NAME

makefs - create a file system image from a directory tree or a mtree manifest

SYNOPSIS

makefs [**-DxZ**] [**-B** *endian*] [**-b** *free-blocks*] [**-d** *debug-mask*] [**-F** *mtree-specfile*] [**-f** *free-files*] [**-M** *minimum-size*] [**-m** *maximum-size*] [**-N** *userdb-dir*] [**-O** *offset*] [**-o** *fs-options*] [**-R** *roundup-size*] [**-S** *sector-size*] [**-s** *image-size*] [**-T** *timestamp*] [**-t** *fs-type*] *image-file* *directory* | *manifest* [*extra-directory* ...]

DESCRIPTION

The utility **makefs** creates a file system image into *image-file* from the directory tree *directory* or from the mtree manifest *manifest*. If any optional directory trees are passed in the *extra-directory* arguments, then the directory tree of each argument will be merged into the *directory* or *manifest* first before creating *image-file*. No special devices or privileges are required to perform this task.

The options are as follows:

-B *endian*

Set the byte order of the image to *endian*. Valid byte orders are '4321', 'big', or 'be' for big endian, and '1234', 'little', or 'le' for little endian. Some file systems may have a fixed byte order; in those cases this argument will be ignored.

-b *free-blocks*

Ensure that a minimum of *free-blocks* free blocks exist in the image. An optional '%' suffix may be provided to indicate that *free-blocks* indicates a percentage of the calculated image size.

-D Treat duplicate paths in an mtree manifest as warnings not error.

-d *debug-mask*

Enable various levels of debugging, depending upon which bits are set in *debug-mask*. XXX: document these

-F *mtree-specfile*

This is almost certainly not the option you are looking for. To create an image from a list of files in an mtree format manifest, specify it as the last argument on the command line, not as a the argument to **-F**.

Use *mtree-specfile* as an mtree(8) 'specfile' specification. This option has no effect when the image is created from a mtree manifest rather than a directory.

If a specfile entry exists in the underlying file system, its permissions and modification time will be used unless specifically overridden by the specfile. An error will be raised if the type of entry in the specfile conflicts with that of an existing entry.

In the opposite case (where a specfile entry does not have an entry in the underlying file system) the following occurs: If the specfile entry is marked **optional**, the specfile entry is ignored. Otherwise, the entry will be created in the image, and it is necessary to specify at least the following parameters in the specfile: **type**, **mode**, **gname**, or **gid**, and **uname** or **uid**, and **link** (in the case of symbolic links). If **time** is not provided, the current time will be used. If **flags** is not provided, the current file flags will be used. Missing regular file entries will be created as zero-length files.

-f *free-files*
Ensure that a minimum of *free-files* free files (inodes) exist in the image. An optional '%' suffix may be provided to indicate that *free-files* indicates a percentage of the calculated image size.

-M *minimum-size*
Set the minimum size of the file system image to *minimum-size*.

-m *maximum-size*
Set the maximum size of the file system image to *maximum-size*. An error will be raised if the target file system needs to be larger than this to accommodate the provided directory tree.

-N *userdb-dir*
Use the user database text file *master.passwd* and group database text file *group* from *userdb-dir*, rather than using the results from the system's getpwnam(3) and getgrnam(3) (and related) library calls.

-O *offset*
Instead of creating the filesystem at the beginning of the file, start at offset. Valid only for **ffs** and **msdos**.

-o *fs-options*
Set file system specific options. *fs-options* is a comma separated list of options. Valid file system specific options are detailed below.

-p Deprecated. See the **-Z** flag.

-R *roundup-size*
Round the image up to *roundup-size*. *roundup-size* should be a multiple of the file system block

size. This option only applies to the **ffs** file system type.

-S *sector-size*
Set the file system sector size to *sector-size*. Defaults to 512.

-s *image-size*
Set the size of the file system image to *image-size*. This is equivalent to setting both the minimum (**-M**) and the maximum (**-m**) sizes to the same value. For **ffs** and **msdos** the *image-size* does not include the *offset*. *offset* is not included in that size.

-T *timestamp*
Specify a timestamp to be set for all filesystem files and directories created so that repeatable builds are possible. The *timestamp* can be a *pathname*, where the timestamps are derived from that file, or an integer value interpreted as the number of seconds from the Epoch. Note that timestamps specified in an mtree(5) spec file, override the default timestamp.

-t *fs-type*
Create an *fs-type* file system image. The following file system types are supported:

ffs BSD fast file system (default).

cd9660 ISO 9660 file system.

msdos FAT12, FAT16, or FAT32 file system.

zfs ZFS pool containing one or more file systems.

-x Exclude file system nodes not explicitly listed in the specfile.

-Z Create a sparse file for **ffs**. This is useful for virtual machine images.

Where sizes are specified, a decimal number of bytes is expected. Two or more numbers may be separated by an "x" to indicate a product. Each number may have one of the following optional suffixes:

b Block; multiply by 512
k Kibi; multiply by 1024 (1 KiB)
m Mebi; multiply by 1048576 (1 MiB)
g Gibi; multiply by 1073741824 (1 GiB)
t Tebi; multiply by 1099511627776 (1 TiB)
w Word; multiply by the number of bytes in an integer

FFS-specific options

ffs images have ffs-specific optional parameters that may be provided. Each of the options consists of a keyword, an equal sign ('='), and a value. The following keywords are supported:

avgfilesize	Expected average file size.
avgfpdir	Expected number of files per directory.
bsize	Block size.
density	Bytes per inode. If unset, will allocate the minimum number of inodes to represent the filesystem if no free space has been requested (free blocks or minimum size set); otherwise the larger of the newfs defaults or what is required by the free inode parameters if set.
fsize	Fragment size.
label	Label name of the image.
maxbpg	Maximum blocks per file in a cylinder group.
minfree	Minimum % free.
optimization	Optimization preference; one of 'space' or 'time'.
extent	Maximum extent size.
maxbpcg	Maximum total number of blocks in a cylinder group.
version	UFS version. 1 for FFS (default), 2 for UFS2.
softupdates	0 for disable (default), 1 for enable

CD9660-specific options

cd9660 images have ISO9660-specific optional parameters that may be provided. The arguments consist of a keyword and, optionally, an equal sign ('='), and a value. The following keywords are supported:

allow-deep-trees	Allow the directory structure to exceed the maximum specified in the spec.
allow-illegal-chars	Allow illegal characters in filenames. This option is not implemented.
allow-lowercase	Allow lowercase characters in filenames. This option is not implemented.
allow-max-name	Allow 37 instead of 33 characters for filenames by omitting the version id.
allow-multidot	Allow multiple dots in a filename.
applicationid	Application ID of the image.
archimedes	Use the 'ARCHIMEDES' extension to encode RISC OS metadata.
bootimagedir	Boot image directory. This option is not implemented.
chrp-boot	Write an MBR partition table to the image to allow older CHRP hardware to boot.
boot-load-segment	Set load segment for the boot image.
bootimage	Filename of a boot image in the format "sysid;filename", where "sysid" is one of 'efi', 'i386', 'mac68k', 'macppc', or 'powerpc'.
generic-bootimage	Load a generic boot image into the first 32K of the cd9660 image.

hard-disk-boot Boot image is a hard disk image.
isolevel An integer representing the ISO 9660 interchange level where "level" is either '1' or '2'. "level" '3' is not implemented.
keep-bad-images Do not discard images whose write was aborted due to an error. For debugging purposes.
label Label name of the image.
no-boot Boot image is not bootable.
no-emul-boot Boot image is a "no emulation" ElTorito image.
no-trailing-padding Do not pad the image (apparently Linux needs the padding).
omit-trailing-period Omit trailing periods in filenames.
platformid Set platform ID of section header entry of the boot image.
preparer Preparer ID of the image.
publisher Publisher ID of the image.
rockridge Use RockRidge extensions (for longer filenames, etc.).
verbose Turns on verbose output.
volumeid Volume set identifier of the image.

msdos-specific options

msdos images have MS-DOS-specific optional parameters that may be provided. The arguments consist of a keyword, an equal sign ('='), and a value. The following keywords are supported (see newfs_msdos(8) for more details):

backup_sector Location of the backup boot sector.
block_size Block size.
bootstrap Bootstrap file.
bytes_per_sector Bytes per sector.
create_size Create file size.
directory_entries Directory entries.
drive_heads Drive heads.
fat_type FAT type (12, 16, or 32).
floppy Preset drive parameters for standard format floppy disks (160, 180, 320, 360, 640, 720, 1200, 1232, 1440, or 2880).
hidden_sectors Hidden sectors.
info_sector Location of the info sector.
media_descriptor Media descriptor.
num_FAT Number of FATs.
OEM_string OEM string.
offset Offset in device. This option will be ignored if **-O** is set to a positive number.
reserved_sectors Reserved sectors.

sectors_per_cluster Sectors per cluster.
sectors_per_fat Sectors per FAT.
sectors_per_track Sectors per track.
size File System size.
volume_id Volume ID.
volume_label Volume Label.

zfs-specific options

Note: ZFS support is currently considered experimental. Do not use it for anything critical.

The image created by **makefs** contains a ZFS pool with a single vdev of type 'disk'. The root dataset is always created implicitly and contains the entire input directory tree unless additional datasets are specified using the options described below.

The arguments consist of a keyword, an equal sign ('='), and a value. The following keywords are supported:

ashift — The base-2 logarithm of the minimum block size. Typical values are 9 (512B blocks) and 12 (4KB blocks). The default value is 12.

bootfs — The name of the bootable dataset for the pool. Specifying this option causes the 'bootfs' property to be set in the created pool.

mssize — The size of metaslabs in the created pool. By default, **makefs** allocates large (up to 512MB) metaslabs with the expectation that the image will be auto-expanded upon first use. This option allows the default heuristic to be overridden.

poolname — The name of the ZFS pool. This option must be specified.

rootpath — An implicit path prefix added to dataset mountpoints. By default it is */<poolname>*. For creating bootable pools, the *rootpath* should be set to */*. At least one dataset must have a mountpoint equal to *rootpath*.

fs — Create an additional dataset. This option may be specified multiple times. The argument value must be of the form *<dataset>[;<prop1=v1>[;<prop2=v2>[;...]]]*, where *dataset* is the name of the dataset and must belong to the pool's namespace. For example, with a pool name of 'test' all dataset names must be prefixed by 'test/'. A dataset must exist at each level of the pool's namespace. For example, to create 'test/foo/bar', 'test/foo' must be created as well.

The dataset mountpoints determine how the datasets are populated with files from the staged directory tree. Conceptually, all datasets are mounted before any are populated with files. The root of the staged directory tree is mapped

to *rootpath*.

Dataset properties, as described in zfsprops(8), may be specified following the dataset name. The following properties may be set for a dataset:

atime
canmount
exec
mountpoint
setuid

SEE ALSO

mtree(5), mtree(8), newfs(8), zfsconcepts(8), zfsprops(8), zpoolprops(8)

HISTORY

The **makefs** utility appeared in NetBSD 1.6. It was ported to FreeBSD and first appeared in FreeBSD 8.0.

AUTHORS

Luke Mewburn <*lukem@NetBSD.org*> (original program),
Daniel Watt,
Walter Deignan,
Ryan Gabrys,
Alan Perez-Rathke,
Ram Vedam (cd9660 support),
Christos Zoulas (msdos support),
Mark Johnston (zfs support).