

# Stream ciphers

Dr. Yee Wei Law <yeewei.law@unisa.edu.au>

2023-09-04

## Contents

1	Introduction	1	3	References	6
2	Definition and classification	2			

## List of acronyms

<b>IETF</b>	Internet Engineering Task Force	2
<b>LFSR</b>	linear feedback shift register	4
<b>NESSIE</b>	New European Schemes for Signatures, Integrity and Encryption	1
<b>PPT</b>	probabilistic polynomial-time	3
<b>TLS</b>	Transport-Layer Security	2

## 1 Introduction

*Provable security* is a reductionist approach to proving the security properties of computational security schemes.

Leveraging provable security, cryptographers produced a useful array of cryptographic primitives.

Let us be absolutely clear right from the outset: a symmetric-key encryption scheme can be constructed using **1** a stream cipher, or **2** a block cipher and a mode of operation.

The term “stream cipher” is not defined in NIST’s [Computer Security Research Center Glossary](#). In fact, there is no NIST Special Publication or FIPS that covers stream ciphers.

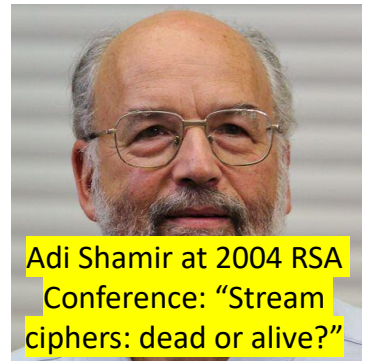
Stream ciphers have had a bumpy road:

- The [New European Schemes for Signatures, Integrity and Encryption \(NESSIE\) project](#) that took place from 1999 till 2003 did not provide any recommendation for stream ciphers, partly because most stream ciphers in use were secret or proprietary designs [[PBO<sup>+</sup>03](#), Sec. 3.3.5], and partly because of the uncertainties with the security of the candidate stream ciphers submitted to the project for study.

- It was around the time of the **NESSIE** project that multiple vulnerabilities of the once-standard stream cipher RC4 were discovered.

» In fact, there is an Internet Engineering Task Force (IETF) RFC, namely RFC 7465 [Pop15], that specifically prohibits the use of the troubled RC4 cipher suite in Transport-Layer Security (TLS).

» One of the most recent attacks is AlFardan et al.'s [ABP<sup>+</sup>13], which managed to recover the first 200 bytes of a plaintext stream after  $2^{28}$  to  $2^{32}$  encryptions of the same plaintext; watch [video](#).



- On the heels of the **NESSIE** project, the **ECRYPT Stream Cipher (eSTREAM) project**, which ran from 2004 till 2008, proposed 4 algorithms for software implementation and 3 algorithms for hardware implementation.
- Among the eSTREAM portfolio [RB08] are **Salsa20** and **Grain**:
  - » **Salsa20** [RB08, pp. 84-97]: Created by D.J. Bernstein, this cipher has been succeeded by **ChaCha20**. Although not standardised by NIST, ChaCha20 is standardised in RFC 8439 [NL18], and implemented in **OpenSSL**, **BoringSSL** and **Tink**.
  - » **Grain** [RB08, pp. 179-190]: This started as two variants: 80-bit and 128-bit. The latter evolved into Grain-128AEAD [HJM<sup>+</sup>21], a finalist (but not a winner) in the NIST Lightweight Cryptography standardisation process.

Fortunately, a stream cipher can be constructed from a block cipher, and this is one of the reasons why there has been much more development in block ciphers than stream ciphers.

Nevertheless, stream ciphers are desirable **1** where exceptionally high throughput is required in software and **2** where exceptionally low resource consumption is required in hardware [RB08].

## 2 Definition and classification

### Definition 1: Stream cipher [KL21, Sec. 3.6.1]

A stream cipher is a pair of deterministic algorithms ( $\text{Init}$ ,  $f$ ) where

- $\text{Init}$  takes as input **1** a secret key serving as a seed  $s$  and **2** an optional initialisation vector (IV), and outputs some initial state  $\sigma_0$ .
- The next-state function  $f$  takes as input the current state  $\sigma_i$  and outputs a bit  $z_i$  along with the updated state  $\sigma_{i+1}$ .

⚠ An IV is practically mandated for security, for example by the eSTREAM project [RB08, p. 2].

⚠ In practice,  $f$  outputs a byte or a larger number of random bits, but a bit is the smallest granularity.

Clearly, a stream cipher is stateful, and is sometimes referred to as a *state-based symmetric-key encryption scheme* [Gol04, Construction 5.3.1.2].

Inspired by the one-time pad, a stream cipher generates a pseudorandom *keystream* from a fixed-length secret key, with the intent that the keystream appears random to a probabilistic polynomial-time (PPT) adversary.

Stream ciphers can be classified as:

**Synchronous/synchronised** [MvV96, Definition 6.2]: Keystream is generated independently of the plaintext and ciphertext.

As per Figure 1, the encryption process can be described by:

$$\sigma_{i+1} = f(\sigma_i, k), \quad (1a)$$

$$z_i = g(\sigma_i, k), \quad (1b)$$

$$c_i = h(z_i, m_i), \quad (1c)$$

where  $\sigma_0$  is the secret initial state and maybe determined from the key  $k$ ;  $g$  is a function that produces the keystream  $z_i$ ;  $h$  is the output function that combines the keystream and plaintext  $m_i$  to produce ciphertext  $c_i$ .

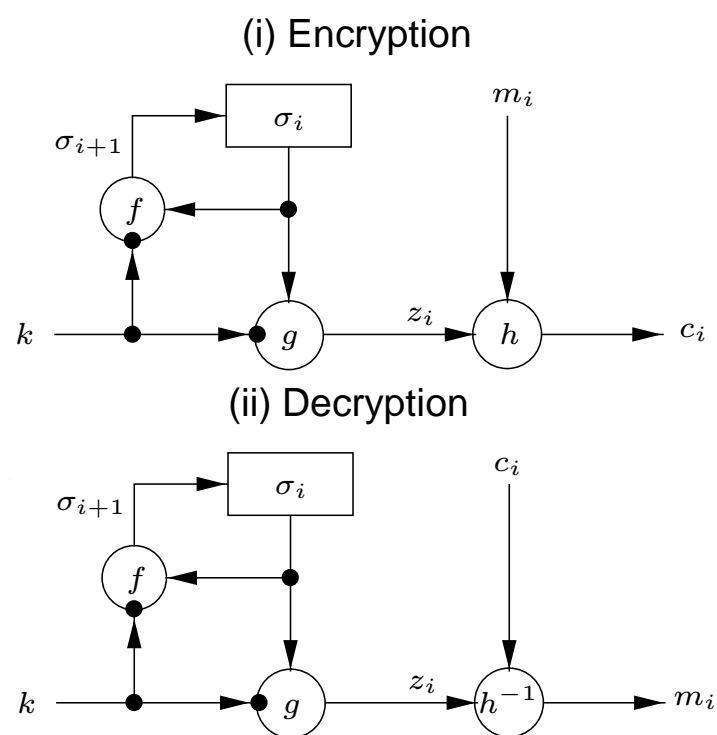


Figure 1: General model of a synchronous stream cipher [MvV96, Figure 6.1].

A synchronous stream cipher can be constructed from a block cipher in the *output feedback* (OFB) mode or *counter* (CTR) mode; see [knowledge base entry](#).

An example of a synchronous stream cipher is:

**Asynchronous/unsynchronised/self-synchronising** [MvV96, Def. 6.5]: Keystream is generated as a function of the key and a fixed number of previous ciphertext bits.

As per Figure 2, the encryption function can be described by:

$$\sigma_i = (c_{i-t}, c_{i-t+1}, \dots, c_{i-1}), \quad (2a)$$

$$z_i = g(\sigma_i, k), \quad (2b)$$

$$c_i = h(z_i, m_i), \quad (2c)$$

where  $\sigma_0 = (c_{-t}, c_{-t+1}, \dots, c_{-1})$  is the non-secret initial state;  $k$  is the key;  $g$  is a function that produces the keystream  $z_i$ ;  $h$  is the output function that combines the keystream and plaintext  $m_i$  to produce ciphertext  $c_i$ .

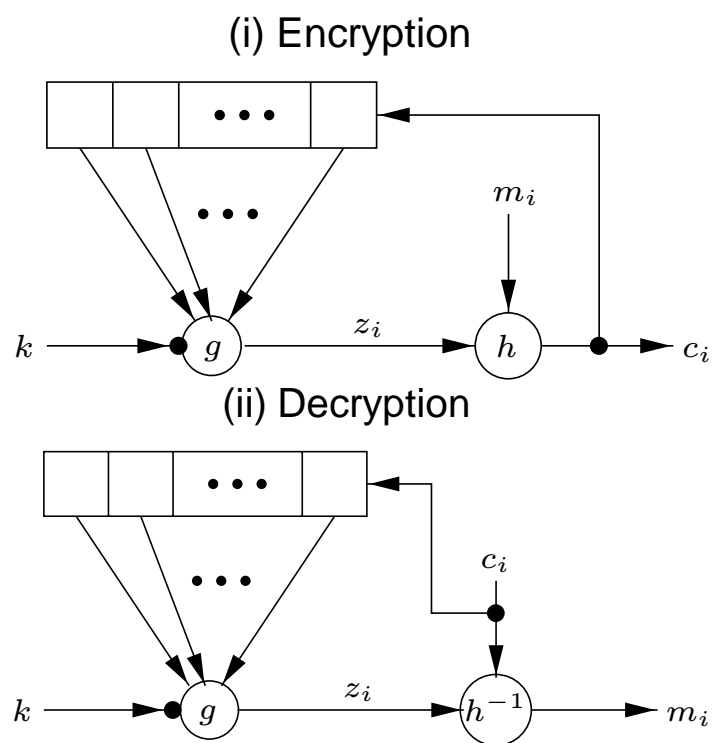


Figure 2: General model of an asynchronous stream cipher [MvV96, Figure 6.3].

An asynchronous stream cipher can be constructed from a block ciphers in the 1-bit *cipher feedback* (CFB) mode; see [knowledge base entry](#).

Definition 2: Binary additive stream cipher [MvV96, Definition 6.4], [Sma16, Sec. 10.2]

A synchronous stream cipher in which the keystream, plaintext, and ciphertext digits are binary digits, and the output function  $h$  is the XOR function.

As shown in Figure 3, the binary additive stream cipher consists a *keystream generator* or *running key generator* implemented by the functions  $f$  and  $g$  in Eq. (1).

The linear feedback shift register (LFSR) has been a staple in keystream generator designs, because

1. it can readily be analysed using algebraic techniques,
2. it can produce bit streams that pass common statistical tests,
3. it is well-suited to efficient hardware implementation, and
4. it is well-studied.

See [RB08, p. 20] and [MvV96, Sec. 6.2.1].

An LFSR is a shift register whose input bit is a linear function of its previous state [Jet19]; watch [LinkedIn Learning video](#).

Each LFSR is associated with a primitive polynomial in a Galois field.

However, the linearity of the LFSR makes it susceptible to algebraic attacks [RB08]. For example, the stream cipher Toyocrypt was broken by an attack that can solve an overdefined system of multivariate equations for the initial state bits [CM03]; recovery of the initial state bits leads to recover of the key.

Thus, an LFSR is typically either used together with nonlinear structures, or replaced by nonlinear structures such as a *nonlinear feedback shift register* (NFSR); see Example 1.

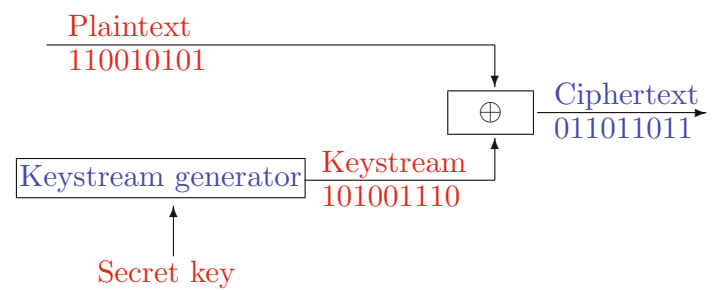


Figure 3: A binary additive stream cipher [Sma16, Figure 10.2]. IV omitted but usually present. Blue annotation indicates publicly available information. Red annotation indicates information available to legitimate parties.

### Example 1

One of the **ten finalists** in NIST's lightweight AEAD standardisation process that started in 2017 is Grain-128AEAD.

Grain-128AEADv2 is the second version of the Grain-128AEAD and consists of binary additive stream cipher, as shown in Figure 4.





### 3 References

- [ABP<sup>+</sup>13] N. ALFARDAN, D. J. BERNSTEIN, K. G. PATERSON, B. POETTERING, and J. C. N. SCHULDT, On the security of RC4 in TLS, in *22nd USENIX Security Symposium (USENIX Security 13)*, USENIX Association, Washington, D.C., August 2013, pp. 305–320. Available at <https://www.usenix.org/conference/usenixsecurity13/technical-sessions/paper/alFardan>.
- [CM03] N. T. COURTOIS and W. MEIER, Algebraic attacks on stream ciphers with linear feedback, in *Advances in Cryptology — EUROCRYPT 2003* (E. BIHAM, ed.), Springer Berlin Heidelberg, Berlin, Heidelberg, 2003, pp. 345–359.
- [Gol04] O. GOLDREICH, *Foundations of Cryptography: Volume II Basic Applications*, Cambridge University Press, 2004. <https://doi.org/10.1017/CB09780511721656>.
- [HJM<sup>+</sup>21] M. HELL, T. JOHANSSON, A. MAXIMOV, W. MEIER, J. SÖNNERUP, and H. YOSHIDA, Grain-128AEADv2 - A lightweight AEAD stream cipher, 2021. Available at <https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/grain-128aead-spec-final.pdf>.
- [Jet19] U. JETZEK, *Galois Fields, Linear Feedback Shift Registers and their Applications*, Carl Hanser Verlag, 2019. <https://doi.org/10.3139/9783446456136>.
- [KL21] J. KATZ and Y. LINDELL, *Introduction to Modern Cryptography*, 3rd ed., CRC Press, 2021. Available at <https://ebookcentral.proquest.com/lib/unisa/detail.action?docID=6425020>.
- [MvV96] A. J. MENEZES, P. C. VAN OORSCHOT, and S. A. VANSTONE, *Handbook of Applied Cryptography*, CRC Press, 1996. Available at <https://cacr.uwaterloo.ca/hac/>.
- [NL18] Y. NIR and A. LANGLEY, ChaCha20 and Poly1305 for IETF Protocols, IETF RFC 8439, June 2018. Available at <https://www.rfc-editor.org/rfc/rfc8439>.
- [Pop15] A. POPOV, Prohibiting RC4 cipher suites, IETF RFC 7465, 2015. Available at <https://www.rfc-editor.org/rfc/rfc7465.html>.
- [PBO<sup>+</sup>03] B. PRENEEL, A. BIRYUKOV, E. OSWALD, B. V. ROMPAY, L. GRANBOULAN, E. DOTTA, S. MURPHY, A. DENT, J. WHITE, M. DICHTL, S. PYKA, M. SCHAFHEUTLE, P. SERF, E. BIHAM, E. BARKAN, O. DUNKELMAN, J.-J. QUISQUATER, M. CIET, F. SICA, L. KNUDSEN, M. PARKER, and H. RADDUM, *NESSIE Security Report*, Deliverable D20, NESSIE Consortium, February 2003, Version 2.0.
- [RB08] M. ROBSHAW and O. BILLET (eds.), *New Stream Cipher Designs: The eSTREAM Finalists*, LNCS 4986, Springer Berlin, Heidelberg, 2008. <https://doi.org/10.1007/978-3-540-68351-3>.
- [Sma16] N. P. SMART, *Cryptography Made Simple, Information Security and Cryptography*, Springer International Publishing Switzerland, 2016. <https://doi.org/10.1007/978-3-319-21936-3>.