NAME

getutent, getutid, getutline, pututline, setutent, endutent, utmpname − access utmp file entry

SYNOPSIS

```
#include <utmp.h>

struct utmp *getutent()

struct utmp *getutid(id)
struct utmp *id ;

struct utmp *getutline(line)
struct utmp *line ;

pututline(utmp)
struct utmp *utmp ;

setutent()

endutent()

utmpname(file)
char *file ;
```

DESCRIPTION

Getutent, *getutid*, and *getutline* each return a pointer to a structure of the following type:

```
/*              @(#)utmp.h     3.2             */

/*              <sys/types.h> must be included.                                   */

#define         UTMP_FILE     "/etc/utmp"
#define         WTMP_FILE     "/etc/wtmp"

struct utmp
 {
                char ut_user[8] ;               /* User login name */
                char ut_id[2] ;                 /* /etc/lines id(usually line #) */
                char ut_line[12] ;              /* device name (console, lnxx) */
                short ut_pid ;                  /* process id */
                struct exit_status
                 {
                   char e_termination ;         /* Process termination status */
                   char e_exit ;                /* Process exit status */
                 }
                ut_exit ;                       /* The exit status of a process
                                                 * marked as DEAD_PROCESS.
                                                 */
                short ut_type ;                 /* type of entry */
                time_t ut_time ;                /* time entry was made */
 } ;

/*              Definitions for ut_type                                           */

#define         EMPTY          0
#define         RUN_LVL        1
#define         BOOT_TIME      2
#define         OLD_TIME       3
#define         NEW_TIME       4
#define         INIT_PROCESS   5                /* Process spawned by "init" */
#define         LOGIN_PROCESS 6                 /* A "getty" process waiting for login */
#define         USER_PROCESS   7                /* A user process */
#define         DEAD_PROCESS   8

#define         UTMAXTYPE    DEAD_PROCESS               /* Largest legal value of ut_type */
```

```
/*          Special strings or formats used in the "ut_line" field when        */
/*          accounting for something other than a process.                    */
/*          ** Note ** each message is such that is takes exactly 11           */
/*          spaces + a null, so that it fills the "ut_line" array.             */

#define     RUNLVL_MSG  "run_level_%c"
#define     BOOT_MSG    "system_boot"
#define     OTIME_MSG   "old_time   "
#define     NTIME_MSG   "new_time   "
```

Getutent reads in the next entry from a *utmp* like file. If the file is not already open, it opens it. If it reaches the end of the file, it fails.

Getutid searches forward from the current point in the *utmp* file until it finds an entry with a *ut_type* matching *id−>ut_type* if the type specified is **RUN_LVL**, **BOOT_TIME**, **OLD_TIME**, or **NEW_TIME**. If the type specified in *id* is **INIT_PROCESS**, **LOGIN_PROCESS**, **USER_PROCESS**, or **DEAD_PROCESS**, then *getutid* will return a pointer to the first entry whose type is one of these four and whose *ut_id* field matches *id−>ut_id*. If the end of file is reached without a match, it fails.

Getutline searches forward from the current point in the utmp file until it finds an entry of the type **LOGIN_PROCESS** or **USER_PROCESS** which also has a *ut_line* string matching *line−>ut_line* string. If the end of file is reached without a match, it fails.

Pututline writes out the supplied *utmp* structure into the utmp file. It uses *getutid* to search forward for the proper place if it finds that it is not already at the proper place. It is expected that normally the user of *pututline* will have searched for the proper entry using one of the *get* routines. If so, *pututline* will not search. If *pututline* does not find a matching slot for the new entry, it will add a new entry to the end of the file.

Setutent resets the input stream to the beginning of the file. This should be done inbetween each search for a new entry if it is desired that the entire file be examined.

Endutent closes the currently open file.

Utmpname allows the user to change the name of the file examined from **/etc/utmp** to any other file. It is most often expected that this other file will be **/etc/wtmp.** If the file doesn't exist, this will not be apparent until the first attempt to reference the file is made. *Utmpname* does not open the file. It just closes the old file if it is currently open and saves the new file name.

FILES

/etc/utmp,
/etc/wtmp

SEE ALSO

utmp(5)

DIAGNOSTICS

A **NULL** pointer is returned upon failure to read, whether for permissions or having reached the end of file, or upon failure to write.

COMMENTS

The most current entry is saved in a static structure. Multiple accesses require that it be copied before further accesses are made. Each call to either *getutid* or *getutline* sees the routine examine the static structure before performing more io. If the contents of the static structure match what it is searching for, it looks no further. For this reason to use *getutline* to search for multiple occurances, it would be necessary to zero out the static after each success, or *getutline* would just return the same pointer over and over again. There is one exception to the rule about removing the structure before further reads are done. The implicit read done by *pututline* if it finds that it isn't already at the correct place in the file will not hurt the contents of the static structure returned by the *getutent*, *getutid*, or *getutline* routines, if the user has just modified those contents and passed the pointer back to *pututline*.

These routines use buffered standand io for input, but *pututline* uses an unbuffered non-standard write to avoid race conditions between processes trying to modify the *utmp* and *wtmp* files.