NAME
tune2fs - adjust tunable filesystem parameters on ext2/ext3/ext4 filesystems

SYNOPSIS
tune2fs [**-l**] [**-c** *max-mount-counts*] [**-e** *errors-behavior*] [**-f**] [**-i** *interval-between-checks*] [**-I** *new_inode_size*] [**-j**] [**-J** *journal-options*] [**-m** *reserved-blocks-percentage*] [**-o** [^]mount-options*[,...]*] [**-p** *mmp_update_interval*] [**-r** *reserved-blocks-count*] [**-s** *sparse-super-flag*] [**-u** *user*] [**-g** *group*] [**-C** *mount-count*] [**-E** *extended-options*] [**-L** *volume-label*] [**-M** *last-mounted-directory*] [**-O** [^]*feature*[,...]] [**-Q** *quota-options*] [**-T** *time-last-checked*] [**-U** *UUID*] [**-z** *undo_file*] device

DESCRIPTION
tune2fs allows the system administrator to adjust various tunable filesystem parameters on Linux ext2, ext3, or ext4 filesystems. The current values of these options can be displayed by using the **-l** option to tune2fs(8) program, or by using the dumpe2fs(8) program.

The *device* specifier can either be a filename (i.e., /dev/sda1), or a LABEL or UUID specifier: "**LABEL**=*volume-label*" or "**UUID**=*uuid*". (i.e., LABEL=home or UUID=e40486c6-84d5-4f2f-b99c-032281799c9d).

OPTIONS
-c *max-mount-counts*

> Adjust the number of mounts after which the filesystem will be checked by e2fsck(8). If *max-mount-counts* is 0 or -1, the number of times the filesystem is mounted will be disregarded by e2fsck(8) and the kernel.

> Staggering the mount-counts at which filesystems are forcibly checked will avoid all filesystems being checked at one time when using journaled filesystems.

> You should strongly consider the consequences of disabling mount-count-dependent checking entirely. Bad disk drives, cables, memory, and kernel bugs could all corrupt a filesystem without marking the filesystem dirty or in error. If you are using journaling on your filesystem, your filesystem will **never** be marked dirty, so it will not normally be checked. A filesystem error detected by the kernel will still force an fsck on the next reboot, but it may already be too late to prevent data loss at that point.

> See also the **-i** option for time-dependent checking.

-C *mount-count*

> Set the number of times the filesystem has been mounted. If set to a greater value than the max-mount-counts parameter set by the **-c** option, e2fsck(8) will check the filesystem at the next reboot.

-e *error-behavior*

> Change the behavior of the kernel code when errors are detected. In all cases, a filesystem error will cause e2fsck(8) to check the filesystem on the next boot. *error-behavior* can be one of the following:

> | | |
> |---|---|
> | **continue** | Continue normal execution. |
> | **remount-ro** | Remount filesystem read-only. |
> | **panic** | Cause a kernel panic. |

-E *extended-options*

> Set extended options for the filesystem. Extended options are comma separated, and may take an argument using the equals ('=') sign. The following extended options are supported:

> > **clear_mmp**
> >
> > > Reset the MMP block (if any) back to the clean state. Use only if absolutely certain the device is not currently mounted or being fscked, or major filesystem corruption can result. Needs '-f'.

mmp_update_interval=*interval*

> Adjust the initial MMP update interval to *interval* seconds. Specifying an *interval* of 0 means to use the default interval. The specified interval must be less than 300 seconds. Requires that the **mmp** feature be enabled.

stride=*stride-size*

> Configure the filesystem for a RAID array with *stride-size* filesystem blocks. This is the number of blocks read or written to disk before moving to next disk. This mostly affects placement of filesystem metadata like bitmaps at **mke2fs(2)** time to avoid placing them on a single disk, which can hurt the performance. It may also be used by block allocator.

stripe_width=*stripe-width*

> Configure the filesystem for a RAID array with *stripe-width* filesystem blocks per stripe. This is typically be stride-size * N, where N is the number of data disks in the RAID (e.g. RAID 5 N+1, RAID 6 N+2). This allows the block allocator to prevent read-modify-write of the parity in a RAID stripe if possible when the data is written.

hash_alg=*hash-alg*

> Set the default hash algorithm used for filesystems with hashed b-tree directories. Valid algorithms accepted are: *legacy*, *half_md4*, and *tea*.

mount_opts=*mount_option_string*

> Set a set of default mount options which will be used when the file system is mounted. Unlike the bitmask-based default mount options which can be specified with the **-o** option, *mount_option_string* is an arbitrary string with a maximum length of 63 bytes, which is stored in the superblock.

> The ext4 file system driver will first apply the bitmask-based default options, and then parse the *mount_option_string*, before parsing the mount options passed from the mount(8) program.

> This superblock setting is only honored in 2.6.35+ kernels; and not at all by the ext2 and ext3 file system drivers.

test_fs   Set a flag in the filesystem superblock indicating that it may be mounted using experimental kernel code, such as the ext4dev filesystem.

ˆtest_fs   Clear the test_fs flag, indicating the filesystem should only be mounted using production-level filesystem code.

-f   Force the tune2fs operation to complete even in the face of errors. This option is useful when removing the **has_journal** filesystem feature from a filesystem which has an external journal (or is corrupted such that it appears to have an external journal), but that external journal is not available. If the filesystem appears to require journal replay, the **-f** flag must be specified twice to proceed.

> **WARNING:** Removing an external journal from a filesystem which was not cleanly unmounted without first replaying the external journal can result in severe data loss and filesystem corruption.

-g *group*

> Set the group which can use the reserved filesystem blocks. The *group* parameter can be a numerical gid or a group name. If a group name is given, it is converted to a numerical gid before it is stored in the superblock.

-i *interval-between-checks*[**d**|**m**|**w**]

> Adjust the maximal time between two filesystem checks. No suffix or **d** will interpret the number *interval-between-checks* as days, **m** as months, and **w** as weeks. A value of zero will disable the time-dependent checking.

> It is strongly recommended that either **-c** (mount-count-dependent) or **-i** (time-dependent) checking be enabled to force periodic full e2fsck(8) checking of the filesystem. Failure to do so may

lead to filesystem corruption (due to bad disks, cables, memory, or kernel bugs) going unnoticed, ultimately resulting in data loss or corruption.

-I   Change the inode size used by the file system. This requires rewriting the inode table, so it requires that the file system is checked for consistency first using e2fsck(8). This operation can also take a while and the file system can be corrupted and data lost if it is interrupted while in the middle of converting the file system.

-j   Add an ext3 journal to the filesystem. If the **-J** option is not specified, the default journal parameters will be used to create an appropriately sized journal (given the size of the filesystem) stored within the filesystem. Note that you must be using a kernel which has ext3 support in order to actually make use of the journal.

If this option is used to create a journal on a mounted filesystem, an immutable file, **.journal**, will be created in the top-level directory of the filesystem, as it is the only safe way to create the journal inode while the filesystem is mounted. While the ext3 journal is visible, it is not safe to delete it, or modify it while the filesystem is mounted; for this reason the file is marked immutable. While checking unmounted filesystems, e2fsck(8) will automatically move **.journal** files to the invisible, reserved journal inode. For all filesystems except for the root filesystem, this should happen automatically and naturally during the next reboot cycle. Since the root filesystem is mounted read-only, e2fsck(8) must be run from a rescue floppy in order to effect this transition.

On some distributions, such as Debian, if an initial ramdisk is used, the initrd scripts will automatically convert an ext2 root filesystem to ext3 if the **/etc/fstab** file specifies the ext3 filesystem for the root filesystem in order to avoid requiring the use of a rescue floppy to add an ext3 journal to the root filesystem.

-J journal-options

Override the default ext3 journal parameters. Journal options are comma separated, and may take an argument using the equals ('=') sign. The following journal options are supported:

        size= *journal-size*

                Create a journal stored in the filesystem of size *journal-size* megabytes. The size of the journal must be at least 1024 filesystem blocks (i.e., 1MB if using 1k blocks, 4MB if using 4k blocks, etc.) and may be no more than 10,240,000 filesystem blocks. There must be enough free space in the filesystem to create a journal of that size.

        location=*journal-location*

                Specify the location of the journal. The argument *journal-location* can either be specified as a block number, or if the number has a units suffix (e.g., 'M', 'G', etc.) interpret it as the offset from the beginning of the file system.

        device=*external-journal*

                Attach the filesystem to the journal block device located on *external-journal*. The external journal must have been already created using the command

                mke2fs -O journal_dev *external-journal*

                Note that *external-journal* must be formatted with the same block size as filesystems which will be using it. In addition, while there is support for attaching multiple filesystems to a single external journal, the Linux kernel and e2fsck(8) do not currently support shared external journals yet.

                Instead of specifying a device name directly, *external-journal* can also be specified by either **LABEL=***label* or **UUID=***UUID* to locate the external journal by either the volume label or UUID stored in the ext2 superblock at the start of the journal. Use dumpe2fs(8) to display a journal device's volume label and UUID. See also the **-L** option of tune2fs(8).

Only one of the **size** or **device** options can be given for a filesystem.

-l   List the contents of the filesystem superblock, including the current values of the parameters that can be set via this program.

-L *volume-label*

Set the volume label of the filesystem. Ext2 filesystem labels can be at most 16 characters long; if *volume-label* is longer than 16 characters, **tune2fs** will truncate it and print a warning. The volume label can be used by mount(8), fsck(8), and **/etc/fstab**(5) (and possibly others) by specifying **LABEL=***volume-label* instead of a block special device name like **/dev/hda5**.

-m *reserved-blocks-percentage*

Set the percentage of the filesystem which may only be allocated by privileged processes. Reserving some number of filesystem blocks for use by privileged processes is done to avoid filesystem fragmentation, and to allow system daemons, such as **syslogd(8),** to continue to function correctly after non-privileged processes are prevented from writing to the filesystem. Normally, the default percentage of reserved blocks is 5%.

-M *last-mounted-directory*

Set the last-mounted directory for the filesystem.

-o [^]*mount-option*[,...]

Set or clear the indicated default mount options in the filesystem. Default mount options can be overridden by mount options specified either in **/etc/fstab**(5) or on the command line arguments to mount(8). Older kernels may not support this feature; in particular, kernels which predate 2.4.20 will almost certainly ignore the default mount options field in the superblock.

More than one mount option can be cleared or set by separating features with commas. Mount options prefixed with a caret character ('^') will be cleared in the filesystem's superblock; mount options without a prefix character or prefixed with a plus character ('+') will be added to the filesystem.

The following mount options can be set or cleared using **tune2fs**:

> **debug**   Enable debugging code for this filesystem.
>
> **bsdgroups**
>> Emulate BSD behavior when creating new files: they will take the group-id of the directory in which they were created. The standard System V behavior is the default, where newly created files take on the fsgid of the current process, unless the directory has the setgid bit set, in which case it takes the gid from the parent directory, and also gets the setgid bit set if it is a directory itself.
>
> **user_xattr**
>> Enable user-specified extended attributes.
>
> **acl**   Enable Posix Access Control Lists.
>
> **uid16**   Disables 32-bit UIDs and GIDs. This is for interoperability with older kernels which only store and expect 16-bit values.
>
> **journal_data**
>> When the filesystem is mounted with journalling enabled, all data (not just metadata) is committed into the journal prior to being written into the main filesystem.
>
> **journal_data_ordered**
>> When the filesystem is mounted with journalling enabled, all data is forced directly out to the main file system prior to its metadata being committed to the journal.
>
> **journal_data_writeback**
>> When the filesystem is mounted with journalling enabled, data may be written into the main filesystem after its metadata has been committed to the

journal. This may increase throughput, however, it may allow old data to appear in files after a crash and journal recovery.

nobarrier
The file system will be mounted with barrier operations in the journal disabled. (This option is currently only supported by the ext4 file system driver in 2.6.35+ kernels.)

block_validity
The file system will be mounted with the block_validity option enabled, which causes extra checks to be performed after reading or writing from the file system. This prevents corrupted metadata blocks from causing file system damage by overwriting parts of the inode table or block group descriptors. This comes at the cost of increased memory and CPU overhead, so it is enabled only for debugging purposes. (This option is currently only supported by the ext4 file system driver in 2.6.35+ kernels.)

discard
The file system will be mounted with the discard mount option. This will cause the file system driver to attempt to use the trim/discard feature of some storage devices (such as SSD's and thin-provisioned drives available in some enterprise storage arrays) to inform the storage device that blocks belonging to deleted files can be reused for other purposes. (This option is currently only supported by the ext4 file system driver in 2.6.35+ kernels.)

nodelalloc
The file system will be mounted with the nodelalloc mount option. This will disable the delayed allocation feature. (This option is currently only supported by the ext4 file system driver in 2.6.35+ kernels.)

-O [ˆ]*feature*[,...]
Set or clear the indicated filesystem features (options) in the filesystem. More than one filesystem feature can be cleared or set by separating features with commas. Filesystem features prefixed with a caret character ('ˆ') will be cleared in the filesystem's superblock; filesystem features without a prefix character or prefixed with a plus character ('+') will be added to the filesystem. For a detailed description of the file system features, please see the man page ext4(5).

The following filesystem features can be set or cleared using **tune2fs**:

dir_index
Use hashed b-trees to speed up lookups for large directories.

dir_nlink
Allow more than 65000 subdirectories per directory.

encrypt
Enable file system level encryption. **Tune2fs** currently only supports setting this filesystem feature.

extent   Enable the use of extent trees to store the location of data blocks in inodes. **Tune2fs** currently only supports setting this filesystem feature.

extra_isize
Enable the extended inode fields used by ext4.

filetype
Store file type information in directory entries.

flex_bg Allow bitmaps and inode tables for a block group to be placed anywhere on the storage media. **Tune2fs** will not reorganize the location of the inode tables and allocation bitmaps, as mke2fs(8) will do when it creates a freshly formatted file system with **flex_bg** enabled.

has_journal
>   Use a journal to ensure filesystem consistency even across unclean shutdowns. Setting the filesystem feature is equivalent to using the **-j** option.

huge_file
>   Support files larger than 2 terabytes in size.

large_file
>   Filesystem can contain files that are greater than 2GB.

metadata_csum
>   Store a checksum to protect the contents in each metadata block.

mmp   Enable or disable multiple mount protection (MMP) feature.

project Enable project ID tracking. This is used for project quota tracking.

quota   Enable internal file system quota inodes.

read-only
>   Force the kernel to mount the file system read-only.

resize_inode
>   Reserve space so the block group descriptor table may grow in the future. **Tune2fs** only supports clearing this filesystem feature.

sparse_super
>   Limit the number of backup superblocks to save space on large filesystems. **Tune2fs** currently only supports setting this filesystem feature.

uninit_bg
>   Allow the kernel to initialize bitmaps and inode tables lazily, and to keep a high watermark for the unused inodes in a filesystem, to reduce e2fsck(8) time. The first e2fsck run after enabling this feature will take the full time, but subsequent e2fsck runs will take only a fraction of the original time, depending on how full the file system is.

After setting or clearing **sparse_super**, **uninit_bg**, **filetype**, or **resize_inode** filesystem features, the file system may require being checked using e2fsck(8) to return the filesystem to a consistent state. **Tune2fs** will print a message requesting that the system administrator run e2fsck(8) if necessary. After setting the **dir_index** feature, **e2fsck -D** can be run to convert existing directories to the hashed B-tree format. Enabling certain filesystem features may prevent the filesystem from being mounted by kernels which do not support those features. In particular, the **uninit_bg** and **flex_bg** features are only supported by the ext4 filesystem.

-p *mmp_check_interval*
>   Set the desired MMP check interval in seconds. It is 5 seconds by default.

-r *reserved-blocks-count*
>   Set the number of reserved filesystem blocks.

-Q *quota-options*
>   Sets 'quota' feature on the superblock and works on the quota files for the given quota type. Quota options could be one or more of the following:

>>   **[ˆ]usrquota**
>>>   Sets/clears user quota inode in the superblock.

>>   **[ˆ]grpquota**
>>>   Sets/clears group quota inode in the superblock.

>>   **[ˆ]prjquota**
>>>   Sets/clears project quota inode in the superblock.

-T *time-last-checked*

Set the time the filesystem was last checked using **e2fsck**. The time is interpreted using the current (local) timezone. This can be useful in scripts which use a Logical Volume Manager to make a consistent snapshot of a filesystem, and then check the filesystem during off hours to make sure it hasn't been corrupted due to hardware problems, etc. If the filesystem was clean, then this option can be used to set the last checked time on the original filesystem. The format of *time-last-checked* is the international date format, with an optional time specifier, i.e. YYYYMMDD[HH[MM[SS]]]. The keyword **now** is also accepted, in which case the last checked time will be set to the current time.

-u *user* Set the user who can use the reserved filesystem blocks. *user* can be a numerical uid or a user name. If a user name is given, it is converted to a numerical uid before it is stored in the superblock.

-U *UUID*

Set the universally unique identifier (UUID) of the filesystem to *UUID*. The format of the UUID is a series of hex digits separated by hyphens, like this: "c1b9d5a2-f162-11cf-9ece-0020afc76f16". The *UUID* parameter may also be one of the following:

> *clear*   clear the filesystem UUID
>
> *random*
> generate a new randomly-generated UUID
>
> *time*   generate a new time-based UUID

The UUID may be used by mount(8), fsck(8), and **/etc/fstab**(5) (and possibly others) by specifying **UUID=***uuid* instead of a block special device name like **/dev/hda1**.

See **uuidgen(8)** for more information. If the system does not have a good random number generator such as */dev/random* or */dev/urandom*, **tune2fs** will automatically use a time-based UUID instead of a randomly-generated UUID.

-z *undo_file*

Before overwriting a file system block, write the old contents of the block to an undo file. This undo file can be used with e2undo(8) to restore the old contents of the file system should something go wrong. If the empty string is passed as the undo_file argument, the undo file will be written to a file named tune2fs-*device*.e2undo in the directory specified via the *E2FSPROGS_UNDO_DIR* environment variable.

WARNING: The undo file cannot be used to recover from a power or system crash.

BUGS

We haven't found any bugs yet. That doesn't mean there aren't any...

AUTHOR

tune2fs was written by Remy Card <Remy.Card@linux.org>. It is currently being maintained by Theodore Ts'o <tytso@alum.mit.edu>. **tune2fs** uses the ext2fs library written by Theodore Ts'o <tytso@mit.edu>. This manual page was written by Christian Kuhtz <chk@data-hh.Hanse.DE>. Time-dependent checking was added by Uwe Ohse <uwe@tirka.gun.de>.

AVAILABILITY

tune2fs is part of the e2fsprogs package and is available from http://e2fsprogs.sourceforge.net.

SEE ALSO

debugfs(8), dumpe2fs(8), e2fsck(8), mke2fs(8), ext4(5)