NAME

getentropy − fill a buffer with random bytes

LIBRARY

Standard C library (*libc*, *−lc*)

SYNOPSIS

#include <unistd.h>

int getentropy(void *buffer***[.***length***], size_t** *length***);**

Feature Test Macro Requirements for glibc (see **feature_test_macros**(7)):

getentropy():
 _DEFAULT_SOURCE

DESCRIPTION

The **getentropy**() function writes *length* bytes of high-quality random data to the buffer starting at the location pointed to by *buffer*. The maximum permitted value for the *length* argument is 256.

A successful call to **getentropy**() always provides the requested number of bytes of entropy.

RETURN VALUE

On success, this function returns zero. On error, −1 is returned, and *errno* is set to indicate the error.

ERRORS

EFAULT
Part or all of the buffer specified by *buffer* and *length* is not in valid addressable memory.

EIO *length* is greater than 256.

EIO An unspecified error occurred while trying to overwrite *buffer* with random data.

ENOSYS
This kernel version does not implement the **getrandom**(2) system call required to implement this function.

VERSIONS

The **getentropy**() function first appeared in glibc 2.25.

STANDARDS

This function is nonstandard. It is also present on OpenBSD.

NOTES

The **getentropy**() function is implemented using **getrandom**(2).

Whereas the glibc wrapper makes **getrandom**(2) a cancelation point, **getentropy**() is not a cancelation point.

getentropy() is also declared in **<sys/random.h>**. (No feature test macro need be defined to obtain the declaration from that header file.)

A call to **getentropy**() may block if the system has just booted and the kernel has not yet collected enough randomness to initialize the entropy pool. In this case, **getentropy**() will keep blocking even if a signal is handled, and will return only once the entropy pool has been initialized.

SEE ALSO

getrandom(2), **urandom**(4), **random**(7)