TABLE OF CONTENTS

WHITE PAPER

RED HAT GLUSTER STORAGE ON SUPERMICRO STORAGE SERVERS POWERED BY INTEL® XEON® PROCESSORS

A Performance And Sizing Guide

EXECUTIVE SUMMARY

Red Hat® Gluster Storage has emerged as a compelling platform for software-defined scale-out distributed file services in the enterprise. Those deploying Gluster can benefit from simple cluster configurations optimized for different workloads that match their application and service needs. At the same time, organizations often require guidance to help them select from appropriate hardware implementations, volume configurations, data protection schemes, caching approaches, and access methods. To address the need for performance and sizing guidance,

Red Hat and Supermicro have performed extensive testing to characterize optimized configurations for deploying Red Hat Gluster Storage on Supermicro storage servers.

Super Micro Computer, Inc.
980 Rock Avenue
San Jose, CA 95131 USA
www.supermicro.com

INTRODUCTION

A rapidly changing application and services landscape and dramatically escalating needs for scalable distributed file storage are fundamentally changing the ways that storage is designed and deployed. After years of inflexible proprietary hardware- and appliance-based approaches, software-defined storage solutions have emerged as a viable alternative. The success of web-scale and public cloud providers has proven the way forward, with many embracing open software-defined storage as a fundamental strategy for deploying flexible, elastic, and cost-effective storage that is matched to specific application needs.

While the success of software-defined storage for web-scale firms is inspirational, most organizations lack the considerable in-house resources required to develop, prototype, deploy, and support effective solutions on their own. Open community-driven software-defined platforms can help, especially when combined with Red Hat's proven storage evaluation methodology. In combination with key strategic partners, Red Hat Gluster Storage can help reduce the cost and risk of deploying software-defined storage. This unique approach helps organizations answer key questions, including:

- How well do software-defined storage solutions scale under various workloads?
- How do storage server platform choices affect performance and cost effectiveness?
- How do choices such as Gluster volume types, data protection methods, client types, and tiering affect performance for key workloads?
- How important are architectural trade-offs for serving large files for throughput, compared to serving smaller files for I/O operations?
- How can solid-state storage impact performance for key workloads?
- How well does software-defined storage perform when compared to manufacturer's baseline specifications and basic I/O testing?

Working closely with Supermicro, Red Hat has conducted extensive evaluation and testing of various workloads with Red Hat Gluster Storage on several storage server configurations. Red Hat's robust evaluation methodology involves building and thoroughly testing various permutations of properly-sized Gluster storage pools. Organizations can then more quickly select configurations and solutions that closely match the needs of their specific workloads and applications. In addition to foundational I/O testing, Red Hat's recent testing also evaluated application simulation testing using SPEC Solution File Server (SFS) 2014 Video Data Acquisition (VDA) benchmark.*

* The SPEC SFS 2014 VDA workload evaluates live streaming high-definition (HD) video capture for sequential write-heavy workloads with latency sensitivity along with random concurrent read workloads. See spec.org for results.

RED HAT GLUSTER STORAGE ON SUPERMICRO SERVERS

Red Hat's testing and the resulting reference architecture evaluated the combination of Red Hat Gluster Storage on both standard and dense Supermicro storage servers across a variety of workloads using a range of volume configurations and file sizes. The testing resulted in configuration recommendations for workload-optimized storage pools.

RED HAT GLUSTER STORAGE

Red Hat Gluster Storage is a software-defined, open source solution that is designed to meet unstructured and semi-structured data storage requirements, with particular strengths in storing digital media files (images, video, and audio). At the heart of Red Hat Gluster Storage is an open source, massively scalable distributed file system (DFS) that allows organizations to combine large numbers of storage and compute resources into a high-performance, virtualized, and centrally managed storage pool (Figure 1). The cluster can be scaled for increased capacity, increased performance, or both. Red Hat Gluster Storage was designed to achieve several major goals, including:

- Elasticity. With Red Hat Gluster Storage, storage volumes are abstracted from the hardware and managed independently. Volumes can grow or shrink by adding or removing systems from the storage pool. Even as volumes change, data remains available without application interruption.
- Petabyte scalability. Modern organizations demand scalability from terabytes to multiple petabytes. Red Hat Gluster Storage lets organizations start small and grow to support multi-petabyte repositories as needed. Those that need very large amounts of storage can deploy massive scale-out storage cost efficiently.
- High performance. Red Hat Gluster Storage provides fast file access by eliminating the typical centralized metadata server. Files are spread evenly throughout the system, eliminating hot spots, I/O bottlenecks, and high latency. Organizations can use enterprise disk drives and 10+ Gigabit Ethernet (GbE) to maximize performance.
- Reliability and high availability. Red Hat Gluster Storage provides automatic replication that helps ensure high levels of data protection and resiliency. For organizations that are disk space conscious and would like integrated data protection without replication or RAID 6, Gluster also supports erasure coding (through dispersed volumes) to maximize price/capacity. In addition to protecting from hardware failures, self-healing capabilities restore data to the correct state following recovery.
- Industry-standard compatibility. For any storage system to be useful, it must support a broad range of file formats. Red Hat Gluster Storage provides native POSIX file system compatibility as well as support for common protocols including CIFS/SMB, NFS, and OpenStack® Swift. The software is readily supported by off-the-shelf storage management software.
- Unified global namespace. Red Hat Gluster Storage aggregates disk and memory resources into a single common pool. This flexible approach simplifies management of the storage environment and eliminates data silos. Global namespaces may be grown and shrunk dynamically, without client access interruption.
- Persistent storage for containerized applications. Containerized applications need

persistent storage. In addition to bare-metal, virtualized, and cloud environments, Red Hat Gluster Storage can be deployed as a container, allowing storage to be deployed along with containerized applications.

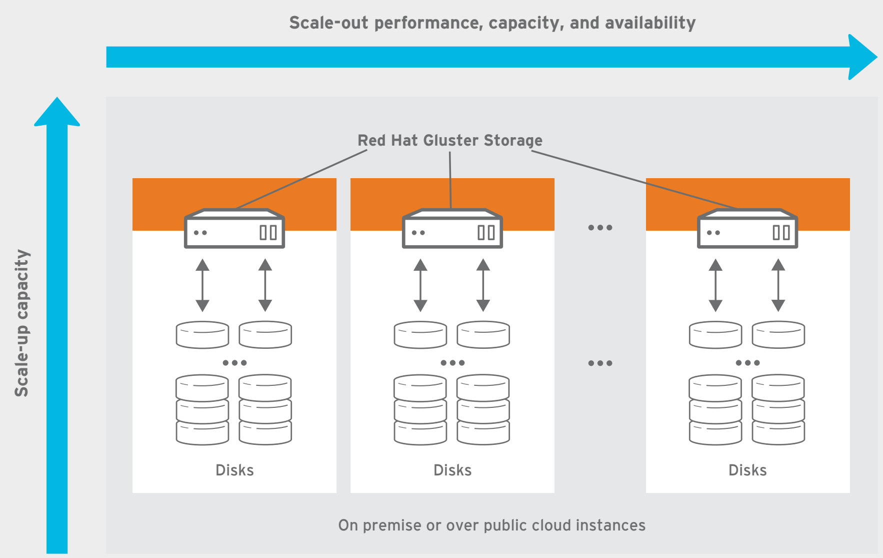

Scale-out performance, capacity, and availability
Scale-up capacity
Red Hat Gluster Storage
Disks
Disks
Disks
On premise or over public cloud instances

Figure 1. *Figure 1. Red Hat Gluster Storage combines server and storage resources in a centrally managed pool with independent capacity and performance scalability.*

From a technical perspective, Red Hat Gluster Storage provides distinct advantages over other technologies, with features that include:

- Software-defined storage. According to Red Hat, storage is a software problem that cannot be solved by locking organizations into a particular storage hardware vendor or a particular hardware configuration. Instead, Red Hat Gluster Storage is designed to work with a wide variety of industry-standard storage, networking, and compute server solutions.
- Open source community. Red Hat believes that the best way to deliver functionality is by embracing the open source model. As a result, Red Hat users benefit from a worldwide community of thousands of developers who are constantly testing the product in a wide range of environments and workloads, providing continuous and unbiased feedback to other users.
- User space operation. Unlike traditional file systems, Red Hat Gluster Storage operates in user space, rather than kernel space. This innovation makes installing and upgrading Red Hat Gluster Storage significantly easier, and greatly simplifies development efforts since specialized kernel experience is not required.
- Modular, stackable architecture. Red Hat Gluster Storage is designed using a modular and stackable architecture approach. Configuring Red Hat Gluster Storage for highly specialized environments is a simple matter of including or excluding particular modules.
- Native POSIX data storage. With Red Hat Gluster Storage, data is stored on disk using

native POSIX files with various self-healing processes established for data at the Gluster level. As a result, the system is extremely resilient and there is no proprietary or closed format used for storing file data.

- No metadata server with the elastic hash algorithm. Unlike other storage systems with a distributed file system, Red Hat Gluster Storage does not create, store, or use a separate index of metadata on a central server. Instead, Red Hat Gluster Storage places and locates files algorithmically. The performance, availability, and stability advantages of this approach are significant, and in some cases produce dramatic improvements.

STANDARD AND DENSE SUPERMICRO SERVERS FOR GLUSTER

Supermicro storage servers employ advanced components available in workload-optimized form factors. These systems offer high storage density coupled with up to 96% power efficiency, as well as advantages in procurement and operational costs for deployments of all sizes. Supermicro storage servers optimized for Red Hat Gluster Storage infrastructure feature the latest
Intel Xeon processors, and offer:

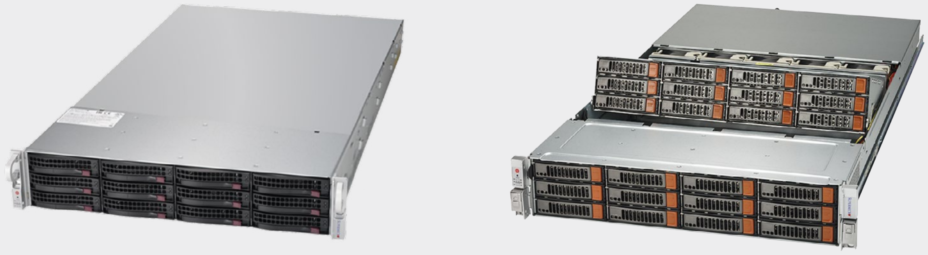
- Role-specific cluster configurations. Supermicro offers turn-key cluster configurations with performance, capacity, and density to fit popular application workloads. Memory and networking options are easily customized to meet specific requirements.
- Optimized network configurations. Cluster- and rack-level integration offers streamlined deployment of Red Hat Gluster Storage and infrastructure with consistency not attainable using improvised methods.
- Storage-to-media ratios to fit user applications. Specific combinations of flash and rotating magnetic media let Supermicro provide diverse solutions that meet workload-tuned performance and density targets.

Supermicro offers a range of storage servers, optimized for different types of workloads.* Two specific Supermicro storage servers were evaluated as a part of this study (Figure 2).

- Standard server. The 2 rack-unit (2U) Supermicro SuperStorage Server SSG-6028R-E1CR12H was used as the standard server. The system provides dual sockets for Intel Xeon Processor E5-2600 v4 processors, up to 2TB of memory, and 12 3.5-inch hot-swap SAS3/SATA3 drive bays. Up to four hot-swap NVMe drives are supported in optional hybrid bays.
- Dense server. The 2U Supermicro SuperStorage Server SSG-6028R-E1CR24N is also a dual-socket server, supporting up to two Intel Xeon Processor E5-2600 v4 processors and up to 3TB of memory. The dense server doubles the storage capacity of the standard server, providing 24 slots for 3.5-inch hot-swap SAS3/SATA3 drives through a dual storage backplane. Up to four hot-swap NVMe drives are supported in optional hybrid bays.

* supermicro.com/products/rack/scale-out_storage.cfm.

Details on actual tested Supermicro storage server configurations are provided elsewhere in this document.



Supermicro SuperStorage
Server SSG-6028R-E1CR12H
(standard server)

Supermicro SuperStorage
Server SSG-6028R-E1CR24N
(dense server)

Figure 2. *Figure 2. Red Hat tested both standard and dense Supermicro storage servers.*

WORKLOAD-OPTIMIZED DISTRIBUTED FILE SYSTEM CLUSTERS

One of the benefits of GlusterFS is the ability to tailor storage infrastructure to different workloads. Red Hat Gluster Storage on Supermicro servers can be optimized and sized to serve specific workloads and I/O patterns through a flexible choice of systems and components. Multiple combinations are possible by varying the density of the server (standard or dense storage servers), the layout of the underlying storage (RAID 6 or JBOD), the data protection scheme (replication or dispersed), and the storage architecture (standalone or tiered storage).

- Replicated volumes on RAID 6 bricks are commonly used for performance-optimized configurations, especially for workloads with smaller file sizes.
- Dispersed volumes (also known as erasure coding) on JBOD bricks are often more cost effective for large-file archive situations. Among supported configurations, dispersed volumes can offer better read and write performance for workloads with large file sizes.
- Standard 12-disk servers are often more performant and cost effective for smaller clusters and all small-file applications, while dense 24-disk storage servers and larger are often more cost effective for larger clusters.
- Depending on multiple factors, caching and tiering with either standard SSDs or NVMe SSDs installed in storage servers can provide significant benefits, especially for read performance.

TEST RESULTS SUMMARY

Red Hat tested a series of supported configurations, file sizes, client types, and client worker counts in order to determine the aggregate capabilities of a six-node Gluster pool for streaming file workloads. These values represent the peak capabilities of the pool under 100% write and 100% read conditions, and yielded the following observations:

- Large file workloads. Red Hat testing showed that for most sequential workloads of large files, a Gluster dispersed (erasure coded) volume on standard-density 12-disk JBOD nodes offered the best throughput-per-disk efficiency.
- Large file read-heavy workloads. For certain read-heavy workloads, a Gluster distributed-replicated volume on high-density 24-disk RAID 6 bricks may offer a throughput efficiency and investment advantage.
- Small file workloads. Testing made clear that the Gluster replicated volume on standard-density 12-disk nodes with RAID 6 bricks is generally the best choice for smaller file workloads. Read throughput efficiency for this configuration was dramatically better than alternatives, while writes were roughly on-par across the standard server configurations.
- Native versus NFS clients. The Gluster native client was found to generally outperform NFS v3 across these workload tests at high levels of client concurrency when the storage system is operating at its peak capabilities. The NFS client may be preferred in some cases where files are smaller and writes can coalesce, or where lower levels of client concurrency apply. The NFS client could also be a preferred choice for many types of random access small file workloads (Note: This type of workload was not included in this study).

Table 1. *Gluster pool optimization criteria*

OPTIMIZATION CATEGORY	SMALL POOLS (250TB)	MEDIUM POOLS (1PB)
Small-file Performance	Standard storage servers 2 x replicated volumes RAID 6 bricks 1 x NVMe hot tier	
Large-file Performance	Standard storage servers 2 x replicated volumes RAID 6 bricks 1 x NVMe hot tier	Standard storage servers Dispersed volumes JBOD bricks
Large-file Archive (write mostly)	Standard storage servers dispersed volumes JBOD bricks	Dense storage servers dispersed volumes JBOD bricks

Table 1 provides general Red Hat Gluster Storage pool configuration recommendations based on Red Hat and Supermicro testing. These categories are provided as guidelines for hardware purchase and configuration decisions, and can be adjusted to satisfy unique workload blends of different operators. As the workload mix varies from organization to organization, actual hardware configurations chosen will vary.

GLUSTER OVERVIEW AND RED HAT DESIGN PRINCIPLES

The sections that follow provide a high-level overview of both the Gluster architecture as well as
Red Hat's software-defined storage evaluation methodology.

GLUSTER DISTRIBUTED FILE SYSTEM OVERVIEW

Gluster distributed file system architecture provides built-in flexibility and a building block approach to constructing and configuring trusted storage pools customized to specific needs. A storage pool is a trusted network of storage servers. When the first server is started, the storage pool consists of that server alone. Additional storage servers are then added to the storage pool using the probe command from a storage server that is already trusted. The architecture presents a number of design choices for how volumes are implemented. For a more in-depth treatment of these concepts, see the Red Hat Gluster administration guide.*

Gluster volumes and bricks

Typically built from many Gluster bricks, Gluster volumes serve as mount points on Gluster clients. Bricks are the fundamental Gluster unit of storage—represented by an export directory on a server in the trusted storage pool and a glusterfsd process. Two types of bricks are supported:

- RAID 6 bricks aggregate read performance across multiple disks and also protect against the loss of up to two physical disks per server. They are most frequently used with 2x replicated volumes.
- JBOD bricks are particularly viable and cost-effective for large-capacity scenarios coupled with dispersed Gluster volumes for data protection.

Support for multiple client types

Individual clients can access volumes within the storage pool using a choice of protocols, including:

- Network File System (NFS).
- Common Internet File System (CIFS), an enhancement of Server Message Block (SMB).
- Gluster native client.
- OpenStack Swift.

GlusterFS volume types

Red Hat Gluster Storage supports several different types of volumes, allowing it to be tailored to fit the specific needs of individual applications.

* http://access.redhat.com/documentation/en/red-hat-gluster-storage/

- Distributed volumes. Distributed volumes spread files across the bricks in the volume, with individual files located on any brick in the distributed volume. Distributed volumes without replication or erasure-coding can suffer significant data loss during a disk or server failure.
- Replicated volumes. Replicated volumes create file copies across multiple bricks in the volume, replicating them between a number of servers to protect against disk and server failures. Replicated volumes are suitable for environments where high availability and high reliability are critical.
- Distributed-replicated volumes. Distributed-replicated volumes are suitable where there is a strong requirement to scale storage, yet high availability remains critical. Distributed-replicated volumes also offer improved read performance in most environments.
- Dispersed (erasure-coded) volumes. Dispersed volumes provide a method of data protection where data is broken into fragments, and then expanded and encoded with redundant data pieces and stored across a set of different locations. For many workloads, flash storage can maximize the performance of software-defined storage stacks, such as Red Hat Gluster Storage, when used as a cache or when used in Gluster tiering strategies.
- Distributed-dispersed volumes. Distributed-dispersed volumes offer scalability combined with the data efficiency of dispersed volumes.
- Tiered volume configurations. Tiering can provide better I/O performance since more active data is stored in a high-performance hot tier, with less-active data stored on a lower-performance cold tier. The hot tier is typically created using better-performing subvolumes—e.g., NVMe SSDs. Depending on the workload I/O pattern and the ratio of the working set to the full data set, tiering may or may not provide performance advantages.

SIX KEY RED HAT GLUSTER DESIGN PRINCIPLES

Red Hat's software-defined storage methodology includes answering important questions that can help organizations choose and shape Gluster implementations and network architectures. Each of the topics described in the sections that follow is intended to be a conversation between peer architects.

Qualifying the need for a software-defined, distributed filesystem

Not every storage situation calls for scale-out storage. The following requirements probably point to a good fit:

- Dynamic storage provisioning. By dynamically provisioning capacity from a pool of storage, organizations are typically building a private storage cloud, mimicking popular public cloud services.
- Standard storage servers. Scale-out storage employs storage clusters built from industry-standard x86 servers rather than proprietary storage appliances, allowing incremental growth of storage capacity and/or performance without forklift appliance upgrades.

- Unified name-spaces. Scale-out storage allows pooling storage across up to 128 storage servers in one or more unified name-spaces.
- High data availability. Scale-out storage provides high-availability of data across what would otherwise be storage silos within the storage cluster.
- Independent multidimensional scalability. Unlike typical NAS and SAN devices that may exhaust throughput before they run out of capacity, scale-out storage allows organizations to add storage performance or capacity incrementally by independently adding more storage servers or disks as required.

Designing for the target workload

Accommodating the target workload I/O profile is perhaps the most crucial design consideration. As a first approximation, organizations need to understand if they are simply deploying low-cost archive storage or if specific storage performance requirements need to be met. For performance-oriented clusters, throughput and latency requirements must be clearly defined, both per-client and as an aggregate. On the other hand, if the lowest cost per terabyte is the overriding need, a cluster architecture can be designed at dramatically lower costs. Additionally, understanding the workload read/write mix can affect architecture design decisions.

Choosing a storage access method

Choosing a storage access method is another important design consideration. As mentioned, Gluster supports a variety of client access methods. While the clients and applications will often dictate the storage access method—for instance, CIFS may be best suited to Windows clients—there are also options for multi-protocol access to the same namespaces that may improve relative efficiency of reads and writes.* Furthermore, particular workloads may benefit from specific client interfaces. Small static file workloads such as PHP (PHP: Hypertext Preprocessor) for web servers will benefit from NFS client-side caching, and larger dynamic file workloads may see greater aggregate throughput with the parallelization of the Gluster native client.

Identifying target storage capacity

Identifying target storage capacity may seem trivial, but it can have a distinct effect on the chosen target server architecture. In particular, storage capacity must be weighed in concert with considerations such as data protection method or fault domain risk tolerance. For example, if an organization is designing a small, quarter-petabyte cluster, minimum server fault-domain recommendations will preclude the use of ultra-dense storage servers in the architecture, to avoid unacceptable failure domain risk on a small number of very large nodes.

Selecting a data protection method

Gluster offers two data protection schemes: replicated volumes and dispersed volumes. As

* Due to locking incompatibility, CIFS/SMB cannot be used with other client access methods.

a design decision, choosing the data protection method can affect the solution's total cost of ownership (TCO) more than any other factor, while also playing a key role in determining cluster performance. The chosen data protection method strongly affects the amount of raw storage capacity that must be purchased to yield the desired amount of usable storage capacity and has particular performance trade-offs, depending on the workload. Figure 3 illustrates the capacity difference. In this example of a cluster of six 12-drive Supermicro SSG-6028R-E1CR12H servers, a Gluster distributed-dispersed volume (4+2 erasure coding) on JBOD bricks can yield 108TB more usable capacity than a distributed-replicated volume (2x replication) on RAID 6 bricks.

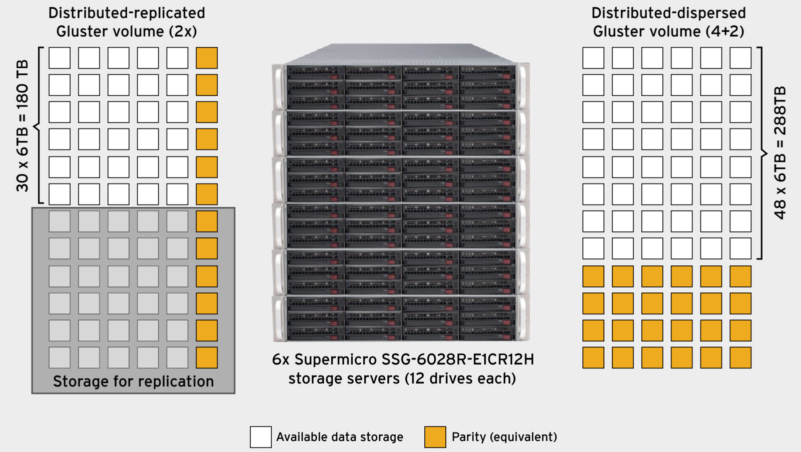

Distributed-replicated Gluster volume (2x)
30 x 6TB = 180 TB
Storage for replication
6x Supermicro SSG-6028R-E1CR12H storage servers (12 drives each)
Distributed-dispersed Gluster volume (4+2)
48 x 6TB = 288TB
□ Available data storage
🟧 Parity (equivalent)

Figure 3. *Figure 3. Dispersed Gluster volumes can yield significantly more usable storage than replicated volumes (space for parity depicted logically).*

Note: Effective parity for both erasure coding and RAID 6 is represented in the figure above with orange boxes for illustrative purposes only. The white boxes represent the effective amount of available storage after space for parity and replication are subtracted.

Determining fault domain risk tolerance

It may be tempting to deploy the largest servers possible in the interest of economics. However, production environments need to provide reliability and availability for the applications they serve, and these requirements extend to the scale-out storage upon which they depend. The fault domain represented by single server is key to cluster design. As such, dense servers should typically be reserved for clusters with more than a petabyte of raw capacity, where the capacity of an individual server accounts for less than 17% of the total cluster capacity. This fault domain recommendation may be relaxed for less critical pilot projects.

Fault domain risk includes accommodating the impact on performance from a drive or server failure. When a drive fails in a RAID 6 back-end volume, Gluster is unaware, as hardware RAID technology masks the failed drive until it can be replaced and the RAID volume re-built. In the case of a node or brick failure, Gluster self-healing will divert a percentage of volume throughput capacity to heal outdated file copies on the failed node from the file replicas on the surviving nodes. The percentage of performance degradation is a function of the number and size of files that changed on the failed node while it was down, and how Gluster is configured. If a node must be replaced, all file replicas assigned to this node must be copied from the surviving replica or reconstituted from the dispersed set.

Red Hat recommends these minimum pool sizes:

- Supported minimum pool size: Two nodes with a third non-storage node to constitute quorum.
- Recommended minimum pool size: Four nodes (distributed-replicated volumes) or six nodes (dispersed volumes).

TESTED CONFIGURATIONS AND METHODS

Red Hat testing exercised several cluster configurations using a variety of methods.

FOUNDATIONAL TESTING

Foundational tests were designed to determine peak aggregate performance of the Gluster pool under different volume geometries and different brick configurations. These tests were performed for 100% write and 100% read streaming file workloads using varying file sizes and worker counts, allowing isolation of the saturation point of the Gluster volume. The results represent best-case scenarios and performance targets to help evaluate real-world workloads. More information on IOzone and smallfile is provided in Appendix A.

WORKLOADS

File sizes were chosen to represent relatable use cases and to stress the system on a gradient—

from high-transaction, metadata-intensive tiny files to very large, throughput-sensitive files. Tested file sizes for sequential writes and reads were placed along a logarithmic line in order to aid interpolation of data between the test points (Figure 4). The smallest file size was selected just off of the log line to better represent a use case.

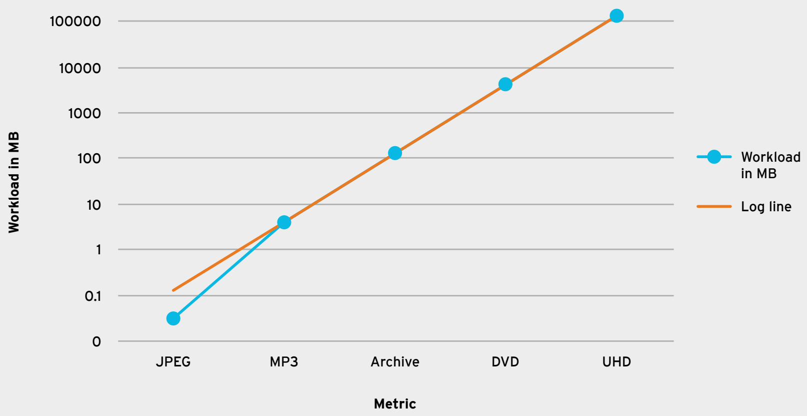


Figure 4. *Figure 4. Tested file sizes were placed along a logarithmic line to evaluate different use cases.*

Nested file sizes related to the following use cases:

- 32KB—JPEG or small document file
- 4MB—MP3 or large document file
- 128MB—Media archive or backup file
- 4GB—DVD video file
- 128GB—Ultra-high-definition (UHD) video file

For smaller files, particularly with a distributed storage system, both latency and throughput contribute to performance concerns. For this reason, the metric considered most relevant to consumers of the file storage system is files per second, which abstracts throughput and latency numbers together into a single unit of measure. Large file transactions, on the other hand, are less affected by latency and are therefore better measured with a focus on throughput (MB/s). A consistent large block size of 4MB was chosen for all throughput tests of 128MB files and above.

Because of these differences, Red Hat selected two open source tools to aid foundational benchmarks, IOzone and smallfile. In testing, client systems ran baseline (1, 6), and then exponentially-increasing workload threads (12, 24, 48, 96, and higher total threads of execution) against GlusterFS volumes on a six-node storage cluster.* This process identified the saturation point of the GlusterFS volume throughput for each workload. An example is given in Figure 5.

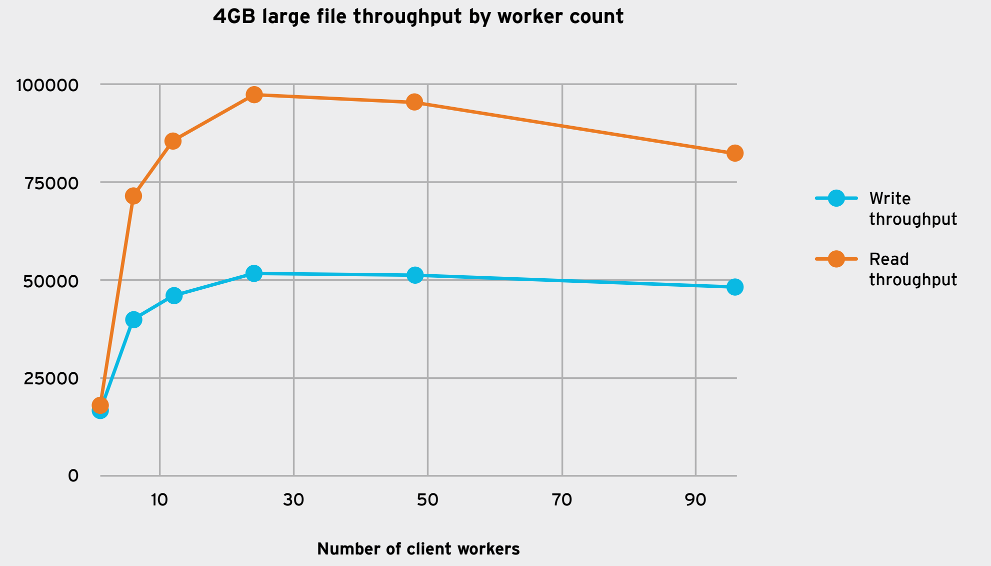


Figure 5. *Figure 5. 4GB large file throughput by worker count.*

EVALUATING FOR PERFORMANCE, EFFICIENCY, AND PRICE-PERFORMANCE

Rather than focusing purely on maximum throughput reporting, the ultimate testing goal was to evaluate the efficiency of a hardware investment against a variety of configurations and workloads. To achieve this goal, all numbers are calculated as total throughput divided by the number of disks in the storage solution, regardless of the RAID, replication, or erasure coding configuration. This approach allows performance to be related directly to the storage investment.

* More information on IOzone and smallfile is provided in Appendix A.

The model is then further refined by offering a price-performance metric that considers performance values alongside the hardware and software costs of the total solution.

APPLICATION SIMULATION

To further characterize the performance capabilities of a Gluster volume under real-world conditions, testing was performed using a benchmark tool that simulates streaming video capture and video analytic read workloads. The VDA workload from the SPEC SFS 2014 benchmark was used for this simulation. This workload provides a mix of write and read operations and defines tolerance for successful criteria based on a percentage of requests fulfilled within a specific time frame (latency). Results are reported in terms of streams, where a stream represents a single high-definition camera feed.

TESTED ARCHITECTURES

Two separate cluster configurations were constructed and tested by Red Hat and Supermicro.

Standard servers: Supermicro SuperStorage Server SSG-6028-E1CR12H

As shown in Figure 6, six Supermicro SuperStorage Server SSG-6028-E1CR2H were tested connected to a 10Gb Ethernet switch. The cluster was driven by three Supermicro SuperServer SYS-2027PR-HC0R, with each providing four client nodes for a total of 12. Each standard Supermicro storage server was configured with:

- Processors: 2 x Intel Xeon CPU E5-2630 v4 @2.20GHz, 10 cores hyperthreaded to 20,

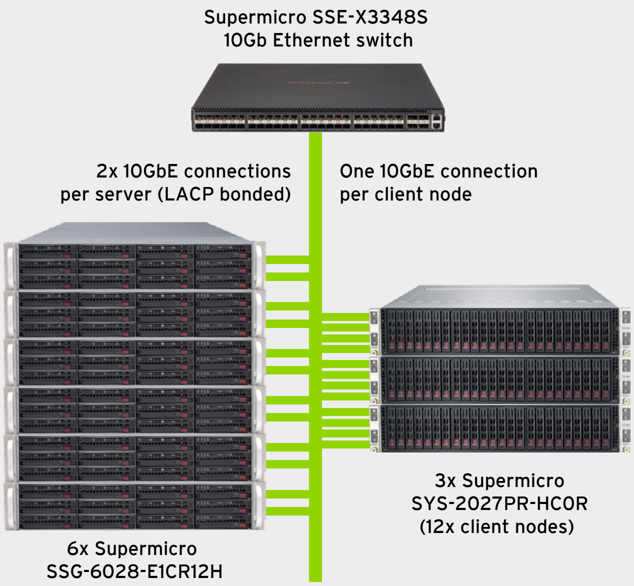


Figure 6. *Figure 6. Six standard Supermicro SuperStorage Server SSG-6028-E1CR12L were driven by 12 client nodes.*

40 total system logical cores

- Memory: 4 x 16GB DDR4 DIMMs, 64GB total system memory
- SAS controller: LSI SAS3 3108 RAID controller
- Onboard storage: 12 x Seagate ST6000NM0034 6TB 12Gb/s 7200 RPM SAS HDDs; 72TB raw storage per node
- Network controller: Intel X540 Dual Port 10GbE, LACP bonded for 20Gb active-active connection per node

Dense Servers: Supermicro SuperStorage Server SSG-6028-E1CR24N

Figure 7 illustrates a similar testing configuration using dense Supermicro SuperStorage Server SSG-6028-E1CR24N driven by 12 client nodes. Each dense Supermicro storage server was configured with:

- Processors: 2 x Intel Xeon CPU E5-2650 v4 @ 2.20GHz, 12 cores hyperthreaded to 24, 48 total system logical cores
- Memory: 8 x 16GB DDR4 DIMMs; 128GB total system memory
- SAS controller: LSI SAS3 3108 RAID controller
- Onboard storage: Dual 12-port SAS3 12G expander backplanes, each supporting 12 x (24 x total) Seagate ST6000NM0034 6TB 12Gb/s 7200 RPM SAS HDDs; 144TB raw storage per node
- Network controller: 2 x Intel X550T 10GbE interfaces onboard; management network; LACP bonded for 20Gb active-active connection per node

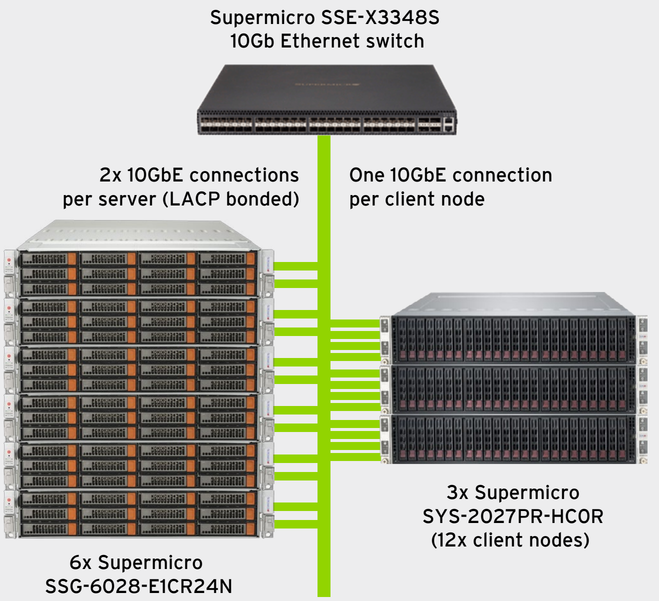


Figure 7. *Figure 7. Dense configurations featured six Supermicro SuperStorage SSG-6028-E1CR24N.*

Load Generating Clients

For all configurations, loads were generated on 12 client nodes provided by three Supermicro SuperServer SYS-2027PR-HC0R quad-server nodes. Each server node was configured with:

Processors: 2x Intel Xeon CPU E5-2670 v2 @ 2.50GHz, 10 cores hyperthreaded to 20, 40 total system logical cores

Memory: 4 x 16GB DDR4 DIMMs; 64GB total system memory

Network controller: 2 x Intel 82599ES 10GbE interfaces; single 10Gb uplink per node

SOFTWARE ENVIRONMENT

Software configurations for Red Hat testing were:

- Red Hat Enterprise Linux® 7.2
- Red Hat Gluster Storage 3.1.3

In addition, Red Hat Gluster Storage 3.2 was evaluated with a subset of the tests to analyze performance improvements. Results on this testing can be found in Appendix C.

GLUSTER VOLUME ARCHITECTURE

The disks in the systems were configured in both RAID 6 and JBOD modes and were combined with both replicated and dispersed Gluster volumes for data protection. The standard SSG-6028-E1CR12L nodes had a total disk count of 72 across six nodes and were configured in the following ways:

- RAID 6: distributed-replicated 3x2
- RAID 6: dispersed 1x(4+2)
- JBOD: distributed-dispersed 12x(4+2)

The dense SSG-6028-E1CR24N nodes had a total disk count of 144 across 6 nodes and were configured in the following ways:

- RAID 6: distributed-replicated 6x2
- RAID 6: distributed-dispersed 2x(4+2)
- JBOD: distributed-dispersed 24x(4+2)
- RAID 0, JBOD: distributed-dispersed 24x(4+2) (Note: used for the streaming video case study)

DESIGNING FOR THROUGHPUT

Large-file sequential I/O workloads are a common use case for Gluster volumes. Media repositories, data backup targets, log servers, and streaming video stores all may fall into this workload category. When optimizing a storage system for this use, file throughput—or how many bytes the storage system can receive or send per second—is the primary concern for performance.

Hardware choice has a large effect on the throughput and throughput efficiency of the storage system. Interestingly, when analyzing the HDD-based systems Red Hat testing showed that the standard-density 12-drive systems are by far the better choice, offering a 67% advantage for reads and an 82% advantage for writes (Figure 8).

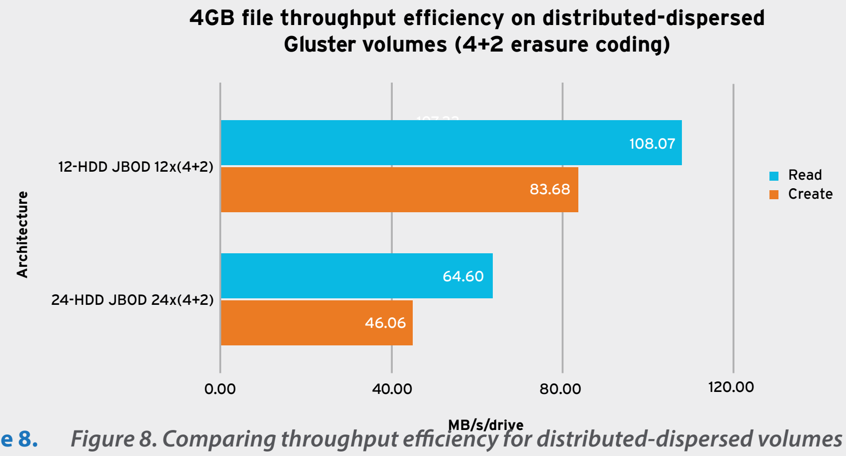


Figure 8. *Figure 8. Comparing throughput efficiency for distributed-dispersed volumes using 4GB files.*

For Gluster distributed-replicated volumes results again demonstrated that the standard-density 12-drive systems offer greater throughput efficiencies than their 24-drive siblings, an advantage of 21% for reads and 60% for writes (Figure 9).

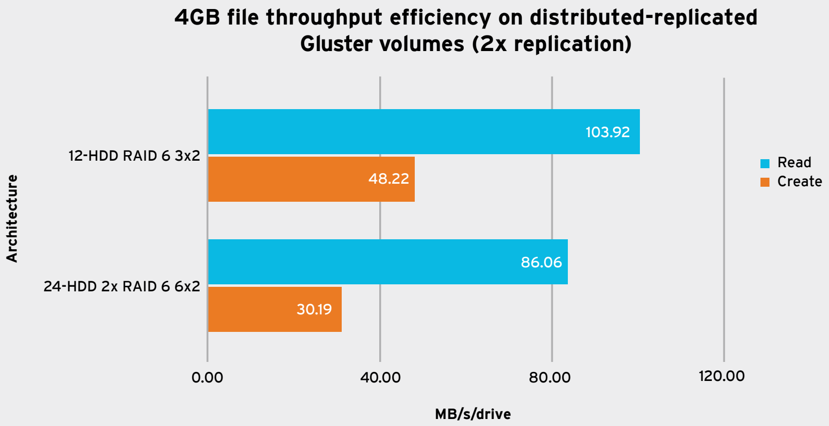


Figure 9. *Figure 9. Throughput efficiency for distributed-replicated volumes (2x replication).*

PRICE-PERFORMANCE ANALYSIS

As described, Red Hat analysis also included comparing the primary HDD test architectures for both standard and dense servers with a relative performance-per-cost metric. These calculations were made using the 4GB file test results with the Gluster native client.

As seen above, the standard-density 12-HDD servers offer a greater overall performance efficiency across configurations. We find that dense 24-HDD servers, on the other hand, offer a greater efficiency of scale per dollar. For replicated volumes, these efficiencies tend to cancel each other out, leaving a relatively flat price-performance comparison between the standard and dense nodes, with a slight write advantage for standard nodes and conversely a slight read advantage for dense nodes (Figure 10).

For dispersed volumes, the efficiency of scaling out is much more pronounced in the price-performance data. The standard 12-HDD servers offer greater price-performance returns than the dense 24-HDD servers, particularly for writes.

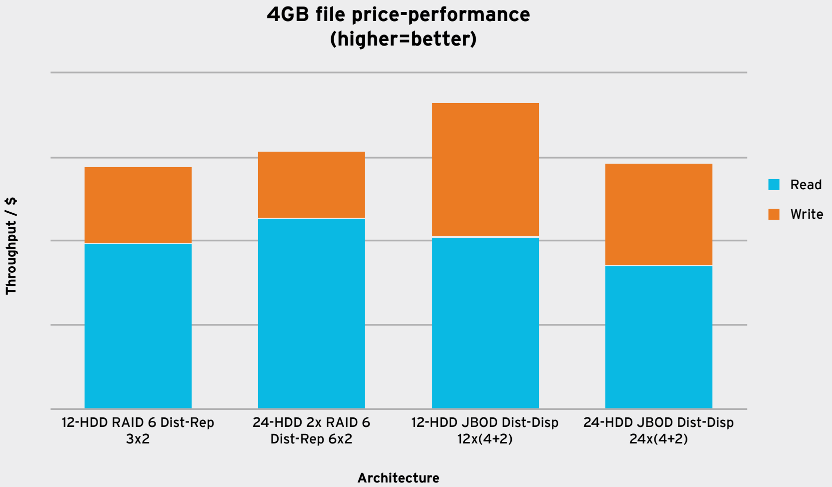


Figure 10. *Figure 10. 4GB file price-performance comparison across HDD-based servers in multiple volume configurations.*

DETAILED TEST RESULTS

With standard-density servers offering superior throughput efficiency for HDD-based systems, the following details analyze configurations of only those servers under different block device and Gluster volume configurations. Figures 11, 12, and 13 illustrate standard server test results for 128GB, 4GB, and 128MB files respectively on standard HDD-based servers with varying volume configurations and both Gluster native and NFS clients.

For these common large file sizes that might represent general media archives or DVD video files, Red Hat found consistently that dispersed volumes built on JBOD bricks offered a distinct advantage for write throughput efficiency—reporting up to 89% greater throughput than a replicated volume on RAID 6 bricks. The sweet spot for write throughput efficiency was found to be for files in the single-digit gigabyte range, where 4GB file tests topped out at 83.68 MB/s/drive for our standard 12-disk HDD server model in a 12x(4+2) distributed-dispersed volume on JBOD (Figure 12).

Read throughput efficiency appears to be less affected by the Gluster volume configuration, reporting similar results for both a dispersed volume on JBOD and a replicated volume on RAID 6. However, for some configurations and file sizes the replicated volume did report a 30-50% advantage over a dispersed volume. The top-performing HDD-based distributed-dispersed volume for writes again showed advantages for reads, reaching a peak of 108.07 MB/s/drive for 4GB files (Figure 12).

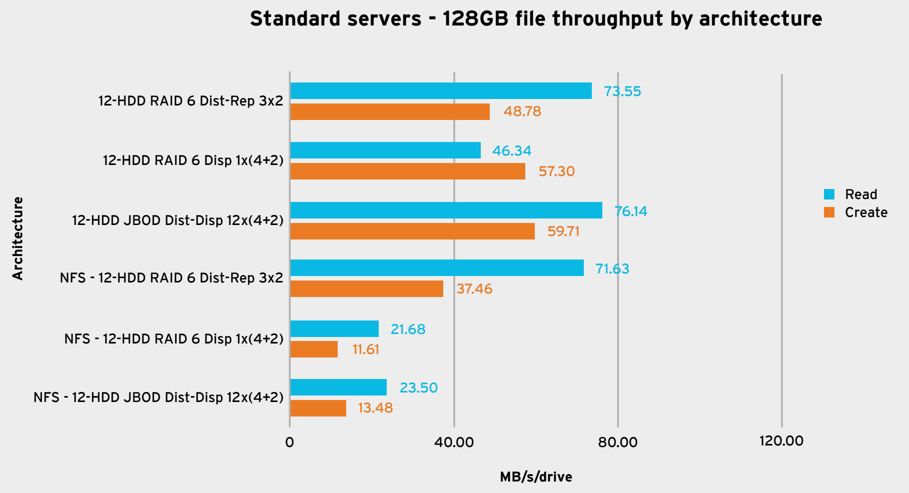


Figure 11. *Figure 11. Standard server throughput tests with large 128GB files across multiple volume configurations.*

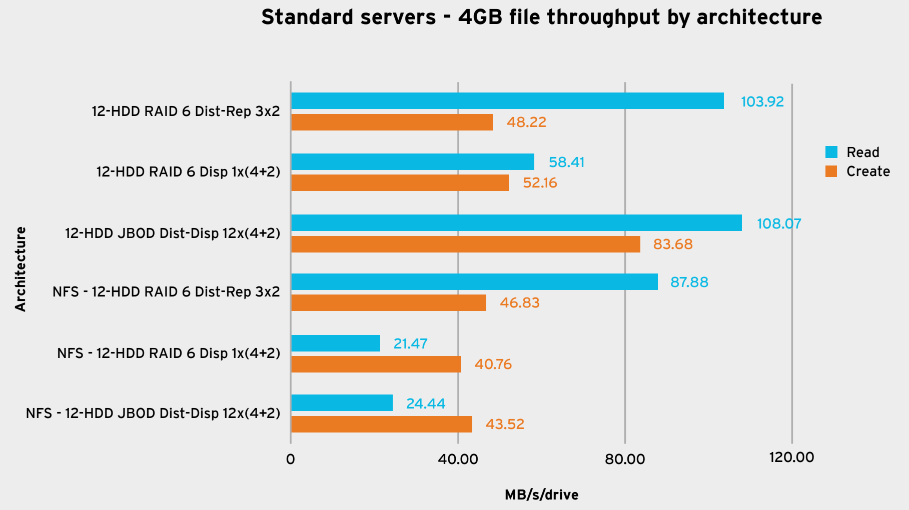


Figure 12. *Figure 12 Standard server throughput tests with 4GB files across multiple volume configurations.*

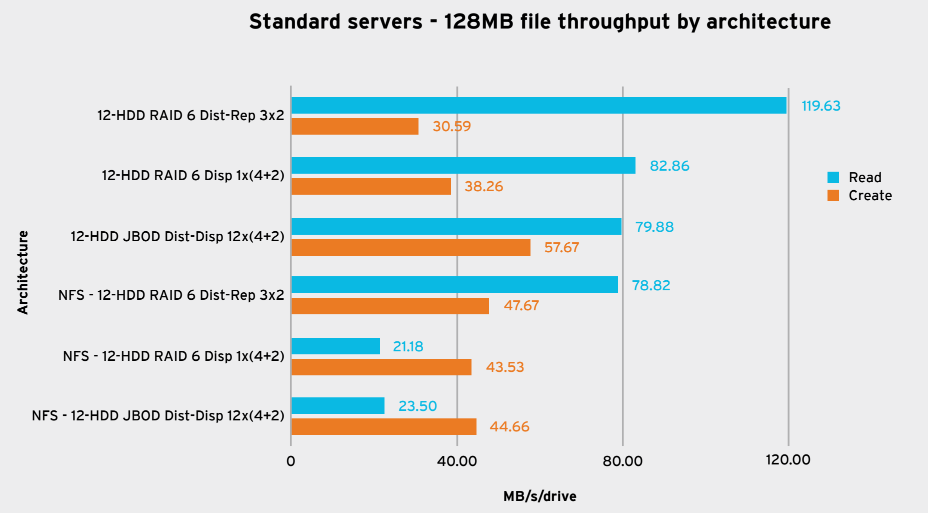


Figure 13. *Figure 13. Throughput tests with 128MB files across multiple volume configurations.*

CONCLUSIONS

Figures 14 and 15 show write and read throughput respectively across 128GB, 4GB, and 128MB files sizes for several volume configurations on standard servers. Based on Red Hat testing, dispersed volumes on JBOD disks offer a distinct advantage for write throughput with the Gluster native client, though that advantage becomes smaller as file sizes approach upper extremes, as visible in the 128GB file size tests.

Conversely, for reads, a lower bound of file size limits throughput. For 128MB medium

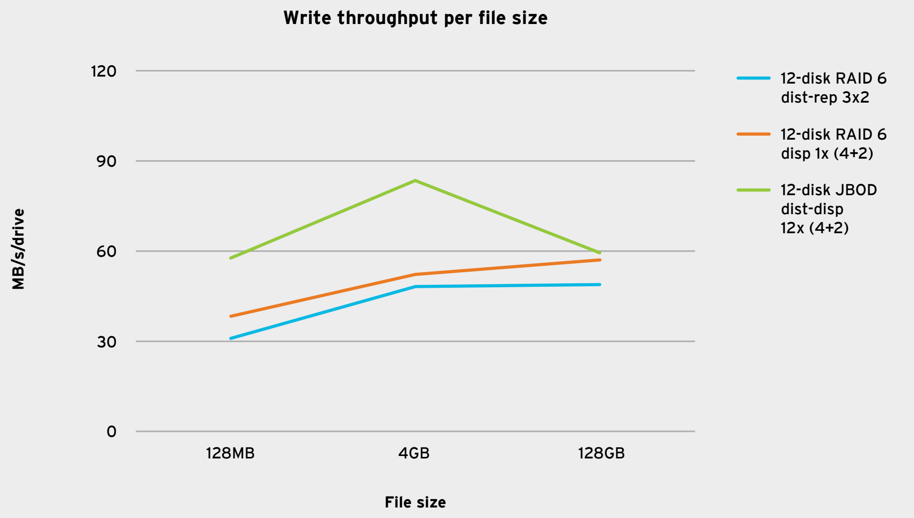


Figure 14. *Figure 14. Gluster native client write throughput per file size (standard server).*

files—the smallest file size in this test range—the dispersed JBOD volume under-performs notably when compared to the replicated volume on RAID 6. For small file sizes, the relative overhead of metadata operations for reads increases, adding significant latency to file operations and limiting available throughput. As the file sizes increase, however, throughput becomes statistically even with the replicated volume.

Overall, for file sizes in the gigabytes to tens-of-gigabytes range, distributed-dispersed volumes on JBOD disks consistently offer the greatest overall throughput capabilities.

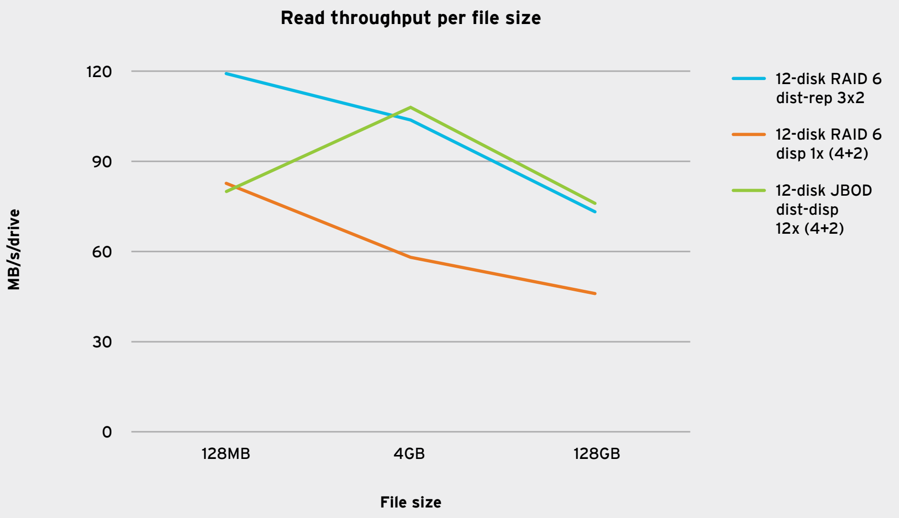


Figure 15. *Figure 15. Gluster native client read throughput per file size (standard server).*

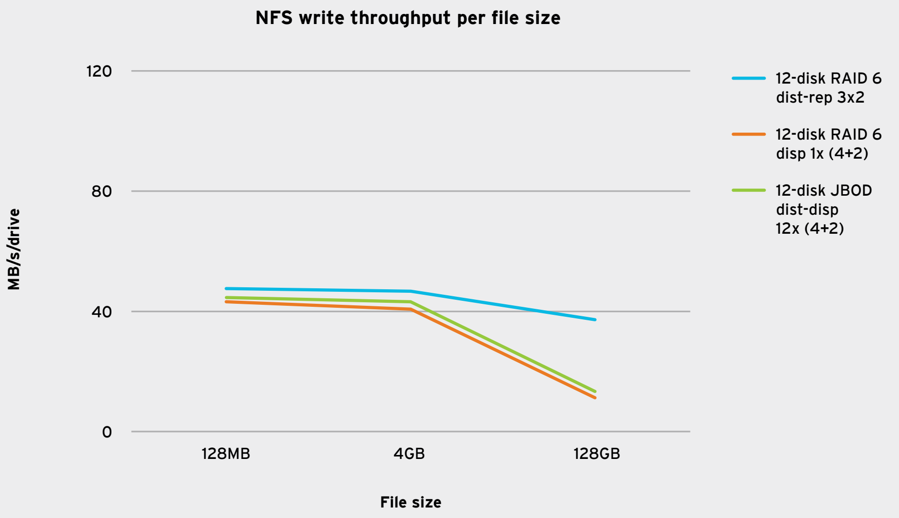


Figure 16. *Figure 16. Write throughput per file size using the NFS client and standard servers.*

As demonstrated, the NFS client under-performs overall compared to the Gluster native client for these larger file workloads. However, for cases where the NFS client may be required, we note that the replicated volume on RAID 6 architecture is advantageous for throughput, particularly for reads (Figures 16 and 17).

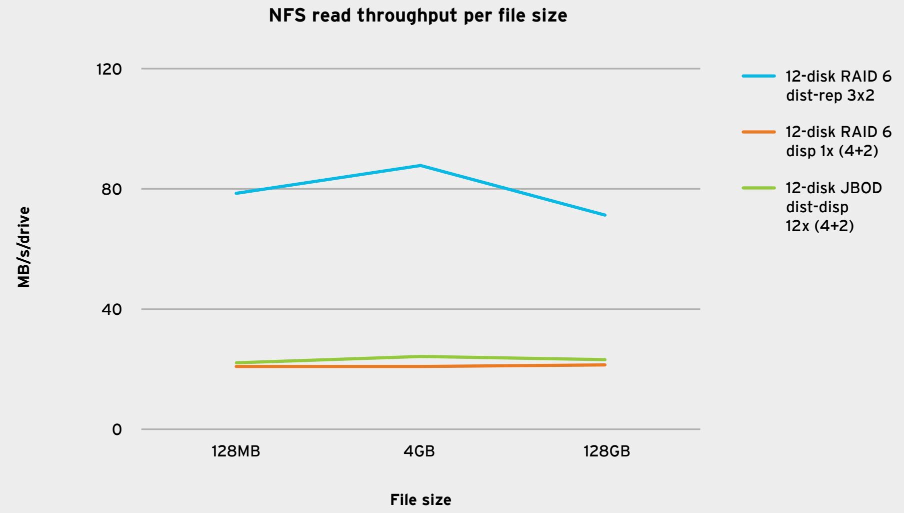


Figure 17. *Figure 17. Read throughput per file size using the NFS client and standard servers.*

DESIGNING FOR FILE OPERATIONS

Small-file workloads are characterized as being transaction-heavy, meaning that the overhead of the non-data portion of the file operation becomes a more significant part of the performance equation. MP3 files, JPEG files, or simple office documents may fall into this category of workload. Optimizing a storage system for small files requires that latency be considered directly alongside throughput. Rather than evaluate these metrics separately, Red Hat testing abstracted latency and throughput into a single files-per-second (files/s) metric.

As with the large file tests, the small file tests again revealed a greater throughput efficiency for 12-disk standard-density HDD nodes over the dense 24-disk HDD-based nodes. The 4MB file tests highlight this advantage, with the standard servers reporting both read advantage and write advantages for standard servers over dense servers. Figure 18 compares results for distributed-dispersed volumes on both standard and dense HDD-based servers.

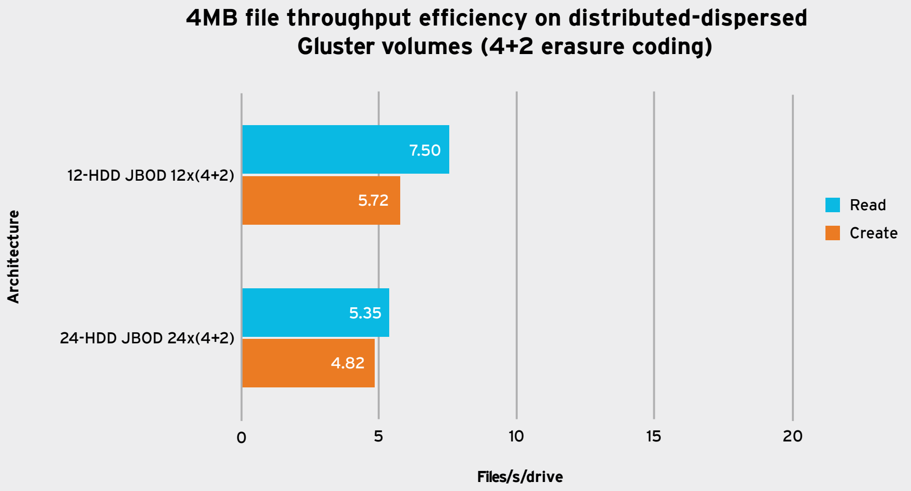


Figure 18. *Figure 18. Again, standard servers demonstrate better throughput efficiency for 4MB files on distributed-dispersed volumes on JBOD bricks.*

Figure 19 shows results for standard and dense servers for distributed-replicated volumes on RAID 6 bricks. While the read tests on dense nodes confirmed earlier findings that a distributed replicated volume on RAID 6 is better suited to these small file workloads than a distributed dispersed volume, testing also revealed a significant write throughput restriction for 4MB files on this configuration, yielding only 1.53 files/s/drive.

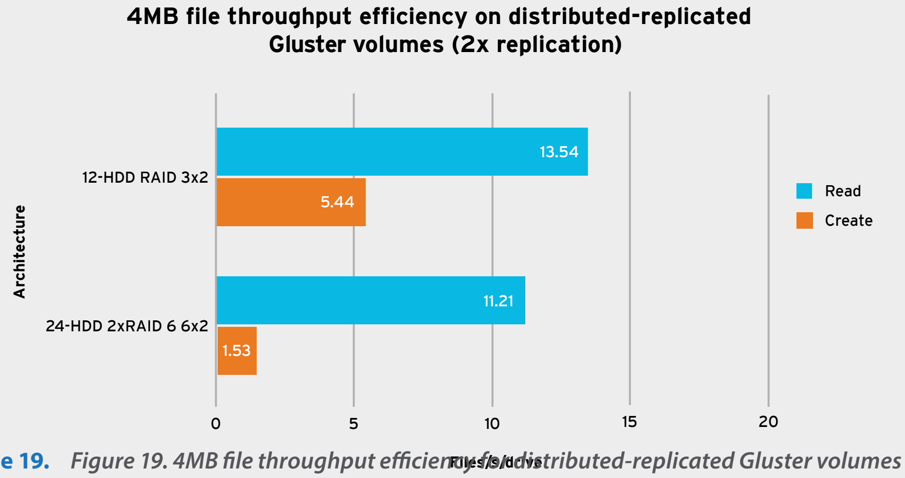


Figure 19. *Figure 19. 4MB file throughput efficiency for distributed-replicated Gluster volumes on RAID 6 bricks.*

Figure 20 illustrates 32KB tests across standard and dense servers with distributed-replicated volumes on RAID 6 bricks.

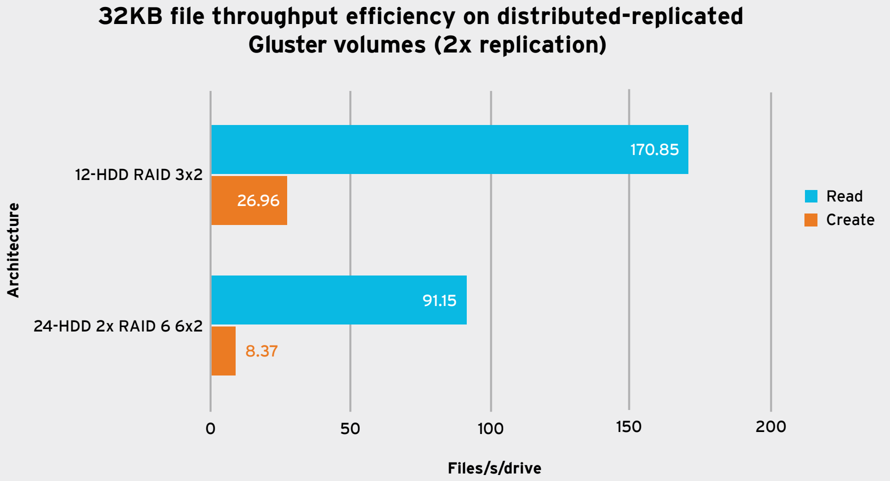


Figure 20. *Figure 20. 32KB throughput efficiency for distributed-replicated Gluster volumes on RAID 6 bricks.*

As the charts above demonstrate, it appears that the smaller the file size, the greater the delta between read and write throughput for the distributed-replicated volume on RAID 6 bricks

(Figure 21). For a distributed-dispersed volume, however, the delta between read and write throughput stays consistent regardless of the file size.

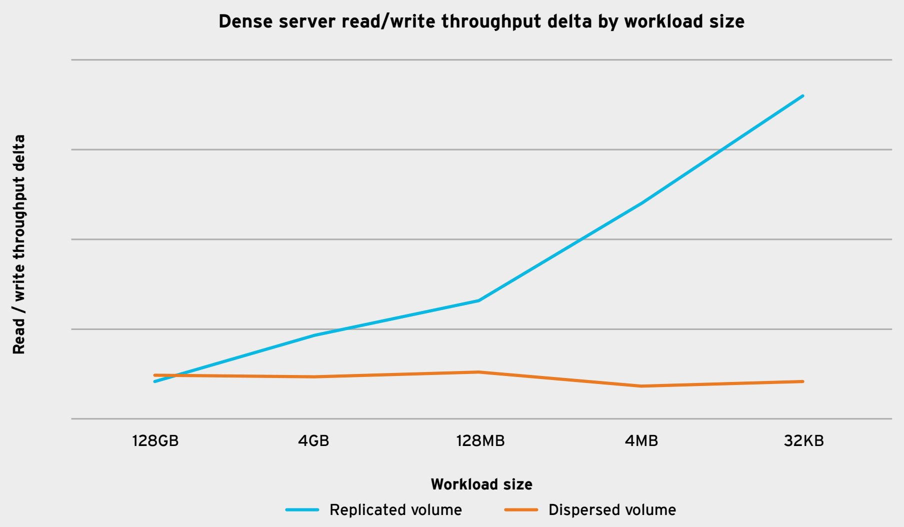


Figure 21. *Figure 21. The smaller the file size (left to right), the larger the delta between read and write throughput for replicated volumes on dense servers.*

PRICE-PERFORMANCE ANALYSIS

Figure 22 compares the relative small file throughput-per-cost for standard, dense, and high-performance server configurations. These comparisons illustrate 4MB file test results with the Gluster native client, adjusted for equivalent capacity. It is clear from this data that a replicated volume on RAID 6 bricks makes the most efficient use of disk investments for read workloads. Dense servers showed a slight advantage for read-heavy workloads, but

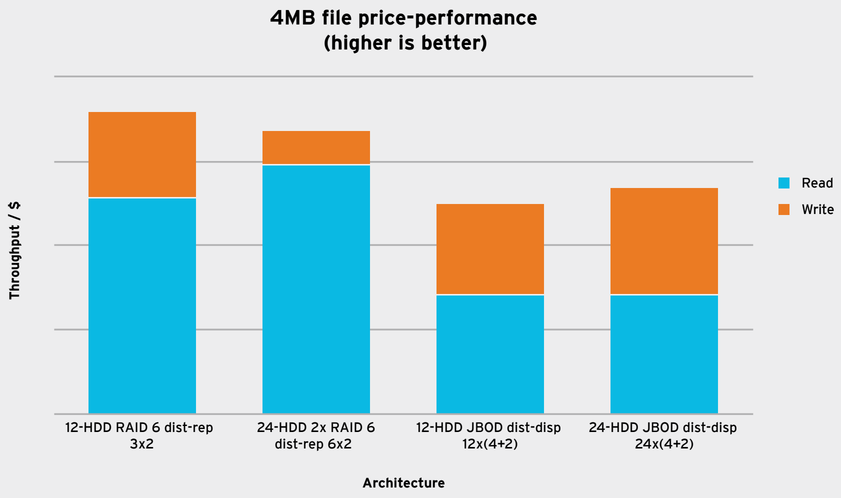


Figure 22. *Figure 22. A replicated volume on RAID 6 makes the most efficient use of disk investments, with the standard server providing the most balanced performance (higher is better).*

the limitations of write scalability on dense servers generally suggest that the standard-density server will be the better choice for most use cases.

Because very small sub-megabyte files introduce unique challenges, Red Hat additionally analyzed the price-performance metrics of these files under the same HDD system configurations. Here the limitations of dispersed volumes for small files become very pronounced, making it clear that a replicated volume is the right choice for small-file workloads. As shown in Figure 23, it becomes immediately clear that scaling out with standard-density 12-HDD systems will be a much more cost efficient choice that scaling up with high-density 24-HDD systems.

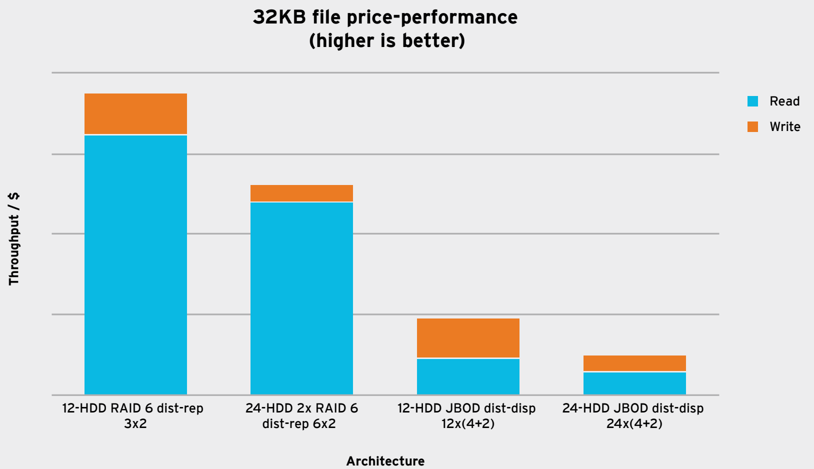


Figure 23. *Figure 23. For 32KB small-file workloads, scaling out with standard servers on distributed-replicated volumes presents clear price-performance advantages.*

DETAILED TEST RESULTS

For small file sizes, the round-trip of each transaction becomes a significant percentage of the performance calculation. Because dispersed Gluster volumes involve each file being split into chunks with parity, the latency of each file operation can be compounded by the scale of the dispersed volume and the calculation overhead. Red Hat testing found accordingly that Gluster replicated volumes on RAID 6 offer the best performance efficiency and price-performance.

For example, the 4MB file tests achieved 13.54 files/s/drive reads on a distributed-replicated geometry, 81% better than the 7.5 files/s/drive achieved on a distributed-dispersed geometry on JBOD bricks (Figure 24). Writes for small files demonstrated the same performance regardless of the architecture that was used. For the 4MB file size, testing showed 5.44 files/s/drive on the replicated volume on JBOD bricks architecture.

Testing was performed using both the Gluster native client and the NFS client.

For 32KB files, a more pronounced difference is seen across the board between read and write performance. As shown in Figure 25, standard servers with distributed-replicated volumes on RAID 6 bricks retain a significant advantage. The NFS client provides additional read performance at the expense of write performance.

CONCLUSIONS

The quick round-trip time and parallelization of aggregate reads with a distributed-replicated volume on RAID 6 bricks leave this architecture as the clear winner for smaller

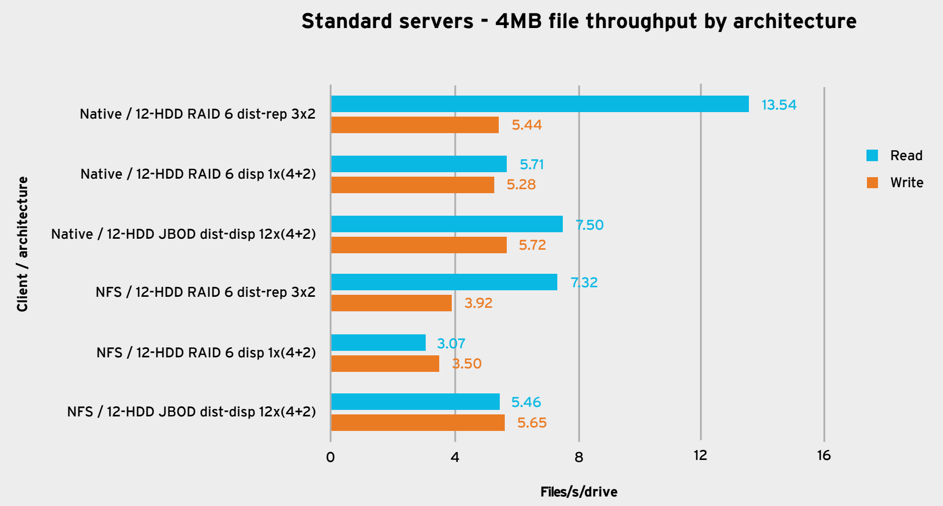


Figure 24. *Figure 24. For 4MB file workloads, standard servers on distributed-replicated volumes present distinct read advantages using the Gluster native client while writes are consistent across configurations.*

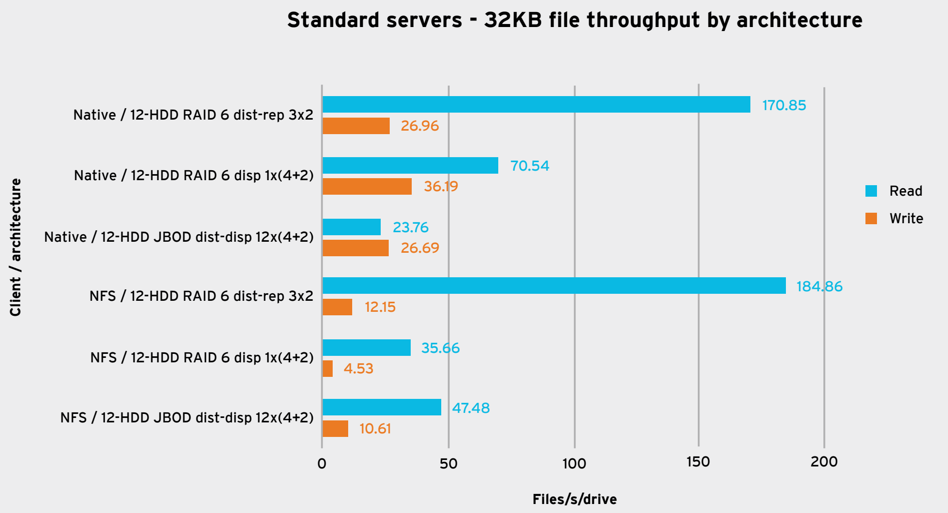


Figure 25. *Figure 25. For 32KB small-file workloads, standard servers on distributed-replicated volumes provide advantages, with native and NFS clients providing different advantages for writes and reads respectively.*

file workloads. Testing indicates that writes are generally constrained by the software.

Figure 26 and 27 provide a comparison with the 32KB and 4MB files/s metrics normalized to MB/s and combined with the data from previous tests (Figures 14-17). From these charts it is easy to see that throughput capabilities for both reads and writes are limited dramatically at very low file sizes.

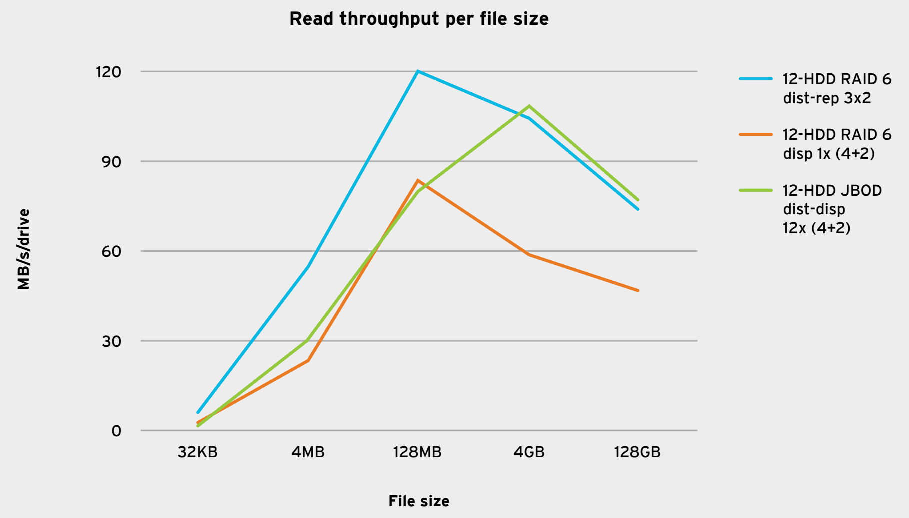


Figure 26. *Figure 26. Read throughput per file size across all file sizes.*

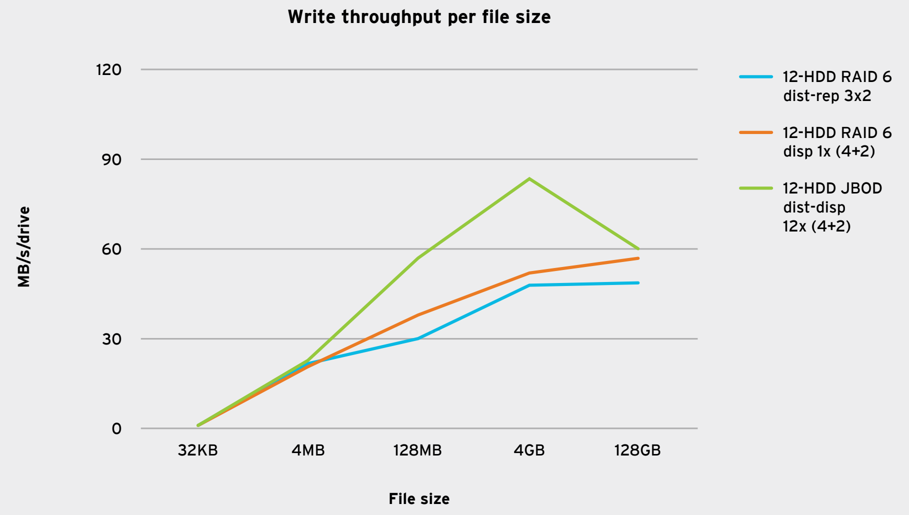


Figure 27. *Figure 27. Write throughput per file size across all file sizes.*

CASE STUDY: STREAMING VIDEO CAPTURE WORKLOAD

The foundational benchmarks described above provide a representation of the throughput capabilities of Gluster volumes under a variety of configurations and workload I/O patterns, as well as general guidance on efficient investments. Production use cases, however, rarely align with the optimized I/O patterns of a synthetic benchmark. In testing a storage solution, it is important to model real-world workloads with mixed data patterns and latency requirements.

Gluster volumes can be used productively for a large variety of data workflows. While it is impossible to model and test every use case, Red Hat chose to test streaming HD video capture, as it is usefully representative of common filesystem needs in the enterprise. Streaming HD video capture for CCTV and surveillance involves a sequential write-heavy workload that is sensitive to overall latency. It also includes accompanying concurrent read workloads that may be smaller and random in nature. The measure of performance for the back-end file store for such a system is stated in terms of how many simultaneous high-definition camera streams the storage system can support before reaching a latency spike that causes data or fidelity loss.

TEST RESULTS SUMMARY

Red Hat performed a variety of tests to determine the optimal hardware configuration, Gluster volume geometry, and specific tuning best suited to an HD video capture workload. Testing was then performed with the SPEC SFS 2014 VDA benchmark. Results showed that the Gluster storage system was able to support a peak of 1,000 HD camera streams with an overall response time of 19.44 ms and a maximum throughput of 4,469 MB/s. These results were achieved using a Gluster distributed-dispersed volume on six dense 24-disk HDD JBOD nodes.* The JBOD disks were provisioned from the RAID controller as single-disk RAID 0 volumes with write-through cache enabled.** The architecture was augmented with the addition of high-speed NVMe SSD drives in each node, which were partitioned and attached as block-level write-back caching layers for each Gluster brick using `lvmcache`.

DETAILED TEST RESULTS

Red Hat tested multiple combinations of back-end configurations, Gluster volume types, Gluster volume tunings, and caching strategies to find the best fit for the streaming video workload. The first thing immediately evident was that the replicated volume on RAID 6 bricks reached a latency spike very early, severely limiting the streams capability of the benchmark (Figure 28). Moving to a dispersed volume on JBOD bricks with the same hardware not only provided the 60% additional storage volume noted above, but allowed for a nearly 95% greater capacity in terms of camera streams.

* Based on other data collected in this study, even better performance is expected for this workload when using 12-disk standard-density servers, but time limitations prevented testing this configuration.

** Tests indicate that write-back cache at the RAID 0 volume is likely to provide better performance, but Red Hat was unable to examine this configuration while still meeting the reporting requirements of the SPEC committee.

Red Hat tested the use of the RAID controller to provide a low-level caching layer for the JBOD disks by configuring each disk as an individual RAID 0 device. Testing revealed that deploying this configuration in write-back mode yielded a 9% improvement in streams capacity. Note that it is important with this configuration to use a RAID controller protected by battery backup or non-volatile memory in order to avoid data loss. Because this protection was unavailable in our test lab, our final results reported here and to the SPEC committee were run with the RAID controller in write-through mode. Therefore the potential caching benefit at this layer had minimal impact to those results.

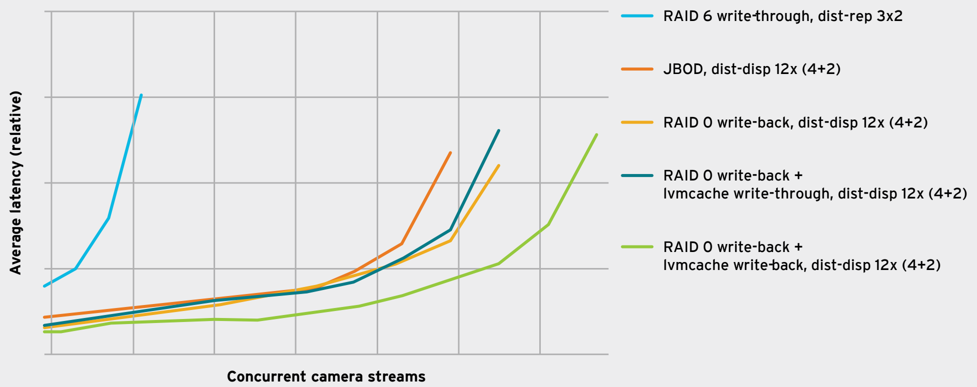


Figure 28. *Figure 28. Using lvmcache cache with a distributed-dispersed volume configuration on RAID 0 provided the best results for a video capture workload.*

Two high-speed NVMe drives were available in each Gluster server in order to tune for maximum performance. Red Hat initially tested the use of these drives with Gluster's volume tiering architecture, but current limitations make this design poorly suited to this use case.*** Instead, the drives were partitioned and configured using lvmcache to provide a second block-level caching layer above the RAID controller. The write-through mode of caching at this layer proved to be ineffective versus no cache at all. In write-back mode, however, this caching layer provided for an additional 15% increase in streams capacity.

To provide the best overall performance capability for the streaming video workload, specific tuning parameters were applied to the the servers, the Gluster volumes, and the client systems.At the system level, all servers were configured to use the `rhgs-sequential-io` tuned profile. The Gluster volume was then augmented with the options below. All server and client Gluster processes were restarted after these changes.

- disperse.eager-lock: off
- cluster.lookup-optimize: on
- performance.client-io-threads: on
- client.event-threads: 4

*** bugzilla.redhat.com/show_bug.cgi?id=1292391

- server.event-threads: 4

On client systems, the `throughput-performance` tuned profile was used, and the following additional kernel parameters were applied per recommendations from the SPEC user's guide:

- net.core.netdev_max_backlog = 300000
- sunrpc.tcp_slot_table_entries = 128
- net.core.somaxconn = 65535
- net.ipv4.tcp_fin_timeout = 5

DISK EFFICIENCY ANALYSIS

Due to time constraints, Red Hat was unable to test peak VDA workload capabilities across multiple system configurations for the purpose of price-performance comparisons. However, it is interesting to study the efficiency of camera streams per physical disk in the system, as per-disk throughput and overall latency are the finite resources for this workload that will cause bottlenecks first. For the purposes of these comparisons the caching layers of the solutions were not included in the per-disk calculations.

The tested distributed-dispersed Red Hat Gluster Storage solution on six Supermicro SSG-6028R-E1CR24N nodes was able to achieve a peak of 6.9 streams-per-HDD, with a total of 144x 6TB SAS HDDs and a usable capacity of 524TB. This streams-per-HDD result is 50% greater than the competing system reporting the highest peak streams as of this writing.*

SPEC SFS 2014 VDA

The SPEC SFS 2014 VDA workload generally simulates applications that store data acquired from a temporally volatile source (e.g. surveillance cameras). The business metric for the benchmark is streams, referring to an instance of the application storing data from a single source—e.g. one video feed. For these kinds of HD video capture application, the storage administrator is primarily concerned with maintaining a minimum fixed bit rate per stream and secondarily about maintaining the fidelity of the stream. The goal is to provide as many simultaneous streams as possible while meeting the bit rate and fidelity constraints for individual streams.**

REPORTED RED HAT RESULTS

Table 2 and Figure 29 provide a SPEC SFS 2014 VDA performance summary for 10 iterations running on Red Hat Gluster storage on a pool of dense Supermicro storage servers. These results demonstrate a consistent latency ramp through 900 streams. Complete SPEC SFS 2014 VDA reports can be found at spec.org/sfs2014/results/.

* Of the SPEC SFS 2014 VDA results reported to the SPEC committee for primarily HDD-based storage solutions as of this writing, the IBM Spectrum Scale 4.2 with Elastic Storage Server GL6 reports the highest peak streams level at 1,600. This solution uses 348x 2TB NL-SAS drives for primary storage with a total usable capacity of 458.5 TB. The streams-per-HDD calculation for this solution is 4.6, compared to 6.9 streams-per-HDD for the Red Hat solution.

** More information on SPEC SFS 2014 can be found at spec.org/sfs2014/docs/usersguide.pdf.

Table 2. *SPEC SFS 2014 VDA results for 10 iterations.*

BUSINESS METRIC (STREAMS)	AVERAGE LATENCY (MSEC)	STREAMS (OPS/SEC)	STREAMS (MB/SEC)
100	6.08	1000.56	449.57
200	6.72	2001.00	903.28
300	10.87	3001.50	1354.17
400	12.75	4002.01	1803.00
500	14.89	5002.33	2246.84
600	18.53	6002.63	2716.07
700	21.51	7003.64	3160.21
800	24.56	8003.50	3599.42
900	28.95	9003.54	4055.00
1000	105.16	9927.00	4469.41

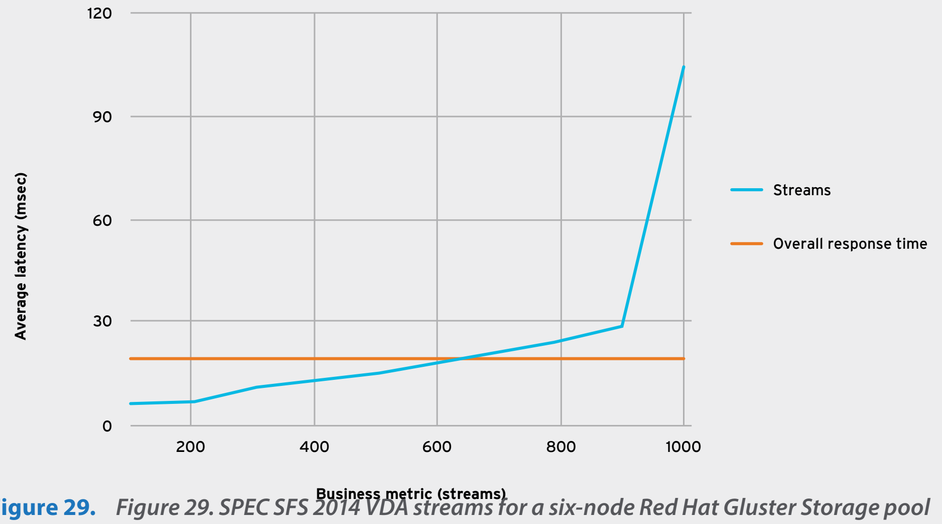


Figure 29. *Figure 29. SPEC SFS 2014 VDA streams for a six-node Red Hat Gluster Storage pool based on dense Supermicro servers with NVMe caching implemented through* `lvmcache`.

VOLUME TIERING

For small file sizes, Gluster's volume tiering architecture can be used to improve performance. Tiering is implemented by attaching a new volume geometry backed by high-performance SSDs as a hot tier to an existing volume, typically backed by spinning HDDs. Once in place, all new writes are initially directed to the hot tier. Gluster manages the process of passing data between the hot tier and the backend cold tier based both on data access and a randomized promotion/demotion algorithm.

A cache-friendly workload may dramatically benefit from the addition of an SSD-based hot tier to a Gluster volume. Such a workload would typically be read-heavy, with a relatively small number of smaller size files being accessed concurrently by multiple clients. These workloads would see a majority of their active file operations performed against the much faster SSDs.

TIERING TESTS

For testing, Red Hat created a distributed-replicated hot tier from high-speed NVMe SSD drives installed on each node. A single NVMe drive was installed on each node. A hot tier was then configured in a distributed-replicated volume across all six Gluster server nodes. The high and low watermarks were left as the system defaults. Figure 30 shows very clearly that the hot tier offers a major advantage for 4MB files where most file operations avoid the cold tier. Read throughput improved over 120% while writes show a more modest 30% improvement.

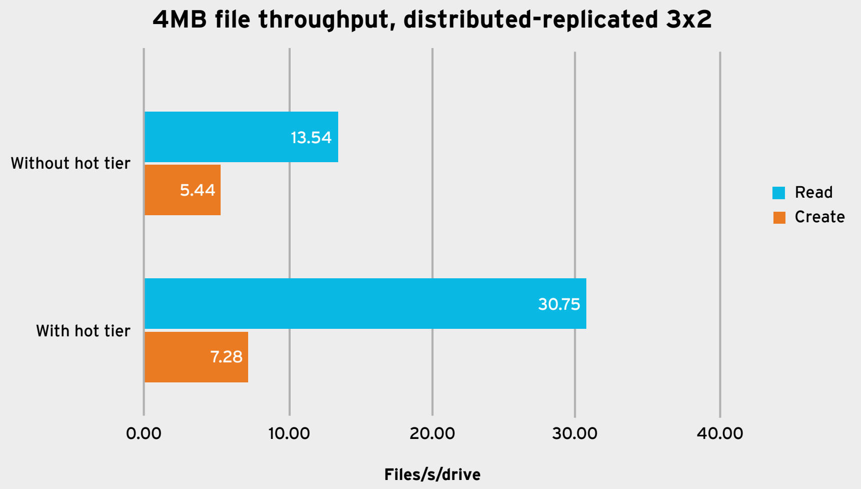


Figure 30. *Figure 30. For 4MB files, an NVMe-based hot tier provided significant improvements in throughput efficiency, demonstrating over 120% improvement in read efficiency.*

The benefit of the hot tier becomes even more dramatic with smaller file sizes. With the 32KB file size, the tiered configuration ran over 310% faster for reads and over 230% faster for writes versus the same distributed-replicated volume without a hot tier (Figure 31).

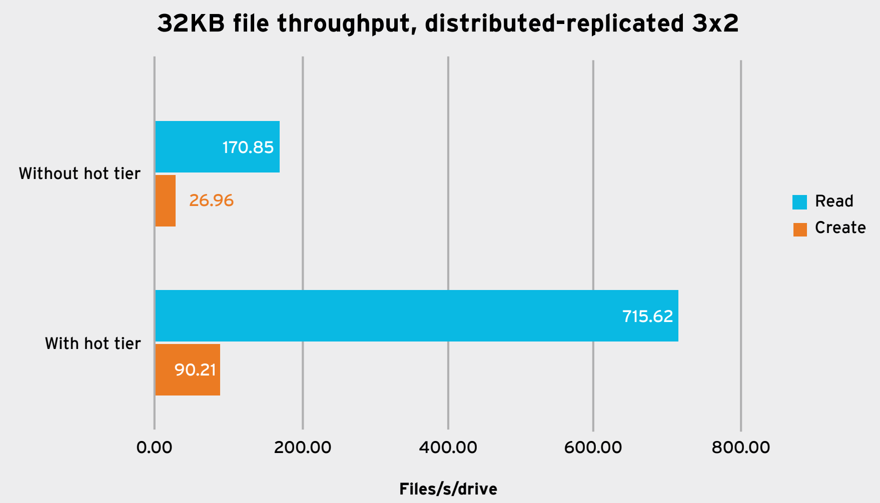


Figure 31. *Figure 31. For 32KB files, an NVMe-based hot tier provides even greater advantages, with 310% faster read and 230% faster write efficiency.*

GENERAL GUIDANCE

Volume tiering should be considered and tested carefully with specific workloads and needs. As noted, a tiering-friendly workload with a relatively small working set of relatively small files is likely to benefit most from an SSD hot tier. Likewise, latency-sensitive workloads may perform better with an appropriately-sized SSD hot tier or otherwise with a cache, as experienced in the streaming video use case study described above.

Additionally, while it may seem obvious that a configuration with a dispersed cold tier would be an economical choice, current limitations in the Gluster code as of this writing may lead to detrimental performance. Red Hat testing has shown that a hot tier on top of a dispersed cold tier can actually significantly reduce the throughput capabilities of the system. Note that this is distinct from the VDA use case described above where a dispersed volume was used successfully in conjunction with SSD caching via lvmcache, rather than with an SSD hot tier.

NFS VERSUS GLUSTER NATIVE CLIENT

As stated previously, the parallelization of the Gluster native client can provide advantages for larger file sizes. The NFS client is expected to be an advantageous option in use cases where a relatively small number of smaller files are accessed consistently by the clients, allowing the bulk of the I/O to remain at the client-side memory cache. One example is a web server farm, where small HTML, CSS, PHP, and similar files are accessed consistently and rarely change.

Red Hat streaming tests described above concentrated on committed writes and consistency across concurrent clients. As a result, tests of small files (4MB and under) were done in such a way that the advantages of NFS client-side caching would have been masked, with those caches being explicitly dropped and the workload pattern designed specifically to measure persistent commits. Therefore, small (4MB) and tiny (32KB) file size NFS tests were omitted and testing described below focused instead on larger file sequential I/O (4GB and 128GB file sizes).

TEST RESULTS SUMMARY

Red Hat engineers observed different advantages, depending on how the volumes were constructed:

Replicated volumes. For replicated volumes at lower levels of client concurrency, the NFS client consistently maintained a slight advantage for both reads and writes. As concurrency increased, however, the NFS client lost its advantage for reads, but remained advantageous or on-par for writes. For larger file sizes with a distributed-replicated volume, splitting protocols could be a worthwhile consideration—namely writing data via NFS while reading via the Gluster native client.

Dispersed volumes. The Gluster native client is by far the best choice for any write operations on dispersed volumes. At high concurrency levels, the Gluster native client will outperform the NFS client for reads as well. At lower levels of client concurrency, the NFS client shows a slight advantage for reads. For most larger file workloads and use cases, the efficiency of the Gluster native client-side algorithms for dispersed volumes make it a clear choice over NFS for throughput efficiency.

CONFIGURATION DETAILS

Red Hat tested both the Gluster native client and the NFS client for all of the sequential I/O workloads. The Gluster volume was configured to use its built-in NFS server rather than the newer user space NFS-Ganesha. While NFS-Ganesha is expected to be the future of NFS access to Gluster volumes, at the time of this testing the built-in NFS server demonstrated a performance advantage.

Clients were mounted with NFS protocol version 3. The clients were manually mounted to the servers in such a way as to evenly distribute the mounts across the Gluster servers. Twelve clients were mounted with the async option from six servers. Each server had two clients. As with all of the native client tests, both client-side and server-side file syncs and cache drops were completed before every test iteration.

MEDIUM 128MB FILES: REPLICATED VERSUS DISPERSED

A distributed-replicated (3x2) volume serving 128MB files was tested with both the Gluster native client and the NFS client. The NFS client shows advantages for writes (Figure 32), particularly as the number of workers and resulting write concurrency increases.

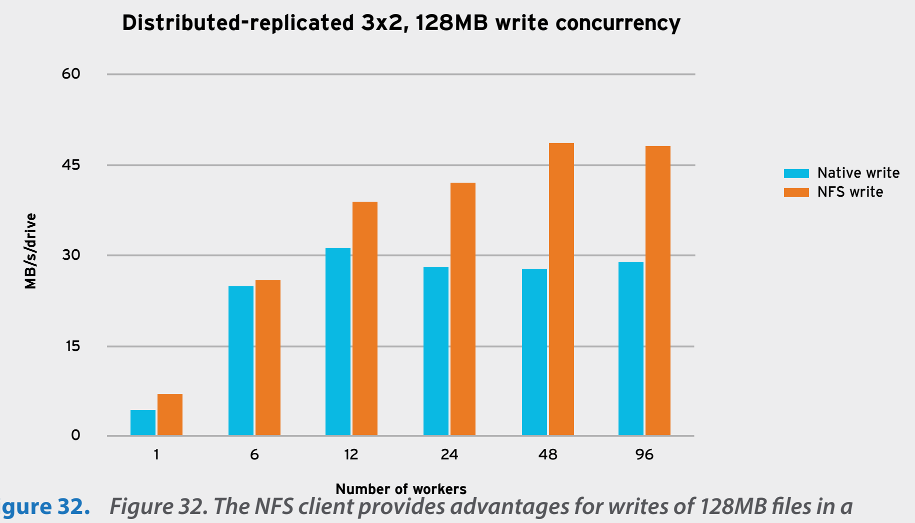


Figure 32. *Figure 32. The NFS client provides advantages for writes of 128MB files in a distributed-replicated (3x2) volume.*

As shown in Figure 33, the opposite effect applies to reads. For small amounts of read concurrency, the NFS client provides a slight advantage. However, the Gluster native client leads as read concurrency grows.

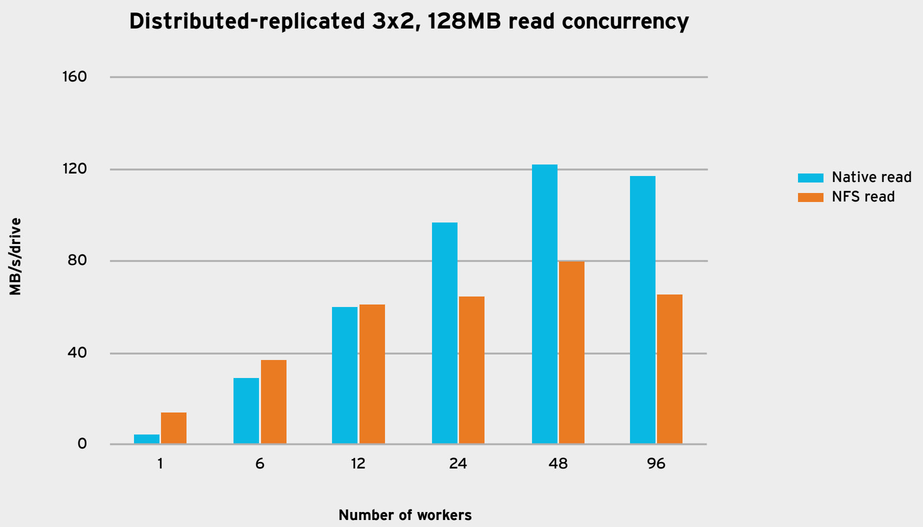


Figure 33. *Figure 33. The Gluster native client provides advantages for reads, particularly as concurrency grows.*

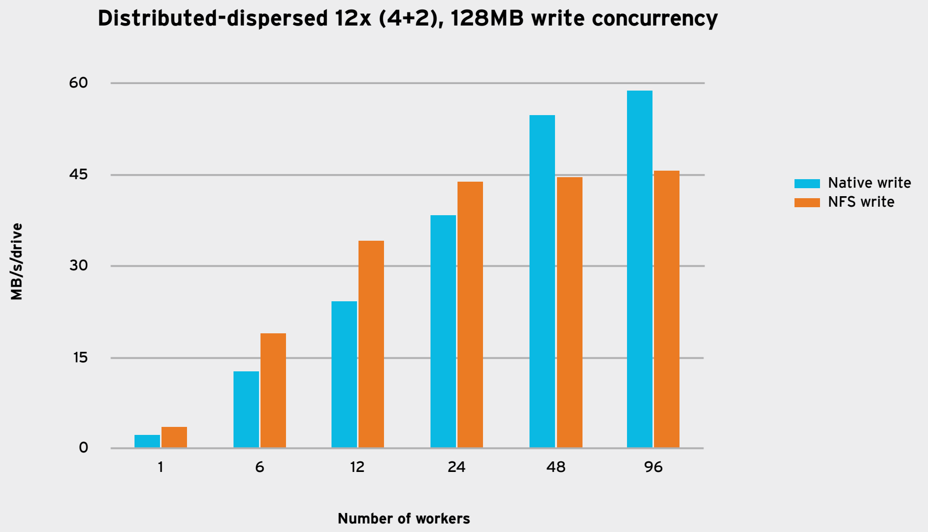


Figure 34. *Figure 34. Distributed-dispersed results for 128MB files are mixed, with added write concurrency favoring the Gluster native client.*

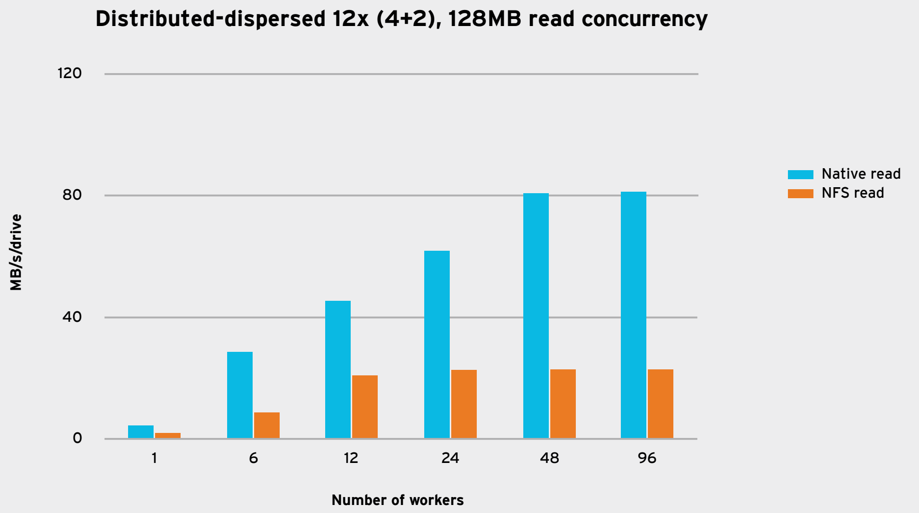


Figure 35. *Figure 35. The Gluster native client presents significant advantages for reads of 128MB files, particularly as read concurrency increases.*

LARGE 4GB FILES: REPLICATED VERSUS DISPERSED

The Gluster native client and the NFS client are closely matched for 4GB writes across a distributed-replicated volume (Figure 36). However, advantages for the Gluster native client grow with concurrency for 4GB reads (Figure 37).

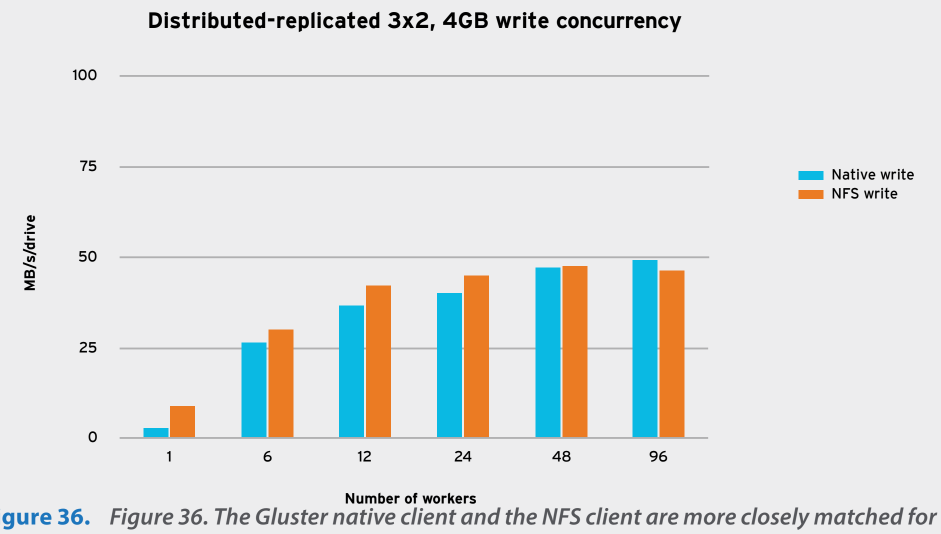


Figure 36. *Figure 36. The Gluster native client and the NFS client are more closely matched for 4GB writes.*

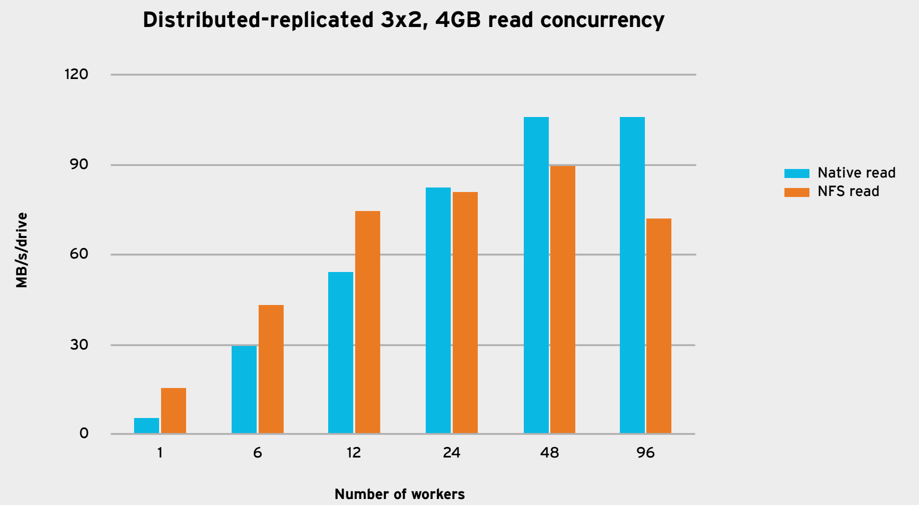


Figure 37. *Figure 37. As read concurrency grows, advantages accrue to the Gluster native client for distributed-replicated volumes and 4GB files.*

Figures 38 and 39 illustrate results for 4GB files on a distributed-dispersed volume, comparing the Gluster native client and the NFS client. Again, the Gluster native client provides advantages for writes with increased write concurrency. Gluster native client advantages for reads are comprehensive, particularly with growing read concurrency.

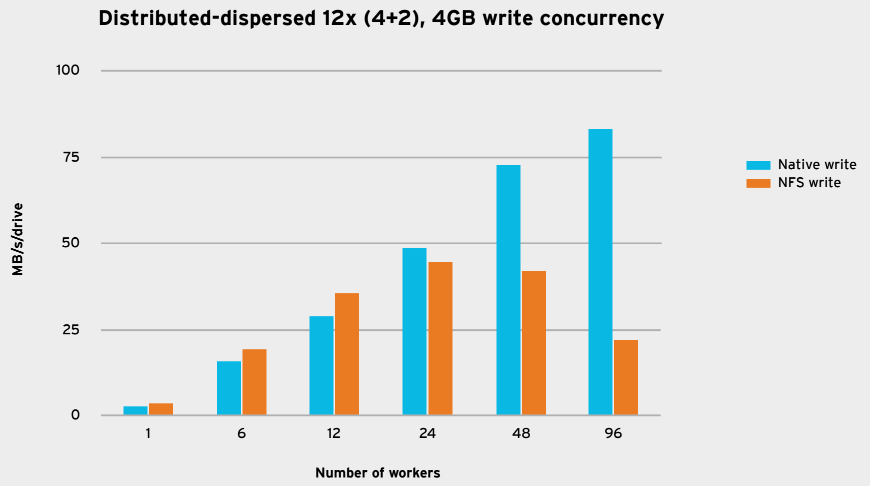


Figure 38. *Figure 38. Gluster native client performance increases with concurrency for distributed-dispersed volumes and 4GB files.*

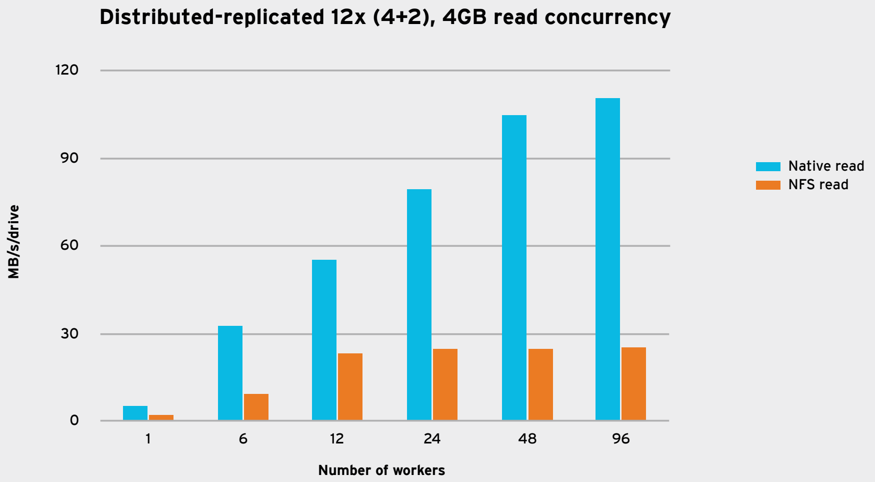


Figure 39. *Figure 39. The GlusterFS native client provides distinct advantages for 4GB reads on distributed-dispersed volumes.*

APPENDIX A: BENCHMARK TOOLS AND BASELINE TESTING

The sections that follow provide additional background on benchmark tools and baseline testing.

IOZONE

IOzone (iozone.org) was used to test the large file (128MB and up) sequential read/write performance of the GlusterFS volumes. IOzone is a file system benchmark tool that generates and measures a variety of file operations. IOzone's cluster mode option is particularly well-suited for distributed storage testing because testers can start many worker threads from various client systems in parallel, targeting the GlusterFS volume. Flags for IOzone are as follows:

```
--t #               Run in throughput mode with # threads
--s #               Size of file to test
--r #               Record size to test
--+m <file>         Input file for client configuration
--c                 Include close() in the timing calculations
--e                 Include flush in the timing calculations
--w                 Keep temporary files
--+z                Enable latency histogram logging
--+n                Do not do retests
--i #               Specify which tests to run
```

Smallfile

Smallfile is a Python-based distributed POSIX workload generator that was used to measure the small file (4MB and below) sequential read/write performance of the GlusterFS volumes. Smallfile has no dependencies on any specific file system or implementation and was written to complement use of the IOzone benchmark for measuring performance of small- and large-file workloads. The benchmark returns the number of files processed per second and the rate that the application transferred data in megabytes per second. Flags for smallfile are as follows:

```
--threads                     Number of workload generator threads
--file-size                   Total size per file
--files                       Number of files for each thread
--top <path>                  Directory I/O is done in
--host-set                    List of clients to run workload
--prefix <name>               Prepend output files with <name>
--stonewall Y                 Measure as soon as one thread finishes
--network-sync-dir <path>     Shared path for client sync
--operation                   File operation to perform
```

Dropping Caches

By default, Linux uses any available memory to cache disk access in order to improve ongoing I/O performance. While this is a major benefit to many use cases, it will easily skew and invalidate the results of an I/O benchmark. To prevent this from having an inappropriate effect on our test results, we sync any outstanding cached writes and force the kernel to drop all disk caches before every test iteration. We perform this operation on all servers and all clients participating in the test.

```
# sync ; echo 3 > /proc/sys/vm/drop_caches
```

Baseline Performance Characteristics

Baseline performance characteristics can be useful in gaining a realistic impression of benchmark testing using software-defined storage.

- Seagate ST6000NM0034 6TB SAS drives installed on all test systems have a manufacturer's specification of 226 MB/s maximum sustained throughput for both reads and writes. Single-disk baseline tests with IOzone, which include a thin-provisioned LVM logical volume and an XFS filesystem, indicate maximum throughputs of 205MB/s writes and 212MB/s reads (91% and 94% of mfg spec respectively).
- A RAID 6 array with 12 disks was measured at maximum throughput of 1,224MB/s writes and 1,107MB/s reads. For the 24-disk dense servers, these throughput numbers were validated for simultaneous saturation of two RAID 6 arrays, indicating that the the dual-RAID backplane was capable of sustaining the full bandwidth to both arrays.
- The Supermicro-based testing network was validated using iperf3. This testing ensured that each server and client node could reach its maximum network bandwidth. Full-mesh saturation was also performed to ensure that all server and client nodes could sustain full bandwidth without limitation at the network fabric.

APPENDIX B: SUBSYSTEM METRICS DURING SATURATION WORKLOADS

The sections that follow detail concurrency testing performed on multiple Red Hat Gluster Storage volumes in the interest of identifying subsystems that limit performance during saturation workloads.

128GB file tests on 6x2 distributed-replicated volume on RAID 6 with Gluster native client

Figure 40 illustrates 128GB file throughput with increasing concurrency on a distributed-replicated volume (6x2) on RAID 6 Gluster bricks.

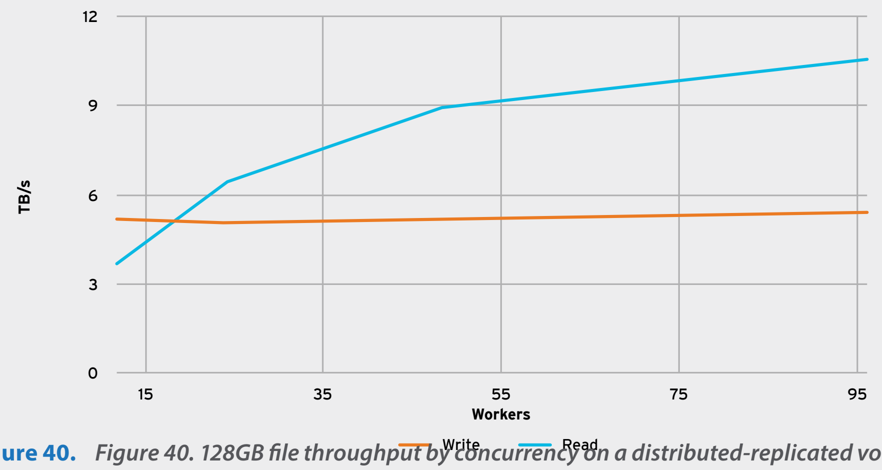


Figure 40. *Figure 40. 128GB file throughput by concurrency on a distributed-replicated volume (6x2) on RAID 6.*

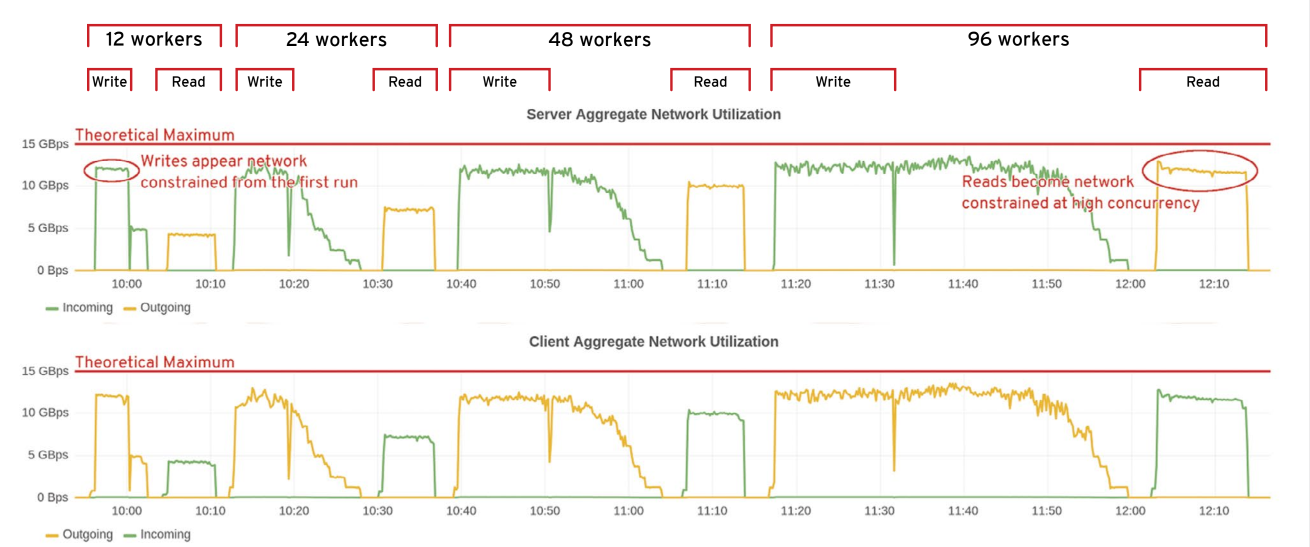


Figure 41. *Figure 41. Writes are network constrained from even low concurrency, while reads become network constrained at high concurrency.*

Figure 41 shows network activity for both writes and reads for increasing levels of concurrency.

Figure 42 illustrates both server and client CPU utilization and system load. System utilization remained relatively consistent with reads at maximum concurrency approaching 25% utilization.

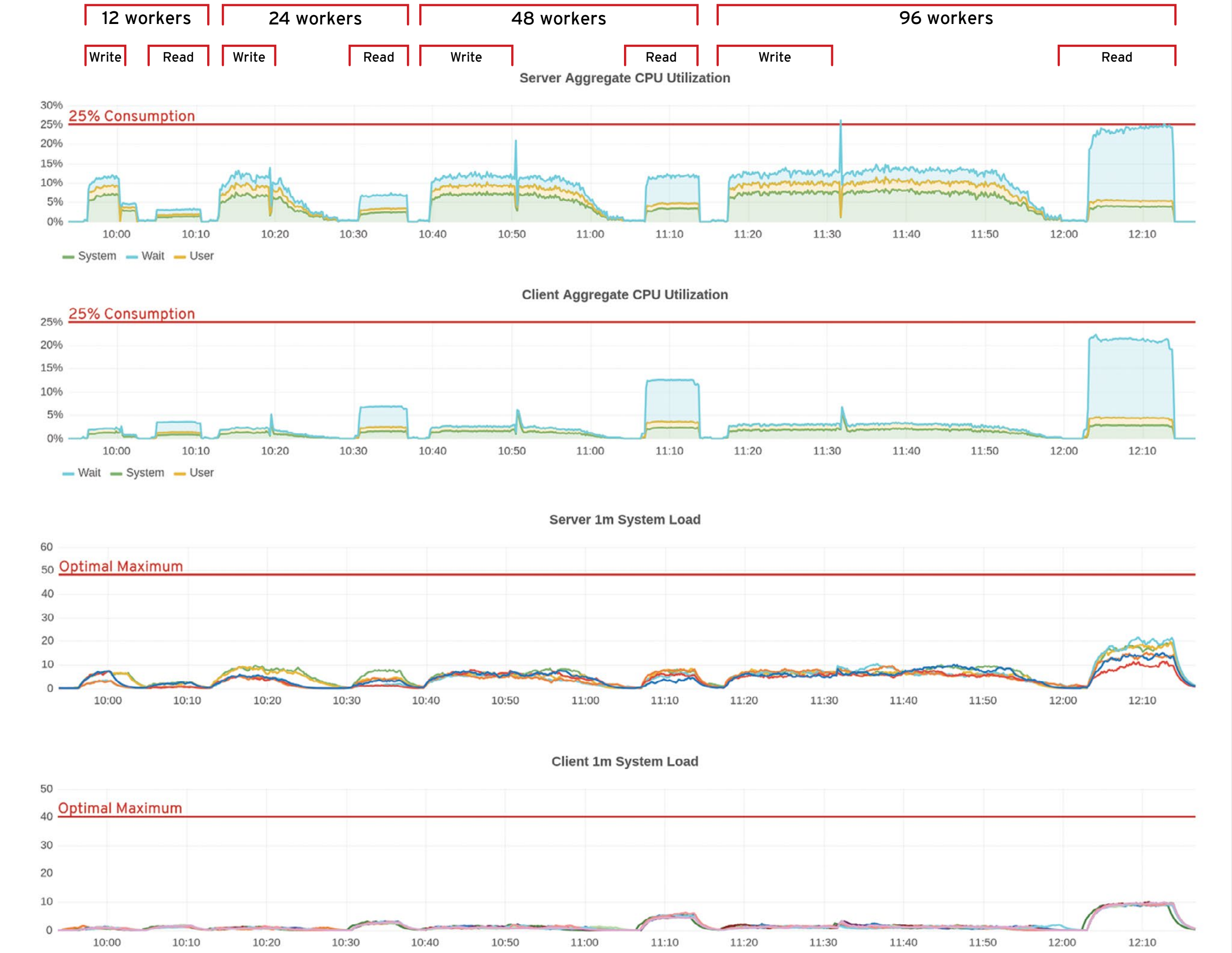


Figure 42. *Figure 42. Server and client CPU and system load for 128GB files on a distributed-replicated volume on RAID 6 bricks.*

Figure 43 illustrates memory utilization and blocked processes for both servers and clients.

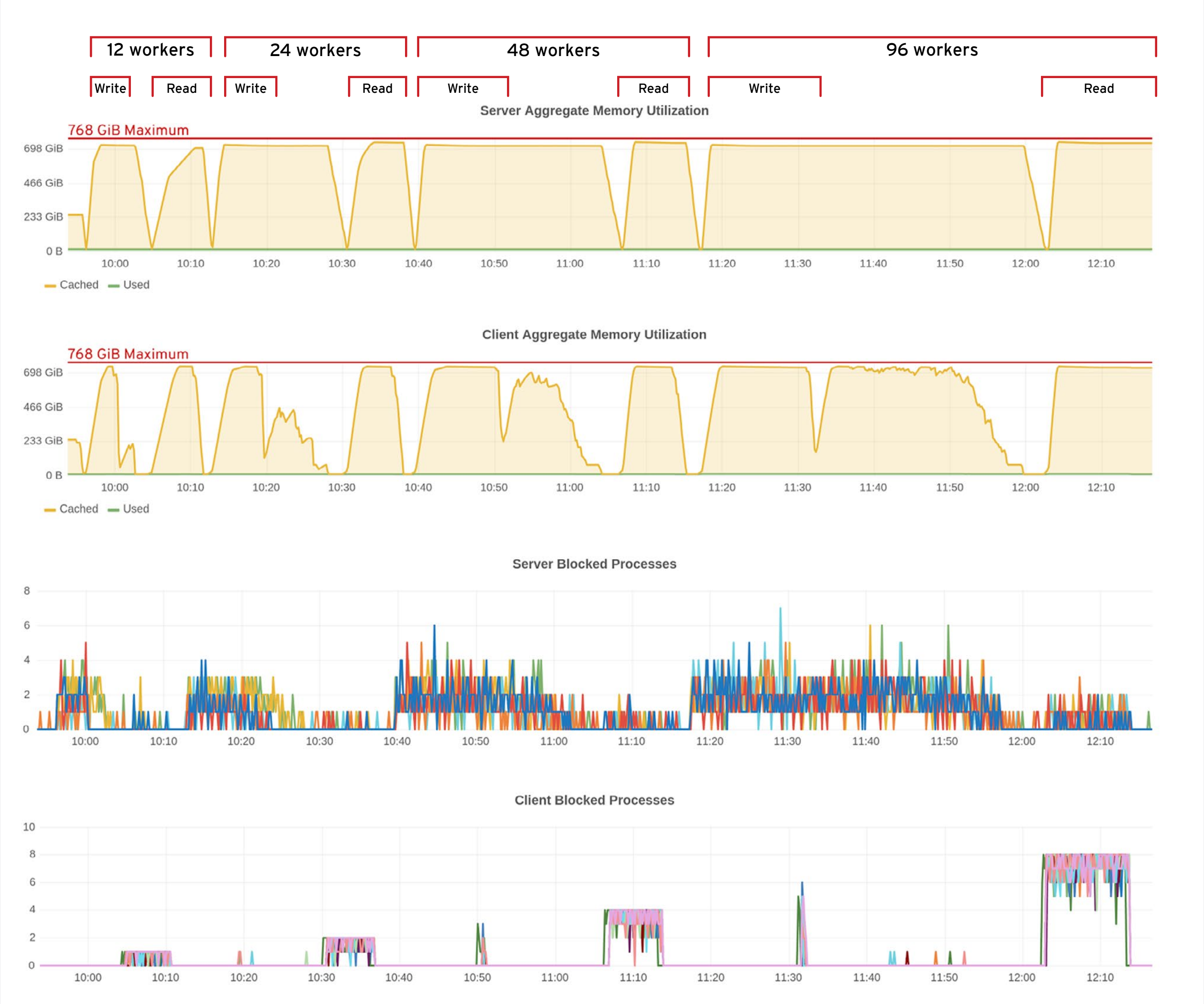


Figure 43. *Figure 43. Memory utilization and blocked processes for both servers and clients.*

Figure 44 shows that disk reads can become a bottleneck at high levels of concurrency (96 workers).

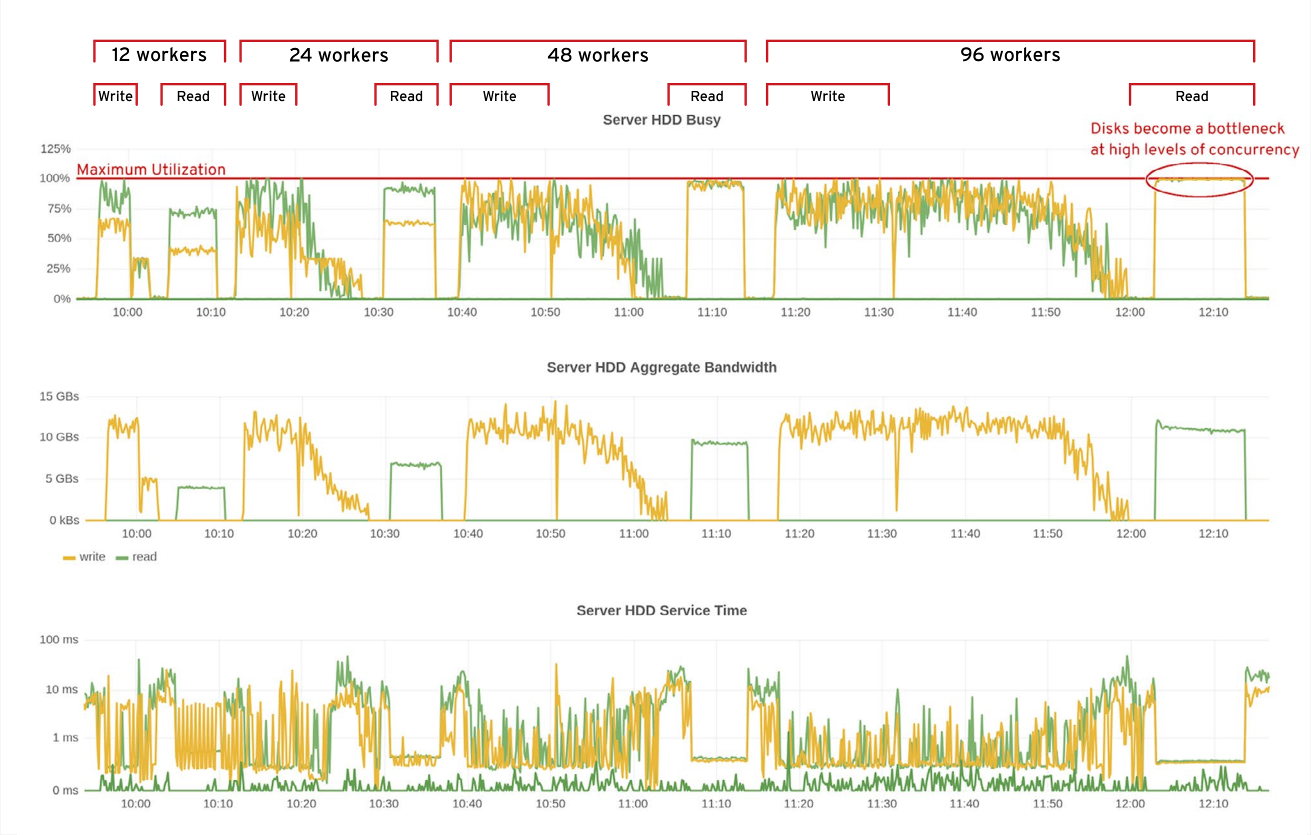


Figure 44. *Figure 44. Server HDD activity, aggregate bandwidth, and service time.*

128GB file tests on 24x(4+2) distributed-dispersed volume on JBOD

Figure 45 illustrates 128GB file throughput with increasing concurrency on a distributed-dispersed volume (24x(4+2)) on JBOD Gluster bricks.

Figure 46 shows network utilization, with a potential bottleneck for high-concurrency writes.

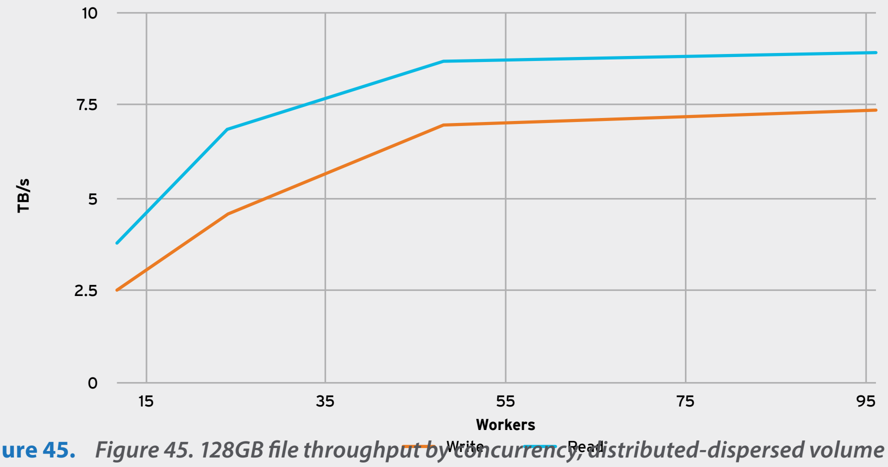


Figure 45. *Figure 45. 128GB file throughput by concurrency, distributed-dispersed volume (24x (4+2) on JBOD).*

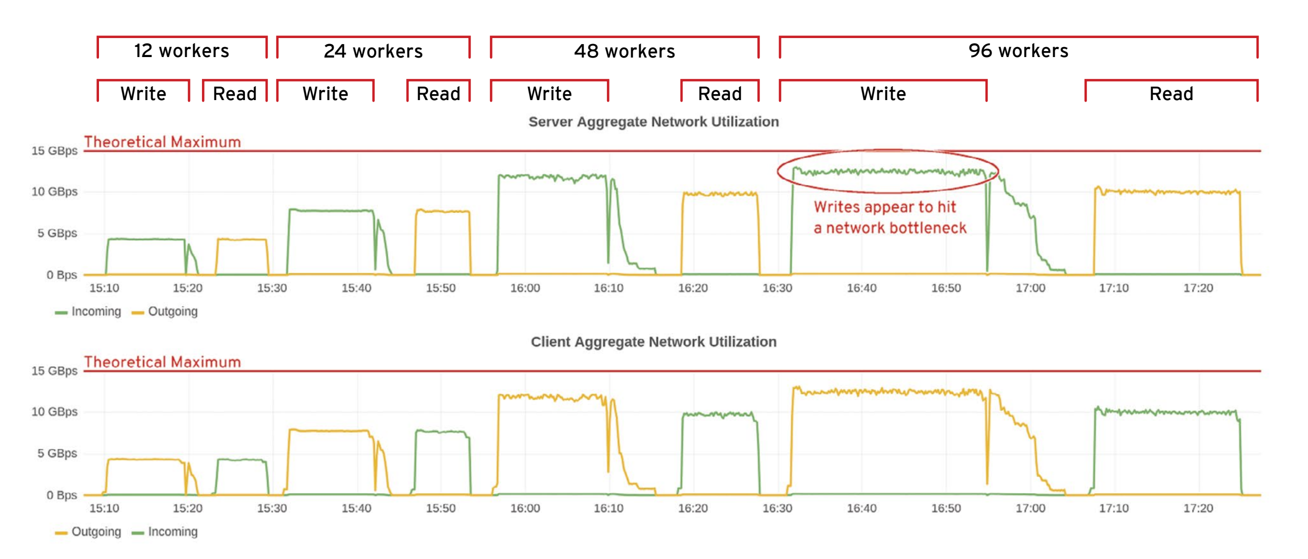


Figure 46. *Figure 46. At high concurrency, (96 workers), writes appear to hit a network bottleneck.*

Figure 47 shows aggregate CPU utilization and system load for both servers and clients. It is interesting to note that server CPU is more taxed on writes and less on reads as compared to the distributed-replicated volume described above.

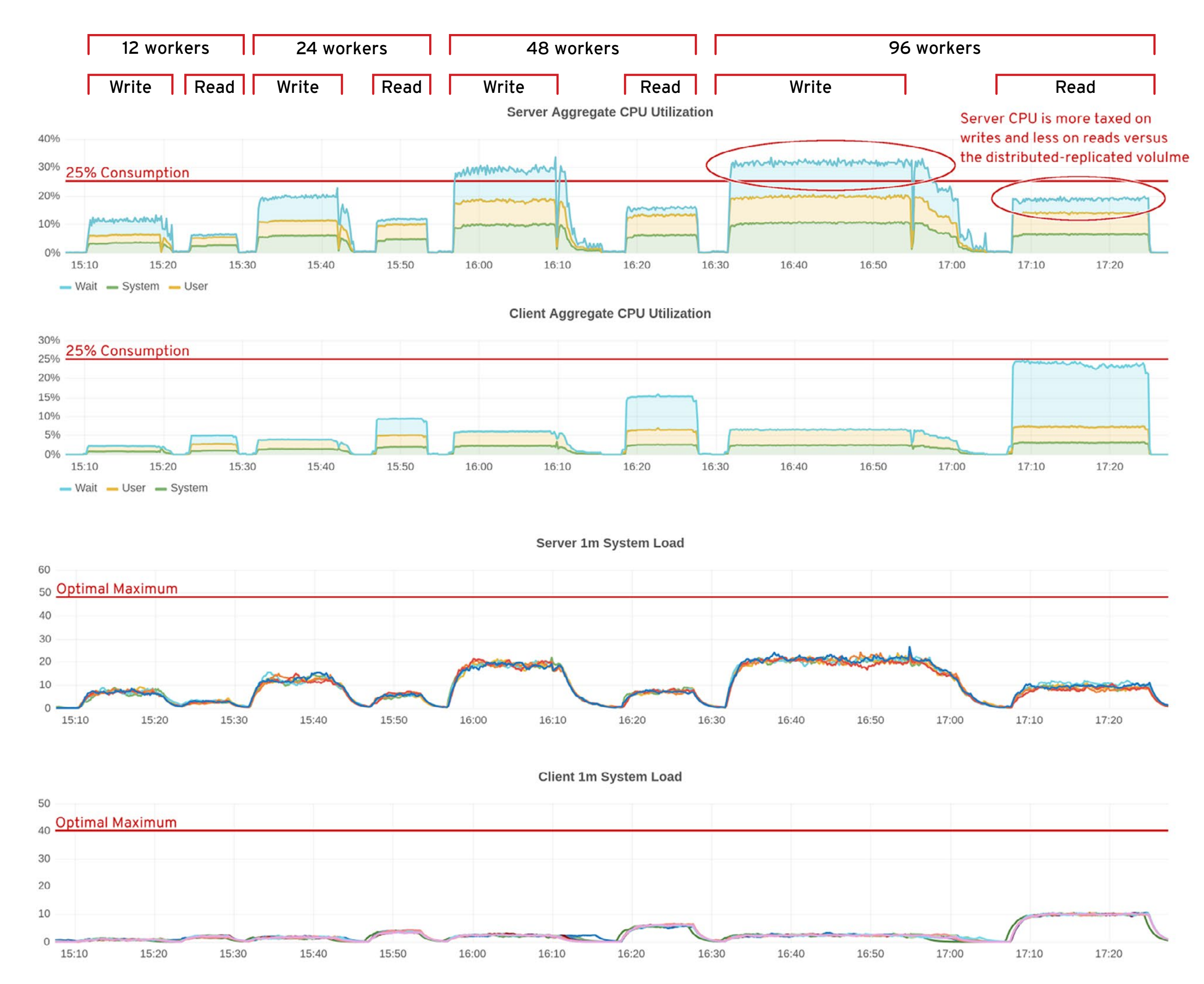


Figure 47. *Figure 47. Server and client CPU utilization and system load.*

Figure 48 shows memory utilization and blocked processes for both servers and clients.

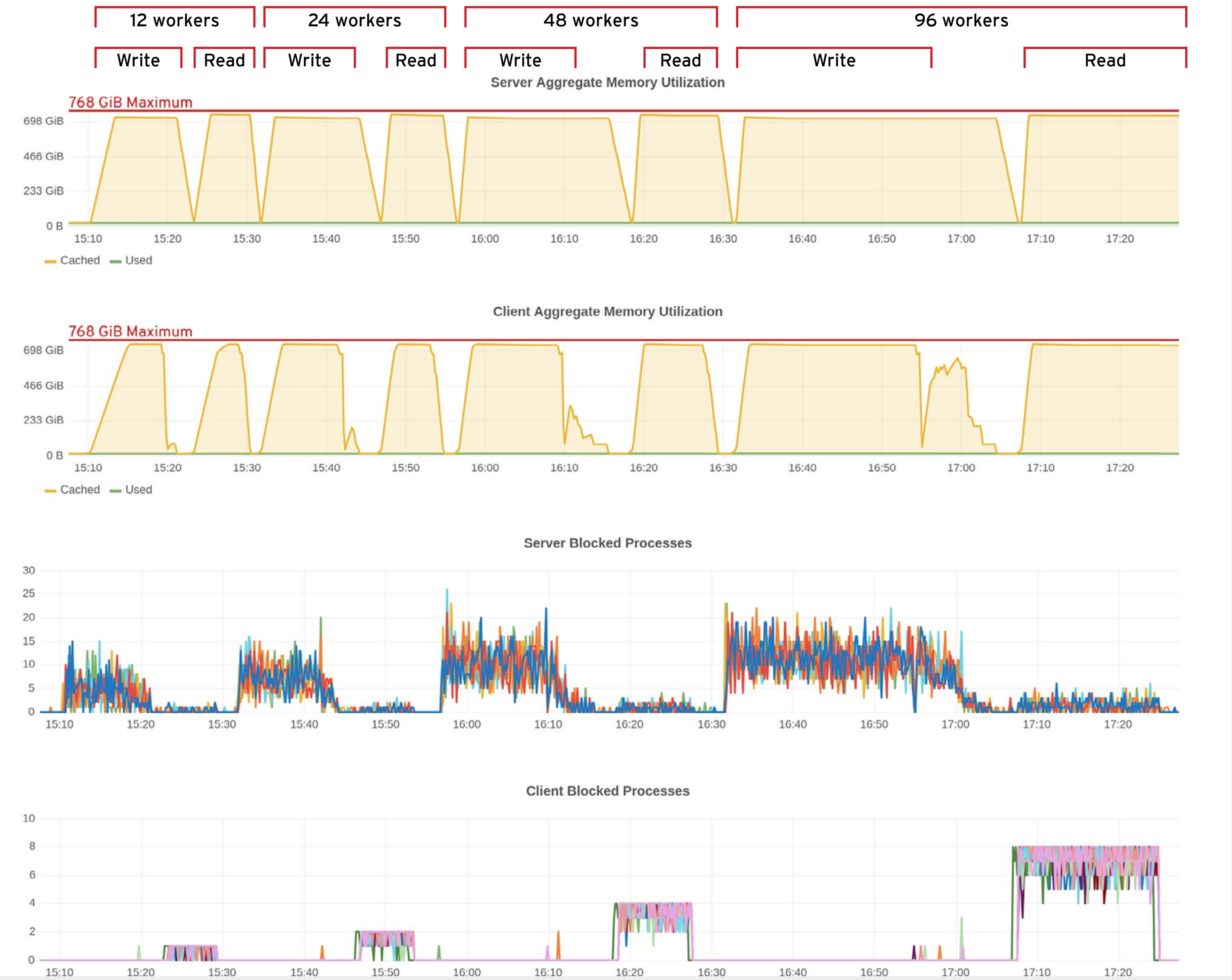


Figure 48. *Figure 48. Memory and blocked processes for both servers and clients.*

Figure 49 illustrates HDD activity, apparently indicating an aggregate HDD throughput limit for reads at high concurrency.

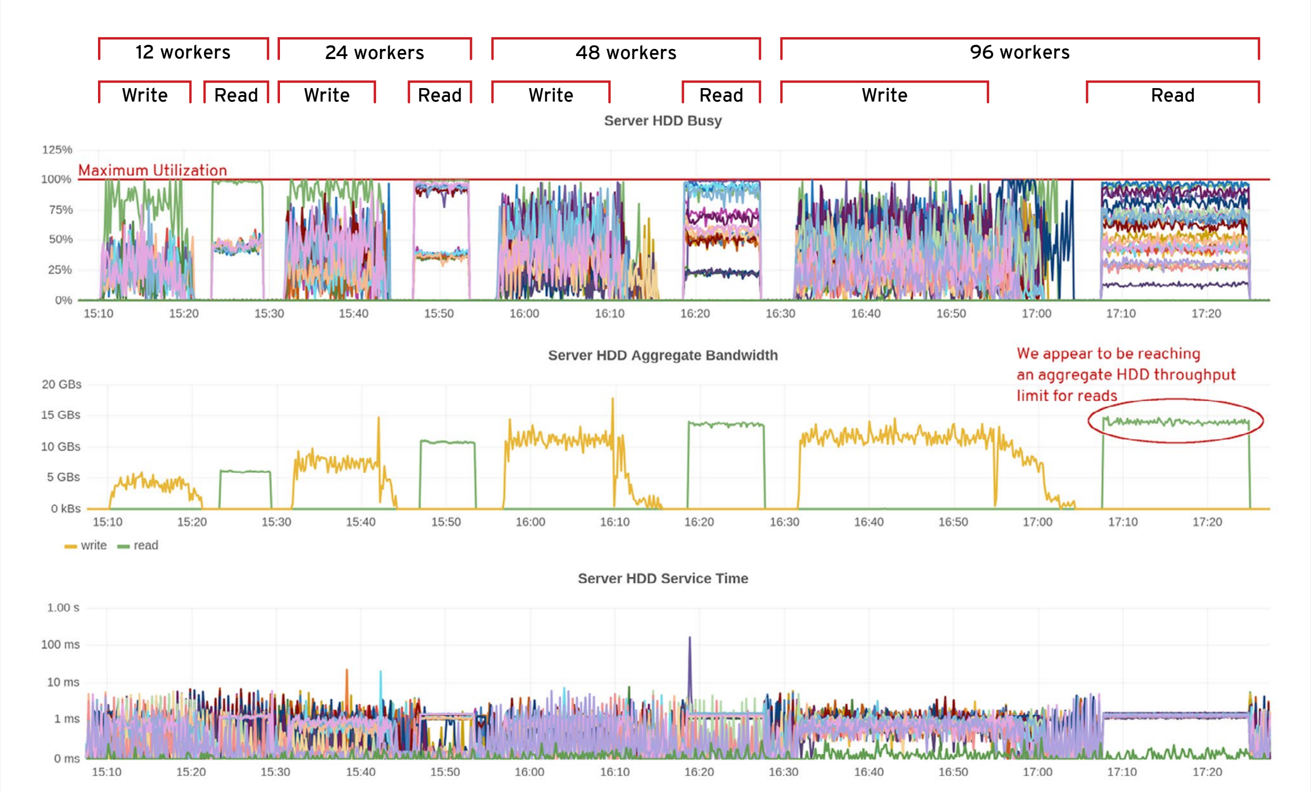


Figure 49. *Figure 49. HDD activity, aggregate bandwidth, and service time.*

32KB file tests on 6x2 distributed-replicated volume on RAID 6

Figure 50 illustrates 32KB file throughput with increasing concurrency on a distributed-replicated volume (6x2) on RAID 6 bricks. No particular subsystem limits stand out for the 32KB file tests, suggesting that performance is ultimately limited at the Gluster application layer.

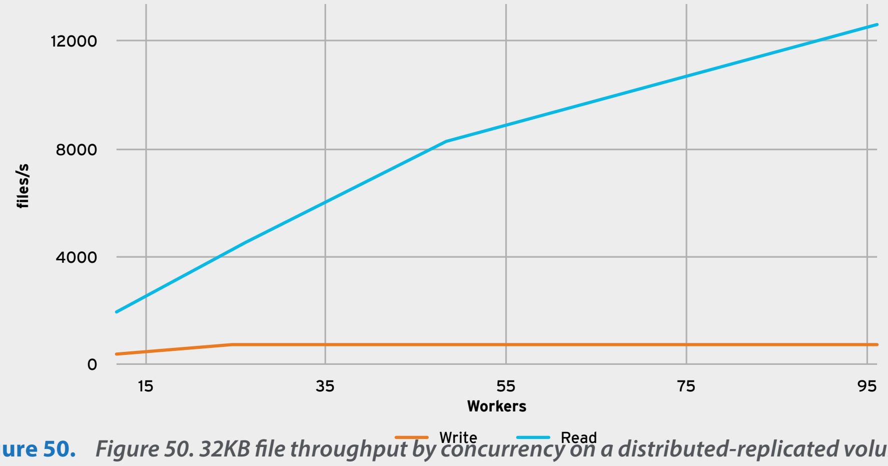


Figure 50. *Figure 50. 32KB file throughput by concurrency on a distributed-replicated volume (6x2) on RAID 6 bricks.*

Figure 51 shows network utilization is not close to any constraints.

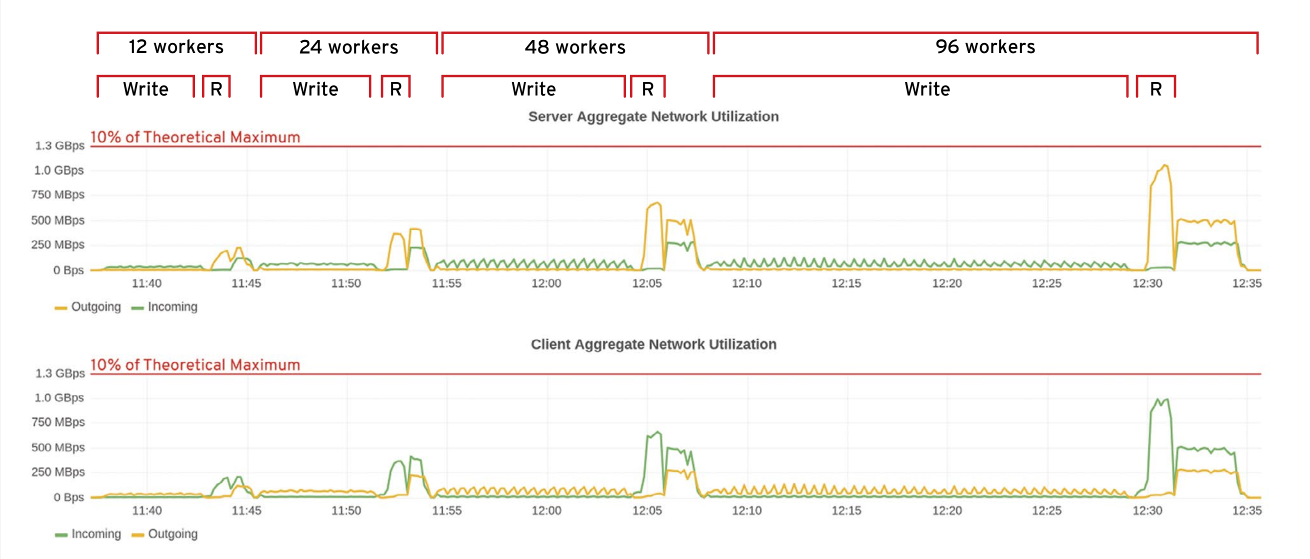


Figure 51. *Figure 51. Network utilization for servers and clients.*

Figure 52 illustrates CPU utilization and system load for the 32KB concurrency tests.

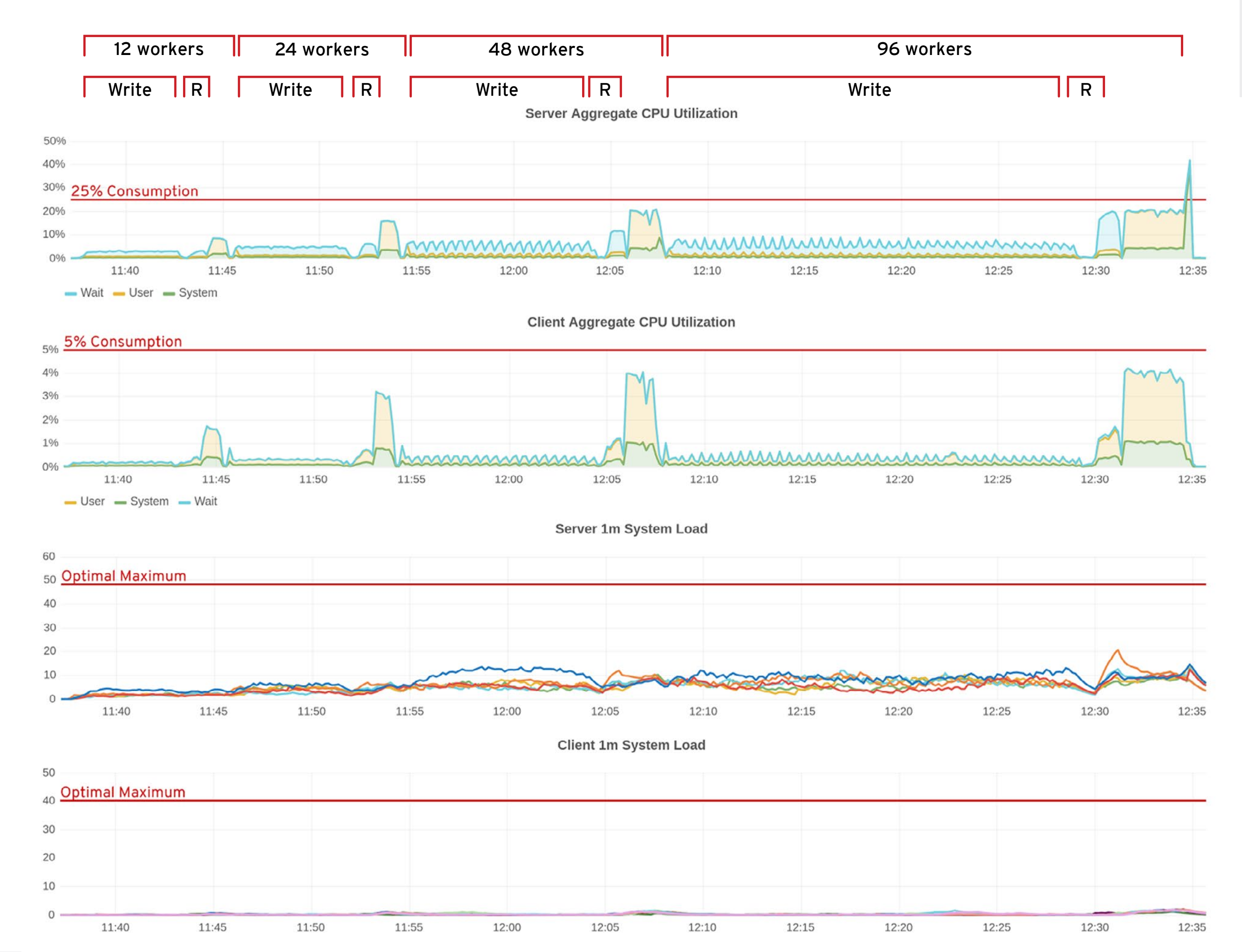


Figure 52. *Figure 52. Server and client aggregate CPU utilization and system load remain well below optimum maximum levels.*

Figure 53 illustrates that aggregate memory utilization and blocked processes for both servers and clients remain relatively low.

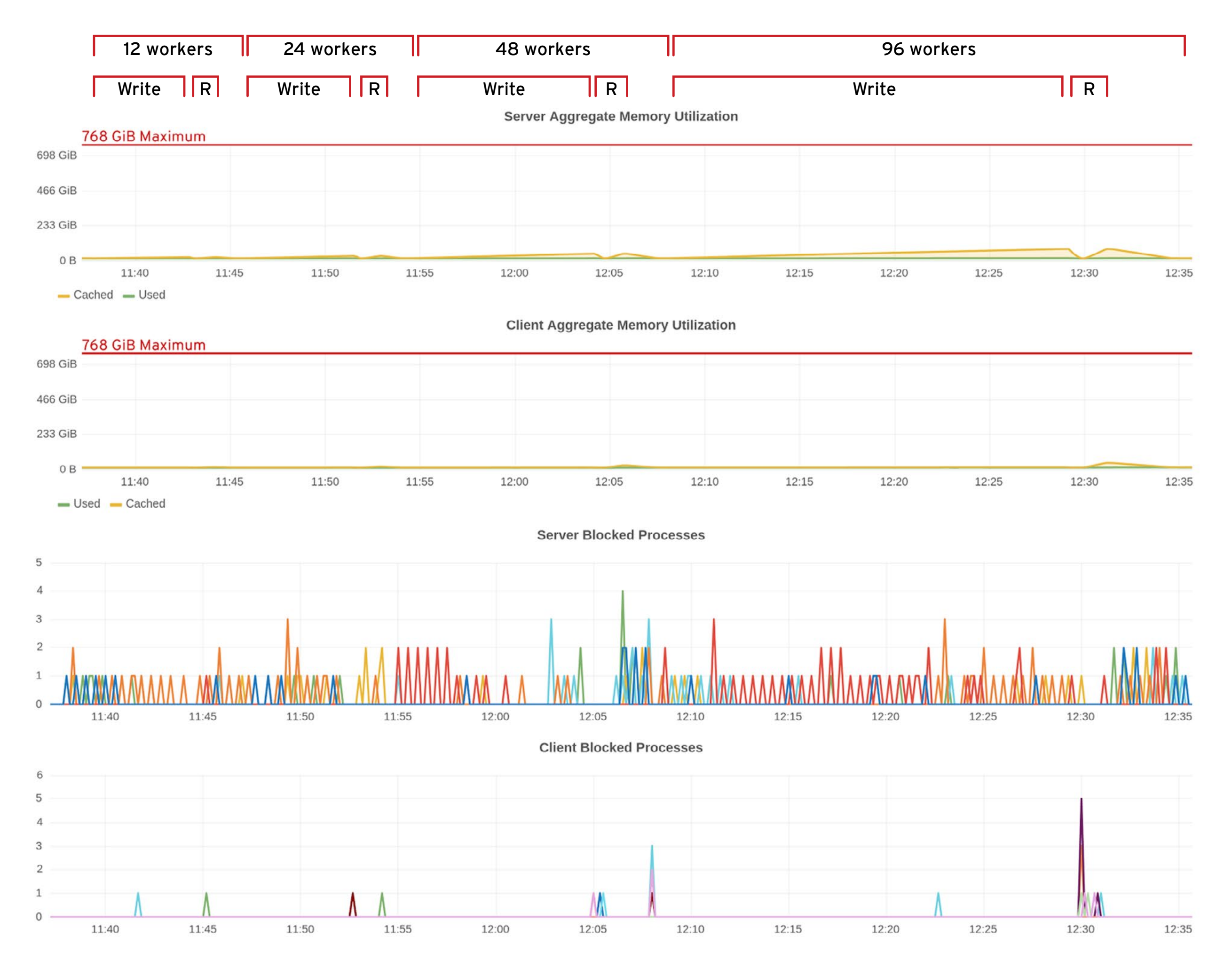


Figure 53. *Figure 53. Memory utilization and blocked processes remained low for both servers and clients.*

Figure 54 illustrates that a potential bottleneck is being approached on server disk reads as concurrency is increased to 96 workers.

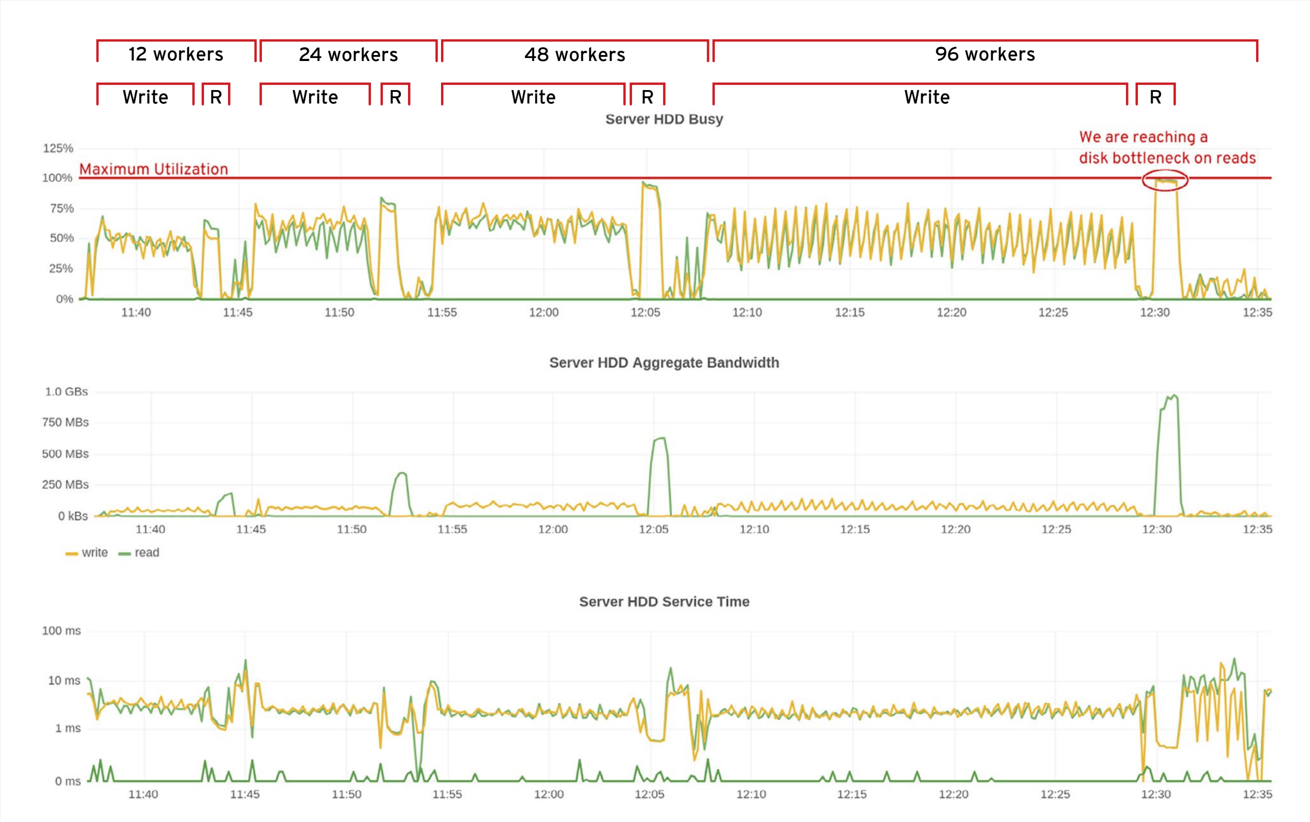


Figure 54. *Figure 54. Server disk reads for 32KB files.*

APPENDIX C: RED HAT GLUSTER STORAGE 3.2 VERSUS 3.1.3

With the release of Red Hat Gluster Storage 3.2, a number of software enhancements promise to improve upon the performance capabilities of many workloads. Some of the expected improvements are outside the scope of this study, including CIFS protocol enhancements and faster directory and file lookup operations. However, some of the changes are expected to affect the smaller file workloads considered in this study.

Red Hat re-ran a subset of the test cycles relevant to the expected improvements on the Red Hat Gluster Server 3.2 release candidate code, and the results are reported in Figure 55. Findings for 32KB and 4MB streaming file tests showed that while file reads are mostly unaffected by the changes, file writes benefit by an average of 18%, with the largest portion of that improvement reported for 32KB files at 34%. This result is a significant performance improvement for an area of the study where results were weakest and the bottleneck pointed to the Gluster implementation. These results demonstrate the value of the ongoing effort and commitment of Red Hat engineering toward improving product quality and performance for open software-defined storage.

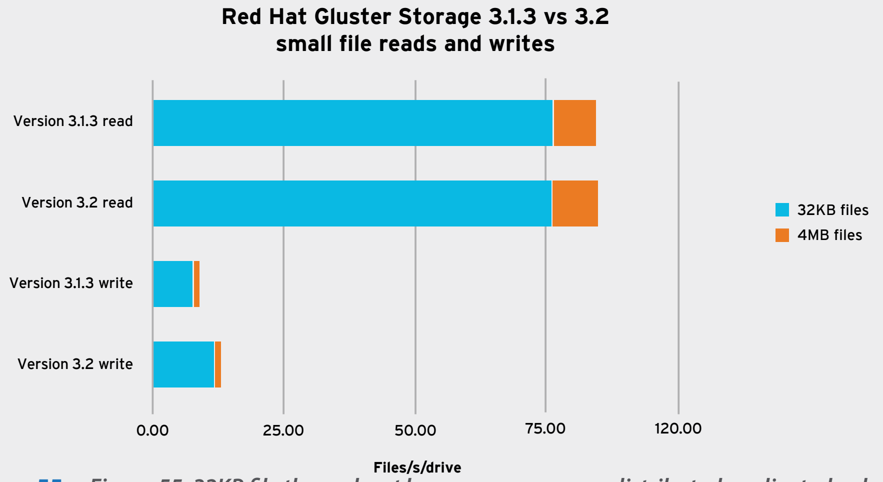


Figure 55. *Figure 55. 32KB file throughput by concurrency on a distributed-replicated volume (6x2) on RAID 6 bricks.*

APPENDIX D: SUPERMICRO BILL OF MATERIALS

This section provides Gluster-optimized server bill of materials (BOMs) available from Supermicro. Table 3 provides the BOM for the 12-disk standard Supermicro server optimized for Red Hat Gluster Storage.

Table 3. *Supermicro 12-disk standard server BOM.*

CATEGORY	PART NUMBER	DESCRIPTION	QUANTITY
Gluster	SSG-6028R-E1CR12H	X10DRH-IT, CSE-826BTS-R920LPBP-1	1
	P4X-DPE52630V4-SR2R7	BDW-EP 10C/20T E5-2630V4 2.2G 25M 8GT 85W R3 2011 R0	2
	MEM-DR416L-SL02-ER24	16GB DDR4-2400 1Rx4 LP ECC RDIMM,HF,RoHS/REACH	4
CV	BTR-TFM8G-LSICVM02	LSI Supercap w 8GB CV 24 cable (whole kit)	1
Performance	HDS-AVM-SSDPEDMD800G4	Intel DC P3700 800GB, NVMe PCIe3.0,HET,MLC AIC 20nm ,HF,RoHS/REACH	1
	BKT-BBU-BRACKET-05	Remote Mounting Bracket for LSI BBUs	1
	MCP-220-82609-0N	Rear 2.5 x 2 Hot swap HDD kit for 216B/826B/417B/846X/847B	1
OS	HDS-2TM-SSDSC2BB240G6	Intel DC S3510 240GB, SATA 6Gb/s, MLC 2.5” 7.0mm, 16nm ,HF,RoHS/REACH	2
	AOC-STGN-I2S-P	STD Dual-port 10G Ethernet with SFP+ W/ CDR	1
	SFT-OOB-LIC	License key for enabling OOB BIOS management	1
	HDD	Seagate 3.5”, 8TB, SATA3.0, 7.2K RPM, 256M, 512E	12

Table 4 lists the BOM for the 24-disk dense Supermicro server optimized for Red Hat Gluster Storage.

Table 4. *Supermicro 24-disk dense server BOM.*

CATEGORY	PART NUMBER	DESCRIPTION	QUANTITY
Gluster	SSG-6028R-E1CR24N	X10DSC+, 826STS-R1K62P1	1
	P4X-DPE52650V4-SR2N3	E5-2650 V4	2
	MEM-DR416L-SL02-ER24	16GB DDR4-2400 1Rx4 LP ECC RDIMM,HF,RoHS/REACH	8
CV	BTR-CV3108-FT1	LSI 3108 CacheVault kit	1
Performance	HDS-AVM-SSDPEDMD800G4	Intel DC P3700 800GB, NVMe PCIe3.0,HET,MLC AIC 20nm ,HF,RoHS/REACH	2
OS	HDS-2TM-SSDSC2BB240G	Intel DC S3510 240GB, SATA 6Gb/s, MLC 2.5” 7.0mm, 16nm ,HF,RoHS/REACH	2
	AOC-STGN-I2S-P	STD Dual-port 10G Ethernet with SFP+ W/ CDR	1
	SFT-OOB-LIC	License key for enabling OOB BIOS management	1
	HDD	Seagate 3.5”, 8TB, SATA3.0, 7.2K RPM, 256M, 512E	24

About the Author

Dustin Black is a Senior Architect, Software Defined Storage for Red Hat. Dustin wishes to thank and acknowledge Annette Clewett, Brent Compton, Narendra Narang, Manoj Pillai, Barry Marson, Ben England, Paul McLeod, Philip Noyes and Eric Liefeld for their contributions to this work.

About Intel

Intel makes possible the most amazing experiences of the future. You may know us for our processors. But we do so much more. Intel invents at the boundaries of technology to make amazing experiences possible for business and society, and for every person on Earth. Harnessing the capability of the cloud, the ubiquity of the Internet of Things, the latest advances in memory and programmable solutions, and the promise of always-on 5G connectivity, Intel is disrupting industries and solving global challenges. Leading on policy, diversity, inclusion, education, and sustainability, we create value for our stockholders, customers, and society.

www.intel.com

About Red Hat

Red Hat is the world's leading provider of open source software solutions, using a community-powered approach to provide reliable and high-performing cloud, Linux, middleware, storage and virtualization technologies. Red Hat also offers award-winning support, training, and consulting services. As a connective hub in a global network of enterprises, partners, and open source communities, Red Hat helps create relevant, innovative technologies that liberate resources for growth and prepare customers for the future of IT. Learn more at http://

www.redhat.com.

About Super Micro Computer, Inc.

Supermicro® (NASDAQ: SMCI), the leading innovator in high-performance, high-efficiency server technology is a premier provider of advanced server Building Block Solutions® for Data Center, Cloud Computing, Enterprise IT, Hadoop/Big Data, HPC and Embedded Systems worldwide. Supermicro is committed to protecting the environment through its "We Keep IT Green®" initiative and provides customers with the most energy-efficient, environmentally-friendly solutions available on the market.

www.supermicro.com

Printed in USA ♻ Please Recycle 14_Redhat_170731_Rev01