NAME

getpwent_r, fgetpwent_r − get passwd file entry reentrantly

LIBRARY

Standard C library (*libc*, *−lc*)

SYNOPSIS

```
#include <pwd.h>

int getpwent_r(struct passwd *restrict pwbuf,
        char buf[restrict .buflen], size_t buflen,
        struct passwd **restrict pwbufp);
int fgetpwent_r(FILE *restrict stream, struct passwd *restrict pwbuf,
        char buf[restrict .buflen], size_t buflen,
        struct passwd **restrict pwbufp);
```

Feature Test Macro Requirements for glibc (see **feature_test_macros**(7)):

getpwent_r(),
Since glibc 2.19:
_DEFAULT_SOURCE
glibc 2.19 and earlier:
_BSD_SOURCE || _SVID_SOURCE

fgetpwent_r():
Since glibc 2.19:
_DEFAULT_SOURCE
glibc 2.19 and earlier:
_SVID_SOURCE

DESCRIPTION

The functions **getpwent_r**() and **fgetpwent_r**() are the reentrant versions of **getpwent**(3) and **fgetpwent**(3). The former reads the next passwd entry from the stream initialized by **setpwent**(3). The latter reads the next passwd entry from *stream*.

The *passwd* structure is defined in <*pwd.h*> as follows:

```
struct passwd {
    char    *pw_name;       /* username */
    char    *pw_passwd;     /* user password */
    uid_t    pw_uid;        /* user ID */
    gid_t    pw_gid;        /* group ID */
    char    *pw_gecos;      /* user information */
    char    *pw_dir;        /* home directory */
    char    *pw_shell;      /* shell program */
};
```

For more information about the fields of this structure, see **passwd**(5).

The nonreentrant functions return a pointer to static storage, where this static storage contains further pointers to user name, password, gecos field, home directory and shell. The reentrant functions described here return all of that in caller-provided buffers. First of all there is the buffer *pwbuf* that can hold a *struct passwd*. And next the buffer *buf* of size *buflen* that can hold additional strings. The result of these functions, the *struct passwd* read from the stream, is stored in the provided buffer **pwbuf*, and a pointer to this *struct passwd* is returned in **pwbufp*.

RETURN VALUE

On success, these functions return 0 and **pwbufp* is a pointer to the *struct passwd*. On error, these functions return an error value and**pwb ufp* is NULL.

ERRORS

ENOENT
No more entries.

ERANGE
Insufficient buffer space supplied. Try again with larger buffer.

ATTRIBUTES

For an explanation of the terms used in this section, see **attributes**(7).

Interface	Attribute	Value
getpwent_r()	Thread safety	MT-Unsafe race:pwent locale
fgetpwent_r()	Thread safety	MT-Safe

In the above table, *pwent* in *race:pwent* signifies that if any of the functions **setpwent**(), **getpwent**(), **endpwent**(), or **getpwent_r**() are used in parallel in different threads of a program, then data races could occur.

STANDARDS

These functions are GNU extensions, done in a style resembling the POSIX version of functions like **getpwnam_r**(3). Other systems use the prototype

```
struct passwd *
getpwent_r(struct passwd *pwd, char *buf, int buflen);
```

or, better,

```
int
getpwent_r(struct passwd *pwd, char *buf, int buflen,
           FILE **pw_fp);
```

NOTES

The function **getpwent_r**() is not really reentrant since it shares the reading position in the stream with all other threads.

EXAMPLES

```
#define _GNU_SOURCE
#include <pwd.h>
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>

#define BUFLEN 4096

int
main(void)
{
    struct passwd pw;
    struct passwd *pwp;
    char buf[BUFLEN];
    int i;

    setpwent();
    while (1) {
        i = getpwent_r(&pw, buf, sizeof(buf), &pwp);
        if (i)
            break;
        printf("%s (%jd)\tHOME %s\tSHELL %s\n", pwp->pw_name,
               (intmax_t) pwp->pw_uid, pwp->pw_dir, pwp->pw_shell);
    }
```

```
            endpwent();
            exit(EXIT_SUCCESS);
        }
```

SEE ALSO

fgetpwent(3), **getpw**(3), **getpwent**(3), **getpwnam**(3), **getpwuid**(3), **putpwent**(3), **passwd**(5)