Dell PowerFlex Performance Characteristics with Broadcom BCM57508 Ethernet Controllers

January 2025

H20229

White Paper

Abstract

This document describes the performance characteristics of Dell PowerFlex with Broadcom BCM57508 2x100GbE Ethernet Controllers.

DELL Technologies

Copyright

The information in this publication is provided as is. Dell Inc. makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any software described in this publication requires an applicable software license.

Copyright © 2025 Dell Inc. or its subsidiaries. Published in the USA January 2025 H20229.

Dell Inc. believes the information in this document is accurate as of its publication date. The information is subject to change without notice.

Contents

Executive summary 4

System architecture 5

Validation 10

Conclusion 46

References 47

Appendix A – Hardware overview 48

Appendix B – fio configuration 55

Executive summary

Overview

Modern infrastructures can benefit from advances in network speed. In this white paper we examine the benefits of using Broadcom BCM57508 Ethernet controllers as part of a Dell PowerFlex environment. Specifically, we explore the impact they make to the storage environment and how performance increases substantially when using Broadcom 100GbE network interface cards (NICs).

We start by providing an overview of the technology used in our design. We then look at the test results and explore how an organization can benefit from leveraging PowerFlex with 100GbE networking to obtain a significant boost in the overall performance of their environment.

The performance increase from the Broadcom NICs means that PowerFlex can deliver a substantial boost to the whole PowerFlex environment. This translates into increased throughput with fewer nodes required to achieve the same storage speeds which could potentially reduce the required storage footprint for organizations, depending on throughput and capacity requirements.

Revisions

Date	Part number/ revision	Description
January 2025	H20229	Initial release

We value your feedback

Dell Technologies and the authors of this document welcome your feedback on this document. Contact the Dell Technologies team by email.

Author: PowerFlex Engineering Technologists

Contributors: Brian Dean, Roy Laverty, Tony Foster, Gautam Dash

Note: For links to other documentation for this topic, see PowerFlex Info Hub

System architecture

PowerFlex testing architecture

For the performance characterization we used a two-tier PowerFlex (v4.5.1.103) architecture with a medium granularity (MG) data layout. There is no discernible difference in the performance of the MG data layout compared with fine granularity (FG) versions, and the performance characteristics observed in this paper should be consistent with other versions of PowerFlex in both the 3.6 and 4.x series.

The architecture used in this white paper is not designed to validate the platforms that run on PowerFlex, or to validate a set of best practices. Our intention is to operate the system at its limit and control the variables that can impede that storage performance, thereby demonstrating the capabilities of an individual component.

Actual results for workloads may vary from the testing in this paper, and it is important that organizational outcomes are considered and applied as necessary to obtain both the correct level of performance and capabilities needed.

Our testing consisted of both virtualized and bare-metal environments to align with commonly deployed operating systems and hypervisors.

VMware by Broadcom virtual environment

The environment consisted of either one or four compute only (CO) nodes with VMware ESXi installed and four PowerFlex storage-only (SO) nodes. The VMware ESXi host was configured with build 8.0.1 – 21495797. The vCenter used to manage the environment was deployed in a separate management cluster so as not to impact testing.

All testing virtual machines (VMs) were installed on local storage to avoid impacting test performance. Under normal operating conditions, it is not advisable to run production workloads on local storage due to its associated limitations, for example VMware High Availability (HA) events.

Each CO node had the PowerFlex Storage Data Client (SDC) software installed and a single VM for testing. The testing VM was a SUSE Linux VM with the following configuration:

Table 1. VMware Virtual Machine Configuration

Resource	Count
vCPUs	8
Test Volumes	320GB (x3)
Memory Reservation	100%
CPU Reservation	50%

For testing of the VMware virtual environment, the architecture used is represented in the following diagram:

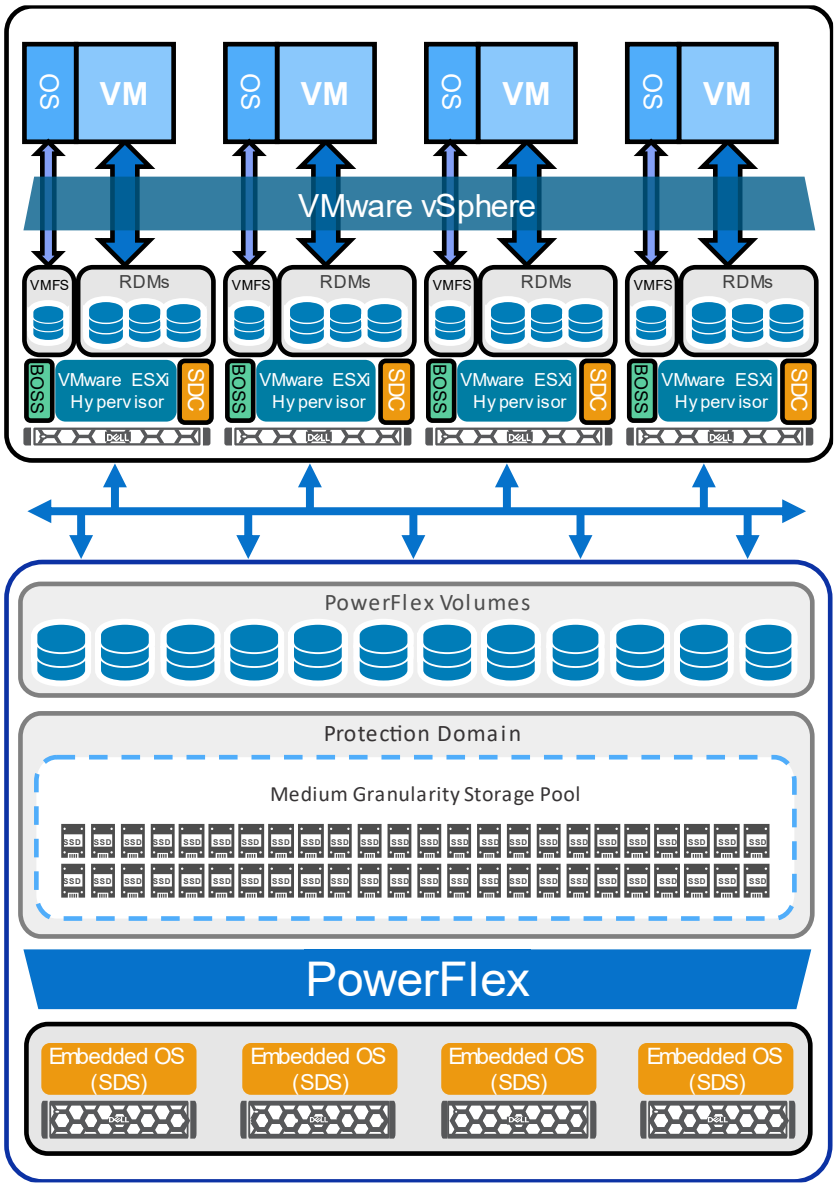


Figure 1. Logical architecture of VMware test environment

Bare-metal environment

For bare-metal compute, the following Linux operating systems (OS) were tested with both four compute nodes and a single PowerFlex compute node that were backed by four PowerFlex storage nodes.

- Red Hat Enterprise Linux (RHEL) 9.2 (kernel 5.14.0-284.30.1)
- SUSE Enterprise Linux 15.4 (kernel 5.14.21-24.111.2)

The architecture used is similar to the virtual environment discussed previously and is shown in the following diagram. The significant difference is there is no hypervisor, or virtual machines present in this design.

The bare-metal hosts each have three 320GB volumes for testing and an OS volume.

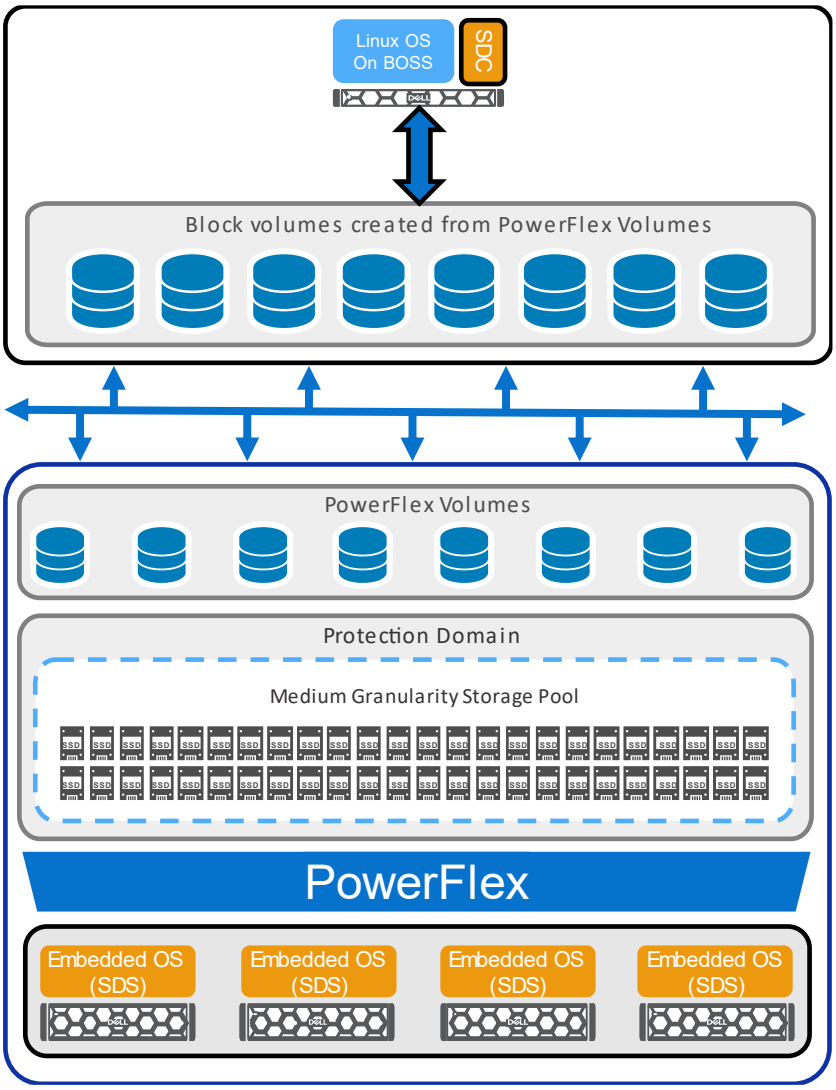


Figure 2. Logical architecture of bare metal test environment

Node specifications

The hardware specifications used for bare-metal and VMware virtual environments are identical and listed in the following table:

Table 2. Node specifications

Node Type	CPU	vCPUs	Storage
Storage (SDS)	Intel® Xeon® Gold 6426Y 2.5Ghz(3.3Ghz)	64	12x12TiB KIOXIA NVMe
Compute (SDC)	Intel® Xeon® Gold 5420+ 2Ghz(2.7Ghz)	112	N/A

The following table describes the SDC performance settings used for all tests:

Table 3. SDC performance settings

Setting	Value
Flow Control	Disabled
SDC_MAX_INFLIGHT_REQUESTS	300
SDC_MAX_INFLIGHT_DATA	20
No. of SDC threads	10

The Ethernet controllers used for the tests are detailed in the following table. The Ethernet controllers are identical for both storage and compute nodes. The controllers were changed from the 100GbE BCM57508 to the 25GbE BCM57414 between tests. The network interfaces used for PowerFlex data traffic were configured with an MTU of 9000.

Table 4. Ethernet controllers

Test Case	Ethernet Controller	Cards	Port Speed (# ports)
25Gbps	Broadcom BCM57414	4	25Gbps (2)
100Gbps	Broadcom BCM57508	2	100Gbps (2)

Network architecture

Beyond the Ethernet controllers the network infrastructure (including switches and cables) plays an important role in the performance of any storage environment. In this section we look at the PowerFlex networking architecture and its configuration.

Physical network architecture

The physical network for the test environment is composed of two Dell S5232F-ON 100GbE top of rack (TOR) switches and a separate management switch dedicated to management traffic, including iDRAC connectivity.

When testing the BCM57508 Ethernet 100GbE adapters we used 100GbE QSFP28 cables. For testing of the BMC57414 Ethernet adapters we used QSFP breakout cables connected to the Dell S5232F-ON switches. This reduced change to the test environment between tests and ensured switch consistency across tests.

Logical network architecture

The logical network architecture is shown in the following diagram.

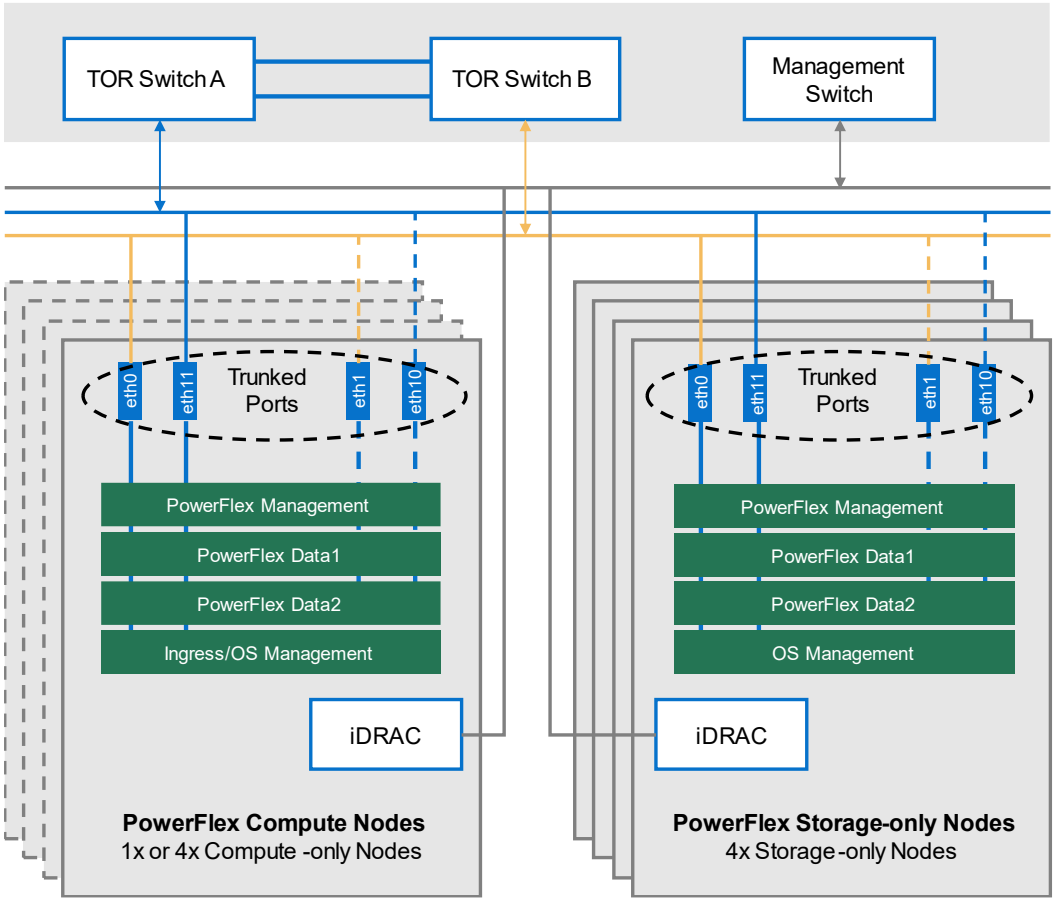


Figure 3. Logical network architecture

At the physical layer, two supported TOR switches are used for redundancy and load-balancing purposes. All network ports are configured as trunks carrying the necessary VLANs. VLANs are created to separate different network traffic. No QoS settings were configured for the different VLANs.

VLAN configuration

PowerFlex uses several different VLANs for communication and traffic isolation. The following table shows the different VLAN networks that were configured for the testing environment:

Table 5. VLANs configuration

Network Type	VLAN ID
Ingress/API/OS Management	105
PowerFlex Management	150
PowerFlex Data 1	151
PowerFlex Data 2	152

Virtual networking for the VMware ESXi Host

For the VMware virtual environment, additional virtual network configuration was necessary. We used a VMware vSwitch. The trunked network connections for the

PowerFlex Data were added to the standard vSwitch and the VLANs were configured accordingly as port groups on the standard vSwitch. The standard vSwitch was configured with an MTU of 9000 to support jumbo frames.

Validation

Test methodology

Flexible IO tester

Flexible I/O tester (fio) (https://fio.readthedocs.io/) is an open-source, benchmarking tool, capable of simulating block device I/O patterns observed in real-world applications. For all tests conducted inside of VMs or on bare-metal nodes, fio versions 3.35 and 3.36 were used.

Test deployment

The PowerFlex SDC and fio tester were installed on both the Linux CO nodes and VMware ESXi CO nodes. For bare-metal testing, specified workloads were executed on raw PowerFlex volumes provisioned to the Linux CO nodes without file systems. Additionally, for VMware ESXi testing, the fio tester was installed in a Linux VM located on datastores backed by PowerFlex volumes

The full test suite was run using four 25GbE Ethernet adapters followed by two – 100GbE Ethernet adapters. For more details on how the fio command was configured for these tests refer to Appendix B – fio configuration.

Hypothesis

We wanted to test the difference between 25GbE and 100GbE Ethernet adapters to observe what impacts it had on throughput and where network bottlenecks were located. In the test configurations used, we wanted to examine if it was possible to fully saturate the network links in the various configurations. The expectation being that by using to 100GbE Ethernet adapters there would be more than sufficient bandwidth to support the demands of the tested operating systems.

Block volume testing - VMware 8.0.1 Testing

This section explores the testing results for a VMware ESXi host configured as a PowerFlex CO node using PowerFlex storage.

Testing observations – VMware using the 25GBbE Ethernet adapter

We began testing using the VMware virtual environment described in the System architecture section with the **25GbE Ethernet adapter**. We used four network ports for the test. In PowerFlex, we mapped three 320GB volumes to the SDCs installed in four ESXi nodes, created, and mounted as datastores. We then created four SLES 15.4 VMs on the PowerFlex backed datastores.

The VMware testing was conducted inside the SLES VMs, using the fio tool. The SUSE Linux Enterprise Server (SLES) VMs were then configured using the fio tool guidance. We then issued the fio command from a console to run a 20% random write test. The configuration of the fio command is detailed in Appendix B – fio configuration for all the test states.

The results for the 20% random write performance using the 25GbE Ethernet adapter are shown in the following table and chart.

Table 6. ESXi 8.0.1 – 20% Random Write IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

Read				Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	175	1,149	294,225	160	287	73,595	172	1,437	367,820
8k	182	2,221	284,270	174	555	71,065	180	2,776	355,335
16k	193	4,207	269,228	209	1,051	67,274	196	5,258	336,502
32k	209	7,650	244,793	249	1,914	61,260	217	9,564	306,052
64k	245	11,737	187,791	292	2,936	46,983	254	14,673	234,774
128k	290	11,772	94,172	373	2,942	23,540	306	14,714	117,712
256k	371	11,782	47,128	539	2,952	11,807	405	14,734	58,935
512k	545	11,712	23,424	853	2,939	5,877	606	14,651	29,302

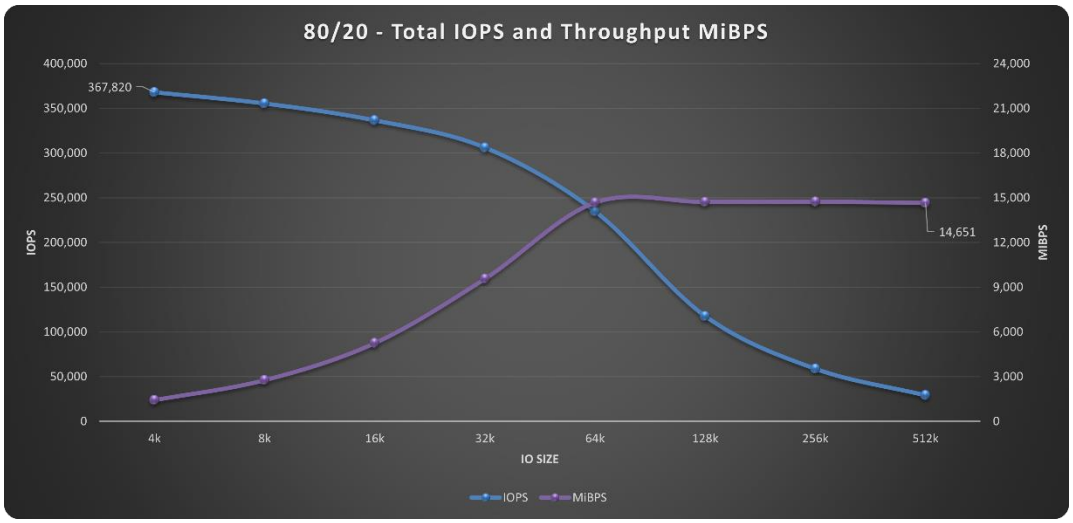


Figure 4. ESXi 8.0.1 - 20% Random Write IOPS and Throughput (MiBPS)

We can see that a single SLES 15.4 VM hosted on an ESXi 8.0.1 node can attain 367,820 IOPS with a 4k block size from the PowerFlex environment. Additionally, we can see IOPS for the other block sizes tested.

We completed tests for 50% and 80%reads and writes with fio on the SLES 15.4 VM running on an ESXi 8.0.1 node.

The results for each of these is shown in the following tables and charts:

Table 7. ESXi 8.0.1 – 50% Random Write IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

Read				Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	178	723	185,088	159	723	185,127	168	1,446	370,215
8k	185	1,372	175,556	173	1,371	175,432	179	2,742	350,989
16k	191	2,555	163,499	204	2,554	163,464	198	5,109	326,962
32k	210	4,727	151,271	244	4,727	151,277	227	9,455	302,547
64k	247	8,000	127,993	282	8,005	128,087	265	16,005	256,081
128k	294	10,026	80,205	361	10,021	80,165	327	20,046	160,371
256k	379	10,560	42,241	523	10,562	42,248	451	21,122	84,489
512k	553	8,058	16,115	834	8,072	16,143	693	16,129	32,258

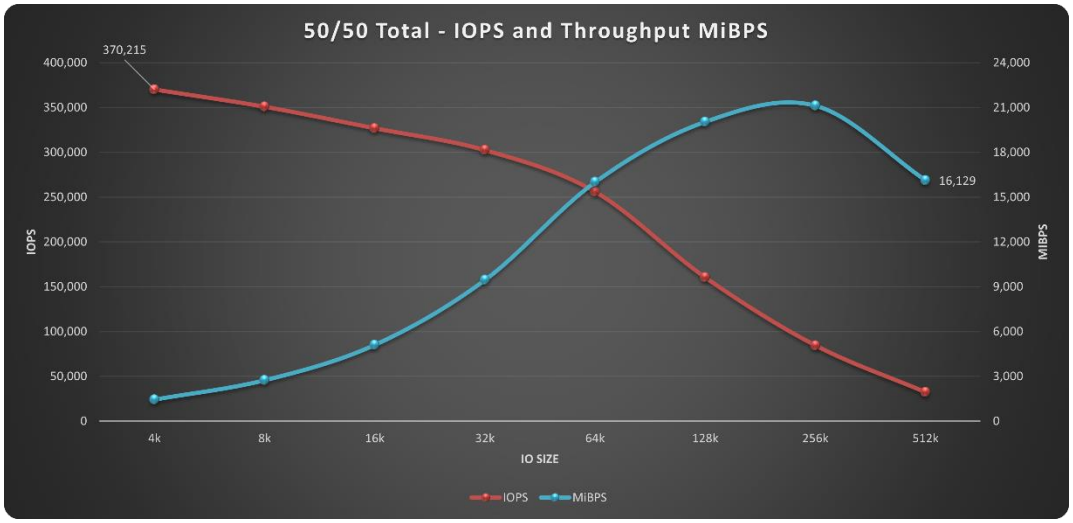


Figure 5. ESXi 8.0.1 - 50% Random Write IOPS and Throughput (MiBPS)

For the 50/50 random workload each SLES VM on an ESXi 8.0.1 node can attain 370,215 IOPS with a 4k block size from the PowerFlex environment.

Table 8. ESXi 8.0.1 – 80% Random Write IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	178	286	73,175	160	1,144	292,748	163	1,429	365,923
8k	187	550	70,362	174	2,198	281,293	177	2,747	351,654
16k	193	1,011	64,725	206	4,047	259,017	203	5,058	323,741
32k	213	1,860	59,529	246	7,443	238,168	239	9,303	297,697
64k	246	2,813	45,008	283	11,248	179,972	276	14,061	224,980
128k	287	2,935	23,479	359	11,732	93,853	344	14,663	117,308
256k	374	2,937	11,750	529	11,742	46,969	498	14,680	58,719
512k	549	2,941	5,883	838	11,744	23,489	781	14,685	29,369

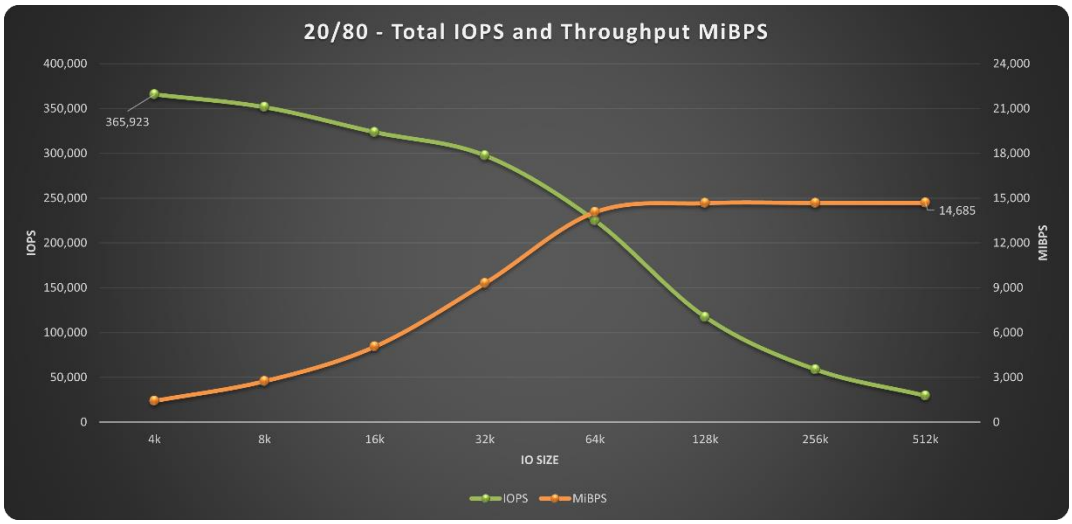


Figure 6. ESXi 8.0.1 - 80% Random Write IOPS and Throughput (MiBPS)

For the 20/80 random workload each SLES VM on an ESXi 8.0.1 node can attain 365,923 IOPS with a 4k block size from the PowerFlex environment.

Table 9. 64k Sequential Reads – IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

Total IO Depth	Read Throughput (MiB/s)	IOPS	Latency (µs)
1	269	4,311	232
4	1,019	16,300	245
8	1,927	30,829	259
16	2,516	40,262	397
32	2,932	46,911	682
64	3,621	57,942	1,104
128	6,078	97,245	1,316
256	9,392	150,265	1,703
512	9,960	159,364	3,212
1024	8,108	129,733	7,891

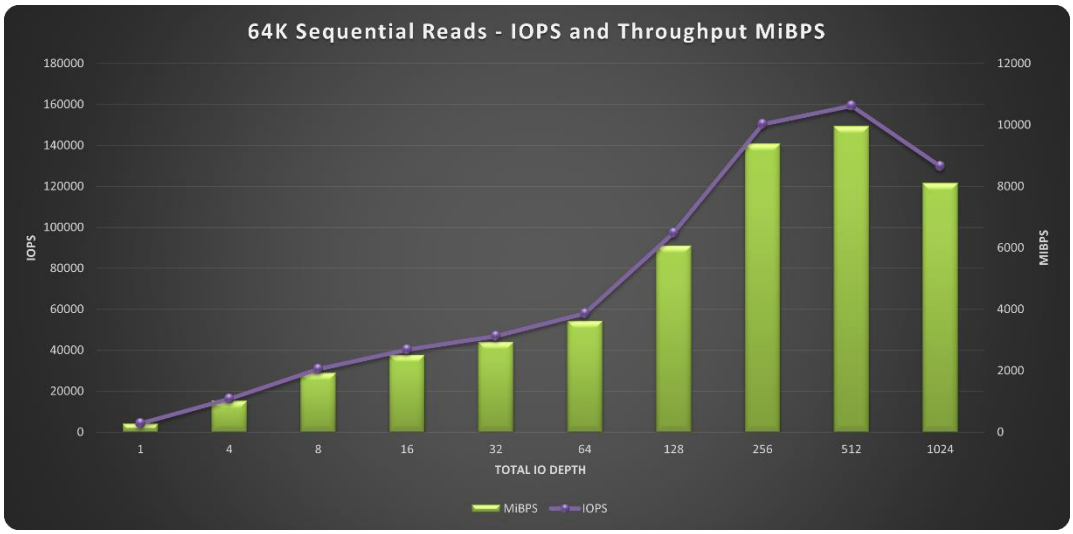


Figure 7. ESXi 8.0.1 - 64k Sequential Read IOPS and Throughput (MiBPS)

For the 64k sequential read workload we ran the fio jobs on four SLES 15.4 VMs hosted on four ESXi 8.0.1 nodes simultaneously. In this test, we found that each SDC can attain 129,733 IOPS. Each SDC can also attain 2,932 MiB/s throughput with less than 1 millisecond response time.

Table 10. 64k Sequential Writes – IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

Total IO Depth	Write Throughput (MiB/s)	IOPS	Latency (µs)
1	296	3,259	204
4	339	11,404	713
8	377	12,850	803
16	1,234	20,561	1,285
32	1,402	22,631	1,414
64	1,586	40,056	2,503
128	2,321	54,857	3,429
256	3,922	65,076	4,067
512	7,234	70,651	4,416
1024	15,824	64,645	4,040

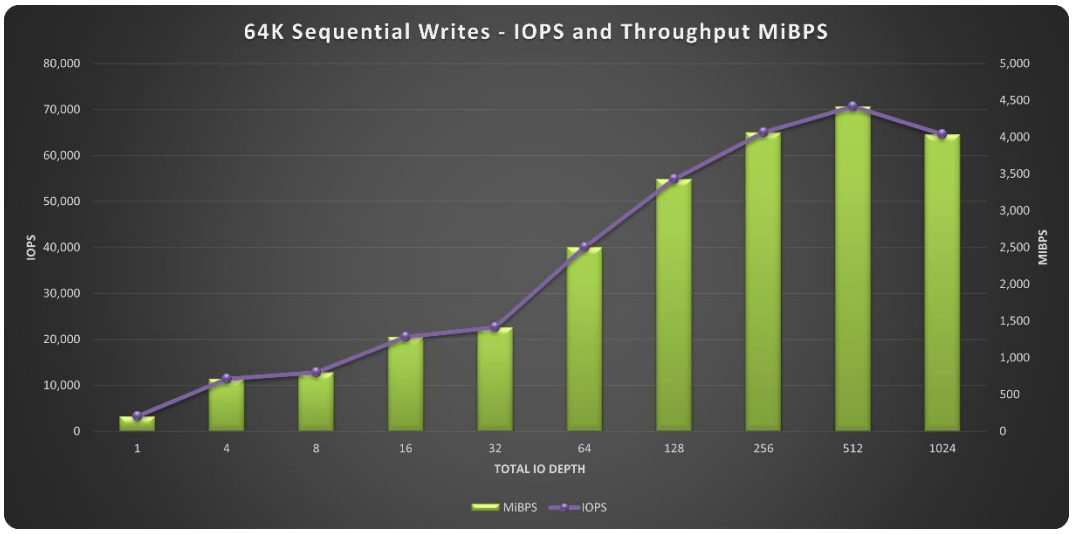


Figure 8. ESXi 8.0.1 - 64k Sequential Writes IOPS and Throughput (MiBPS)

For the 64k sequential write workload we ran the fio jobs on four SLES 15.4 VMs hosted on four ESXi 8.0.1 nodes simultaneously. In this test, we found that each SDC can attain 64,645 IOPS. Each SDC can also attain 377 MiB/s throughput with less than 1 millisecond response time.

Note: All testing with the 25GbE Ethernet adapter detailed in this paper was performed prior to switching to the 100GbE adapter. This was done to reduce number of changes from one adapter to the other. Ethernet

Testing observations – VMware using the 100GBbps Ethernet Adapter

Following completion of the 25GbE Ethernet adapter testing, the environment was shut down and reconfigured with the 100GbE Ethernet adapters. This included the four PowerFlex SO nodes and the VMware ESXi CO node.

We began testing with the VMware virtual environment described in the System architecture section with the **100GbE Ethernet adapter**. We used four network ports for the test. In PowerFlex we mapped three-320GB volumes to the SDCs installed in four ESXi nodes, created, and mounted as datastores

The results for the 20% random write performance using the 100GbE Ethernet adapter are shown in the following table and chart.

Table 11. ESXi 8.0.1 – 20% Random Write IOPS and Throughput (MiBPS) Over Two - 100GbE Ethernet Adapters

Read				Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	173	1,198	306,736	160	300	76,681	170	1,498	383,417
8k	178	2,314	296,218	168	578	73,995	176	2,892	370,213
16k	186	4,409	282,167	179	1,102	70,515	184	5,511	352,682
32k	194	8,242	263,757	202	2,059	65,888	196	10,301	329,646
64k	228	13,680	218,882	265	3,416	54,655	236	17,096	273,537
128k	259	20,320	162,557	323	5,087	40,696	272	25,407	203,254
256k	327	23,203	92,813	447	5,802	23,209	351	29,006	116,022
512k	476	23,287	46,573	721	5,820	11,641	525	29,107	58,214

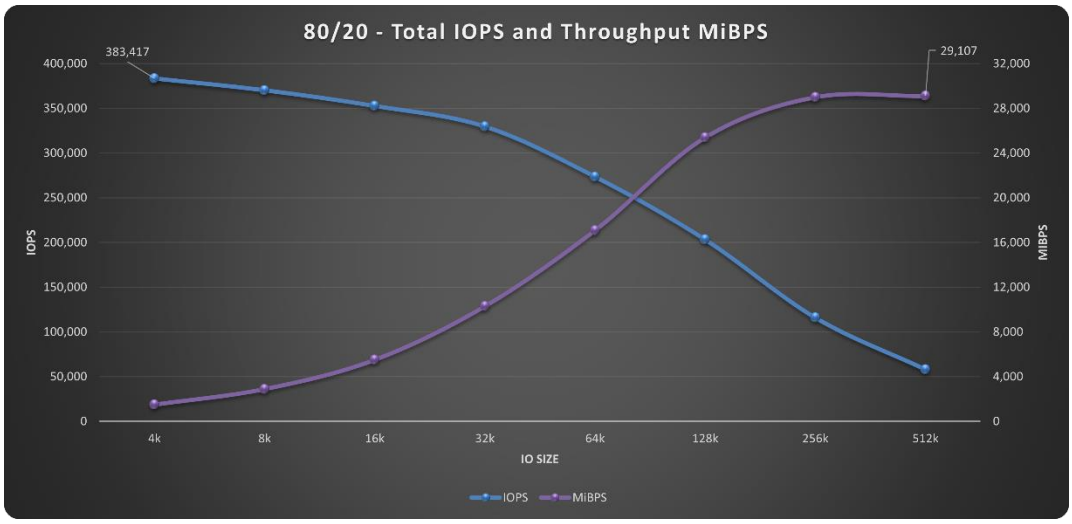


Figure 9. ESXi 8.0.1 - 20% Random Write IOPS and Throughput (MiBPS)

We can see that a single SLES 15.4 VM hosted on an ESXi 8.0.1 node can attain 383,417 IOPS with a 4k block size from the PowerFlex environment. Additionally, we can see IOPS for the other block sizes tested.

We show the tests for the 50% and 80% reads and writes with fio on the SLES 15.4 VM running on an ESXi 8.0.1 node.

The results for each of these is shown in the following tables and charts:

Table 12. ESXi 8.0.1 – 50% Random Write IOPS and Throughput (MiBPS) Over Two - 100GbE Ethernet Adapters

Read				Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	174	741	189,759	158	741	189,778	166	1,483	379,536
8k	179	1,443	184,719	165	1,443	184,748	172	2,886	369,467
16k	183	2,733	174,893	175	2,732	174,861	179	5,465	349,754
32k	198	4,998	159,945	200	5,000	160,009	199	9,999	319,953
64k	233	8,425	134,807	263	8,429	134,856	248	16,854	269,663
128k	265	11,669	93,352	315	11,678	93,420	290	23,346	186,772
256k	338	12,850	51,399	437	12,836	51,345	388	25,686	102,743
512k	492	14,311	28,623	706	14,306	28,611	598	28,617	57,234

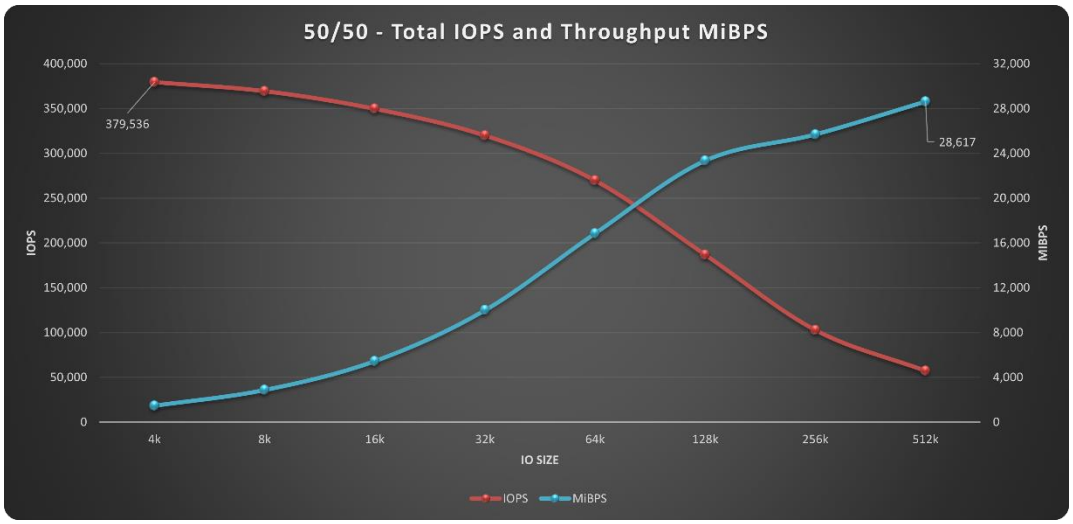


Figure 10. ESXi 8.0.1 - 50% Random Write IOPS and Throughput (MiBPS)

We can see that a single SLES 15.4 VM hosted on an ESXi 8.0.1 node can attain 379,536 IOPS with a 4k block size from the PowerFlex environment. Additionally, we can see IOPS for the other block sizes tested.

Table 13. ESXi 8.0.1 – 80% Random Write IOPS and Throughput (MiBPS) Over Two - 100GbE Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	176	299	76,480	158	1,195	306,025	161	1,494	382,505
8k	186	553	70,764	164	2,211	283,006	168	2,764	353,770
16k	185	1,101	70,492	174	4,407	282,073	177	5,509	352,566
32k	196	2,046	65,488	198	8,182	261,817	197	10,228	327,305
64k	232	2,937	46,991	259	11,753	188,047	254	14,690	235,038
128k	263	3,603	28,823	307	14,420	115,361	298	18,023	144,184
256k	332	3,755	15,020	430	15,022	60,088	410	18,777	75,108
512k	474	4,084	8,169	684	16,347	32,694	642	20,431	40,863

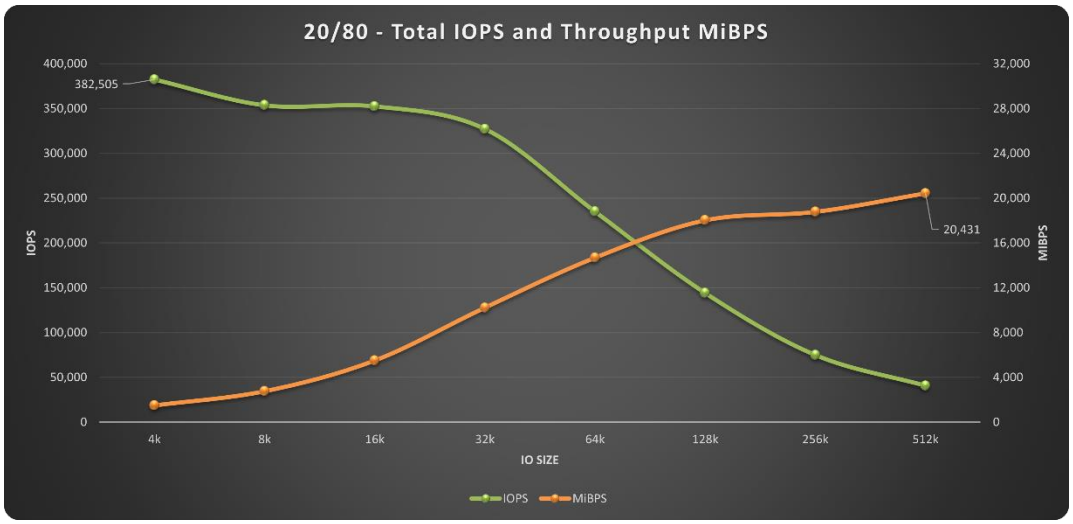


Figure 11. ESXi 8.0.1 - 80% Random Write IOPS and Throughput (MiBPS)

We can see that a single SLES 15.4 VM hosted on an ESXi 8.0.1 node can attain 382,505 IOPS with a 4k block size from the PowerFlex environment.

Table 14. 64k Sequential Reads – IOPS and Throughput (MiBPS) Over Four - 100GbE Ethernet Adapters

Total IO Depth	Read Throughput (MiB/s)	IOPS	Latency (µs)
1	302	4,839	206
4	1,159	18,550	215
8	2,176	34,810	229
16	3,347	53,553	298
32	5,185	82,954	385
64	5,868	93,883	681
128	7,247	115,958	1,104
256	13,637	218,184	1,173
512	14,316	229,060	2,235
1024	14,144	226,299	4,524

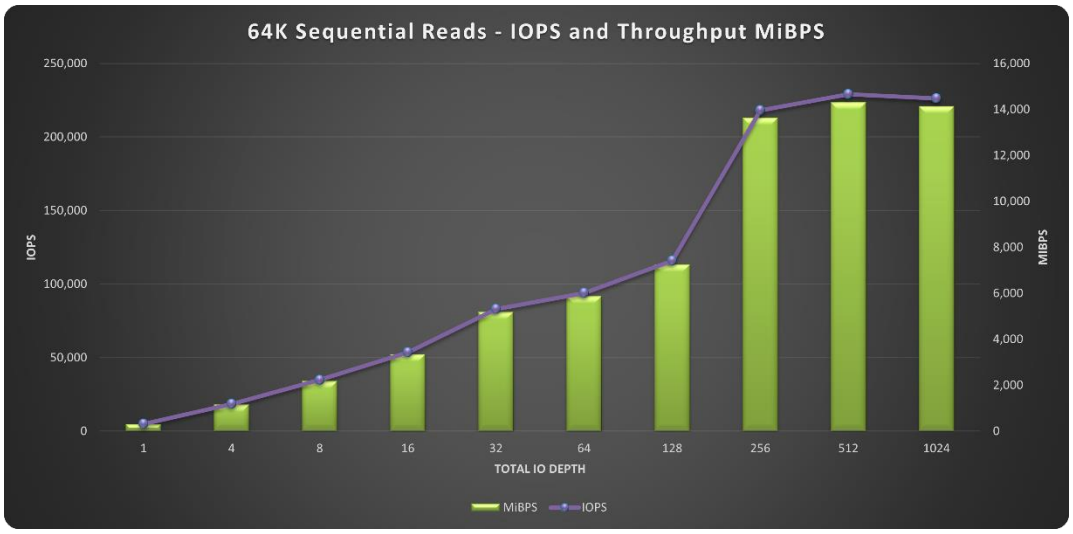


Figure 12. ESXi 8.0.1 - 64k Sequential Reads IOPS and Throughput (MiBPS)

For the 64k sequential read workload, we ran the fio jobs on four SLES 15.4 VMs hosted on four ESXi 8.0.1 nodes simultaneously. In this test, we found that each SDC can attain 226,299 IOPS. Each SDC can also attain 5,868 MiB/s throughput with less than 1 millisecond response time.

Table 15. 64k Sequential Writes – IOPS and Throughput (MiBPS) Over Four - 100GbE Ethernet Adapters

Total IO Depth	Write Throughput (MiB/s)	IOPS	Latency (µs)
1	229	3,661	262
4	817	13,078	294
8	1,424	22,781	340
16	2,205	35,284	441
32	3,172	50,746	619
64	3,334	53,350	1,188
128	4,527	72,428	1,756
256	4,911	78,583	3,247
512	5,201	83,214	6,141
1024	5,304	84,857	12,049

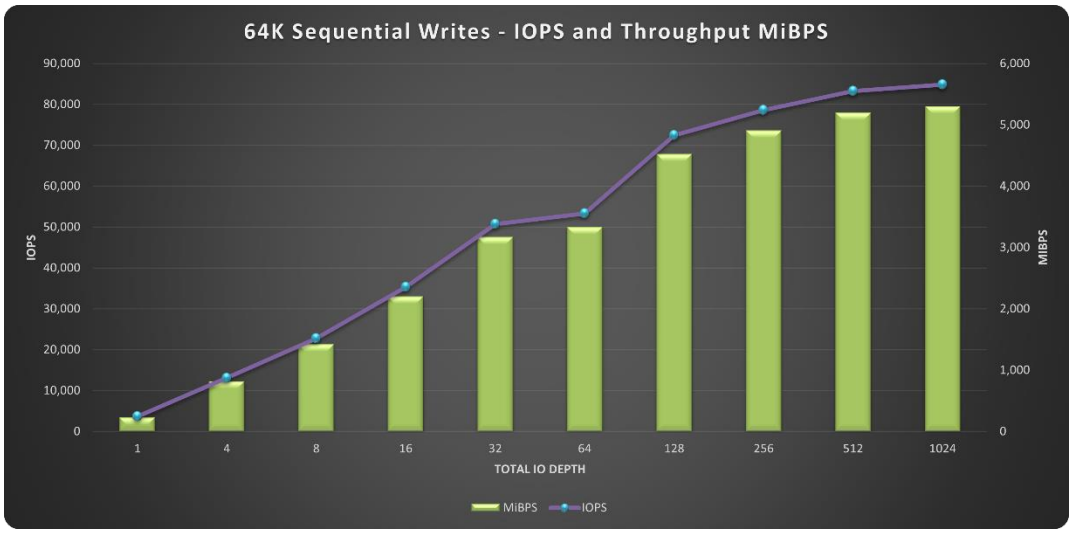


Figure 13. ESXi 8.0.1 - 64k Sequential Writes IOPS and Throughput (MiBPS)

For the 64k sequential write workload, we ran the fio jobs on four SLES 15.4 VMs hosted on four ESXi 8.0.1 nodes simultaneously. In this test, we found that each SDC can attain 84,857 IOPS. Each SDC can also attain 3,172 MiB/s throughput with less than 1 millisecond response time.

Evaluation of results for VMware ESXi volumes

We compiled the results in the following charts to show the comparison of the 25GbE and 100GbE Ethernet adapters. The following charts compare the throughput results of four 25GbE Ethernet adapters, two 100GbE Ethernet adapters, and four 100GbE Ethernet adapters.

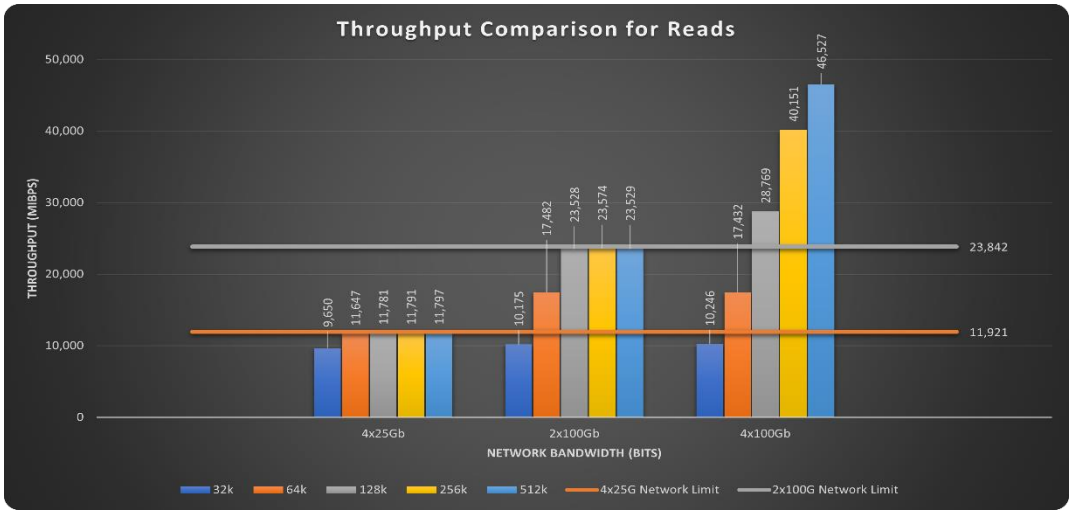


Figure 14. ESXi 8.0.1 – Read Throughput Comparison – Four 25GbE vs. Two 100GbE vs. Four 100GbE Ethernet adapters

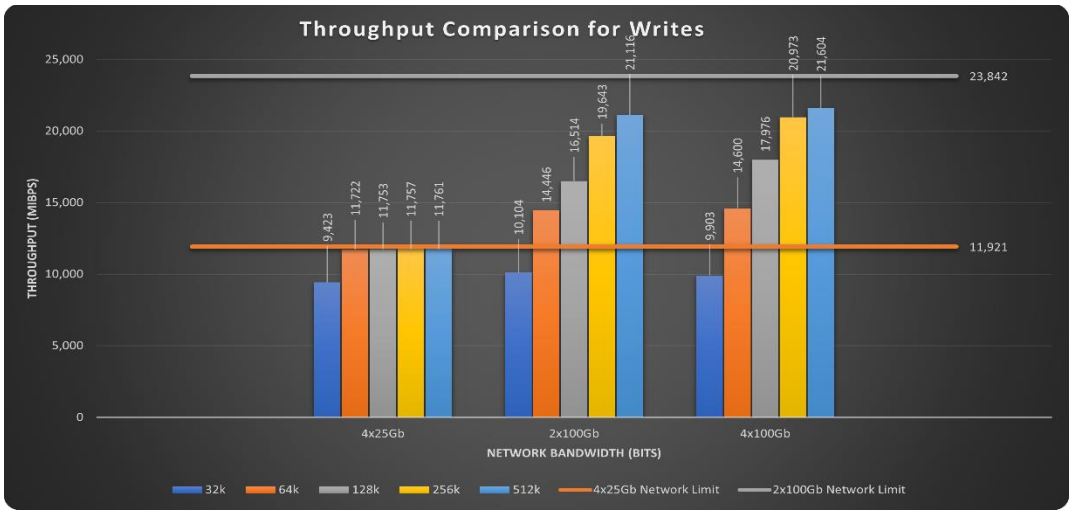


Figure 15. ESXi 8.0.1 – Write Throughput Comparison – Four 25GbE vs. Two 100GbE vs. Four 100GbE Ethernet adapters

The test results show that two 100GbE Ethernet adapters provide an almost 100% improvement in both 256k read throughput and 512k read throughput compared to four 25GbE Ethernet adapters. For write performance, two 100GbE Ethernet adapters provide a 67% improvement in 256k write throughput and an 80% improvement in 512k write throughput.

This shows an increase in storage performance and means that it is beneficial for VMware ESXi environments to use 100GbE Ethernet adapters with PowerFlex storage. This benefit can be seen across the five different test scenarios in this section.

This also indicates that the optimal network configuration for a SUSE Enterprise Linux environment using PowerFlex as the underlying storage platform is to leverage 100GbE Ethernet adapters to obtain increased storage throughput when compared with 25GbE Ethernet adapters.

Block volume testing – Red Hat Enterprise Linux 9.2

This section examines the testing results for a RHEL 9.2 host configured as a PowerFlex CO node using PowerFlex storage.

Testing observations – Red Hat Enterprise Linux using the 25GBbps Ethernet Adapter

We tested with the bare-metal environment described in the System architecture section using four 25GbE Ethernet adapters. The fio tester was installed during the initial setup of the node.

In the PowerFlex Manager UI we created three volumes of 320GB in size and presented them to four Red Hat hosts.

The environment was then configured using the fio tool guidance. We issued the fio command from a console to run a 20% random write test. The configuration of the fio command is detailed in Appendix B – fio configuration for all test states.

The results for the 20% random write performance using the 25GbE Ethernet adapter are shown in the following table and chart:

Table 16. RHEL 9.2 – 20% Random Write IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	147	1,955	500,429	131	489	125,085	143	2,443	625,514
8k	153	3,648	466,934	142	913	116,813	151	4,561	583,747
16k	169	6,763	432,834	174	1,692	108,259	170	8,455	541,093
64k	210	11,751	188,021	236	2,939	47,030	215	14,691	235,051
256k	345	11,786	47,143	503	2,948	11,793	377	14,734	58,936

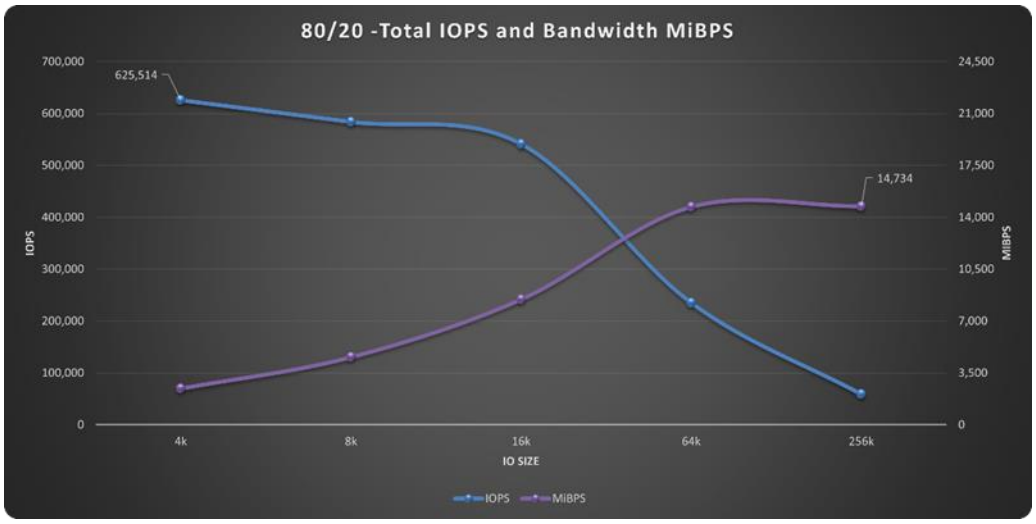


Figure 16. RHEL 9.2 – 20% Random Write IOPS and Throughput (MiBPS)

We can see that a single SDC in a RHEL 9.2 Linux node can attain 625,514 IOPS with a 4k block size from the PowerFlex environment. Additionally, we can see IOPS for the other block sizes tested.

We show the tests for the 50% and 80% reads and writes with fio on the RHEL host.

Table 17. RHEL 9.2 – 50% Random Write IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	149	1,169	299,136	131	1,169	299,235	140	2,337	598,372
8k	155	2,139	273,827	142	2,139	273,760	148	4,278	547,587
16k	171	3,493	223,573	172	3,494	223,626	172	6,987	447,199
64k	216	9,078	145,245	237	9,076	145,209	226	18,153	290,454
256k	362	10,779	43,115	501	10,798	43,192	432	21,570	86,279

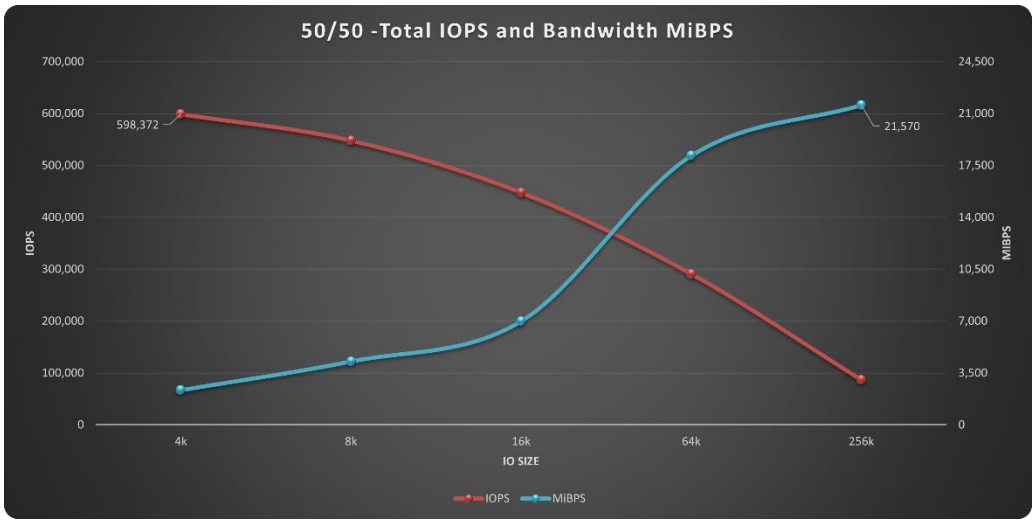


Figure 17. RHEL 9.2 – 50% Random Write IOPS and Throughput (MiBPS)

The results for each of these is shown in the following tables and charts:

For the 50/50 random workload a single SDC in a RHEL 9.2 Linux node can attain 598,372 IOPS with a 4k block size from the PowerFlex environment.

Table 18. RHEL 9.2 – 80% Random Write IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

Read				Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	147	464	118,842	126	1,857	475,497	130	2,322	594,338
8k	156	807	103,297	142	3,231	413,591	144	4,038	516,888
16k	170	1,270	81,303	171	5,084	325,399	171	6,355	406,702
64k	212	2,934	46,940	235	11,745	187,918	231	14,679	234,857
256k	338	2,954	11,816	492	11,780	47,119	462	14,733	58,932

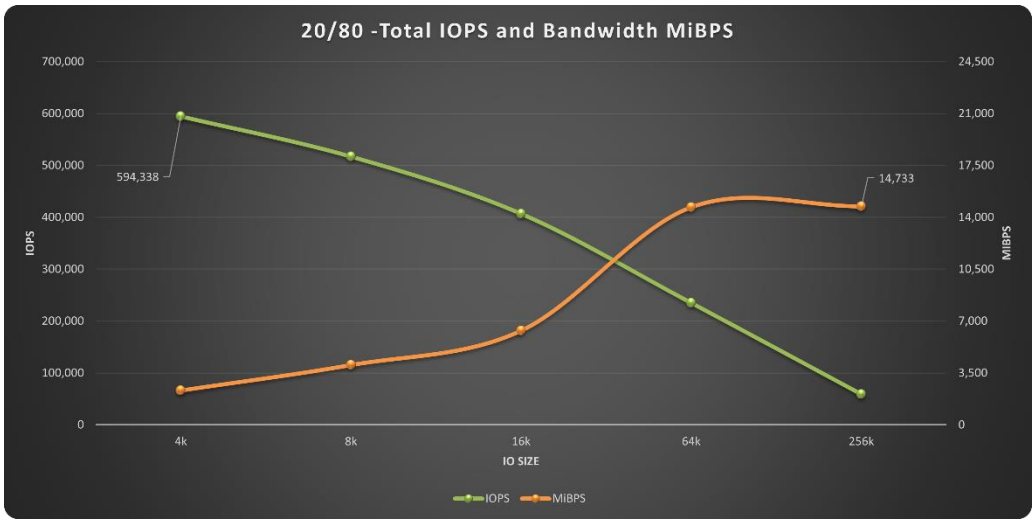


Figure 18. RHEL 9.2 – 80% Random Write IOPS and Throughput (MiBPS)

For the 20/80 workload, a single SDC in a RHEL 9.2 Linux node can attain 594,338 IOPS with a 4k block size from the PowerFlex environment.

Table 19. RHEL 9.2 – 64k Sequential Reads – IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

Total IO Depth	Read Throughput (MiB/s)	IOPS	Latency (µs)
1	310	4,953	202
4	1,082	17,310	231
8	1,990	31,837	251
16	3,574	57,191	279
32	6,077	97,225	329
64	8,568	137,085	467
128	9,918	158,690	806
256	10,776	172,411	1,484
512	11,347	181,557	2,820
1024	11,653	186,451	5,492



Figure 19. RHEL 9.2 – 64k Sequential Read IOPS and Throughput (MiBPS)

For the 64k sequential read workload, we ran the fio jobs on four RHEL 9.2 Linux nodes simultaneously. In this test, we found that each SDC can attain 186,451 IOPS. An SDC can also attain 9,918MiB/s throughput with less than 1 millisecond response time.

Table 20. RHEL 9.2 – 64k Sequential Writes - IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

Total IO Depth	Write Throughput (MiB/s)	IOPS	Latency (µs)
1	240	3,848	249
4	827	13,236	291
8	1,482	23,714	326
16	2,514	40,228	387
32	3,617	57,878	542
64	4,168	66,682	948
128	4,450	71,197	1,786
256	4,680	74,877	3,407
512	4,898	78,362	6,522
1024	5,018	80,292	12,741

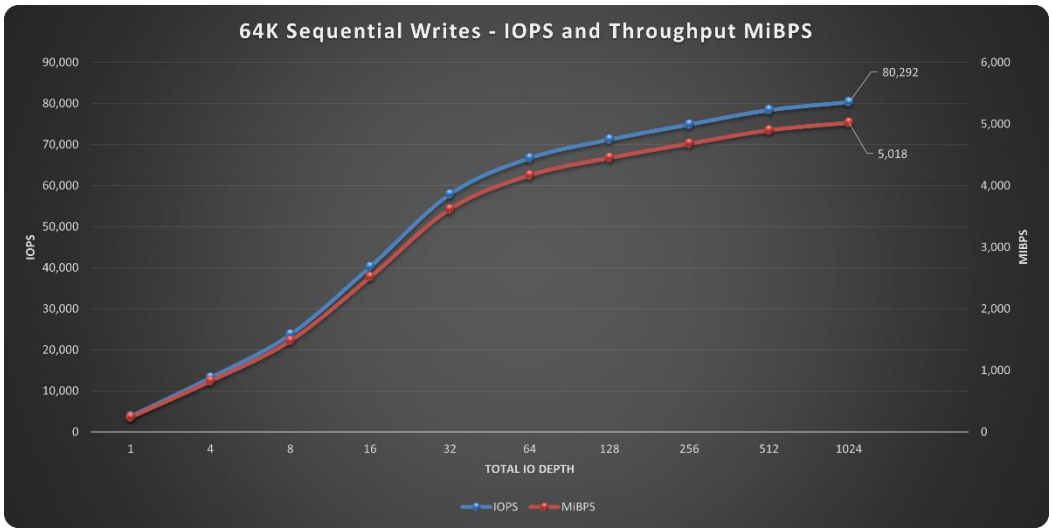


Figure 20. RHEL 9.2 – 64k Sequential Write IOPS and throughput (MiBPS)

For the 64k sequential write workload, we ran the fio jobs on four RHEL 9.2 Linux nodes simultaneously. In this test, we found that each SDC can attain 80,292 IOPS. An SDC can also attain 4,168 MiB/s throughput with less than 1 millisecond response time.

Note: All testing with the 25GbE Ethernet adapter detailed in this paper was performed prior to switching the 100GbE adapter. This was done to reduce number of changes from one adapter to the other.

Testing observations – Red Hat Enterprise Linux Volumes using the 100GBbps Ethernet Adapter

Following completion of the 25GbE Ethernet adapter testing, the environment was shut down and reconfigured with the 100GbE Ethernet adapters. This included the four PowerFlex SO nodes and the four Red Hat Enterprise Linux CO nodes.

Using the reconfigured Red Hat Enterprise Linux environment described in the System architecture section with two **100GbE Ethernet adapters**, fio was installed during the initial setup of the node. In the PowerFlex Manager UI we created 3 volumes of 320GB in size and presented them to the Red Hat host.

The environment was then configured using the fio tool guidance. We then used the same fio commands from the previous section to obtain results for the 100GbE Ethernet adapter.

The results for the 20% random write performance using the 100GbE Ethernet adapter are shown in the following table and chart.

Table 21. RHEL 9.2 - 20% Random Write IOPS and Throughput (MiBPS) Over Two - 100GbE Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	149	2,071	530,214	132	518	132,597	145	2,589	662,812
8k	151	3,874	495,868	137	968	123,921	149	4,842	619,789
256k	308	19,667	78,668	388	4,912	19,648	324	24,579	98,316

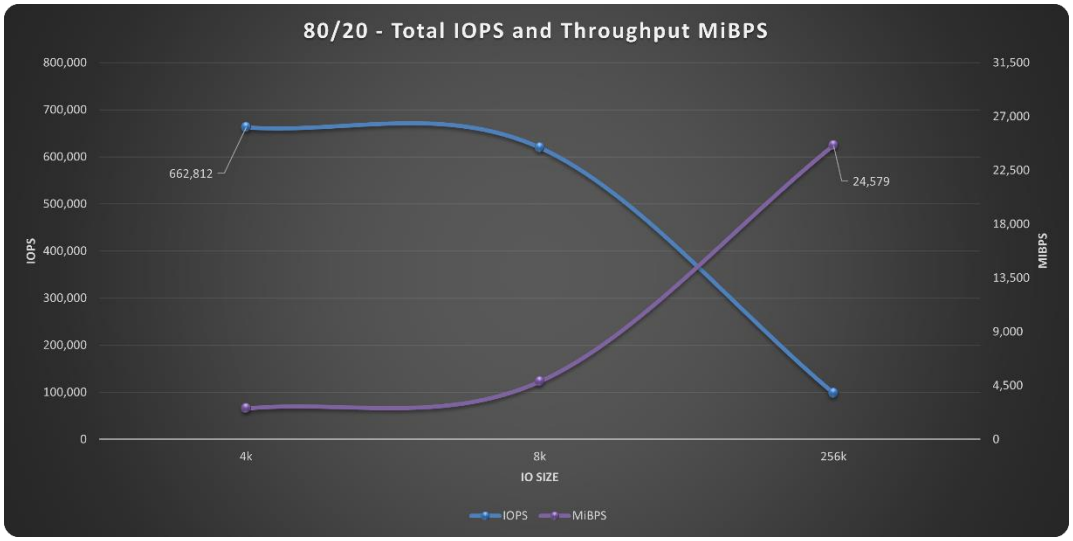


Figure 21. RHEL 9.2 – 20% Random Write IOPS and Throughput (MiBPS)

For the 20% write mixed workload we can see that a single SDC in a RHEL 9.2 Linux node can attain 662,812 IOPS with a 4k block size from the PowerFlex environment. Additionally, we can see IOPS for the other block sizes tested.

We then proceeded to run tests for 50% and 80% sequential reads and writes with fio.

The results for each of these is shown in the following tables and charts:

Table 22. RHEL 9.2 – 50% Random Write IOPS and Throughput (MiBPS) Over Two - 100Gbs Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	147	1,264	323,470	130	518	1,263	138	2,527	646,847
8k	150	2,287	292,727	135	968	2,287	142	4,574	585,467
256k	292	13,595	54,382	387	4,912	13,587	339	27,182	108,729

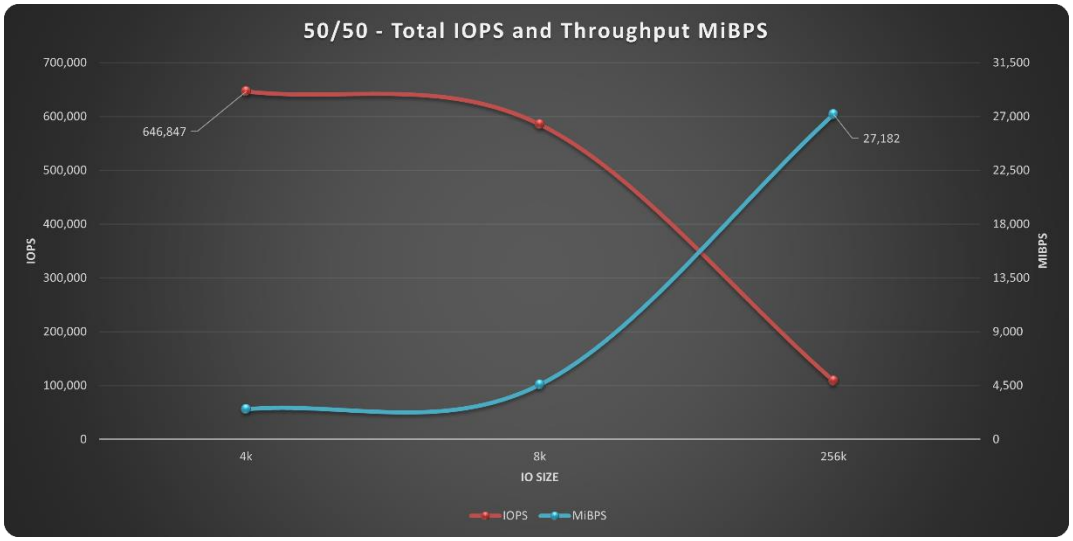


Figure 22. RHEL 9.2 – 50% Random Write IOPS and Throughput (MiBPS)

For the 50/50 mixed workload we can see that a single SDC in a RHEL 9.2 Linux node can attain 646,847 IOPS with a 4k block size from the PowerFlex environment.

Table 23. 80% Random Write IOPS and Throughput (MiBPS) Over Two - 100Gbs Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	148	516	132,113	130	2,063	528,054	133	2,579	660,168
8k	150	879	112,514	135	3,517	450,197	138	4,396	562,711
256k	283	5,307	21,229	380	21,226	84,904	361	26,533	106,133

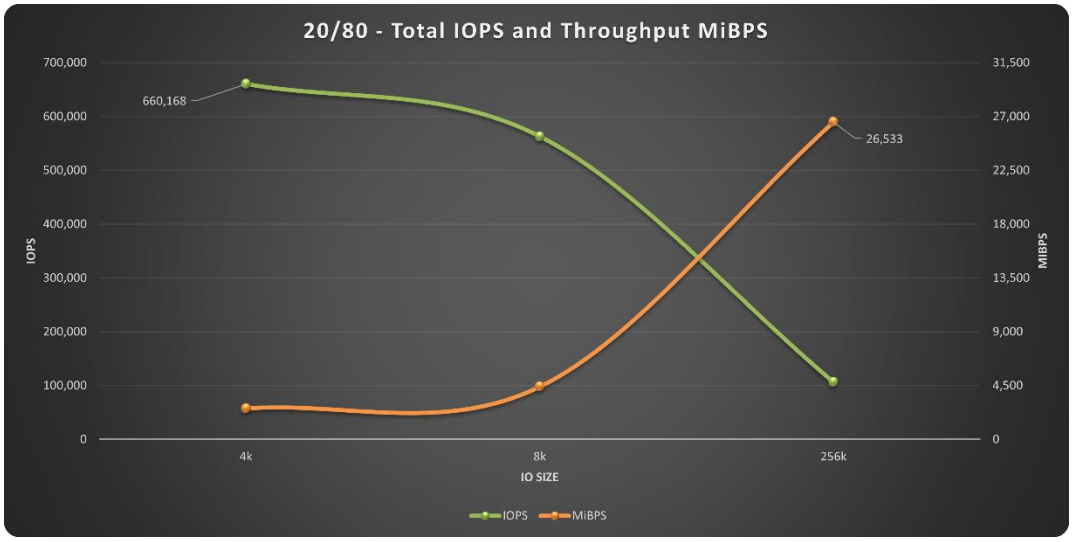


Figure 23. RHEL 9.2 – 80% Random Write IOPS and Throughput (MiBPS)

For the 20/80 write mixed workload we can see that a single SDC in a RHEL 9.2 Linux node can attain 660,168 IOPS with a 4k block size from the PowerFlex environment.

Table 24. RHEL 9.2 – 64k Sequential Reads – IOPS and Throughput (MiBPS) Over Four - 100Gbs Ethernet Adapters

Total IO Depth	Read Throughput (MiB/s)	IOPS	Latency (µs)
1	317	5,073	197
4	1,157	18,514	216
8	2,204	35,263	227
16	3,830	61,287	261
32	6,870	109,915	291
64	11,206	179,290	357
128	13,976	223,618	572
256	16,011	256,179	999
512	17,777	284,430	1,800
1024	19,230	307,679	3,328

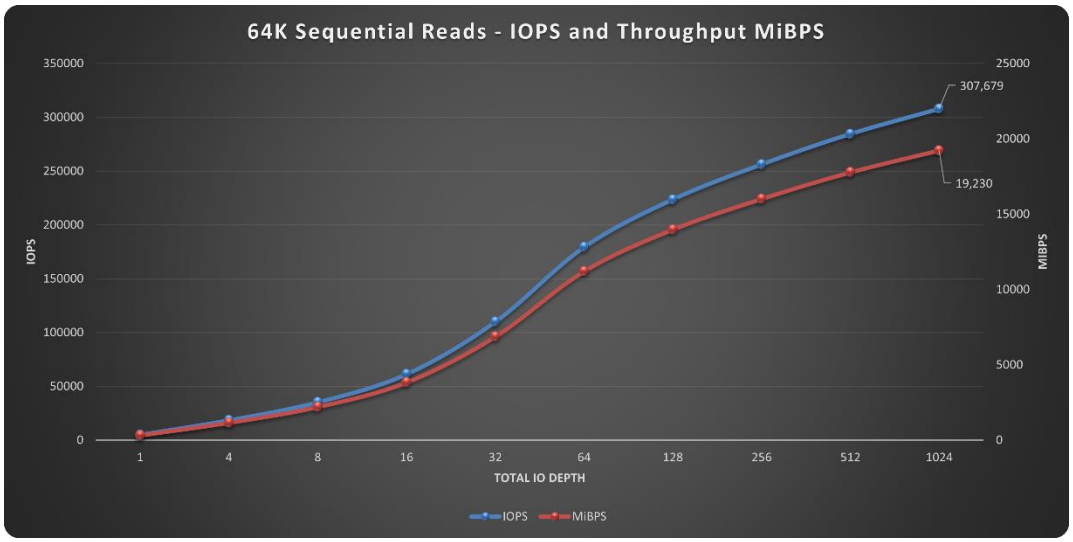


Figure 24. RHEL 9.2 – 64k Sequential Read IOPS and Throughput (MiBPS)

For the 64k sequential read workload, we ran the fio jobs on four RHEL 9.2 Linux nodes simultaneously. In this test, we found the SDCs can attain 307,679 IOPS. An SDC can also attain 16,011 MiB/s throughput with less than 1 millisecond response time.

Table 25. RHEL 9.2 – 64k Sequential Writes – IOPS and Throughput (MiBPS) Over Four - 100Gbs Ethernet Adapters

Total IO Depth	Write Throughput (MiB/s)	IOPS	Latency (µs)
1	245	3,919	241
4	895	14,323	268
8	1,655	26,481	291
16	2,912	46,597	331
32	4,121	65,930	474
64	4,554	72,863	866
128	4,829	77,262	1,645
256	5,022	80,345	3,175
512	5,216	83,451	6,123
1024	5,366	85,858	11,914

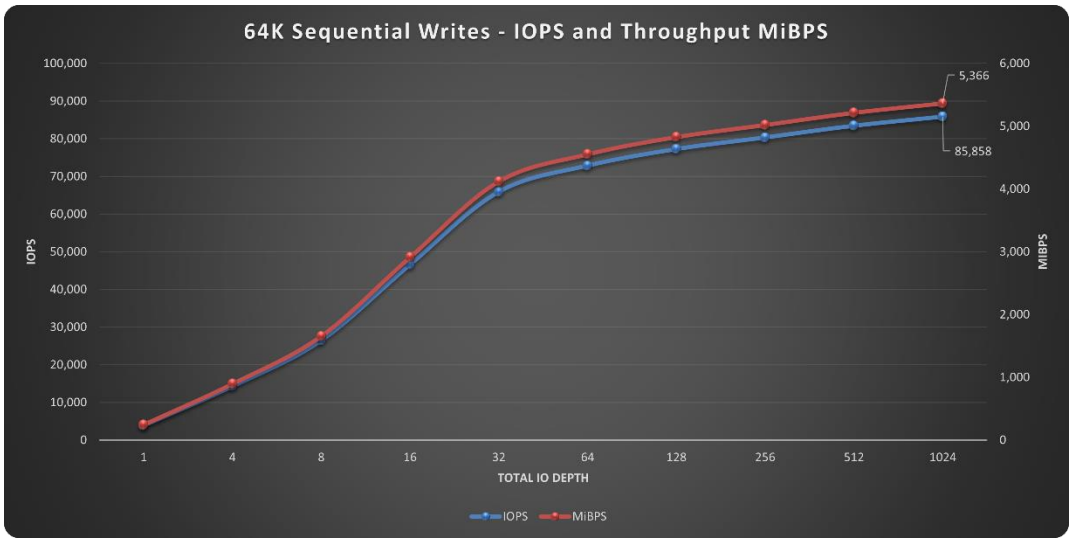


Figure 25. RHEL 9.2 – 64k Sequential Writes – IOPS and Throughput (MiBPS) Over Four - 100Gbs Ethernet Adapters

For the 64k sequential write workload, we ran the fio jobs on four RHEL 9.2 Linux nodes simultaneously. In this test, we found the SDCs can attain 85,858. An SDC can also attain 4,554 MiB/s throughput with less than 1 millisecond response time.

Evaluation of results for Red Hat Enterprise Linux Volumes

We have compiled the results in the following charts to show the comparison of the 25GbE and 100GbE Ethernet adapter. The following charts compare the throughput results of four 25GbE Ethernet adapters, two 100GbE Ethernet adapters, and four 100GbE Ethernet adapters.

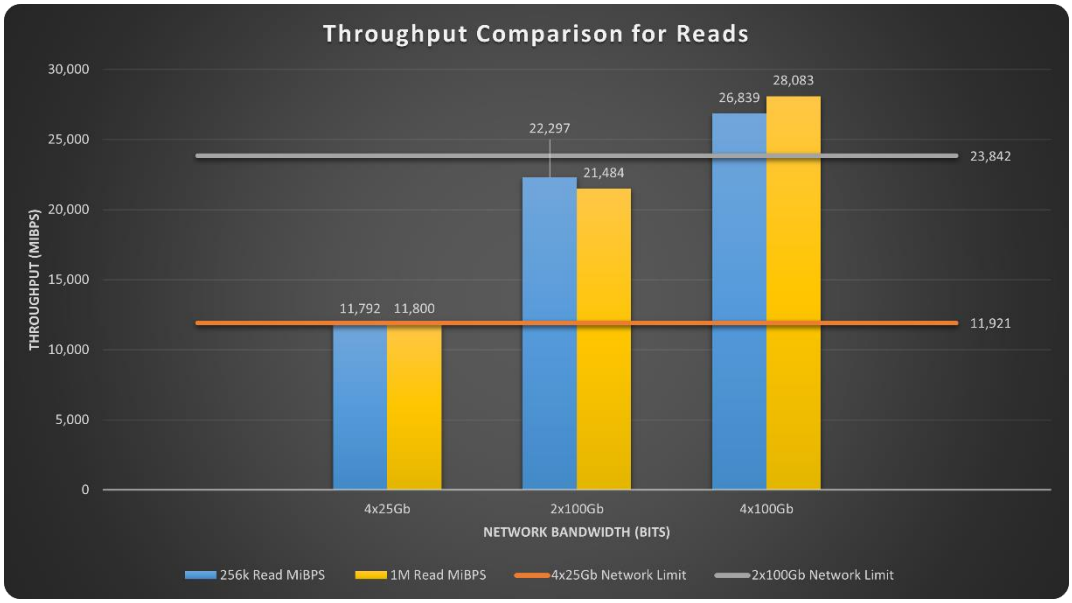


Figure 26. RHEL 9.2 – Read Throughput Comparison – Four 25GbE vs. Two 100GbE vs. Four 100GbE Ethernet adapters

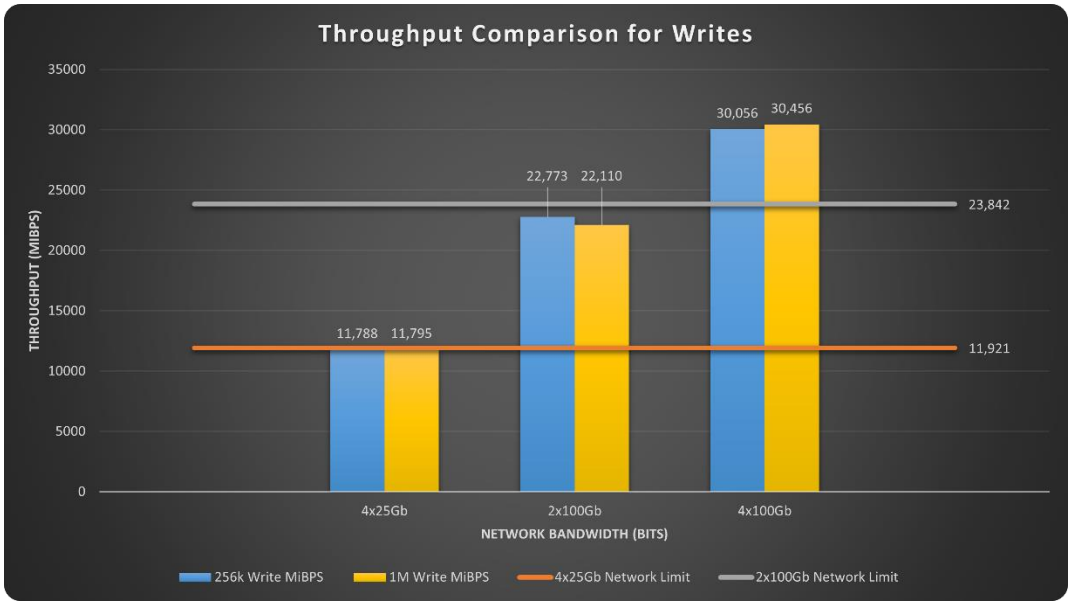


Figure 27. RHEL 9.2 – Write Throughput Comparison – Four 25GbE vs. Two 100GbE vs. Four 100GbE Ethernet adapters

The test results show that two 100GbE Ethernet adapters provide an 89% improvement in 256k read throughput, and an 82% improvement in 1M read throughput compared to four 25Gbs Ethernet adapters. The improvement in writes is even greater with a 93% improvement in 256k write throughput and an 87% improvement in 1M write throughput.

The two 100GbE Ethernet adapters are within 10% of the theoretical line rate of two 100GbE Ethernet adapters.

This shows an increase in storage performance and means that it is more beneficial for a RHEL environment to use 100GbE Ethernet adapters when using PowerFlex for the underlying storage. This benefit can be seen across the different test scenarios performed.

This also indicates that the optimal network configuration for a RHEL environment using PowerFlex as the underlying storage platform is to leverage 100GbE Ethernet adapters for optimal storage throughput as compared to 25GbE Ethernet adapters.

Block volume testing – SUSE Enterprise Linux 15.4

This section looks at the testing results for a SUSE Enterprise Linux 15.4 host configured as a PowerFlex CO node using PowerFlex storage.

Testing observations – SUSE Enterprise Linux using the 25GBbps Ethernet Adapter

We tested with the bare-metal environment described in the System architecture section using the **25GbE Ethernet adapter**. The fio was installed during the initial setup of the node.

In the PowerFlex Manager UI we created three volumes of 320GB in size and presented them to the SUSE Enterprise Linux hosts.

The environment was then configured using the fio tool guidance. We issued the fio command from a console to run a 20% random write test. The configuration of the fio command is detailed in Appendix B – fio configuration for all the test states.

The results for the 20/80 random write performance using the 25GbE Ethernet adapter are shown in the following table and chart:

Table 26. SLES 15.4 – 20% Random Write IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	143	1,973	505,025	127	493	126,310	140	2,466	631,335
8k	149	3,737	478,379	138	934	119,549	147	4,671	597,928
16k	166	6,739	431,322	170	1,685	107,832	167	8,424	539,154
64k	201	11,751	188,012	235	2,941	47,048	208	14,690	235,046
256k	340	11,785	47,142	479	2,953	11,813	368	14,739	58,954

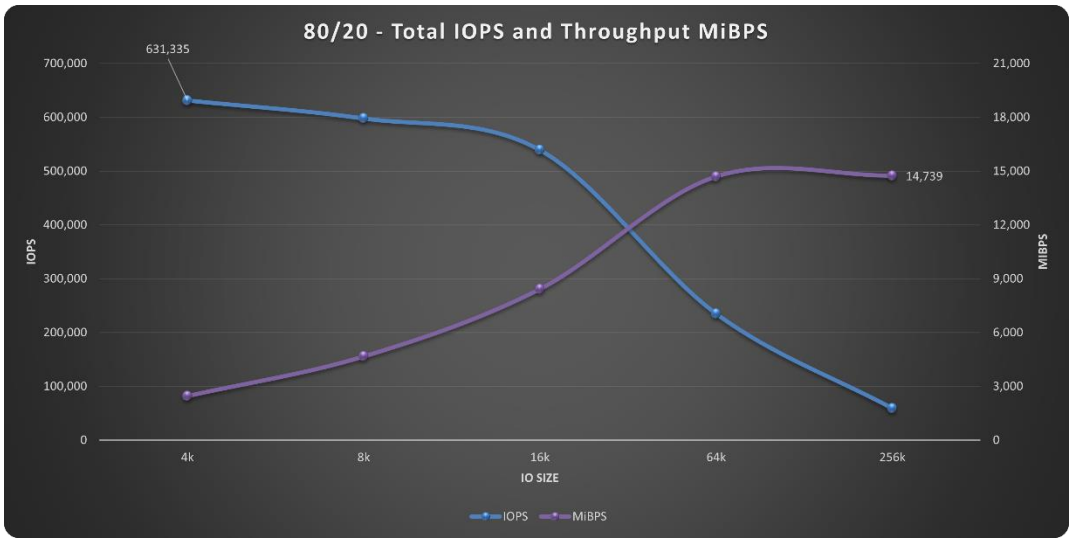


Figure 28. SLES 15.4 – 20% Random Write IOPS and Throughput (MiBPS)

We can see that a single SDC in a SLES 15.4 Linux node can attain 631,335 IOPS with a 4k block size from the PowerFlex environment. Additionally, we can see IOPS for the other block sizes tested.

We show the tests for 50% and 80% reads and writes with fio on the RHEL host. The results for each of these is shown in the following tables and charts:

Table 27. SLES 15.4 – 50% Random Write IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	144	1,220	312,275	126	1,220	312,345	135	2,440	624,620
8k	150	2,156	276,007	138	2,157	276,060	144	4,313	552,068
16k	166	3,721	238,156	166	3,723	238,271	166	7,444	476,427
64k	206	9,265	148,246	231	9,270	148,316	218	18,535	296,562
256k	340	9,444	37,776	477	9,441	37,763	409	18,878	75,513

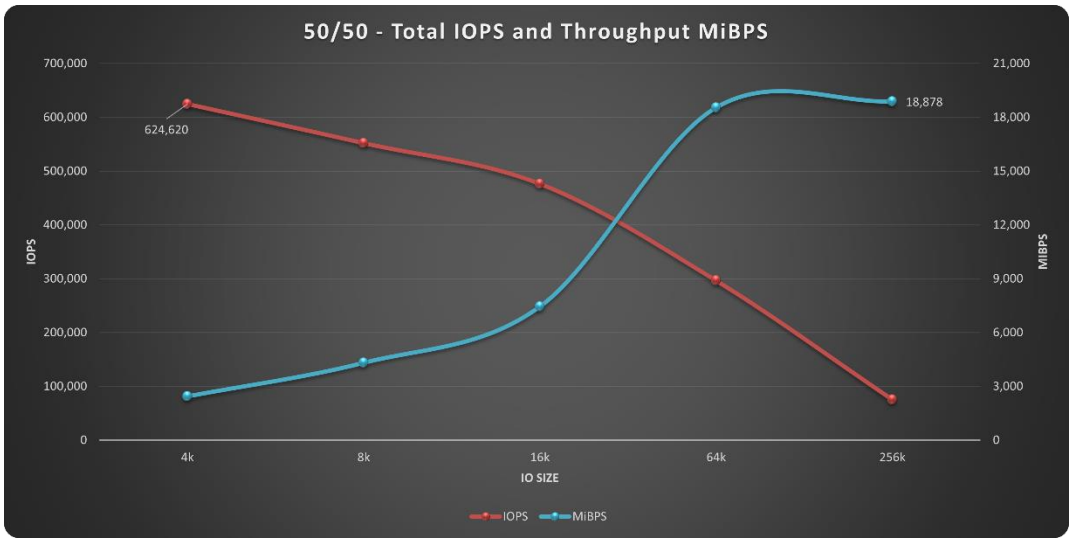


Figure 29. SLES 15.4 – 50% Random Write IOPS and Throughput (MiBPS)

We can see that a single SDC in a SLES 15.4 Linux node can attain 624,620 IOPS with a 4k block size from the PowerFlex environment.

Table 28. SLES 15.4 – 80% Random Write IOPS and Throughput (MiBPS) Over Four - 25GbE Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	143	429	109,937	124	1,718	439,742	128	2,147	549,679
8k	150	759	97,203	136	3,036	388,557	139	3,795	485,759
16k	166	1,285	82,214	166	5,137	328,763	166	6,422	410,976
64k	205	2,350	37,592	230	9,394	150,304	225	11,743	187,896
256k	325	2.365	9,460	476	9,458	37,833	446	11,823	47,293

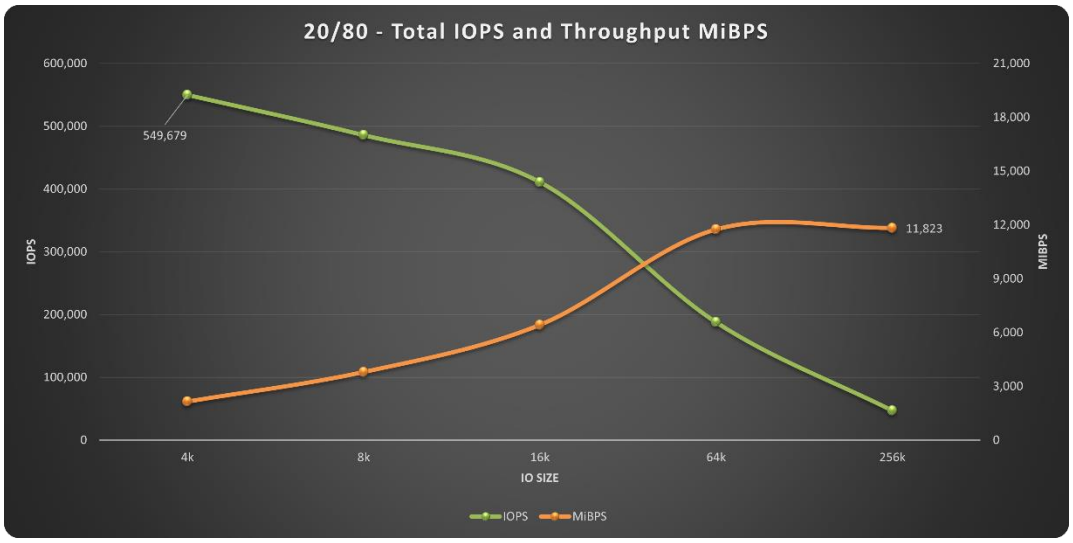


Figure 30. SLES 15.4 – 80% Random Write IOPS and Throughput (MiBPS)

We can see that a single SDC in a SLES 15.4 Linux node can attain 549,679,620 IOPS with a 4k block size from the PowerFlex environment.

Table 29. SLES 15.4 – 64k Sequential Reads – IOPS and Throughput (MiBPS) Over Four - 25Gbs Ethernet Adapters

Total IO Depth	Read Throughput (MiB/s)	IOPS	Latency (µs)
1	322	5,145	194
4	1,111	17,777	225
8	2,023	32,365	247
16	3,674	58,776	272
32	6,226	99,622	321
64	8,581	137,298	466
128	9,935	158,957	805
256	10,790	172,648	1,482
512	11,362	181,788	2,816
1024	11,655	186,475	5,491

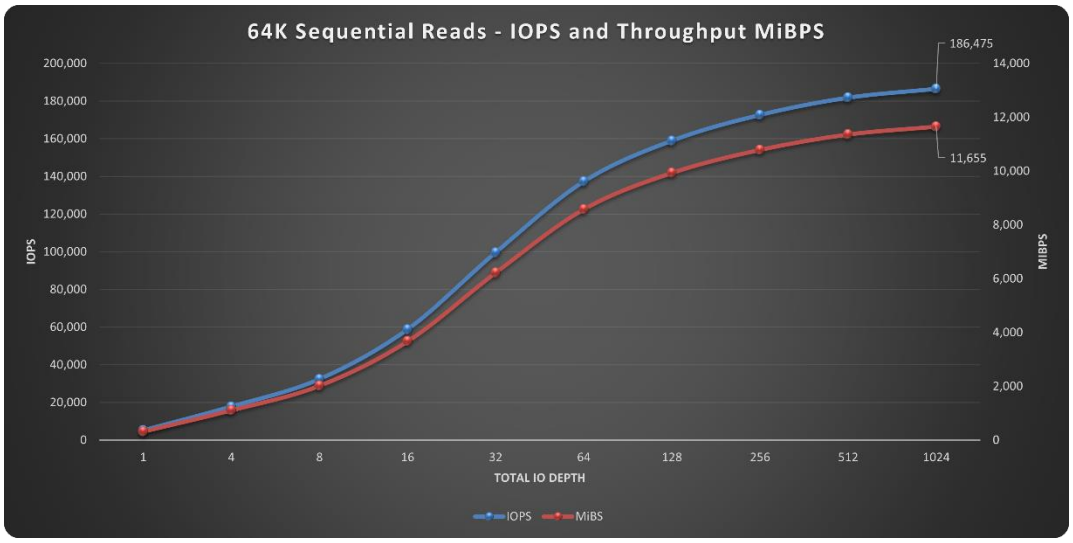


Figure 31. SLES 15.4 – 64k Sequential Reads IOPS and Throughput (MiBPS)

For the 64k sequential read workload, we ran the fio jobs on four SLES 15.4 Linux nodes simultaneously. In this test, we found the SDCs can attain 186,475. An SDC can also attain 9,935 MiB/s throughput with less than 1 millisecond response time.

Table 30. SLES 15.4 – 64k Sequential Writes – IOPS and Throughput (MiBPS) Over Four - 25Gbs Ethernet Adapters

Total IO Depth	Write Throughput (MiB/s)	IOPS	Latency (µs)
1	244	3,903	239
4	844	13,505	284
8	1,503	24,055	321
16	2,429	38,859	400
32	3,027	48,440	649
64	3,260	52,160	1,216
128	3,406	54,498	2,338
256	3,496	55,936	4,566
512	3,564	57,024	8,967
1024	3,628	58,044	17,629

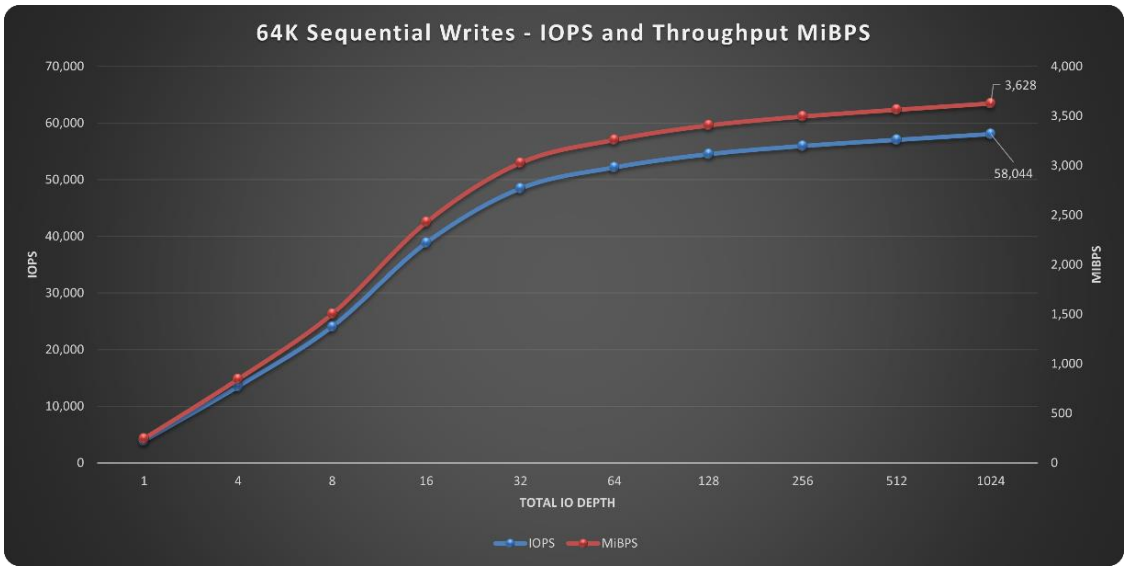


Figure 32. SLES 15.4 – 64k Sequential Writes IOPS and Throughput (MiBPS)

For the 64k sequential write workload, we ran the fio jobs on four SLES 15.4 Linux nodes simultaneously. In this test, we found the SDCs can attain 58,044. An SDC can also attain 3,027 MiB/s throughput with less than 1 millisecond response time.

Note: All testing with the 25GbE Ethernet adapter detailed in this paper was performed prior switching to the 100GbE adapter. This was done to reduce number of changes from one adapter to the other.

Testing observations – SUSE Enterprise Linux Volumes Using the 100GBbps Ethernet Adapter

Following completion of the 25GbE Ethernet adapter testing, the environment was shut down and reconfigured with the 100GbE Ethernet adapters. This included the four PowerFlex SO nodes and the SLES Enterprise Linux CO nodes.

Using the reconfigured SUSE Enterprise Linux environment described in the System architecture section with the 100GbE Ethernet adapters. The fio was installed during the initial setup of the node. In the PowerFlex Manager UI we created three volumes of 320GB in size and presented them to the SUSE Enterprise Linux host.

The environment was then configured using the fio tool guidance. We issued the fio command from a console to run a 20% random write test. The configuration of the fio command is detailed in Appendix B – fio configuration for all the test states.

The results for the 20% random write performance using the 100GbE Ethernet adapter are shown in the following table and chart:

Table 31. SLES 15.4 – 20% Random Write IOPS and Throughput (MiBPS) Over Two - 100Gbs Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	142	1,220	534,858	128	522	133,696	140	2,612	668,554
8k	156	3,842	491,832	134	961	122,945	143	4,803	614,777
16k	150	6,568	420,384	140	1,641	105,017	148	8,209	525,401
64k	196	15,144	242,310	233	3,786	60,580	204	18,931	302,891
256k	263	21,104	84,414	371	5,275	21,102	285	26,379	105,516

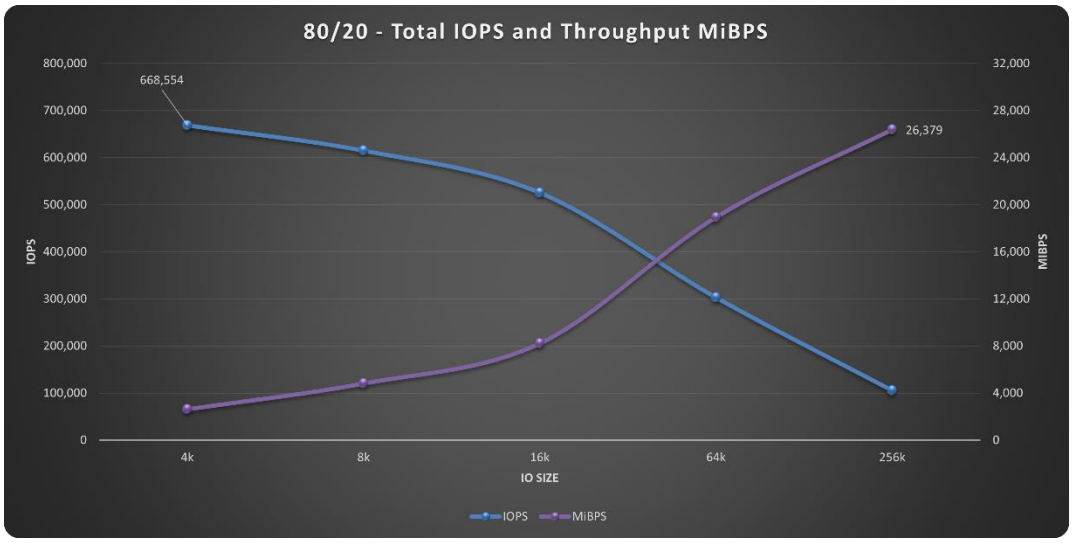


Figure 33. SLES 15.4 – 20% Random Write IOPS and Throughput (MiBPS)

We can see that a single SDC in a SLES 15.4 Linux node can attain 668,554 IOPS with a 4k block size from the PowerFlex environment. Additionally, we can see IOPS for the other block sizes tested.

We then proceeded to run tests for the 50% and 80% reads and writes with fio.

The results for each of these is shown in the following tables and charts:

Table 32. SLES 15.4 – 50% Random Write IOPS and Throughput (MiBPS) Over Two - 100Gbs Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	143	1,248	319,475	127	1,248	319,363	135	2,495	638,837
8k	146	2,235	286,122	133	2,236	286,211	139	4,471	572,333
16k	151	3,683	235,713	140	3,683	235,728	146	7,366	471,441

64k	196	9,129	146,057	230	9,131	146,101	213	18,260	292,158
256k	270	14,042	56,170	366	14,063	56,250	318	28,105	112,420

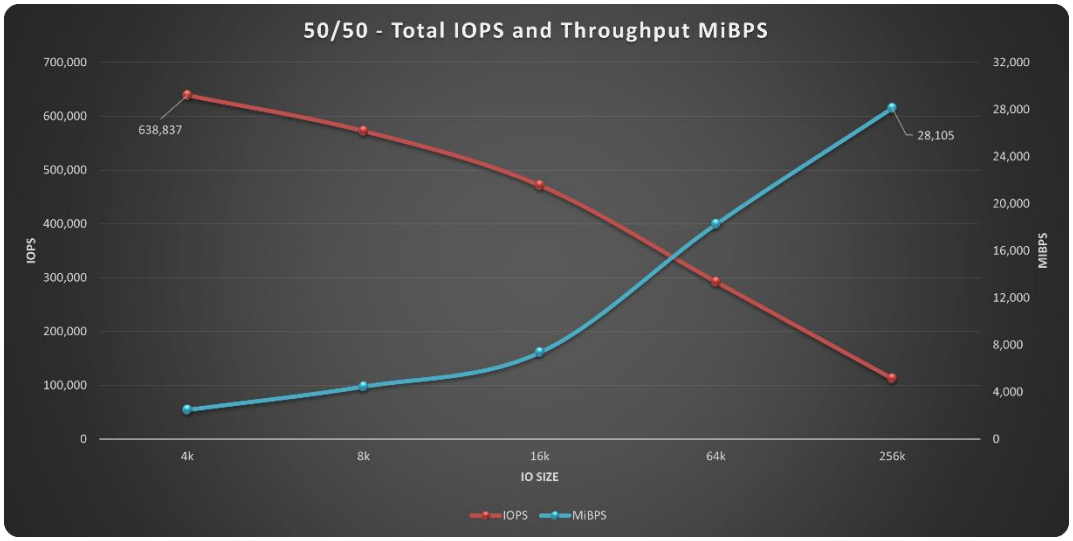


Figure 34. SLES 15.4 – 50% Random Write IOPS and Throughput (MiBPS)

We can see that a single SDC in a SLES 15.4 Linux node can attain 638,837 IOPS with a 4k block size from the PowerFlex environment.

Table 33. SLES 15.4 – 80% Random Write IOPS and Throughput (MiBPS) Over Two - 100Gbs Ethernet Adapters

	Read			Write			Total		
IO Size	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**	**Mean RT (µs)**	**MiBPS**	**IOPS**
4k	143	518	132,709	126	2,075	531,234	129	2,594	663,943
8k	146	869	111,186	132	3,472	444,439	135	4,341	555,624
16k	151	1,409	90,176	139	5,632	360,474	142	7,041	450,650
64k	197	3,582	57,318	229	14,330	229,275	222	17,912	286,593
256k	272	5,471	21,885	364	21,837	87,348	346	27,308	109,233

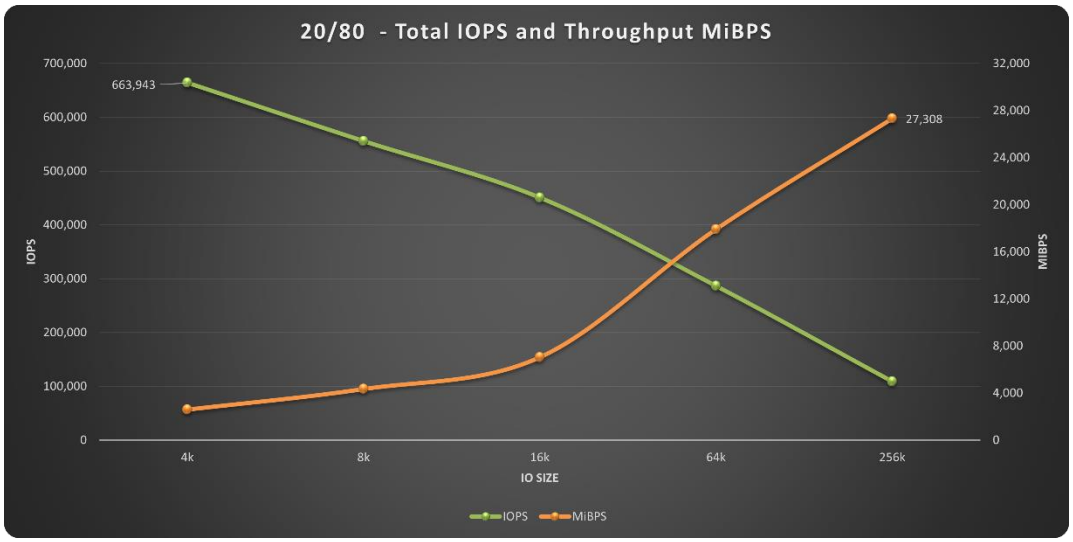


Figure 35. SLES 15.4 – 80% Random Write IOPS and Throughput (MiBPS)

We can see that a single SDC in a SLES 15.4 Linux node can attain 663,943 IOPS with a 4k block size from the PowerFlex environment.

Table 34. SLES 15.4 – 64k Sequential Reads – IOPS and Throughput (MiBPS) Over Four - 100Gbs Ethernet Adapters

Total IO Depth	Read Throughput (MiB/s)	IOPS	Latency (µs)
1	336	5,382	185
4	1,221	19,531	204
8	2,267	36,267	220
16	3,929	62,866	254
32	6,972	111,546	287
64	11,277	180,431	354
128	14,407	230,515	555
256	16,834	269,349	950
512	18,754	300,069	1,706
1024	20,077	321,237	3,187

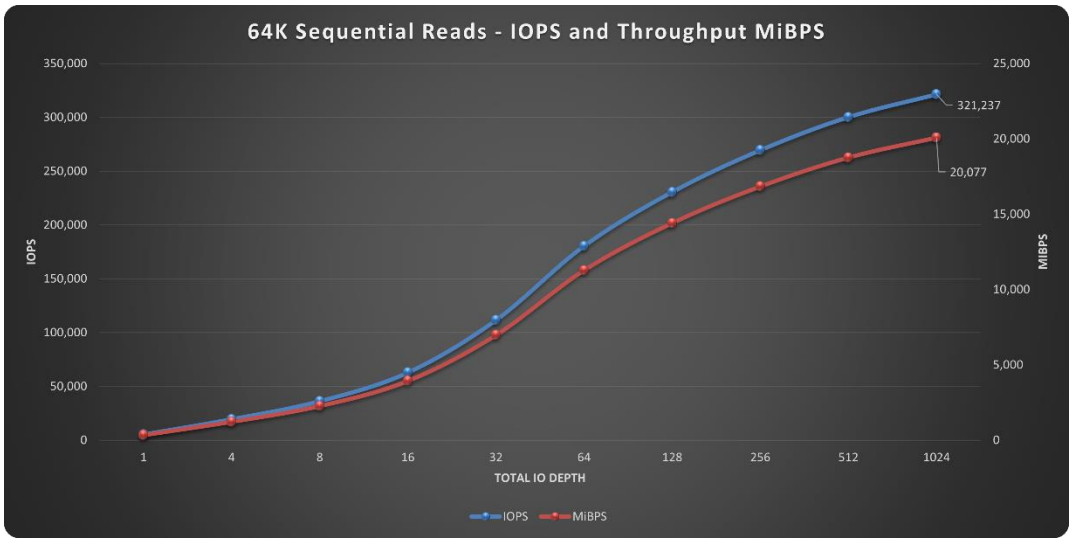


Figure 36. SLES 15.4 – 64k Sequential Reads IOPS and Throughput (MiBPS)

For the 64k sequential read workload, we ran the fio jobs on four SLES 15.4 Linux nodes simultaneously. In this test, we found the SDCs can attain 321,217 IOPS. An SDC can also attain 16,834 MiB/s throughput with less than 1 millisecond response time.

Table 35. SLES 15.4 – 64k Sequential Writes – IOPS and Throughput (MiBPS) Over Four - 100Gbs Ethernet Adapters

Total IO Depth	Write Throughput (MiB/s)	IOPS	Latency (µs)
1	251	4,024	237
4	906	14,498	264
8	1,688	27,012	285
16	3,043	48,694	318
32	4,370	69,921	446
64	4,863	77,812	811
128	5,162	82,599	1,538
256	5,327	85,234	2,993
512	5,426	86,821	5,886
1024	5,553	88,845	11,513

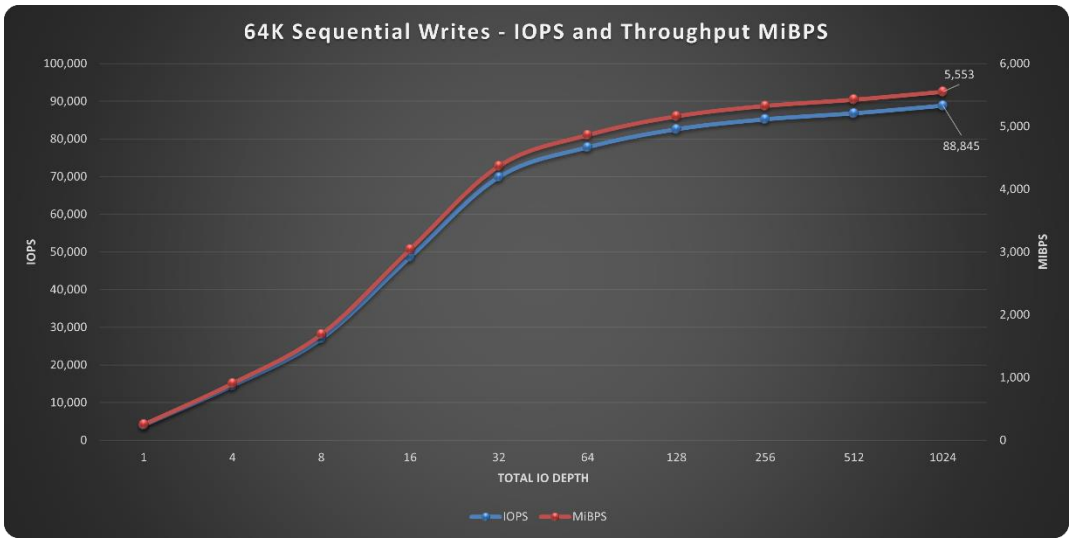


Figure 37. SLES 15.4 – 64k Sequential Writes IOPS and Throughput (MiBPS)

For the 64k sequential read workload we ran the fio jobs on four SLES 15.4 Linux nodes simultaneously. In this test we found that the SDCs can attain 88,845 IOPS. An SDC can also attain 4,863 MiB/s throughput with less than 1 millisecond response time.

Evaluation of results for SUSE Enterprise Linux Volumes

We compiled the results in the following charts to show the comparison of the 25GbE and 100GbE Ethernet adapters:

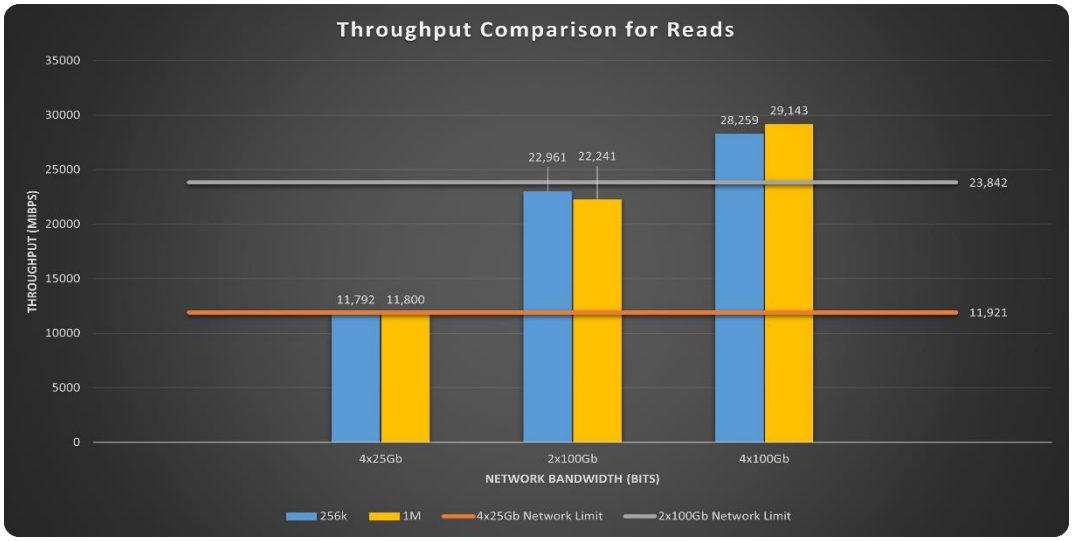


Figure 38. SLES 15.4 – Read Throughput Comparison – Four 25GbE vs. Two 100GbE vs. Four 100GbE Ethernet adapters

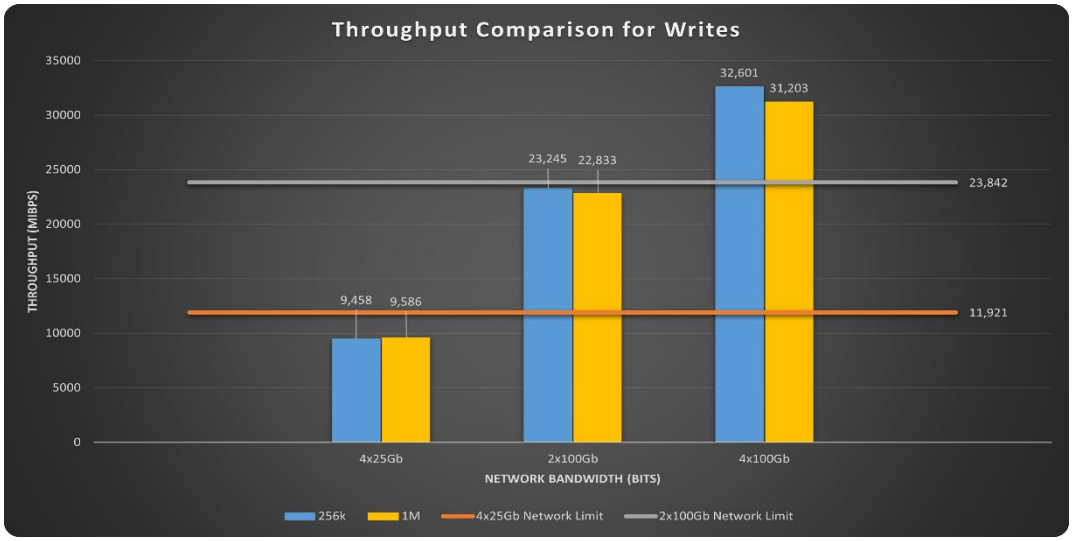


Figure 39. SLES 15.4 – Write Throughput Comparison – Four 25GbE vs. Two 100GbE vs. Four 100GbE Ethernet adapters

The test results show that two 100GbE Ethernet adapters provide a 94% improvement in 256k read throughput and an 88% improvement in 1M read throughput, compared to four 25Gbs Ethernet adapters. The improvement in writes is even greater with a 145% improvement in 256k write throughput and an 138% improvement in 1M write throughput. The two 100GbE Ethernet adapters are within 4% of the theoretical line rate of two 100GbE Ethernet adapters.

This shows an increase in storage performance and means that it is beneficial for a SUSE Enterprise Linux environment to use 100GbE Ethernet adapters when using PowerFlex for storage. This benefit can be seen across the different test scenarios.

This also indicates that the optimal network configuration for a SUSE Enterprise Linux environment using PowerFlex as the underlying storage platform is to leverage 100GbE Ethernet adapters instead of 25GbE Ethernet adapters.

Conclusion

The extensive testing performed clearly shows the performance advantages of using Broadcom 100GbE network interfaces in Dell PowerFlex software-defined infrastructure. Using 100GbE networking interfaces enhances PowerFlex's unmatched performance, scalability, and workload consolidation. By adopting 100GbE networking in PowerFlex, customers can consolidate even more performance intensive applications.

From this testing it is clear that the use of 100GbE networking is preferred over 25GbE to obtain the best performance from a PowerFlex environment. To discover more about optimal networking for PowerFlex storage, contact your Dell representative.

References

Dell Technologies documentation

The following Dell Technologies documentation provides other information related to this document. Access to these documents depends on your login credentials. If you do not have access to a document, contact your Dell Technologies representative.

- *PowerFlex Info Hub*
- *PowerFlex Overview*

Broadcom documentation

The following Broadcom documentation provides additional and relevant information:

- *BCM 957508-P2100G Data Sheet*
- *High-Performance, Feature-Rich NetXtreme® E-Series BCM57508 200GbE PCIe 4.0 Ethernet Controller*
- *BCM57500 200 Gb/s Ethernet Controller with PCIe 4.0, 50G PAM-4, and 1/2/4-Port – Product Brief*

Appendix A – Hardware overview

Dell PowerFlex overview

PowerFlex family

PowerFlex software-defined infrastructure enables broad consolidation across the data center, encompassing almost any type of workload and architecture. The software-defined architecture offers automation and programmability of the complete infrastructure and provides scalability, performance, and resiliency to enable effortless adherence to stringent workload service level agreements (SLAs).

The PowerFlex family provides a foundation that combines compute and high-performance storage resources in a managed unified fabric. PowerFlex comes in flexible deployment options (rack, appliance, or custom nodes and in the public cloud) that enable independent (two-layer), HCI (single-layer), or mixed architectures. PowerFlex is ideal for high-performance applications and databases, building an agile private/hybrid cloud, or consolidating resources in heterogeneous environments.

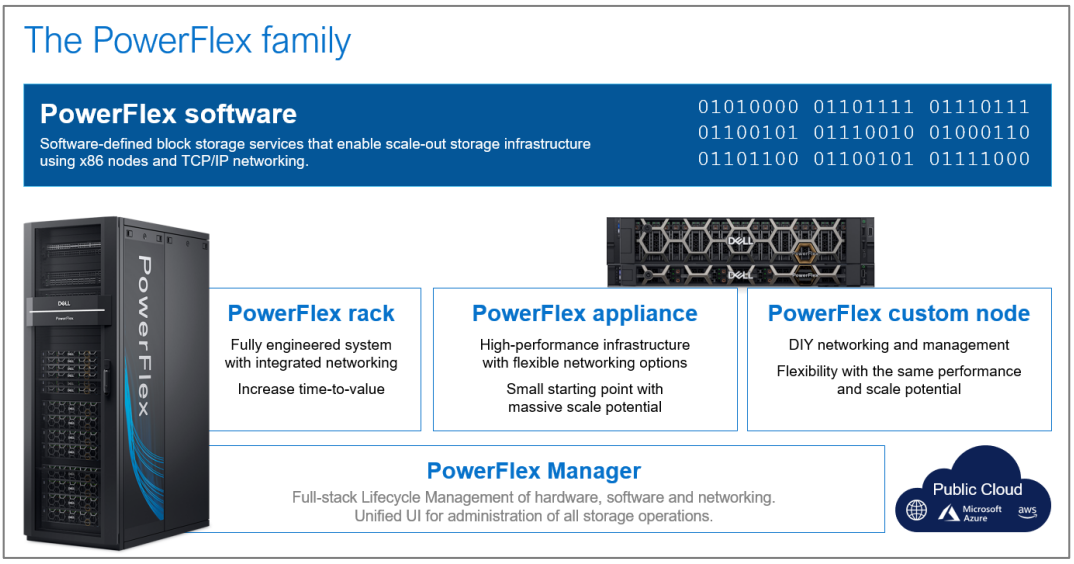


Figure 40. PowerFlex family

PowerFlex software components

Software is the key differentiation in the PowerFlex offering. PowerFlex software components not only provide software-defined storage services, but also help simplify infrastructure management and orchestration. This enables comprehensive IT Operational Management (ITOM) and Life Cycle Management (LCM) capabilities that span compute as well as storage infrastructure, from BIOS and firmware to nodes, software, and networking.

PowerFlex

PowerFlex is the software foundation of PowerFlex software-defined infrastructure. It is a scale-out block and file storage service that is designed to deliver flexibility, elasticity, and simplicity with predictable high performance and resiliency at scale.

PowerFlex Manager

PowerFlex Manager is the software component in the PowerFlex family that enables ITOM automation and LCM capabilities for PowerFlex systems. Starting with PowerFlex

4.0, the unified PowerFlex Manager brings together three separate components that are used in previous releases, PowerFlex Manager, the core PowerFlex UI, and the PowerFlex gateway. The new PowerFlex UI runs in Kubernetes and embraces a modern development framework.

PowerFlex with NVMe over TCP (NVMe/TCP)

NVMe is a family of open specifications that defines an industry standard for both a logical interface and a communication protocol for accessing block storage across an internal PCIe fabric or through external fabrics such as TCP/IP and FC. NVMe is designed to take advantage of the internal parallelism and low latency inherent in flash-based media SSDs.

NVMe/TCP is an extension of the NVMe base specification that defines the binding of the NVMe protocol to message-based fabrics using TCP. By binding the NVMe protocol to TCP, NVMe/TCP enables the efficient end-to-end transfer of commands and data between NVMe hosts and NVMe controller devices by any standard Ethernet-based TCP/IP networks.

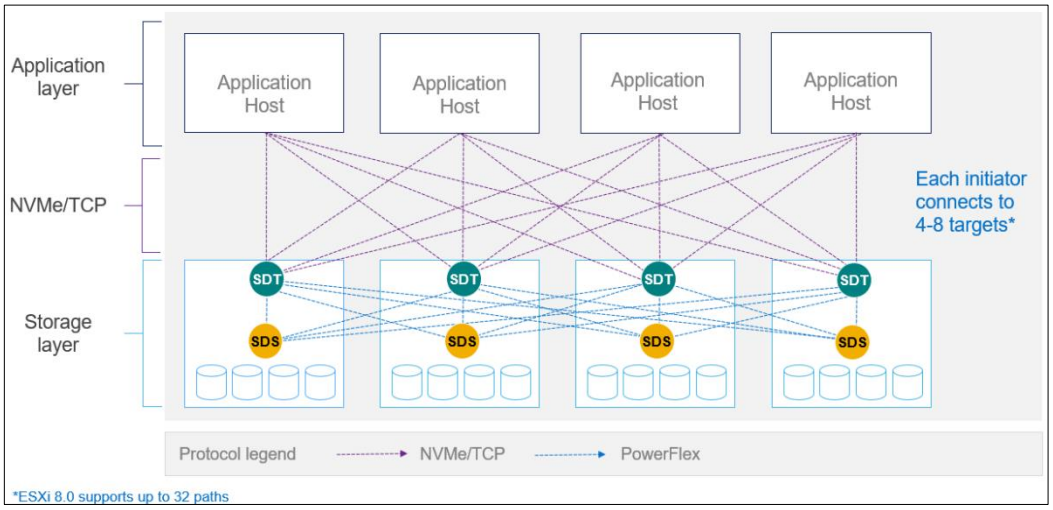


Figure 41. PowerFlex with NVMe/TCP

From PowerFlex version 4.x onwards, support for NVMe/TCP has been added for on-premises deployment with the introduction of a new Storage Data Target (SDT) component installed in the storage layer. The NVMe initiator in the OS or hypervisor communicates with the SDT, which then communicates with the Storage Data Server (SDS). The NVMe initiator is part of the kernel of the application host operating system or hypervisor.

For more information, see PowerFlex with NVMe over TCP.

PowerFlex deployment architectures

PowerFlex software-defined infrastructure excels in deployment flexibility. PowerFlex can be deployed in a two-layer (independent compute and storage layers), single-layer (hyperconverged infrastructure, or HCI), or a mixture of the two architectures (mixed).

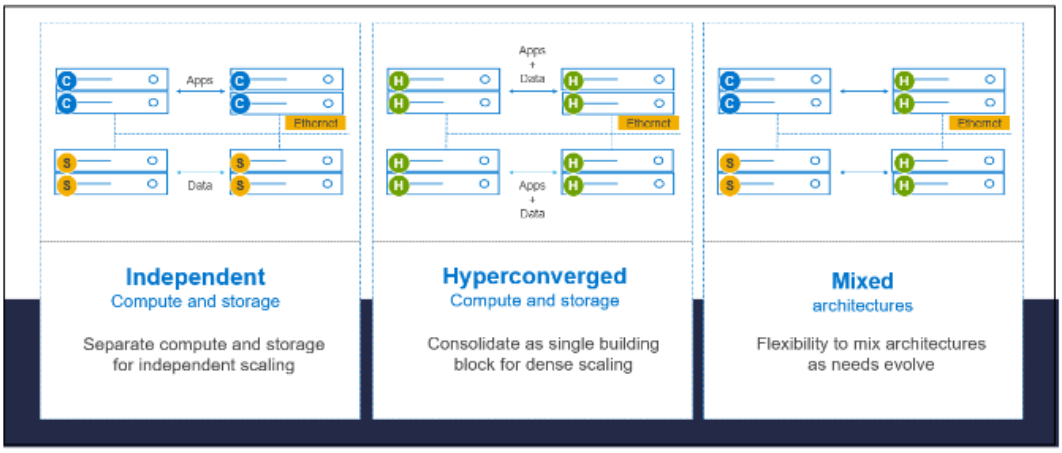


Figure 42. PowerFlex deployment architectures

Independent architecture

In an independent architecture or two-layer architecture, some nodes provide storage capacity for data in applications, while other separate and independent nodes provide compute resources for applications and workloads. Compute and storage resources can be scaled independently by adding nodes to the cluster while it remains active. This separation of compute and storage resources helps to minimize software licensing costs in certain situations. This architecture can be ideal for high-performance databases and application workloads.

Hyperconverged architecture

In an HCI architecture, each node in the cluster contributes storage and compute resources simultaneously to the applications and workloads. This architecture allows you to scale your infrastructure uniformly with building blocks that add both storage and compute resources. This architecture is appropriate for data center and workload consolidation.

Mixed architecture

A mixed architecture is a combination of both the HCI and independent architectures. As shown in Figure 42. PowerFlex deployment architectures. There are some storage only nodes, compute only nodes, and HCI nodes as a part of the same PowerFlex cluster. This is a desirable architecture when working with an existing compute infrastructure and adding high-performance software-defined infrastructure. This can also be a starting point for a two-layer deployment design as external workloads are migrated to PowerFlex.

PowerFlex consumption options

PowerFlex rack

PowerFlex rack is a software-defined infrastructure platform that delivers flexibility, elasticity, and simplicity with predictable performance and resiliency at scale by combining compute as well as high-performance storage resources in a managed unified network. This rack-based engineered system, with integrated networking, enables customers to achieve the scalability and management requirements of a modern data center.

PowerFlex appliance

PowerFlex appliance is a PowerEdge server which has been configured to be a node in a software-defined infrastructure deployment that runs PowerFlex software components.

This offering allows customers the flexibility and savings to bring their own compatible networking.

PowerFlex Custom Nodes

PowerFlex Custom Nodes are validated server building-blocks configured for use with PowerFlex. They are available with thousands of configuration options and are available for customers who prefer to build their own environments.

Dell APEX Block Storage for Public Cloud

PowerFlex software is also available in the public cloud as Dell APEX Block Storage for Public Cloud. It offers the same on-premises benefits of high-performance, linear scalability, and high resilience as in cloud. It also adds cloud-specific benefits, such as large volume sizes, extreme performance based on NVMe drives, and predictable scalability. The Dell APEX Block Storage self-healing architecture simplifies and increases the availability of infrastructure. It is designed to remain running even when it encounters failures. With fast rebuilds, Dell APEX Block Storage recovers rapidly from failures and keeps operating, without degrading performance or compromising availability. You can also extend this availability and configure it to protect your application availability across multiple availability zones.

PowerFlex CSI/CSM

An important component outside of PowerFlex that enables a flexible consumption model for Kubernetes is the PowerFlex CSI driver, developed as a part of the Dell Kubernetes strategy. After loading the CSI driver for PowerFlex into Kubernetes, it can be used to provision persistent volumes from the underlying PowerFlex storage resource. If the Kubernetes deployment is running low on PowerFlex storage resources, you can add PowerFlex storage nodes to increase the system capacity and performance.

The CSI driver connects the PowerFlex system and Kubernetes deployments. It is a storage broker which dynamically provisions volumes from PowerFlex through the PowerFlex API gateway to the Kubernetes cluster. Once the volume is available on PowerFlex, it is immediately mapped to the requesting pod. If a pod is destroyed or rescheduled, the CSI plug-in ensures that the volumes are remapped upon rescheduling of that pod.

Customers running Kubernetes clusters on PowerFlex use the Dell Container Storage Modules (CSM), which extend the CSI driver capabilities. These modules:

- Provide enterprise storage capabilities to Kubernetes for cloud-native stateful applications.
- Reduce management complexity, so that developers can independently use enterprise storage with ease and automate daily operations.
- Extend storage functionality and capabilities beyond using the CSI driver alone.

These modules include snapshot, observability, authorization, application mobility, and resiliency.

PowerFlex supports multiple operating systems, different deployment options, and on-premises and public cloud deployment models (available in AWS and Azure). PowerFlex is validated with the leading Kubernetes distributions as shown in the following figure:

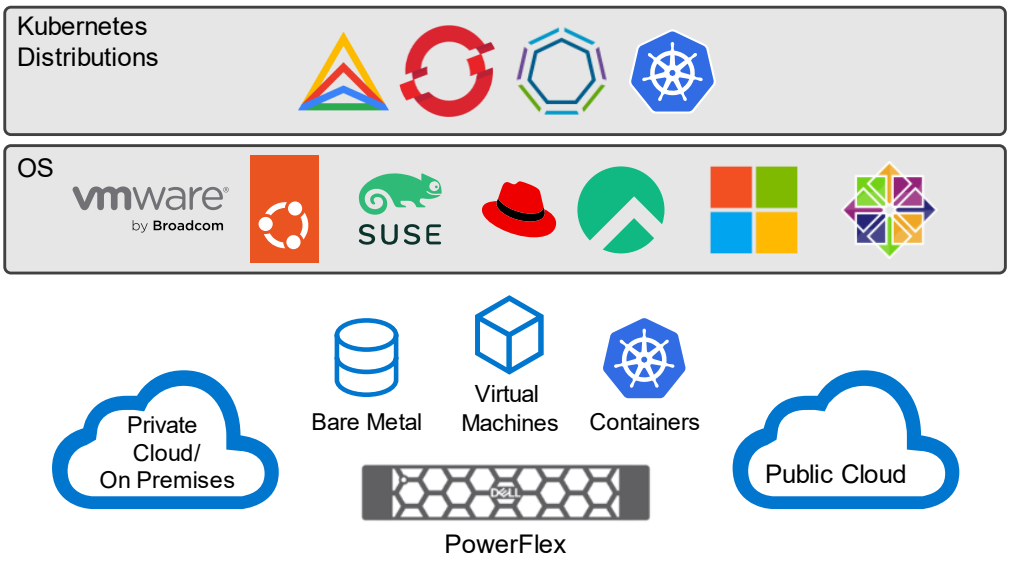


Figure 43. PowerFlex for different Kubernetes distributions

Kubernetes containers

Kubernetes is an open-source container orchestration system that is portable and can be extended to manage containerized workloads and services. Thus, Kubernetes facilitates both declarative configuration and automation. Containers are an effective way to bundle and run your applications because they each have their own file system, CPU, memory, process space, and more. As the containers are decoupled from the underlying infrastructure, they are portable across clouds and OS distributions. Kubernetes provides a framework to run distributed systems resiliently.

For more information about Kubernetes, see the Kubernetes documentation.

Dell CSM Modules

Dell Technologies provides various CSM that are designed to help organizations manage their containerized applications efficiently. These modules provide persistent storage for containerized applications running in a cloud-native environment, while also addressing the challenges of storage in container environments, such as data persistence, scalability, and data protection.

Dell CSM provide a range of features and capabilities that can help organizations manage their containerized applications efficiently, while ensuring scalability, flexibility, and data protection.

The current features available with Dell CSM are:

- **Snapshots**: Built on the CSIs point-in-time recovery with additional capabilities such as group and crash consistent snapshots with referential integrity. CSM for Volume Group Snapshots allows to take crash consistent snapshots of all the volumes that are attached to a pod simultaneously avoiding inconsistent or corrupt volumes which can be used for backup, testing, and restoring.
- **Authorization**: CSM for Authorization provides storage and Kubernetes administrators the ability to apply Role-Based Access Control (RBAC) for Dell CSI Drivers.

- **Resiliency**: CSM for Resiliency is designed to make Kubernetes applications, including the ones that use persistent storage, more resilient to various failures.
- **Application Mobility**: CSM for Application Mobility provides Kubernetes administrators the ability to clone their stateful application workloads and application data to other clusters, either on-premises or in the cloud.
- **Replication**: CSM for Replication is part of the open-source suite of Kubernetes storage enablers for Dell products.
- **Observability**: CSM for Observability is part of the open-source suite of Kubernetes storage enablers for Dell products.
- **Encryption**: Encryption provides the capability to encrypt user data residing on volumes created by Dell CSI Drivers.

For more information, see Dell Container Storage Module.

Broadcom BCM57508 2x100GbE Ethernet Controllers – Overview

Based on Broadcom's scalable 10/25/50/100/200G Ethernet controller architecture, the NetXtreme-E Series BCM57508 200G PCIe 4.0 Ethernet controller is designed to build highly scalable, feature-rich networking solutions in servers for enterprise and cloud-scale networking and storage applications, including high-performance computing, telco, machine learning, storage disaggregation, and data analytics.

The BCM57508 200G PCIe 4.0 Ethernet Controller builds upon the success of the widely deployed NetXtreme E-Series architecture by combining a high-bandwidth Ethernet controller with a unique set of highly optimized hardware acceleration engines to enhance network performance and improve server efficiency.

Driven by the growing need for higher server bandwidth and storage capacity to support new and emerging applications, enterprises, telcos and cloud providers are modernizing their data centers by adopting latest networking technologies, such as server virtualization and NVMe-oF. The resulting expansion in data volumes increases server-to-server traffic and CPU loads for networking tasks.

With its market-leading hardware acceleration technologies, the NetXtreme-E Series BCM57508 200G PCIe 4.0 Ethernet Controller addresses performance and service demands of mega-scale data center networks with high throughput and advanced flow processing. Features such as TruFlow increase VM density up to 50 percent, freeing more CPU cycles for additional virtual machines. On-chip tunneling protocol processing for Geneve, VXLAN, and NVGRE provides up to a 5X throughput increase while lowering CPU utilization up to 90 percent compared to software-only solutions. NetXtreme-E Series solutions also support advanced networking technologies such as RoCE, SDN, and NFV, to facilitate the management of data networks and to enable service provider solutions.

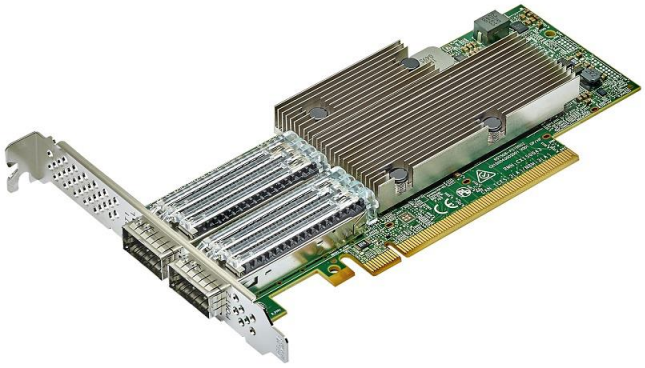


Figure 44. BCM957508-P2100G Network Interface Card

Building upon the success of the widely deployed Broadcom NetXtreme-E architecture, the BCM57500 series Ethernet Controllers set the bar higher by supporting single, dual, and quad-port configurations with 200 Gb/s aggregate throughput. The BCM57500 supports network interfaces of 200GbE, 100GbE, 50GbE, 40GbE, 25GbE, and 10GbE.

Incorporating industry-leading PHY technology, the BCM57500 provides highly efficient network connectivity using eight lanes of built-in SerDes, each supporting 50G PAM-4 and 25G NRZ. When paired with Broadcom's market-leading Ethernet switches StrataXGS or StrataDNS, the BCM57500 assures the highest level of performance, interoperability, and features.

The BCM57500 supports RDMA over Converged Ethernet (RoCE) with hardware-based congestion control. Broadcom's RoCE congestion control not only delivers the lowest latency in real-life scenarios but dramatically reduces the complexity of RoCE deployment at scale.

The BCM57500 TruFlow technology integrates hardware acceleration with enhanced programmable parser/lookup/match/action engine. TruFlow increases virtual machine density and improves application performance.

The BCM57500 TruTrust technology enables the industry's most secure server platform with Broadcom's Silicon Root of Trust and Secure Boot Loader. TruTrust provides absolute protection by locking down the lowest level resources against attacks.

The BCM57500 is PCIe 4.0 compliant and backwards compatible with versions 3.0, 2.0, and 1.1.

In this paper, we chose to use the 2-port 100GB configuration of the BCM57508 Ethernet controller. We did not test the 10/25/50/200G controller configurations.

Appendix B – fio configuration

fio CLI Command

The fio tool has many arguments that can be passed to it. In this section we describe the arguments used for testing in the environment as defined in the fio documentation. For more information about fio visit https://fio.readthedocs.io/en/latest/

The following is the example fio command that was used to test the PowerFlex environment with Linux CO nodes.

```
fio --name=$OUTPUTFILE --readwrite=randrw --rwmixwrite=$RWMIXWRITE
--bs=$BLOCKSIZE --invalidate=1 --direct=1 --group_reporting --
filename=/dev/scinia --filename=/dev/scinib --filename=/dev/scinic
--time_based --runtime=100 --refill_buffers --norandommap --
randrepeat=0 --ioengine=libaio --numjobs=$NUMJOBS --
iodepth=$QUEUEDEPTH --output-format=json --
output=/root/$OUTPUTFILE.json
```

This command performs the following actions:

- `--name=<str>`
  ASCII name of the job. This may be used to override the name printed by fio for this job. Otherwise, the job name is used. On the command line this parameter has the special purpose of also signaling the start of a new job.
  Using variable: *`$OUTPUTFILE`*

- `--readwrite=randrw`
  Instructs fio to run in a random read/write configuration

- `--rwmixwrite=<int write percentage>`
  Percentage of a mixed workload that should be writes.
  Using variable: *`$RWMIXWRITE`*

- `--bs=<Block size in KiB ([int]k)>`
  Block size used in KiB
  Using variable: *`$BLOCKSIZE`*

- `--invalidate=1`
  Invalidate the buffer/page cache parts of the files to be used prior to starting I/O if the platform and file type support it. Defaults to true. This will be ignored if pre_read is also specified for the same job.

- `--direct=1`
  If value is true, use non-buffered I/O. This is usually O_DIRECT. Note that OpenBSD and ZFS on Solaris don't support direct I/O. On Windows the synchronous ioengines don't support direct I/O. Default: false.

- `--group_reporting`
  To see the final report per-group instead of per-job, use group_reporting. Jobs in a file will be part of the same reporting group, unless if separated by a stonewall, or by using new_group.

- `--filename=<device name>`
  fio normally creates a filename based on the job name, thread number, and file number (see filename_format). If you want to share files between threads in a job or several jobs with fixed file paths, specify a filename for each of them to override the default. If the I/O engine is file based, you can specify a number of files by separating the names with a ':' colon. If you want a job to open /dev/sda and /dev/sdb as the two working files, you can use filename=/dev/sda:/dev/sdb. This also means that whenever this option is specified, nrfiles is ignored. The size of regular files specified by this option will be size divided by number of files unless an explicit size is specified by file size.
  Using variable: Based on OS mount paths.

- `--time_based`
  If set, fio runs for the duration of the runtime specified even if the file(s) are completely read or written. It will simply loop over the same workload as many times as the runtime allows.

- `--runtime=100<seconds>`
  Limit runtime. The test will run until it completes the configured I/O workload or until it has run for this specified amount of time, whichever occurs first. It can be difficult to determine for how long a specified job will run, so this parameter can cap the total runtime to a given time. When the unit is omitted, the value is interpreted in seconds.

- `--refill_buffers`
  If this option is given, fio refills the I/O buffers on every submit.

- `--norandommap`
  Normally fio will cover every block of the file when doing random I/O. If this option is given, fio will just get a new random offset without looking at past I/O history. This means that some blocks may not be read or written, and that some blocks may be read/written more than once. If this option is used with verify and multiple blocksizes (using bsrange), only intact blocks are verified, and partially overwritten blocks are ignored. With an async I/O engine and an I/O depth > 1, it is possible for the same block to be overwritten, which can cause verification errors. Either do not use norandommap in this case, or also use the lfsr random generator.

- `--randrepeat=0`
  Seed all random number generators in a predictable way so the pattern is repeatable across runs. Default: true.

- `--ioengine=libaio`
  Defines how the job issues I/O to the file. Libaio is Linux native asynchronous I/O. Note that Linux may only support queued behavior with non-buffered I/O (set direct=1 or buffered=0). This engine defines engine specific options.

- `--numjobs=<int>`
  Create the specified number of clones of this job. Each clone of job is spawned as an independent thread or process. May be used to setup a larger number of threads/processes doing the same thing. Each thread is reported separately; to see statistics for all clones as a whole, use group_reporting in conjunction with

new_group. See --max-jobs. Default: 1.
Using variable: `$NUMJOBS`

- `--iodepth=<queue depth>`
  Number of I/O units to keep in flight against the file. Note that increasing iodepth beyond 1 will not affect synchronous I/O engines (except for small degrees when verify_async is in use). Even async engines may impose OS restrictions causing the desired depth not to be achieved. This may happen on Linux when using libaio and not setting direct=1, since buffered I/O is not async on that OS. Check the I/O depth distribution in the fio output to verify that the achieved depth is as expected. Default: 1.
  Using variable: `$QUEUEDEPTH`

- `--output-format=json`
  Set the reporting format to *normal*, *terse*, *json*, or *json+*. Multiple formats can be selected, separated by a comma. *terse* is a CSV based format. *json+* is like JSON but adds a full dump of the latency buckets.

- `--output=/root/$OUTPUTFILE.json`
  Write output to file filename.
  Using variable: `$OUTPUTFILE`

Variable values for fio

The variables used in the fio command were given the following values during testing:

```
RWMIXWRITE = [0, 20, 50, 80, 100]

BLOCKSIZE = [512b, 4k, 8k, 16k, 32k, 64k, 128k, 256k, 512k, 1m]

NUMJOBS = [1, 4, 8, 16, 24, 32]

QUEUEDEPTH = [1, 2, 4, 8, 16, 32, 64, 128, 256]

NUMJOBS*QUEUEDEPTH <=1024
```