NAME

zerofree — zero free blocks from ext2, ext3 and ext4 file-systems

SYNOPSIS

zerofree [**-n**] [**-v**] [**-f fillval**] *filesystem*

DESCRIPTION

zerofree finds the unallocated, blocks with non-zero value content in an ext2, ext3 or ext4 *filesystem* (e.g. /dev/hda1) and fills them with zeroes (or another octet of your choice).

Filling unused areas with zeroes is useful if the device on which this file-system resides is a disk image. In this case, depending on the type of disk image, a secondary utility may be able to reduce the size of the disk image after zerofree has been run.

Filling unused areas may also be useful with solid-state drives (SSDs). On some SSDs, filling blocks with ones (0xFF) is reported to trigger Flash block erasure by the firmware, possibly giving a write performance increase.

The usual way to achieve the same result (zeroing the unallocated blocks) is to run **dd** (1) to create a file full of zeroes that takes up the entire free space on the drive, and then delete this file. This has many disadvantages, which zerofree alleviates:

- it is slow;
- it makes the disk image (temporarily) grow to its maximal extent;
- it (temporarily) uses all free space on the disk, so other concurrent write actions may fail.

filesystem has to be unmounted or mounted read-only for **zerofree** to work. It will exit with an error message if the *filesystem* is mounted writable. To remount the root file-system readonly, you can first switch to single user runlevel (**telinit 1**) then use **mount −o remount,ro** *filesystem*.

zerofree has been written to be run from GNU/Linux systems installed as guest OSes inside a virtual machine. In this case, it is typically run from within the guest system, and a utility is then run from the host system to shrink disk image (**VBoxManage modifyhd −−compact**, provided with virtualbox, is able to do that for some disk image formats).

It may however be useful in other situations: for instance it can be used to make it more difficult to retrieve deleted data. Beware that securely deleting sensitive data is not in general an easy task and usually requires writing several times on the deleted blocks.

OPTIONS

-n Perform a dry run (do not modify the file-system);

-v Be verbose: show the number of blocks modified by **zerofree** (or that would be modified, in case the **-n** is used), the number of free blocks and the total number of blocks on the filesystem;

-f value Specify the octet value to fill empty blocks with (defaults to 0). Argument must be within the range 0 to 255.

SEE ALSO

dd (1).

AUTHOR

This manual page was written by Thibaut Paumard <paumard@users.sourceforge.net> for the **Debian** system (but may be used by others). Permission is granted to copy, distribute and/or modify this document