NAME

cap_clear, cap_clear_flag, cap_compare, cap_copy_ext, cap_copy_int, cap_drop_bound, cap_dup, cap_fill, cap_fill_flag, cap_free, cap_from_name, cap_from_text, cap_get_ambient, cap_get_bound, cap_get_fd, cap_get_file, cap_get_flag, cap_get_mode, cap_get_nsowner, cap_get_pid, cap_get_pid, cap_get_proc, cap_get_secbits, cap_init, cap_max_bits, cap_prctl, cap_prctlw, cap_proc_root, cap_reset_ambient, cap_set_ambient, cap_set_fd, cap_set_file, cap_set_flag, cap_setgroups, cap_set_mode, cap_set_nsowner, cap_set_proc, cap_set_secbits, cap_setuid, cap_size, cap_to_name, cap_to_text − capability data object manipulation

SYNOPSIS

```
#include <sys/capability.h>

int cap_clear(cap_t cap_p);
int cap_fill(cap_t cap_p, cap_flag_t to, cap_flag_t from);
int cap_fill_flag(cap_t cap_p, cap_flag_t to, const cap_t ref, cap_flag_t from);
int cap_clear_flag(cap_t cap_p, cap_flag_t flag);
int cap_compare(cap_t cap_a, cap_t cap_b);
ssize_t cap_copy_ext(void *ext_p, cap_t cap_p, ssize_t size);
cap_t cap_copy_int(const void *ext_p);
int cap_free(void *obj_d);
int cap_from_name(const char *name, cap_value_t *cap_p);
cap_t cap_from_text(const char *buf_p);
cap_t cap_get_fd(int fd);
cap_t cap_get_file(const char *path_p);
int cap_get_flag(cap_t cap_p, cap_value_t cap ,
        cap_flag_t flag, cap_flag_value_t *value_p);
cap_value_t cap_max_bits();

#include <sys/types.h>

cap_t cap_get_pid(pid_t pid);
cap_t cap_get_proc(void);
int cap_set_fd(int fd, cap_t caps);
int cap_set_file(const char *path_p, cap_t cap_p);
int cap_set_flag(cap_t cap_p, cap_flag_t flag, int ncap ,
        const cap_value_t *caps, cap_flag_value_t value);
int cap_set_proc(cap_t cap_p);
ssize_t cap_size(cap_t cap_p);
char *cap_to_name(cap_value_t cap);
char *cap_to_text(cap_t caps, ssize_t *length_p);
cap_t cap_get_pid(pid_t pid);
cap_t cap_init();
cap_t cap_dup(cap_t cap_p);

char *cap_proc_root(const char *root);
int cap_get_nsowner(cap_t cap_p);
int cap_set_nsowner(cap_t cap_p, uid_t rootuid);
int cap_get_bound(cap_value_t cap);
int cap_drop_bound(cap_value_t cap);
int cap_get_ambient(cap_value_t cap);
int cap_set_ambient(cap_value_t cap, cap_flag_value_t value);
int cap_reset_ambient(void);
int cap_set_mode(cap_mode_t flavor);
cap_mode_t cap_get_mode(void);
const char *cap_mode_name(cap_mode_t flavor);
```

```
unsigned cap_get_secbits();
int cap_set_secbits(unsigned bits);
int cap_prctl(long int pr_cmd, long int arg1, long int arg2, long int arg3,
        long int arg4, long int arg5);
int cap_prctlw(long int pr_cmd, long int arg1, long int arg2, long int arg3,
         long int arg4, long int arg5);
int cap_setuid(uid_t uid);
int cap_setgroups(gid_t gid, size_t ngroups, const gid_t groups[]);
```

Link with *−lcap*.

DESCRIPTION

These primary functions work on a capability state held in working storage and attempt to complete the POSIX.1e (draft) user space API for Capability based privilege.

A *cap_t* holds information about the capabilities in each of the three sets, Permitted, Inheritable, and Effective. Each capability in a set may be clear (disabled, 0) or set (enabled, 1).

These functions work with the following data types:

cap_value_t identifies a capability, such as **CAP_CHOWN**.

cap_flag_t identifies one of the three flags associated with a capability (i.e., it identifies one of the three capability sets). Valid values for this type are **CAP_EFFECTIVE**, **CAP_INHERITABLE** or **CAP_PERMITTED**.

cap_flag_value_t identifies the setting of a particular capability flag (i.e, the value of a capability in a set). Valid values for this type are **CAP_CLEAR**(0) or **CAP_SET**(1).

RETURN VALUE

The return value is generally specific to the individual function called. On failure, *errno* is set appropriately.

CONFORMING TO

These functions are as per the withdrawn POSIX.1e draft specification. The following functions are Linux extensions: **cap_clear_flag**(), **cap_drop_bound**(), **cap_fill**(), **cap_fill_flag**(), **cap_from_name**(), **cap_get_ambient**(), **cap_get_bound**(), **cap_get_mode**(), **cap_get_nsowner**(), **cap_get_secbits**(), **cap_mode_name**(), **cap_proc_root**(), **cap_prctl**(), **cap_prctlw**(), **cap_reset_ambient**(), **cap_setgroups**(), **cap_setuid**(), **cap_set_ambient**(), **cap_set_mode**(), **cap_set_nsowner**(), **cap_set_secbits**(), **cap_to_name**() and **cap_compare**().

A Linux, *IAB*, extension of Inheritable, Bounding and Ambient tuple capability vectors are also supported by **libcap**. Those functions are described in a companion man page: **cap_iab**(3). Further, for managing the complexity of launching a sub-process, **libcap** supports the abstraction: **cap_launch**(3).

In addition to the **cap_** prefixed **libcap** API, the library also provides prototypes for the Linux system calls that provide the native API for process capabilities. These prototypes are:

```
int capget(cap_user_header_t header, cap_user_data_t data);
int capset(cap_user_header_t header, const cap_user_data_t data);
```

Further, **libcap** provides a set-up function,

```
void cap_set_syscall(
    long int (*new_syscall)(long int, long int, long int, long int),
    long int (*new_syscall6)(long int,
                    long int, long int, long int,
                    long int, long int, long int));
```

which can be used to redirect its use of the **capset**() and other system calls that write kernel managed state.

This is especially useful when supporting POSIX semantics for security state. When a program is linked against **libpsx**(3) as described in that man page, this function is used to connect **libcap** to POSIX semantics system calls.

REPORTING BUGS

The **libcap** library is distributed from https://sites.google.com/site/fullycapable/ where the release notes may already cover recent issues. Please report newly discovered bugs via:

https://bugzilla.kernel.org/buglist.cgi?component=libcap&list_id=1090757

SEE ALSO

cap_clear(3), **cap_copy_ext**(3), **cap_from_text**(3), **cap_get_file**(3), **cap_get_proc**(3), **cap_iab**(3), **cap_init**(3), **cap_launch**(3), **capabilities**(7), **getpid**(2), **capsh**(1), **captree**(8), **getcap**(8), **getpcaps**(8), **setcap**(8) and **libpsx**(3).