NAME

arc4rand, **arc4random**, **arc4random_buf**, **is_random_seeded**, **random**, **read_random**, **read_random_uio** - supply pseudo-random numbers

SYNOPSIS

#include <sys/libkern.h>

uint32_t
arc4random(*void*);

void
arc4random_buf(*void *ptr*, *size_t len*);

void
arc4rand(*void *ptr*, *u_int length*, *int reseed*);

#include <sys/random.h>

bool
is_random_seeded(*void*);

void
read_random(*void *buffer, int count*);

int
read_random_uio(*struct uio *uio*, *bool nonblock*);

LEGACY ROUTINES

#include <sys/libkern.h>

u_long
random(*void*);

DESCRIPTION

The **arc4random**() and **arc4random_buf**() functions will return very good quality random numbers, suited for security-related purposes. Both are wrappers around the underlying **arc4rand**() interface. **arc4random**() returns a 32-bit random value, while **arc4random_buf**() fills *ptr* with *len* bytes of random data.

The **arc4rand**() CSPRNG is seeded from the random(4) kernel abstract entropy device. Automatic reseeding happens at unspecified time and bytes (of output) intervals. A reseed can be forced by passing a non-zero *reseed* value.

The **read_random**() function is used to read entropy directly from the kernel abstract entropy device. **read_random**() blocks if and until the entropy device is seeded. The provided *buffer* is filled with no more than *count* bytes. It is strongly advised that **read_random**() is not used directly; instead, use the **arc4rand**() family of functions.

The **is_random_seeded**() function can be used to check in advance if **read_random**() will block. (If random is seeded, it will not block.)

The **read_random_uio**() function behaves identically to read(2) on */dev/random*. The *uio* argument points to a buffer where random data should be stored. If *nonblock* is true and the random device is not seeded, this function does not return any data. Otherwise, this function may block interruptibly until the random device is seeded. If the function is interrupted before the random device is seeded, no data is returned.

The deprecated **random**() function will return a 31-bit value. It is obsolete and scheduled to be removed in FreeBSD 14.0. Consider prng(9) instead and see *SECURITY CONSIDERATIONS*.

RETURN VALUES

The **arc4rand**() function uses the Chacha20 algorithm to generate a pseudo-random sequence of bytes. The **arc4random**() function uses **arc4rand**() to generate pseudo-random numbers in the range from 0 to (2**32)-1.

The **read_random**() function returns the number of bytes placed in *buffer*.

read_random_uio() returns zero when successful, otherwise an error code is returned.

random() returns numbers in the range from 0 to (2**31)-1.

ERRORS

read_random_uio() may fail if:

[EFAULT]	*uio* points to an invalid memory region.
[EWOULDBLOCK]	The random device is unseeded and *nonblock* is true.

AUTHORS

Dan Moschuk wrote **arc4random**().
Mark R V Murray wrote **read_random**().

SECURITY CONSIDERATIONS

Do not use **random**() in new code.

It is important to remember that the **random**() function is entirely predictable. It is easy for attackers to predict future output of **random**() by recording some generated values. We cannot emphasize strongly enough that **random**() must not be used to generate values that are intended to be unpredictable.