Package 'cryptorng'

May 30, 2024

Type Package

Title Access System Cryptographic Pseudorandom Number Generators

Version 0.1.3

Maintainer Mike Cheng <mikefc@coolbutuseless.com>

Description Generate random numbers from the Cryptographically Secure Pseudorandom Number Generator (CSPRNG) provided by the underlying operating system. System CSPRNGs are seeded internally by the OS with entropy it gathers from the system hardware. The following system functions are used: arc4random_buf() on macOS and BSD; BCryptgenRandom() on Windows; Sys_getrandom() on Linux.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.1

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://github.com/coolbutuseless/cryptorng

BugReports https://github.com/coolbutuseless/cryptorng/issues

NeedsCompilation yes

Author Mike Cheng [aut, cre, cph]

Repository CRAN

Date/Publication 2024-05-30 13:00:02 UTC

R topics documented:

`rcrypto` *Generate random numbers using platform-specific cryptographically secure pseudorandom number generators*

Description

Generate random numbers using platform-specific cryptographically secure pseudorandom number generators

Usage

```
rcrypto(n, type = "raw")
```

Arguments

`n`	Number of random numbers to generate. Note: if the entropy pool is exhausted on your system it may not be able to provide the requested number of bytes - in this case an error is thrown.
`type`	Type of returned values - 'raw', 'chr', 'lgl', 'int' or 'dbl'. Default: 'raw'

'raw' Uniform random bytes from the CSPRNG returned as a raw vector

'chr' Uniform random bytes from the CRPRNG returned as a hexadecimal string

'lgl' Uniform random bytes return as random logical values

'int' Combines 4 random bytes to create uniform random integers. This output is further filtered to remove any NA values which may occur

'dbl' Combines 8 random bytes to create uniform random numbers in the range [0, 1]

Value

Depending on the `type` argument: a hexadecimal string, a raw vector, a logical vector, an integer vector or a numeric vector.

Details for `type = 'dbl'`

An 8-byte double-precision floating point number is obtained by first concatenating 8 random bytes into an 8-byte unsigned integer (i.e. `uint64_t`).

This `uint64_t` value is converted to an 8-byte double using: `(x >> 11) * 0x1.0p-53`.

Details for `type = 'int'`

A 4-byte random R integer value is obtained by concatenating 4 random bytes. These integer values are then filtered to exclude the special `NA_integer` value used by R.

Platform notes

The method used for generating random values varies depending on the operating system (OS):

- For macOS and BSDs: `arc4random_buf()`
- For linux: `syscall(SYS_getrandom())`
- For win32: `BCryptGenRandom()`

All these random number generators are internally seeded by the OS using entropy gathered from multiple sources and are considered cryptographically secure.

Examples

```
rcrypto(16, type = 'raw')
rcrypto(16, type = 'chr')
rcrypto(16, type = 'lgl')
rcrypto(16, type = 'int')
rcrypto(16, type = 'dbl')
```

Index