



Use of the Shrinking Generator in Lightweight Cryptography for RFID

Raja Ghosal and Peter H. Cole

Auto-ID Labs White Paper WP-SWNET-024

| | |
|---|---|
|  | <p>Mr. Raja Ghosal PhD Student, Auto-ID Lab, ADELAIDE School of Electrical and Electronics Engineering, The University of Adelaide</p> |
|  | <p>Prof. Peter H. Cole Research Director, Auto-ID Lab, ADELAIDE School of Electrical and Electronics Engineering, The University of Adelaide</p> |
| <p>Contact: rghosal@eleceng.adelaide.edu.au or cole@eleceng.adelaide.edu.au. Internet: www.autoidlabs.org</p> | |

Abstract

RFID tags have severe constraints in computing power and hence offer particular challenges in the provision of e-Security. Whatever is chosen to provide security in an RFID tag should have low computational overhead. New approaches, differing from the traditional cryptosystems based on RSA, Diffie-Hellman, et al. are required. The use of one time codes is particularly appropriate as they guarantee perfect security and offer simple implementation. Research, experimentation, and field deployment in RFID has been done using different types of algorithms including bit shift, xor operations, pseudorandom bit generators including linear feedback shift registers (LFSR).

There are many different types of pseudorandom bit generators. Some use RSA, or discrete logarithm like arithmetic, but use the last bit of the generated sequence as the random bit. These are infeasible in RFID and other lightweight devices due to high computational loads arising from the complex arithmetic operations. Another approach to the generation of random sequences is to use a combination of LFSRs. These are used in portable devices such as the GSM mobile phones. They have lightweight computational loads, but are susceptible to attacks. Of the many ingenious combinations of LFSRs, the Shrinking Generator, designed in 1993, seems to have withstood the challenges of attacks if the polynomial connection structure and the internal seeds are kept secret.

1. Introduction

There has been much research in bit stream pseudo random number generators (PRNG) to achieve near One-Time Codes [8]. This is to enable secure keystreams to encrypt messages. The message is encrypted by bit xor'ing with the keystream. The receiver then decrypts via a reverse xor'ing of the same keystream. Shannon had shown that the only codes which provided perfect secrecy were One-Time Codes [8,14,22,24]. Such codes are often generated in complex apparatus from galactic or quasar noise, or some thermal or electronic noise. To eliminate such complexity, research has been directed towards obtaining pseudo-random numbers or bit streams which exhibit near one-time codes behaviour. This makes breaking-in difficult for an intruder, contrasting with the security provided in GSM mobiles that are susceptible to attacks [2]. There are many candidate bit generators. Of these the Shrinking Generator, a very simple system based on 2 LFSR's, has some excellent properties for use as a lightweight secure key stream source, and thus far withstood attacks if the tap polynomials and the initial seeds are kept secret (or treated as private keys) [3,17].

1.1 Why Bit Oriented Generators

Byte or block oriented cryptographic methods, or pseudo random number generators such as those using the RSA, Diffie-Hellman, or Elliptic Curve Cryptography (ECC) methods [13,14] have a high computational complexity. They are not suitable in contexts which provide only

the limited computational powers in lightweight devices, including RFID tags. Yao, Blum, and other researchers [14] have presented research in the area of pseudo-random bit generators (PRBG). The most used choice is based on LFSRs. Menezes et al [22] give the definition of a random bit generator as a device or an algorithm whose output is a sequence of bits (binary digits), which are statistically independent and unbiased. Further, Menezes et al [22] define a PRBG as a deterministic algorithm, which taking an input of a pure random binary sequence of some length m , outputs a binary sequence of length n , where $n \geq m$. The input is the seed. The output, pseudorandom bit sequence “appears” to be random. The deterministic component of the algorithm [22] is that if the same seed is input, the output will correspondingly be the same, on each occasion. The definition of pseudorandomness is given later in this section.

One-Time Codes are the only provably secure random codes (Shannon, 1949) [13,14, 22]. Their limitation is, as key stream, they should be at least of the same length as the plaintext message (the key stream is XOR'd with the plaintext to produce the ciphertext). LFSRs are promising as they output pseudo-random bit sequence, and are much shorter (in bit length). The shortcoming is that LFSR sequences can be easily broken. Linearity as a consequence of use of LFSRs, is a weakness in cryptography. A single LFSR can be broken into (such as finding the secret initial seed) by solving a set of linear equations over $GF(2)$, such as via Gaussian elimination [15, 24]. A system of equations in non-linear variables over $GF(2)$ is NP, and hence a hard problem to solve [15]. This was the foundation of the Lucifer system, the forerunner of the DES system [15], and further trends to the current AES system [10]. These systems are based on bit wise operations xor, shift, rotations which can be represented via such system of non-linear equations [15].

As an example, an input X , XOR'd with input Y is represented as “ $X + Y$ ”, while and input X AND'd with input Y is represented as “ $X.Y$ ” [15, 24]. “ $X.Y + Z = K$ ” is a non-linear equation, and is also referred to as a “quadratic equation” as there is a $X.Y$ term (a $X.Y.Z$ term makes an equation “cubic”). “ $X+Y = M$ ” is a linear equation. In “attacks”, in the right hand side of the equations above such as K and M , would represent the observable output ciphertext bit values. The intruder observing, has to solve the equations above to obtain values for the internal parameters, X , Y , Z etc. The system of equations are easy if the internal LFSR bit lengths and connections are known, such as in GSM mobile phones [6]. Otherwise, the intruder, may have to observe long pattern of ciphertext bit streams to arrive at possible inferred set of equations [24]. Chosen plaintext, or ciphertext attacks [6, 13, 14, 15, 22, 24] could prune, or reduce the permutations, of the variables to infer a system of equations to solve, as the objective of the attack. There are many other methods [22], such as divide and conquer attacks where only one LFSR at a time is “attacked” via such methods, such as for the Geffe generator [14, 24]. The mathematical basis of algebraic attacks, based on design of the system of equations in linear or non-linear variables is well explained in [6, 15, 24]. The solution of such equations are used to obtain the initial seeds (or initial values (IV)) and other useful information of the LFSRs as a consequence of such attacks [6, 15, 24]. Such system of equations are developed after prudent observations of the output ciphertext bit stream by an intruder [6, 15, 24].

Several combination of LFSRs, are required to introduce non-linearity. Clocking of one LFSR output to the other is a method of introducing non-linearity. This is used in the Alternating Step Generator (ASG), Shrinking Generator (SG). ASG (whose order of security, for k bit registers is (2^k) requires 3 LFSRs of about 128 bits, while the SG (whose corresponding order of security is (2^{2k}) requires 2 LFSRs of about 64 bits each bits for security against all

known attacks [3,22]. The Shrinking Generator is very simple, robust method, based on 2 LFSRs if initial seeds and connection polynomials are kept secret [3, 22].

Goldreich [6] has shown that an ensemble $\{X_n\}_{(n \in \mathbb{N})}$ is pseudorandom if and only if it is unpredictable in Polynomial Time (P). Welsh [15] has shown that a pseudo-random number generator passes all Polynomial-Time statistical tests if and only if it passes all Polynomial Time (P) next-bit tests. The next-bit test due to Yao, is the inability to predict the next bit as 0 or 1 with a probability > 0.5 , given the preceding bit pattern [6, 15, 22].

Yao also defined “Effective Entropy” [15]. For example. the integer factorization problem as used in the RSA algorithm, in theory, is solvable, and hence the entropy = 0. In practice, given the current computing resources, the integer factorization problem with prudently chosen large random prime numbers, is intractable, and hence leads to some “effective entropy”. Bit stream generators must satisfy the two major criteria of: (a) unpredictability, and (b) balance [15, 22, 24]. Balance refers to the number of 0’s and 1’s in an output sequence being equal.

2. Types of Bit Stream Generators

There are many bit oriented stream generators [10,13,14, 22] that have been designed to be difficult for adversaries to attack. They are generally based on computationally hard problems or non-linear Boolean algebra. This list includes:

(a) the Linear Congruential Generator; (b) the Blum-Blum-Shub (BBS) Generator; (c) the Geffe Generator; (d) the Alternating Step Generator; (e) the Shrinking Generator; (f) the Self-Shrinking Generator; (i) the Rabin Generator; (h) the RSA generator; and (i) the Discrete-Logarithm Generator.

All the arithmetic based schemes output the lowest significant bit as the keystream. They use an iterative method, where the next value depends on the function applied as per the generator’s arithmetic rules. The Linear Congruential generator is a very simple scheme using a linear function over modulus arithmetic field [14]. The Blum-Blum-Shub generator is the most used in practice. It is based on the hardness of quadratic residues and the integer factorization over modulus field arithmetic [13,14]. The Rabin Generator is based on the hardness of the quadratic residues over modular arithmetic [13,14]. The Geffe generator [14, 22] is a simple logic based generator, which is a combination of 3 LFSRs, connected via AND and OR gates. This logic structure makes it prone to correlation attacks [14, 22, 24]. The RSA generator uses the underlying RSA arithmetic, which is based on the hardness of integer factorization over modular field [10,13,14]. The Discrete Logarithm Generator uses the underlying discrete logarithm arithmetic over modular fields [14]. The Alternating Step Generator (ASG) [22] uses a single clock and 3 LFSRs connected via AND and NOT gates. The Shrinking Generator (SG) appears to provide the simplest structure and most secure output.

Both, the ASG, and the SG, are clock controlled generators. In such clock controlled generators, non-linearity is introduced by having the output of one LFSR clock (or step),

sampled in accordance with a second LFSR, or having the clocking of some generators controlled by the output of others. In the SG, the additional non-linearity of shrunken sequence of the output is introduced. The Self-Shrinking Generator (SSG) is a simplified version of the Shrinking Generator, and does not require a clock [22]. In that generator a predefined combination of bits gives rise to a shorter or a shrunken output, for example a bit stream 11 generated may give rise to an output 1. However the fairly simplified structure makes this generator vulnerable to attacks. However some crypto analyses of the self-shrinking generator also consider the shrinking generator and hence provide valuable inputs for study of the Shrinking Generator [16].

3. Attacks on the LFSR based GSM Mobile A5/X, GPRS security schemes

This section has been included to show the weakness of some LFSR schemes, and because it outlines some attack strategies that may be directed toward other schemes.

Barkan et al. (referenced in [2]) in 2003 showed how the A5/2 used in GSM mobile phone communications may be broken. In their more recent paper 2006 [2], Barkan et al. generalize the attacks to all the GSM's encryption algorithms A5/1, A5/2, A5/3 (or in general A5/X), and also GPRS used for data communications in the GSM mobile network.

In the more recent work, [2], those authors have shown how an attack on the GSM system based on LFSRs can be mounted. The sizes of the 3 to 4 LFSRs used in the GSM system are in the range of 17 to 23 bit registers [19, 22]. They use a method of solving linear quadratic equations [15, 24] to obtain the keystream (as outlined in sec 1.1). This method can be extended to several other methods, which are based on LFSR's, and perhaps also to the Shrinking Generator if secrecy of the structure is not maintained. The methods and the ideas are covered by patent and a written authorization is required from the authors for their use.

The authors [2] point out that the GSM's security scheme designed in the 1980's, was the first developed security method in the public telephony domain, due to then newly emerging public key cryptography. They have also indicated that the security design was then influenced by 1980's environment, where civilians were not permitted to use strong cryptography. Hence the development of then LFSR based light cryptography methods.

The authors take advantage of the fact that the error correcting part of the transmission is done before the encryption. Hence in the encrypted text already some redundant information is available. Supposing the parity bit obtained by XOR'ing of a subset of bits is 0. Then XOR'ing the same subset of bits, in the encrypted bits, shall help obtain the parity of the corresponding keystream bits [2].

4. The Shrinking Generator

Coppersmith et al [3] first presented a design of the shrinking generator. As shown in Fig. 1, this was based on two Linear Feedback Shift Registers (LFSR)'s, S (selector), and the A (data) registers. If the output of the S register is 1, then the output of the A register becomes part of the keystream. If the output of the S register is 0, then the output of the A register is discarded [3,14,17].

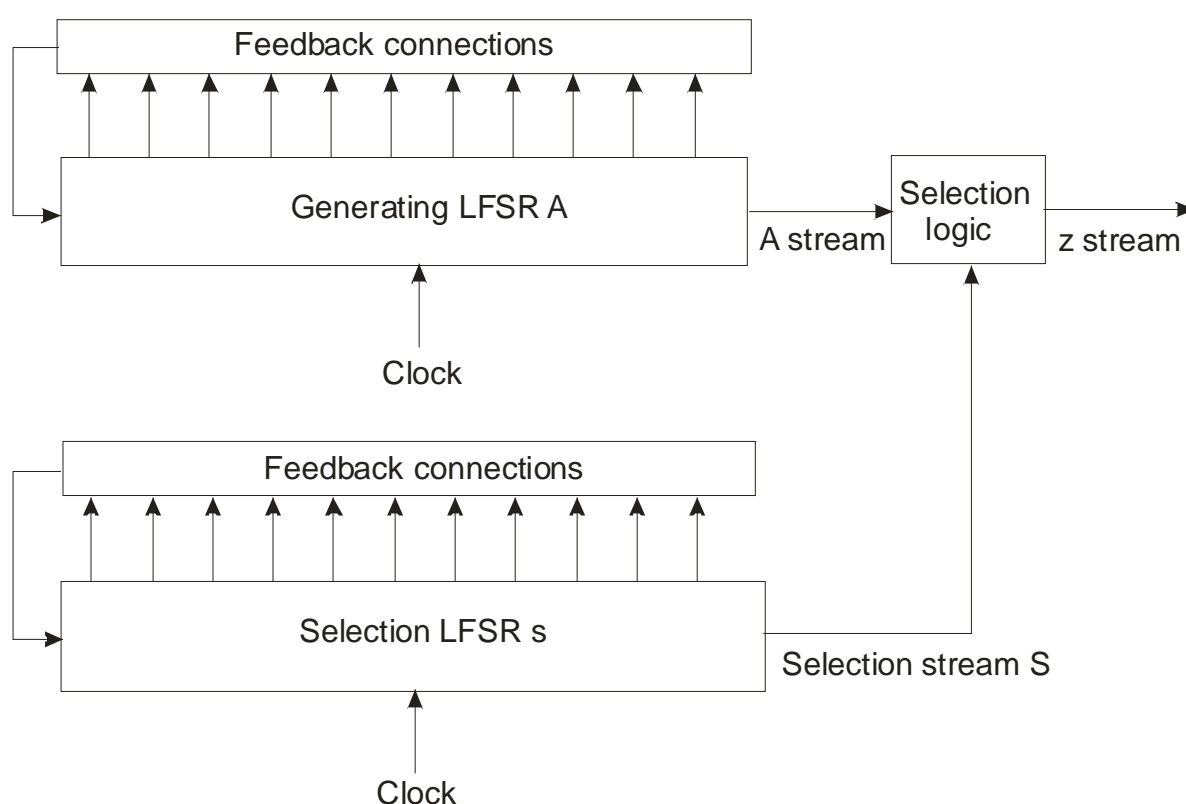


Fig. 1. The Shrinking generator.

This is both a simple hardware implementation, and provides an output, the z stream, that has lost the linearity that leads to the insecurity of the simple LFSR schemes. Its practical strength has become clear only after years of public scrutiny. There is scope for further research into the theoretical basis of its security and into test procedures.

The original contribution of Coppersmith et al, is that the Shrinking Generator has the good random like properties that exist in LFSR's, and important properties not present in LFSR sequences, such as the Exponential Linear Complexity. The standard tests for bit streams, such as pseudorandomness, and predictability such as Yao's next bit test, are passed by the Shrinking Generator [3,14]. Coppersmith et al. [3] show period of the keystream, z-sequences is exponential in both the lengths of the A and the S sequences. They show that their scheme passes the minimal tests, as indicated above.

Coppersmith et al [3] suggest generalization to any pair of pseudorandom sources. It is open to analysis if any other similar linear structures of LFSR's exist maybe also pseudo random, close to one-time codes.

Those authors define some of the necessary statistical properties for pseudo random number generator: low correlation between sequence bits; normalized appearance of 0's and 1's; and balanced distribution of sub-patterns. They also perform Fourier and ϵ -analysis. Coppersmith et al. say the above properties are only necessary (but are not enough for sufficiency), for conditions on the cryptographic strength of the pseudorandom generators.

The period and linear complexity bounds are proven mainly through algebraic techniques [3, 22].

If both LFSR's, S and A use primitive polynomials, then the periods of the LFSR's are [15]: $L_S = 2^S - 1$, and $L_A = 2^A - 1$. where S denotes the size of the LFSR S, and A the size of the LFSR A.

Some of the conditions that must be satisfied, in a secure design are:

- (a) The periods of the two LFSR's must be co-prime, or $\gcd(L_S, L_A) = 1$.
- (b) Use of secret connection polynomials that are not sparse.
- (c) Use of maximum length LFSR's for the LFSR A and S in the Shrinking Generator.
(Using primitive polynomials for A and S ensures maximum length)

Some of the properties of the Shrinking Generator are [3, 22]:

- (a) If $\gcd(L_S, L_A) = 1$, then the output sequence, z, has a period $(2^A - 1) \cdot (2^S - 1)$
- (b) The linear complexity $L(z)$, of the output sequence z, satisfies:
$$L_A \cdot (2^S - 2) < L(z) \leq L_A \cdot (2^S - 1)$$

Additional relevant properties are outlined in [22]. The linear complexity $L(z)$, is the length of the smallest LFSR that can produce the output bit stream [22]. For example for a bit stream, "0000...00", or "111...11", $L(z) = 0$. If a bit stream cannot be produced by an LFSR, then $L(z)$ is infinity [22]. One-Time codes, perhaps belong to the latter class. In general observing any output bit stream over certain interval, a reasonable estimate of $L(z)$ of the size of the LFSRs could be deduced. From this information, the connection polynomials and initial seeds can be derived via algebraic attacks as briefly introduced in sec 1.1. However, if the length of the LFSRs are chosen large as recommended below, for security [3, 22], the length of the ciphertext bit stream to be observed, will become so large, that this is infeasible.

The following are some metrics that reflect the security of the Shrinking Generator [3, 22]:

- (a) If the connection polynomials for LFSRs, S and A, are known, then the best attack known for recovering the secret keys takes $O(2^{L_A} \cdot L_S^3)$.
- (b) If the secret (and variable) connection polynomials are used, the best attack known takes $O(2^{L_A} \cdot L_A \cdot L_S)$

(c) There is also an attack through the linear complexity of the shrinking generator which takes $O(2^{L_A} \cdot L_S^2)$ steps irrespective of whether the connections are known or are secret. This attack however requires $2^{L_A} \cdot L_S$ consecutive bits from the output sequence, z . This is infeasible for moderately large L_A and L_S .

Menezes et al [22], illustrate the security of the shrinking generator by an example as follows. If secret connections are used, and $A = k$ bits, and $S = k$ bits, then the shrinking generator has security of the order 2^{2k} . If $A = 64$ bits, and $S = 64$ bits, then the shrinking generator appears to be secure against all presently known attacks.

5. Prediction of the Shrinking Generator

Ekdahl et al [5] have shown an ingenious scheme to predict the output of a shrinking generator with known connections, given a sufficient number of observations of the output. They have applied a system of mathematical linear equations on similar lines as Barkan et al [2]. The paper [5] exploits newly detected non-randomness in the distribution of output of the generated key stream. The method is not applicable to a shrinking generator with unknown feedback polynomial for the generating LFSR. They have used the imbalance property in a pseudo-random bit stream output. They define the imbalance, as the difference between the number of 0's and the number of 1's in a period.

6. Attacks on the Shrinking Generator

There have been several methods of attack proposed on the Shrinking Generator.

- (a) Use of Cellular Automata to attack the SG [21, 23].
- (b) Basic Divide and Conquer method which requires an exhaustive search through possible initial states and feedback polynomials of the Selector LFSR as presented in [3, 4].
- (c) Correlation Attack targeting LFSR A is proposed in [6]

7. Limitations of the Shrinking Generator in Cryptography

The statistical properties, viz. low correlations, distributions of 1's and 0's, and randomness depend on the choice of the size of the LFSR's and the initial seeds. There can be situations where the randomness is not close to a near one-time code, as desired for a pseudo random

bit generator. There is scope for research in the patterns, cycles, of the S and A registers, in particular the properties of the S register.

Welsh [15] has pointed out that in a LFSR based on a primitive polynomial, with the size of register b , the period is $2^b - 1$. Welsh [15] also points out if there is a stream of n consecutive 1's in a period there shall be n consecutive 0's in the period, given the initial bit patterns.

It is possible there can be a long stream of 0's, from the S (selector) register. This means there will be no output from the A (data) register for a long time, as part of the keystream. This is a severe limitation of the shrinking generator. One possible solution, so far unexplored in the literature, is that if after a fixed number, n , of 0's, from S, the output from A is allowed to proceed to the keystream to avoid a gap. What is an optimal number, n , of 0's, such that the gap is filled in this way requires considerable analysis. Another possible direction is when there is a corresponding long stream of 1's from the S register. In this case, the linear complexity of the system is reduced to that only of the single LFSR A (data register) [15]. Moreover a single LFSR being linear, is weak cryptographically. Hence there may be scope for some research of prudently introducing extra 0's in between a long stream of 1's from the S register, for greater randomness of the output. Another possible direction may be prudent choice of lengths and connection polynomials for the S register, so that the long "silences" due to long stream of 0's from S register, and correspondingly long streams of 1's [15], is reduced to some acceptable limits.

8. Conclusions

The Shrinking Generator offers a promising approach in the areas of public-key or lightweight cryptography, and especially in securing RFID systems. Wikipedia [17] indicates that, subject to elementary precautions in designing the structure, the shrinking generator has not been broken into last 12 years.

There is scope for considerable research to more firmly establish the security and to remove the potential for the generator to have an unavailable output.

References:

[1] Avoine, Gildas (2006): "Bibliography on Security and Privacy in RFID Systems", MIT; Cambridge, Massachusetts; Retrieved May, 17, 2006 from <http://lasecwww.epfl.ch/~gavoine/rfid/>

[2] Barkan , Biham E., and Keller N. (2006): "Instant Ciphertext-Only Cryptanalysis on GSM Encrypted Communications"; Retrieved July, 17, 2007 from:<http://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-get.cgi/2006/CS/CS-2006-07.pdf>

[3] **Coppersmith, D., Krawczyk H., and Mansour, Y. (1994):** “The Shrinking Generator”, Proceedings of Crypto-93, LNCS Vol 773, Springer-Verlag, 1994, pp. 22-39.

[4] **Coppersmith, D., Halevi , and Jutla (2002):** Cryptoanalysis of Stream Ciphers with Linear Masking”, CRYPTO 2002, LNCS, vol 2442, Springer-Verlag

[5] **Ekdahl, P., Meier, W., and Johansson T. (2003):** “Predicting the Shrinking Generator with Fixed Connections”, in Proceedings EuroCrypt 2003 (ed E. Biham), LNCS 2656, Springer-Verlag, pp 330-344.

<http://www.springerlink.com/content/2aqafyat94h1lpx3/fulltext.pdf>

[6] **Goldreich, Oded (2001):** Foundations of Cryptography, Cambridge University Press, Cambridge 2001, ISBN 0-521-79172-3

[7] **Golic, Jovan Dj., and Menicocci, Renato (2001):** “A New Statistical Distinguisher for the Shrinking Generator”,
<http://citeseer.ist.psu.edu/560671.html>

[8] **Jantscher, Manfred (2006):** Use of Random and Varying Codes in Object Identification, Auto-ID Labs, School of Electrical and Electronics Engineering, University of Adelaide, Adelaide, Australia.

[9] **Kanso, Ali (2003):** “Clock-Controlled Shrinking Generator of Feedback Shift Registers”, in R.Safavi-Naini, and J. Seberry (eds), ACISP 2003, LNCS 2727, pp. 443-451, 2003. Springer-Verlag, Berlin, Heidelberg, 2003.

[10] **Mollin, Richard (2001),** Introduction to Cryptography, Chapman & Hall/CRC, Boca Raton, 2001, ISBN” 1-58488-127-5.

[11] **Ranasinghe, D., Engels, D. and Cole, P. (2005),** “Low-Cost RFID Systems: Confronting Security and Privacy”, White Paper Series; AutoID Labs; Edition 1; 2005.

[12] **[RSA01]** <http://www.rsasecurity.com/rsalabs/node.asp?id=2187> - P, NP

[13] **Schneier, Bruce (1994),** Applied Cryptography: Protocols, Algorithms and Source Code in C, John-Wiley and Sons, New York, 1994. ISBN: 0-471-5975602

[14] **Stinson, Douglas R. (1995)**, Cryptography: Theory and Practice, CRC Press, Boca Raton, Florida, 1995, ISBN: 0-8493-8521-0.

[15] **Welsh, Dominic (1990)**, Codes and Cryptography, Oxford University Press, Oxford. 1990, ISBN 0-19-853287-3.

[16] **Zenner et al (2001)**, “ Improved cryptoanalysis self shrinking generator” ,

<http://citeseer.ist.psu.edu/zenner01improved.html>

[17] **[Wik01] Wikipedia reference on Shrinking Generator (2007)**,
http://en.wikipedia.org/wiki/Shrinking_generator

[18] **[Wik02] Wikipedia reference on LFSR (2007)**,
http://en.wikipedia.org/wiki/Linear_feedback_shift_register

[19] **[Wik03] Wikipedia reference on GSM (2007)**,
<http://en.wikipedia.org/wiki/GSM>

[20] **[Wik04] Wikipedia reference on GSM’s encryption algorithms, A5 (2007)**,
<http://en.wikipedia.org/wiki/A5/1>

[21] **Ranasinghe D. (2007)**, “Lightweight Cryptography for Low Cost RFID”, in press, Springer-Verlag monograph, eds P H. Cole, D. Ranasinghe, 2007.

[22] **Menezes A., Van Oorschot, P., and Vanstone, S. (1997)** “Handbook of Cryptography”, CRC Press, Boca-Raton, 1997. (ISBN: 0-849308523-7)

[23] **Caballero-Gil, P., Fuster-Sabater, A. (2006)** “Using linear hybrid cellular automata to attack the shrinking generator”, IEICE Trans. Fundamentals, Vol. E90-A (5) May, 2006 pp. 1166-1172.

[24] **Seberry, Jennifer (2007)**, useful papers, tutorials, and links, off home page:
<http://www.uow.edu.au/~jennie>