Red Hat Gluster Storage 3.4

3.4 Release Notes

Release Notes for Red Hat Gluster Storage 3.4

Edition 1

Last Updated: 2020-04-16

Red Hat Gluster Storage 3.4 3.4 Release Notes

Release Notes for Red Hat Gluster Storage 3.4
Edition 1

Gluster Storage Documentation Team
Red Hat Customer Content Services
gluster-docs@redhat.com

Legal Notice

Abstract

These release notes provide high-level coverage of the improvements and additions that have been implemented in Red Hat Gluster Storage 3.4.

Table of Contents

CHAPTER 1. INTRODUCTION

Red Hat Gluster Storage is a software only, scale-out storage solution that provides flexible and agile unstructured data storage for the enterprise. Red Hat Gluster Storage provides new opportunities to unify data storage and infrastructure, increase performance, and improve availability and manageability to meet a broader set of the storage challenges and needs of an organization.

GlusterFS, a key building block of Red Hat Gluster Storage, is based on a stackable user space design and can deliver exceptional performance for diverse workloads. GlusterFS aggregates various storage servers over different network interfaces and connects them to form a single large parallel network file system. The POSIX compatible GlusterFS servers use XFS file system format to store data on disks. These servers be accessed using industry standard access protocols including Network File System (NFS) and Server Message Block SMB (also known as CIFS).

Red Hat Gluster Storage Servers for On-premises can be used in the deployment of private clouds or data centers. Red Hat Gluster Storage can be installed on commodity servers and storage hardware resulting in a powerful, massively scalable, and highly available NAS environment. Additionally, Red Hat Gluster Storage can be deployed in the public cloud using Red Hat Gluster Storage Server for Public Cloud with Amazon Web Services (AWS), Microsoft Azure, or Google Cloud. It delivers all the features and functionality possible in a private cloud or data center to the public cloud by providing massively scalable and high available NAS in the cloud.

Red Hat Gluster Storage Server for On-premises

Red Hat Gluster Storage Server for On-premises enables enterprises to treat physical storage as a virtualized, scalable, and centrally managed pool of storage by using commodity servers and storage hardware.

Red Hat Gluster Storage Server for Public Cloud

Red Hat Gluster Storage Server for Public Cloud packages GlusterFS for deploying scalable NAS in AWS, Microsoft Azure, and Google Cloud. This powerful storage server provides a highly available, scalable, virtualized, and centrally managed pool of storage for users of these public cloud providers.

CHAPTER 2. WHAT CHANGED IN THIS RELEASE?

2.1. WHAT'S NEW IN THIS RELEASE?

This section describes the key features and enhancements in the Red Hat Gluster Storage 3.4 release.

Red Hat Gluster Storage volumes exported using SMB can now be mounted on macOS clients

Red Hat Gluster Storage volumes exported using SMB can now be mounted on macOS clients.

For more information about configuring user access and mounting volumes on macOS, see SMB in the Red Hat Gluster Storage 3.4 *Administration Guide*.

Support for upgrading across underlying Red Hat Enterprise Linux versions

You can now upgrade the underlying Red Hat Enterprise Linux 6 to Red Hat Enterprise Linux 7 by performing an offline upgrade on a gluster system using the preupgrade-assistant.

For more information, see Upgrading Red Hat Gluster Storage to Red Hat Enterprise Linux 7 chapter in the Red Hat Gluster Storage 3.4 *Installation Guide*.

Identify files that skipped rebalance operation

We can identify the files that skipped rebalance operation. Until this release, rebalance status would only indicate the 'count' of failed and skipped entries. Now, users can search for the msgid **109126** to fetch the list of skipped files.

For more information, see Displaying Rebalance Progress section in the Red Hat Gluster Storage 3.4 *Administration Guide*.

Reserve disk space on the bricks

Administrators can now reserve disk space on the bricks. The **storage.reserve** option helps reserve enough space for gluster processes, preventing the disks from reaching full capacity.

For more information, see Reserving Storage on a Volume section in the Red Hat Gluster Storage 3.4 *Administration Guide*.

Capability to resolve GFID split brain from CLI

GFID split brain can be analysed and resolved automatically through a new CLI command.

For more information, see Recovering GFID Split-brain from the gluster CLI section in the Red Hat Gluster Storage 3.4 *Administration Guide*.

Stopping the remove-brick operation

Stopping a remove-brick operation is now fully supported. If you have started a remove-brick operation, but have not yet committed the operation, you can now stop the operation. Files migrated during the remove-brick operation are not migrated back to the original brick when the remove-brick operation is stopped.

For more information, see Stopping a remove-brick operation in the Red Hat Gluster Storage 3.4 *Administration Guide*.

Mounting volumes read-only

Mounting volumes with read-only permissions is now fully supported. Volumes can be mounted with read-only permissions at either the volume or the mount point level.

To mount a volume as read-only, use the **ro** option when you mount the volume.

```
# mount -t glusterfs -o ro hostname:volname mountpoint
```

To specify that a volume can only be mounted with read-only permissions, enable the **read-only** volume option by running the following command on any Red Hat Gluster Storage server in the storage pool that hosts that volume.

```
# gluster volume set volname read-only enable
```

Mounting sub-directories using Native Client (FUSE)

Mounting sub-directories of a gluster volume using the Native Client is now fully supported. Giving multiple users access to an entire mounted volume can be a security risk, as users can obtain information belonging to other users. Mounting subdirectories ensures that users can access only their part of the storage. It also provides namespace isolation for users, so that multiple users can access the storage without risking namespace collision with other users.

See Manually Mounting Sub-directories Using Native Client in the Red Hat Gluster Storage 3.4 *Administration Guide* for more information.

Firewall Configuration automated by tendrl-ansible for Red Hat Gluster Storage Web Administration

Previously, firewall configuration was done manually leading to firewall misconfiguration. With this release, firewall configuration is automated in tendrl-ansible and applied during automated installation, allowing proper firewall rules to be configured for Web Administration without affecting other existing rules.

For more information, see 2.4. Firewall Configuration chapter in the Red Hat Gluster Storage 3.4 *Administration Guide*.

Setting customized and user-friendly cluster name for easy identification in Red Hat Gluster Storage Web Administration

Previously, clusters were imported without being able to set a user-friendly cluster name. The clusters were identified using the UUID which caused difficulty in locating and identifying a particular cluster when multiple clusters existed. With this new feature, users can provide a customized and user-friendly cluster name during importing cluster for easy identification of clusters managed by the Web Administration environment.

Unmanage clusters through Web Administration UI

Previously, there was no UI-based feature to unmanage a specific cluster. With this release, users are able to unmanage a specific cluster using the UI-based feature available in the Web Administration interface.

For more information, see chapter 3.1. Unmanaging Cluster in the Red Hat Gluster Storage Web Administration 3.4 *Monitoring Guide*.

2.2. SUPPORT LIMITATIONS

Virtual Disk Optimizer (VDO)

Virtual Disk Optimizer (VDO) volumes are only supported as part of a Red Hat Hyperconverged Infrastructure for Virtualization 2.0 deployment. VDO is not supported with other Red Hat Gluster Storage deployments.

2.3. DEPRECATED FEATURES

The following features are deprecated as of Red Hat Gluster Storage 3.4, or will be considered deprecated in subsequent releases. Review individual items for details about the likely removal time frame of the feature.

Red Hat Gluster Storage Console

As of Red Hat Gluster Storage 3.4, the existing Red Hat Gluster Storage Console management infrastructure is supported through the current Red Hat Gluster Storage 3.x life cycle, which ends on October 31, 2019. Red Hat Gluster Storage Web Administration is now the recommended monitoring tool for Red Hat Storage Gluster clusters.

For information on Red Hat Gluster Storage life cycle, see Red Hat Gluster Storage Life Cycle .

For Red Hat Gluster Storage Web Administration installation instructions, see the Red Hat Gluster Storage Web Administration Quick Start Guide and for instructions to monitor your Gluster servers, see the Red Hat Gluster Storage Web Administration Monitoring Guide

Nagios Monitoring

As of Red Hat Gluster Storage 3.4, Nagios is considered deprecated. Nagios plugins and Nagios server are no longer maintained and would not be provided in releases post Red Hat Gluster Storage 3.4. Nagios remains supported for this release, but Red Hat no longer recommends its use, and plans to remove support in future versions of Red Hat Gluster Storage.

Nagios is being deprecated because of the limited capabilities of monitoring and aggregation of results for a gluster cluster. These limitations are addressed in Red Hat Gluster Storage Web Administration.

Red Hat Gluster Storage users need to set up Red Hat Gluster Storage Web Administration in order to monitor a cluster. There is no migration path for the data collected in Nagios.

For Red Hat Gluster Storage Web Administration installation instructions, see the Red Hat Gluster Storage Web Administration Quick Start Guide and for instructions to monitor your Gluster servers, see the Red Hat Gluster Storage Web Administration Monitoring Guide.

gstatus command

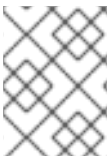
gstatus is deprecated. The gstatus command provided an overview of the health of Red Hat Gluster Storage clusters and volumes. To view health of Red Hat Gluster Storage clusters and volumes, access the Grafana Dashboard integrated to the Web Administration environment.

Parallel NFS (pNFS)

As of Red Hat Gluster Storage 3.4, Parallel NFS is considered unsupported and is no longer available as a Technology Preview. Several long-term issues with this feature that affect stability remain unresolved upstream. Information about using this feature has been removed from Red Hat Gluster Storage 3.4 but remains available in the documentation for releases that provided Parallel NFS as a Technology Preview.

CHAPTER 3. NOTABLE BUG FIXES

This chapter describes bugs fixed in this release of Red Hat Gluster Storage that have significant impact on users.



NOTE

Bugzilla IDs that are not hyperlinked are private bugs that have potentially sensitive data attached.

arbiter

BZ#1401969

Previously, when bricks went down in a particular order while parallel I/O was in progress, the arbiter brick became the source for data heal. This led to data being unavailable, since arbiter bricks store only metadata. With this fix, arbiter brick will not be marked as source.

BZ#1446125

Red Hat Gluster Storage volumes exported using SMB can now be mounted on macOS clients using Finder.

bitrot

BZ#1519740

Previously, the BitRot version internal extended attribute was passed up to AFR making it launch spurious metadata self-heals on the files. With this fix, the attribute is filtered out by bit-rot preventing these heals and the corresponding messages in the log files.

BZ#1517463

When scrub-throttle or scrub-frequency is set, the output for all successful BitRot operation (enable, disable or scrub options) would appear as volume bitrot: scrub-frequency is set to **FREQUENCY** successfully for volume **VOLUME_NAME.**

common-ha

BZ#1226874

Previously, the NFS-Ganesha startup script did not clean up the ganesha processes if the setup failed. Hence, ganesha processes continued to run on the nodes even if the 'gluster nfs-ganesha enable' command failed. With this fix, the ganesha processes are stopped on the nodes when the 'gluster nfs-ganesha enable' command fails, and the processes are cleaned up.

core

BZ#1324531

Previously, users were able to create trash directory on volumes, even when the trash directory feature was disabled. The users did not have permissions to remove the trash directory from the mount point. With this fix, a trash directory can only be created if the trash directory feature is enabled. This also allows the users to remove the directory even when the feature is disabled.

BZ#1446046

Earlier to configure ssl-cert-depth option parameter needs to set in **/etc/glusterfs/glusterd.vol** but after apply the patch parameter ***transport.socket.ssl-cert-depth*** needs to be set in **/var/lib/glusterd/secure-access**. This parameter is useful only while management Secure Sockets Layer is enabled.

BZ#1484446

Some gluster daemons (like glustershd) consumes higher CPU and memory while healing a large amount of data/entries. This can be resolved by running the control-cpu-load.sh script. This script controls the groups for regulation of CPU and memory consumption by any gluster daemon(s).

BZ#1559886

Earlier, while dumping clients, inode dictionaries were redundantly accessed under client_table lock. As a result of this behavior, gluster volume status inode suspended the brick. With this fix, the inode dictionaries under client_table lock are not redundantly populated.

disperse

BZ#1561733

Previously, when **lookup-optimize** was enabled, some files were not migrated on a disperse volume due to incomplete data returned by a special rebalance request. This fix ensures that all required data is transferred and all files are migrated.

BZ#1499865

This feature implements support for discard operation on Erasure Coded volumes. This operation can be used to deallocate blocks inside a file. Within the given range, partial fragments are zeroed, and whole fragments are deallocated.

BZ#1459101

Previously, the performance of write FOPs was affected due to FOPs modifying a non overlapping range of offset of the same file. This behavior prevented optimum performance, especially, if a brick was slow that caused each FOP taking more time to return. With the implementation of the parallel-writes feature, the FOPs performance is significantly improved.

BZ#1509830

Disperse translator is optimized to perform xattr update operation in parallel on the bricks during self-heal to improve performance.

BZ#1530519

Red Hat Gluster Storage 3.4 introduces a new option 'other-eager-lock' to keep eager-lock enabled for regular files but disabled for directory access. Earlier, the eager-lock option was used to accelerate performance for file access, however, directory access suffered when this option was enabled.

BZ#1555261

An issue prevented seal-heal to progress causing delays in the heal process after replacing a brick or bringing a failed brick online. This fix keeps self-heal active until all pending files are completely healed.

BZ#1611151

A delay in lock contention caused entry fops like Ls and renames to be slow when two client accessed the same directory on an EC volume. As a consequence, listing of directories and other entry operations were slow. This issue is solved in this version of RHGS.

distribute

BZ#1550315

Earlier for a distributed volume, the custom extended attribute value for a directory was shown incorrect value after executing stop, start command or while adding a new brick. DHT failed to sync the extended attribute value. This release introduces a new volume that will be referred at the time of update custom xattrs for the directory.

BZ#1463114

Skipped files are now logged in the rebalance log with MSGID: **109126**. The users can search for the list of skipped files using the message id.

BZ#1550771

Renaming a directory on a gluster volume involves renaming the directory on every brick of the volume. If the rename failed on one or more bricks, directories with both old and new names on one or more bricks were present on the bricks. This caused some of the contents of those directories to become inaccessible from the mount point. With this fix, the directory rename operation is rolled back if it fails on any brick.

BZ#1118770

Previously, a lookup on a directory that was being renamed could lead to directories with both old and new names existing on the volume. This could cause some of the contents of these directories to be inaccessible from the mount point. Now, additional synchronization has been introduced to prevent this from occurring.

BZ#1392905

During a rebalance process, hardlinks were reported as failures instead of being marked as skipped. With this fix, no failures for hardlink migration are noticed, rather they are added to a skipped list during rebalance operation

BZ#1557365

With this update, the lookup-optimize option is enabled by default. This option enhances the lookup and create performances.

BZ#1577051

Previously, the remove-brick process did not show any failure during a lookup failure. It is recommended to check the decommissioned brick before doing a **remove-brick commit** for any left out files. With this fix, the remove brick status shows failure count.

eventsapi

BZ#1466122

Previously, gluster-events daemon failed to send events to a registered webhook if the webhook was https enabled. With this fix, gluster-events daemon registers the webhook if it is https enabled.

BZ#1466129

Earlier, Gluster did not add HMAC signature (hash-based message authentication code) to the events pushed to the webhook. With this fix, gluster-event daemon generates an HMAC token and adds it to the authorization header while sending it to the webhook.

fuse

BZ#1508999

subdir mounted clients cannot heal the directory structure when an **add-brick** is performed because distribute would not know the parent directories of subdirectories while performing directory self-heal. You can fix this by mounting the volume (without the subdirectory) on one of the server after add-brick, and run self-heal operations on the directories. You can perform these tasks by using 'hook' scripts, so that no user intervention is required.

BZ#1501013

Earlier, Gluster did not add HMAC signature (hash-based message authentication code) to the events pushed to the webhook. With this fix, gluster-event daemon generates an HMAC token and adds it to the authorization header while sending it to the webhook.

geo-replication

BZ#1568655

Previously, if symbolic links (symlinks) were created by a non-privileged user to the current directory on the master volume, geo-replication failed to synchronize them to the slave volume. Instead of setting the permissions on the symlink, it used to dereference the symlink to the virtual directory **.gfid** and the geo-replication session failed with **Operation not supported** error. With this fix, geo-replication does not dereference the symlink while setting the permissions on the file and synchronizes the symlinks created by a non-privileged user to the current directory.

BZ#1288115

Red Hat Gluster Storage 3.4 provides an option to grant read-only access on a gluster volume using the following command:

```
#gluster volume set VOLNAME features.read-only on
```

This option is useful in geo-replication, when the data should be written on the slave volume by any other client except geo-replication.

BZ#1468972

Red Gluster Gluster Storage 3.4 introduces structured logging in geo-replication to improve the log messages. With structured logging you can easily parse the log messages to get the required information.

BZ#1599037

Previously, when geo-replication was started or recovered from a **Faulty** state, it regenerated xsync changelogs that needed to be processed. Whereas the unsynced xsync changelogs, which were generated before the stop or restart operation, were not processed. This resulted in consumption of inodes and space. With this release, the unsynced changelogs are unprocessed every time geo-replication is restarted or recovered from **Faulty** state and also does not consume space and inodes.

BZ#1342785

Previously, metadata changes such as ownership change on a symbolic link (symlink) file resulted in a crash with **Permission Denied** error. With this release, geo-replication is fixed to synchronize metadata of the symlink files, and the ownership change of symlink files is replicated properly without crashing.

BZ#1470967

Geo-replication expects the GFID to be the same on master volume and slave volume. However, geo-replication failed to synchronize the entry to slave volume when a file already existed with a different GFID. With this release, the GFID mismatch failures are handled by verifying it on the master volume.

BZ#1498391

Geo-replication uses changelog feature and enables it as part of geo-replication session creation. But the command to disable changelog did not warn the user about any existing geo-replication session. With this release, a geo-replication session check is added while disabling the changelog. Hence, while disabling the changelog, a warning is generated for the user about any existing geo-replication sessions.

BZ#1503173

Geo-replication mounts the master and slave volume internally to synchronize the data, thus the mount points were not available to debug. As a result, the user failed to retrieve the client volume profile information. This fix brings two new options to users to access the geo-replication mount points: **slave_access_mount**: for slave mount points and **master_access_mount**: for master mount points.

BZ#1557297

Earlier, geo-replication could be paused or resumed by an unauthorized user. As a result, snapshot creation operation failed. With this release, pausing or resuming a geo-replication session by an unauthorized user, other than the session creator is restricted, and will display the following error message: **Geo-replication session between *USERNAME* and *SLAVE_HOSTNAME* does not exist**

BZ#1565399

Previously, if symbolic links (symlinks) were created by a non-privileged user pointing to current directory on the master volume, geo-replication failed to synchronize them to slave. Instead of setting the permissions on the symlink, it used to dereference the symlink pointing to the virtual directory .gfid and failed with Operation not supported error. With this fix, geo-replication does not dereference symlink while setting permissions on file and syncs symlinks created by non-privileged user pointing to the current directory.

BZ#1601314

Previously, when a symbolic link (symlink) was created and renamed, followed by directory creation with the same name as the original symlink, the file caused out-of-order synchronization. This caused the directory to be synchronized first without synchronizing the renamed symlink (or symlink renaming). As a consequence, geo-replication sometimes failed to synchronize the renamed symlink.

With this release, while processing out-of-order file rename operation, if the file name already exists, the users can use GFID to check the identity of the file and synchronize accordingly.

samba

BZ#1379444

Previously, sharing a subdirectory of a gluster volume failed with an I/O error when the **shadow_copy2 vfs** object was specified in the **smb.conf** file. This occurred because gluster volumes are remote file systems, and **shadow_copy2** only detected shared paths in the local file system. This update forces the value of **shadow:mountpath** to /, skipping the code related to mount point detection and preventing this problem. This fix requires that the **glusterfs vfs** object is listed after the **shadow_copy2 vfs** object in the **smb.conf** file.

NFS-ganesha

BZ#1480138

NFS-Ganesha internally uses a different file descriptor for every lock request. When a client removes a file with a specific file descriptor, other clients who are trying to access the same file with the same file descriptor receive a **No such file or directory** error. With this release, lock request uses the same file descriptor obtained from an open call and no additional open call in the lock handling path is required.

BZ#1514615

To limit the NFS version to 3 and 4.0, NFSv4 blocks were added in the **ganesha.conf** file manually. With this release, these options are added as a default in the **ganesha.conf** file.

BZ#1481040

Currently, the default interval between upcall polls is 10 microseconds. For large numbers of threads, this results in heavy CPU utilization. With this release, the default polling interval has been increased to 100 milliseconds (100000 microseconds), thus, helping in the reduction of CPU utilization.

BZ#1545523

Previously, if a client requests a **SET_ATTR_MODE** in a create call, the NFS server needed to perform a **setattr** operation post creation. Incase of gluster NFS server, GFID used for the **setattr** operation was **NULL** and thus resulted in **EINVAL** error. With this release, you can use the GFID from the linked node and the create call with **SET_ATTR_MODE** is executed successfully.

BZ#1489378

With this release, the **USE_GLUSTER_XREADDIRPLUS** option is introduced that enables enhanced readdirp instead of standard readdir. This option is turned on by default. When this option is turned off, NFS falls back to standard readdir. Turning off this option would result in more lookup and stat requests being sent from the client, which may impact performance.

BZ#1516699

With Red Hat Gluster Storage 3.4, NFS-Ganesha log files have been moved to the **/var/log/ganesha** subdirectory.

glusterd

BZ#1449867

Earlier in case of a node reboot, if the network interface was not completely functional before the glusterd service was initiated, glusterd failed to resolve the brick addresses for different peers, thus, resulting in glusterd service to fail to initiate. With this release, glusterd initialization would not fail even when the network interface is not completely functional.

BZ#1599823

Earlier, during a volume create operation, glusterd failed to handle blank real paths while checking if the brick path is already a part of another volume. Hence, volume create requests would fail. This release fixes the path comparison logic. Now, glusterd can handle blank paths and prevents failure of volume create requests.

BZ#1575539

The memory leak at glusterd resulted during a gluster volume geo-replication status operation has been addressed in Red Hat Gluster Storage 3.4.

BZ#1369420

Previously, when glusterd service was restarted, the AVC denial message was displayed for port 61000. With this release, if you configure the max-port in **glusterd.vol** below 61000, then the AVC denial message is no longer seen.

BZ#1529451

Previously, executing **gluster volume status** command multiple times caused excessive memory consumption by the glusterd process. With this release, this issue has been fixed.

BZ#1474745

Red Hat Gluster Storage 3.4 allows you to define the max-port value in **glusterd.vol** to control the range of the port that gluster bricks can consume.

locks

BZ#1507361

Earlier, the **gluster volume clear-locks** command failed to release the acquired memory completely. This caused increasingly high memory utilization on the brick processes over time. With this fix, the associated memories are released when the clear-locks command is executed.

BZ#1510725

When a stale lock on the volume is cleared using the **gluster volume clear-locks *volname*** command, one of the references to lock is persistent in the memory even after the lock is destroyed. Upon disconnection from the client, this invalid memory is accessed which leads to a crash. With this release, the final reference to the lock is cleared as part of the clear-locks command, so the operation does not lead to unofficial memory access.

BZ#1495161

Previously, processes that used multiple POSIX locks, possibly in combination with gluster clear-locks command, would lead to memory leak causing high memory consumption on brick processes. In some cases, this triggered a **OOM killer** error. This release fixes the issue related to leaks present in translators.

VDSM

BZ#1503070

The VDSM package has been upgraded to upstream version 4.19, which provides a number of bug fixes and enhancements over the previous version. This update allows Red Hat Gluster Storage 3.4 nodes with a cluster compatibility version of 4.1 to be managed by Red Hat Virtualization.

BZ#1542859

When a response from VDSM did not include the 'device' field, ovirt-engine did not update the database, which resulted in incorrect status information being displayed in the Administration Portal. This fix ensures that the device field is always part of VDSM responses so that brick status is accurately reflected in the Administration Portal.

posix

BZ#1464350

The POSIX translator is now enhanced with an option that allows user to reserve disk space on the bricks. Some administrative operations, like expanding storage or rebalancing data across nodes, require spare working space on the disk. The storage.reserve option lets users expand disk or cluster when backend bricks are full preventing ENOSPC errors on mount points.

BZ#1620765

Previously, file rename operation failed to rename the **linkto** file because of an incorrect xattr value set on the **linkto** file. This was seen on volumes upon which **glusterfind** was used. As a result, files were inaccessible if the lookup-optimize option was enabled on the volume. With this fix, the value of the xattr set on the **linkto** file allows the rename to proceed.

protocol

BZ#1319271

Previously, **auth.allow** or **auth.reject** options did not accept hostnames as values when provided with FQDN. With this fix, these options now accept hostnames when provided with FQDN.

quota

BZ#1414456

Previously, the path ancestry was not accurately populated when a symbolic link file had multiple hard links to it. This resulted in pending entry heal. With this fix, the ancestry is populated by handling the scenario of the symbolic link file with multiple hard links.

BZ#1475779

Previously, a directory failed to get healed post an add-brick operation if the directory's quota had already exceeded hard limit prior to add-brick. As a result, the directory structure remained incomplete on the newly added brick. With this fix, the directory heal happens irrespective of the resulting quota limit.

BZ#1557551, BZ#1581231

Previously, while quota was enabled on a volume, the quota used values were not updated to the list command until a lookup was done from the client mount point. Due to this, there was inaccuracy while reporting the file size even after performing the crawl operation. With this fix, it is ensured that the crawl operation looks up all files and reports the accurate quota used.

BZ#1511766

Previously, it was not possible to have more than 7712 limits configured for a volume due to limitations on how the quota.conf file was read and written. With this fix, you can now configure more than 65000 limits on a single volume.

readdir-ahead

BZ#1463592

The parallel-readdir volume options were not a part of any of the translators. Because of this, the following warning message was shown in the client log when parallel-readdir was enabled: "option 'parallel-readdir' is not recognized". With this fix, parallel-readdir is now added as an option of readdir-ahead translator and the warning message is not seen in client logs.

BZ#1559884

When the gluster volume had the combinations mentioned below, some of the files appeared twice in mountpoint after performing a readdir operation. The following combination of volume options caused the error: performance.stat-prefetch off performance.readdir-ahead on performance.parallel-readdir on With this fix, when readdir is issued on the mountpoint, no double entries are seen for a single file.

replicate

BZ#1286820

Red Hat Gluster Storage 3.4 introduces the **summary** command. This command displays the statistics of entries pending heal in split-brain and the entries undergoing healing.

BZ#1413959

With this release, GFID split-brains can be resolved from the CLI using any of the policies: choice of brick, mtime or size. You need to provide the absolute path of the file which needs GFID heal.

BZ#1452915

Previously, when the heal daemon was disabled by using the **heal disable** command, you had to manually trigger a heal by using **gluster volume heal volname** command. The **heal** command displayed an incorrect and misleading error message. With this fix, when you try to trigger a manual heal on a disabled daemon, a useful and correct error message is displayed directing the user to enable the daemon in order to trigger a heal.

BZ#1489876

Since replica 2 volumes are prone to split-brain, they will be deprecated in the future releases of Red Hat Gluster Storage 3.4. Therefore, while creating a replica 2 volume, an appropriate warning message is displayed which recommends to use the Arbiter or replica 3 configurations.

BZ#1501023

Previously, volume-set command used to re-configure the choose-local option was not working as expected due to AFR not handling reconfiguration of the choose-local option. With this fix, appropriate changes are made to make AFR handle reconfiguring choose-local option.

BZ#1593865

glusterd can send heal related requests to self-heal daemon before the latter's graph is fully initialized. In this case, the self-heal daemon used to crash when trying to access certain data structures. With the fix, if the self-heal daemon receives a request before its graph is initialized, it will ignore the request.

BZ#1361209

Previously, when the heal daemon was disabled by using the **heal disable** command, you had to

manually trigger a heal by using **gluster volume heal volname** command. The **heal** command displayed an incorrect and misleading error message. With this fix, when you try to trigger a manual heal on a disabled daemon, a useful and correct error message is displayed directing the user to enable the daemon in order to trigger a heal.

BZ#1470566

The users can now convert a plain distributed volume to a distributed-replicate volume by executing the 'gluster volume add-brick` command, provided there is no I/O happening on the client.

BZ#1552414

In replica 3 volumes, there was a possibility of ending up in split brain, when multiple clients simultaneously write data on the same file at non overlapping regions. With the new **cluster.full-lock**option, you can take full file lock which helps you in maintaining data consistency and avoid ending up in split-brain. By default, the **cluster.full-lock**option option is set to take full file lock and can be reconfigured to take range locks, if needed.

BZ#1566336

Previously, a successful file creation on a brick in a replica 3 volume would set the pending changelog as part of a new entry marking for that file. As the entry transaction failed on quorum number of bricks, the parent file will not have any entry pending changelog set. As a consequence, the entry transaction would be listed in the heal info output, but would never get healed by the SHD crawl or index heal. With this fix, if an entry transaction fails on quorum number of bricks, a dirty marking is set on the parent file of the brick where the transaction was successful. This action allows the entry to be healed as part of the next SHD crawl or index heal.

BZ#1397798, BZ#1479335

Some gluster daemons like glustershd have a higher CPU or memory consumption when there is a large amount of data or entries to be healed. This resulted in slow consumption of resources. With this fix, the users can resolve the slow consumption of resources by running the control-cpu-load.sh script. This script uses the control groups for regulating CPU and memory consumption of any gluster daemon.

rpc

BZ#1408354

This feature controls the TCP keep-alive feature timings per socket. Applications deployed by users had prerequisites concerning the system call latencies. Using gluster as the file-system adds further network latency to the existing system-call latency. These gluster options opened to the user will help them fine-tune the application to an acceptable network latency beyond which the TCP socket will be assumed as disconnected. Default values assigned to these options configure every TCP socket to the system default values when no such options are explicitly set. The gluster volume options opened to the users are:

- **server.tcp-user-timeout**
- **client.tcp-user-timeout**
- **transport.socket.keepalive-interval**
- **transport.socket.keepalive-count**
- **server.tcp-user-timeout**

These options are 1:1 compatible with similar options described in the TCP(7) manual page. This feature provides a mechanism to fine tune the application behaviour for better user experience.

snapshot

1464150

GlusterFS used to mount deactivated snapshot(s) under /run/gluster/snaps by default. Furthermore, the snapshot status command should show relevant information for the deactivated snapshot(s). Since we have a mount, there is a possibility that some process may access the mount causing issues while unmounting the volume during the snapshot deletion. This feature assures that GlusterFS does not mount deactivated snapshot(s) and displays the text **N/A (Deactivated Snapshot)** in Volume Group filed for snapshot status command.

vulnerability

BZ#1601298

It was found that glusterfs server does not properly sanitize file paths in the ***trusted.io-stats-dump*** extended attribute which is used by the ***debug/io-stats***translator. An attacker can use this flaw to create files and execute arbitrary code. To exploit this, the attacker would require sufficient access to modify the extended attributes of files on a gluster volume.

BZ#1601642

It was found that glusterfs server is vulnerable to mulitple stack based buffer overflows due to functions in server-rpc-fopc.c allocating fixed size buffers using **alloca(3)**. An authenticated attacker could exploit this by mounting a gluster volume and sending a string longer that the fixed buffer size to cause crash or potential code execution.

BZ#1610659

It was found that the **mknod** call derived from mknod(2) can create files pointing to devices on a glusterfs server node. An authenticated attacker could use this to create an arbitrary device and read data from any device attached to the glusterfs server node.

BZ#1612658

A flaw was found in RPC request using **gfs3_lookup_req** in glusterfs server. An authenticated attacker could use this flaw to leak information and execute remote denial of service by crashing gluster brick process.

BZ#1612659

A flaw was found in RPC request using **gfs3_symlink_req** in glusterfs server which allows symbolic link (symlink) destinations to point to file paths outside of the gluster volume. An authenticated attacker could use this flaw to create arbitrary symlinks pointing anywhere on the server and execute arbitrary code on glusterfs server nodes.

BZ#1612660

A flaw was found in RPC request using **gfs2_create_req** in glusterfs server. An authenticated attacker could use this flaw to create arbitrary files and execute arbitrary code on glusterfs server nodes.

BZ#1612664

A flaw was found in RPC request using **gfs3_rename_req** in glusterfs server. An authenticated attacker could use this flaw to write to a destination outside the gluster volume.

BZ#1613143

A flaw was found in RPC request using **gfs3_mknod_req** supported by glusterfs server handles. An authenticated attacker could use this flaw to write files to an arbitrary location via path traversal and execute arbitrary code on a glusterfs server node.

BZ#1601657

A flaw was found in the way **dict.c:dict_unserialize** function of glusterfs, dic_unserialize function does not handle negative key length values. An attacker could use this flaw to read memory from other locations into the stored dict value.

BZ#1607617

It was found that an attacker could issue a xattr request via glusterfs FUSE to cause gluster brick process to crash which will result in a remote denial of service. If gluster multiplexing is enabled this will result in a crash of multiple bricks and gluster volumes.

BZ#1607618

An information disclosure vulnerability was discovered in glusterfs server. An attacker could issue a xattr request via glusterfs FUSE to determine the existence of any file.

write-behind

BZ#1426042

The **performance.write-behind-trickling-writes** and **performance.nfs.write-behind-trickling-writes** options enables the trickling-write strategy for the write-behind translator for FUSE and NFS clients respectively.

Web Administration

BZ#1590405

tendrl-ansible runs **yum update** by default that causes all involved systems to update with latest packages. As a consequence, unintentional patching of OS and services occurs which is not desirable for production systems.

To avoid unintentional package updates by tendrl-ansible, drop the **yum update** variable from the ansible playbooks.

BZ#1518525

Previously, during Red Hat Gluster Storage Web Administration installation, if the server had multiple active IP addresses, tendrl-ansible failed to automatically choose the correct IP address, causing installation failure. In this version, the user has to set all the required variables for tendrl-ansible as per the installation instructions.

BZ#1563648

Earlier, at infra level while persisting the object details in the central store (etcd), each field used to be written separately causing multiple REST API calls to etcd.

This method of persisting the objects to etcd triggered multiple REST API calls resulting in high volume of network calls. Also, this used to end up in race conditions when other threads tried to update an object whereas the saving thread was still writing the data to etcd.

As a performance improvement and getting away from the race condition, now the whole object gets serialized into a single JSON and is written to etcd. While reading object state from etcd, this single JSON is read and then object is weaved back using the details.

As a result, the read and write operations of each object to and from etcd now results in a single REST API call. It also avoids the critical race condition which caused lot of issues in various flows of the system.

BZ#1512937

The host-level dashboard in Grafana listed duplicate hostnames with FQDN and IP addresses for a given storage node irrespective of how the peer probe was done. This caused duplicate data for the same node being displayed in the time series data and the Grafana dashboard.

With this fix, the host-level dashboard and other dashboards display the name of the hosts only once which was used for peer probe while Gluster cluster creation.

BZ#1514442

Earlier, if import cluster failed in Web Administration interface, there was no way to initiate import again from UI. The user needed to clean the etcd details and initiate import again. As a consequence, successive attempts to import the same cluster failed.

With the latest changes around import cluster and a new feature to un-manage a cluster and import, if import cluster fails for a cluster due to invalid repositories configured in the storage nodes for installation of components, the user can now correct the issues in the underlying node and then reinitiate import of the cluster. The unmanage cluster operation also helps in this workflow, as the cluster can be un-managed and then re-imported. The import job is successful only if all the nodes report that all required components are installed and running on them with first round of synchronization complete. If any node fails to report the same, import cluster job fails.

With this fix, if import fails, the user can correct the issues on underlying nodes and execute reimport for the cluster.

BZ#1516417

Previously, Web Administration was unable to detect or expand new storage nodes added to gluster trusted storage pool. As a consequence, Web Administration could not manage and provide provide metrics to newly added nodes to a cluster after initial Import Cluster.

With this fix, Web Administration can now detect and expand new nodes to an already managed cluster once the new nodes are added to the gluster trusted storage pool.

BZ#1549146

Previously, the Grafana dashboard reported unusual and unrealistic values for different performance panels along with missing performance units. Few of the affected panels that displayed unrealistic numbers were Weeks Remaining, IOPS, Throughput, etc.

With this fix, the Grafana panels including the IOPS and Weeks Remaining panels display realistic values understandable by the user along with the appropriate performance units. Additionally, the Inode panels were removed from the brick-level and volume-level dashboards of Grafana.

BZ#1516135

Previously, there was no way to unmanage a cluster which was partially imported due to tendrl-gluster-integration install failure on few nodes of the cluster. Despite the unsuccessful import, the cluster displayed status as being successfully imported and managed by Web Administration.

As a consequence, the depiction of the cluster in the Web Administration interface was not correct as not all the peers of the cluster were successfully imported and reported in the interface.

With this fix, now an import job would be marked as finished only if all the peers report tendrl-gluster-integration running on them and their first round of synchronization of data is done in the Web Administration environment. With the unmanage cluster functionality in place, any affected cluster can be unmanaged and the underlying issues can be fixed before reimporting the cluster in the Web Administration environment.

If import fails, the issues can be corrected in the underlying cluster and re-imported in the Web Administration environment.

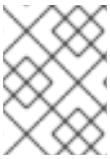
BZ#1519158

Previously, in the Clusters view of the Web Administration interface, the filter button to filter the cluster attributes was unresponsive on Mozilla Firefox web browser. Due to this issue, users were not able to filter the clusters view based on a particular cluster attribute.

This browser issue is now fixed with this release and the filter button is responsive on Firefox.

CHAPTER 4. KNOWN ISSUES

This chapter provides a list of known issues at the time of release.



NOTE

Bugzilla IDs that are not hyperlinked are private bugs that have potentially sensitive data attached.

4.1. RED HAT GLUSTER STORAGE

Issues related to glusterd

BZ#1567616

If the **enable-shared-storage** option is disabled when any one of the glusterd is down, disabling the shared storage operation will be a success. However, subsequent requests of enabling and disabling of **enable-shared-storage** operations will fail.

Workaround: Run the following commands to overcome this behavior:

```
# gluster v delete gluster_shared_storage
```

```
# gluster v set all cluster.enable-shared-storage enable
```

BZ#1400092

Performing add-brick to increase replica count while I/O is going on can lead to data loss.

Workaround: Ensure that increasing replica count is done offline, i.e. without clients accessing the volume.

BZ#1403767

On a multi node setup where NFS-Ganesha is configured, if the setup has multiple volumes and a node is rebooted at the same time as when volume is stopped, then, once the node comes up the volume status shows that volume is in started state where as it should have been stopped.

Workaround: Restarting the glusterd instance on the node where the volume status reflects **started** resolves the issue.

BZ#1417097

glusterd takes time to initialize if the setup is slow. As a result, by the time **/etc/fstab** entries are mounted, glusterd on the node is not ready to serve that mount, and the glusterd mount fails. Due to this, shared storage may not get mounted after node reboots.

Workaround: If shared storage is not mounted after the node reboots, check if glusterd is up and mount the shared storage volume manually.

BZ#1394138

If a node is deleted from the NFS-Ganesha HA cluster without performing umount, and then a peer detach of that node is performed, that volume is still accessible in **/var/run/gluster/shared_storage/** location even after removing the node in the HA-Cluster.

Workaround: After a peer is detached from the cluster, manually unmount the shared storage on that peer.

Issues related to gdeploy

BZ#1408926

Currently in a gdeploy configuration file, the **ssl_enable** option is part of the **volume** section. If more than one volume section is used in a single gdeploy configuration file for a single storage pool and **ssl_enable** is set in all the volume sections, then the SSL operation steps are performed multiple times. This fails to mount the older volumes. Thus, users will not be able to set SSL with a single line in the gdeploy configuration file.

Workaround: If there are more than one volume sections in a single gdeploy configuration file for a single storage pool, set the variable **enable_ssl** under only one volume section and set the keys: '**client.ssl**', value: 'on'; '**server.ssl**', value: 'on';'auth.ssl-allow', value: *comma separated SSL hostnames*

Issues related to Arbitrated Volumes

BZ#1387494

Currently, if the data bricks of the arbiter volume are completely consumed, further creation of new data entries may succeed in the arbiter brick without failing with an **ENOSPC** error. However, the clients will correctly receive the creation failure error on the mount point. Thus the arbiter bricks might have more entries. When an **rm -rf** command is executed from the client, **readdir** operation is performed on one of the databricks to get the list of files to deleted. Consequently, only those entries will get deleted on all bricks. When the **rmdir** command is executed on the parent directory, it succeeds on the data bricks but fails on the arbiter with an **ENOTEMPTY** error because it has some files in it.

Workaround: If the deletion from the mount does not encounter an error while the arbiter bricks still contain the directories, the directories and its associated GFID symlinks need to be manually removed. If the directory to be deleted contains files, these files and their associated GFID hard links need to be removed manually.

BZ#1388074

If some of the bricks of a replica or arbiter sub volume go offline or get disconnected from the client while a **rm -rf** command is being executed, the directories may re-appear when the bricks are back online and self-heal is complete. When the user tries to create a directory with the same name from the mount, it may heal this existing directory into other DHT subvolumes of the volume.

Workaround: If the deletion from the mount did not complete, but the bricks still contain the directories, the directories and its associated GFID symlink must be removed manually. If the directory to be deleted contains files, these files and their associated GFID hard links need to be removed manually.

BZ#1361518

If a file is being created on all bricks but succeeds only on the arbiter brick, the application using the file will fail. But during self-heal, the file gets created on the data bricks with arbiter brick marked as the data source. Since data self-heal should not be done from the arbiter brick, the output for the **gluster volume heal *volname* info** command will list the entries indefinitely.

Workaround: If the output of the **gluster volume heal *volname* info** command indefinitely displays the pending heals for a file, check if the issue is persistent by performing the following steps:

1. Use the **getfattr** command to check the following:
   - If the trusted.afr.*volname*-client* xattrs are zero on the data bricks
   - If the trusted.afr.*volname*-client* xattrs is non-zero on the arbiter brick only for the data part. The data part is the first 4 bytes.

     For example:

     ```
     #getfattr -d -m . -e hex /bricks/arbiterbrick/file |grep trusted.afr.volname*
     getfattr: Removing leading '/' from absolute path names
     trusted.afr.volname-client-0=0x000000540000000000000000
     trusted.afr.volname-client-1=0x000000540000000000000000
     ```

2. If the command output matches the mentioned state, delete the xattr using the following command:

   ```
   # for i in $(getfattr -d -m . -e hex /bricks/arbiterbrick/file |grep trusted.afr.volname*|cut -f1 -d'='); do  setfattr -x $i file; done
   ```

Issues related to Distribute (DHT) Translator

BZ#1136718

The automatic file replication (AFR) self-heal can have a partially healed file if the brick containing the AFR self-heal source file goes offline during a heal operation. If this partially healed file is migrated before the brick is back online, the migrated file would have incorrect data and the original file would be deleted.

Issues related to Replication (AFR)

BZ#1426128

In a replicate volume, if a gluster volume snapshot is created when a file creation is in progress, the file may be present in one brick of the replica but not the other brick on the snapshotted volume. Due to this, when this snapshot is restored and a **rm -rf dir** command is executed on a directory from the mount, it may fail with an **ENOTEMPTY** error.

Workaround: If you receive the ENOTEMPTY error during the **rm -rf *dir*** command execution, but the output of the **ls** command of the directory shows no entries, check the backend bricks of the replica to verify if files exist on some bricks and not the other. Execute the **stat** command with the file name from the mount, so that it is healed to all bricks of the replica. Once the file is completely healed, executing the **rm -rf *dir*** command is successful.

Issues related to gNFS

BZ#1413910

From Red Hat Gluster Storage 3.2 onwards, for every volume the option **nfs.disable** will be explicitly set to either **on** or **off**. The default value for new volumes created is **on**, due to which these volumes will not be exported via. Gluster NFS. The snapshots which were created from 3.1.x version or earlier does not have this volume option.

Workaround: Execute the following command on the volumes:

```
# gluster v set nfs.disable volname off
```

The restored volume will not be exported via. Gluster NFS.

Issues related to Tiering

BZ#1334262

If the **gluster volume tier attach** command times out, it could result in either of two situations. Either the volume does not become a tiered volume, or the tier daemon is not started.

Workaround: When the timeout is observed, perform the following:

1. Check if the volume has become a tiered volume.
   - If not, then rerun the **gluster volume tier attach** command.
   - If it has, then proceed with the next step.
2. Check if the tier daemons were created on each server.
   - If the tier daemons were not created, execute the following command:

     ```
     # gluster volume tier volname start
     ```

BZ#1303298

Listing the entries on a snapshot of a tiered volume displays incorrect permissions for some files. This is because the User Serviceable Snapshot (USS) returns the **stat** information for the link to files in the cold tier instead of the actual data file. These files appear to have **-----T** permissions.

Workaround: FUSE clients can work around this issue by applying any of the following options:

- **use-readdirp=no** This is the recommended option.
- **attribute-timeout=0**
- **entry-timeout=0**

NFS clients can work around the issue by applying the **noac** option.

BZ#1303045

When a tier is attached while I/O operation is in progress on an NFS mount, I/O pauses temporarily, usually for between 3 to 5 minutes.

Workaround: If I/O does not resume within 5 minutes, use the **gluster volume start *volname* force** command to resume I/O without interruption.

BZ#1273741

Files with hard links are not promoted or demoted on tiered volumes.

There is no known workaround for this issue.

BZ#1305490

A race condition between tier migration and hard link creation results in the hard link operation failing with a **File exists** error, and logging **Stale file handle** messages on the client. This does not impact functionality, and file access works as expected.

This race occurs when a file is migrated to the cold tier after a hard link has been created on the cold tier, but before a hard link is created to the data on the hot tier. In this situation, the attempt to create a hard link on the hot tier fails. However, because the migration converts the hard link on the cold tier to a data file, and a linkto already exists on the cold tier, the links exist and works as expected.

BZ#1277112

When hot tier storage is full, write operations such as file creation or new writes to existing files fail with a **No space left on device** error, instead of redirecting writes or flushing data to cold tier storage.

Workaround: If the hot tier is not completely full, it is possible to work around this issue by waiting for the next CTR promote/demote cycle before continuing with write operations.

If the hot tier does fill completely, administrators can copy a file from the hot tier to a safe location, delete the original file from the hot tier, and wait for demotion to free more space on the hot tier before copying the file back.

BZ#1278391

Migration from the hot tier fails when the hot tier is completely full because there is no space left to set the extended attribute that triggers migration.

BZ#1283507

Corrupted files can be identified for promotion and promoted to hot tier storage.

In rare circumstances, corruption can be missed by the BitRot scrubber. This can happen in two ways:

1. A file is corrupted before its checksum is created, so that the checksum matches the corrupted file, and the BitRot scrubber does not mark the file as corrupted.

2. A checksum is created for a healthy file, the file becomes corrupted, and the corrupted file is not compared to its checksum before being identified for promotion and promoted to the hot tier, where a new (corrupted) checksum is created.

When tiering is in use, these unidentified corrupted files can be 'heated' and selected for promotion to the hot tier. If a corrupted file is migrated to the hot tier, and the hot tier is not replicated, the corrupted file cannot be accessed or migrated back to the cold tier.

BZ#1306917

When a User Serviceable Snapshot is enabled, attaching a tier succeeds, but any I/O operations in progress during the attach tier operation may fail with stale file handle errors.

Workaround: Disable User Serviceable Snapshots before performing **attach tier**. Once **attach tier** has succeeded, User Serviceable Snapshots can be enabled.

Issues related to Snapshot

BZ#1403169

If NFS-ganesha was enabled while taking a snapshot, and during the restore of that snapshot it is disabled or shared storage is down, then the snapshot restore will fail.

BZ#1403195

Snapshot create might fail, if a brick has started but not all translators have initialized.

BZ#1169790

When a volume is down and there is an attempt to access **.snaps** directory, a negative cache entry is created in the kernel Virtual File System (VFS) cache for the **.snaps** directory. After the volume is brought back online, accessing the **.snaps** directory fails with an ENOENT error because of the negative cache entry.

Workaround: Clear the kernel VFS cache by executing the following command:

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

Note that this can cause temporary performance degradation.

BZ#1174618

If the User Serviceable Snapshot feature is enabled, and a directory has a pre-existing **.snaps** folder, then accessing that folder can lead to unexpected behavior.

Workaround: Rename the pre-existing **.snaps** folder with another name.

BZ#1394229

Performing operations which involve client graph changes such as volume set operations, restoring snapshot, etc. eventually leads to out of memory scenarios for the client processes that mount the volume.

BZ#1129675

Performing a snapshot restore while **glusterd** is not available in a cluster node or a node is unavailable results in the following errors:

- Executing the **gluster volume heal *vol-name* info** command displays the error message **Transport endpoint not connected**.
- Error occurs when clients try to connect to glusterd service.

Workaround: Perform snapshot restore only if all the nodes and their corresponding **glusterd** services are running. Start **glusterd** by running the following command:

```
# service glusterd start
```

BZ#1118780

On restoring a snapshot which was created while the rename of a directory was in progress (the directory has been renamed on the hashed sub-volume but not on all of the sub-volumes), both the old and new directories will exist and have the same GFID. This can cause inconsistencies and issues accessing files in those directories.

In DHT, a rename (source, destination) of a directory is done first on the hashed sub-volume and if successful, on the remaining sub-volumes. At this point in time, both source and destination directories are present in the volume with same GFID - destination on hashed sub-volume and

source on rest of the sub-volumes. A parallel lookup (on either source or destination) at this time can result in creation of these directories on the sub-volumes on which they do not yet exist- source directory entry on hashed and destination directory entry on the remaining sub-volumes. Hence, there would be two directory entries - source and destination - having the same GFID.

BZ#1236149

If a node/brick is down, the **snapshot create** command fails even with the force option. This is an expected behavior.

BZ#1240227

LUKS encryption over LVM is currently not supported.

BZ#1246183

User Serviceable Snapshots is not supported on Erasure Coded (EC) volumes.

Issues related to Geo-replication

BZ#1393362

If a geo-replication session is created while gluster volume rebalance is in progress, then geo-replication may miss some files/directories sync to slave volume. This is caused because of internal movement of files due to rebalance.

Workaround: Do not create a geo-replication session if the master volume rebalance is in progress.

BZ#1561393

If the quick-read performance feature is enabled on the geo-rep slave volume, it could serve stale data as it fails to invalidate its cache in a corner case. This could affect applications reading slave volume as it might get served with stale data.

Workaround: Disable quick read performance feature on the slave volume:

```
# gluster vol set slave-vol-name quick-read off
```

With quick-read performance feature disabled, slave will not serve stale data and serves consistent data.

BZ#1344861

Geo-replication configuration changes when one or more nodes are down in the Master Cluster. Due to this, the nodes that are down will have the old configuration when the nodes are up.

Workaround: Execute the Geo-replication config command again once all nodes are up. With this, all nodes in Master Cluster will have same Geo-replication config options.

BZ#1293634

Sync performance for geo-replicated storage is reduced when the master volume is tiered, resulting in slower geo-replication performance on tiered volumes.

BZ#1302320

During file promotion, the rebalance operation sets the sticky bit and suid/sgid bit. Normally, it removes these bits when the migration is complete. If readdirp is called on a file before migration completes, these bits are not removed and remain applied on the client.

If rsync happens while the bits are applied, the bits remain applied to the file as it is synced to the destination, impairing accessibility on the destination. This can happen in any geo-replicated configuration, but the likelihood increases with tiering as the rebalance process is continuous.

BZ#1102524

The Geo-replication worker goes to faulty state and restarts when resumed. It works as expected when it is restarted, but takes more time to synchronize compared to resume.

BZ#1238699

The Changelog History API expects brick path to remain the same for a session. However, on snapshot restore, brick path is changed. This causes the History API to fail and geo-rep to change to **Faulty**.

Workaround:

1. After the snapshot restore, ensure the master and slave volumes are stopped.
2. Backup the **htime** directory (of master volume).

```
cp -a <brick_htime_path> <backup_path>
```

NOTE

Using **-a** option is important to preserve extended attributes.

For example:

```
cp -a
/var/run/gluster/snaps/a4e2c4647cf642f68d0f8259b43494c0/brick0/b0/.glusterfs/changeslog
s/htime  /opt/backup_htime/brick0_b0
```

3. Run the following command to replace the **OLD** path in the htime file(s) with the new brick path, where *OLD_BRICK_PATH* is the brick path of the current volume, and *NEW_BRICK_PATH* is the brick path after snapshot restore.

```
find <new_brick_htime_path> - name 'HTIME.*' -print0  | \
xargs -0 sed -ci 's|<OLD_BRICK_PATH>|<NEW_BRICK_PATH>|g'
```

For example:

```
find
/var/run/gluster/snaps/a4e2c4647cf642f68d0f8259b43494c0/brick0/b0/.glusterfs/changelogs
/htime/ -name 'HTIME.*' -print0  | \
xargs -0 sed -ci
's|/bricks/brick0/b0/|/var/run/gluster/snaps/a4e2c4647cf642f68d0f8259b43494c0/brick0/b0/|g
'
```

4. Start the Master and Slave volumes and Geo-replication session on the restored volume. The status should update to **Active**.

Issues related to Self-heal

BZ#1240658

When files are accidentally deleted from a brick in a replica pair in the back-end, and **gluster volume heal *VOLNAME* full** is run, then there is a chance that the files may not heal.

Workaround: Perform a lookup on the files from the client (mount). This triggers the heal.

BZ#1173519

If you write to an existing file and go over the **_AVAILABLE_BRICK_SPACE_**, the write fails with an I/O error.

Workaround: Use the **cluster.min-free-disk** option. If you routinely write files up to *n*GB in size, then you can set min-free-disk to an *m*GB value greater than *n*.

For example, if your file size is 5GB, which is at the high end of the file size you will be writing, you might consider setting min-free-disk to 8 GB. This ensures that the file will be written to a brick with enough available space (assuming one exists).

```
# gluster v set _VOL_NAME_ min-free-disk 8GB
```

Issues related to replace-brick operation

- After the **gluster volume replace-brick *VOLNAME* Brick *New-Brick* commit force** command is executed, the file system operations on that particular volume, which are in transit, fail.
- After a replace-brick operation, the stat information is different on the NFS mount and the FUSE mount. This happens due to internal time stamp changes when the **replace-brick** operation is performed.

Issues related to NFS

- After you restart the NFS server, the unlock within the grace-period feature may fail and the locks held previously may not be reclaimed.
- **fcntl** locking (NFS Lock Manager) does not work over IPv6.
- You cannot perform NFS mount on a machine on which glusterfs-NFS process is already running unless you use the NFS mount **-o nolock** option. This is because glusterfs-nfs has already registered NLM port with portmapper.
- If the NFS client is behind a NAT (Network Address Translation) router or a firewall, the locking behavior is unpredictable. The current implementation of NLM assumes that Network Address Translation of the client's IP does not happen.
- **nfs.mount-udp** option is disabled by default. You must enable it to use posix-locks on Solaris when using NFS to mount on a Red Hat Gluster Storage volume.

- If you enable the **nfs.mount-udp** option, while mounting a subdirectory (exported using the **nfs.export-dir** option) on Linux, you must mount using the **-o proto=tcp** option. UDP is not supported for subdirectory mounts on the GlusterFS-NFS server.

- For NFS Lock Manager to function properly, you must ensure that all of the servers and clients have resolvable hostnames. That is, servers must be able to resolve client names and clients must be able to resolve server hostnames.

Issues related to NFS-Ganesha

BZ#1570084

The **dbus** command used to export the volumes fails, if the volumes are exported before completing nfs-ganesha start up.

Workaround: Restart the nfs-ganesha process and then export the volumes.

BZ#1535849

In case of NFS-Ganesha, the memory created for a cache entry is recycled instead of freeing it. For example, if there is a file "foo" and it is removed from different client cache entry for "foo", it still exists. As a result, memory used by NFS-Ganesha will increase till cache is full.

BZ#1461114

While adding a node to an existing Ganesha cluster, the following error messages are displayed, intermittently:

> Error: Some nodes had a newer tokens than the local node. Local node's tokens were updated. Please repeat the authentication if needed

> Error: Unable to communicate with pcsd

Workaround: These messages can safely be ignored since there is no known functionality impact.

BZ#1402308

The Corosync service will crash, if ifdown is performed after setting up the ganesha cluster. This may impact the HA functionality.

BZ#1330218

If a volume is being accessed by heterogeneous clients (i.e, both NFSv3 and NFSv4 clients), it is observed that NFSv4 clients take longer time to recover post virtual-IP failover due to node shutdown.

Workaround: Use different VIPs for different access protocol (i.e, NFSv3 or NFSv4) access.

BZ#1416371

If **gluster volume stop** operation on a volume exported via NFS-ganesha server fails, there is a probability that the volume will get unexported on few nodes, inspite of the command failure. This will lead to inconsistent state across the NFS-ganesha cluster.

Workaround: To restore the cluster back to normal state, perform the following

- Identify the nodes where the volume got unexported

- Re-export the volume manually using the following dbus command:

```
# dbus-send --print-reply --system --dest=org.ganesha.nfsd /org/ganesha/nfsd/ExportMgr
org.ganesha.nfsd.exportmgr.AddExport string:/var/run/gluster/shared_storage/nfs-
ganesha/exports/export.<volname>.conf string:""EXPORT(Path=/<volname>)"""
```

BZ#1381416

When a READDIR is issued on directory which is mutating, the cookie sent as part of request could be of the file already deleted. In such cases, server returns **BAD_COOKIE** error. Due to this, some applications (like bonnie test-suite) which do not handle such errors may error out.

This is an expected behaviour of NFS server and the applications has to be fixed to fix such errors.

BZ#1398280

If any of the PCS resources are in the failed state, then the teardown requires a lot of time to complete. Due to this, the command **gluster nfs-ganesha disable** will timeout.

Workaround: If **gluster nfs-ganesha disable** encounters a timeout, perform the **pcs status** and check whether any resource is in failed state. Then perform the cleanup for that resource using following command:

```
# pcs resource --cleanup <resource id>
```

Re-execute the **gluster nfs-ganesha disable** command.

BZ#1328581

After removing a file, the nfs-ganesha process does a lookup on the removed entry to update the attributes in case of any links present. Due to this, as the file is deleted, lookup will fail with ENOENT resulting in a misleading log message in **gfapi.log**.

This is an expected behaviour and there is no functionality issue here. The log message needs to be ignored in such cases.

BZ#1259402

When vdsmd and abrt services are installed alongside each other, vdsmd overwrites abrt core dump configuration in **/proc/sys/kernel/core_pattern** file. This prevents NFS-Ganesha from generating core dumps.

Workaround: Set ***core_dump_enable*** to **false** in **/etc/vdsm/vdsm.conf** file to disable core dumps, then restart the **abrt-ccpp** service:

```
# systemctl restart abrt-ccpp
```

BZ#1257548

nfs-ganesha service monitor script which triggers IP failover runs periodically every 10 seconds. By default, the ping-timeout of the glusterFS server (after which the locks of the unreachable client gets flushed) is 42 seconds. After an IP failover, some locks are not cleaned by the glusterFS server process. Therefore, reclaiming the lock state by NFS clients fails.

Workaround: It is recommended to set the **nfs-ganesha** service monitor period interval (default 10s) to at least twice the Gluster server ping-timeout (default 42s) period.

Therefore, you must decrease the network ping-timeout by using the following command:

```
# gluster volume set <volname> network.ping-timeout <ping_timeout_value>
```

or increase nfs-service monitor interval time by using the following commands:

```
# pcs resource op remove nfs-mon monitor
```

```
# pcs resource op add nfs-mon monitor interval=<interval_period_value> timeout=
<timeout_value>
```

BZ#1470025

PCS cluster IP resources may enter FAILED state during failover/failback of VIP in NFS-Ganesha HA cluster. As a result, VIP is inaccessible resulting in mount failures or system freeze.

Workaround: Clean up the failed resource by using the following command:

```
# pcs resource cleanup resource-id
```

BZ#1474716

After a reboot, systemd may interpret NFS-Ganesha to be in STARTED state when it is not running.

Workaround: Manually start the NFS-Ganesha process.

BZ#1473280

Executing the **gluster nfs-ganesha disable** command stops the NFS-Ganesha service. In case of pre-exported entries, NFS-Ganesha may enter FAILED state.

Workaround: Restart the NFS-Ganesha process after failure and re-run the following command:

```
# gluster nfs-ganesha disable
```

Issues related to Object Store

- The GET and PUT commands fail on large files while using Unified File and Object Storage.

  Workaround: You must set the ***node_timeout=60*** variable in the proxy, container, and the object server configuration files.

Issues related to Red Hat Gluster Storage Volumes

BZ#1578703

Large number of inodes can cause the itable to be locked for a longer period during inode status dump. This behavior causes performance issues on clients command timeout on inode status dump.

Workaround: Reduce the LRU inodes when performing 'inode status' by running the following commands.

Set to small value:

```
# gluster v set v1 inode-lru-limit 256
```

Take inode dump:

```
# gluster v status v1 inode
```

Set to previous value:

```
# gluster v set v1 inode-lru-limit 16384
```

BZ#1286050

On a volume, when read and write operations are in progress and simultaneously a rebalance operation is performed followed by a remove-brick operation on that volume, then the **rm -rf** command fails on a few files.

BZ#1224153

When a brick process dies, BitD tries to read from the socket used to communicate with the corresponding brick. If it fails, BitD logs the failure to the log file. This results in many messages in the log files, leading to the failure of reading from the socket and an increase in the size of the log file.

BZ#1224162

Due to an unhandled race in the RPC interaction layer, brick down notifications may result in corrupted data structures being accessed. This can lead to NULL pointer access and segfault.

Workaround: When the **Bitrot** daemon (bitd) crashes (segfault), you can use **volume start *VOLNAME* force** to restart **bitd** on the node(s) where it crashed.

BZ#1227672

A successful scrub of the filesystem (objects) is required to see if a given object is clean or corrupted. When a file is corrupted and a scrub has not been run on the filesystem, there is a good chance of replicating corrupted objects in cases when the brick holding the good copy was offline when I/O was performed.

Workaround: Objects need to be checked on demand for corruption during healing.

Issues related to Samba

BZ#1329718

Snapshot volumes are read-only. All snapshots are made available as directories inside .snaps directory. Even though snapshots are read-only, the directory attribute of snapshots is same as the directory attribute of root of snapshot volume, which can be read-write. This can lead to confusion, because Windows will assume that the snapshots directory is read-write. **Restore previous version** option in file properties gives **open** option. It will open the file from the corresponding snapshot. If opening of the file also creates temp files (for example, Microsoft Word files), the open will fail. This is because temp file creation will fail because snapshot volume is read-only.

Workaround: Copy such files to a different location instead of directly opening them.

BZ#1322672

When vdsm and abrt's ccpp addon are installed alongside each other, vdsmd overwrites abrt's core dump configuration in /proc/sys/kernel/core_pattern. This prevents Samba from generating core dumps due to SELinux search denial on new coredump location set by vdsmd.

Workaround: To workaround this issue, execute the following steps:

1. Disable core dumps in **/etc/vdsm/vdsm.conf**:

```
core_dump_enable = false
```

2. Restart the abrt-ccpp and smb services:

```
# systemctl restart abrt-ccpp
# systemctl restart smb
```

BZ#1300572

Due to a bug in the Linux CIFS client, SMB2.0+ connections from Linux to Red Hat Gluster Storage currently will not work properly. SMB1 connections from Linux to Red Hat Gluster Storage, and all connections with supported protocols from Windows continue to work.

Workaround: If practical, restrict Linux CIFS mounts to SMB version 1. The simplest way to do this is to not specify the **vers mount** option, since the default setting is to use only SMB version 1. If restricting Linux CIFS mounts to SMB1 is not practical, disable asynchronous I/O in Samba by setting **aio read size** to 0 in **smb.conf** file. Disabling asynchronous I/O may have performance impact on other clients

BZ#1282452

Attempting to upgrade to ctdb version 4 fails when ctdb2.5-debuginfo is installed, because the ctdb2.5-debuginfo package currently conflicts with the samba-debuginfo package.

Workaround: Manually remove the ctdb2.5-debuginfo package before upgrading to ctdb version 4. If necessary, install samba-debuginfo after the upgrade.

BZ#1164778

Any changes performed by an administrator in a Gluster volume's share section of **smb.conf** are replaced with the default Gluster hook scripts settings when the volume is restarted.

Workaround: The administrator must perform the changes again on all nodes after the volume restarts.

Issues related to SELinux

BZ#1256635

Red Hat Gluster Storage does not currently support SELinux Labeled mounts.

On a FUSE mount, SELinux cannot currently distinguish file systems by subtype, and therefore cannot distinguish between different FUSE file systems (BZ#1291606). This means that a client-specific policy for Red Hat Gluster Storage cannot be defined, and SELinux cannot safely translate client-side extended attributes for files tracked by Red Hat Gluster Storage.

A workaround is in progress for NFS-Ganesha mounts as part of BZ#1269584. When complete, BZ#1269584 will enable Red Hat Gluster Storage support for NFS version 4.2, including SELinux Labeled support.

BZ#1291194 , BZ#1292783

Current SELinux policy prevents ctdb's 49.winbind event script from executing smbcontrol. This can create inconsistent state in winbind, because when a public IP address is moved away from a node, winbind fails to drop connections made through that IP address.

Issues related to Sharding

BZ#1520882 , BZ#1568758

When large number of shards are deleted in a large file, the shard translator synchronously sends unlink operation on all the shards in parallel. This action causes replicate translator to acquire locks on the .shard directory in parallel.

After a short period, large number of locks get accumulated in the locks translatory, and the search for possible matching locks is slowed down, sometimes taking several minutes to complete. This behaviour causes timeouts leading to disconnects and also subsequent failure of file deletion leading to stale shards being left out under the .shard directory.

Workaround: Use shard block size of 64 MB as the lower the shard-block-size, the higher the chances of timeouts.

BZ#1332861

Sharding relies on block count difference before and after every write as gotten from the underlying file system and adds that to the existing block count of a sharded file. But XFS' speculative preallocation of blocks causes this accounting to go bad as it may so happen that with speculative preallocation the block count of the shards after the write projected by xfs could be greater than the number of blocks actually written to.

Due to this, the block-count of a sharded file might sometimes be projected to be higher than the actual number of blocks consumed on disk. As a result, commands like **du -sh** might report higher size than the actual number of physical blocks used by the file.

General issues

GFID mismatches cause errors

If files and directories have different GFIDs on different back-ends, the glusterFS client may hang or display errors. Contact Red Hat Support for more information on this issue.

BZ#1236025

The time stamp of files and directories changes on snapshot restore, resulting in a failure to read the appropriate change logs. **glusterfind pre** fails with the following error: **historical changelogs not available**. Existing glusterfind sessions fail to work after a snapshot restore.

Workaround: Gather the necessary information from existing glusterfind sessions, remove the sessions, perform a snapshot restore, and then create new glusterfind sessions.

BZ#1573083

When **storage.reserve** limits are reached and a directory is created, the directory creation fails with ENOSPC error and lookup on the directory throws ESTALE errors. As a consequence, file operation is not completed.

Workaround: No workaround is available.

BZ#1578308

Stale statistics cached in the **md-cache** and **readdir-ahead**, fail to get updated on write operations from the application. As a result, the application does not see the effect of write operations like size from the statistics which does not reflect the writes that are successfully completed.

Workaround: Turn off performance.stat-prefetch and performance.readdir-ahead options and the application will no longer receive stale statistics.

BZ#1260119

glusterfind command must be executed from one node of the cluster. If all the nodes of cluster are not added in **known_hosts** list of the command initiated node, then **glusterfind create** command hangs.

Workaround: Add all the hosts in peer including local node to **known_hosts**.

BZ#1058032

While migrating VMs, libvirt changes the ownership of the guest image, unless it detects that the image is on a shared filesystem and the VMs cannot access the disk images as the required ownership is not available.

Workaround: Before migration, power off the VMs. When migration is complete, restore the ownership of the VM Disk Image (107:107) and start the VMs.

BZ#1127178

When a replica brick comes back online, it might have self-heal pending. Executing the **rm -rf** command on the brick will fail with the error message *Directory not empty*.

Workaround: Retry the operation when there are no pending self-heals.

BZ#1460629

When the command **rm -rf** is executed on the parent directory, which has a pending self-heal entry involving purging files from a sink brick, the directory and files awaiting heal may not be removed from the sink brick. Since the readdir for the **rm -rf** will be served from the source brick, the file pending entry heal is not deleted from the sink brick. Any data or metadata which is pending heal on such a file are displayed in the output of the command **heal-info**, until the issue is fixed.

Workaround: If the files and parent directory are not present on other bricks, delete them from the sink brick. This ensures that they are no longer listed in the next 'heal-info' output.

BZ#1462079

Due to incomplete error reporting, statedump is not generated after executing the following command:

```
# gluster volume statedump volume client host:port
```

Workaround: Verify that the ***host:port*** is correct in the command.

The resulting statedump file(s) are placed in **/var/run/gluster** on the host running the gfapi application.

Issues related to Red Hat Gluster Storage AMI

BZ#1267209

The redhat-storage-server package is not installed by default in a Red Hat Gluster Storage Server 3 on Red Hat Enterprise Linux 7 AMI image.

Workaround: It is highly recommended to manually install this package using yum.

```
# yum install redhat-storage-server
```

The redhat-storage-server package primarily provides the **/etc/redhat-storage-release** file, and sets the environment for the storage node. package primarily provides the **/etc/redhat-storage-release** file, and sets the environment for the storage node.

Issues related to Red Hat Gluster Storage Web Administration

BZ#1622461

When central store (etcd) is stopped which could happen either due to stopping of etcd or shutting down the Web Administration server node itself, all the Web Administration services start reporting exceptions regarding reachability to the etcd. As a consequence, Web Administration services crash as etcd is not reachable.

Workaround: Once etcd is back, restart Web Administration services.

4.2. RED HAT GLUSTER STORAGE AND RED HAT ENTERPRISE VIRTUALIZATION INTEGRATION

All images in data center displayed regardless of context

In the case that the Red Hat Gluster Storage server nodes and the Red Hat Enterprise Virtualization Hypervisors are present in the same data center, the servers of both types are listed for selection when you create a virtual machine or add a storage domain. Red Hat recommends that you create a separate data center for the Red Hat Gluster Storage server nodes.

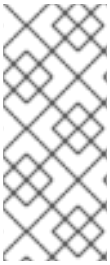
CHAPTER 5. TECHNOLOGY PREVIEWS

This chapter provides a list of all available Technology Preview features in this release.

Technology Preview features are currently not supported under Red Hat Gluster Storage subscription services, may not be functionally complete, and are generally not suitable for production environments. However, these features are included for customer convenience and to provide wider exposure to the feature.

Customers may find these features useful in a non-production environment. Customers are also free to provide feedback and functionality suggestions for a Technology Preview feature before it becomes fully supported. Errata will be provided for high-severity security issues.

During the development of a Technology Preview feature, additional components may become available to the public for testing. Red Hat intends to fully support Technology Preview features in future releases.



NOTE

All Technology Preview features in Red Hat Enterprise Linux 6.10 and 7.5 are also considered technology preview features in Red Hat Gluster Storage 3.4. For more information on the technology preview features available in Red Hat Enterprise Linux 6.10, see the Technology Previews chapter of the *Red Hat Enterprise Linux 6.10 Technical Notes*. For Red Hat Enterprise Linux 7.5, see Technology Previews.

5.1. SMB MULTI-CHANNEL

Multi-Channel is an SMB version 3 feature that allows the client to bind multiple transport connections into one authenticated SMB session. This allows for increased fault tolerance and throughput for clients running Windows 8 and newer and Windows Server 2012 and newer.

See SMB3 Multi-Channel with Samba on Red Hat Gluster Storage (Technology Preview) for more information.

APPENDIX A. REVISION HISTORY

Revision 3.4-2 **Thu Oct 04 2018** **Red Hat Gluster Storage Documentation Team**

Improved the bug descriptions, fixes/workarounds in Chapter 3 and 4.

Revision 3.4-1 **Tue Sep 04 2018** **Red Hat Gluster Storage Documentation Team**

Updates for Red Hat Gluster Storage 3.4 release.