Red Hat Gluster Storage 3.3

3.3 Release Notes

Release Notes for Red Hat Gluster Storage 3.3

Edition 1

Last Updated: 2018-02-07

Red Hat Gluster Storage 3.3 3.3 Release Notes

Release Notes for Red Hat Gluster Storage 3.3
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 3.3.

Table of Contents

CHAPTER 1. INTRODUCTION **3**

CHAPTER 2. WHAT CHANGED IN THIS RELEASE? **4**
2.1. WHAT'S NEW IN THIS RELEASE? 4
2.2. DEPRECATED FEATURES 7

CHAPTER 3. NOTABLE BUG FIXES **8**

CHAPTER 4. KNOWN ISSUES **14**
4.1. RED HAT GLUSTER STORAGE 14
4.2. RED HAT GLUSTER STORAGE CONSOLE 31
4.3. RED HAT GLUSTER STORAGE AND RED HAT ENTERPRISE VIRTUALIZATION INTEGRATION 34

CHAPTER 5. TECHNOLOGY PREVIEWS **35**
5.1. STOP REMOVE BRICK OPERATION 35
5.2. SMB MULTI-CHANNEL 35
5.3. READ-ONLY VOLUME 35
5.4. PNFS 35

APPENDIX A. REVISION HISTORY **37**

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 3.3 release.

Resource limits now configurable through gdeploy

Administrators can now set limits on the resources that Red Hat Gluster Storage has access to by configuring a slice for gluster processes in their gdeploy configuration files.

For more information, see Managing Resource Usage in *Red Hat Gluster Storage 3.3 Administration Guide*.

statedump feature for application using gfapi enhancement

The statedump feature supports gathering of information from application using gfapi. With Red Hat Gluster Storage 3.3, administrators can take statedump for application using gfapi using the following command:

```
# gluster volume statedump vol host:pid
```

Executing the command places the statedump in the **/var/run/gluster** directory. The user running the application must have write permissions to **/var/run/gluster**.

As **gluster** group is created for application using gfapi, it does not require root privilege. The user executing the application needs must be added to the **gluster** group using the following command:

```
# usermod -a -G gluster qemu
```

For more information, see Viewing complete volume state with statedump in *Red Hat Gluster Storage 3.3 Administration Guide*.

Gluster block storage

Red Hat Gluster Storage 3.3 introduces block storage for containerized workloads with the gluster-block command line utility. Block storage supports only Container-Native Storage (CNS) and Container-Ready Storage(CRS) use cases. Currently, it enables the creation of high-performance individual storage units, allowing each unit to be treated as an independent disk drive and to support an individual file system.

Gluster-block aims to make the creation and maintenance of Gluster-backed block storage as simple as possible. With gluster-block you can provision block devices, export them as iSCSI LUNs across multiple nodes, and use the iSCSI protocol for data transfer.

For more information, see Storage Concepts in *Red Hat Gluster Storage 3.3 Administration Guide*.

Brick Multiplex support

Red Hat Gluster Storage 3.3 introduces brick multiplexing which supports only Container-Native Storage (CNS) and Container-Ready Storage(CRS) use cases. It allows administrators to reduce the number of ports and processes used by gluster bricks on the same server. When brick multiplexing is enabled, compatible bricks on the same server share a port and a process. Thus reducing per-brick memory usage and port consumption.

For more information, see Many Bricks per Node in *Red Hat Gluster Storage 3.3 Administration Guide*.

Performance improvements

`find` command on a volume

- High CPU usage was experienced when a **`find`** command was executed on a volume with large number of files as the command accessed the versioning data of the files.

  With this release, object versioning is enabled only when BitRot detection is enabled for a volume. Thus, CPU usage is lower when the **`find`** command is executed on a volume with large number of files. Hence, improving performance when BitRot daemon is in disabled state, and a **`find`** command is executed on a volume with large number of files.

  For more information, see Detecting BitRot in *Red Hat Gluster Storage 3.3 Administration Guide*.

Parallel **`readdirp`** support

- Now, **`readdirp fops`** are sent parallelly to all the bricks. This enhances the performance for find and a recursive listing of small directories.

  For more information, see *Enhancing Directory Listing Performance* under Tuning Performance in *Red Hat Gluster Storage 3.3 Administration Guide*.

Negative lookup cache for Samba

- With Red Hat Gluster Storage 3.3, negative lookup caching is developed for Samba to improve the small file workloads. This removes redundant lookup operations, improving speed for file and directory creation from Samba clients.

  For more information, see Enhancing File/Directory Create Performance in *Red Hat Gluster Storage 3.3 Administration Guide*.

Enhancement to `glusterfind` command

The command **`glusterfind`** now provides a **`query`** sub command that provides a list of changed files.

For more information, see *Glusterfind Query* under Glusterfind Configuration Options in *Red Hat Gluster Storage 3.3 Administration Guide*.

Configurable chunksize

Users can now configure the chunksize from the backend-setup. It is simpler compared to ‘pv’, ‘vg’ or ‘lv’. Previously, ‘lv’ module was the only way to use chunksize as a parameter.

Dynamic update of export configuration options

Most NFS-Ganesha export configuration options can be updated dynamically during normal operation without needing to export and re-export the volume.

Enhancement to `geo-replication status` command

The detailed **`geo-replication status`** command no longer requires master volume, slave host, and slave volume. It can be executed with or without these additional details. For example:

```
# gluster volume geo-replication status detail
```

For more information, see Displaying Geo-replication Status Information in *Red Hat Gluster Storage 3.3 Administration Guide*.

RAID 5 disk support

RAID 5 disks are now supported. RAID 5 disk-type is supported in gdeploy along with JBOD, RAID 6 and RAID 10.

For more information, see Configuration File in *Red Hat Gluster Storage 3.3 Administration Guide*.

NFS-Ganesha installation without firewalld

The dependency of firewalld for NFS-Ganesha installation has now been removed. Now, NFS-Ganesha is installed without firewalld.

For more information, see Prerequisites to run NFS-Ganesha in *Red Hat Gluster Storage 3.3 Administration Guide*.

gluster-swift packages updated to OpenStack Swift Newton 2.10.1

The gluster-swift packages are updated to OpenStack Swift Newton 2.10.1 to integrate with the supported version of OpenStack Swift as the previous version of OpenStack Swift (Kilo) has reached end of support.

Enhanced `rebalance` status

The command **`gluster volume volname rebalance status`** now provides an estimate of the time left to rebalance completion. Note that calculations are based on each brick having its own file system partition.

For more information, see Displaying Rebalance Progress in *Red Hat Gluster Storage 3.3 Administration Guide*.

Volume extension using Heketi

A new section has been added to the *Red Hat Gluster Storage 3.3 Administration Guide*, which contains instructions to expand a volume using Heketi.

For more information, see Expanding a Volume in *Red Hat Gluster Storage 3.3 Administration Guide*.

NFS chapter restructure

The NFS chapter in the *Red Hat Gluster Storage 3.3 Administration Guide* is re-structured with significant improvements. Some of the key updates are:

- Clear division between Gluster-NFS and NFS-Ganesha.
- For better clarity, large sections in the NFS-Ganesha chapter are divided into smaller sub sections.
- To improve usability and ease of deployment/configuration, flow of content is revisited and additional details are added in the NFS-Ganesha section.

For more information, see NFS in *Red Hat Gluster Storage 3.3 Administration Guide*.

Dispersed Volume Enhancements - new variants supported

Dispersed volumes are based on erasure coding. Erasure coding is a method of data protection in which data is broken into fragments, expanded and encoded with redundant data pieces and stored across a set of different locations. This allows the recovery of the data stored on one or more bricks in case of failure. Dispersed volume requires less storage space when compared to a replicated volume. The following new variants are supported with this release:

- 10 bricks with redundancy level 2 (8 + 2)
- 20 bricks with redundancy level 4 (16 + 4)

For more information, see Expanding a Volume in *Red Hat Gluster Storage 3.3 Administration Guide*.

2.2. DEPRECATED FEATURES

The following features are deprecated as of Red Hat Gluster Storage 3.3, or will be considered deprecated in subsequent releases. Review individual items for details about the likely removal time frame of the feature.

Two-way Replication

As of Red Hat Gluster Storage 3.3, two-way replication is considered deprecated. Two-way replication 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. This change affects both replicated and distributed-replicated volumes.

Two-way replication is being deprecated because it does not provide adequate protection from split-brain conditions. Even in distributed-replicated configurations, two-way replication cannot ensure that the correct copy of a conflicting file is selected without the use of a tie-breaking node.

While a dummy node can be used as an interim solution for this problem, Red Hat recommends that all volumes that currently use two-way replication are migrated to use either arbitrated replication or three-way replication.

For instructions to migrate a two-way replicated volume to an arbitrated replicated volume, see Converting to an arbitrated volume in the *Red Hat Gluster Storage 3.3 Administration Guide*.

Further information about replicated and arbitrated replicated volumes is also available in the *Red Hat Gluster Storage 3.3 Administration Guide*:

- Arbitrated replicated volumes
- Replicated volumes
- Distributed replicated volumes

NFS-Ganesha on Red Hat Enterprise Linux 6 based Red Hat Gluster Storage

As of Red Hat Gluster Storage 3.3, new installations using NFS-Ganesha on Red Hat Enterprise Linux 6 based Red Hat Gluster Storage is not supported. Existing Red Hat Enterprise Linux 6 based installations are advised to migrate to Red Hat Enterprise Linux 7 based Red Hat Gluster Storage in order to continue receiving updates for NFS-Ganesha.

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.

common-ha

BZ#1426523

Stale export entries were not cleaned up correctly from the **`ganesha.conf`** file when NFS-Ganesha was disabled; this has now been corrected so that stale export entries are removed.

BZ#1441055

During NFS-Ganesha cluster setup, the pcsd service first destroyed any existing cluster, which disabled the pacemaker service. This meant that the pacemaker service did not start automatically after a reboot. The pacemaker service is now explicitly re-enabled after successful NFS-Ganesha cluster setup, and the pacemaker service is started automatically after node reboot.

core

BZ#1449684

Previously, certain FOPs (fentrylk, rchecksum, seek, lease, getactivelk, setactivelk, compound fops) were missing from volume profile info output. These are now captured as expected.

distribute

BZ#1260779

The **`getfattr -n replica.split-brain-status`** ***`path-to-dir`*** command now shows accurate split brain status.

BZ#1381142

Rebalance throttling previously had three modes; however, two of these modes had similar performance because of bottlenecks in thread management. This has been corrected. In addition, rebalance throttling can now be configured by specifying a number of migration threads to use as well as with the previous lazy, normal, and aggressive modes.

BZ#1409474

A bug in the remove-brick code can cause file migration on some files with multiple hard links to fail. Files may be left behind on the removed brick. These will not be available on the gluster volume once the remove-brick operation is committed.

Workaround: Once the remove-brick operation is complete, check for any files left behind on the removed bricks and copy them to the volume via a mount point.

BZ#1411352

Renaming a file could result in duplicate files if multiple clients had caching enabled. This occurred because the lookup operation to verify file existence on the volume was based on the file's GFID and therefore always succeeded. The lookup is now based on the file's name instead of GFID, and duplicates no longer occur in this situation.

BZ#1442943

The maximum block size for rebalance operations (DHT_REBALANCE_BLKSIZE) has been increased to 1MB from the previous value of 128KB in order to improve rebalance operation time for large files.

BZ#1445195

Directories that contained only link to files appeared empty when accessed through the mount point. This meant that when users attempted to remove these directories using **`rmdir`**, the remove operation failed with a 'Directory not empty' error. The **`rmdir`** operation has been updated to check for and delete all stale link to files in an otherwise empty directory, and now removes the directory as expected in this circumstance.

geo-replication

BZ#1414750

Geo-replication workers now have separate slave mount logs to make debugging easier. Log files are named according to the following format:

```
master_volume-uuid:master-host:master-brickpath:slavevol.gluster.log
```

gluster-nfs

BZ#1372283

Sub directories of a volume can now be mounted on Solaris 10 clients using WebNFS.

glusterd

BZ#1351185

The output of the **`gluster volume status volname clients`** command has been enhanced to show client operating versions so that compatibility is easier to determine. This works only for clients using Red Hat Gluster Storage 3.3 or higher.

glusterfs

BZ#1445570

Access to the **`/var/run/gluster/`** directory is typically restricted. As a result, attempting to write a state dump to this directory fails. To write to this location, ensure that the user that runs the application is added to the 'gluster' user group, and restart gluster processes so that the new group is applied.

io-cache

BZ#1435357

The ioc_inode_wakeup process did not lock the ioc_inode queue. This meant that the ioc_prune process could free a structure that ioc_inode_wakeup later attempted to access, resulting in an unexpected termination of the gluster mount process. The ioc_inode queue is now locked during access so that this issue cannot occur.

libgfapi

BZ#1378085

The statedump feature now supports gathering information from gfapi applications.

quota

BZ#1414758

Quota list and limit commands now create and destroy aux mounts each time they are run to reduce the risk of encountering stale mount points. Additionally, a separate mount point is now used for list and limit commands in order to avoid issues when these commands are run in parallel.

replicate

BZ#1315781

The rebalance process uses an extended attribute to determine which node migrates a file. In replicated and erasure-coded (dispersed) volumes, only the first node of a replica set was listed in this attribute, so only the first node of a replica set migrated files. Replicated and erasure-coded volumes now list all nodes in a replica set, ensuring that rebalance processes on all nodes migrate files as expected.

sharding

BZ#1447959

The checksum of a file could change when it was copied from a local file system to a volume with sharding enabled. If write and truncate operations were in progress simultaneously, the aggregated size was calculated incorrectly, resulting in a changed checksum. Aggregated file size is now calculated correctly in this circumstance.

snapshot

BZ#1309209

The names and locations of previously cloned and deleted volumes were not cleaned up correctly. This meant that creating a clone with the same name as a previously deleted clone failed with a 'Commit failed' message. Cleanup is now handled correctly and the same name can be used for a clone in this situation.

write-behind

BZ#1297743

The write-behind-window-size parameter was not validated correctly, and could be set to a value greater than its allowed maximum. This has been corrected so that only valid values (524288 - 1073741824) can be set.

gdeploy

BZ#1417596

Previously, checking for the gdeploy version would end up exiting if PyYAML package was not installed. Now PyYAML package check is not done for commands like **`--version`** or **`-- help`**.

BZ#1394796

With this release, *volname* is made optional while creating NFS Ganesha cluster, which was mandatory before. You might want to create a NFS Ganesha cluster without exporting or building any volumes i.e. creating a cluster without having any gluster volume.

BZ#1405966

Previously, gdeploy would fail if a subscribed user tries to subscribe again as subscription-manager would report error stating user already subscribed. Since the failure did not report to anything fatal, it can be ignored. Users can also debug in case they face any genuine subscription-manager error.

BZ#1452001

Previously, **script** module was used which would result in gdeploy failure if NFS-Ganesha packages are not installed in the local system. From this release onward, **shell** module will be used which runs gdeploy without having the packages installed on the local system.

nfs-ganesha

BZ#1421130

This rebase includes the following enhancements:

- Rebase package(s) to version:
  - nfs-ganesha-2.4.4
  - nfs-ganesha-gluster-2.4.4
  - nfs-ganesha-debuginfo-2.4.4
- Includes the dynamic update export feature.

BZ#1425753

When multiple paths with the same parent volume are exported via NFS-Ganesha, the handles maintained by the server of the files/directories common to those paths would get merged. Due to this, unexporting one of those shares may result in segmentation fault of the server when accessed via another share mount-.

With this fix, the refcount of such shared objects is maintained and are released only when all the exports accessing them are unexported. There is hence no issues accessing one share while unexporting another one which shares the same parent volume.

BZ#1451981

The NFS-Ganesha configuration file is stored in shared storage and if the shared storage is not mounted, then the NFS-Ganesha service will not start.

With this fix, system init scripts have been defined and updated to make sure that shared storage is mounted before starting the NFS-Ganesha service and NFS-Ganesha will start post reboot.

samba

BZ#1428368 and BZ#1436265

In the samba configuration, by default the 'posix locking' is enabled and 'stat cache' is disabled. Enabling 'posix locking' sends the file lock request to the bricks too, and disabling 'stat cache' blocks samba to cache certain information at the samba layer. This led to decrease in performance of SMB access of Red Hat Gluster Storage volumes.

As a fix, the following two options are included in the Samba configuration file:

- posix locking = No
- stat cache = Yes

Due to this, a slight improvement in the performance is observed.

vulnerability

BZ#1429472

A race condition was found in samba server. A malicious samba client could use this flaw to access files and directories, in areas of the server file system not exported under the share definitions.

BZ#1459464

A flaw was found in the way Samba handled dangling symlinks. An authenticated malicious Samba client could use this flaw to cause the smbd daemon to enter an infinite loop and use an excessive amount of CPU and memory.

gluster-nagios-addons

BZ#1425724 and BZ#1451997

Gluster monitoring stops working when the default NRPE config file is overwritten due to use of configuration management tools. With this update, the gluster command definitions are moved to a custom folder. Hence, the Gluster command definitions are retained and they are not affected by multiple tools writing to the default NRPE config file.

gluster-swift

BZ#1447684 and BZ#1451998

Previously, volume names with underscore symbol [_] could not be used, as swift-gen-builders syntax recognized underscore symbol as a delimiter for device and metadata. With this update, you can now use a verbose syntax with swift-gen-builders and volume names can now contain underscores.

gstatus

BZ#1458249

Previously, gstatus expected the PATH environment variable to be set, and ran the executable using only the executable name. This meant that when the PATH variable was reset to an empty string, the cluster executable could not be located and gstatus failed with an IOError. gstatus now ensures that the environment path is set so that executables can be located in this situation.

BZ#1454544

In deployments with a very large number of volumes, gstatus timed out before it was able to finish gathering data and generating a status report. When this happened, the status report was not provided to users. The timeout value for this process has been increased so that this issue no longer occurs.

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#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#1425681

Running volume rebalance/volume profile commands concurrently from all the nodes can cause one of the glusterd instance in a node to hold a volume lock for ever. Due to this, all the further commands on the same volume will fail with `another transaction is in progress` or `locking failed` error message. This is primarily seen when sosreport is executed on all the nodes at a same time.

Workaround: Restart the glusterd instance on the node where the stale lock exists.

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.

BZ#1369420

AVC denial message is seen on port 61000 when glusterd is (re)started.

Workaround: Execute **`setsebool -P nis_enabled on`** and restart glusterd.

Issues related to gdeploy

BZ#1408926

Currently the **`ssl_enable`** option is part of the **`volume`** section. It is a site wide change. If more than one volume is used in the same configuration (and within the same set of servers) and **`ssl_enable`** is set in all the volume sections, then the ssl operation steps are performed multiple times. This causes the older volumes to fail to mount. Users will then not be able to set SSL automatically with a single line of configuration.

Workaround: If there are more than one volume on a node. Set the variable**`enable_ssl`** under one [volume] section and set the keys: '**`client.ssl`**', value: 'on'; '**`server.ssl`**', value: 'on';'auth.ssl-allow', value: <comma separated ssl hosts>

Issues related to Arbiter Volumes

BZ#1387494

If the data bricks of the arbiter volume get filled up, further creation of new entries might succeed in the arbiter brick despite failing on the data bricks with ENOSPC and the application (client) itself receiving an error on the mount point. Thus the arbiter bricks might have more entries. Now, when an **`rm -rf`** is performed from the client, if the **`readdir`** (as a part of **`rm -rf`**) gets served on the data brick, it might delete only those entries and not the ones present only in the arbiter. When the **`rmdir`** on the parent dir of these entries comes, it won't succeed on the arbiter (errors out with ENOTEMPTY), leading to it not being removed from arbiter.

Workaround: If the deletion from the mount did not complain but the bricks still contain the directories, we would need to remove the directory and its associated gfid symlink from the back end. If the directory contains files, they (file + its gfid hardlink) would need to be removed too.

BZ#1388074

If some of the bricks of a replica or arbiter sub volume go down or get disconnected from the client while performing **`rm -rf`**, the directories may re-appear on the back end when the bricks come up and self-heal is over. When the user again tries to create a directory with the same name from the mount, it may heal this existing directory into other DHT subvols of the volume.

Workaround: If the deletion from the mount did not complain but the bricks still contain the directories, the directory and its associated gfid symlink must be removed from the back end. If the directory contains files, they (file + its gfid hardlink) would need to be removed too.

BZ#1361518

If a file create is wound to all bricks, and it succeeds only on arbiter, the application will get a failure. But during self-heal, the file gets created on the data bricks with arbiter marked as source. Since data self-heal can never happen from arbiter, "heal-info" will list the entries forever.

Workaround: If 'gluster vol heal <volname> info` shows the pending heals for a file forever, then check if the issue is the same as mentioned above by

1. checking that trusted.afr.volname-client* xattrs are zero on the data bricks

2. checking that trusted.afr.volname-client* xattrs is non-zero on the arbiter brick *only* for the data part (first 4 bytes)

   For example:

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

3. If it is in the above mentioned state, then delete the xattr:

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

Issues related to Distribute (DHT) Translator

BZ#1118770

There is no synchronization between **`mkdir`** and directory creation as part of self heal. This results in scenarios where **`rmdir`** or rename can proceed and remove the directory while mkdir is completed only on some subvolumes of DHT. Post completion of **`rmdir`** or **`rename`**, **`mkdir`** recreates the just removed or renamed directory with same gfid. Due to this, in the case of rename, both source and destination directories with the same gfid are present. In the case of **`rmdir`**, the directory can be present on some subvols even after **`rmdir`** and it can be healed back. In both cases of rename or rmdir, the directory may not be visible on mount point and hence **`rm -rf`** of parent directory will fail with an error "Directory not empty"

Workaround: As a workaround, follow the following steps:

1. If **`rm -rf`** ***`dir`*** fails with ENOTEMPTY for *dir*, check whether *dir* contains any sub directories on the bricks. If present, then delete them.

2. If post rename both the source and destination directories exist with the same gfid, then please contact redhat support for assistance.

BZ#1136718

The AFR self-heal can leave behind a partially healed file if the brick containing AFR self-heal source file goes down in the middle of heal operation. If this partially healed file is migrated before the brick that was down comes online again, 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 taken when a create is in progress the file may be present in one brick of the replica and not the other on the snapshotted volume. Due to this, when this snapshot is restored and a rm -rf is executed on a directory from the mount, it may fail with

ENOTEMPTY.

Workaround: If you get an ENOTEMPTY during **`rm -rf dir`**, but **`ls`** 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. Perform a stat of that file name from the mount so that it is healed to all bricks of the replica. Now when you do `**`rm -rf dir`**`, it should succeed.

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 snapshots which were created from 3.1.x or earlier does not have that 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, follow these steps:

1. Check if the volume has become a tiered volume.
   - If not, then rerun attach tier.
   - 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, then execute the following command:

     ```
     # gluster volume tier <volname> start
     ```

BZ#1303298

Listing the entries on a snapshot of a tiered volume shows incorrect permissions for some files. This is because the USS returns the stat information for the linkto files in the cold tier instead of the actual data file and 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`** (recommended)
- **`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 is occurring on an NFS mount, I/O pauses temporarily, usually for between 3 to 5 minutes. 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.

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#1201820

When a snapshot is deleted, the corresponding file system object in the User Serviceable Snapshot is also deleted. Any subsequent file system access results in the **`snapshot`** daemon becoming unresponsive. To avoid this issue, ensure that you do not perform any file system operations on the snapshot that is about to be deleted.

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#1133861

New snap bricks fails to start if the total snapshot brick count in a node goes beyond 1K. Until this bug is corrected, Red Hat recommends deactivating unused snapshots to avoid hitting the 1K limit.

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.

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 Nagios

BZ#1327017

Log messages related to quorum being regained are missed by Nagios server as it is either shutdown or has communication issues with nodes. Due to this, if Cluster Quorum status was critical prior to connection issues, then it continues to remain so.

Workaround: Administrator can check the alert from the Nagios UI and once the quorum is regained, the plugin result can be manually changed using "Submit passive check result for this service" option from the service page

BZ#1136207

Volume status service shows *All bricks are Up* message even when some of the bricks are in unknown state due to unavailability of **`glusterd`** service.

BZ#1109683

When a volume has a large number of files to heal, the **`volume self heal info`** command takes time to return results and the nrpe plug-in times out as the default timeout is 10 seconds.

Workaround: In **`/etc/nagios/gluster/gluster-commands.cfg`** increase the timeout of nrpe plug-in to 10 minutes by using the -t option in the command. For example:

```
$USER1$/gluster/check_vol_server.py $ARG1$ $ARG2$ -o self-heal -t 600
```

BZ#1094765

When certain commands invoked by Nagios plug-ins fail, irrelevant outputs are displayed as part of performance data.

BZ#1107605

Executing **`sadf`** command used by the Nagios plug-ins returns invalid output.

Workaround: Delete the datafile located at **`/var/log/sa/saDD`** where DD is current date. This deletes the datafile for current day and a new datafile is automatically created and which is usable by Nagios plug-in.

BZ#1107577

The Volume self heal service returns a WARNING when there unsynchronized entries are present in the volume, even though these files may be synchronized during the next run of self-heal process if **`self-heal`** is turned on in the volume.

BZ#1121009

In Nagios, CTDB service is created by default for all the gluster nodes regardless of whether CTDB is enabled on the Red Hat Gluster Storage node or not.

BZ#1089636

In the Nagios GUI, incorrect status information is displayed as *Cluster Status OK : None of the Volumes are in Critical State*, when volumes are utilized beyond critical level.

BZ#1111828

In Nagios GUI, Volume Utilization graph displays an error when volume is restored using its snapshot.

Issues related to Rebalancing Volumes

BZ#1286074

While Rebalance is in progress, adding a brick to the cluster displays an error message, **`failed to get index`** in the gluster log file. This message can be safely ignored.

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#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.

This means that, 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 because 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.

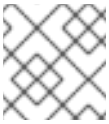
BZ#1238699

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/changeslogs/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/a4e2c4647cf642f68d0f8
259b43494c0/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#1230092

When you create a replica 3 volume, client quorum is enabled and set to **`auto`** by default. However, it does not get displayed in **`gluster volume info`**.

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 get healed.

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 help 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#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 a 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** is encounters a timeout, then 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 are installed alongside each other, vdsmd overwrites abrt core dump configuration in **/proc/sys/kernel/core_pattern**. This prevents NFS-Ganesha from generating core dumps.

Workaround: Disable core dumps in **/etc/vdsm/vdsm.conf** by setting ***core_dump_enable*** to **false**, and 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. The ping-timeout of the glusterFS server (after which the locks of the unreachable client gets flushed) is 42 seconds by default. After an IP failover, some locks may not get cleaned by the glusterFS server process, hence reclaiming the lock state by NFS clients may fail.

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

Hence, either you must decrease the network ping-timeout using the following command:

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

or increase nfs-service monitor interval time 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#1226874

If NFS-Ganesha is started before you set up an HA cluster, there is no way to validate the cluster state and stop NFS-Ganesha if the set up fails. Even if the HA cluster set up fails, the NFS-Ganesha service continues running.

Workaround: If HA set up fails, run service nfs-ganesha stop on all nodes in the HA cluster.

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 resource that failed, using the following command:

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

BZ#1461507

When duplicate request cache (DRC) entries maintained by NFS-Ganesha server reaches the high watermark limit, the server tries to reclaim old entries which may still be in use. As a result, every time the server cannot reclaim an entry, it logs a warning. This may flood the log file at times if there are too many requests being processed.

Workaround: Increase the DRC limit by executing the following steps:

1. Edit the **/run/gluster/shared_storage/nfs-ganesha/ganesha.conf** file and add the following parameters in NFS_Core_Param block:

```
NFS_Core_Param
{
  DRC_TCP_Hiwat = 1024; #default is 256
}
```

2. Restart the NFS-Ganesha process on all the nodes in the NFS-Ganesha cluster using the following command:

```
# systemctl restart nfs-ganesha
```

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

The command **`gluster nfs-ganesha disable`** when executed 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 rerun 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#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 gets 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.

BZ#1241336

When an Red Hat Gluster Storage node is shut down due to power failure or hardware failure, or when the network interface on a node goes down abruptly, subsequent gluster commands may time out. This happens because the corresponding TCP connection remains in the **`ESTABLISHED`** state. You can confirm this by executing the following command: **`ss -tap state established '( dport = :24007 )' dst`** ***`IP-addr-of-powered-off-RHGS-node`***

Workaround: Restart **`glusterd`** service on all other nodes.

BZ#1223306

`gluster volume heal VOLNAME info` shows stale entries, even after the file is deleted. This happens due to a rare case when the *gfid-handle* of the file is not deleted.

Workaround: On the bricks where the stale entries are present, for example, **`<gfid:5848899c-b6da-41d0-95f4-64ac85c87d3f>`**, check if the file's **`gfid`** handle is not deleted by running the following command and checking whether the file appears in the output, for example, **`<brick-path>/.glusterfs/58/48/5848899c-b6da-41d0-95f4-64ac85c87d3f`**.

```
# find <brick-path>/.glusterfs -type f -links 1
```

If the file appears in the output of this command, delete the file using the following command.

```
# rm <brick-path>/.glusterfs/58/48/5848899c-b6da-41d0-95f4-64ac85c87d3f
```

Issues related to Samba

BZ#1419633

CTDB fails to start on those setups where the real time schedulers have been disabled. One such example is where vdsm is installed.

Workaround: Enable real time schedulers by **`echo 950000 > /sys/fs/cgroup/cpu,cpuacct/system.slice/cpu.rt_runtime_us`** and then restart the ctdb service. For more information, refer the cgroup section of Red Hat Enterprise Linux administration guide, https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html-single/System_Administrators_Guide/index.html

BZ#1379444

Sharing of subdirectories of Gluster volume does not work if shadow_copy2 vfs module is also used. This is because shadow_copy2 checks on local filesystem for path being shared and Gluster volumes are remote filesystems accessed using libgfapi.

Workaround: Add **`shadow:mountpoint = /`** in share section of **`smb.conf`** to bypass this check.

BZ#1329718

Snapshot volumes are read-only. All snapshots are made available as directories inside .snaps. 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 create temp files (e.g. 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 asynchronus 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#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#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 can not 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

If a replica brick goes down and comes up when **`rm -rf`** command is executed, the operation may fail with the message *Directory not empty*.

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

BZ#1449638

The flexible I/O tester tool sends write calls of 1 Byte. For a sequential write, if a write call on a dispersed volume is not aligned to strip size, it first reads the whole stripe and then calculates the erasure code and then writes it back on the bricks. As a result, these Read calls have their own latency thus causing slow write performance.

Workaround: There is currently no known workaround for this issue.

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 ouput 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. 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.

4.2. RED HAT GLUSTER STORAGE CONSOLE

Red Hat Gluster Storage Console

BZ#1303566

When a user selects the auto-start option in the **Create Geo-replication Session** user interface, the **use_meta_volume** option is not set. This means that the geo-replication session is started without a metadata volume, which is not a recommended configuration.

Workaround: After session start, go to the geo-replication options tab for the master volume and set the ***use_meta_volume*** option to **true**.

BZ#1246047

If a logical network is attached to the interface with boot protocol DHCP, the IP address is not assigned to the interface on saving network configuration, if DHCP server responses are slow.

Workaround: Click **Refresh Capabilities** on the **Hosts** tab and the network details are refreshed and the IP address is correctly assigned to the interface.

BZ#1164662

The **Trends** tab in the Red Hat Gluster Storage Console appears to be empty after the ovirt engine restarts. This is due to the Red Hat Gluster Storage Console UI-plugin failing to load on the first instance of restarting the ovirt engine.

Workaround: Refresh (F5) the browser page to load the **Trends** tab.

BZ#1167305

The **Trends** tab on the Red Hat Gluster Storage Console does not display the thin-pool utilization graphs in addition to the brick utilization graphs. Currently, there is no mechanism for the UI plugin to detect if the volume is provisioned using the thin provisioning feature.

BZ#838329

When incorrect create request is sent through REST api, an error message is displayed which contains the internal package structure.

BZ#1042808

When remove-brick operation fails on a volume, the Red Hat Gluster Storage node does not allow any other operation on that volume.

Workaround: Perform *commit* or *stop* for the failed remove-brick task, before another task can be started on the volume.

BZ#1200248

The **Trends** tab on the Red Hat Gluster Storage Console does not display all the network interfaces available on a host. This limitation is because the Red Hat Gluster Storage Console **ui-plugin** does not have this information.

Workaround: The graphs associated with the hosts are available in the Nagios UI on the Red Hat Gluster Storage Console. You can view the graphs by clicking the **Nagios home** link.

BZ#1224724

The **Volume** tab loads before the dashboard plug-in is loaded. When the dashboard is set as the default tab, the volume sub-tab remains on top of dashboard tab.

Workaround: Switch to a different tab and the sub-tab is removed.

BZ#1225826

In Firefox-38.0-4.el6_6, check boxes and labels in **`Add brick`** and **`Remove Brick`** dialog boxes are misaligned.

BZ#1228179

`gluster volume set help-xml` does not list the **`config.transport`** option in the UI.

Workaround: Type the option name instead of selecting it from the drop-down list. Enter the desired value in the value field.

BZ#1231725

Red Hat Gluster Storage Console cannot detect bricks that are created manually using the CLI and mounted to a location other than **`/rhgs`**. Users must manually type the brick directory in the **`Add Bricks`** dialog box.

Workaround: Mount bricks in the **`/rhgs`** folder, which are detected automatically by Red Hat Gluster Storage Console.

BZ#1232275

Blivet provides only partial device details on any major disk failure. The Storage Devices tab does not show some storage devices if the partition table is corrupted.

Workaround: Clean the corrupted partition table using the **`dd`** command. All storage devices are then synced to the UI.

BZ#1234445

The task-id corresponding to the previously performed retain/stop remove-brick is preserved by engine. When a user queries for remove-brick status, it passes the bricks of both the previous remove-brick as well as the current bricks to the status command. The UI returns the error **`Could not fetch remove brick status of volume`**.

In Gluster, once a remove-brick has been stopped, the status can't be obtained.

BZ#1238540

When you create volume snapshots, time zone and time stamp details are appended to the snapshot name. The engine passes only the prefix for the snapshot name. If master and slave clusters of a geo-replication session are in different time zones (or sometimes even in the same time zone), the snapshot names of the master and slave are different. This causes a restore of a snapshot of the master volume to fail because the slave volume name does not match.

Workaround: Identify the respective snapshots for the master and slave volumes and restore them separately from the gluster CLI by pausing the geo-replication session.

BZ#1242128

Deleting a gluster volume does not remove the **`/etc/fstab`** entries for the bricks. A Red Hat Enterprise Linux 7 system may fail to boot if the mount fails for any entry in the **`/etc/fstab`** file. If the LVs corresponding to the bricks are deleted but not the respective entry in **`/etc/fstab`**, then the system may not boot.

Workaround:

1. Ensure that **/etc/fstab** entries are removed when the Logical Volumes are deleted from system.

2. If the system fails to boot, start it in emergency mode, use your root password, remount '/' in rw, edit fstab, save, and then reboot.

BZ#1167425

Labels do not show enough information for the Graphs shown on the **Trends** tab. When you select a host in the system tree and switch to the **Trends** tab, you will see two graphs for the mount point '/': one graph for the total space used and another for the inodes used on the disk.

Workaround:

1. The graph with y axis legend as %(Total: ** GiB/Tib) is the graph for total space used.

2. The graph with y axis legend as %(Total: *number*) is the graph for inode usage.

BZ#1134319

When run on versions higher than Firefox 17, the Red Hat Storage Console login page displays a browser incompatibility warning.

4.3. 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.

BZ#1482994

When creating a Virtual Machine (VM) using a template, shard translator returned an incorrect file size. With stat-prefetch enabled, the incorrect file size is cached and served as a part of lookups/stat etc. As a result, VMs that were created using these templates were unbootable.

Workaround: Disable the stat-prefetch translator and re-create the VM. This ensures that the newly created VMs using the templates are bootable.

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 the future releases.

NOTE

All Technology Preview features in Red Hat Enterprise Linux 6.7, 7.1, and 7.2 are also considered technology preview features in Red Hat Gluster Storage 3.2. For more information on the technology preview features available in Red Hat Enterprise Linux 6.7, see the Technology Previews chapter of the *Red Hat Enterprise Linux 6.7 Technical Notes*

5.1. STOP REMOVE BRICK OPERATION

You can stop a remove brick operation after you have opted to remove a brick through the Command Line Interface and Red Hat Gluster Storage Console. After executing a remove-brick operation, you can choose to stop the remove-brick operation by executing the **`remove-brick stop`** command. The files that are already migrated during remove-brick operation, will not be reverse migrated to the original brick.

For more information, see Stopping a **`remove-brick`** Operation in *Red Hat Gluster Storage 3.3 Administration Guide*.

5.2. SMB MULTI-CHANNEL

Multi-Channel is an SMB3 protocol feature that allows the client to bind multiple transport connections into one authenticated SMB session. This allows for increased fault tolerance and throughput on Windows 8 and newer and Windows Server 2012 and newer.

For more information, see SMB3 Multi-Channel with Samba on Red Hat Gluster Storage (Technology Preview).

5.3. READ-ONLY VOLUME

Red Hat Gluster Storage enables you to mount volumes with read-only permission. While mounting the client, you can mount a volume as read-only and also make the entire volume as read-only, which applies for all the clients using the **`volume set`** command.

5.4. PNFS

The Parallel Network File System (pNFS) is part of the NFS v4.1 protocol that allows compute clients to access storage devices directly and in parallel.

For more information, see pNFS in *Red Hat Gluster Storage 3.3 Administration Guide*.

APPENDIX A. REVISION HISTORY

Revision 3.3-1 **Thu Sept 14 2017** **Red Hat Gluster Storage Documentation Team**

Release Notes for the Red Hat Gluster Storage 3.3 release.