NAME

priv - kernel privilege checking API

SYNOPSIS

#include <sys/priv.h>

int
priv_check(*struct thread *td, int priv*);

int
priv_check_cred(*struct ucred *cred, int priv*);

DESCRIPTION

The **priv** interfaces check to see if specific system privileges are granted to the passed thread, *td*, or credential, *cred*. This interface replaces the now removed suser(9) privilege checking interface. Privileges typically represent rights in one of two categories: the right to manage a particular component of the system, or an exemption to a specific policy or access control list. The caller identifies the desired privilege via the *priv* argument.

Privilege Policies

Privileges are typically granted based on one of two base system policies: the superuser policy, which grants privilege based on the effective (or sometimes real) UID having a value of 0, and the jail(2) policy, which permits only certain privileges to be granted to processes in a jail. The set of available privileges may also be influenced by the TrustedBSD MAC Framework, described in mac(9).

IMPLEMENTATION NOTES

When adding a new privilege check to a code path, first check the complete list of current privileges in *sys/priv.h* to see if one already exists for the class of privilege required. Only if there is not an exact match should a new privilege be added to the privilege list. As privilege numbers becomes encoded in the kernel module ABI, privilege constants must not be changed as any kernel modules depending on privileges will then need to be recompiled. When adding a new privilege, be certain to also determine whether it should be listed in **prison_priv_check**(), which includes a complete list of privileges granted to the root user in jail(2).

Certain catch-all privileges exist, such as PRIV_DRIVER, intended to be used by device drivers, rather than adding a new driver-specific privilege.

RETURN VALUES

Typically, 0 will be returned for success, and EPERM will be returned on failure. Most consumers of **priv** will wish to directly return the error code from a failed privilege check to user space; a small

number will wish to translate it to another error code appropriate to a specific context.

When designing new APIs, it is preferable to return explicit errors from a call if privilege is not granted rather than changing the semantics of the call but returning success. For example, the behavior exhibited by stat(2), in which the generation field is optionally zero'd out when there is insufficient privilege is highly undesirable, as it results in frequent privilege checks, and the caller is unable to tell if an access control failure occurred.

SEE ALSO

jail(2), mac(9), ucred(9)

AUTHORS

The **priv** API and implementation were created by Robert Watson under contract to nCircle Network Security, Inc.