Red Hat Gluster Storage 3.5

3.5 Release Notes

Release Notes for Red Hat Gluster Storage 3.5

Edition 1

Last Updated: 2020-12-14

Red Hat Gluster Storage 3.5 3.5 Release Notes

Release Notes for Red Hat Gluster Storage 3.5
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 3.5.

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 can 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 3.5 release.

- Red Hat Gluster Storage has been updated to upstream glusterfs version 6.
- Samba has been updated to upstream version 4.9.8.
- NFS-Ganesha is updated to upstream version 2.7.3
- NFS version 4.1 is now supported.
- Directory contents are now read in configurable chunks so that very large directory listings can start to be served faster, instead of needing to wait for the whole directory to be read before serving to NFS-Ganesha clients.
- Red Hat Gluster Storage now provides the option of its own ctime attribute as an extended attribute across replicated sub-volumes, to avoid the consistency issues between replicated and distributed bricks that occurred when using file system based ctime, such as after self-healing occurred.
- Bricks in different subvolumes can now be different sizes, and Gluster's algorithms account for this when determining placement ranges for files. Available space algorithms are updated to work better for heterogeneous brick sizes. Same size bricks are still needed for bricks belonging to the same replica set or disperse set.
- The default maximum port number for bricks is now 60999 instead of 65535.
- Administrators can now prevent nodes with revoked certificates from accessing the cluster by adding a banned node's certificate to a Certificate Revocation List file, and specifying the file's path in the new ssl.crl-path volume option.
- Gluster-ansible can now configure IPv6 networking for Red Hat Gluster Storage.
- The storage.fips-mode-rchecksum volume option is now enabled by default for new volumes on clusters with an op-version of 70000 or higher.
- Configuring geo-replication was a lengthy and error prone process. Support for geo-replication is now provided by gdeploy, automating configuration and reducing error in this process.
- A new API, glfs_set_statedump_path, lets users configure the directory to store statedump output. For example, the following call sets the /tmp directory as the statedump path:

  ```
  glfs_set_statedump_path(fs2, "/tmp");
  ```

- New configuration options have been added to enable overriding of umask. The **storage.create-directory-mask** and **storage.create-mask** options restrict file mode to the given mask for directories and other files respectively. The **storage.force-directory-mode** and **storage.force-create-mode** options enforce the presence of the given mode bits for directories and other files respectively. Note that these mode constraints are maintained as long as the given option values are in effect and do not only apply to file creation.

- NFS-Ganesha now receives the upcall notifications needed to maintain active-active high availability configurations via asynchronous callbacks, which avoids the need for continuous polling and reduces CPU and memory usage.
- A new dbus command is available for obtaining access control lists and other export information from the NFS-Ganesha server.
- The storage.reserve option can now reserve available space in size in addition to reserving in percentage.
- Asynchronous I/O operations were impeded by a bottleneck in the workflow at the point of notification of successful completion. The bottleneck has been removed and asynchronous I/O operations now perform better.
- Performance improvements compared to Red Hat Gluster Storage 3.4

  NFS-Ganesha v4.1 mounts

  - 6-10 times improvement of **ls -l/stat/chmod** on small files on replica 3 and arbiter volumes.
  - Improvements for metadata-heavy operations that improve small file performance for dispersed volumes:
    - chmod - 106%
    - creates - 27%
    - stat - 808%
    - reads - 130%
    - appends - 52%
    - rename -36%
    - delete-renamed - 70%
    - mkdir - 52%
    - rmdir - 58%
  - Large file sequential write performance improvement of 47% for dispersed volumes
  - Large file sequential read/write improvements (20 and 60% respectively) for replica 3 volumes

  Gluster-FUSE mounts

  - Large file sequential read improvements (20%) for arbiter volumes
  - Small file mkdir/rmdir/rename improvements (24/25/10 % respectively) for dispersed volumes

2.2. SUPPORT LIMITS AND REMOVALS

Read this section to understand which technologies are no longer supported, or support reduced functionality, in Red Hat Gluster Storage 3.5.

gluster-swift (Object Store)

Using Red Hat Gluster Storage as an object store for Red Hat OpenStack Platform is no longer supported.

gstatus

Red Hat Gluster Storage 3.5 does not support gstatus.

Heketi for standalone Red Hat Gluster Storage

Heketi is supported only for use with OpenShift Container Storage. It is not supported for general use with Red Hat Gluster Storage.

Nagios Monitoring

Red Hat Gluster Storage 3.5 does not support monitoring using Nagios. Red Hat Gluster Storage Web Administration is now the recommended monitoring tool for Red Hat Storage Gluster clusters.

Red Hat Atomic Host

Red Hat Atomic Host is not supported for use with Red Hat Gluster Storage 3.5.

Red Hat Gluster Storage Console

Red Hat Gluster Storage Console is not supported for use with Red Hat Gluster Storage 3.5. Red Hat Gluster Storage Web Administration is now the recommended monitoring tool for Red Hat Storage Gluster clusters.

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

NOTE

This is the last Red Hat Gluster Storage release supporting Red Hat Enterprise Linux 6. Customers are adviced to upgrade to Red Hat Enterprise Linux 7 .

The following features are deprecated as of Red Hat Gluster Storage 3.5, or will be considered deprecated in subsequent releases. Review individual items for details about the likely removal time frame of the feature.

Gluster-NFS

Gluster-NFS is considered deprecated as of Red Hat Gluster Storage 3.5. Red Hat no longer recommends the use of Gluster-NFS, and does not support its use in new deployments on Red Hat Gluster Storage 3.5 and above. Existing deployments that upgrade to Red Hat Gluster Storage 3.5 remain supported, but users are encouraged to migrate to NFS-Ganesha, which provides enhanced functionality, additional security features, and performance improvements detailed in Section 2.1, “What's New in this Release?”.

Remote Direct Memory Access (RDMA)

Using RDMA as a transport protocol is considered deprecated in Red Hat Gluster Storage 3.5. Red Hat no longer recommends its use, and does not support new deployments on Red Hat Gluster Storage 3.5. Existing deployments that upgrade to Red Hat Gluster Storage 3.5 remain supported.

Tiering

Tiering is considered deprecated as of Red Hat Gluster Storage 3.5. Red Hat no longer recommends its use, and does not support tiering in new deployments on Red Hat Gluster Storage 3.5. Existing deployments that upgrade to Red Hat Gluster Storage 3.5 remain supported.

Red Hat Enterprise Linux 6 (RHEL 6) based ISO image for RHGS server

Starting with Red Hat Gluster Storage 3.5 release, Red Hat Enterprise Linux 6 based ISO image for Red Hat Gluster Storage server will no longer be included.

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.

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.

Security Fixes

CVE-2019-10197 (Moderate)

A combination of parameters and permissions could allow user to escape from the share path definition.

General Fixes

BZ#1578703

Previously, running **gluster volume status <volname> inode** output the entire inode table, which could time out and create performance issues. The output of this command is now more streamlined, and the original information should now be obtained by performing a statedump.

BZ#1734423, BZ#1736830, BZ#1737674

Previously, dynamically allocated memory was not freed correctly, which led to an increase in memory consumption and out-of-memory management on gluster clients. Memory is now freed correctly so that memory overruns do not occur.

BZ#1676468

Previously, glusterfs enabled kernel auto-invalidation, which invalidates page cache when ctime changes. This meant that whenever writes occurred before, during, and after a ctime change, the page cache was purged, and the performance of subsequent writes did not benefit from caching.

Two new options are now available to improve performance.

The mount option **auto-invalidation[=on|off]** is now enabled by default, and specifies whether the kernel can automatically invalidate attribute, dentry, and page cache. To retain page cache after writes, set this to 'off', but only if files cannot be accessed by two different mount points concurrently.

The volume option **performance.global-cache-invalidation=[on|off]** overrides the value of **performance.cache-invalidation**. This option is disabled by default, but when enabled purges all read caches related to gluster when a stat change is detected. Turn this option on only when a file can be accessed from different mount points and caches across these mount points are required to be coherent.

If both options are turned off, data written is retained in page cache and performance of overlapping reads in the same region improves.

BZ#1726991

Brick status was displayed as started when the brick was in a starting or stopping state because the **get-status** operation only tracked the started and stopped states. The **get-status** operation now reports state more accurately.

BZ#1720192

When a gluster volume has a ***bind-address*** specified, the name of the rebalance socket file becomes greater than the allowed character length, which prevents rebalance from starting. A hash is now generated based on the volume name and UUID, avoiding this issue.

BZ#1670415, BZ#1686255

A small memory leak that occurred when viewing the status of all volumes has been fixed.

BZ#1652461

If a user configured more than 1500 volumes in a 3 node cluster, and a node or glusterd service became unavailable, then during reconnection there was too much volume information to gather before the handshake process timed out. This issue is resolved by adding several optimizations to the volume information gathering process.

BZ#1058032

Previously, while migrating a virtual machine, libvirt changed ownership of the machine image if it detected that the image was on a shared file system. This prevented virtual machines from accessing the image. This issue can no longer be reproduced.

BZ#1685246

Access Control List settings were not being removed from Red Hat Gluster Storage volumes because the removexattr system call was not being passed on to the brick process. This has been corrected and attributes are now removed as expected.

Fixes for Dispersed Volumes

BZ#1732774

If a file on a bad brick was being healed while a write request for that file was being performed, the read that occurs during a write operation could still read the file from the bad brick. This could lead to corruption of data on good bricks. All reads are now done from good bricks only, avoiding this issue.

BZ#1706549

When bricks are down, files can still be modified using the **O_TRUNC** flag. When bricks function again, any operation that modified the file using file descriptor starts open-fd heal. Previously, when open-fd heal was performed on a file that was opened using **O_TRUNC**, a truncate operation was triggered on the file. Because the truncate operation usually happened as part of an operation that already took a lock, it did not take an explicit lock, which in this case led to a NULL lock structure, and eventually led to a crash when the NULL lock structure was de-referenced. The **O_TRUNC** flag is now ignored during an open-fd heal, and a truncate operation occurs during the data heal of a file, avoiding this issue.

BZ#1745107

Previously, when an update to a file's size or version failed, the file descriptor was not marked as bad. This meant that bricks were assumed to be good when this was not necessarily true and that the file could show incorrect data. This update ensures that the file descriptor is marked as bad with the change file sync or flush fails after an update failure.

Fixes for Distributed Volumes

BZ#1672869

Previously, when **parallel-readdir** was enabled, stale linkto files could not be deleted because they were incorrectly interpreted as data files. Stale linkto files are now correctly identified.

Fixes for Events

BZ#1732443

Previously, the network family was not set correctly during events socket initialization. This resulted in an invalid argument error and meant that events were not sent to consumers. Network family is now set correctly and events work as expected.

Fixes for automation with gdeploy

BZ#1759810

The configuration options **group=samba** and **user.cifs=enable** are now set on the volume during Samba setup via gdeploy, ensuring setup is successful.

BZ#1712904

Previously, when samba was configured using gdeploy, the samba user was not created on all nodes in a cluster. This caused problems during failover of CTDB, as the required user did not exist. gdeploy now creates the samba user on all nodes, avoiding this issue.

Fixes for Geo-replication

BZ#1708116

During geo-replication, when a sync was attempted for a large number of files that had been unlinked and no longer existed on master, the tarssh process hung because of a deadlock. When the stderr buffer of the tar process filled before tar completed, it hung. Workers expected tar to complete before reading stderr, but tar could not complete until the buffer was freed by being read. Workers now begin reading stderr output as soon as the tar process is created, avoiding the issue.

BZ#1708121

Geo-replication now synchronizes correctly instead of creating additional files when a large number of different files have been created and renamed to the same destination path.

BZ#1712591

In non-root geo-replication sessions, gluster binary paths were not added to PATH variable, which meant that gluster commands were not available to the session. Existing **gluster-command-dir** and **gluster-command-slave-dir** options can be used to ensure that sessions have access to gluster commands.

BZ#1670429

Geo-replication now succeeds when a symbolic link is renamed multiple times between syncs.

Fixes for NFS-Ganesha

BZ#1728588

A race condition existed where, when attempting to re-establish a connection with an NFS client, the server did not clean up existing state in time. This led to the new connection being incorrectly identified as having expired, rendering the mount point inaccessible. State is now cleaned before a new connection is accepted so this issue no longer occurs.

BZ#1751210

NFS-Ganesha used client credentials for all operations on Gluster storage. In cases where a non-root user was operating on a read-only file, this resulted in 'permission denied' errors. Root permissions are now used where appropriate so that non-root users are able to create and write to files using 0444 mode.

Fixes for Replication

BZ#1688395

When eager-lock lock acquisition failed during a write transaction, the previous lock was retained, which blocked all subsequent writes and caused a hang. This is now handled correctly and more specific log messages have been added to assist in diagnosing related issues.

BZ#1642425

The **cluster.quorum-count** volume option was not being updated in the volume configuration file for Gluster NFS volumes because when the last part of the file read is smaller than the buffer size, the data written from the buffer was a combination of new and old data. This has been corrected and Gluster NFS clients now honor **cluster.quorum-count** when **cluster.quorum-type** is set to **fixed**.

Fixes for Sharding

BZ#1568758

Deleting a file with a large number of shards timed out because unlink operations occurred on all shards in parallel, which led to contention on the **.shard** directory. Timeouts resulted in failed deletions and stale shards remaining in the **.shard** directory. Shard deletion is now a background process that deletes one batch of shards at a time, to control contention on the **.shard** directory and prevent timeouts. The size of shard deletion batches is controlled with the **features.shard-deletion-rate** option, which is set to **100** by default.

Fixes for Web Administration

BZ#1645428

The previously shipped version of the python2-pyasn1 package caused IPA client installation to fail. This package is replaced with updates to tendrl-notifier and tendrl-commons so that pysnmp is used instead of python2-pyasn1, and installation works as expected.

Before upgrading to Red Hat Gluster Storage Web Administration 3.5, remove the python2-pyasn1 and pysnmp packages (but not their dependencies) by running the following commands:

```
# rpm -e --nodeps $(rpm -qa 'python2-pyasn1')
# rpm -e --nodeps $(rpm -qa 'pysnmp')
```

BZ#1647322

Previously, tendrl did not set an owner for the **/var/lib/carbon/whisper/tendrl** directory. When the owner of this directory was not the **carbon** user, carbon-cache could not create whisper files in this location. Tendrl now ensures the directory is owned by the **carbon** user to ensure whisper files can be created.

BZ#1688630

Previously, errors that occurred because **tendrl-monitoring-integration** was not running were reported with generic error messages. More specific error messages about **tendrl-monitoring-integration** status is now logged in this situation.

BZ#1645221

Previously, Red Hat Gluster Storage web administration expected all nodes to be online before any node could stop being managed by web administration. It is now possible to remove a node from being managed even when one or more nodes in the cluster are not online.

BZ#1666386

Red Hat Gluster Storage web administration previously received all split brain related events and displayed these as errors in the user interface, even when they were part of correctly operating heal processes. Events are now filtered based on the client identifier to remove unnecessary and erroneous errors from the user interface.

BZ#1687333

Previously, when all nodes in a cluster were offline, the web administration interface did not report the correct number of nodes offline. Node status is now correctly tracked and reported.

BZ#1686888

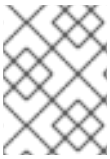
The node-agent service is responsible for import and remove (stop managing) operations. These operations timed out with a generic log message when the node-agent service was not running. This issue is now logged more clearly when it occurs.

BZ#1702412

Previously, Ansible 2.8 compatibility did not work correctly. Red Hat Storage Web Administration is now compatible with Ansible 2.8.

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#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.

NOTE

With the release of 3.5 Batch Update 3, the mount point of shared storage is changed from /var/run/gluster/ to /run/gluster/ .

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 Dispersed Volumes

BZ#1766640

Special handling is sometimes required to ensure I/O on clients with older versions works correctly during an in-service upgrade. Servers with dispersed volumes do not do this handling for Red Hat Gluster Storage 3.3.1 clients when upgrading to version 3.5.

Workaround: If you use dispersed volumes and have clients on Red Hat Gluster Storage 3.3.1, perform an offline upgrade when moving server and client to version 3.5.

BZ#1735666

In some cases, both kernel read-ahead and gluster read-ahead mechanisms cause a large increase in the amount of data read during sequential read operations. In these cases, performance for sequential reads, especially on dispersed volumes, is significantly worse.

If you are affected by this issue, you can work around the problem by manually disabling gluster read-ahead and io-cache behavior using the following commands.

```
# gluster volume set <volname> read-ahead off
# gluster volume set <volname> io-cache off
```

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.

BZ#1760713

When the **cluster.lookup-optimize** option is enabled, running a rename operation after a rebalance can lead to some files being inaccessible from the client.

Workaround: Disable **cluster.lookup-optimize** on the volume.

```
# gluster volume set VOLNAME cluster.lookup-optimize off
```

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.

BZ#1721355

Both writing to volumes and reading information for the **gluster heal info** command require blocking locks. This means that when a volume is under heavy write load, running **gluster heal info** is blocked by existing locks.

Workaround: Reduce the number of write operations from clients to assist in releasing locks and completing the **gluster heal info** operation.

Issues related to Gluster NFS

Older NFS over UDP issues

- 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 Gluster-NFS server.
- **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.

Older locking issues

- **fcntl** locking (NFS Lock Manager) does not work over IPv6.
- 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.
- 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.

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#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-Ganesha

Unlocks may fail after restart

After you restart the NFS server, the unlock within the grace-period feature may fail and the locks held previously may not be reclaimed.

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#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>)"""
```

NOTE

With the release of 3.5 Batch Update 3, the mount point of shared storage is changed from /var/run/gluster/ to /run/gluster/ .

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 behavior 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 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#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: 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#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.

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#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#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#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

Statedump reports success even when it fails because of incorrect client host value, for example:

```
# gluster volume statedump volume client not-a-host:port
```

Workaround: Ensure the value of *host* is correct.

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 7.7 are also considered technology preview features in Red Hat Gluster Storage 3.5. 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 7.7, 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 3.5-0	**Wed Oct 30 2019**	**Red Hat Gluster Storage Documentation Team**

Created and populated release notes for Red Hat Gluster Storage 3.5