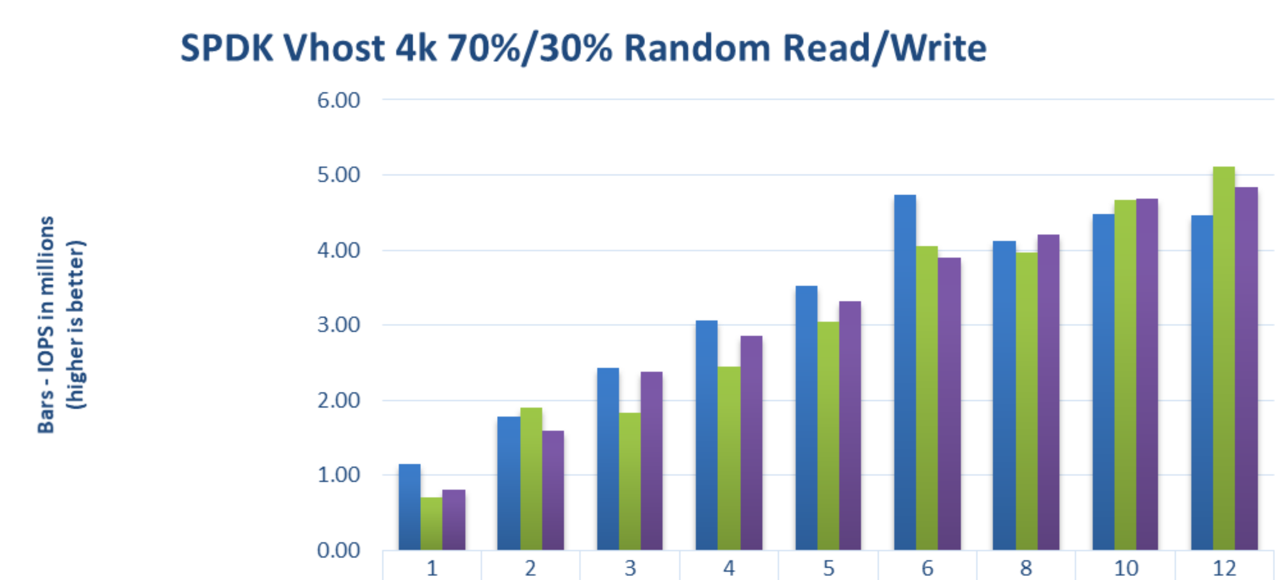
Figure 2: Comparison of performance between various SPDK Vhost stack-bdev combinations for 4k Random Write QD=32 workload

4k Random Read-Write Results

Table 3: 4k Random 70% Read 30% Write IOPS, QD=32

# of CPU cores	# of VMs	Stack / Backend	IOPS (millions)
1	6	SCSI / Split NVMe Bdev	1.15
		SCSI / Lvol Bdev	0.71
		BLK / Lvol Bdev	0.81

2	12	SCSI / Split NVMe Bdev	1.78
		SCSI / Lvol Bdev	1.90
		BLK / Lvol Bdev	1.59
3	18	SCSI / Split NVMe Bdev	2.44
		SCSI / Lvol Bdev	1.83
		BLK / Lvol Bdev	2.38
4	24	SCSI / Split NVMe Bdev	3.06
		SCSI / Lvol Bdev	2.44
		BLK / Lvol Bdev	2.85
5	30	SCSI / Split NVMe Bdev	3.52
		SCSI / Lvol Bdev	3.04
		BLK / Lvol Bdev	3.32
6	36	SCSI / Split NVMe Bdev	4.73
		SCSI / Lvol Bdev	4.05
		BLK / Lvol Bdev	3.90
8	36	SCSI / Split NVMe Bdev	4.11
		SCSI / Lvol Bdev	3.97
		BLK / Lvol Bdev	
10	36	SCSI / Split NVMe Bdev	4.49
		SCSI / Lvol Bdev	4.67
		BLK / Lvol Bdev	4.68
12	36	SCSI / Split NVMe Bdev	4.46
		SCSI / Lvol Bdev	5.10
		BLK / Lvol Bdev	4.83



	1	2	3	4	5	6	8	10	12
Scsi + Split NVMe IOPS (million)	1.15	1.78	2.44	3.06	3.52	4.73	4.11	4.49	4.46
Scsi + Lvol Bdev IOPS (million)	0.71	1.90	1.83	2.44	3.04	4.05	3.97	4.67	5.10
BLK + Lvol Bdev IOPS (million)	0.81	1.59	2.38	2.85	3.32	3.90	4.20	4.68	4.83

Figure 3: Comparison of performance between various SPDK Vhost stack-bdev combinations for 4k Random 70% Read 30% Write QD=32 workload

Conclusions

1. SPDK vhost performance scales near linearly up to 10 CPU cores. There is none or small improvement when increasing to 12 CPUs, which might be because there is no enough CPU resources on the platform.
   (With 10 and 12 CPU cores for Vhost and 36VMs some of the VMs need to share CPU resources or be cross-NUMA configured in respect to NVMe hardware)

2. BLK stack scaling is not linear. There are visible and reproducible performance drops, especially at 12 CPU cores used for Vhost process. This might suggest some software problem.

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 & Linux Kernel. 10K IOPS were chosen as the rate limiter using linux cgroups.

Note: For those comparing the results with 17.07 Vhost Performance Report - the rate limiter value was lowered to 10k IOPS because of the change in the hardware setup. P4610 1.6TB disks for workloads running with QD=1 are able to reach about 12-13k IOPS at most, so previous 20k IOPS limiter would never be reached.

Each individual VM was running FIO with the following workloads:

- 4KB 100% Random Read
- 4KB 100% Random Write

Item	Description
Test case	Test rate limiting IOPS/VM to 10000 IOPS

Test configuration	**FIO Version:** fio-3.3
	
	VM Configuration:
	• Common settings described in **Virtual Machine Settings** chapter
	• Total of 24 / 48 / 72 VMs
	• 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.
	  ○ 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 throttled to run at maximum of 10k IOPS (read and write)
	
	SPDK vhost target configuration:
	• Test run with vhost-scsi and vhost-blk stacks
	• Vhost-scsi stack run with Split NVMe bdevs and Logical Volume bdevs
	• Vhost-blk stack run with Logical Volume bdevs
	• Test run with 4 CPU cores (NUMA optimized)
	
	Kernel vhost-scsi configuration:
	• Used cgroups to limit vhost process to 4 cores
	• NUMA optimization 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=300s ramp_time=10s bs=4k iodepth=1 numjobs=1

Test Case 2 Results

Test result: 4K 100% Random Reads QD=1

# of VMs	Stack	Backend bdev	IOPS (k)	Avg Lat. (usec)	Host CPU utilization
24 VMs	SPDK-SCSI	Split NVMe	236.75	100.38	
	SPDK-SCSI	Logical Volume	238.88	99.23	
	SPDK-BLK	Logical Volume	239.29	99.10	
	Kernel-SCSI	Partitioned NVMe	150.36	161.67	
48 VMs	SPDK-SCSI	Split NVMe	474.64	99.66	
	SPDK-SCSI	Logical Volume	464.99	100.30	
	SPDK-BLK	Logical Volume	471.94	99.55	
	Kernel-SCSI	Partitioned NVMe	142.30	344.77	
72 VMs	SPDK-SCSI	Split NVMe	689.52	102.87	
	SPDK-SCSI	Logical Volume	687.00	102.23	
	SPDK-BLK	Logical Volume	692.61	101.95	
	Kernel-SCSI	Partitioned NVMe	221.09	349.33	

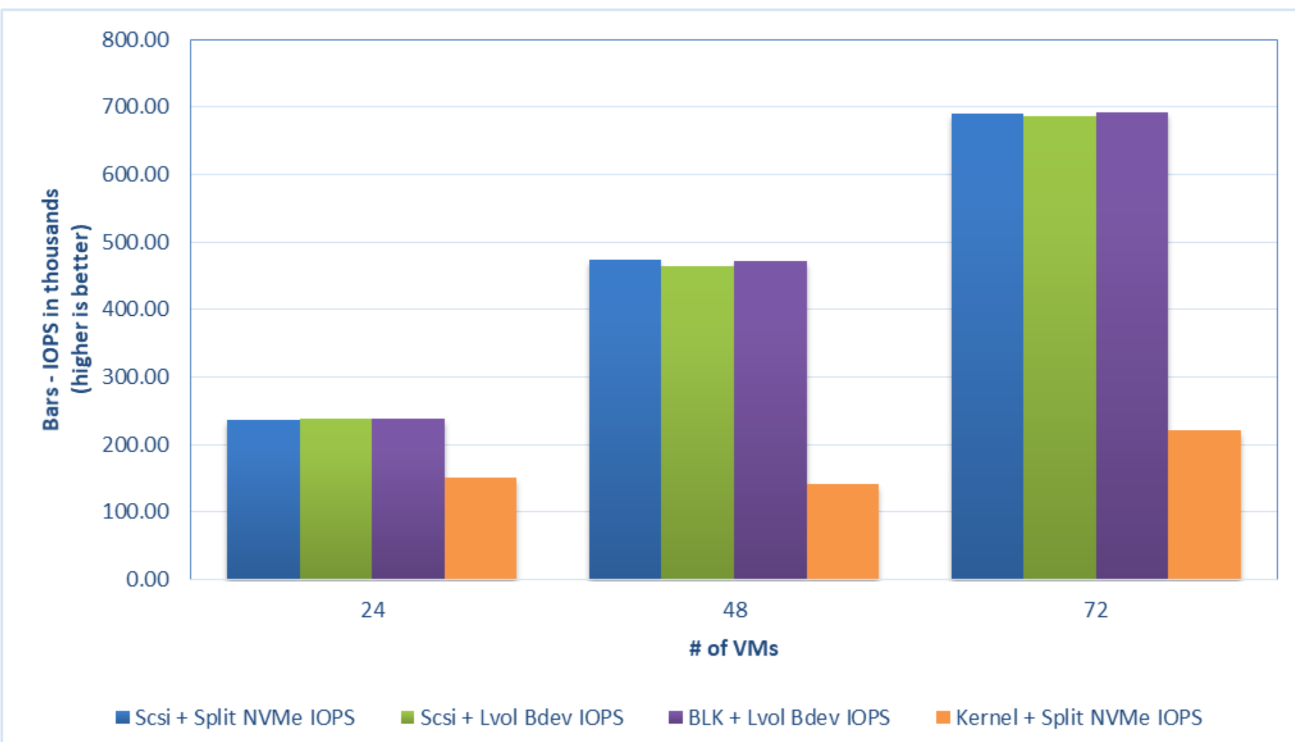


Figure 4: 4k 100% Random Reads IOPS and latency, QD=1, throttling = 10k IOPS

Test result: 4K 100% Random Writes QD=1

# of VMs	Stack	Backend bdev	IOPS (k)	Avg Lat. (usec)	Host CPU utilization
24 VMs	SPDK-SCSI	Split NVMe	240.00	98.51	
	SPDK-SCSI	Logical Volume	240.00	98.49	
	SPDK-BLK	Logical Volume	240.00	98.47	
	Kernel-SCSI	Partitioned NVMe	134.75	186.18	
48 VMs	SPDK-SCSI	Split NVMe	480.00	98.39	
	SPDK-SCSI	Logical Volume	479.99	98.35	
	SPDK-BLK	Logical Volume	480.00	98.37	
	Kernel-SCSI	Partitioned NVMe	147.00	349.65	
72 VMs	SPDK-SCSI	Split NVMe	719.93	98.30	
	SPDK-SCSI	Logical Volume	719.94	98.30	
	SPDK-BLK	Logical Volume	719.97	98.31	
	Kernel-SCSI	Partitioned NVMe	256.54	283.03	

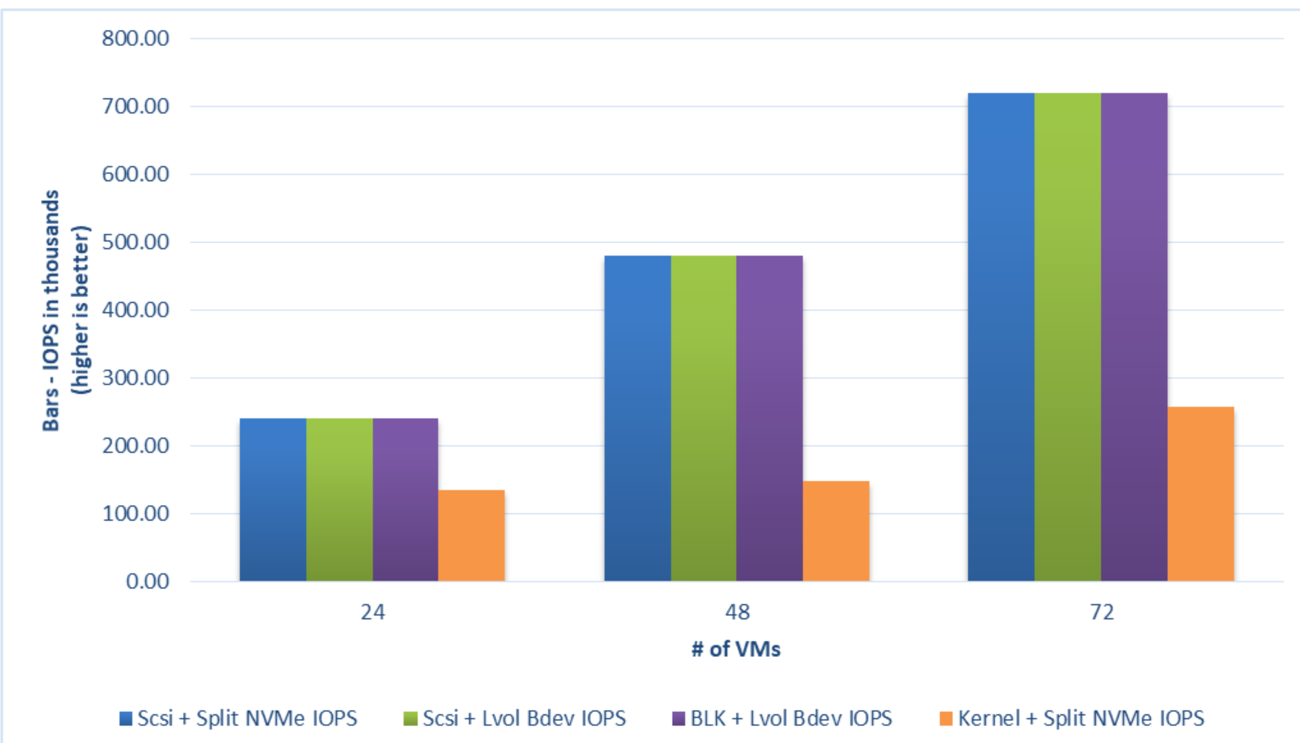


Figure 5: 4k 100% Random Writes IOPS and latency, QD=1, throttling = 10k IOPS

Conclusions

1. In most of the cases VMs using SPDK Vhost exposed devices were able to reach desired IOPS levels. One exception was 4k Random Read workload with 72 VMs configured, which was a bit lower than expected.

2. SPDK vhost was able to serve IO at desired level for increasing number of VMs.

3. Average latencies were up to 3.5x times better for Random Read and Random Write workloads in case of SPDK Vhost when compared to Kernel Vhost.

Note: Kernel-vhost process was not NUMA-optimized for this scenario. We found that using taskset or cgroups to limit kernel vhost to CPUs on 2 NUMA nodes causes the CPUs from the other node to be not used at all.

Test Case 3: Performance per NVMe drive

This test case was performed in order to understand performance and efficiency of the vhost scsi and blk process using SPDK vs. Linux Kernel with single NVMe drive on 2 VMs. Each VM has 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 following workloads:

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

Item	Description
Test case	Test SPDK vhost target I/O core scaling performance
Test configuration	**FIO Version:** fio-3.3 **VM Configuration**: • Common settings described in **Virtual Machine Settings** chapter • 2 VMs tested • Each VM has a single vhost device which is one of equal partitions of single NVMe drive. **SPDK vhost target configuration:** • SPDK vhost process run on a single, separate individual physical CPU core • Vhost-scsi stack run with Split NVMe bdevs and Logical Volume bdevs • Vhost-blk stack run with Logical Volume bdevs **Kernel vhost target configuration**: • Vhost process was run on separate individual physical core using cgroups.
FIO configuration	[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 numjobs=1

Test Case 3 results

SPDK Vhost-Scsi

Table: IOPS and latency results, SCSI stack

Access pattern	Backend	QD	Throughput (IOPS)	Avg. latency (usec)
4k 100% Random Reads	Split NVMe	1	19477.05	139.7
4k 100% Random Reads	Split NVMe	8	162600.41	104.03
4k 100% Random Reads	Split NVMe	32	439374.38	145.01
4k 100% Random Reads	Split NVMe	64	537829.43	237.98
4k 100% Random Reads	Lvol	1	20027.01	126.16
4k 100% Random Reads	Lvol	8	157057.92	93.92
4k 100% Random Reads	Lvol	32	439368.73	145.08
4k 100% Random Reads	Lvol	64	532520.48	243.77
4k 100% Random Writes	Split NVMe	1	129505.38	14.03
4k 100% Random Writes	Split NVMe	8	464621.08	33.53
4k 100% Random Writes	Split NVMe	32	496140.4	129.39
4k 100% Random Writes	Split NVMe	64	487387.06	261.23
4k 100% Random Writes	Lvol	1	130766.36	14.65
4k 100% Random Writes	Lvol	8	444462.39	35.4
4k 100% Random Writes	Lvol	32	491667.39	129.27
4k 100% Random Writes	Lvol	64	472954.86	269.48
4k 70%/30% Random Read Writes	Split NVMe	1	31822.15	62.099
4k 70%/30% Random Read Writes	Split NVMe	8	172381.15	108.555
4k 70%/30% Random Read Writes	Split NVMe	32	389797.37	163.468
4k 70%/30% Random Read Writes	Split NVMe	64	465005.94	280.394
4k 70%/30% Random Read Writes	Lvol	1	31709.07	62.438
4k 70%/30% Random Read Writes	Lvol	8	186396.87	85.251
4k 70%/30% Random Read Writes	Lvol	32	394948.44	161.578
4k 70%/30% Random Read Writes	Lvol	64	481164.06	262.532

SPDK Vhost-Blk

Table: IOPS and latency results, BLK stack

Access pattern	Backend	QD	Throughput (IOPS)	Avg. latency (usec)
4k 100% Random Reads	Lvol	1	15177.1	135.42
4k 100% Random Reads	Lvol	8	169655.95	92.67
4k 100% Random Reads	Lvol	32	446361	143.28
4k 100% Random Reads	Lvol	64	579263.89	220.55
4k 100% Random Writes	Lvol	1	109482.38	17.39
4k 100% Random Writes	Lvol	8	450804.26	35.66
4k 100% Random Writes	Lvol	32	491125.86	130.07
4k 100% Random Writes	Lvol	64	476960.29	268.66
4k 70%/30% Random Read Writes	Lvol	1	29162.43	61.93
4k 70%/30% Random Read Writes	Lvol	8	193458.02	82.102
4k 70%/30% Random Read Writes	Lvol	32	397032.87	157.937
4k 70%/30% Random Read Writes	Lvol	64	470333.4	257.44

Kernel Vhost-Scsi

Table: IOPS and latency results, Kernel Vhost-Scsi

Access pattern	Backend	QD	Throughput (IOPS)	Avg. latency (usec)
4k 100% Random Reads	NVMe	1	12917.92	164.86
4k 100% Random Reads	NVMe	8	70868.25	301.6
4k 100% Random Reads	NVMe	32	219263.46	291.18
4k 100% Random Reads	NVMe	64	214580.54	594.4
4k 100% Random Writes	NVMe	1	58274.64	34.86
4k 100% Random Writes	NVMe	8	110475.42	142.75
4k 100% Random Writes	NVMe	32	127096.91	392.31
4k 100% Random Writes	NVMe	64	163445.64	708.5
4k 70%/30% Random Read Writes	NVMe	1	19711.14	99.857
4k 70%/30% Random Read Writes	NVMe	8	70368.22	315.175
4k 70%/30% Random Read Writes	NVMe	32	154684.98	580.636
4k 70%/30% Random Read Writes	NVMe	64	198266.89	578.135

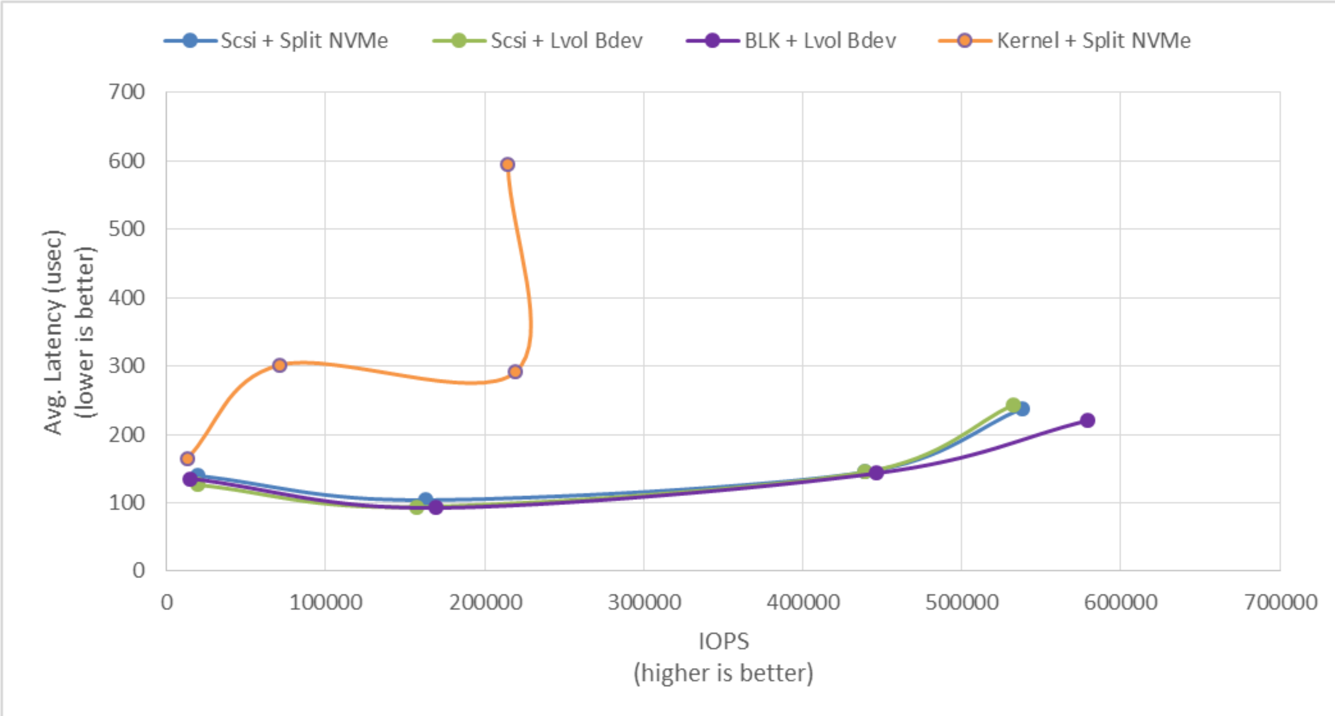


Figure 9: *4k 100% Random Reads IOPS and latency*

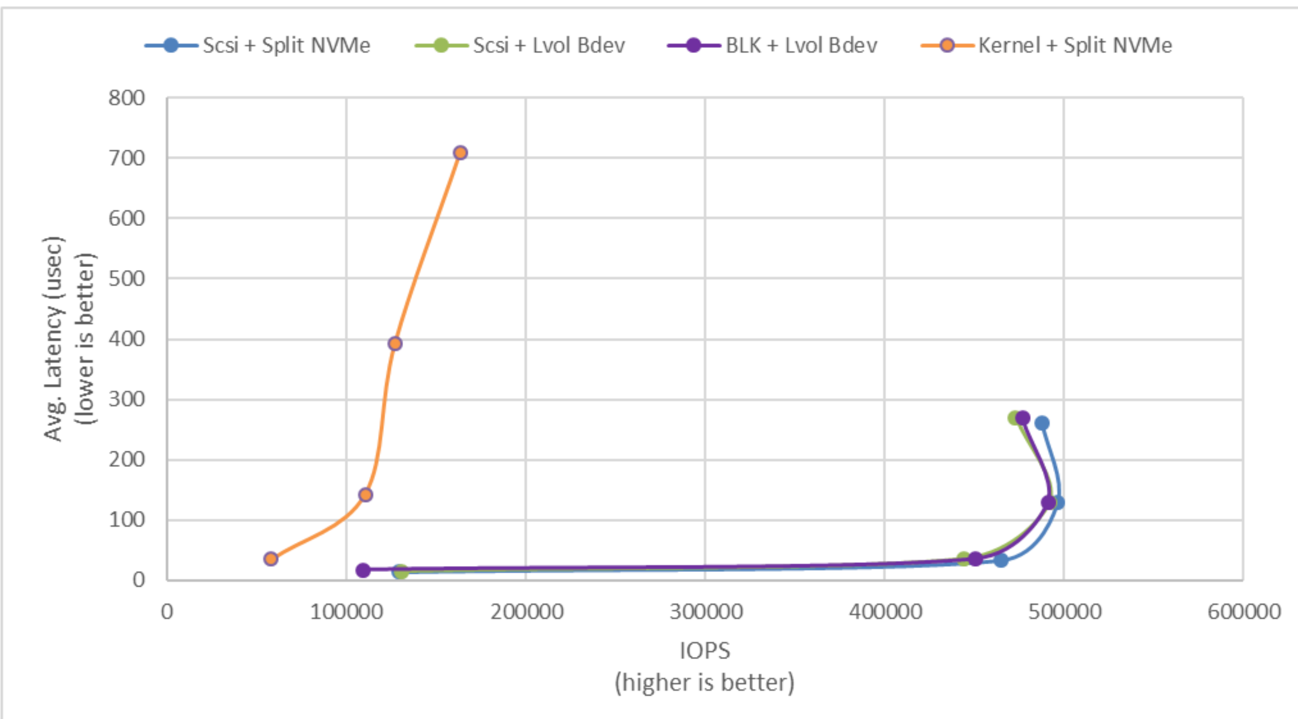


Figure 10: 4k 100% Random Writes IOPS and latency

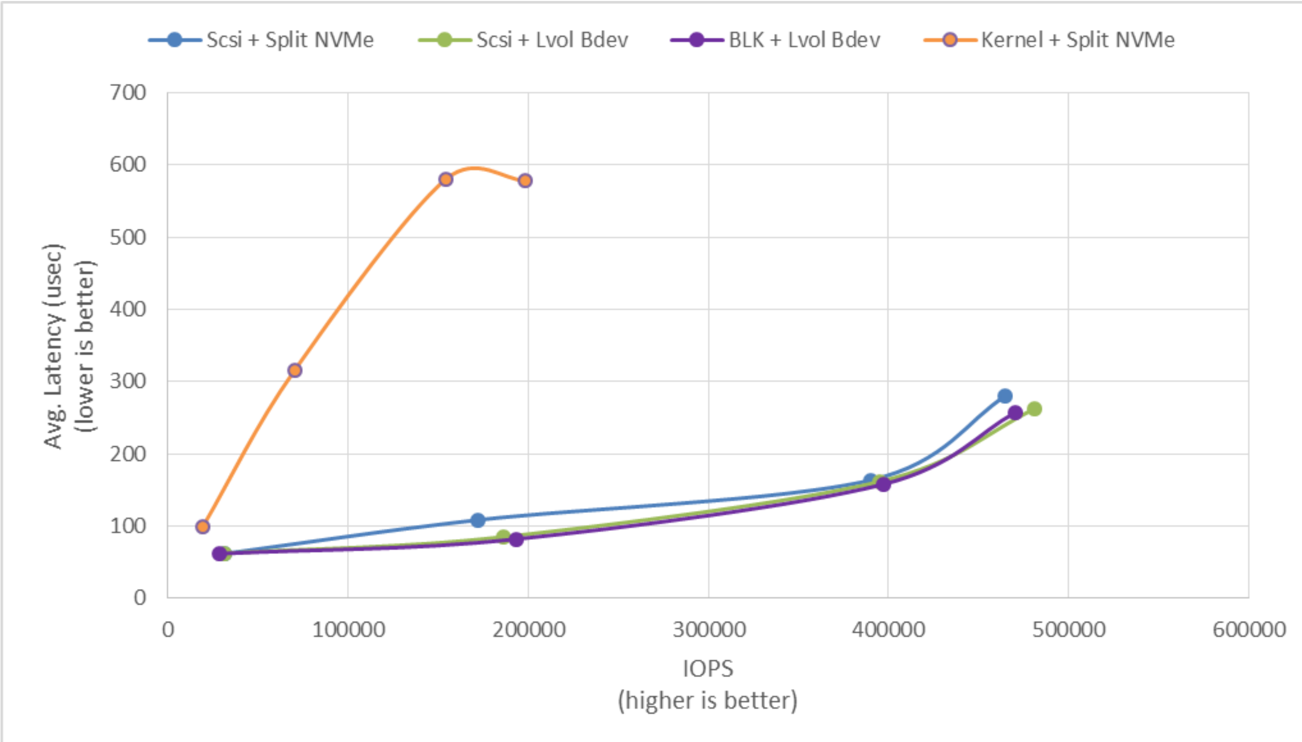


Figure 11: 4k 70%/30% Random Read/Write IOPS and latency

Conclusions

1. SPDK vhost-scsi with NVMe Split bdevs has lower latency and higher throughput than Kernel vhost-scsi in case of all run 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 underlying system when comparing kernel vs. SPDK vhost-scsi target implementations.

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

This report provides information regarding methodologies and practices while benchmarking vhost-scsi and vhost-blk using SPDK, as well as, 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.

DISCLAIMERS

§