Red Hat OpenShift Container Storage 3.10

3.10 Release Notes

Release Notes for Red Hat OpenShift Container Storage

Edition 1

Last Updated: 2018-09-12

Red Hat OpenShift Container Storage 3.10 3.10 Release Notes

Release Notes for Red Hat OpenShift Container Storage
Edition 1

Bhavana Mohan
Customer Content Services Red Hat
bmohanra@redhat.com

Legal Notice

Abstract

These release notes provide high-level coverage of the improvements and additions that have been implemented in Red Hat OpenShift Container Storage 3.10.

Table of Contents

CHAPTER 1. WHAT'S NEW IN THIS RELEASE?

This section describes the key features and enhancements in the Red Hat Openshift Container Storage 3.10 release.

- **Arbiter volume support enabling availability with efficient storage utilization and better performance**: Arbiter volume supports all persistent volume types with better consistency and less disk space requirements. The arbiter uses client quorum to compare this metadata with the metadata of the other nodes to ensure consistency in the volume and prevent split-brain conditions.
  - Information about setting up arbiter volumes is available in https://access.redhat.com/documentation/en-us/red_hat_openshift_container_storage/3.10/html-single/deployment_guide/#chap-creating_arbiter
  - Information about managing arbiter volumes is available in https://access.redhat.com/documentation/en-us/red_hat_openshift_container_storage/3.10/html-single/deployment_guide/#Managing_Arbiter_Volumes
- **Rebranding**: With the release of OpenShift Container Platform 3.10, we are rebranding Container Native Storage to Red Hat OpenShift Container Storage. Along with this rebranding, the terms describing the method of deployment have also been changed to:
  - Converged mode (formerly referred to as ‘Container Native Storage’ or ‘CNS’) refers to running storage services in a container itself, like any other service.
  - Independent mode (formerly referred to as ‘Container Ready Storage’ or ‘CRS’) refers to running storage services outside the container platform as a standalone subsystem which gets consumed by OCP.
- **Ansible as the primary mode of installation**: Ansible is now the preferred mode of installation and end-to-end ansible workflows are integrated into the Red Hat Openshift Container Storage Deployment Guide. Ansible automates the process of installing Red Hat Openshift Container Storage in both converged and independent mode, and this makes it less error prone when compared to manually executing numerous commands:
  - To install Red Hat Openshift Container Storage in converged mode, see https://access.redhat.com/documentation/en-us/red_hat_openshift_container_storage/3.10/html-single/deployment_guide/#chap-Documentation-Deploy-CNS
  - To install Red Hat Openshift Container Storage in independent mode, see https://access.redhat.com/documentation/en-us/red_hat_openshift_container_storage/3.10/html-single/deployment_guide/#chap-Documentation-Deploy-CRS
- **Separate Deployment Guide and Operations Guide**: Enhancements and features available in the Red Hat Openshift Container Storage are now available in two separate guides - one for Deployment and upgrade and another for administrative tasks. With Red Hat Openshift Container Storage 3.10 we have the following two guides:
  - Deployment Guide: https://access.redhat.com/documentation/en-us/red_hat_openshift_container_storage/3.10/html-single/deployment_guide/

   - Operations Guide: https://access.redhat.com/documentation/en-us/red_hat_openshift_container_storage/3.10/html-single/operations_guide/

- **Enhanced OCS monitoring and configuration visibility using the Prometheus framework**: The statistics of gluster block PVs can be obtained by setting up Prometheus and enabling volume metrics for glusterfs plugin. The different metrics of the PVs that can be viewed on Prometheus are:

   - **`kubelet_volume_stats_available_bytes`**: Number of available bytes in the volume.
   - **`kubelet_volume_stats_capacity_bytes`**: Capacity in bytes of the volume.
   - **`kubelet_volume_stats_inodes`**: Maximum number of inodes in the volume.
   - **`kubelet_volume_stats_inodes_free`**: Number of free inodes in the volume.
   - **`kubelet_volume_stats_inodes_used`**: Number of used inodes in the volume.
   - **`kubelet_volume_stats_used_bytes`**: Number of used bytes in the volume.

  For more information about Volume metrics, see https://access.redhat.com/documentation/en-us/red_hat_openshift_container_storage/3.10/html-single/operations_guide/#enable_vol_metrics.

- **Reclaiming persistent volume**: You can now set the persistentVolumeReclaimPolicy in the storage class. When this parameter is set to "Retain" the underlying persistent volume is retained even after the corresponding persistent volume claim is deleted. This ensures you do not lose your data if you accidentally delete the PVCs.

   - For more information about setting the persistentVolumeReclaimPolicy in the storage class for file storage, seehttps://access.redhat.com/documentation/en-us/red_hat_openshift_container_storage/3.10/html-single/operations_guide/#chap-Documentation-Red_Hat_Gluster_Storage_Container_Native_with_OpenShift_Platform-OpenShift_Creating_Persistent_Volumes-Dynamic_Prov
   - For more information about setting the persistentVolumeReclaimPolicy in the storage class for block storage, see https://access.redhat.com/documentation/en-us/red_hat_openshift_container_storage/3.10/html-single/operations_guide/#Block_dynamic_prov_vol

- **Replace a block on block storage**: With the release of Red Hat OpenShift Container Storage 3.10, you can replace a block from a node that is out of resource or is faulty, to a new node. For more information see, https://access.redhat.com/documentation/en-us/red_hat_openshift_container_storage/3.10/html-single/operations_guide/#replace_block

- **Ansible-based uninstall (cleanup) of all resources and artifacts from the cluster:** By using ansible based uninstall, any existing GlusterFS-related resources and configurations from a Red Hat OpenShift Container Storage deployment is removed. For more information about Uninstall Red Hat Openshift Container Storage see, https://access.redhat.com/documentation/en-us/red_hat_openshift_container_storage/3.10/html-single/deployment_guide/#chap-Documentation-Red_Hat_Gluster_Storage_Container_Native_with_OpenShift_Platform-Uninstall_CNS

CHAPTER 2. NOTABLE BUG FIXES

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

heketi

BZ#1613260

Previously, the heketi startup script for the container image used a blocking file lock with a time out. This could cause issues when the heketi db is stored on gluster volumes. With this update, the startup script now polls for db availability with a non-blocking file lock.

BZ#1541323

Previously, some heketi client requests failed with ‘Token used before issued’ error because time synchronization was not properly handled by JSON web tokens. This update adds a margin of 120 seconds to iat claim validation to ensure that client requests can succeed in this situation. This margin can be changed by editing the ‘HEKETI_JWT_IAT_LEEWAY_SECONDS’ environment variable.

BZ#1572561

Previously, the arbiter brick was created after data bricks and could fail to be placed if the device available for the arbiter brick had already got a data brick. The fix ensures arbiter brick placement is prioritised, to increase the likelihood that it is placed on the most appropriate device.

BZ#1601436

Previously, if heketi got a request to delete a volume it recorded a pending operation in the db prior to making changes on the underlying storage nodes. When multiple requests to delete the same volume were received before the first request could be completed, Heketi would record multiple pending delete operations. Hence, after the first operation was successful it would remove the volume from the db in such a way that the extra volume delete operations are never completed and the db contained dangling references to items which did not exist. With this update, Heketi now rejects delete requests for volumes that are already in pending deletion. The database structures that track volumes pending deletion remains consistent and are cleaned up correctly.

BZ#1607821

Heketi can automatically create a new block hosting volume even when there was insufficient disk space on existing block hosting volume. Previously, if more than one request for block volumes were received concurrently Heketi might create more block hosting volumes than necessary. With this update, Heketi no longer allows clients to automatically trigger the creation of extra block hosting volumes when another request is already creating a block hosting volume.

BZ#1432340

Previously, a misconfigured router or proxy in Kubernetes or Openshift could fail to provide error messages in the HTTP response body, leading heketi-cli to display nothing for these error conditions. With this update, heketi-cli will display the HTTP status code, rather than nothing, for these error conditions.

BZ#1550919

With this update, heketi offers a metrics REST endpoint from which information about heketi's topology can be retrieved in a text based format that can be consumed by Prometheus.

BZ#1561680

Previously, when PVCs were created/deleted in environments that are managed by Heketi, the LVM-metadata backups resulted in large storage consumption under /etc/lvm/archive which resulted in the filesystem running out of space. With this fix, Heketi no longer creates backups of LVM-metadata by default. This configuration can be changed by editing the `backup_lvm_metadata: true` variable in the heketi.json configuration file.

BZ#1428654

Previously, heketi attempted to allocate a volume by iterating over all available clusters to determine where a volume request would fit. If none of the clusters were suitable Heketi returned a generic "No space" error. This generic error hid useful debugging information from the user. With this fix, Heketi gathers all the error conditions from the clusters into a single error response. If more than one cluster is configured the error for each cluster is prefixed with the cluster's ID.

BZ#1459233

Previously, Heketi would attempt creation of bricks on devices with insufficient space because of an inconsistency in the free space accounting on the device in Heketi's database. This fix corrects the accounting logic during error handling to prevent the inconsistency.

BZ#1459628

With this update, Heketi can now create and manage arbitrated volumes. Arbitrated volumes can reduce overall space consumption in the cluster's storage and reduce total transferred data by creating specialized arbiter bricks.

BZ#1534850

Previously, if a block volume of size greater than default block hosting volume size was requested, the request failed with "Error: Default Block Hosting Volume size is less than block volume requested". With this fix, block volumes of size greater than default block hosting volume size are created if sufficient space is available and no error message is displayed.

BZ#1554796

Previously, some heketi client requests failed with ‘Token used before issued’ error because time synchronization was not properly handled by JSON web tokens. With this fix This update adds a margin of 120 seconds to iat claim validation to ensure that client requests can succeed in this situation. This margin can be changed by editing the ‘HEKETI_JWT_IAT_LEEWAY_SECONDS’ environment variable.

BZ#1566398

Previously, a rebalance operation was not run after expanding a volume in Heketi. This meant that only new files could be written to newly added space. With this fix, Heketi performs a rebalance operation on the volume after expansion.

BZ#1548157

Previously, Heketi could not delete volumes that were fully or partially removed from the underlying storage system. Running Heketi commands such as heketi-cli volume delete <ID> on a volume that was deleted from gluster failed due to synchronization issues. With this fix, Heketi volume delete functionality can incrementally delete a volume, skipping over items that have already been deleted.

BZ#1568251

Previously, Heketi configuration files that lacked a specific log level which caused the application to display a meaningless log message on startup. With this fix, Heketi ignores the lack of an explicit log level in the configuration file.

BZ#1569306

Previously, a rebalance operation was not run after expanding a volume in heketi, meaning only new files could be written to newly added space. With this fix, heketi performs a rebalance operation on the volume after expansion.

BZ#1578383

Heketi’s algorithm for creating a brick to be used by the arbiter feature creates the brick with a size smaller than the size of the data bricks, using an expected average file size factor that can be supplied by the user. Previously, supplying a large value for this file size factor created bricks too small to hold a file system. With this fix, Heketi sets a minimum brick size so that a valid file system can be created.

BZ#1581864

Previously, some heketi client requests failed with ‘Token used before issued’ error because JSON web tokens did not properly handle clock skew. With this fix, this update adds a margin of 120 seconds to iat claim validation to ensure that client requests can succeed in this situation. This margin can be changed by editing the ‘HEKETI_JWT_IAT_LEEWAY_SECONDS’ environment variable.

BZ#1584123

Previously, If creating block volumes failed Heketi incorrectly added the size of the block volume to the free space counter of the block hosting volume even if the space had not been deducted. Failure to create a block hosting volume would lead to incorrect free space values. With this fix, the free space accounting in Heketi is corrected and Heketi no longer generates incorrect free space values.

BZ#1584191

Previously, if volume create operation failed, space allocated on the device for the bricks was not released in heketi database during clean up operation. With this fix, space allocated for the bricks is released.

BZ#1590143

Previously, Heketi executed LVM management operations on the underlying storage nodes such that the commands might prompt for input. Commands that triggered input prompts would never complete without administrative intervention. With this fix Heketi is changed to run the LVM operations for volume delete and will not prompt for input. Heketi operations that manage LVM storage will either succeed or fail .

BZ#1591437

Previously, Heketi could not delete volumes that were fully or partially removed from the underlying storage system. Running heketi commands such as 'heketi-cli volume delete <ID>' on a volume that was deleted from gluster failed. With this fix, Heketi volume delete functionality can incrementally delete a volume, skipping over items that have already been deleted.

BZ#1591624

Earlier, all block hosting volumes were created only with group gluster-block option set on the Gluster volume. With this update, this feature enables an administrator to define options while creating block hosting volumes by setting block_hosting_volume_options in heketi.json OR by setting environment variable HEKETI_BLOCK_HOSTING_VOLUME_OPTIONS. However, Red Hat recommends not removing the group gluster-block option, but additional options can be added to this option. For example, to make block hosting volumes be of arbiter type, set the option to "group gluster-block,user.heketi.arbiter true".

BZ#1592649

Previously, If the block hosting volume was expanded free space attribute of the volume was not updated leading to block volume failures even though space was available.With this fix, free size in block hosting volume information is updated.

BZ#1595531

Previously, the same set of nodes were selected by Heketi as block volume targets when distributing block volumes across all available nodes. With this fix, Heketi shuffles the list of nodes such that different block volumes are distributed across different nodes.

BZ#1595535

Previously, because of an off by one error, Heketi added one host more than ha count to the list of hosts provided to gluster-block cli during block volume create. This lead to the incorrect usage of one extra node. With this fix, the number of hosts used for iscsi targets is equal to ha count parameter.

BZ#1596018

Previously, the logic in Heketi incorrectly assumed block volumes always fit within block hosting volumes if the size of the block volume was smaller than the total size of the hosting volume minus the sizes of the existing block volumes. This logic failed to account for the fact that the file system and the block volumes contained within it require space to manage metadata. Trying to place such a volume failed with an out of space error instead of allocating a new block hosting volume. With this fix, Heketi reserves a small percentage of the total size of the block hosting volume for metadata.

BZ#1596022

Previously, when Heketi received a request to create a new volume or block volume it recorded a pending operation in db prior to making changes on underlying storage nodes. These pending items were normally filtered out of command output such as "heketi-cli volume list" or "heketi-cli topology info". However, when a block hosting volume implicitly created by a new block volume was in the heketi db in pending state commands such as "heketi-cli topology info" it would fail with an "Id not found" error. With this fix, Heketi's filtering logic was corrected and Heketi no longer produces an "Id not found" error when pending items exist in the heketi db.

BZ#1596626

Previously, when the heketi database contained entries with broken references, various operations failed with the error "Id not found". With this fix, broken references are ignored when deleting a block hosting volume, cleaning up bricks with empty paths, and starting the heketi service when removing said reference would not lead to any additional broken references.

BZ#1600160

Previously, when under load due to many concurrent requests, Heketi consumed high amounts of memory and network connections and contended with itself for resources, leading to unpredictable behavior when provisioning new volumes. With this fix, Heketi now throttles client requests by considering the number of in-flight operations it is working on and telling clients to retry their requests later when that number reaches a threshold. The Heketi command line client and OpenShift/Kubernetes provisioner automatically retry these requests later.

BZ#1601907

Previously, after expansion of block hosting volume, the free space attribute of the volume wasn't updated, which lead to block volume failures even though space was available. With this fix, free size in block hosting volume information is also updated.

BZ#1612058

Heketi passed incorrect options to the XFS format command when creating bricks. This meant that inode tables on arbiter bricks were being sized as though they were for data bricks, and therefore reserved a large number of inodes for data that would never be written to the arbiter brick. Heketi now passes the correct options and the inode tables of arbiter bricks are sized to ensure they can hold metadata for the rest of the volume or volume set.

BZ#1612782

Previously, if the first block volume creation failed the block hosting volume was deleted but free space amount was not updated. This fix updates the free size if block volume creation fails.

BZ#1612784

Previously, volume mount options were not updated to remove node references after removing a node from the heketi configuration. With this fix, references to remove nodes are removed from volume mount options and backup-volfile-servers options.

BZ#1612786

Previously, if a node was removed from heketi management, that node was still considered for block volume creations because the node's status was not updated in the hosts of already created block volume. With this fix, once the node is deleted from heketi management, the hosts in all created block volumes are updated. This ensures only hosts present in heketi management are considered for block volume creation.

BZ#1619017

Previously, the storage space used by the block volume was not deducted from the block hosting volume until the block volume was created. If two block volumes were requested at the same time then Heketi would allow both volume requests to pass to the underlying storage system. This lead to unnecessary out of space errors. With this fix, Heketi reserves the space needed for the block volume prior to requesting the block volume from the underlying storage system. Heketi no longer requests for more than one block volume if the free space on the block hosting volume can contain only one volume.

BZ#1622645

Previously, heketi would allow a block-hosting volume space to be filled completely. Hence, restarting the application pods using block volumes of a completely filled block-hosting volume would fail remounting the block-volume with an I/O error. With this update, heketi reserves 2% of the block hosting volume's raw space and thereby preventing the space from being filled completely and thus remounts are now successful.

CNS-deployment

BZ#1598751

Till this release, the glusterd pod's health or liveness check was performed based entirely on the glusterd service status. However, there were scenarios where the metadata directory of glusterd ran out of space which caused glusterd to unsuccessfully complete further transactions without failing the service. With this update, an extra liveness probe has been added which ensures glusterd is down if the metadata directory is full or out of space.

rhgs-server-container

BZ#1614251

Previously, installing Red Hat OpenShift Container Storage changed the secontext for /dev/log from devlog_t to device_t. This meant that OpenShift logging and metrics could not use the relabelled /dev/log. With this fix, the services responsible for relabelling /dev/log are removed, and /dev/log files label does not change.

BZ#1576764

Previously, NTP service ran in both rhgs-server-container pod and the node where rhgs-server-container pod was running. With this fix, NTP service is enabled only in the node and the instance that ran in rhgs-server-container pod is removed.

gluster-block

BZ#1537170

Previously, when targets were not loaded due to block hosting volume being down or not ready or target_core_user module not loaded on host or targets loading failed, the preceding gluster-block operations calling global target saveconfig would override the non-loaded target's configuration. This resulted in loss of previously non-loaded targets configuration. With this update, the issue is addressed by introducing a target/block-volume level save configuration option.

BZ#1598322

Previously, the gluster-block daemon depended on glusterd to be running and did not verify if the block hosting volumes were online and ready to be consumed before beginning its operations. This resulted in failures when the gluster-block daemon attempted to load target configuration. With this update, gluster-block daemon now waits for block hosting volume's bricks to be available before attempting to load target configuration.

BZ#1507767

With this update, you can now replace block volume path from one node to another, in case if the existing node is running out of resources, permanently taken down or is in maintenance.

BZ#1575529

Currently, multipathing priority configuration is set to constant with all the HA paths. Hence it is possible that the load might not be distributed uniformly across the available HA nodes. With this update, priority based load balancing at gluster-block is introduced. The management daemon gluster-block reads the load balance information from the metadata and selects a high priority path from HA based on the data. When the block device is requested for creation, high priority is set on a path whose node is least used. While logging to the device, the initiator side multipath tools picks the high priority path and marks it as active. This way it is possible to distribute the balance across the nodes.

BZ#1593124

Previously, the close operation of gluster volume entity was missing. This caused 'df -Th' to show an incorrect used size for the brick of the block hosting volume in container. With this fix, tcmu-runner closes the gluster volume entity when removing the target device and ‘df-th’ shows correct used size of brick of block hosting volume.

BZ#1622681

Previously, if the block volume was attempted for delete when block hosting volume was 100% consumed, the delete operation failed. Because the writes to metadata file residing in the same block hosting volume failed with ENOSPC and hence the overall delete operation failed. Now, with this fix at every new block volume create request, the gluster-block checks for minimum free space equal to 'requested block volume size' + 10M which is reserved for block metadata on the block hosting volume, this way block volume delete will succeed even when block hosting volume is almost full.

CNS-deployment

BZ#1484412

Previously, if any glusterd to glusterd connection reconnected during a volume delete operation it could have led to a stale volume being present in the gluster pool. This stale volume led to the 'brick part of another brick' error during subsequent volume create operations. With this fix, subsequent volume create operations don't fail.

kubernetes

BZ#1501439

Even though glusterfs fuse client log level has been set to ERROR, due to an issue in glusterfs client, it logs Info|Warn|Error messages regardless of the log level value set in the client process. This issue has been fixed with latest glusterfs client release.

CHAPTER 3. KNOWN ISSUES

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

- BZ#1623433

  During node reboot, the corresponding rhgs-server pod gets restarted. When the pod boots up, the brick is not mounted as the corresponding lv device is not available.

  To workaround this issue, mount the brick(s) using the following command:

  ```
  # mount -a --fstab /var/lib/heketi/fstab
  ```

  Start the corresponding volume using the following command:

  ```
  # gluster volume start <volume name> force
  ```

- BZ#1597320

  Few paths are missing for iscsi mpath device. This is because, either CHAP security values mismatch or iscsi CSG: stage 0 has been skipped. Both will result in iSCSI login negotiation to fail.

  To workaround this issue, delete the app pod which will trigger the restart of the pod. The pod start will then update the credentials and relogins.

- BZ#1596021

  Sometimes in OCP, instead of using the multipath mapper device, such as /dev/mapper/mpatha, to mount the device, individual paths, such as /dev/sdb, is used. This is because OCP is currently not waiting for the mpath checker default timeout of 30 seconds, but waits for only 10 second and then picks the individual path if the mapper device is not ready. Due to this, we are not benefited by all the advantages of high availability and multipathing for block device.

  To workaround this issue, delete the app pod which will then trigger the restart of the pod. The pod then relogins and undergoes a multipath check.

- BZ#1476730

  Gluster-block operations (create/delete/modify) or gluster-block-target service restart, performed when tcmu-runner is in offline state, can trigger netlink hung issue, with targetcli process entering uninterruptible sleep (D state) state forever.

  To workaround this issue, reboot the node to recover from this state.

- BZ#1409848

  The following two lines might be repeatedly logged in the rhgs-server-docker container/gluster container logs.

  ```
  [MSGID: 106006] [glusterd-svc-
  mgmt.c:323:glusterd_svc_common_rpc_notify] 0-management: nfs has
  disconnected from glusterd.
  [socket.c:701:__socket_rwv] 0-nfs: readv on
  /var/run/gluster/1ab7d02f7e575c09b793c68ec2a478a5.socket failed
  (Invalid argument)
  ```

- These logs are added as glusterd is unable to start the NFS service. There is no functional impact as NFS export is not supported in Containerized Red Hat Gluster Storage.

APPENDIX A. REVISION HISTORY

Revision 1.0-02 **Wed Sep 12 2018** **Bhavana Mohan**

Publishing for Red Hat Openshift Container Storage 3.10

Revision 1.0-01 **Tue Sep 11 2018** **Bhavana Mohan**

Included information about what is new for OCS 3.10.

Documented all the major bugs fixed for this release.

Documented the known issues.

Revision 1.0-1 **Mon 23 2018** **Anjana Sriram**

Created for 3.10 release

INDEX