Security proof of the canonical form of self-synchronizing stream ciphers

Brandon Dravie, Philippe Guillot, Gilles Millérioux

Security Proof of the Canonical Form of Self-Synchronizing Stream Ciphers

Brandon Dravie[1], Philippe Guillot[2] and Gilles Millérioux[1]

[1] Université de Lorraine
Centre de Recherche en Automatique de Nancy (CRAN CNRS UMR 7039), France,
{brandon.dravie,gilles.millerioux}@univ-lorraine.fr
[2] Université Paris 8
Laboratoire Analyse, Géométrie et Applications (LAGA CNRS UMR 7539), France
philippe.guillot@univ-paris8.fr

Abstract. This paper studies the security level expected by the canonical form of the Self-Synchronizing Stream Cipher (SSSC). A SSSC can be viewed as the combination of a shift register together with a filtering function. The maximum security of such a cipher is reached when the filtering function is random. However, in practice, Pseudo Random Functions (PRF) are used as filtering functions. In this case, it is shown that the security against chosen-ciphertext attacks (IND-CCA security) cannot be reached for the canonical form of the SSSC, but it is however secure against chosen plaintext attacks (IND-CPA secure). This result guarantees the existence of SSSC that can be IND-CPA secure although till now, the SSSC proposed in the open literature had be broken against IND-CPA attacks. The security proof lies on the property of indistinguishability.

1 Introduction

Self-Synchronizing Stream Ciphers (SSSC) was patented in 1946. The basic principe of such ciphers is to encrypt every plaintext symbol with a transformation that only involves a fixed number of previous ciphertexts symbols. Therefore, every ciphertext symbol is correctly decrypted provided that previous symbols have been properly received. This self-synchronisation property has many advantages and is especially relevant to group communications.

Regarding security, as each plaintext symbol influences potentially all subsequent ciphertexts, they naturally have good diffusion properties and are efficient against attacks based on plaintext redundancy. Furthermore, they can prevent from traffic analysis as the information which is conveyed through the channel is encrypted whether there is traffic or not.

In the early 90s, studies have been performed [Mau91,DGV92] to propose secure design of SSSC. These works have been followed by effective constructions ([DGV92,Sar03,DK08]), but till now, all of these SSSC schemes have been broken, what may make believe that a secure SSSC is impossible to design.

The canonical form of the Self-Synchronizing Stream Cipher (SSSC) is constituted by the combination of a shift register, which acts as a state register with the ciphertext as input, together with a filtering function that provides the running keystream. In this paper, it is shown that this architecture is not resistant against chosen ciphertext attack (IND-CCA security), but can reach the resistance against chosen plaintext attack (IND-CPA security), provided that the filtering function is pseudo random. The technical developments used to establish the security proof follow similar lines that those used when dealing with block cipher symmetric encryption scheme.

The paper is organized as follows. In Section 2, we recall the characteristic of the canonical form of the SSSC. Section 3 is devoted to broad security notions following by the study of the security of the canonical form of SSSC. Finally, we end up with concluding remarks in Section 4. In particular, we raise the problem of deriving minimal conditions on the output function of the canonical form in order to guarantee the IND-CPA security.

2 Canonical form of the Self-Synchronization Stream Cipher

2.1 Generalities on SSSC

A conventional way for designing an SSSC is to resort to a shift-register-like architecture, giving the so-called canonical representation of the SSSC [MvOV96]. This canonical representation is depicted in Figure 1. It has also been studied in [Par12] and admits at the cipher and decipher sides the respective equations:

$$\text{cipher: } \begin{cases} z_t = f_\kappa(c_{t-1}, \ldots, c_{t-n}) \\ c_t = z_t \oplus m_t \end{cases} \qquad \text{decipher: } \begin{cases} \widehat{z}_t = f_\kappa(c_{t-1}, \ldots, c_{t-n}) \\ \widehat{m}_t = \widehat{z}_t \oplus c_t \end{cases} \tag{1}$$

where n is the dimension of the shift register, $f_\kappa : \{0,1\}^n \to \{0,1\}$ denotes the filtering function parametrized by the secret key κ and for all instant t, $m_t \in \{0,1\}$ is the plaintext symbol, $c_t \in \{0,1\}$ is the ciphertext symbol, $z_t \in \{0,1\}$ is the keystream symbol. The quantity $\widehat{z}_t \in \{0,1\}$ is the keystream symbol on the deciphering side and $\widehat{m}_t \in \{0,1\}$ is the recovered plaintext symbol. If n ciphertext symbols are properly received, then the receiver recovers the plaintext since $\widehat{m}_t = m_t$ whenever $\widehat{z}_t = z_t$. Finally, the vector $x_t = (c_{t-1}, \ldots, c_{t-n})$ stands for the internal state.
NB: it is worth pointing out that all the results are established here in the Boolean case but still hold when considering any other finite alphabet.

The secret keys κ is some suitable parameters that select the function f among a family of filtering functions. The dimension of the shift register is given by the integer n. The keystream symbol z_t is the output of the filtering function. It only depends on the secret key κ shared by the cipher and the decipher and on n past values of the ciphertext. The ciphertext c_t is worked out from an exclusive

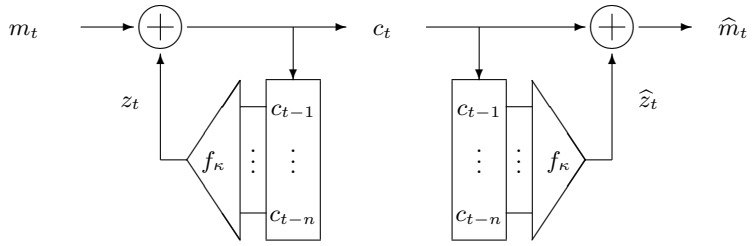

Fig. 1. Canonical form of a SSSC. Left: the ciphering. Right: the deciphering.

or of the plaintext symbol m_t and of the keystream symbol z_t. It is conveyed through the public channel. Since the function f shares, at the transmitter and receiver sides, the same values, namely the n past ciphertexts, it is clear that the keystreams synchronize automatically after a finite transient time of length n. As a result, the dimension n of the shift register defines the synchronization delay of the decipher.

The SSSC model (1) is actually a conceptual model that can be implemented by different architectures resulting from different design approaches. An overview has been given in [Dae95] and more recently in the survey paper [MG10]. The so called Cipher Feedback (CFB) mode of operation consists in building a closed-loop architecture involving both a shift register and a block cipher primitive. The mostly adopted method to implement a binary SSSC is to resort to a block cipher (AES for instance) in 1-bit CFB mode [oS80]. Nevertheless, this mode is quite inefficient in terms of encryption speed since one block cipher operation is required for encrypting a single plaintext bit. The state transition function is described by the shift register. It is very simple and is secret key independent. Therefore, the security relies entirely on the security of the filtering function. Maurer's approach is suggested in [Mau91] as an alternative. The most important concept of this approach is the use of several finite state automata in parallel and serial combinations. However, it is shown in [Dae95] that this method is not really relevant and a recursive approach is preferred. In order to guarantee the self-synchronization in finite-time, the state transitions functions of the recursive model proposed in [Dae95] are the T–functions (T for Triangle), which are functions that propagate dependencies in one direction only, for instance from left to right. In [MG10][PGM11], it has been shown that some concepts borrowed from control theory should appear as promising to design new SSSC architectures.

2.2 Encryption/decryption mechanisms of SSSC

In order to establish the security proof of the canonical SSSC, it is necessary to formally define the encryption and the decryption mechanisms.

2.2.1 Setup A random secret key κ is chosen in the keyspace for both the cipher and the decipher.

2.2.2 Encryption For encrypting a message m consisting in ℓ binary symbols, the steps are the following:

1. Choose randomly an n-dimensional binary vector as the initial state x_0 of the shift register.
2. Choose randomly n binary symbols to build the synchronization sequence and concatenate it with the message to be encrypted yielding a binary sequence $m_0, \ldots, m_{n+\ell-1}$. The first n symbols are those of the synchronization sequence, and the other ones are those of the message to be encrypted.
3. For each symbol at instant $t \geq 0$,
   (a) Compute the keystream symbol as $z_t = f_\kappa(x_t)$.
   (b) Compute the ciphertext symbol as $c_t = m_t \oplus z_t$.
   (c) Update the shift register state by shifting its components and feeding it by the computed ciphertext symbol c_t to obtain the next state x_{t+1}.

2.2.3 Decryption For decrypting the cryptogram consisting in $n + \ell$ binary symbols, the steps are the following:

1. Initialize the shift register state to any arbitrary value, for example the n-dimensional zero vector $\widehat{x}_0 = 0$.
2. For each symbol at instant $t \geq 0$,
   (a) Compute the keystream symbol as $\widehat{z}_t = f_\kappa(\widehat{x}_t)$.
   (b) Compute the plaintext symbol as $\widehat{m}_t = c_t \oplus \widehat{z}_t$.
   (c) Update the shift register state by shifting its components and feeding it by the computed ciphertext symbol c_t to obtain the next state $\hat{x}_{t+1}$.
3. The first n decrypted symbols correspond to the synchronization sequence and are ignored. The decrypted message consists of the ℓ last decrypted symbols.

3 Security proof of the canonical SSSC

3.1 Indistinguishability

The framework used here to assess the security of SSSC is the one defined in [BR05] for symmetric encryption schemes. The notions of security games is used within this framework. They are based on security concepts introduced by Goldwasser and Micali [GM82,GM84]. The first concept is the semantic security, which is a computational analogue to the Shannon perfect secrecy. However, the semantic security does not allow to fluently deal with security proof. This leads the authors in [GM82,GM84] to define another concept known as indistinguishability which better facilitates proving the security of practical cryptosystems.

A cryptographic scheme is said to be secure, in the sense of indistinguishability, if a polynomial complexity algorithm has a negligible advantage in guessing from which of both messages m_0 and m_1 it has provided, emanates the cryptogram returned to it. This is formalized by the so called IND-game.

IND-game. This game involves two players: an algorithm $\mathcal{A}$ called adversary or challenger, and an oracle. The game consists of the following steps:

1. The challenger chooses two messages m_0 and m_1 and provides the oracle with them.
2. The oracle randomly chooses a binary digit $b \in \{0, 1\}$, encrypts the message m_b and returns the cryptogram to the challenger.
3. The challenger is challenged to guess the value of b, that is which of the two messages has been encrypted. The answer of the challenger is a binary value $\widehat{b}$. The challenger wins if $b = \widehat{b}$.

Before giving its answer, the challenger can be supported by the oracle to decrypt ciphertexts the challenger chooses (Chosen Ciphertext Attack, CCA) or to encrypt plaintexts the challenger chooses (Chosen Plaintext Attack, CPA). Of course, in a CCA attack, the challenger is not allowed to request the decryption of the challenge it has received.

For any b and $\widehat{b}$ in the set $\{0, 1\}$, let $\text{Prob}_{m_b}(\mathcal{A} = \widehat{b})$ be the probability that the challenger $\mathcal{A}$ answers $\widehat{b}$ when the ciphertext corresponds to the encryption of the message m_b. The advantage of $\mathcal{A}$ in this IND-game is by definition:

$$\text{Adv}(\mathcal{A}) = \Big|\text{Prob}_{m_0}(\mathcal{A} = 0) - \text{Prob}_{m_1}(\mathcal{A} = 0)\Big|.$$

As m_0 and m_1 are randomly chosen with the same probability $1/2$, it holds that

$$\text{Adv}(\mathcal{A}) = \Big|2\text{Prob}(b = \widehat{b}) - 1\Big|,$$

where $\text{Prob}(b = \widehat{b})$ denotes the probability that $\mathcal{A}$ wins.

If the challenger answers at random, then its advantage is null. If it always wins, or always looses, then its advantage equals 1.

A cryptographic scheme is defined by a security parameter, which is for example the key size. Such a scheme is said to be IND-secure if the advantage of the challenger is negligible. A function of the real number x is said to be negligible if it decreases faster that the inverse of any polynomial in x as x grows to infinity. A cryptographic algorithm is said to be IND-CCA if it is IND-secure during a CCA attack. It is said to be IND-CPA if it is IND-secure during a CPA attack.

For the canonical SSSC, the security parameters are the size of the shift register which equals the number of inputs of the filtering function and the size of the secret key.

3.2 IND-CCA security

The ideal case for a canonical SSSC is when the filtering function is randomly chosen among all the 2^{2^n} Boolean functions. This ideal case is not realistic as it would imply an exponential key size. However, even in this situation, the following property holds:

Proposition 1 *The canonical SSSC cannot reach the IND-CCA security*

Proof. Consider the special situation when the challenger provides $m_0 = 0 \cdots 0$, the message that only involves symbols 0, and $m_1 = 1 \cdots 1$, the message that only involves symbols 1. It receives a cryptogram c. It modifies only the last symbol of the cryptogram and asks the oracle to decrypt. If the answer of the oracle starts with a sequence of 0, then c is the encryption of m_0 and if it starts with a sequence of 1, then c is the encryption of m_1. Following this strategy, the challenger always wins. That suffices to complete the proof.

Actually, this is due to the fact that the cryptograms produced by an SSSC are malleable. Indeed, they can be modified at their end without any impact on the beginning of the plaintext.

The only security level that can be expected for the canonical SSSC is the IND-CPA as shown in next section.

3.3 IND-CPA security of the canonical SSSC

The main result of this section is to prove that, when the filtering function is a so called Pseudo Random Function (PRF), then the canonical SSSC reaches the IND-CPA security. Let us first recall what a Pseudo Random Function is.

3.3.1 Pseudo Random Functions. A Pseudo Random Function is an element randomly chosen in a family of Pseudo Random Functions. And a family of functions is said to be Pseudo Random if it is computationally undistinguishable from the set of all the functions. This means that for polynomial complexity adversaries, a Pseudo Random Function behaves as if it was a true randomly chosen function.

The pseudo random property is assessed by the so called PRF-game that involves an adversary $\mathcal{B}$, challenged to guess whether a given function is either a true random function, randomly chosen in the set of all the functions, or is an element of the family. The entries of the algorithm $\mathcal{B}$ are an integer n together with an oracle that, on request for an n-dimensional vector, returns the value of the function.

PRF-game. The element of the game for the parameter n is a family of functions $f_\kappa : \{0,1\}^n \to \{0,1\}$. The steps of the game are the followings:

1. The oracle chooses a random bit $b \in \{\mathrm{rf}, \mathrm{prf}\}$. If $b = \mathrm{prf}$ (pseudo random world) then it randomly chooses a function $f = f_\kappa$ in the family of pseudo random functions. If $b = \mathrm{rf}$ (random world), it randomly chooses a Boolean function f in the set of all 2^{2^n} Boolean functions $\{0,1\}^n \to \{0,1\}$.
2. The challenger asks the oracle for the values of $f(x_i)$ for inputs $x_1, \ldots, x_q$. The number q of queries the challenger is allowed to perform is bounded by a polynomial in n.

3. After the q queries, the challenger must answer a binary value $\widehat{b}$ that means that the challenger guesses that the chosen function was pseudo random ($\widehat{b} = \text{prf}$) or was purely random ($\widehat{b} = \text{rf}$). If $\widehat{b} = b$ then the challenger wins.

The adversary $\mathcal{B}$ does not know in which world it plays, pseudo random or random world? Let $\text{Prob}_{\text{prf}}(\mathcal{B} = \text{prf})$ be the probability that $\mathcal{B}$ answers prf when it plays in the pseudo random world and let $\text{Prob}_{\text{rf}}(\mathcal{B} = \text{prf})$ be the probability that $\mathcal{B}$ answers prf when it plays in the random world. In the first case, the answer of $\mathcal{B}$ is correct, and in the second case, it is wrong. The advantage of $\mathcal{B}$ in the PRF-game is by definition:

$$\text{Adv}(\mathcal{B}) = \left|\text{Prob}_{\text{prf}}(\mathcal{B} = \text{prf}) - \text{Prob}_{\text{rf}}(\mathcal{B} = \text{prf})\right|$$

A family of Boolean functions is said to be pseudo random (PRF) if the maximum advantage of any adversary is negligible. *N.B.* For a family of functions to be PRF, it is necessary that its number of elements increases faster than any polynomial in the security parameter n. If not, the exhaustive search algorithm has polynomial complexity. In cryptographic context, the secret key selects an element of the family. So, the key size must be greater that the logarithm of any power of n. In practice, an n-bit long key is admissible.

3.3.2 Main result.

Let us first state our main result.

Proposition 2 *If the filtering function of a canonical SSSC is a Pseudo Random Function, then it reaches the IND-CPA security.*

Before proceeding to the detailed proof, we explain the main lines of the reasoning which are, by the way, usual for security proof. It must be shown that any adversary $\mathcal{A}$ against the SSSC has a negligible advantage. Let $\mathcal{F}$ be the family where the filtering function is chosen. This family is assumed to be PRF. Then, an adversary $\mathcal{B}$ against the family $\mathcal{F}$ is constructed, using the adversary $\mathcal{A}$ as a subroutine. In orther words, $\mathcal{B}$ orchestrates the IND-game of $\mathcal{A}$ against the SSSC. The game played by $\mathcal{B}$ is the PRF-game. The advantage of $\mathcal{B}$ is evaluated and lower bounded by the advantage of $\mathcal{A}$, up to a negligible function $n \mapsto \varepsilon(n)$:

$$\text{Adv}(\mathcal{A}) \leq 2\text{Adv}(\mathcal{B}) + \varepsilon(n). \tag{2}$$

As the family $\mathcal{F}$ is assumed to be pseudo random, the advantage of $\mathcal{B}$ is negligible, and so is the advantage of $\mathcal{A}$. Thus, the canonical SSSC reaches the IND-CPA security. We are now able to proceed to the detailed proof of Proposition 2.

Proof. The proof involves two main steps.

Step 1: Construction of an adversary $\mathcal{B}$ against $\mathcal{F}$. Let $\mathcal{A}$ be an IND-CPA adversary against the SSSC. Let us introduce a PRF adversary $\mathcal{B}$. The adversary $\mathcal{B}$ will act as an oracle for $\mathcal{A}$ and will call upon the function oracle to produce ciphertexts on request. If $\mathcal{A}$ distinguishes correctly which message has been encrypted,

then it may be supposed that the ciphertexts produced by $\mathcal{B}$ are correctly built with the pseudo-random function and it returns prf, otherwise it returns rf. The precise definition of algorithm $\mathcal{B}$ is the following:

1. Algorithms $\mathcal{B}$ receives from $\mathcal{A}$ two plaintexts m_0 and m_1. It picks up a random bit $b \in \{0,1\}$. If $b = 0$, then it encrypts m_0, otherwise it encrypts m_1. The result c is returned to $\mathcal{A}$.
2. For each message encryption, $\mathcal{B}$ uses the equation of the SSSC and calls for the function oracle to get the value of the keystream symbol and thus, the value of the ciphertext symbol.
3. After a polynomial time, $\mathcal{A}$ returns the value $\widehat{b}$. If $\widehat{b} = b$, that is $\mathcal{A}$ answers correctly, then $\mathcal{B}$ returns prf, else, if $\widehat{b} \neq b$, that is $\mathcal{A}$ answers wrongly, then it returns rf.

Step 2: Bound for the advantage of $\mathcal{A}$. We aim at proving the inequality (2), where ε is a negigible function $n \mapsto \varepsilon(n)$.
By definition of the advantage of an adversary in the PRF-game and from the above definition of the algorithm $\mathcal{B}$, one has:

$$\begin{aligned} \mathrm{Adv}(\mathcal{B}) &= \left|\mathrm{Prob}_{\mathrm{prf}}(\mathcal{B} = \mathrm{prf}) - \mathrm{Prob}_{\mathrm{rf}}(\mathcal{B} = \mathrm{prf})\right| \\ &= \left|\mathrm{Prob}_{\mathrm{prf}}(b = \widehat{b}) - \mathrm{Prob}_{\mathrm{rf}}(b = \widehat{b})\right| \end{aligned}$$

In the prf world, the algorithms $\mathcal{B}$ acts exactly as a true SSSC cipher and thus, the probability that $\mathcal{A}$ wins is exactly the probability that $\mathcal{A}$ answers the value chosen by algorithm $\mathcal{B}$:

$$\begin{aligned} \mathrm{Adv}(\mathcal{A}) &= \left|2\mathrm{Prob}_{\mathrm{prf}}(b = \widehat{b}) - 1\right| \\ &\leq \left|2\mathrm{Prob}_{\mathrm{prf}}(b = \widehat{b}) - 2\mathrm{Prob}_{\mathrm{rf}}(b = \widehat{b})\right| + \left|2\mathrm{Prob}_{\mathrm{rf}}(b = \widehat{b}) - 1\right| \end{aligned} \tag{3}$$

The first term of the righthandside of the above inequality is twice the advantage of $\mathcal{B}$. It remains to prove that the second term is negligible. While the algorithm $\mathcal{B}$ runs, two situations may happen. Either all the inputs x_i in the queries to the function oracle are different or there exists one or several collisions in those inputs. Let us denote with E the former event, that is there is no collision in the queries to the function oracle. The converse event is denoted by $\overline{E}$. As E and $\overline{E}$ are disjoint, one has:

$$\mathrm{Prob}_{\mathrm{rf}}(b = \widehat{b}) = \mathrm{Prob}_{\mathrm{rf}}(b = \widehat{b} \text{ and } E) + \mathrm{Prob}_{\mathrm{rf}}(b = \widehat{b} \text{ and } \overline{E}) \tag{4}$$

When the event E occurs, in the rf world, the answers of the function oracle are purely random and independent. The ciphertexts that $\mathcal{B}$ returns to $\mathcal{A}$ are merely random and in this case, $\mathcal{A}$ cannot have any advantage. It results that:

$$\mathrm{Prob}_{\mathrm{rf}}(b = \widehat{b} \text{ and } E) = \frac{1}{2}. \tag{5}$$

From (4) and (5), it is straightforward to see that

$$\left|2\mathrm{Prob}_{\mathrm{rf}}(b=\widehat{b})-1\right| = 2\left|\mathrm{Prob}_{\mathrm{rf}}(b=\widehat{b} \text{ and } \overline{E})\right| \tag{6}$$

Regarding the event $\overline{E}$, *i.e.* if there exist collisions in the queries of $\mathcal{B}$, then the ciphertexts provided to $\mathcal{A}$ cannot be considered as purely random. One has:

$$\mathrm{Prob}_{\mathrm{rf}}(b=\widehat{b} \text{ and } \overline{E}) = \mathrm{Prob}_{\mathrm{rf}}(b=\widehat{b} \mid \overline{E}) \times \mathrm{Prob}(\overline{E}). \tag{7}$$

Finally, it suffices to prove that the probability of the event $\overline{E}$ is negligible. Let q be the number of messages that the algorithm $\mathcal{B}$ encrypts, and ℓ be the largest length of a message. Thus, the number of queries of $\mathcal{B}$ to the function oracle is upper bounded by $q \times \ell$.

During the encryption of a single message, the inputs sent to the function oracle are the internal state of the n dimensional shift register. They follow a path in the so called De Bruijn graph. Let us denote by $X = (x_1, \ldots, x_\ell)$ and $Y = (y_1, \ldots, y_\ell)$, two paths of length ℓ in this graph. As the initial state of the shift register is random, the states of paths X and Y are random (but not independent). We have that

$$\mathrm{Prob}(X \cap Y \neq \emptyset) \leq \sum_{i=1}^{\ell} \mathrm{Prob}(y_i \in X)$$

This sum involves ℓ terms, all of them being less than or equal to $\ell/2^n$, thus

$$\mathrm{Prob}(X \cap Y \neq \emptyset) \leq \frac{\ell^2}{2^n}.$$

The probability of event $\overline{E}$ is the probability that there exist at least two paths among the q paths that correspond to the encryption of the q messages which collide. Thus, one has:

$$\mathrm{Prob}(\overline{E}) \leq \frac{q(q-1)}{2} \times \frac{\ell^2}{2^n}$$

As q and ℓ are bounded by a polynomial in n, this probability is negligible and (2) is fulfilled. That completes the proof.

4 Concluding remarks

The cryptological complexity of the canonical form of the Self-Synchronizing Stream Cipher lies in the filtering function. The maximum security is achieved when this function is a pure random function. It has been shown in this paper that, even in this ideal case, this kind of cipher cannot reach the IND-CCA security. This is due to the fact that the ciphertexts are malleable. Thus, the maximum expected security is IND-CPA. In realistic implementation, the filtering is a pseudo random function, for example a single output of a block cipher

primitive. We have proved that the IND-CPA security is reached. The interest of the result lies in that it guarantees the existence of SSSC that can be IND-CPA secure although till now, the SSSC proposed in the open literature had be broken against IND-CPA attacks.

Future work will aim at relaxing the pseudo randomness assumption. Indeed, it is not mandatory. For example, clearly, functions built from a class of pseudo random functions where every element f satisfies $f(0) = 0$ does not define a class of pseudo random functions. And yet, for an SSSC built from this class, the IND-CPA security is guaranteed. The same holds if a polynomial number of values are fixed in each element of the class. As a result, checking minimal properties for the filtering function to guarantee IND-CPA security is still challenging.

Acknowledgement
This work was partially supported by Research Grants ANR-13-INSE-0005-01 from the Agence Nationale de la Recherche.

References

BR05. Mihir Bellare and Phillip Rogaway. Introduction to modern cryptography. In *UCSD CSE 207 Course Notes*, page 207, 2005.

Dae95. J. Daemen. *Cipher and Hash function design, strategies based on linear and differential cryptanalysis.* PhD thesis, Katholieke Universiteit Leuven, 1995.

DGV92. Joan Daemen, René Govaerts, and Joos Vandewalle. A practical approach to the design of high speed self-synchronizing stream ciphers. In *SINGAPORE ICCS/ISITA '92*, p. 279–293. IEEE, 1992.

DK08. J. Daemen and P. Kitsos. The self-synchronizing stream cipher moustique. In *New Stream Cipher Designs - The eSTREAM Finalists*, p. 210–223. 2008.

GM82. Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA*, pages 365–377, 1982.

GM84. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

Mau91. Ueli M. Maurer. New approaches to the design of self-synchronizing stream ciphers. In *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, pages 458–471, 1991.

MG10. Gilles Millerioux and Philippe Guillot. Self-synchronizing stream ciphers and dynamical systems: State of the art and open issues. *I. J. Bifurcation and Chaos*, 20(9):2979–2991, 2010.

MvOV96. Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1996.

oS80. National Bureau of Standards. Des mode of operations. Technical report, Institute for Computer Sciences and Technology, National Bureau of Standards, Springfield, VA, Decembre 1980.

Par12. J. Parriaux. *Control, synchronization and encryption.* PhD thesis, Université de Lorraine, 2012.

PGM11. J. Parriaux, P. Guillot, and G. Millérioux. Towards a spectral approach for the design of self-synchronizing stream ciphers. *Cryptography and Communications*, 3:259–274, 2011. 10.1007/s12095-011-0046-2.

Sar03. Palash Sarkar. Hiji-bij-bij: A new stream cipher with a self-synchronizing mode of operation. *IACR Cryptology ePrint Archive*, 2003:14, 2003.