NAME

cap_getpwent, **cap_getpwnam**, **cap_getpwuid**, **cap_getpwent_r**, **cap_getpwnam_r**, **cap_getpwuid_r**, **cap_setpassent**, **cap_setpwent**, **cap_endpwent**, **cap_pwd_limit_cmds**, **cap_pwd_limit_fields**, **cap_pwd_limit_users** - library for password database operations in capability mode

LIBRARY

library "libcap_pwd"

SYNOPSIS

#include <libcasper.h>
#include <casper/cap_pwd.h>

*struct passwd **
cap_getpwent(*cap_channel_t *chan*);

*struct passwd **
cap_getpwnam(*cap_channel_t *chan, const char *login*);

*struct passwd **
cap_getpwuid(*cap_channel_t *chan, uid_t uid*);

int
cap_getpwent_r(*cap_channel_t *chan, struct passwd *pwd, char *buffer, size_t bufsize, struct passwd **result*);

int
cap_getpwnam_r(*cap_channel_t *chan, const char *name, struct passwd *pwd, char *buffer, size_t bufsize, struct passwd **result*);

int
cap_getpwuid_r(*cap_channel_t *chan, uid_t uid, struct passwd *pwd, char *buffer, size_t bufsize, struct passwd **result*);

int
cap_setpassent(*cap_channel_t *chan, int stayopen*);

void
cap_setpwent(*cap_channel_t *chan*);

void

cap_endpwent(*cap_channel_t *chan*);

int
cap_pwd_limit_cmds(*cap_channel_t *chan, const char * const *cmds, size_t ncmds*);

int
cap_pwd_limit_fields(*cap_channel_t *chan, const char * const *fields, size_t nfields*);

int
cap_pwd_limit_users(*cap_channel_t *chan, const char * const *names, size_t nnames, uid_t *uids, size_t nuids*);

DESCRIPTION

The functions **cap_getpwent**(), **cap_getpwnam**(), **cap_getpwuid**(), **cap_getpwent_r**(), **cap_getpwnam_r**(), **cap_getpwuid_r**(), **cap_setpassent**(), **cap_setpwent**(), and **cap_endpwent**() are respectively equivalent to getpwent(3), getpwnam(3), getpwuid(3), getpwent_r(3), getpwnam_r(3), getpwuid_r(3), setpassent(3), setpwent(3), and cap_endpwent(3) except that the connection to the **system.pwd** service needs to be provided.

The **cap_pwd_limit_cmds**() function limits the functions allowed in the service. The *cmds* variable can be set to getpwent, getpwnam, getpwuid, getpwent_r, getpwnam_r, getpwuid_r, setpassent, setpwent, or endpwent which will allow to use the function associated with the name. The *ncmds* variable contains the number of *cmds* provided.

The **cap_pwd_limit_fields**() function allows limit fields returned in the structure *passwd*. The *fields* variable can be set to pw_name, pw_passwd, pw_uid, pw_gid, pw_change, pw_class, pw_gecos, pw_dir, pw_shell, pw_expire or pw_fields The field which was set as the limit will be returned, while the rest of the values not set this way will have default values. The *nfields* variable contains the number of *fields* provided.

The **cap_pwd_limit_users**() function allows to limit access to users. The *names* variable allows to limit users by name and the *uids* variable by the user number. The *nnames* and *nuids* variables provide numbers of limited names and uids.

All of these functions are reentrant but not thread-safe. That is, they may be called from separate threads only with different *cap_channel_t* arguments or with synchronization.

EXAMPLES

The following example first opens a capability to casper and then uses this capability to create the **system.pwd** casper service and uses it to get a user name.

```
cap_channel_t *capcas, *cappwd;
const char *cmds[] = { "getpwuid" };
const char *fields[] = { "pw_name" };
uid_t uid[] = { 1 };
struct passwd *passwd;

/* Open capability to Casper. */
capcas = cap_init();
if (capcas == NULL)
        err(1, "Unable to contact Casper");

/* Enter capability mode sandbox. */
if (cap_enter() < 0 && errno != ENOSYS)
        err(1, "Unable to enter capability mode");

/* Use Casper capability to create capability to the system.pwd service. */
cappwd = cap_service_open(capcas, "system.pwd");
if (cappwd == NULL)
        err(1, "Unable to open system.pwd service");

/* Close Casper capability, we don’t need it anymore. */
cap_close(capcas);

/* Limit service to one single function. */
if (cap_pwd_limit_cmds(cappwd, cmds, nitems(cmds)))
                err(1, "Unable to limit access to system.pwd service");

/* Limit service to one field as we only need name of the user. */
if (cap_pwd_limit_fields(cappwd, fields, nitems(fields)))
                err(1, "Unable to limit access to system.pwd service");

/* Limit service to one uid. */
if (cap_pwd_limit_users(cappwd, NULL, 0, uid, nitems(uid)))
                err(1, "Unable to limit access to system.pwd service");

passwd = cap_getpwuid(cappwd, uid[0]);
if (passwd == NULL)
                err(1, "Unable to get name of user");

printf("UID %d is associated with name %s.\n", uid[0], passwd->pw_name);
```

```
cap_close(cappwd);
```

SEE ALSO

cap_enter(2), endpwent(3), err(3), getpwent(3), getpwent_r(3), getpwnam(3), getpwnam_r(3), getpwuid(3), getpwuid_r(3), setpassent(3), setpwent(3), capsicum(4), nv(9)

HISTORY

The **cap_pwd** service first appeared in FreeBSD 10.3.

AUTHORS

The **cap_pwd** service was implemented by Pawel Jakub Dawidek <*pawel@dawidek.net*> under sponsorship from the FreeBSD Foundation.

This manual page was written by
Mariusz Zaborski <*oshogbo@FreeBSD.org*>.