Feature Design Document

Trash translator for GlutserFS

Anoop C S
achiraya@redhat.com

Jiffin Tony Thottan
jthottan@redhat.com

Feature summary

Trash translator will provide temporary storage for deleted files from a GlusterFS volume. This translator will help users access and recover accidentally deleted data. Policies like eliminate patterns and purging of the files in trash directory will also be supported. This can also be used internally by volume operations like self-healing and rebalancing which perform delete and truncate operations on crucial data.

Effort driven by

Current status

There is no support for trash in GlusterFS as of upstream version 3.5.

Table of Contents

Introduction

Trash translator will allows users to access deleted or truncated files. Every brick will maintain a hidden .trash directory, which will be used to store the files deleted or truncated from the respective brick. The translator can also be further enhanced to support recovery of the deleted or truncated files, with the help of new clis.

The trash directory location should itself be configurable. Other policies like purging of files, and alert logs can also be implemented. The trash translator can also be used internally for operations like self-heal and rebalance of volumes.

Trash Translator

As of now the trash translator has some compatibility issues with the existing GlusterFS code base. The trash translator is designed to intercept unlink, truncate and ftruncate fops, store a copy of the current file in the trash directory, and then perform the fop on the original file.

During an unlink operation, the trash translator will intercept the unlink call, and check if it falls under the eliminate pattern. If the file on which the fop is being performed, falls under the eliminate pattern, the translator, will wind down to perform an unlink call, thereby removing the file, without saving it in the .trash directory.

However, if it doesn't fall under the eliminate pattern, a stat is performed on the file. After a successful stat operation, the trash translator will create the same path inside the .trash directory, as is the path of the file, if the path doesn't already exist. At this point, the translator will simply rename the file from its original path to the one under trash, thereby saving the original file, and also removing the file from its original location.

Truncate and ftruncate operations shrink or extend the size of a file to a specified size. The only difference is that in truncate we provide path name of the file whereas in ftruncate we use the file descriptor for truncation. Hence ftruncate can handle files which are already opened.

When the trash translator intercepts a truncate/ftruncate call, a new file will be created in the trash, and the contents of the original file will be copied into this new file, using readv and writev system calls. Once all the contents are copied, the trash translator will wind down to perform a truncate call on the original file.

All the files saved in the trash directory are time-stamped, so as to maintain versioning, in case of same files being truncated/deleted again and again.

CLI Design

Additional CLI ops to be supported:
< > : Required arguments
[ ] : Optional arguments

- CLI interface for switching trash on/off:
  gluster volume set <VOLNAME> feature.trash <on | off>

  This command can be used to enable trash translator in a volume. By default the trash translator will not be loaded in the volume graph. The user can use this command to load the trash translator in the volume graph. Once it is loaded, and the volume is started, the trash-directory will be created in every brick of the volume.

- CLI interface for setting trash directory path:
  gluster volume set <VOLNAME> feature.tarsh-dir <name>

  This command is used to set the trash directory to a specified name. The argument is a valid directory name. Directory will be created inside every brick under this name. If not specified by the user, the trash translator will create the trash directory with the default name ".trash". If a user tries to re-configure the trash directory, while data is still present the volume set operation will fail, prompting the user to delete existing data in the trash first.

- CLI interface for setting maximum trash file size:
  gluster volume set <VOLNAME> feature.trash-max-fs <size>

  This command can be used to filter files entering trash directory based on their size. Files above trash_max_fs are deleted/truncated directly. If not specified the default maximum trashable file size will be set.

- CLI interface for specifying eliminate pattern:
  gluster volume set <VOLNAME> feature.trash-elm-pat <path1>|<paht2>|...

  This command can be used to set the eliminate pattern for the trash translator. It can take multiple path patterns separated with '|'. Files residing under these patterns will not be moved to trash directory during deletion/truncation. Paths must be an absolute path present in volume .

- CLI interface for listing trash files:
  gluster volume trash <VOLNAME> list

  This command can be used to display the files that comes under trash. Files will be listed serially with their path.

- CLI interface for purging files:
  gluster volume trash <VOLNAME> purge <all | path | position> . . . [force]

  This command can be used to permanently delete file/files from trash and can accept multiple arguments. Files must be mentioned by their path or position in the list. If all is specified entire trash content will be purged. Force option allows users to by-pass any failures in purging, and proceed with deletion of the files from trash.

- CLI interface for restoring files:
  gluster volume trash <VOLNAME> restore <all | path | position> . . . [force]

  This command can be used to recover file/files from trash and can accept multiple arguments. Files must be mentioned by their path or position in the list. If all is specified entire trash content will be restored. Force option allows users to by-pass any failures in restoring, and proceed with restoring of the files from trash.

Restore

Mere storing of deleted or truncated files may not exploit the complete use of a trash translator. Data residing in the hidden trash directory must be made available to the users so as to restore those in future. Each brick will have a configurable trash directory, which will store the deleted or truncated data from that brick. The deleted data stored in the trash directory will be available to the user in the backend for recovery.

Along with that a gluster cli command can be provided to list all the contents of the trash directories. The command can be further explored to display stats like size of the trash, size used, size available etc. Users can restore individual files with the help of restore option specifying the corresponding position in the list. In this case trash translator will automatically move the file to its original location which is stored as an extended attribute.

As a stretch goal, a common consolidated view of the all the trash directories can also be made available for the user. This view will be available from gluster mount itself, and will display all the files in all the .trash directories of the bricks. The user will have the option to directly look into the trash directory and restore files manually.

Use cases

The primary intended use-case for trash translator is to store accidentally or intentionally deleted files. Since there exists no methods to retrieve those files, trash translator can be used to retain the same.

The trash trasnlator will also allow the user to restore the delete data. Also gluster operations like self-heal and rebalance, which perform operations on cucial user-data, can leverage the trash translator.

Policies

A set of policies can be defined to regulate the behaviour of the trash translator

- During an unlink operation following files are excluded from moving to trash directory,
  - An eliminate pattern can be defined. Files matching the pattern will not be moved to trash.
  - Files already inside trash.
  - Files having size greater than maximum trash file size.
  - Directories

- During truncate operation following files are excluded from moving to trash directory,
  - Files matching the eliminate pattern.
  - Files already inside trash.

- The trash directory per brick should be limited by a configurable size limit. As the trash exceeds this limit, an alert log can be generated warning the user about the same.

- Purging policies as follows, can be set by the user to regulate the contents of the trash.
  - Files whose age exceeds a user configurable time period can be purged.

Appendix