

# FUNGIBLE STORAGE CLUSTER DELIVERS BREAKTHROUGH PERFORMANCE FOR IBM SPECTRUM SCALE



Are you seeking higher storage performance for your Spectrum Scale cluster? Are you getting the full benefits of pooled storage without suffering performance bottlenecks? Fungible Storage Cluster, serving as the backend shared storage platform for Spectrum Scale gives you the efficiency *and* performance to break through the data bottleneck. This paper shows you the breakthrough performance of the Fungible Storage Cluster with IBM Spectrum Scale.

## IBM SPECTRUM SCALE OVERVIEW

IBM Spectrum Scale is a cluster file system that provides concurrent access to a single file system or set of file systems from multiple nodes. The nodes can be SAN attached, network attached, a mixture of SAN attached, and network attached, or in a shared nothing cluster configuration. This enables high performance access to this common set of data to support a scale-out solution or to provide a high availability platform.

IBM Spectrum Scale has many features beyond common data access including data replication, policy-based storage management and multi-site operations. You can create a cluster of AIX nodes, Linux nodes, Windows server nodes or a mix of all three. IBM Spectrum Scale can run on virtualized instances providing common data access in environments, leverage logical partitioning or other hypervisors. Multiple IBM Spectrum Scale clusters can share data within a location or across wide area network (WAN) connections.

## BENCHMARK TOPOLOGY

To demonstrate a Spectrum Scale cluster performance with the Fungible Storage Cluster (3xFS1600 nodes), we setup a 10-node Spectrum Scale cluster attached to the FSC configured with 36 Erasure Coded volumes and 48 replicated volumes (shared), via one 100GbE Mellanox CX5 network card on each node running NVMe over TCP (NVMe/TCP). We used gpfsperf performance test tool included in the GPFS software to run a random read with 4K block size. For comparison, we used FIO with 4K block size and compared the two results.

Figure 1 below shows the test topology for a 10-node Spectrum Scale cluster.

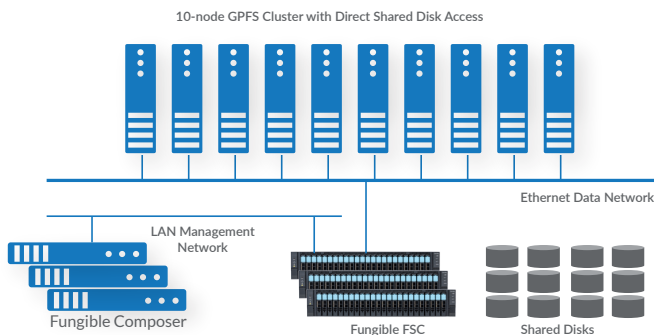


Figure 1: A 10-node Spectrum Scale Cluster

## BENCHMARK RESOURCES

The following tables list all the hardware and software used during the benchmark.

Spectrum Scale Cluster Nodes Components	Quantity / Description
Server Type	10 x Supermicro
Memory	256GB per server
CPU	2x20 Intel(R) Xeon(R) Gold 6138 CPU @ 2.00GHz per server
Network Card	1 x Mellanox ConnectX-5 - 100GbE 1 x 10GbE per server
Direct Attached Storage	1 x SATA SSD as Boot Disk per server
Disaggregated Storage	36 x FSC Erasure Coded Volumes (shared)

Table 1: Spectrum Scale Cluster Nodes Component

Software	Description
Operating System	CentOS 8.2
Linux Kernel	4.18.0-193.6.3.el8_2.x86_64
Spectrum Scale	Version 5.0.5.1
FIO	Version 3.7

Table 2: Software Resources

Network Component	Purpose / Model
Network Switches	NVMe/TCP / Juniper
	Management Network (LAN) / Cisco

Table 3: Network Component

FSC Component	Quantity / Description
Fungible FSC	3 x FS1600 nodes
FS1600 Network Ports	6 x 100GbE
Fungible Composer	Control Plane

Table 4: Fungible FSC Component

## BENCHMARK METHODOLOGY

To compare a random read workload between FIO and gpfsperf, we created a total of 36 Erasure Coded volumes and attached all 36 volumes to all ten Spectrum Scale nodes over NVMe/TCP via the 100GbE Mellanox CX5 network card. We created a single GPFS cluster with ten nodes. Three out of ten nodes were designated as quorum and manager nodes. A single file system was created and mounted under /mnt/gpfs01 across the ten GPFS nodes. We

used gpfspcr tool to create ten 500GB files so each GPFS node can perform the read test on its own file. After the files have been created, we ran a random read test with 4K block size using gpfspcr tool. Then we used FIO to run the same random read workload with 4K block size against the same files which were created by gpfspcr. For the gpfspcr test, we used the DirectIO option (-dio). For the FIO test, we also set "direct=1".

The tables below show the GPFS cluster setup that was used for the test.

```

GPFS cluster name:  gpfsc1uster.fungible.local
GPFS cluster id:   13550697441972321508
GPFS UID domain:  gpfsc1uster.fungible.local
Remote shell command: /usr/bin/ssh
Remote file copy command: /usr/bin/scp
Repository type:   CCR
    
```

Node	Daemon node name	IP address	Admin node name	Designation
1	gtm-server-801.fungible.local	10.90.2.171	gtm-server-801.fungible.local	quorum-manager
2	gtm-server-802.fungible.local	10.90.2.172	gtm-server-802.fungible.local	quorum-manager
3	gtm-server-803.fungible.local	10.90.2.173	gtm-server-803.fungible.local	quorum-manager
4	gtm-server-804.fungible.local	10.90.2.174	gtm-server-804.fungible.local	
5	gtm-server-805.fungible.local	10.90.2.175	gtm-server-805.fungible.local	
6	gtm-server-806.fungible.local	10.90.2.176	gtm-server-806.fungible.local	
7	gtm-server-807.fungible.local	10.90.2.177	gtm-server-807.fungible.local	
8	gtm-server-808.fungible.local	10.90.2.178	gtm-server-808.fungible.local	
9	gtm-server-809.fungible.local	10.90.2.179	gtm-server-809.fungible.local	
10	gtm-server-810.fungible.local	10.90.2.180	gtm-server-810.fungible.local	

Table 5: GPFS Cluster Information

```

%nsd: device=/dev/nvme0n1 nsd=nsd1 usage=dataAndMetadata pool=system thinDiskType=auto
%nsd: device=/dev/nvme12n1 nsd=nsd2 usage=dataAndMetadata pool=system thinDiskType=auto
%nsd: device=/dev/nvme13n1 nsd=nsd3 usage=dataAndMetadata pool=system thinDiskType=auto
%nsd: device=/dev/nvme15n1 nsd=nsd4 usage=dataAndMetadata pool=system thinDiskType=auto
%nsd: device=/dev/nvme16n1 nsd=nsd5 usage=dataAndMetadata pool=system thinDiskType=auto
%nsd: device=/dev/nvme18n1 nsd=nsd6 usage=dataAndMetadata pool=system thinDiskType=auto
...
%nsd: device=/dev/nvme63n1 nsd=nsd32 usage=dataAndMetadata pool=system thinDiskType=auto
%nsd: device=/dev/nvme67n1 nsd=nsd33 usage=dataAndMetadata pool=system thinDiskType=auto
%nsd: device=/dev/nvme6n1 nsd=nsd34 usage=dataAndMetadata pool=system thinDiskType=auto
%nsd: device=/dev/nvme7n1 nsd=nsd35 usage=dataAndMetadata pool=system thinDiskType=auto
%nsd: device=/dev/nvme8n1 nsd=nsd36 usage=dataAndMetadata pool=system thinDiskType=auto
    
```

Table 6: NSD Device Files

File system	Disk name	NSD servers
gpfs01	nsd1	(directly attached)
gpfs01	nsd2	(directly attached)
gpfs01	nsd3	(directly attached)
gpfs01	nsd4	(directly attached)
gpfs01	nsd5	(directly attached)
gpfs01	nsd6	(directly attached)
gpfs01	nsd7	(directly attached)
gpfs01	nsd8	(directly attached)
gpfs01	nsd9	(directly attached)
gpfs01	nsd10	(directly attached)
gpfs01	nsd11	(directly attached)
gpfs01	nsd12	(directly attached)
gpfs01	nsd13	(directly attached)
gpfs01	nsd14	(directly attached)
gpfs01	nsd15	(directly attached)
gpfs01	nsd16	(directly attached)
gpfs01	nsd17	(directly attached)
gpfs01	nsd18	(directly attached)
gpfs01	nsd19	(directly attached)
gpfs01	nsd20	(directly attached)
gpfs01	nsd21	(directly attached)
...		
gpfs01	nsd30	(directly attached)
gpfs01	nsd31	(directly attached)
gpfs01	nsd32	(directly attached)
gpfs01	nsd33	(directly attached)
gpfs01	nsd34	(directly attached)
gpfs01	nsd35	(directly attached)
gpfs01	nsd36	(directly attached)

Table 7: NSD List

flag	value	description
-f	819	Minimum fragment (subblock) size in bytes
-i	4096	Node size in bytes
-l	32768	Indirect block size in bytes
-m	1	Default number of metadata replicas
-M	1	Maximum number of metadata replicas
-r	1	Default number of data replicas
-R	1	Maximum number of data replicas
-j	cluster	Block allocation type
-D	nfs4	File locking semantics in effect
-k	all	ACL semantics in effect
-n	3	Estimated number of nodes that will mount file system
-B	1048576	Block size
-Q	none	Quotas accounting enabled
	none	Quotas enforced
	none	Default quotas enabled
--perfileset-quota	no	Per-fileset quota enforcement
--filesetdf	no	Fileset df enabled?
-V	23.00(5.0.5.0)	File system version
--create-time	Mon Oct 19 14:04:40 2020	File system creation time
-z	no	Is DMAP1 enabled?
-L	33554432	Logfile size
-E	yes	Exact mtime mount option
-S	relatime	Suppress atime mount option
-K	whenpossible	Strict replica allocation option
--fastea	yes	Fast external attributes enabled?
--encryption	no	Encryption enabled?
--inode-limit	29491456	Maximum number of inodes
--log-replicas	0	Number of log replicas
--is4KAligned	yes	is4KAligned?
--rapid-repair	yes	rapidRepair enabled?
--write-cache-threshold	0	HAWC Threshold (max 65536)
--subblocks-per-full-block	128	Number of subblocks per full block
-P	system	Disk storage pools in file system
--file-audit-log	no	File Audit Logging enabled?
--maintenance-mode	no	Maintenance Mode enabled?
-d	nsd1 ; nsd2 ; nsd3 ; nsd4 ; nsd5 ; nsd6 ; nsd7 ; nsd8 ; nsd9 ; nsd10 ; nsd11 ; nsd12 ; nsd13 ; nsd14 ; nsd15 ; nsd16 ; nsd17 ; nsd18 ; nsd19 ; nsd20 ; nsd21 ; nsd22 ; nsd23 ; nsd24 ; nsd25 ; nsd26 ; nsd27 ; nsd28 ; nsd29 ;	Disks in file system
-d	nsd30 ; nsd31 ; nsd32 ; nsd33 ; nsd34 ; nsd35 ; nsd36	Disks in file system
-A	yes	Automatic mount option
-o	none	Additional mount options
-t	/mnt/gpfs01	Default mount point
--mount-priority	0	Mount priority

Table 8: GPFS Filesystem Information

disk name	disk size in KB	failure group	holds metadata	holds data	free in KB in full blocks	free in KB in fragments
Disks in storage pool: system (Maximum disk size allowed is 32.94 TB)						
nsd1	838860800	-1	yes	yes	815367168 (97%)	9816 (0%)
nsd2	838860800	-1	yes	yes	815369216 (97%)	9816 (0%)
nsd3	838860800	-1	yes	yes	815370240 (97%)	9816 (0%)
nsd4	838860800	-1	yes	yes	815367168 (97%)	9808 (0%)
nsd5	838860800	-1	yes	yes	815369216 (97%)	9496 (0%)
nsd6	838860800	-1	yes	yes	815366144 (97%)	9752 (0%)
...						
nsd29	838860800	-1	yes	yes	815369216 (97%)	9752 (0%)
nsd30	838860800	-1	yes	yes	815371264 (97%)	9752 (0%)
nsd31	838860800	-1	yes	yes	815371264 (97%)	9784 (0%)
nsd32	838860800	-1	yes	yes	815370240 (97%)	9752 (0%)
nsd33	838860800	-1	yes	yes	815370240 (97%)	9752 (0%)
nsd34	838860800	-1	yes	yes	815367168 (97%)	9816 (0%)
nsd35	838860800	-1	yes	yes	815364096 (97%)	9816 (0%)
nsd36	838860800	-1	yes	yes	815367168 (97%)	9816 (0%)
(pool total)	30198988800				29353241600 (97%)	351608 (0%)
(total)	30198988800				29353241600 (97%)	351608 (0%)
Inode Information						
Number of used inodes:			4047			
Number of free inodes:			502833			
Number of allocated inodes:			506880			
Maximum number of inodes:			29491456			

Table 9: GPFS Storage Pool Information

## BENCHMARK RESULTS

The graph below shows the measured results of gpfsperf and FIO running random reads with 4K block size. What we noticed is that a single GPFS node can deliver a maximum of ~800,000 to ~900,000 4K IOPS. In this test, we used only 10 nodes so the performance is right on par with what GPFS can deliver.

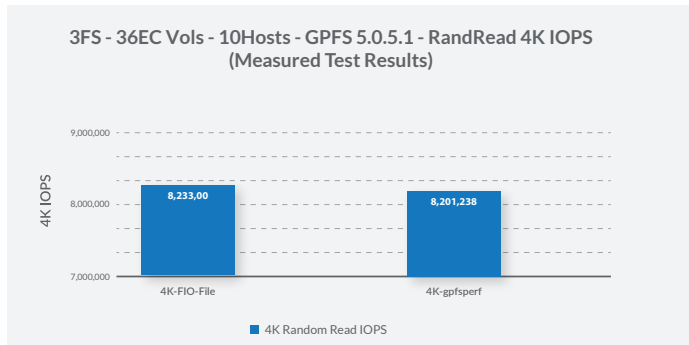
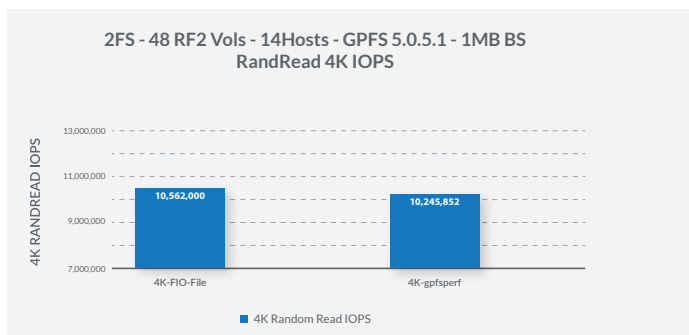


Figure 2: gpfsperf & FIO Results

We also tested the FSC's replicated volumes feature (RF=2) by creating 48 RF2 volumes across 2 FS1600 and attached those 48 volumes to 14 GPFS nodes. Similar to the Erasure Coded volumes, we created a single GPFS file system and ran the gpfsperf tool on all 14 GPFS nodes. Below is the graph that shows the results of gpfsperf and FIO.



## KEY TAKEAWAY

The Fungible FSC is a scale-out storage platform that provides unprecedented performance on IBM Spectrum Scale. Measured on a IOPS / PB basis, **FSC delivers 87 M IOPS / PB**. In addition, you can offload the computational intensive work such as compression, erasure coding and encryption to the FSC with little performance hit, thus freeing up CPU resources on the server to process additional workloads. The FSC tests have proven it to be the fastest disaggregated storage for IBM Spectrum Scale, achieving significantly higher IOPS performance than any other storage system.

Whether you need high performance for your analytic workloads or database workloads, rest assured the FSC can meet your requirement expectations because it is a scale-out platform. Multiple workloads such as GPFS clusters, RDBMS databases or NoSQL database clusters (Cassandra, CockroachDB or MongoDB just to name a few), can use the same FSC. Since the FSC uses erasure coding or replications across appliances or nodes, data is protected even for rack failures.

The figure below shows a sample FSC setup which can be used for multiple workloads.

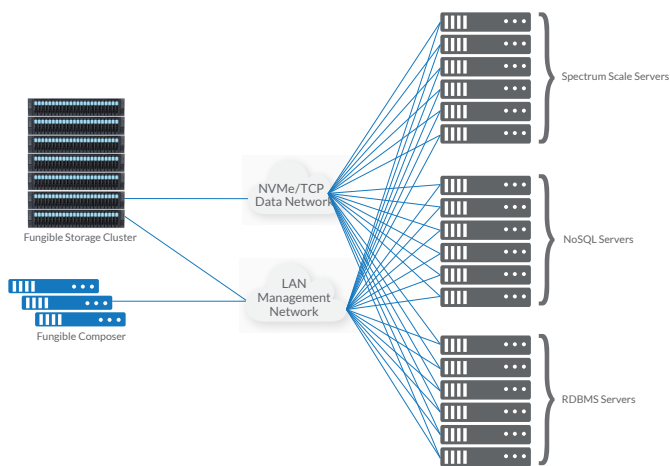


Figure 3: Sample FSC Configuration for Multiple Workloads

## CONCLUSION

This whitepaper showed that running IBM Spectrum Scale on the Fungible Storage Cluster would give you a highly performant GPFS cluster. The Fungible Storage Cluster provides all the benefits of pooled storage with outstanding GPFS performance.

## ABOUT FUNGIBLE

Silicon Valley-based Fungible is reimagining the performance, economics, reliability, security and agility of today's data centers.

## CONTACT US

sales@fungible.com

FUNGIBLE, INC.

3201 Scott Blvd., Santa Clara, CA 95054, USA  
669-292-5522

www.fungible.com | [in](#) [▶](#) [🐦](#) [✉](#)