NAME

dmsetup — low level logical volume management

SYNOPSIS

dmsetup clear *device_name*
dmsetup create *device_name* [**-u** *uuid*] [**-n**|**--notable**|**--table** *<table>*|
    table_file] [{**--addnodeoncreate**|**--addnodeonresume**}] [**--readahead** [*+*]*<sectors>*|*auto*|*none*]
dmsetup deps [**-o** *options*] [*device_name*]
dmsetup help [**-c**|**-C**|**--columns**]
dmsetup info [*device_name*]
dmsetup info -c|**-C**|**--columns** [**--noheadings**] [**--separator** *separator*]
    [**-o** *fields*] [**-O**|**--sort** *sort_fields*] [**-S**|**--select** *Selection*] [*device_name*]
dmsetup load *device_name* [**--table** *<table>*|*table_file*]
dmsetup ls [**--target** *target_type*] [**--exec** *command*] [**--tree**]
    [**-o** *options*]
dmsetup message *device_name sector message*
dmsetup mknodes [*device_name*]
dmsetup mangle [*device_name*]
dmsetup reload *device_name* [**--table** *<table>*|*table_file*]
dmsetup wipe_table *device_name*
dmsetup remove [**-f**|**--force**] [**--retry**] [**--deferred**] *device_name*
dmsetup remove_all [**-f**|**--force**] [**--deferred**]
dmsetup rename *device_name new_name*
dmsetup rename *device_name* **--setuuid** *uuid*
dmsetup resume *device_name* [{**--addnodeoncreate**|**--addnodeonresume**}]
    [**--readahead** [*+*]*<sectors>*|*auto*|*none*]
dmsetup setgeometry *device_name cyl head sect start*
dmsetup splitname *device_name* [*subsystem*]
dmsetup status [**--target** *target_type*] [**--noflush**] [*device_name*]
dmsetup suspend [**--nolockfs**] [**--noflush**] *device_name*
dmsetup table [**--target** *target_type*] [**--showkeys**] [*device_name*]
dmsetup targets
dmsetup udevcomplete *cookie*
dmsetup udevcomplete_all [*age_in_minutes*]
dmsetup udevcookies
dmsetup udevcreatecookie
dmsetup udevflags *cookie*
dmsetup udevreleasecookie [*cookie*]
dmsetup version
dmsetup wait [**--noflush**] *device_name* [*event_nr*]

devmap_name *major minor*
devmap_name *major:minor*

DESCRIPTION

dmsetup manages logical devices that use the device-mapper driver. Devices are created by loading a table that specifies a target for each sector (512 bytes) in the logical device.

The first argument to dmsetup is a command. The second argument is the logical device name or uuid.

Invoking the command as **devmap_name** is equivalent to
dmsetup info -c --noheadings -j *major* **-m** *minor***.**

OPTIONS

--addnodeoncreate

Ensure /dev/mapper node exists after dmsetup create.

--addnodeonresume

Ensure /dev/mapper node exists after dmsetup resume (default with udev).

--checks

Perform additional checks on the operations requested and report potential problems. Useful when debugging scripts. In some cases these checks may slow down operations noticeably.

-c|-C|--columns

Display output in columns rather than as Field: Value lines.

-h|--help

Outputs a summary of the commands available, optionally including the list of report fields (synonym with **help** command).

--inactive

When returning any table information from the kernel report on the inactive table instead of the live table. Requires kernel driver version 4.16.0 or above.

--manglename {*none|hex|auto*}

Mangle any character not on a whitelist using mangling_mode when processing device-mapper device names and UUIDs. The names and UUIDs are mangled on input and unmangled on output where the mangling mode is one of: *none* (no mangling), *hex* (always do the mangling) and *auto* (only do the mangling if not mangled yet, do nothing if already mangled, error on mixed) Default mode is *auto*. Character whitelist: 0-9, A-Z, a-z, #+-.:=@_. This whitelist is also supported by udev. Any character not on a whitelist is replaced with its hex value (two digits) prefixed by x. Mangling mode could be also set through **DM_DEFAULT_NAME_MANGLING_MODE** environment variable.

-j|--major *major*

Specify the major number.

-m|--minor *minor*

Specify the minor number.

-n|--notable

When creating a device, don't load any table.

--noheadings

Suppress the headings line when using columnar output.

--noopencount

Tell the kernel not to supply the open reference count for the device.

--noudevrules

Do not allow udev to manage nodes for devices in device-mapper directory.

--noudevsync

Do not synchronise with udev when creating, renaming or removing devices.

-o|--options

Specify which fields to display.

--readahead [*+*]*<sectors>|auto|none*

Specify read ahead size in units of sectors. The default value is *auto* which allows the kernel to choose a suitable value automatically. The *+* prefix lets you specify a minimum value which will not be used if it is smaller than the value chosen by the kernel. The value *none* is equivalent to specifying zero.

-r|--readonly
> Set the table being loaded read-only.

-S|--select *Selection*
> Display only rows that match Selection criteria. All rows are displayed with the additional
> selected column (-o selected) showing 1 if the row matches the Selection and 0 otherwise.
> The Selection criteria are defined by specifying column names and their valid values while
> making use of supported comparison operators. As a quick help and to see full list of col-
> umn names that can be used in Selection and the set of supported selection operators,
> check the output of **dmsetup info -c -S help** command.

--table *<table>*
> Specify a one-line table directly on the command line.

--udevcookie *cookie*
> Use cookie for udev synchronisation. Note: Same cookie should be used for same type of
> operations i.e. creation of multiple different devices. It's not adviced to combine different
> operations on the single device.

-u|--uuid
> Specify the uuid.

-y|--yes
> Answer yes to all prompts automatically.

-v|--verbose [**-v|--verbose**]
> Produce additional output.

--verifyudev
> If udev synchronisation is enabled, verify that udev operations get performed correctly
> and try to fix up the device nodes afterwards if not.

--version
> Display the library and kernel driver version.

COMMANDS

clear   *device_name*
> Destroys the table in the inactive table slot for device_name.

create *device_name* [**-u** *uuid*] [**-n|--notable|--table** *<table>|table_file*] [{**--addnodeoncre-
ate|--addnodeonresume**}] **--readahead** [*+*]*<sectors>|auto|none*]
> Creates a device with the given name. If table_file or <table> is supplied, the table is
> loaded and made live. Otherwise a table is read from standard input unless **--notable** is
> used. The optional uuid can be used in place of device_name in subsequent dmsetup
> commands. If successful a device will appear as /dev/mapper/<device-name>. See below
> for information on the table format.

deps   [**-o** *options*] [*device_name*]
> Outputs a list of devices referenced by the live table for the specified device. Device
> names on output can be customised by following options: devno (major and minor pair,
> used by default), blkdevname (block device name), devname (map name for device-map-
> per devices, equal to blkdevname otherwise).

help   [**-c|-C|--columns**]
> Outputs a summary of the commands available, optionally including the list of report
> fields.

info   [*device_name*]
> Outputs some brief information about the device in the form:
>> State: SUSPENDED|ACTIVE, READ-ONLY Tables present: LIVE and/or
>> INACTIVE Open reference count Last event sequence number (used by **wait**)
>> Major and minor device number Number of targets in the live table UUID

info   **-c|-C|--columns** [**--noheadings**] [**--separator** *separator*] [**-o** *fields*] [**-O|--sort** *sort_fields*]
          [*device_name*]
          Output you can customise. Fields are comma-separated and chosen from the following
          list: name, major, minor, attr, open, segments, events, uuid. Attributes are: (L)ive,
          (I)nactive, (s)uspended, (r)ead-only, read-(w)rite. Precede the list with '+' to append to
          the default selection of columns instead of replacing it. Precede any sort_field with - for a
          reverse sort on that column.

ls   [**--target** *target_type*] [**--exec** *command*] [**--tree**] [**-o** *options*]
          List device names. Optionally only list devices that have at least one target of the speci-
          fied type. Optionally execute a command for each device. The device name is appended to
          the supplied command. Device names on output can be customised by following options:
          devno (major and minor pair, used by default), blkdevname (block device name), dev-
          name (map name for device-mapper devices, equal to blkdevname otherwise). **--tree** dis-
          plays dependencies between devices as a tree. It accepts a comma-separate list of options.
          Some specify the information displayed against each node: device/nodevice; blkdevname;
          active, open, rw, uuid. Others specify how the tree is displayed: ascii, utf, vt100; com-
          pact, inverted, notrunc.

load|reload *device_name* [**--table** *<table>|table_file*]
          Loads <table> or table_file into the inactive table slot for device_name. If neither is sup-
          plied, reads a table from standard input.

wipe_table *device_name*
          Wait for any I/O in-flight through the device to complete, then replace the table with a
          new table that fails any new I/O sent to the device. If successful, this should release any
          devices held open by the device's table(s).

message *device_name sector message*
          Send message to target. If sector not needed use 0.

mknodes [*device_name*]
          Ensure that the node in /dev/mapper for device_name is correct. If no device_name is
          supplied, ensure that all nodes in /dev/mapper correspond to mapped devices currently
          loaded by the device-mapper kernel driver, adding, changing or removing nodes as neces-
          sary.

mangle [*device_name*]
          Ensure existing device-mapper device name and UUID is in the correct mangled form
          containing only whitelisted characters (supported by udev) and do a rename if necessary.
          Any character not on the whitelist will be mangled based on the **--manglename** setting.
          Automatic rename works only for device names and not for device UUIDs because the
          kernel does not allow changing the UUID of active devices. Any incorrect UUIDs are
          reported only and they must be manually corrected by deactivating the device first and
          then reactivating it with proper mangling mode used (see also **--manglename**).

remove [**-f|--force**] [**--retry**] [**--deferred**] *device_name*
          Removes a device. It will no longer be visible to dmsetup. Open devices cannot be
          removed, but adding **--force** will replace the table with one that fails all I/O. **--deferred**
          will enable deferred removal of open devices - the device will be removed when the last
          user closes it. The deferred removal feature is supported since version 4.27.0 of the device-
          mapper driver available in upstream kernel version 3.13. (Use **dmsetup version** to check
          this.) If an attempt to remove a device fails, perhaps because a process run from a quick
          udev rule temporarily opened the device, the **--retry** option will cause the operation to
          be retried for a few seconds before failing. Do NOT combine **--force** and **--udevcookie**,
          as udev may start to process udev rules in the middle of error target replacement and
          result in nondeterministic result.

remove_all [-f|--force] [--deferred]
> Attempts to remove all device definitions i.e. reset the driver. This also runs **mknodes** afterwards. Use with care! Open devices cannot be removed, but adding **--force** will replace the table with one that fails all I/O. **--deferred** will enable deferred removal of open devices - the device will be removed when the last user closes it. The deferred removal feature is supported since version 4.27.0 of the device-mapper driver available in upstream kernel version 3.13.

rename *device_name new_name*
> Renames a device.

rename *device_name* **--setuuid** *uuid*
> Sets the uuid of a device that was created without a uuid. After a uuid has been set it cannot be changed.

resume
> *device_name* [{**--addnodeoncreate**|**--addnodeonresume**}] [**--readahead** [+]<*sectors*>|*auto*|*none*]
> Un-suspends a device. If an inactive table has been loaded, it becomes live. Postponed I/O then gets re-queued for processing.

setgeometry *device_name cyl head sect start*
> Sets the device geometry to C/H/S.

splitname *device_name* [*subsystem*]
> Splits given device name into subsystem constituents. The default subsystem is LVM. LVM currently generates device names by concatenating the names of the Volume Group, Logical Volume and any internal Layer with a hyphen as separator. Any hyphens within the names are doubled to escape them. The precise encoding might change without notice in any future release, so we recommend you always decode using the current version of this command.

status [**--target** *target_type*] [**--noflush**] [*device_name*]
> Outputs status information for each of the device's targets. With **--target**, only information relating to the specified target type any is displayed. With **--noflush**, the thin target (from version 1.3.0) doesn't commit any outstanding changes to disk before reporting its statistics.

suspend [**--nolockfs**] [**--noflush**] *device_name*
> Suspends a device. Any I/O that has already been mapped by the device but has not yet completed will be flushed. Any further I/O to that device will be postponed for as long as the device is suspended. If there's a filesystem on the device which supports the operation, an attempt will be made to sync it first unless **--nolockfs** is specified. Some targets such as recent (October 2006) versions of multipath may support the **--noflush** option. This lets outstanding I/O that has not yet reached the device to remain unflushed.

table [**--target** *target_type*] [**--showkeys**] [*device_name*]
> Outputs the current table for the device in a format that can be fed back in using the create or load commands. With **--target**, only information relating to the specified target type is displayed. Encryption keys are suppressed in the table output for the crypt target unless the **--showkeys** parameter is supplied.

targets
> Displays the names and versions of the currently-loaded targets.

udevcomplete *cookie*
> Wake any processes that are waiting for udev to complete processing the specified cookie.

udevcomplete_all [*age_in_minutes*]
> Remove all cookies older than the specified number of minutes. Any process waiting on a cookie will be resumed immediately.

udevcookies

> List all existing cookies. Cookies are system-wide semaphores with keys prefixed by two predefined bytes (0x0D4D).

udevcreatecookie

> Creates a new cookie to synchronize actions with udev processing. The output is a cookie value. Normally we don't need to create cookies since dmsetup creates and destroys them for each action automatically. However, we can generate one explicitly to group several actions together and use only one cookie instead. We can define a cookie to use for each relevant command by using **--udevcookie** option. Alternatively, we can export this value into the environment of the dmsetup process as **DM_UDEV_COOKIE** variable and it will be used automatically with all subsequent commands until it is unset. Invoking this command will create system-wide semaphore that needs to be cleaned up explicitly by calling udevreleasecookie command.

udevflags *cookie*

> Parses given cookie value and extracts any udev control flags encoded. The output is in environment key format that is suitable for use in udev rules. If the flag has its symbolic name assigned then the output is DM_UDEV_FLAG_<flag_name>='1', DM_UDEV_FLAG<flag_position>='1' otherwise. Subsystem udev flags don't have symbolic names assigned and these ones are always reported as DM_SUBSYSTEM_UDEV_FLAG<flag_position>='1'. There are 16 udev flags altogether.

udevreleasecookie [*cookie*]

> Waits for all pending udev processing bound to given cookie value and clean up the cookie with underlying semaphore. If the cookie is not given directly, the command will try to use a value defined by **DM_UDEV_COOKIE** environment variable.

version

> Outputs version information.

wait   [**--noflush**] *device_name* [*event_nr*]

> Sleeps until the event counter for device_name exceeds event_nr. Use **-v** to see the event number returned. To wait until the next event is triggered, use **info** to find the last event number. With **--noflush**, the thin target (from version 1.3.0) doesn't commit any outstanding changes to disk before reporting its statistics.

TABLE FORMAT

Each line of the table specifies a single target and is of the form:

logical_start_sector num_sectors **target_type** <*target_args*>

Simple target types and <target_args> include:

linear *destination_device start_sector*

> The traditional linear mapping.

striped *num_stripes chunk_size* [*destination start_sector*]+

> Creates a striped area.
> e.g. striped 2 32 /dev/hda1 0 /dev/hdb1 0 will map the first chunk (16k) as follows:
>> LV chunk 1 -> hda1, chunk 1 LV chunk 2 -> hdb1, chunk 1 LV chunk 3 -> hda1, chunk 2 LV chunk 4 -> hdb1, chunk 2 etc.

error

> Errors any I/O that goes to this area. Useful for testing or for creating devices with holes in them.

zero

> Returns blocks of zeroes on reads. Any data written is discarded silently. This is a block-device equivalent of the /dev/zero character-device data sink described in **null(4)**.

More complex targets include:

crypt
>   Transparent encryption of block devices using the kernel crypto API.

delay
>   Delays reads and/or writes to different devices. Useful for testing.

flakey
>   Creates a similar mapping to the linear target but exhibits unreliable behaviour periodically. Useful for simulating failing devices when testing.

mirror
>   Mirrors data across two or more devices.

multipath
>   Mediates access through multiple paths to the same device.

raid
>   Offers an interface to the kernel's software raid driver, md.

snapshot
>   Supports snapshots of devices.

To find out more about the various targets and their table formats and status lines, please read the files in the Documentation/device-mapper directory in the kernel source tree. (Your distribution might include a copy of this information in the documentation directory for the device-mapper package.)

EXAMPLES
>   # A table to join two disks together
>   0 1028160 linear /dev/hda 0
>   1028160 3903762 linear /dev/hdb 0
>
>   # A table to stripe across the two disks,
>   # and add the spare space from
>   # hdb to the back of the volume
>
>   0 2056320 striped 2 32 /dev/hda 0 /dev/hdb 0
>   2056320 2875602 linear /dev/hdb 1028160

ENVIRONMENT VARIABLES
DM_DEV_DIR
>   The device directory name. Defaults to /dev and must be an absolute path.

DM_UDEV_COOKIE
>   A cookie to use for all relevant commands to synchronize with udev processing. It is an alternative to using **--udevcookie** option.

DM_DEFAULT_NAME_MANGLING_MODE
>   A default mangling mode. Defaults to *auto* and it is an alternative to using **--mangle-name** option.

AUTHORS
>   Original version: Joe Thornber (thornber@redhat.com)

SEE ALSO
>   LVM2 resource page https://www.sourceware.org/lvm2/
>   Device-mapper resource page: http://sources.redhat.com/dm/