Chapter 7
Random Number Generator

Thomas Lugrin

7.1 Introduction

Most modern encryption and authentication methods rely on the generation of random numbers [1], such as for key generation, initial vectors, or nonces. Therefore, a reliable source of entropy is fundamental in making encryption and authentication methods secure—weak sources of randomness can compromise otherwise secure encryption and authentication schemes.

7.2 Analysis

7.2.1 Definition

A Random Number Generator (RNG) is cryptographically secure if the sequences of numbers that it generates are unpredictable (Section 3.3.1 of [2]). RNGs are typically grouped in two categories: Pseudo-Random Number Generators (PRNG) and True Random Number Generators (TRNG).

PRNGs depend on a seed value, from which a seemingly erratic albeit deterministic sequence is produced; it is a quick and debug-friendly version of RNGs often used in statistical applications. They are not suitable for cryptographic applications in isolation. However, they may be used when correctly combined (seeded) with a reliable entropy source.

T. Lugrin (✉)
Federal Administration, Bern, Switzerland
e-mail: thomas.lugrin@vtg.admin.ch

V. Mulder et al. (eds.), *Trends in Data Protection and Encryption Technologies*,
https://doi.org/10.1007/978-3-031-33386-6_7

TRNGs rely on physical phenomena, e.g., radioactive decay, thermal noise, small-scale hardware activity, or particular hardware based on quantum physics (abbreviated QRNG; see for example, Chapter 2 of [3]). As it is hard to balance physical processes such that the probability of 0's and 1's is exactly ½, the output of TRNGs must be adequately post-processed. Secure mixing functions such as hash functions or symmetric encryption schemes may produce unbiased output [4]. These mixing functions also remove serial dependence between bits. An excellent example of such an implementation is the Linux kernel RNG `/dev/urandom` [5].

Quantum RNGs are often presented as the only means to protect infrastructure against future powerful quantum computers. However, this is misleading, as any reliable source of randomness remains unpredictable against any adversary with arbitrary computing power.

7.2.2 Trends

Small-size, low-cost QRNGs have already been integrated into off-the-shelf devices such as smartphones, computers, and hardware security modules.

7.3 Consequences for Switzerland

People, businesses, and authorities in Switzerland should continue using and promoting research on secure random hardware number generators. This will ensure that they can benefit from the newest technological advances when they become available.

7.3.1 Implementation Possibilities: Make or Buy

Using secure RNGs that cannot be manipulated or tampered with and whose output is not predictable is fundamental as a basis for encryption methods. Applications involving particularly sensitive data can combine the output from two or more independent sources of randomness for improved security. PRNGs, which produce deterministic outcomes, must not be used in cryptography in isolation and must at least blend in TRNG's randomness.

Open-source solutions such as the Linux kernel RNG `/dev/urandom` are considered reliable [6]. Hardware products dedicated to producing randomness from reliable and reputable producers can be used as a complement after appropriate verification and approval.

Several companies are operating in the TRNG market, e.g., developing QRNG chips that can be integrated into hardware. A few companies selling QRNG chips

Table 7.1 Different companies active in the QRNG field

Company	Description	Technology	Country
ID Quantique	Technology pioneers, well established, integrated into a chip, promote cost-effectiveness.	Photonic (Optical)	Switzerland (Linked to South Korea through SK Telecom)
Quintessence Labs	Well established, fastest generators, not chip integrated	Barrier Tunneling.	Australia
RandomPower	Newcomers, growing, qualification and MVP in place, offers new technology. RUAG Switzerland ran tests on their products.	In-silico	Italy

or systems are listed in Table 7.1. These QRNG chips do not offer stronger guarantees than other TRNGs; they are just another means of potentially generating cryptographically secure randomness.

7.3.2 Variation and Recommendation

RNGs should be appropriately isolated and integrity protected to prevent tampering or access to internal states that could leak information about the random sequence. Combining the output of several RNGs (e.g., using XOR) can mitigate the potential weaknesses of individual RNGs.

The US National Institute of Standards and Technology (NIST) published a range of hypothesis tests [7] that can provide evidence of potentially complex dependence patterns. Its German equivalent (Bundesamt für Sicherheit in der Informationstechnik, BSI) also suggests a suite of tests [8]. These tests do not provide proof of randomness; they can, at best, reject the null hypothesis that a specific dependence pattern occurs in a sequence at a given confidence level. The longer the test sequence, the more confidence can be placed in the test results. A good understanding of the inner workings of a TRNG is key to assuring the unpredictability of its output.

7.4 Conclusion

A reliable source of randomness is critical to ensuring the security of most modern encryption and authentication systems. Unfortunately, pseudo-random number

generators are not suited in such a context, except if suitably combined with a reliable entropy source.

Proving that a source of bits is truly random is impossible on finite sequences, but statistical test suites exist that provide evidence against non-randomness. Good physical sources of entropy must be chained with robust post-processing techniques to remove biases and serial dependencies.

Standard tools like `/dev/urandom` on Linux systems provide a good source of random numbers based on multiple hardware-based entropy sources. Additional security can be achieved by combining independent RNGs, typically based on physical processes of different types, e.g., quantum physics.

References

1. Kinga Marton, Alin Suciu, and Iosif Ignat. Randomness in Digital Cryptography: A Survey. *Romanian Journal of Information Science and Technology*, 13:219–240, 2010.
2. Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. CRC Press, Boca Raton, 3rd edition, 2021.
3. David Johnston. *Random Number Generators—Principles and Practices: A Guide for Engineeers and Programmers*. Walter de Gruyter GmbH, Berlin/Boston, 2018.
4. Steve Crocker, Donald E. Eastlake 3rd, and Jeffrey I. Schiller. Randomness Recommendations for Security. Request for Comments RFC 1750, Internet Engineering Task Force, December 1994.
5. random(4) - Linux manual page. https://man7.org/linux/man-pages/man4/random.4.html, August 2022.
6. Stephan Müller. Documentation and Analysis of the Linux Random Number Generator. Technical report, Bundesamt für Sicherheit in der Informationstechnik, April 2020.
7. Andrew Rukhin, Juan Soto, James Nechvatal, Elaine Barker, Stefan Leigh, Mark Levenson, David Banks, Alan Heckert, and James Dray. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. Technical report, National Institute of Standards and Technology, April 2010.
8. Wolfgang Killmann and Werner Schindler. A proposal for: Functionality classes for random number generators. Technical report, Bundesamt für Sicherheit in der Informationstechnik, September 2011.