# mkfs.ext4(8) - Linux man page

## Name

mke2fs - create an ext2/ext3/ext4 filesystem

## Synopsis

**mke2fs** [ **-c** | **-l** *filename* ] [ **-b** *block-size* ] [ **-f***fragment-size* ] [ **-g** *blocks-per-group* ] [ **-G***number-of-groups* ] [ **-i** *bytes-per-inode* ] [ **-I***inode-size* ] [ **-j** ] [ **-J** *journal-options* ] [ **-K** ] [ **-N** *number-of-inodes* ] [ **-n** ] [ **-m** *reserved-blocks-percentage* ] [ **-o** *creator-os* ] [ **-O***feature*[,...] ] [ **-q** ] [ **-r** *fs-revision-level* ] [ **-E***extended-options* ] [ **-v** ] [ **-F** ] [ **-L** *volume-label* ] [ **-M** *last-mounted-directory* ] [ **-S** ] [ **-t***fs-type* ] [ **-T** *usage-type* ] [ **-U** *UUID* ] [ **-V** ] *device* [ *blocks-count* ]

**mke2fs -O journal\_dev** [ **-b** *block-size* ] [ **-L***volume-label* ] [ **-n** ] [ **-q** ] [ **-v** ] *external-journal* [ *blocks-count* ]

## Description

**mke2fs** is used to create an ext2, ext3, or ext4 filesystem, usually in a disk partition. *device* is the special file corresponding to the device (e.g */dev/hdXX*). *blocks-count* is the number of blocks on the device. If omitted, **mke2fs** automagically figures the file system size. If called as **mkfs.ext3** a journal is created as if the **-j** option was specified.

The defaults of the parameters for the newly created filesystem, if not overridden by the options listed below, are controlled by the **/etc/mke2fs.conf** configuration file. See the ***[mke2fs.conf](https://linux.die.net/man/5/mke2fs.conf)****(5)* manual page for more details.

## Options

**-b** *block-size*

Specify the size of blocks in bytes. Valid block-size values are 1024, 2048 and 4096 bytes per block. If omitted, block-size is heuristically determined by the filesystem size and the expected usage of the filesystem (see the **-T** option). If *block-size* is negative, then **mke2fs** will use heuristics to determine the appropriate block size, with the constraint that the block size will be at least *block-size* bytes. This is useful for certain hardware devices which require that the blocksize be a multiple of 2k.

**-c**

Check the device for bad blocks before creating the file system. If this option is specified twice, then a slower read-write test is used instead of a fast read-only test.

**-E** *extended-options*

Set extended options for the filesystem. Extended options are comma separated, and may take an argument using the equals ('=') sign. The **-E** option used to be **-R** in earlier versions of **mke2fs**. The **-R** option is still accepted for backwards compatibility. The following extended options are supported:

**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 the next disk, which is sometimes referred to as the *chunk* size. This mostly affects placement of filesystem metadata like bitmaps at **mke2fs** time to avoid placing them on a single disk, which can hurt performance. It may also be used by the block allocator.

**stripe-width=***stripe-width*

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

**resize=***max-online-resize*

Reserve enough space so that the block group descriptor table can grow to support a filesystem that has *max-online-resize* blocks.

**lazy\_itable\_init[**= *<0 to disable, 1 to enable>*]

If enabled and the uninit\_bg feature is enabled, the inode table will not be fully initialized by **mke2fs**. This speeds up filesystem initialization noticeably, but it requires the kernel to finish initializing the filesystem in the background when the filesystem is first mounted. If the option value is omitted, it defaults to 1 to enable lazy inode table initialization.

**test\_fs**

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

**discard**

Attempt to discard blocks at mkfs time (discarding blocks initially is useful on solid state devices and sparse / thin-provisioned storage). When the device advertises that discard also zeroes data (any subsequent read after the discard and before write returns zero), then mark all not-yet-zeroed inode tables as zeroed. This significantly speeds up filesystem initialization. This is set as default.

**nodiscard**

Do not attempt to discard blocks at mkfs time.

**-f** *fragment-size*

Specify the size of fragments in bytes.

**-F**

Force **mke2fs** to create a filesystem, even if the specified device is not a partition on a block special device, or if other parameters do not make sense. In order to force **mke2fs** to create a filesystem even if the filesystem appears to be in use or is mounted (a truly dangerous thing to do), this option must be specified twice.

**-g** *blocks-per-group*

Specify the number of blocks in a block group. There is generally no reason for the user to ever set this parameter, as the default is optimal for the filesystem. (For administrators who are creating filesystems on RAID arrays, it is preferable to use the *stride* RAID parameter as part of the **-E**option rather than manipulating the number of blocks per group.) This option is generally used by developers who are developing test cases.

**-G** *number-of-groups*

Specify the number of block groups that will be packed together to create a larger virtual block group (or "flex\_bg group") in an ext4 filesystem. This improves meta-data locality and performance on meta-data heavy workloads. The number of groups must be a power of 2 and may only be specified if the **flex\_bg** filesystem feature is enabled.

**-i** *bytes-per-inode*

Specify the bytes/inode ratio. **mke2fs** creates an inode for every *bytes-per-inode* bytes of space on the disk. The larger the *bytes-per-inode* ratio, the fewer inodes will be created. This value generally shouldn't be smaller than the blocksize of the filesystem, since in that case more inodes would be made than can ever be used. Be warned that it is not possible to expand the number of inodes on a filesystem after it is created, so be careful deciding the correct value for this parameter.

**-I** *inode-size*

Specify the size of each inode in bytes. **mke2fs** creates 256-byte inodes by default. In kernels after 2.6.10 and some earlier vendor kernels it is possible to utilize inodes larger than 128 bytes to store extended attributes for improved performance. The *inode-size* value must be a power of 2 larger or equal to 128. The larger the *inode-size* the more space the inode table will consume, and this reduces the usable space in the filesystem and can also negatively impact performance. Extended attributes stored in large inodes are not visible with older kernels, and such filesystems will not be mountable with 2.4 kernels at all. It is not possible to change this value after the filesystem is created.

**-j**

Create the filesystem with an ext3 journal. 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.

**-J** *journal-options*

Create the ext3 journal using options specified on the command-line. Journal options are comma separated, and may take an argument using the equals ('=') sign. The following journal options are supported:

**size=***journal-size*

Create an internal journal (i.e., stored inside 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 102,400 filesystem blocks.

**device=***external-journal*

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

**mke2fs -O journal\_dev**

*external-journal*

Note that

*external-journal* must have been created with the same block size as the new filesystem. In addition, while there is support for attaching multiple filesystems to a single external journal, the Linux kernel and ***[e2fsck](https://linux.die.net/man/8/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](https://linux.die.net/man/8/dumpe2fs)****(8)* to display a journal device's volume label and UUID. See also the **-L** option of ***[tune2fs](https://linux.die.net/man/8/tune2fs)****(8)*.

Only one of the

**size** or **device** options can be given for a filesystem.

**-K**

Keep, do not attempt to discard blocks at mkfs time (discarding blocks initially is useful on solid state devices and sparse / thin-provisioned storage).

**-l** *filename*

Read the bad blocks list from *filename*. Note that the block numbers in the bad block list must be generated using the same block size as used by **mke2fs**. As a result, the **-c** option to **mke2fs** is a much simpler and less error-prone method of checking a disk for bad blocks before formatting it, as **mke2fs** will automatically pass the correct parameters to the **badblocks** program.

**-L** *new-volume-label*

Set the volume label for the filesystem to *new-volume-label*. The maximum length of the volume label is 16 bytes.

**-m** *reserved-blocks-percentage*

Specify the percentage of the filesystem blocks reserved for the super-user. This avoids fragmentation, and allows root-owned daemons, such as ***[syslogd](https://linux.die.net/man/8/syslogd)****(8)*, to continue to function correctly after non-privileged processes are prevented from writing to the filesystem. The default percentage is 5%.

**-M** *last-mounted-directory*

Set the last mounted directory for the filesystem. This might be useful for the sake of utilities that key off of the last mounted directory to determine where the filesystem should be mounted.

**-n**

Causes **mke2fs** to not actually create a filesystem, but display what it would do if it were to create a filesystem. This can be used to determine the location of the backup superblocks for a particular filesystem, so long as the **mke2fs** parameters that were passed when the filesystem was originally created are used again. (With the **-n** option added, of course!)

**-N** *number-of-inodes*

Overrides the default calculation of the number of inodes that should be reserved for the filesystem (which is based on the number of blocks and the *bytes-per-inode* ratio). This allows the user to specify the number of desired inodes directly.

**-o** *creator-os*

Overrides the default value of the "creator operating system" field of the filesystem. The creator field is set by default to the name of the OS the **mke2fs** executable was compiled for.

**-O *feature*[,...]**

Create a filesystem with the given features (filesystem options), overriding the default filesystem options. The features that are enabled by default are specified by the *base\_features* relation, either in the *[defaults]* section in the **/etc/mke2fs.conf** configuration file, or in the *[fs\_types]* subsections for the usage types as specified by the **-T** option, further modified by the *features* relation found in the *[fs\_types]* subsections for the filesystem and usage types. See the ***[mke2fs.conf](https://linux.die.net/man/5/mke2fs.conf)****(5)* manual page for more details. The filesystem type-specific configuration setting found in the *[fs\_types]* section will override the global default found in *[defaults]*.

The filesystem feature set will be further edited using either the feature set specified by this option, or if this option is not given, by the *default\_features* relation for the filesystem type being created, or in the *[defaults]* section of the configuration file.

The filesystem feature set is comprised of a list of features, separated by commas, that are to be enabled. To disable a feature, simply prefix the feature name with a caret ('^') character. The pseudo-filesystem feature "none" will clear all filesystem features.

**dir\_index**

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

**extent**

Instead of using the indirect block scheme for storing the location of data blocks in an inode, use extents instead. This is a much more efficient encoding which speeds up filesystem access, especially for large files.

**filetype**

Store file type information in directory entries.

**flex\_bg**

Allow the per-block group metadata (allocation bitmaps and inode tables) to be placed anywhere on the storage media. In addition, **mke2fs** will place the per-block group metadata together starting at the first block group of each "flex\_bg group". The size of the flex\_bg group can be specified using the **-G** option.

**has\_journal**

Create an ext3 journal (as if using the **-j** option).

**journal\_dev**

Create an external ext3 journal on the given device instead of a regular ext2 filesystem. Note that *external-journal* must be created with the same block size as the filesystems that will be using it.

**large\_file**

Filesystem can contain files that are greater than 2GB. (Modern kernels set this feature automatically when a file > 2GB is created.)

**resize\_inode**

Reserve space so the block group descriptor table may grow in the future. Useful for online resizing using **resize2fs**. By default **mke2fs** will attempt to reserve enough space so that the filesystem may grow to 1024 times its initial size. This can be changed using the **resize** extended option.

**sparse\_super**

Create a filesystem with fewer superblock backup copies (saves space on large filesystems).

**uninit\_bg**

Create a filesystem without initializing all of the block groups. This feature also enables checksums and highest-inode-used statistics in each blockgroup. This feature can speed up filesystem creation time noticeably (if lazy\_itable\_init is enabled), and can also reduce **e2fsck** time dramatically. It is only supported by the ext4 filesystem in recent Linux kernels.

**-q**

Quiet execution. Useful if **mke2fs** is run in a script.

**-r** *revision*

Set the filesystem revision for the new filesystem. Note that 1.2 kernels only support revision 0 filesystems. The default is to create revision 1 filesystems.

**-S**

Write superblock and group descriptors only. This is useful if all of the superblock and backup superblocks are corrupted, and a last-ditch recovery method is desired. It causes **mke2fs** to reinitialize the superblock and group descriptors, while not touching the inode table and the block and inode bitmaps. The **e2fsck** program should be run immediately after this option is used, and there is no guarantee that any data will be salvageable. It is critical to specify the correct filesystem blocksize when using this option, or there is no chance of recovery.

**-t** *fs-type*

Specify the filesystem type (i.e., ext2, ext3, ext4, etc.) that is to be created. If this option is not specified, **mke2fs** will pick a default either via how the command was run (for example, using a name of the form mkfs.ext2, mkfs.ext3, etc.) or via a default as defined by the**/etc/*mke2fs.conf****(5)* file. This option controls which filesystem options are used by default, based on the **fstypes** configuration stanza in **/etc/*mke2fs.conf****(5)*.

If the **-O** option is used to explicitly add or remove filesystem options that should be set in the newly created filesystem, the resulting filesystem may not be supported by the requested *fs-type*. (e.g., "**mke2fs -t ext3 -O extents /dev/sdXX**" will create a filesystem that is not supported by the ext3 implementation as found in the Linux kernel; and "**mke2fs -t ext3 -O ^has\_journal /dev/hdXX**" will create a filesystem that does not have a journal and hence will not be supported by the ext3 filesystem code in the Linux kernel.)

**-T** *usage-type[,...]*

Specify how the filesystem is going to be used, so that **mke2fs** can choose optimal filesystem parameters for that use. The usage types that are supported are defined in the configuration file **/etc/*mke2fs.conf****(5)*. The user may specify one or more usage types using a comma separated list.

If this option is is not specified, **mke2fs** will pick a single default usage type based on the size of the filesystem to be created. If the filesystem size is less than or equal to 3 megabytes, **mke2fs** will use the filesystem type *floppy*. If the filesystem size is greater than 3 but less than or equal to 512 megabytes, ***mke2fs****(8)* will use the filesystem *small*. Otherwise, ***mke2fs****(8)* will use the default filesystem type *default*.

**-U** *UUID*

Create the filesystem with the specified UUID.

**-v**

Verbose execution.

**-V**

Print the version number of **mke2fs** and exit.

## Author

This version of **mke2fs** has been written by Theodore Ts'o <[tytso@mit.edu](mailto:tytso@mit.edu)>.

## Bugs

**mke2fs** accepts the **-f** option but currently ignores it because the second extended file system does not support fragments yet.  
There may be other ones. Please, report them to the author.

## Availability

**mke2fs** is part of the e2fsprogs package and is available from [http://e2fsprogs.sourceforge.net](http://e2fsprogs.sourceforge.net/).

## See Also

***[mke2fs.conf](https://linux.die.net/man/5/mke2fs.conf)****(5)*, ***[badblocks](https://linux.die.net/man/8/badblocks)****(8)*, ***[dumpe2fs](https://linux.die.net/man/8/dumpe2fs)****(8)*, ***[e2fsck](https://linux.die.net/man/8/e2fsck)****(8)*, ***[tune2fs](https://linux.die.net/man/8/tune2fs)****(8)*

## Referenced By

**[mkfs](https://linux.die.net/man/8/mkfs)**(8)

**-i** *bytes-per-inode 每个inode占用的字节数。字节数越大，inode越少。但是不能比blacksize小，为啥呢？Inode数比black数还要多，inode就浪费了。*

**-n** 模拟

[root@aliyun ~]# mkfs.ext4 -n -i 2048 zero

mke2fs 1.41.12 (17-May-2010)

zero is not a block special device.

Proceed anyway? (y,n) y

Filesystem label=

OS type: Linux

Block size=4096 (log=2)

Fragment size=4096 (log=2)

Stride=0 blocks, Stripe width=0 blocks

7681680 inodes, 3840000 blocks

192000 blocks (5.00%) reserved for the super user

First data block=0

Maximum filesystem blocks=2153779200

235 block groups

16400 blocks per group, 16400 fragments per group

32688 inodes per group

Superblock backups stored on blocks:

16400, 49200, 82000, 114800, 147600, 410000, 442800, 803600, 1328400,

2050000

[root@aliyun ~]# mkfs.ext4 -n -i 2048 zero -b 2048

mke2fs 1.41.12 (17-May-2010)

zero is not a block special device.

Proceed anyway? (y,n) y

Filesystem label=

OS type: Linux

Block size=2048 (log=1)

Fragment size=2048 (log=1)

Stride=0 blocks, Stripe width=0 blocks

7680344 inodes, 7680000 blocks

384000 blocks (5.00%) reserved for the super user

First data block=0

Maximum filesystem blocks=545259520

469 block groups

16384 blocks per group, 16384 fragments per group

16376 inodes per group

Superblock backups stored on blocks:

16384, 49152, 81920, 114688, 147456, 409600, 442368, 802816, 1327104,

2048000, 3981312, 5619712

[root@aliyun ~]# mkfs.ext4 -n -i 1200 zero -b 2048

mke2fs 1.41.12 (17-May-2010)

zero is not a block special device.

Proceed anyway? (y,n) y

Filesystem label=

OS type: Linux

Block size=2048 (log=1)

Fragment size=2048 (log=1)

Stride=0 blocks, Stripe width=0 blocks

13107200 inodes, 7680000 blocks

384000 blocks (5.00%) reserved for the super user

First data block=0

Maximum filesystem blocks=322828800

800 block groups

9608 blocks per group, 9608 fragments per group

16384 inodes per group

Superblock backups stored on blocks:

9608, 28824, 48040, 67256, 86472, 240200, 259416, 470792, 778248,

1201000, 2334744, 3295544, 6005000, 7004232

[root@aliyun ~]# mkfs.ext4 -n zero

mke2fs 1.41.12 (17-May-2010)

zero is not a block special device.

Proceed anyway? (y,n) y

Filesystem label=

OS type: Linux

Block size=4096 (log=2)

Fragment size=4096 (log=2)

Stride=0 blocks, Stripe width=0 blocks

960992 inodes, 3840000 blocks

192000 blocks (5.00%) reserved for the super user

First data block=0

Maximum filesystem blocks=3934257152

118 block groups

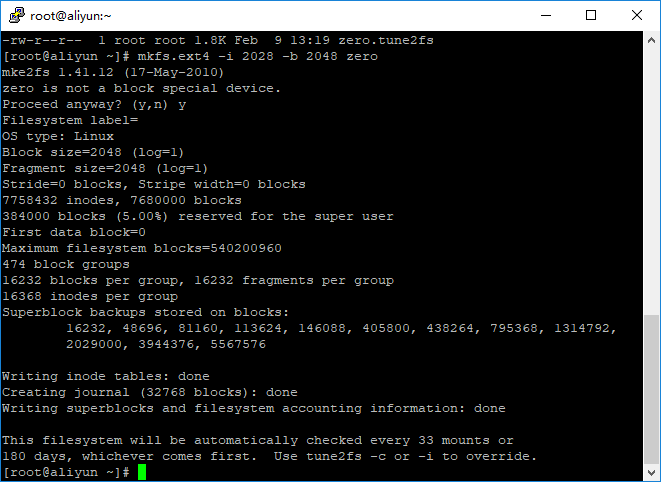
32768 blocks per group, 32768 fragments per group

8144 inodes per group

Superblock backups stored on blocks:

32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208

[root@aliyun ~]#



[root@aliyun ~]# mkfs.ext4 -i 2028 -b 2048 zero

mke2fs 1.41.12 (17-May-2010)

zero is not a block special device.

Proceed anyway? (y,n) y

Filesystem label=

OS type: Linux

Block size=2048 (log=1)

Fragment size=2048 (log=1)

Stride=0 blocks, Stripe width=0 blocks

7758432 inodes, 7680000 blocks

384000 blocks (5.00%) reserved for the super user

First data block=0

Maximum filesystem blocks=540200960

474 block groups

16232 blocks per group, 16232 fragments per group

16368 inodes per group

Superblock backups stored on blocks:

16232, 48696, 81160, 113624, 146088, 405800, 438264, 795368, 1314792,

2029000, 3944376, 5567576

Writing inode tables: done

Creating journal (32768 blocks): done

Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 33 mounts or

180 days, whichever comes first. Use tune2fs -c or -i to override.

[root@aliyun ~]# mount -o loop zero /mnt

[root@aliyun ~]# df -li

Filesystem Inodes IUsed IFree IUse% Mounted on

/dev/vda1 2621440 2589555 31885 99% /

tmpfs 127534 1 127533 1% /dev/shm

/root/zero 7758432 11 7758421 1% /mnt

[root@aliyun ~]# df -lh

Filesystem Size Used Avail Use% Mounted on

/dev/vda1 40G 34G 4.1G 90% /

tmpfs 499M 0 499M 0% /dev/shm

/root/zero 13G 78M 12G 1% /mnt

[root@aliyun ~]#

