

**NAME**

**cap\_rights\_init**, **cap\_rights\_set**, **cap\_rights\_clear**, **cap\_rights\_is\_set**, **cap\_rights\_is\_valid**,  
**cap\_rights\_merge**, **cap\_rights\_remove**, **cap\_rights\_contains** - manage `cap_rights_t` structure

**LIBRARY**

Standard C Library (libc, -lc)

**SYNOPSIS**

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

```
cap_rights_t *  
cap_rights_init(cap_rights_t *rights, ...);
```

```
cap_rights_t *  
cap_rights_set(cap_rights_t *rights, ...);
```

```
cap_rights_t *  
cap_rights_clear(cap_rights_t *rights, ...);
```

```
bool  
cap_rights_is_set(const cap_rights_t *rights, ...);
```

```
bool  
cap_rights_is_valid(const cap_rights_t *rights);
```

```
cap_rights_t *  
cap_rights_merge(cap_rights_t *dst, const cap_rights_t *src);
```

```
cap_rights_t *  
cap_rights_remove(cap_rights_t *dst, const cap_rights_t *src);
```

```
bool  
cap_rights_contains(const cap_rights_t *big, const cap_rights_t *little);
```

**DESCRIPTION**

The functions documented here allow to manage the `cap_rights_t` structure.

Capability rights should be separated with comma when passed to the **cap\_rights\_init()**, **cap\_rights\_set()**, **cap\_rights\_clear()** and **cap\_rights\_is\_set()** functions. For example:

```
cap_rights_set(&rights, CAP_READ, CAP_WRITE, CAP_FSTAT, CAP_SEEK);
```

The complete list of the capability rights can be found in the `rights(4)` manual page.

The `cap_rights_init()` function initialize provided `cap_rights_t` structure. Only properly initialized structure can be passed to the remaining functions. For convenience the structure can be filled with capability rights instead of calling the `cap_rights_set()` function later. For even more convenience pointer to the given structure is returned, so it can be directly passed to `cap_rights_limit(2)`:

```
cap_rights_t rights;
```

```
if (cap_rights_limit(fd, cap_rights_init(&rights, CAP_READ, CAP_WRITE)) < 0)
    err(1, "Unable to limit capability rights");
```

The `cap_rights_set()` function adds the given capability rights to the given `cap_rights_t` structure.

The `cap_rights_clear()` function removes the given capability rights from the given `cap_rights_t` structure.

The `cap_rights_is_set()` function checks if all the given capability rights are set for the given `cap_rights_t` structure.

The `cap_rights_is_valid()` function verifies if the given `cap_rights_t` structure is valid.

The `cap_rights_merge()` function merges all capability rights present in the `src` structure into the `dst` structure.

The `cap_rights_remove()` function removes all capability rights present in the `src` structure from the `dst` structure.

The `cap_rights_contains()` function checks if the `big` structure contains all capability rights present in the `little` structure.

## RETURN VALUES

The functions never fail. In case an invalid capability right or an invalid `cap_rights_t` structure is given as an argument, the program will be aborted.

The `cap_rights_init()`, `cap_rights_set()` and `cap_rights_clear()` functions return pointer to the `cap_rights_t` structure given in the `rights` argument.

The **cap\_rights\_merge()** and **cap\_rights\_remove()** functions return pointer to the *cap\_rights\_t* structure given in the *dst* argument.

The **cap\_rights\_is\_set()** returns *true* if all the given capability rights are set in the *rights* argument.

The **cap\_rights\_is\_valid()** function performs various checks to see if the given *cap\_rights\_t* structure is valid and returns *true* if it is.

The **cap\_rights\_contains()** function returns *true* if all capability rights set in the *little* structure are also present in the *big* structure.

## EXAMPLES

The following example demonstrates how to prepare a *cap\_rights\_t* structure to be passed to the `cap_rights_limit(2)` system call.

```
cap_rights_t rights;
int fd;

fd = open("/tmp/foo", O_RDWR);
if (fd < 0)
    err(1, "open() failed");

cap_rights_init(&rights, CAP_FSTAT, CAP_READ);

if (allow_write_and_seek)
    cap_rights_set(&rights, CAP_WRITE, CAP_SEEK);

if (dont_allow_seek)
    cap_rights_clear(&rights, CAP_SEEK);

if (cap_rights_limit(fd, &rights) < 0 && errno != ENOSYS)
    err(1, "cap_rights_limit() failed");
```

## SEE ALSO

`cap_rights_limit(2)`, `open(2)`, `capsicum(4)`, `rights(4)`

## HISTORY

The functions **cap\_rights\_init()**, **cap\_rights\_set()**, **cap\_rights\_clear()**, **cap\_rights\_is\_set()**, **cap\_rights\_is\_valid()**, **cap\_rights\_merge()**, **cap\_rights\_remove()** and **cap\_rights\_contains()** first appeared in FreeBSD 8.3. Support for capabilities and capabilities mode was developed as part of the

TrustedBSD Project.

**AUTHORS**

This family of functions was created by Pawel Jakub Dawidek <*pawel@dawidek.net*> under sponsorship from the FreeBSD Foundation.