NAME

jail, **jail_get**, **jail_set**, **jail_remove**, **jail_attach** - create and manage system jails

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <sys/param.h>
#include <sys/jail.h>

int
jail(*struct jail *jail*);

int
jail_attach(*int jid*);

int
jail_remove(*int jid*);

#include <sys/uio.h>

int
jail_get(*struct iovec *iov, u_int niov, int flags*);

int
jail_set(*struct iovec *iov, u_int niov, int flags*);

DESCRIPTION

The **jail**() system call sets up a jail and locks the current process in it.

The argument is a pointer to a structure describing the prison:

```
struct jail {
        uint32_t   version;
        char            *path;
        char            *hostname;
        char            *jailname;
        unsigned int    ip4s;
        unsigned int    ip6s;
        struct in_addr  *ip4;
```

```
        struct in6_addr     *ip6;
};
```

"version" defines the version of the API in use. JAIL_API_VERSION is defined for the current version.

The "path" pointer should be set to the directory which is to be the root of the prison.

The "hostname" pointer can be set to the hostname of the prison. This can be changed from the inside of the prison.

The "jailname" pointer is an optional name that can be assigned to the jail for example for management purposes.

The "ip4s" and "ip6s" give the numbers of IPv4 and IPv6 addresses that will be passed via their respective pointers.

The "ip4" and "ip6" pointers can be set to an arrays of IPv4 and IPv6 addresses to be assigned to the prison, or NULL if none. IPv4 addresses must be in network byte order.

This is equivalent to, and deprecated in favor of, the **jail_set**() system call (see below), with the parameters *path*, *host.hostname*, *name*, *ip4.addr*, and *ip6.addr*, and with the JAIL_ATTACH flag.

The **jail_set**() system call creates a new jail, or modifies an existing one, and optionally locks the current process in it. Jail parameters are passed as an array of name-value pairs in the array *iov*, containing *niov* elements. Parameter names are a null-terminated string, and values may be strings, integers, or other arbitrary data. Some parameters are boolean, and do not have a value (their length is zero) but are set by the name alone with or without a "no" prefix, e.g. *persist* or *nopersist*. Any parameters not set will be given default values, generally based on the current environment.

Jails have a set of core parameters, and modules can add their own jail parameters. The current set of available parameters, and their formats, can be retrieved via the *security.jail.param* sysctl MIB entry. Notable parameters include those mentioned in the **jail**() description above, as well as *jid* and *name*, which identify the jail being created or modified. See jail(8) for more information on the core jail parameters.

The *flags* arguments consists of one or more of the following flags:

JAIL_CREATE
: Create a new jail. If a *jid* or *name* parameters exists, they must not refer to an existing jail.

JAIL_UPDATE
: Modify an existing jail. One of the *jid* or *name* parameters must exist, and must refer to an existing jail. If both JAIL_CREATE and JAIL_UPDATE are set, a jail will be created if it does not yet exist, and modified if it does exist.

JAIL_ATTACH
: In addition to creating or modifying the jail, attach the current process to it, as with the **jail_attach**() system call.

JAIL_DYING
: Allow setting a jail that is in the process of being removed.

The **jail_get**() system call retrieves jail parameters, using the same name-value list as **jail_set**() in the *iov* and *niov* arguments. The jail to read can be specified by either *jid* or *name* by including those parameters in the list. If they are included but are not intended to be the search key, they should be cleared (zero and the empty string respectively).

The special parameter *lastjid* can be used to retrieve a list of all jails. It will fetch the jail with the jid above and closest to the passed value. The first jail (usually but not always jid 1) can be found by passing a *lastjid* of zero.

The *flags* arguments consists of one or more following flags:

JAIL_DYING
: Allow getting a jail that is in the process of being removed.

The **jail_attach**() system call attaches the current process to an existing jail, identified by *jid*. It changes the process's root and current directories to the jail's *path* directory.

The **jail_remove**() system call removes the jail identified by *jid*. It will kill all processes belonging to the jail, and remove any children of that jail.

RETURN VALUES

If successful, **jail**(), **jail_set**(), and **jail_get**() return a non-negative integer, termed the jail identifier (JID). They return -1 on failure, and set *errno* to indicate the error.

The **jail_attach**() and **jail_remove**() functions return the value 0 if successful; otherwise the value -1 is returned and the global variable *errno* is set to indicate the error.

ERRORS

The **jail**() system call will fail if:

[EPERM] This process is not allowed to create a jail, either because it is not the super-user, or because it would exceed the jail's *children.max* limit.

[EFAULT] *jail* points to an address outside the allocated address space of the process.

[EINVAL] The version number of the argument is not correct.

[EAGAIN] No free JID could be found.

The **jail_set**() system call will fail if:

[EPERM] This process is not allowed to create a jail, either because it is not the super-user, or because it would exceed the jail's *children.max* limit.

[EPERM] A jail parameter was set to a less restrictive value then the current environment.

[EFAULT] *Iov*, or one of the addresses contained within it, points to an address outside the allocated address space of the process.

[ENOENT] The jail referred to by a *jid* or *name* parameter does not exist, and the JAIL_CREATE flag is not set.

[ENOENT] The jail referred to by a *jid* is not accessible by the process, because the process is in a different jail.

[EEXIST] The jail referred to by a *jid* or *name* parameter exists, and the JAIL_UPDATE flag is not set.

[EINVAL] A supplied parameter is the wrong size.

[EINVAL] A supplied parameter is out of range.

[EINVAL] A supplied string parameter is not null-terminated.

[EINVAL] A supplied parameter name does not match any known parameters.

[EINVAL] One of the JAIL_CREATE or JAIL_UPDATE flags is not set.

[ENAMETOOLONG]
A supplied string parameter is longer than allowed.

[EAGAIN] There are no jail IDs left.

The **jail_get**() system call will fail if:

[EFAULT] *Iov*, or one of the addresses contained within it, points to an address outside the allocated address space of the process.

[ENOENT] The jail referred to by a *jid* or *name* parameter does not exist.

[ENOENT] The jail referred to by a *jid* is not accessible by the process, because the process is in a different jail.

[ENOENT] The *lastjid* parameter is greater than the highest current jail ID.

[EINVAL] A supplied parameter is the wrong size.

[EINVAL] A supplied parameter name does not match any known parameters.

The **jail_attach**() and **jail_remove**() system calls will fail if:

[EPERM] A user other than the super-user attempted to attach to or remove a jail.

[EINVAL] The jail specified by *jid* does not exist.

Further **jail**(), **jail_set**(), and **jail_attach**() call chroot(2) internally, so they can fail for all the same reasons. Please consult the chroot(2) manual page for details.

SEE ALSO

chdir(2), chroot(2), jail(8)

HISTORY

The **jail**() system call appeared in FreeBSD 4.0. The **jail_attach**() system call appeared in FreeBSD 5.1. The **jail_set**(), **jail_get**(), and **jail_remove**() system calls appeared in FreeBSD 8.0.

AUTHORS

The jail feature was written by Poul-Henning Kamp for R&D Associates who contributed it to FreeBSD. James Gritton added the extensible jail parameters and hierarchical jails.