NAME

mdconfig - create and control memory disks

SYNOPSIS

mdconfig -a -t *type* [**-n**] [**-o** [**no**]*option*] ... [**-f** *file*] [**-s** *size*] [**-S** *sectorsize*] [**-u** *unit*] [**-x** *sectors/track*] [**-y** *heads/cylinder*] [**-L** *label*]
mdconfig -d -u *unit* [**-o** [**no**]*force*]
mdconfig -r -u *unit* **-s** *size* [**-o** [**no**]*force*]
mdconfig -l [**-n**] [**-v**] [**-f** *file*] [**-u** *unit*]
mdconfig *file*

DESCRIPTION

The **mdconfig** utility creates and controls md(4) devices.

Options indicate an action to be performed:

-a Attach a memory disk. This will configure and attach a memory disk with the parameters specified and attach it to the system. If the **-u** *unit* option is not provided, the newly created device name will be printed on stdout.

-d Detach a memory disk from the system and release all resources.

-r Resize a memory disk.

-t *type*
Select the type of the memory disk.

malloc Storage for this type of memory disk is allocated with malloc(9). This limits the size to the malloc bucket limit in the kernel. If the **-o reserve** option is not set, creating and filling a large malloc-backed memory disk is a very easy way to panic the system.

vnode A file specified with **-f** *file* becomes the backing store for this memory disk.

swap Storage for this type of memory disk is allocated from buffer memory. Pages get pushed out to swap when the system is under memory pressure, otherwise they stay in the operating memory. Using **swap** backing is generally preferred instead of using **malloc** backing.

null Bitsink; all writes do nothing, all reads return zeroes.

-f *file* Filename to use for the vnode type memory disk. The **-a** and **-t vnode** options are implied if not specified.

-l List configured devices. If given with **-u**, display details about that particular device. If given with **-f** *file*, display md(4) device names of which *file* is used as the backing store. If both of **-u** and **-f** options are specified, display devices which match the two conditions. If the **-v** option is specified, show all details.

-n When printing md(4) device names, print only the unit number without the md(4) prefix.

-s *size*

Size of the memory disk. *Size* is the number of 512 byte sectors unless suffixed with a **b**, **k**, **m**, **g**, **t**, or **p** which denotes byte, kilobyte, megabyte, gigabyte, terabyte and petabyte respectively. When used without the **-r** option, the **-a** and **-t swap** options are implied if not specified.

-S *sectorsize*

Sectorsize to use for the memory disk, in bytes.

-x *sectors/track*

See the description of the **-y** option below.

-y *heads/cylinder*

For **malloc** or **vnode** backed devices, the **-x** and **-y** options can be used to specify a synthetic geometry. This is useful for constructing bootable images for later download to other devices.

-L *label*

Associate a label (arbitrary string) with the new memory disk. The label can then be inspected with

mdconfig -l -v

-o [**no**]*option*

Set or reset options.

[**no**]**async**

For **vnode** backed devices: avoid IO_SYNC for increased performance but at the risk of deadlocking the entire kernel.

[**no**]**cache**

For **vnode** backed devices: enable/disable caching of data in system caches. The default

is to not cache.

Accesses via the device are converted to accesses via the vnode. The caching policy for the vnode is used initially. This is normally to cache. This caching policy is retained if the **cache** option is used. Otherwise, caching is limited by releasing data from caches soon after each access. The release has the same semantics as the POSIX_FADV_DONTNEED feature of posix_fadvise(2). The result is that with normal (non-zfs) caching, buffers are released from the buffer cache soon after they are constructed, but their data is kept in the page cache at lower priority.

The **cache** option tends to waste memory by giving unwanted double caching, but it saves time if there is memory to spare.

[**no**]**reserve**
Allocate and reserve all needed storage from the start, rather than as needed.

[**no**]**cluster**
Enable clustering on this disk.

[**no**]**compress**
Enable/disable compression features to reduce memory usage.

[**no**]**force**
Disable/enable extra sanity checks to prevent the user from doing something that might adversely affect the system. This can be used with the **-d** flag to forcibly destroy an md(4) disk that is still in use.

[**no**]**mustdealloc**
For **vnode** backed devices: detect whether hole-punching is supported by the underlying file system. If the file system supports hole-punching, then to handle a BIO_DELETE request, some or all of the request's operation range may be turned into a hole in the file used for backing store. Any parts which are not turned into holes are zero-filled in the file. If the file system does not support hole-punching, BIO_DELETE requests to the device are not handled and will fail with EOPNOTSUPP.

When **mustdealloc** is not specified or [**no**]**mustdealloc** is specified, for a BIO_DELETE request, if the file system supports hole-punching, some or all of the request's operation range may be turned into a hole in the file used for backing store. Any parts which are not turned into holes are zero-filled in the file. If the file system of the vnode type memory disk does not support hole-punching, the request's operation range is zero-filled

in the file.

[**no**]**readonly**
Enable/disable readonly mode.

[**no**]**verify**
For **vnode** backed devices: enable/disable requesting verification of the file used for backing store. The type of verification depends on which security features are available. One example of verification is testing file integrity with checksums or cryptographic signatures.

-u *unit*
Request a specific unit number or device name for the md(4) device instead of automatic allocation. If a device name is specified, it must start with "md" followed by the unit number.

The last form, **mdconfig** *file*, is provided for convenience as an abbreviation of **mdconfig -a -t vnode -f** *file*.

EXAMPLES

Create a disk with */tmp/boot.flp* as backing storage. The name of the allocated unit will be printed on stdout, such as "md0":

```
mdconfig /tmp/boot.flp
```

Create a 1 gigabyte swap backed memory disk named "md3":

```
mdconfig -s 1g -u md3
```

Detach and free all resources used by */dev/md3*:

```
mdconfig -du md3
```

Show detailed information on current memory disks:

```
mdconfig -lv
```

Resize the "md3" memory disk to 2 gigabytes:

```
mdconfig -rs 2g -u md3
```

Create a 1 gigabyte swap backed disk, initialize an ffs(7) file system on it, and mount it on */tmp*:

```
mdconfig -s 1g -u md10
newfs -U /dev/md10
mount /dev/md10 /tmp
chmod 1777 /tmp
```

Create a memory disk out of an ISO 9660 CD image file, using the first available md(4) device, and then mount it:

```
mount -t cd9660 /dev/‘mdconfig -f cdimage.iso‘ /mnt
```

Create a file-backed device from a hard disk image that begins with 512K of raw header information. gnop(8) is used to skip over the header information, positioning *md1.nop* to the start of the filesystem in the image.

```
mdconfig -u md1 -f diskimage.img
gnop create -o 512K md1
mount /dev/md1.nop /mnt
```

SEE ALSO

fpathconf(2), fspacectl(2), open(2), md(4), ffs(7), gpart(8), mdmfs(8), malloc(9), vn_deallocate(9)

HISTORY

The **mdconfig** utility first appeared in FreeBSD 5.0 as a cleaner replacement for the vn kernel module and the vnconfig utility combo.

AUTHORS

The **mdconfig** utility was written by Poul-Henning Kamp <*phk@FreeBSD.org*>.