NAME

RAND − the OpenSSL random generator

DESCRIPTION

Random numbers are a vital part of cryptography, they are needed to provide unpredictability for tasks like key generation, creating salts, and many more. Software-based generators must be seeded with external randomness before they can be used as a cryptographically-secure pseudo-random number generator (CSPRNG). The availability of common hardware with special instructions and modern operating systems, which may use items such as interrupt jitter and network packet timings, can be reasonable sources of seeding material.

OpenSSL comes with a default implementation of the RAND API which is based on the deterministic random bit generator (DRBG) model as described in [NIST SP 800−90A Rev. 1]. The default random generator will initialize automatically on first use and will be fully functional without having to be initialized ('seeded') explicitly. It seeds and reseeds itself automatically using trusted random sources provided by the operating system.

As a normal application developer, you do not have to worry about any details, just use **RAND_bytes** (3) to obtain random data. Having said that, there is one important rule to obey: Always check the error return value of **RAND_bytes** (3) and do not take randomness for granted. Although (re−)seeding is automatic, it can fail because no trusted random source is available or the trusted source(s) temporarily fail to provide sufficient random seed material. In this case the CSPRNG enters an error state and ceases to provide output, until it is able to recover from the error by reseeding itself. For more details on reseeding and error recovery, see **RAND_DRBG** (7).

For values that should remain secret, you can use **RAND_priv_bytes** (3) instead. This method does not provide 'better' randomness, it uses the same type of CSPRNG. The intention behind using a dedicated CSPRNG exclusively for private values is that none of its output should be visible to an attacker (e.g., used as salt value), in order to reveal as little information as possible about its internal state, and that a compromise of the ''public'' CSPRNG instance will not affect the secrecy of these private values.

In the rare case where the default implementation does not satisfy your special requirements, there are two options:

- Replace the default RAND method by your own RAND method using **RAND_set_rand_method** (3).
- Modify the default settings of the OpenSSL RAND method by modifying the security parameters of the underlying DRBG, which is described in detail in **RAND_DRBG** (7).

Changing the default random generator or its default parameters should be necessary only in exceptional cases and is not recommended, unless you have a profound knowledge of cryptographic principles and understand the implications of your changes.

SEE ALSO

RAND_add (3), **RAND_bytes** (3), **RAND_priv_bytes** (3), **RAND_get_rand_method** (3), **RAND_set_rand_method** (3), **RAND_OpenSSL** (3), **RAND_DRBG** (7)

COPYRIGHT

Copyright 2018−2019 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the OpenSSL license (the ''License''). You may not use this file except in compliance with the License. You can obtain a copy in the file LICENSE in the source distribution or at <https://www.openssl.org/source/license.html>.