RED HAT® STORAGE

TECHNOLOGY DETAIL

PERFORMANCE AND SIZING GUIDE

Red Hat Gluster Storage on QCT servers

QCT (Quanta Cloud Technology) offers a family of servers for building different types of scale-out storage clusters based on Red Hat Gluster Storage—each optimized to suit different workload and budgetary needs.

Red Hat Gluster Storage offers a range of distributed file storage solutions, supporting both standard and dense storage server configurations.

Extensive Red Hat and QCT testing helps take the risk out of deploying scale-out storage solutions based on Gluster.

facebook.com/redhatinc
@redhatnews
linkedin.com/company/red-hat

redhat.com

ABSTRACT

As a software-defined scale-out storage solution, Red Hat® Gluster Storage has emerged as a compelling platform for distributed file services in the enterprise. Those deploying Gluster can benefit from having simple cluster configurations, optimized for different file service loads. For example, workloads based on small files can often benefit from different kinds of underlying hardware than workloads based on large files. To address the need for performance and sizing guidance, Red Hat and QCT (Quanta Cloud Technology) have performed extensive testing to characterize optimized configurations for deploying Red Hat Gluster Storage on several QCT servers.

TABLE OF CONTENTS

Note: The recommendations in this guide pertain to Red Hat Gluster Storage release 3.1.2. Future enhancements may alter performance and corresponding recommendations.

INTRODUCTION

With the rapidly escalating need for distributed file storage, enterprises of all kinds are seeking to emulate efficiencies achieved by public cloud providers — with their highly successful software-defined cloud data-center models based on standard servers and open source software. At the same time, the $35 billion storage market is undergoing a fundamental structural shift, with storage capacity returning to the server following decades of external network-attached storage (NAS) and storage area network (SAN) growth[1]. Software-defined scale-out storage has emerged as a viable alternative, where standard servers and independent software unite to provide data access and highly available services across the enterprise.

The combination of QCT servers and Red Hat Gluster Storage software squarely addresses these industry trends, and both are already at the heart of many public cloud datacenters[2]. QCT is reinventing data-center server technology to boost storage capacity and density, and redesigning scalable hardware for cloud applications. As the world's largest enterprise software company with an open source development model, Red Hat has partnered with many Fortune 100 companies to provide Gluster storage software in production environments. Together, QCT servers and Red Hat Gluster Storage provide software-defined storage solutions for both private and public clouds, helping to accelerate the shift away from costly proprietary external storage solutions.

Proprietary hardware-based storage segregates information, making it hard to find, access, and manage. Moreover, adding capacity to traditional storage systems often disrupts access to data. If hardware fails, it can bring the business to a standstill. In contrast, Red Hat Gluster Storage is open, software-defined file storage that scales out. Organizations can easily and securely manage large, unstructured, and semi-structured data at a fraction of the cost of traditional, monolithic storage. Importantly, only Red Hat lets organizations deploy the same storage services on premise; in private, public, or hybrid clouds; and in Linux® containers.

Running Red Hat Gluster Storage on QCT servers provides open interaction with a community-based software development model, backed by the 24x7 support of the world's most experienced open source software company. Use of standard hardware components helps ensure low costs, while QCT's innovative development model enables organizations to iterate more rapidly on a family of server designs optimized for different types of Gluster workloads. Unlike monolithic scale-up storage solutions, Red Hat Gluster Storage on QCT servers lets organizations scale out flexibly, with the ability to scale storage performance and capacity independently, depending on the needs of the application and the chosen storage server platform.

---

1 *IDC Worldwide Quarterly Disk Storage Systems Tracker, June 5, 2015*
2 *vault2016.sched.org/event/68kA/glusterfs-facebook-richard-warning-facebook*

WORKLOAD-OPTIMIZED DISTRIBUTED FILE SYSTEM CLUSTERS

One of the benefits of scale-out storage solutions is their ability to be tailored to different workloads. Red Hat Gluster Storage on QCT servers can be easily optimized and sized to serve specific workloads through a flexible choice of systems and components. By carefully choosing and configuring underlying server hardware, Red Hat Gluster storage can be easily configured to serve different kinds of file storage. 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 just a bunch of disks (JBOD) mode), the data protection scheme (replication or erasure coding), and the storage architecture (standalone or tiered storage).

- Replicated volumes on RAID 6 bricks are commonly used for performance-optimized configurations, independent of file size.
- Erasure-coded volumes on JBOD bricks are often more cost-effective for large-file archive situations.
- Standard servers are often more performant and cost-effective for smaller clusters and all small-file applications, while dense storage servers are often more cost-effective for larger clusters.
- Depending on file size, tiering with either solid state drives (SSDs) or NVM Express (NVMe) SSDs installed in storage servers can provide significant benefits, especially for read performance.

Table 1 provides generally optimal Red Hat Gluster Storage pool configuration recommendations. 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. The Performance Summary section includes more detailed information on Red Hat and QCT testing.

TABLE 1. GLUSTER POOL OPTIMIZATION CRITERIA.

<table>
<tr><th>OPTIMIZATION CATEGORY</th><th>SMALL POOLS (250TB)</th><th>MEDIUM POOLS (1PB)</th></tr>
<tr><td>SMALL-FILE PERFORMANCE</td><td colspan="2">Standard storage servers
2x replicated volumes
RAID 6 storage
1x NVMe hot tier or 4x SSD hot tier</td></tr>
<tr><td>LARGE-FILE PERFORMANCE</td><td>Standard storage servers
2x replicated volumes
RAID 6 storage
1x NVMe hot tier or 4x SSD hot tier</td><td>Dense storage servers
2x replicated volumes
RAID 6 storage
2x NVMe hot tier</td></tr>
<tr><td>LARGE-FILE ARCHIVE (WRITE MOSTLY)</td><td>Standard storage servers
erasure-coded volumes
JBOD storage</td><td>Dense storage servers
erasure-coded volumes
JBOD storage</td></tr>
</table>

REFERENCE ARCHITECTURE ELEMENTS

The following sections discuss the overall architecture of the Red Hat and QCT reference architecture, as well as key technical aspects of the principal components.

RED HAT GLUSTER STORAGE

Red Hat Gluster Storage is a software-defined, open source solution that is designed to meet unstructured, semi-structured, and big data storage requirements. At the heart of Red Hat Gluster Storage is an open source, massively scalable distributed file system 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 independently, both in terms of capacity and in terms of performance. 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.** Today's 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. Organizations that need very large amounts of storage can deploy massive scale-out storage from the outset.
- **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 commodity disk drives and 10+ Gigabit Ethernet 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 customers that are disk space conscious and would like integrated data protection without replication or RAID 6, Gluster also supports erasure coding that also provides for faster rebuild times. 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, NFS, and Hypertext Transfer Protocol (HTTP). 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 interruption to client acess.
- **Rapid and random access to archive tiers**. Unlike archival systems based on tape, Red Hat Gluster Storage offers automated data movement between hot and cold tiers. Additionally, archived data can be both accessed and recovered rapidly from the cold disk tier.
- **Persistent storage for containerized applications**. Containerized applications need persistent storage. Red Hat has announced container-native storage capabilities integrated with Red Hat OpenShift Container Platform, allowing converged containerized applications and storage.

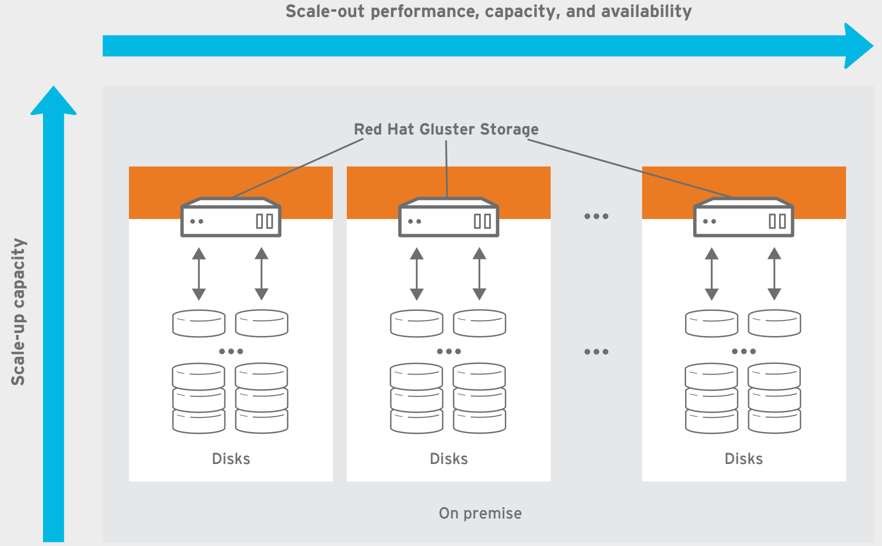

Figure 1. Red Hat Gluster Storage combines server and storage resources in a centrally managed pool with independent capacity and performance scalability.

From a technical perspective, Red Hat Gluster Storage provides distinct advantages over other technologies, including:

- **Software-defined storage.** Red Hat believes that 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.** 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.
- **Complete storage operating system stack.** Red Hat Gluster Storage delivers more than just a distributed file system. The complete storage solution adds distributed memory management, I/O scheduling, software RAID, self-healing, local N-way synchronous replication, and asynchronous long-distance replication via Red Hat Gluster Geo-Replication.
- **User space.** 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.

- **Data stored in native formats.** With Red Hat Gluster Storage, data is stored on disk using native formats (XFS) with various self-healing processes established for data. As a result, the system is extremely resilient and files will always stay naturally readable, even without the Red Hat Gluster Storage software. There is no proprietary or closed format used for storing file data.
- **No metadata 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.

QCT SERVERS FOR GLUSTER

Scale-out storage requires capable and scalable server platforms that can be selected and sized to meet the needs of specific workloads. Driven by social media, mobile applications and the demands of hyperscale datacenters, storage servers and storage platforms must provide increasing capacity and performance to store increasing volumes of data with ever-longer retention periods. QCT servers are offered in a range of configurations to allow optimization for diverse application workloads. Servers range from standard single rack unit (1U) systems to dense models providing massive storage capacity using only four rack units. Servers enable the addition of SSDs and NVMe SSDs for tiered implementations. Illustrated in Figure 2, Red Hat and QCT tested two QCT servers optimized for Gluster workloads:

- **QCT QxStor RGT-200 or RGC-200 "standard" servers.** Based on the QCT D51PH-1ULH or D51B-2U servers, these configurations are delivered with 12 hot-swappable disk drives. The D51PH-1ULH supports an additional four hot-swappable SSDs in an ultra-compact one rack unit (1U) package without sacrificing space for 12 disk drives. The D51B-2U is designed with complete features to suit demanding workloads with flexibility. It provides expansion slots for NVMe SSDs in addition to 12 HDDs. The innovative hot swappable drive design of the D51PH-1ULH and D51B-2U servers means that no external cable management arm is required — significantly reducing system deployment and rack assembly time. As a result, IT administrators can service drives with minimal effort or downtime.
- **QCT QxStor RCT-400 or RGC-400 "dense" servers.** Based on the QCT T21P-4U dual-node server capable of delivering up to 620TB of storage in just one system, these servers efficiently serve the most demanding cloud storage environments. The servers maximize storage density to meet the demand for growing storage capacity in hyperscale datacenters. Two models are available: a single storage node can be equipped with 78 hard disk drives (HDDs) to achieve ultra-dense capacity and low cost per gigabyte, or the system can be configured as dual nodes, each with 35 HDDs to optimize rack density. Along with support for two PCIe Gen3 slots for PCIe-based SSDs, the server offers flexible and versatile I/O expansion capacity. The servers feature a unique, innovative screw-less hard drive carrier design to let operators rapidly complete system assembly, significantly reducing deployment and service time.

See Appendix A for configuration details of these Gluster-ready server configurations.

QCT QuantaGrid D51PH-1ULH   QCT QuantaPlex T21P-4U

Figure 2. QCT storage servers are ideal for running Red Hat Gluster Storage.

GLUSTER DISTRIBUTED FILE SYSTEM ARCHITECTURE OVERVIEW

Storage strategies are undergoing tremendous change, particularly as organizations deploy infrastructure to support big data and private clouds. Traditional scale-up arrays are limited in scalability, and complexity can compromise cost-effectiveness. In contrast, scale-out storage infrastructure based on commodity servers has emerged as a way to deploy cost-effective and manageable storage at scale. The Gluster distributed file system architecture provides built-in flexibility and a building block approach to constructing and configuring storage pools customized to specific needs.

STANDARD AND DENSE QCT SERVERS

Red Hat Gluster Storage is able to run on multiple industry-standard hardware configurations. The purpose of this document is to help organizations evaluate key architectural concepts with corresponding test results in order to architect appropriately sized and optimized Red Hat Gluster Storage pools on QCT servers. To this end, Red Hat and QCT architects conducted extensive Gluster testing on various configurations of two QCT servers.

- **QCT QuantaGrid D51PH-1ULH or D51B-2U server (standard server).** Ideal for smaller-capacity pools, the compact 1U QuantaGrid D51PH-1ULH server provides 12 hot-swappable disk drives and support for four additional hot-swappable solid state drives (SSDs). The 2U QuantaGrid D51B-2U server provides 12 hot-swappable disk drives and NVMe SSDs.
- **QCT QuantaPlex T21P-4U server (dense server).** The QuantaPlex T21P-4U server is configurable as a single-node (up to 78 HDDs) or dual-node system (up to 35 HDDs per node), maximizing storage density to meet the demand for growing storage capacity.

GLUSTERFS VOLUMES AND BRICKS

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

- **RAID 6.** Red Hat Gluster Storage has traditionally used RAID 6 for local back-end storage. RAID 6 aggregates read performance across multiple disks and also protects against the loss of up to two physical disks within the server.
- **JBOD.** JBOD back ends are increasingly popular, especially for large-capacity scenarios. JBOD also mitigates the risk of proprietary hardware RAID implementations on individual servers.

CLIENT TYPES

Figure 3 illustrates how bricks are combined into volumes within a Red Hat Gluster Storage pool. Individual clients can access volumes within the storage pool using these supported protocols:

- **NFS (the Network File System)**. Originally developed by Sun Microsystems, Inc., NFS allows a user on a client computer to access files over a network, much like local storage is accessed.
- **CIFS (the Common Internet File System)**. Commonly used with Microsoft Windows clients, CIFS is an enhanced version of the Microsoft Server Message Block (SMB) protocol, and one of the standard ways that computer users share files across corporate intranets and the Internet.
- **Gluster Native Client**. The Gluster Native Client utilizes FUSE (filesystem in user space), a loadable kernel module for UNIX-like operating systems that lets non-privileged users create their own file systems without editing kernel code
- **OpenStack® Swift**. GlusterFS exposes an OpenStack Swift-compatible API, allowing organizations to use Swift to store large volumes of data efficiently using a highly-available, distributed, and eventually consistent object/blob store.

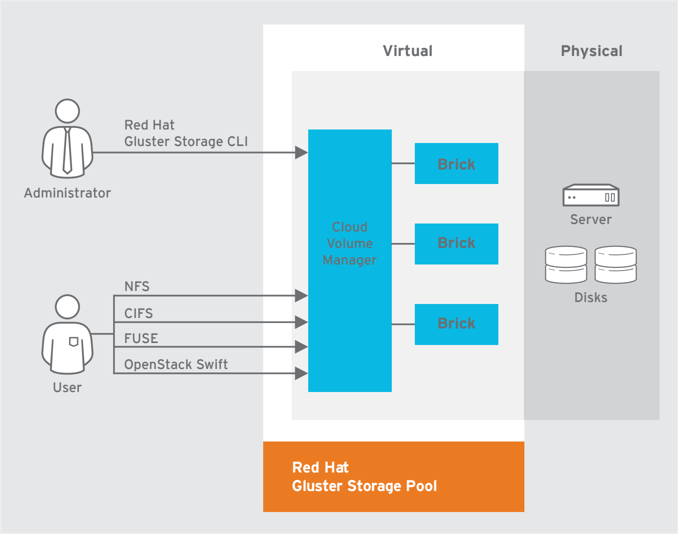

Figure 3. Administrators and a range of clients access glusterFS bricks via the Cloud Volume Manager.

GLUSTERFS VOLUME TYPES

Red Hat Gluster Storage supports several different types of volumes, allowing it to be tailored to support the specific needs of individual applications. GlusterFS volume definitions include:

- Distributed volumes
- Distributed-replicated volumes
- Dispersed (erasure-coded) volumes
- Distributed-dispersed volumes

Given the flexibility of the Red Hat Gluster Storage platform, these volume types can then be combined to create tiered storage architectures as well. Each of these volume types is described in the sections that follow.

Distributed volumes

Distributed volumes spread files across the bricks in the volume. As shown in Figure 4, individual files may be located on any brick in the distributed volume. Importantly, distributed volumes can suffer significant data loss during a disk or server failure, since directory contents are spread randomly across the bricks in the volume. As such, distributed volumes should only be used where scalable storage and redundancy are either not important, or are provided by other hardware of software layers.

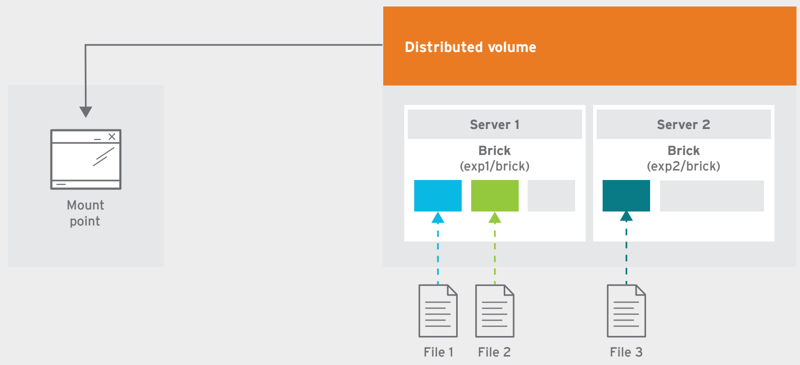

Figure 4. Distributed volumes are subject to data loss during disk or server failures.

Replicated volumes

Replicated volumes create copies of files across multiple bricks in the volume, replicating them between a number of servers to protect against disk and server failures. As such, replicated volumes are suitable for environments where high availability and high reliability are critical. Both two-way (Figure 5) and three-way (Figure 6) replicated volumes are supported in Red Hat Gluster Storage as of this writing. Three-way replicated volumes are supported only on JBOD bricks.

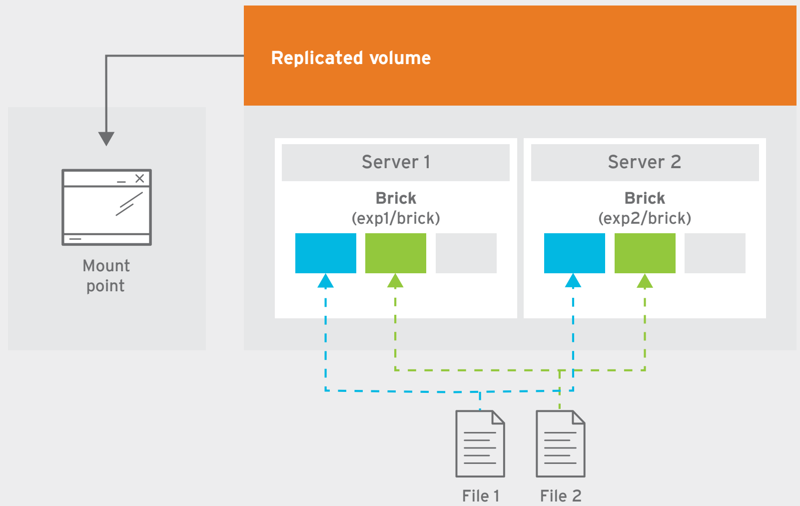

Figure 5. Replicated volumes provide high availability by creating copies on multiple servers (two-way replication shown).

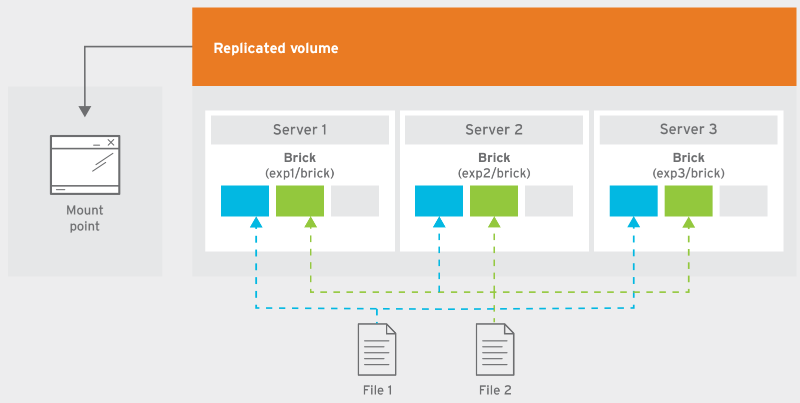

Figure 6. GlusterFS also supports three-way replication.

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. The number of bricks must be a multiple of the replica count for a replicated volume. To protect against server and disk failures, the bricks of the volume should be from different servers. Two-way distributed replicated volumes are shown in Figure 7. Synchronous three-way distributed replicated volumes (Figure 8) are now fully supported in Red Hat Gluster Storage (on JBOD bricks only).

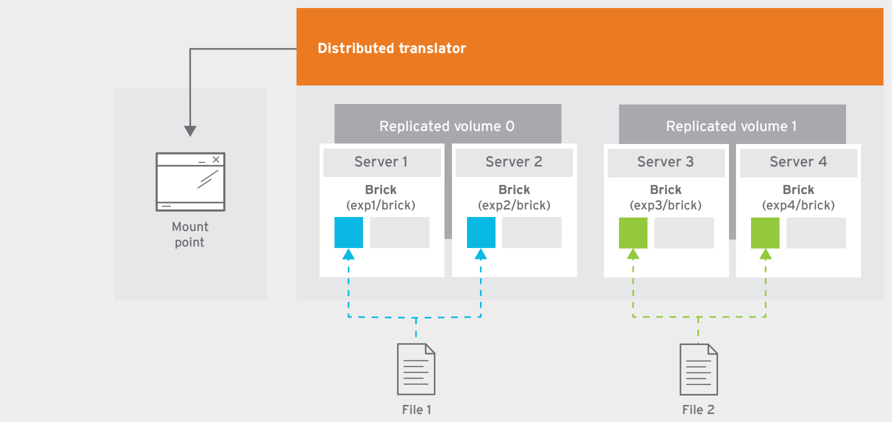

Figure 7. Distributed replicated volumes interpose a distributed translator across replicated volumes.

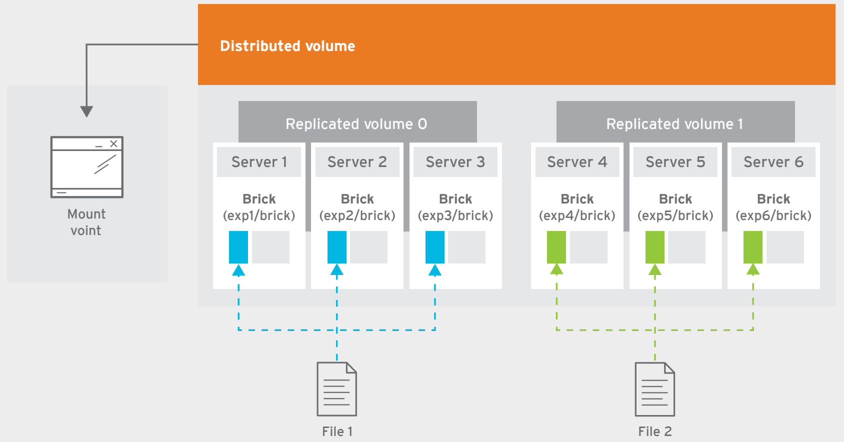

Figure 8. GlusterFS also supports three-way distributed replicated volumes.

Dispersed (erasure-coded) volumes

In Red Hat Gluster Storage, dispersed volumes are based on erasure coding — 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. Dispersed volumes allow the recovery of the data stored on one or more bricks in case of failure. The number of bricks that can fail without losing data is configurable.

Dispersed volumes (Figure 9) require less storage space when compared to replicated volumes. For this reason, they are often more cost-effective for larger capacity storage clusters. A dispersed volume sustains the loss of data based on the redundancy level. For example, a dispersed volume with a redundancy level of "2" is equivalent to a replicated pool of size two, but requires 1.5TB instead of 2TB to store 1TB of data.

The data protection offered by erasure coding can be represented in simple form by the equation **`n = k + m`**, where "n" is the total number of bricks, and the system would require any "k" bricks for recovery. In other words, the system can tolerate failure up to any "m" bricks. Red Hat Gluster Storage currently supports these dispersed volume configurations:

- 6 bricks with redundancy level 2 (4 + 2)
- 11 bricks with redundancy level 3 (8 + 3)
- 12 bricks with redundancy level 4 (8 + 4)

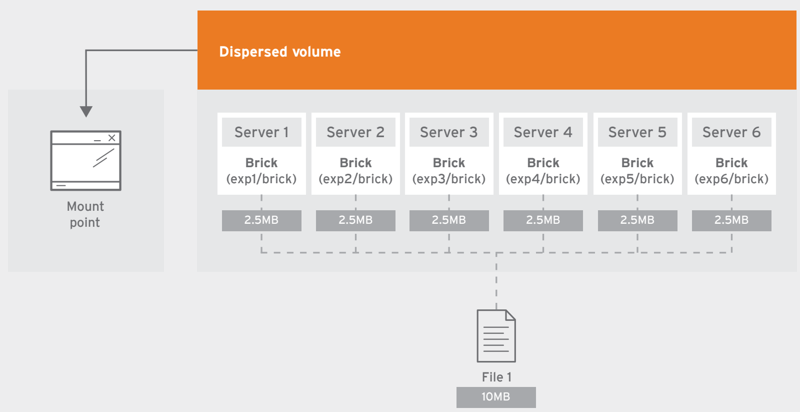

Figure 9. A six-brick dispersed volume (4 + 2).

Distributed dispersed volumes

Distributed dispersed volumes (Figure 10) support the same configurations of erasure coding as dispersed volumes. The number of bricks in a distributed dispersed volume must be a multiple of (k+m). Red Hat Gluster Storage supports these configurations of distributed dispersed volumes:

- Multiple disperse sets containing 6 bricks with redundancy level 2
- Multiple disperse sets containing 11 bricks with redundancy level 3
- Multiple disperse sets containing 12 bricks with redundancy level 4

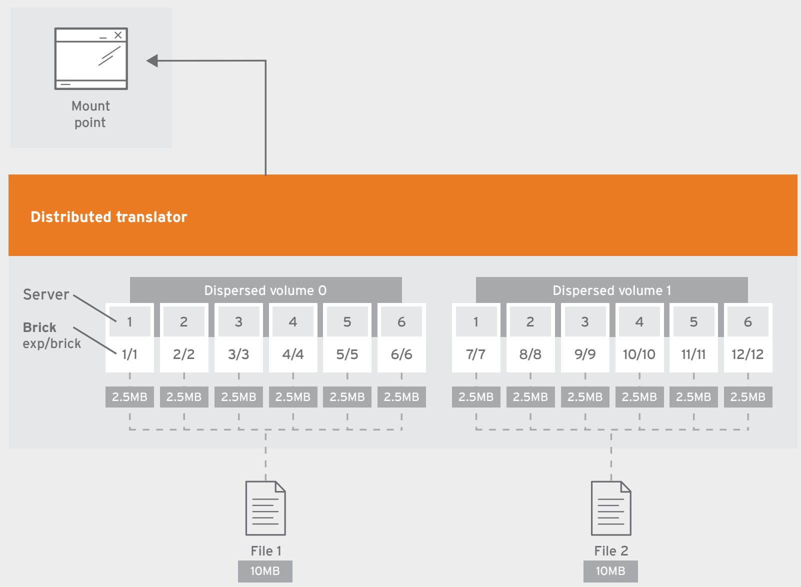

Figure 10. Distributed disbursed volumes interpose a distributed translator across two disbursed volumes.

Tiered volume configurations

Tiering can provide better I/O performance since more active data is stored in a higher-performance hot tier, with less-active data stored on a lower-performance cold tier. The hot tier is created using better performing subvolumes (e.g., SSDs). The cold tier is typically a Red Hat Gluster Storage volume created using slower storage such as spinning disks. Tiering can bring several key advantages, including:

- Automatic classification and movement of files based on the access patterns
- Faster response time and reduced latency
- Better I/O performance
- Improved data-storage efficiency
- Reduced deployment and operating costs

Tiering monitors and identifies the activity level of the data and auto rebalances the active and inactive data to the most appropriate storage tier. This process both improves the storage performance and resource use. Moving data between tiers of hot and cold storage is a computationaly expensive task. To address this, Red Hat Gluster Storage supports automated promotion and demoting of data within a volume in the background, so as to minimize the impact on foreground I/O activity.

The promotion and demotion of files between tiers is determined by how full the hot tier is. Data becomes hot or cold based on the rate at which it is accessed, relative to high and low watermarks set by the administrator. If access to a file increases, it moves, or retains its place in the hot tier. If the file is not accessed in a while, it moves, or retains its place in the cold tier. As a result, data movement can happen in either direction, based on the access frequency.

Figure 11 illustrates how tiering works. In this case, the existing slower distributed replicated volume would become a cold tier while the new faster distributed replicated tier would serve as a hot tier. Frequently-accessed files will be promoted from the cold tier to the hot tier for better performance. If that data becomes subsequently unused, it will be demoted to the cold tier.

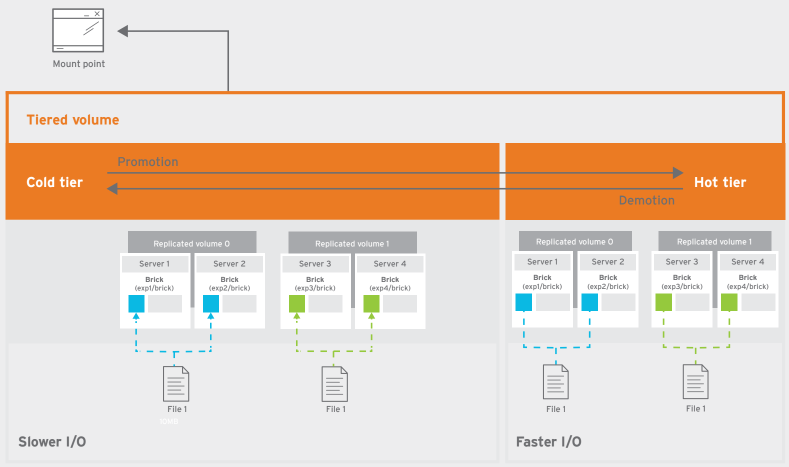

Figure 11. Tiering lets a distributed replicated volume serve as a hot tier for distributed disbursed volume.

The promotion and demotion of files is also moderated by the fullness of the hot tier. Data accumulates on the hot tier until it reaches the low watermark (default 75% of capacity), even if it is not accessed for a period of time. This prevents files from being demoted unnecessarily when there is plenty on free space on the hot tier. When the hot tier is fuller than the lower watermark but less than the high watermark (default 90% capacity), data is randomly promoted and demoted where the likelihood of promotion decreases as the tier becomes fuller. The opposite holds for demotion. If the hot tier is fuller than the high watermark, promotions stop and demotions happen more frequently in order to provide additional free space.

SIX KEY GLUSTER POOL DESIGN PRINCIPLES

Key design considerations can help organizations shape Gluster cluster and network architectures. Each of these topics is intended to be a conversation between peer architects. Later sections describe testing results that help to illustrate how some of these design considerations affect Gluster pool price/performance.

QUALIFYING THE NEED FOR A SOFTWARE-DEFINED, DISTRIBUTED FILESYSTEM

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

- **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 their storage needs to meet specific performance requirements. For performance-oriented clusters, throughput and latency requirements must be clearly defined. 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. For example, erasure-coded storage pools can provide much lower cost per terabyte than replicated pools, and are often chosen for capacity-archives.

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 will 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.[3] Furthermore, particular workloads may benefit from specific client interfaces. Small static file workloads such as PHP 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 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.

3 Note that due to locking incompatibility, CIFS cannot be used with other client access methods.

SELECTING A DATA PROTECTION METHOD

Gluster offers two data protection schemes: replication and erasure coding (dispersed volumes in Gluster parlance). As 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, dependent upon the workload.

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 this necessity extends to the scale-out storage upon which they depend. The fault domain that a single server represents is key to cluster design, so dense servers should be reserved for clusters larger than a petabyte, where the capacity of an individual server accounts for less than 15% of the total cluster capacity. This fault domain recommendation may be relaxed for less critical pilot projects or smaller clusters.

Fault domain risk includes accommodating the impact on performance. 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. During Gluster self-healing, a percentage of volume throughput capacity will be diverted to healing 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 disperse set.

Red Hat and QCT recommend these minimum pool sizes:

- **Supported minimum cluster size:** Two nodes with a third non-storage node to constitute quorum.
- **Recommended minimum pool size:** Six nodes.

TESTED CONFIGURATIONS

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

TESTING APPROACH

Red Hat and QCT testing exercised file create and read operations using file sizes of 50KB (jpeg), 5MB (mp3), and 4GB (DVD). Two separate benchmarking tools were used to exercise the glusterFS file system in Red Hat and QCT testing:

- **IOzone.** IOzone (www.iozone.org) was used to test the 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. In testing, 16 client systems each ran eight IOzone threads (128 total threads of execution) for a four-node storage cluster.
- **SmallFile.** Smallfile is a python-based distributed POSIX workload generator which can be used to measure performance for a variety of metadata-intensive workloads across an entire cluster. It 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.

QUANTAGRID D51PH-1ULH CONFIGURATION

As shown in Figure 12, six QuantaGrid D51PH-1U servers were tested with cluster and public 10 GbE networks. Each QuantaGrid server had four S3710 (200 GB) 2.5-inch SSD drives for the tiering test. The test workload was driven by 16 client nodes. Each QuantaGrid D51PH-1ULH server was configured with:

- **Processors**: Two Intel Xeon processor E5-2630 V3 8-core 2.4 GHz
- **Memory**: 4x 16 GB 2133 MHz DDR4 RDIMM 1.2V
- **SAS controller**: QCT SAS Mezz (LSI 3008 for JBOD, LSI 3108 for RAID 6)
- **Network controller**: QCT Intel 82599ES dual-port 10 GbE SFP+ OCP mezzanine, or QCT Intel X540 dual-port 10 GbE BASE-T OCP mezzanine
- **Onboard storage**: Flash/M SATADOM 32 GB
- **Hard disk drives**: 12x 3.5-inch SAS 6TB 7.2K RPM
- **Optional SSD for tiering**: 4x Intel SSD Data Center (DC) S3710 200 GB, 2.5-inch SATA 6Gb/s, MLC

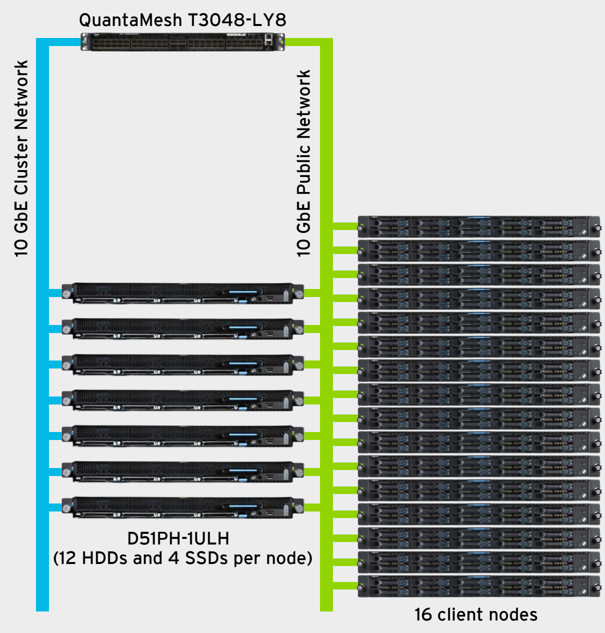

Figure 12. Six standard QuantaGrid D51PH-1ULH servers were tested with back end and public 10 GbE networks.

QUANTAPLEX T21P-4U CONFIGURATION

As shown in Figure 13, two 4U QuantaPlex T41P-4U/Dual Node servers were connected, each with dual 40 GbE interfaces (one per node) to a shared public network. Sixteen client nodes were likewise attached to the public network via 10 GbE for load generation. Each QuantaPlex T41P-4U/Dual Node server was configured with:

- **Processors**: (2x 2) Intel Xeon processor E5-2650 V3 10-core 2.3 GHz
- **Memory**: (2x 8) 16 GB 2133 MHz DDR4 RDIMM 1.2V
- **SAS controller**: (2x 1) QCT SAS Mezz LSI 3108 SAS controller
- **Network controller**: (2x 1) QCT Mellanox ConnectX-3 EN 40 GbE SFP+ single-port OCP mezzanine
- **Onboard storage**: (2x 1)Intel SSD DC S3510 120 GB, 2.5-inch SATA 6 Gb/s, MLC flash
- **Hard disk drives**: (2x 35) Seagate 3.5-inch SAS 6TB 7.2K RPM
- **Optional SSD for tiering**: (2x 2) Intel SSD DC P3700 800GB, 1/2 hight PCIe 3.0, MLC

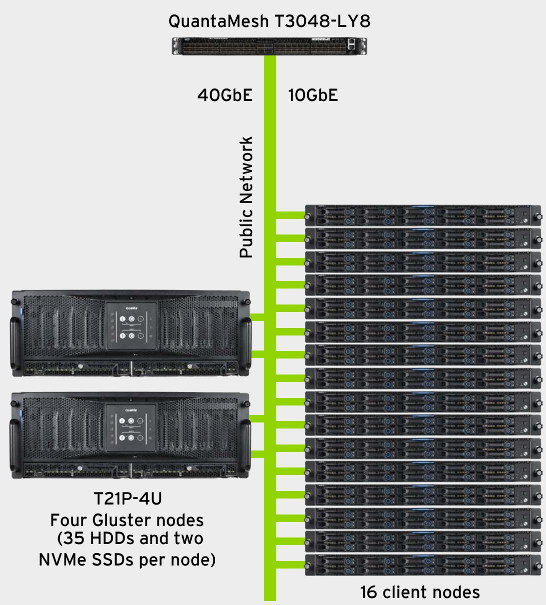

Figure 13. Two QuantaPlex T21P-4U/Dual Node servers (four Gluster nodes) were configured, each node with a 40 GbE interface, with the configuration driven by 16 client nodes.

For the tiered configuration testing, the cold tier was configured to use three RAID 6 bricks on each of the four T21P-4U servers. The hot tier was configured to use the two NVMe SSDs on each server. The Gluster volume for both the hot and cold tier are configured using two-way replication. The Gluster volume settings that are best suited for this configuration are listed in Appendix B.

SOFTWARE CONFIGURATION

Server systems were configured with this storage server software:

- Red Hat Gluster Storage 3.1.2

Client systems were configured with this software:

- Red Hat Enterprise Linux 7.1

PERFORMANCE SUMMARY

To characterize performance, Red Hat and QCT ran a series of tests across different cluster configurations, varying file sizes (jumbo, medium, and small), server density (standard vs. dense) the ways that the bricks were defined (JBOD vs. RAID6), and data protection schemes (replication vs. erasure coding). For medium and small file testing, tiered configurations were also tested, utilizing SSDs at varying degrees of fullness.

JUMBO FILES: DESIGNING FOR OPTIMAL THROUGHPUT

Figure 14 illustrates the test results for jumbo (4GB) files with 4MB sequential I/O. These tests simulate I/O for something approaching DVD video size. The read and write performance is shown for various configurations using different combinations of server density, brick formats, and data protection. Key observations are included in the sections that follow.

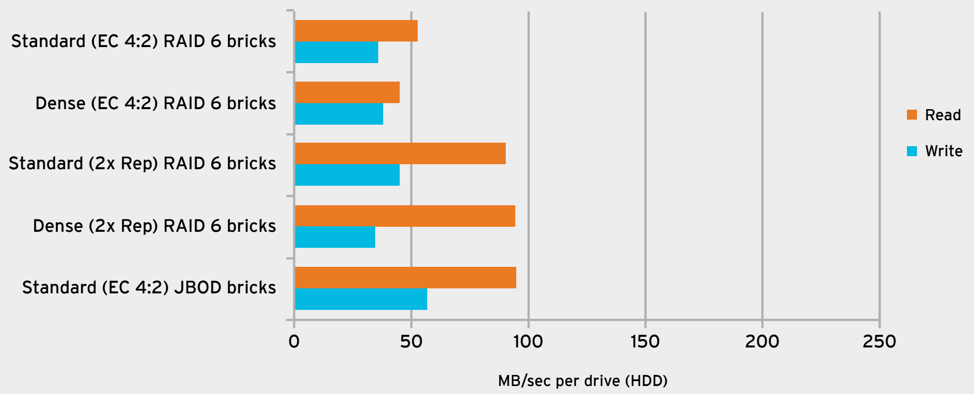

Figure 14. Performance results for jumbo (4GB) files.

Erasure-coded versus replicated volumes

For some software-defined storage solutions (notably Ceph), erasure coding performs significantly worse than replication. Remarkably, Red Hat Gluster Storage demonstrates very similar performance for jumbo files between erasure-coded and replicated volumes. Comparing the third and fifth data sets from the top of Figure 14 (standard servers, replication/RAID 6 vs. EC 4:2/JBOD), erasure coding provides roughly equivalent read and write performance compared to 2x replicated volumes over RAID 6 bricks.

JBOD versus RAID6 bricks with erasure-coded volumes

Results show large file sequential IO performance to be dramatically better on JBOD bricks vs. RAID6 bricks, when using erasure-coded data protection. This is made clear by examining the first and fifth data sets in Figure 14 (standard servers, EC4:2/RAID6 versus EC4:2/JBOD).

Dense versus standard storage server price-performance

Figure 15 shows several server configurations compared by a relative performance per dollar metric for jumbo file throughput (MBPS throughput per server divided by the cost of the server, larger bars are better). Three standard server configurations are compared to two dense server configurations to help gauge relative economic value. Of interest, standard servers with JBOD bricks and erasure coding are dramatically superior from a performance versus price perspective for both reads and writes. Note that dense servers with erasure-coded JBOD bricks are expected to perform similarly well, but were not evaluated due to a current limitation in the maximum number of JBOD bricks per server.[4]

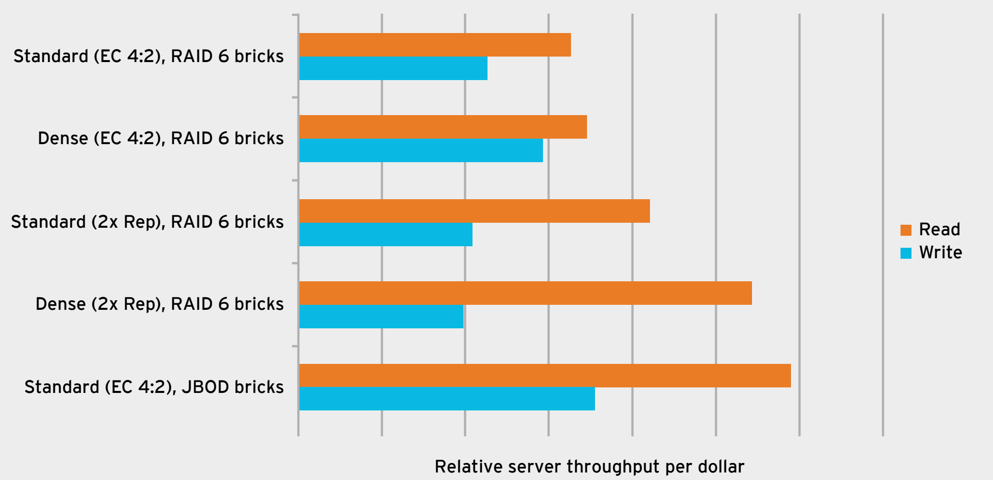

Figure 15. Standard servers with 4:2 erasure coding and JBOD bricks provide superior value in terms of throughput per server cost.

SMALL AND MEDIUM SIZED FILES: DESIGNING FOR OPTIMAL FILE OPERATIONS PER SECOND

Figure 16 shows the results of testing using small (50KB) files against a variety of configurations. Files of this size might equate to a small JPEG file. Testing on small files was done to evaluate the performance effects of various tiering configurations on reads and writes.[5]

4 This issue is currently under review by Red Hat engineering.
5 Unfortunately, a testing anomaly caused read performance results for non-tiered standard servers to be discarded. Differences between tiered and non-tiered standard servers are expected to be similar to dense servers.

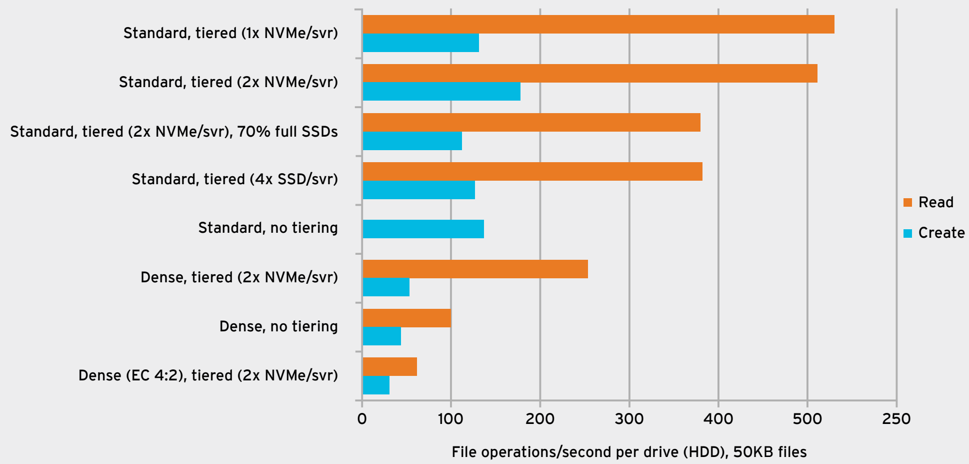

Figure 16. Small file testing utilized 50KB files against a range of tiered and non-tiered configurations.

Effects of tiering on read and write performance

As expected, the effect of tiering is quite pronounced. All tiered configurations provided dramatic performance improvements for reads over non-tiered systems. This is not surprising as the philosophy behind tiering is to provide a place for more frequently accessed (hot) files to be served from a faster storage tier. Also as expected, write performance saw no material performance improvement from tiering. As such, tiering for a write-intensive workload may not be a good investment.

Erasure-coded versus replicated volumes for cold tier

Testing also compared using an erasure-coded volume against a replicated volume for the cold tier. Importantly, testing revealed a stark contrast. Performance with a replicated cold tier provided dramatic read performance improvements over a non-tiered configuration. Note that the erasure-coded cold tier offered performance that was even lower than the non-tiered configurations. This data suggests that — with this release of Gluster — tiering should only be configured over a replicated cold tier, and not be configured over an erasure-coded cold tier.

Dense versus standard storage servers (small-sized files)

For small-sized files, testing also clearly demonstrated a performance advantage for standard servers over dense servers. All standard server configurations provided dramatically better performance for both reads and writes over dense server configurations.

Ratio of NVMe SSDs to HDDs for tiering on standard servers

Testing also evaluated the performance effect of configuring one versus two SSDs per server. While it might be tempting to expect that "more is better" in this regard, Red Hat's test results showed the opposite. Though adding two SSDs per server did raise write performance slightly over a single SSD, it reduced read performance, which is the overall goal of a tiered architecture. This finding is important as it can help organizations avoid spending money needlessly on additional SSDs with no additional performance gain for reads.

Effects of filling the hot tier on performance

Reflected in the third data set of Figure 16, the testing also evaluated the effect of starting with the hot tier at 70% capacity, just below the default 75% low watermark. As expected, both read and write performance drop materially with a substantially full hot tier. As the top tier fills, the systems begins to aggressively demote and flush to disk, more heavily utilizing the cold tier.

Optimal configurations for small file operations/second

Figure 17 shows several tiered and non-tiered server configurations compared by a relative performance per dollar metric for small file throughput (MBPS throughput per server divided by the cost of the server, larger bars are better). Multiple standard server configurations are compared to dense server configurations to help gauge relative economic value. Again, standard servers in tiered configurations demonstrate a strong advantage in terms of performance over cost. A standard tiered server with a single NVMe SSD per server provides the greatest value in terms of read throughput, while a standard server with no tiering provides the greatest value in terms of write throughput.

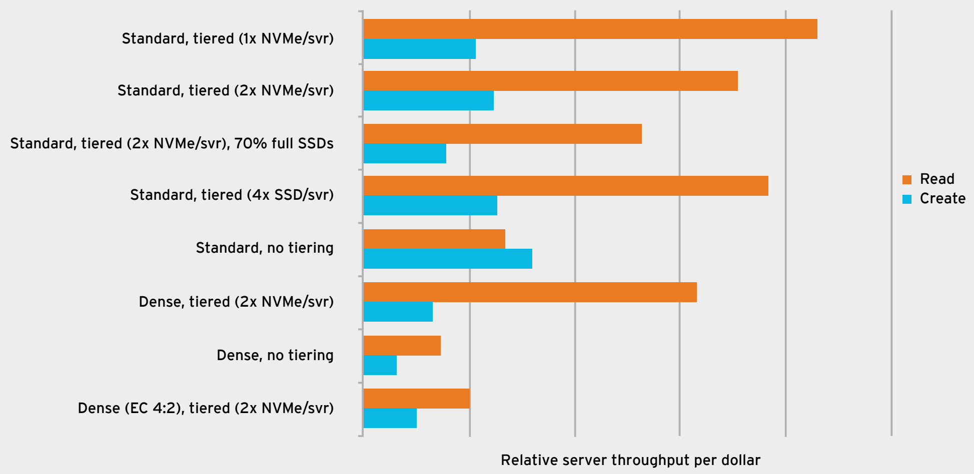

Figure 17. For small files, standard tiered servers provide the greatest value in terms of read throughput.

Dense versus standard storage servers, medium-sized files

Figure 18 illustrates the results of performance testing conducted for medium files, defined as 5MB files such as might be used for mp3 audio files. Importantly, unlike small file test results, read performance for medium files was greatest for dense servers, particularly dense servers in a tiered configuration with dual NVMe SSDs per server.

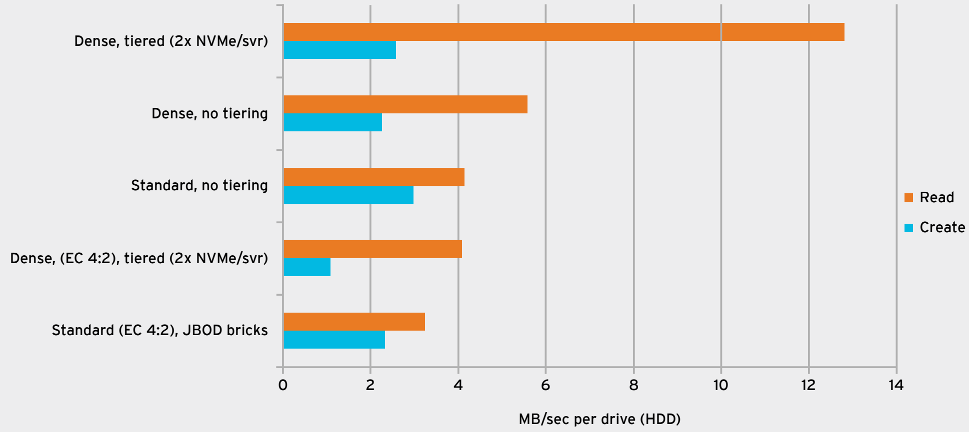

Figure 18. For medium files, dense tiered servers provide the greatest read throughput.

Optimal configurations for medium file operations per second

Figure 19 illustrates test results for medium files in terms of relative server throughput per dollar (bigger is better). For read-heavy workloads, dense tiered servers with two SSDs per server present a very pronounced advantage in terms of value. Value for write throughput is similar.

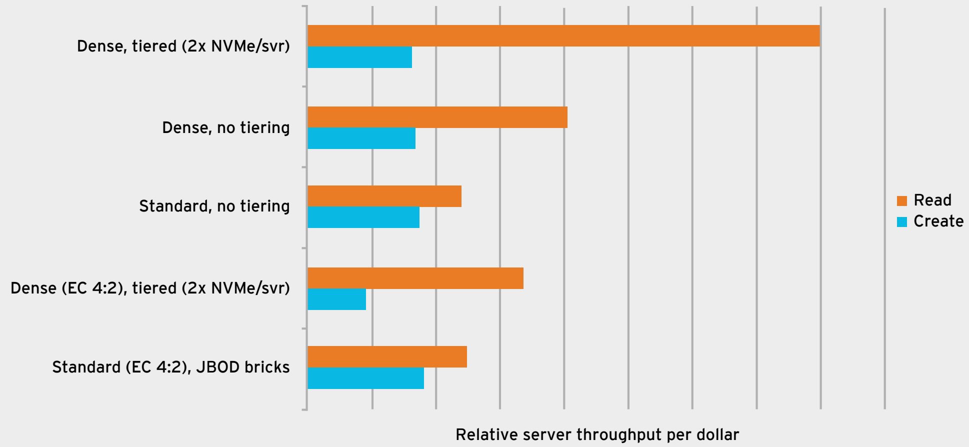

Figure 19. For medium-sized files, dense tiered servers provide the greatest value.

Performance under self-healing

GlusterFS has a built-in daemon to automate self-healing functionality in a case where a brick is taken off line and then returned to the pool. This functionality is closely related to volume rebalancing, which redistributes files as bricks are added to the pool. Self-healing does have a relative effect on overall performance, depending on the size of the pool and number of files that changed while a brick was off-line.

Figure 20 shows the impact of self-heal operation on a replicated volume. The graph shows the negative impact on read performance due to self-heal operation when the read data set includes files that are being healed. Importantly, the relative impact on performance decreases as the size of the cluster increases because each storage node now constitutes a smaller portion of the entire glusterFS volume.

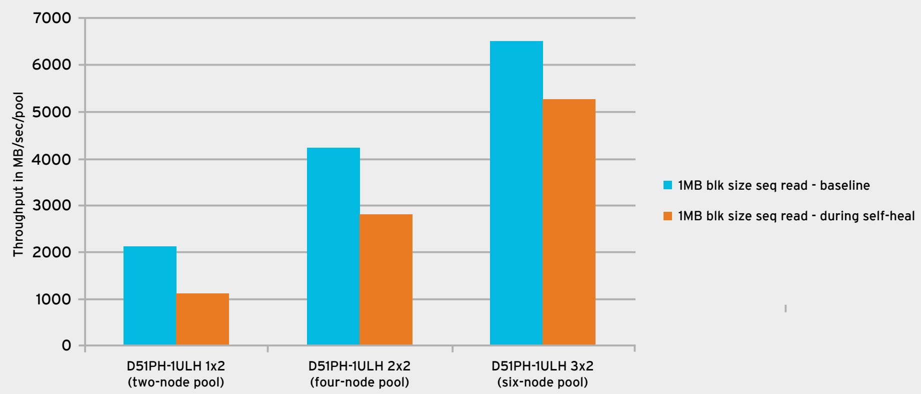

Figure 20. Self-healing on smaller cluster sizes consumes a larger percentage of overall cluster resources.

Figure 21 compares the impact of the self-heal operation on a replicated volume both when the client workload read data set is and is not included in the data to be self-healed. The graph shows that there is no negative impact from the self-heal process when the data set read does not include any of the files that are being healed. Of course, performance is also determined by the available network capacity for client communication as some of the network bandwidth will be used by the self-heal operation. Importantly, testing showed that there was no client workload performance penalty for a dispersed volume during the read after a failed node recovery, when the client workload did not access files being healed.

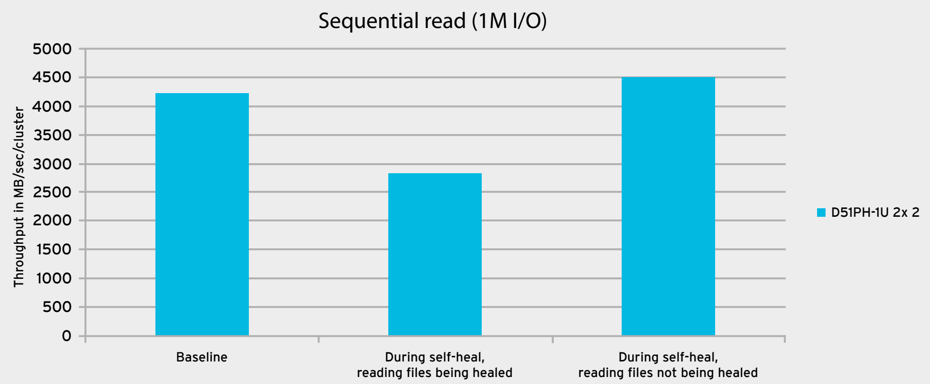

Figure 21. There is no self-heal impact if the read set is not included in the files being healed.

SUMMARY AND BEST PRACTICES

Red Hat and QCT testing has revealed that appropriate server configurations for different workloads depend strongly on the size of the files being written. Figure 22 and 23 show the relative cross over points for reads and write performance respectively for several file sizes, as tested on a standard server (QCT D51PH-1ULH). Comparing 2x replication and erasure coding (4:2), these charts demonstrate clearly that performance favors replication for smaller files and erasure coding for larger files.

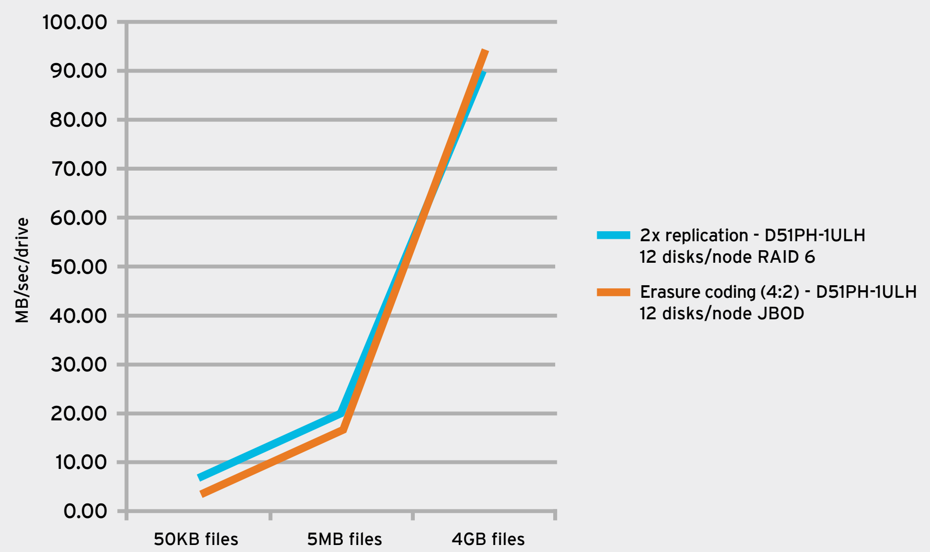

Figure 22. Read throughput and cross over across different sized files.

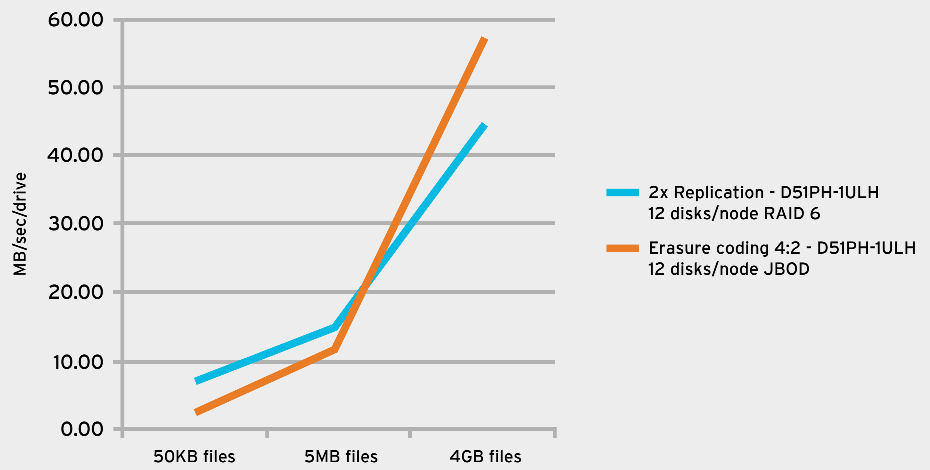

Figure 23. Write throughput and cross over across different file sizes.

Overall, Red Hat and QCT testing has revealed a number of best practices for serving different sizes of files.

- **Small (50KB) files.** Tiering provides a pronounced advantage for read-heavy workloads with hot-file access patterns. Standard servers in a tiered configuration with a single SSD per server provide both superior performance and value over dense servers. The cold tier should be replicated, not erasure-coded, and adding additional SSDs for the hot tier is generally not necessary or cost-effective.
- **Medium (5MB) files.** For medium-sized files, dense tiered servers with two SSDs per server provide the best read performance and comparable write performance to other configurations. These servers also provided the best value in terms of performance over cost.
- **Jumbo (4GB) files.** Unlike other software-defined storage solutions, erasure-coded and replicated volumes provide similar performance for jumbo-sized files. Erasure coding should be used on JBOD bricks only, as RAID6 combined with erasure coding provides inferior performance. From a value perspective, standard servers with erasure coding and JBOD bricks represent a superior value.

APPENDIX A: GLUSTER-OPTIMIZED BUILDING BLOCKS FROM QCT

To help provide simplified deployment choices for Red Hat Gluster Storage, QCT is offering several configurations (SKUs) optimized as Gluster servers. These configurations provide different controllers to help optimize them for different use cases.

- **QxStor RGT-200(SF).** Based on the QCT D51PH-1ULH or D51B-2U 12-bay server, this configuration provides 12 HDDs and support for four SATA SSDs or one NVMe SSD. The configuration is best suited when utilizing the Gluster tiering feature for small file workloads. The storage server includes LSI 3108 RAID controller to support RAID 6 configuration for the HDDs.
- **QxStor RGT-200(LF).** Based on the QCT D51PH-1ULH or D51B-2U 12-bay server, this configuration provides 12 HDDs and LSI 3108 RAID controller to support RAID 6 configuration for the HDDs. The configuration offers high throughput and reliability with a small failure domain.
- **QxStor RGC-200.** Based on the QCT D51PH-1ULH or D51B-2U 12-bay server, this configuration provides 12 HDDs and an LSI 3008 RAID controller to support JBOD configuration for the HDDs. This configuration is best suited for archival workloads of medium capacity where cost is more important than performance.
- **QxStor RGT-400(LF).** Based on the QCT T21P-4U dual-node server with 35 drive bays for each node, this configuration offers an LSI 3108 RAID controller to support RAID 6 configuration for the HDDs. The configuration provides the capability to scale to a half-petabyte of raw capacity in only four rack units. Organizations can obtain the best throughput and density simultaneously with this configuration.
- **QxStor RGC-400**. Based on QCT T21P-4U dual-node server with 35 drive bays for each node, this configuration offers an LSI 3008 RAID controller to support JBOD[6] configuration for the HDDs. The configuration provides the capability to scale to a half-petabyte of raw capacity in only four rack units. This configuration is most cost-effective for deployments that are a petabyte in size or larger.

These Gluster-optimized configurations can be used as building blocks to construct clusters in a wide range of sizes, focused at serving either small or large files as shown in Table 2.

6 The 24 JBOD drive per server support limitation in Red Hat Gluster Storage 3.1.2 is under study for future releases.

TABLE 2. WORKLOAD OPTIMIZED QCT QXSTOR CONFIGURATIONS.

SHARED FILE WORKLOADS	SMALL (UP TO 500TB)	MEDIUM (GREATER THAN 1PB)	LARGE (GREATER THAN 2PB)
THROUGHPUT OPTIMIZED (SMALL FILE PERFORMANCE)	14x QxStor RGT-200(SF) • 2x replicated volume • 12x 8TB HDDs, RAID 6 • 4x SSD or 1x NVMe hot tier • 1x dual-port 10GbE • 2x Intel Xeon E5-2620 v4 • 64GB memory • Server node quantity: 14x (560TB usable)		• NA
THROUGHPUT-OPTIMIZED (LARGE FILE PERFORMANCE)	14x QxStor RGT-200(LF) • 2x replicated volume • 12x 8TB HDDs, RAID 6 • 1x dual-port 10GbE • 2x Intel Xeon E5-2620 v4 • 64GB memory • Server node quantity: 14x (560TB usable)	5x QxStor RGT-400(LF) • 2x replicated volume • 2x 35x 8TB HDDs, RAID 6 • 2x single-port 40GbE • 2x 2 Intel Xeon E502650 v4 • 128GB memory • Server node quantity: 10 (1.1PB usable)	
COST/CAPACITY-OPTIMIZED (LARGE FILE ARCHIVE)	6x QxStor RGC-200 • Erasure-coded volume • 12x 8TB HDDs, JBOD • 1x dual-port 10GbE • 2x Intel Xeon e5-2620 v4 • 64GB memory • Server node quantity: 6 (384TB usable)	3x QxStor RGC-400 • Erasure-coded volume • 2x 35x 8TB HDDs, JBOD • 2x dual-port 10GbE • 2x 2 Intel Xeon E5-2650 v4 • 128GB memory • Server node quantity: 6 (1.1PB usable)	

APPENDIX B: TIERED GLUSTER VOLUME SETTINGS

The Gluster volume settings that follow were used in configuring the T21P-4U servers tiered configuration testing.

First, the replicated Gluster volume was created for the cold tier:

```
gluster volume create gvol62 replica 2 \
  qct135:/gluster/brick1/gvol62 \
  qct136:/gluster/brick1/gvol62 \
  qct137:/gluster/brick1/gvol62 \
  qct138:/gluster/brick1/gvol62 \
  qct135:/gluster/brick2/gvol62 \
  qct136:/gluster/brick2/gvol62 \
  qct137:/gluster/brick2/gvol62 \
  qct138:/gluster/brick2/gvol62 \
  qct135:/gluster/brick3/gvol62 \
  qct136:/gluster/brick3/gvol62 \
  qct137:/gluster/brick3/gvol62 \
  qct138:/gluster/brick3/gvol62
gluster volume start gvol62
```

Next, the hot tier was configured, comprised of two NVMe SSD drives per storage node:

```
gluster volume attach-tier gvol62 replica 2 \
 qct135:/gluster/ssd0/tier4x2 \
 qct136:/gluster/ssd0/tier4x2 \
 qct137:/gluster/ssd0/tier4x2 \
 qct138:/gluster/ssd0/tier4x2 \
 qct135:/gluster/ssd1/tier4x2 \
 qct136:/gluster/ssd1/tier4x2 \
 qct137:/gluster/ssd1/tier4x2 \
 qct138:/gluster/ssd1/tier4x2
```

Finally, the volume settings are configured:

```
gluster volume set gvol62 cluster.lookup-optimize on
gluster volume set gvol62 client.event-threads 4
gluster volume set gvol62 server.event-threads 4
gluster volume set gvol62 performance.io-cache off
gluster volume set gvol62 performance.quick-read off
```

The Gluster volume information output was:

```
Volume Name: gvol62
Type: Tier
Volume ID: 81d573e6-322b-49bd-9b22-7c56f7827f2e
Status: Started
Number of Bricks: 20
Transport-type: tcp
Hot Tier :
Hot Tier Type : Distributed-Replicate
Number of Bricks: 4 x 2 = 8
Brick1: qct138:/gluster/ssd1/tier4x2
Brick2: qct137:/gluster/ssd1/tier4x2
Brick3: qct136:/gluster/ssd1/tier4x2
Brick4: qct135:/gluster/ssd1/tier4x2
Brick5: qct138:/gluster/ssd0/tier4x2
Brick6: qct137:/gluster/ssd0/tier4x2
Brick7: qct136:/gluster/ssd0/tier4x2
Brick8: qct135:/gluster/ssd0/tier4x2
Cold Tier:
Cold Tier Type : Distributed-Replicate
Number of Bricks: 6 x 2 = 12
Brick9: qct135:/gluster/brick1/gvol62
Brick10: qct136:/gluster/brick1/gvol62
Brick11: qct137:/gluster/brick1/gvol62
Brick12: qct138:/gluster/brick1/gvol62
Brick13: qct135:/gluster/brick2/gvol62
```

```
Brick14: qct136:/gluster/brick2/gvol62
Brick15: qct137:/gluster/brick2/gvol62
Brick16: qct138:/gluster/brick2/gvol62
Brick17: qct135:/gluster/brick3/gvol62
Brick18: qct136:/gluster/brick3/gvol62
Brick19: qct137:/gluster/brick3/gvol62
Brick20: qct138:/gluster/brick3/gvol62
Options Reconfigured:
performance.quick-read: off
performance.io-cache: off
cluster.lookup-optimize: on
server.event-threads: 4
client.event-threads: 4
cluster.tier-mode: cache
features.ctr-enabled: on
performance.readdir-ahead: on
```

ABOUT QCT (QUANTA CLOUD TECHNOLOGY)

QCT (Quanta Cloud Technology) is a global datacenter solution provider extending the power of hyperscale datacenter design in standard and open SKUs to all datacenter customers. Product lines include servers, storage, network switches, integrated rack systems and cloud solutions, all delivering hyperscale efficiency, scalability, reliability, manageability, serviceability and optimized performance for each workload. QCT offers a full spectrum of datacenter products and services from engineering, integration and optimization to global supply chain support, all under one roof. The parent of QCT is Quanta Computer Inc., a Fortune Global 500 technology engineering and manufacturing company. www.qct.io

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.

facebook.com/redhatinc
@redhatnews
linkedin.com/company/red-hat

redhat.com
INC0436676_0016

NORTH AMERICA
1 888 REDHAT1

EUROPE, MIDDLE EAST, AND AFRICA
00800 7334 2835
europe@redhat.com

ASIA PACIFIC
+65 6490 4200
apac@redhat.com

LATIN AMERICA
+54 11 4329 7300
info-latam@redhat.com

Copyright © 2016 Red Hat, Inc. Red Hat, Red Hat Enterprise Linux, the Shadowman logo, and JBoss are trademarks of Red Hat, Inc., registered in the U.S. and other countries. The OpenStack® Word Mark and OpenStack Logo are either registered trademarks / service marks or trademarks / service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation or the OpenStack community. Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.