NAME

dotlockfile - Utility to manage lockfiles

SYNOPSIS

/usr/bin/dotlockfile [**-l** [**-r** *retries*] **|-u|-t|-c**] [**-p**] [**-m|lockfile**]

DESCRIPTION

dotlockfile is a command line utility to reliably create, test and remove lockfiles. It creates lockfiles *reliably* on local and NFS filesystems, because the crucial steps of testing for a preexisting lockfile and creating it are performed *atomically* by a *single* call to link(2). Manpage *lockfile_create(3)* describes the used algorithm.

dotlockfile is installed with attribute **SETGID** *mail* and thus can also be used to lock and unlock mailboxes *even* if the mailspool directory is only writable by group mail.

The name **dotlockfile** comes from the way mailboxes are locked for updates on a lot of UNIX systems. A lockfile is created with the same filename as the mailbox but with the string *.lock* appended.

The names **dotlock** and **lockfile** were already taken - hence the name **dotlockfile** *:)*.

OPTIONS

-l   Create a lockfile if no preexisting valid lockfile is found, else wait and retry according to option **-r**. This option is the default.

A lockfile is treated as valid,
- if it holds the *process-id* of a running process,
- or if it does not hold any *process-id* and has been touched less than 5 minutes ago (timestamp is younger than 5 minutes).

-r retries   The number of times **dotlockfile** retries to acquire the lock if it failed the first time before giving up. The initial sleep after failing to acquire the lock is 5 seconds. After each retry the sleep intervall is increased incrementally by 5 seconds up to a maximum sleep of 60 seconds between tries. The default number of retries is 5. To try only once, use **-r 0**. To try indefinitely, use **-r -1**.

-u   Remove a lockfile.

-t   Touch an existing lockfile (update the timestamp). Useful for lockfiles on NFS filesystems. For lockfiles on local filesystems the **-p** option is preferable.

-c   For debugging only: Check for the existence of a *valid* lockfile.

Note: Testing for a preexisting lockfile and writing of the lockfile *must* be done by the *same* **dotlockfile -l** or **dotlockfile -m** command, else the lockfile creation cannot be reliable.

-p   Write the *process-id* of the calling process into the lockfile. Also when testing for an existing lockfile, check the contents for the *process-id* of a running process to verify if the lockfile is still valid. Obviously useful only for lockfiles on local filesystems.

-m   Lock or unlock the current users mailbox. The path to the mailbox is the default system mailspool directory (usually */var/mail*) with the username as gotten from *getpwuid*() appended. If the environment variable *MAIL* is set, that is used instead. Then the string *.lock* is appended to get the name of the actual lockfile.

lockfile The lockfile to be created or removed. Must not be specified, if the **-m** option is in effect.

RETURN VALUE

Zero on success, and non-zero on failure. For the **-c** option, sucess means that a valid lockfile is already present. When locking (the default, or with the **-l** option) **dotlockfile** returns the same values as the library function *lockfile_create(3)*. Unlocking a non-existant lockfile is not an error.

NOTES

The lockfile is created exactly as named on the command line. The extension *.lock* is *not* automatically appended.

This utility is a lot like the *lockfile(1)* utility included with *procmail*, and the *mutt_dotlock(1)* utility included with *mutt*. However the command-line arguments differ, and so does the return status. It is believed, that *dotlockfile* is the most flexible implementation, since it automatically detects when it needs to use priviliges to lock a mailbox, and does it safely.

The above mentioned *lockfile_create(3)* manpage is present in the *liblockfile-dev* package.

BUGS

None known.

SEE ALSO

lockfile_create(3), maillock(3)

AUTHOR

Miquel van Smoorenburg, miquels@cistron.nl