NAME

btrfs-rescue − Recover a damaged btrfs filesystem

SYNOPSIS

btrfs rescue <*subcommand*> <*args*>

DESCRIPTION

btrfs rescue is used to try to recover a damaged btrfs filesystem.

SUBCOMMAND

chunk−recover [options] <*device*>

Recover the chunk tree by scanning the devices

Options

−y
: assume an answer of *yes* to all questions.

−h
: help.

−v
: (deprecated) alias for global *−v* option

Note

Since **chunk−recover** will scan the whole device, it will be **VERY** slow especially executed on a large device.

fix−device−size <*device*>

fix device size and super block total bytes values that are do not match

Kernel 4.11 starts to check the device size more strictly and this might mismatch the stored value of total bytes. See the exact error message below. Newer kernel will refuse to mount the filesystem where the values do not match. This error is not fatal and can be fixed. This command will fix the device size values if possible.

BTRFS error (device sdb): super_total_bytes 92017859088384 mismatch with fs_devices total_rw_bytes 920178590

The mismatch may also exhibit as a kernel warning:

WARNING: CPU: 3 PID: 439 at fs/btrfs/ctree.h:1559 btrfs_update_device+0x1c5/0x1d0 [btrfs]

clear−uuid−tree <*device*>

Clear uuid tree, so that kernel can re−generate it at next read−write mount.

Since kernel v4.16 there and more sanity check performed, and sometimes non−critical trees like uuid tree can cause problems and reject the mount. In such case, clearing uuid tree may make the filesystem to be mountable again without much risk as it’s built from other trees.

super−recover [options] <*device*>

Recover bad superblocks from good copies.

Options

−y
: assume an answer of *yes* to all questions.

−v
: (deprecated) alias for global *−v* option

zero−log <*device*>

clear the filesystem log tree

This command will clear the filesystem log tree. This may fix a specific set of problem when the filesystem mount fails due to the log replay. See below for sample stacktraces that may show up in system log.

The common case where this happens was fixed a long time ago, so it is unlikely that you will see this particular problem, but the command is kept around.

Note

clearing the log may lead to loss of changes that were made since the last transaction commit. This may be up to 30 seconds (default commit period) or less if the commit was implied by other filesystem activity.

One can determine whether **zero−log** is needed according to the kernel backtrace:

```
? replay_one_dir_item+0xb5/0xb5 [btrfs]
? walk_log_tree+0x9c/0x19d [btrfs]
? btrfs_read_fs_root_no_radix+0x169/0x1a1 [btrfs]
? btrfs_recover_log_trees+0x195/0x29c [btrfs]
? replay_one_dir_item+0xb5/0xb5 [btrfs]
? btree_read_extent_buffer_pages+0x76/0xbc [btrfs]
? open_ctree+0xff6/0x132c [btrfs]
```

If the errors are like above, then **zero−log** should be used to clear the log and the filesystem may be mounted normally again. The keywords to look for are *open_ctree* which says that it's during mount and function names that contain *replay*, *recover* or *log_tree*.

EXIT STATUS

btrfs rescue returns a zero exit status if it succeeds. Non zero is returned in case of failure.

AVAILABILITY

btrfs is part of btrfs−progs. Please refer to the btrfs wiki **http://btrfs.wiki.kernel.org** for further details.

SEE ALSO

mkfs.btrfs(8), **btrfs−scrub**(8), **btrfs−check**(8)