Proceedings of the Fourth IASTED International Conference
Advances in Computer Science and Technology (ACST 2008)
April 2-4, 2008 Langkawi, Malaysia
ISBN CD: 978-0-88986-730-7

BIT-ORIENTED GENERATORS FOR WIRELESS SENSOR NETWORKS AND LOW COST RFID TRANSPONDERS

Raja Ghosal[1], Damith C. Ranasinghe[2] and Peter H. Cole[1]
[1]School of Electrical and electronic Engineering, The University of Adelaide
Adelaide, SA 5005, Australia
{rghosal, cole}@eleceng.adelaide.edu.au
[2]Department of Engineering, University of Cambridge
Cambridge, CB2 1RX, UK, rdcr2@cam.ac.uk
rdcr2@cam.ac.uk

ABSTRACT
Wireless Sensor Network (WSN) nodes and passive RFID transponders (tags) have severe constraints in computing power and hence offer particular challenges in the provision of e-Security. Passive RFID tags are at the lowest end of the spectrum in the family of devices with constrained computing power and memory. New approaches, differing from the traditional cryptosystems such as RSA or Diffie-Hellman, focusing on reducing complexity while being capable of providing an adequate level of security, are required for such devices. The use of one time codes is particularly appropriate as they guarantee perfect security and offer simple implementation. However, a significant hurdle to using one time codes is the size of the keys required for encrypting data. The task is further complicated by the need to have a purely randomly generated key. This paper explores lightweight security methods based on bit-oriented generators and one time codes, with the capability of meeting the constraints of resource limited devices and the demands for security services. The paper suggests as a lightweight security option, the unique capabilities and suitability of the shrinking generator for WSN nodes and low cost RFID tags.

KEY WORDS
Shrinking Generator, RFID, Wireless Sensor Networks, Security, Lightweight cryptography

1. Overview

Upcoming large-scale introduction of RFID in supply chains and growth of sensor integration has various potential security and privacy issues [1, 2]. Some key issues are related to the traceability of tags without knowledge of their owner, and also the ability to clone tags allowing persons to create tags for counterfeits.

It has been outlined in [3] that it is easier to trace with RFID than other technologies such as video, credit cards, mobile phones or Bluetooth devices. This is because of properties of tags such as: that they cannot be switched off, tags can answer without the agreement of their bearers, tags can be almost invisible, it is easy to analyse the logs of the readers, and it is easy to increase the communications range. The forecast future growth of RFID technology and its wide scale use can only lead to the emergence of new vulnerabilities [3]. There is thus the corresponding need for security. Considerable research is currently in progress throughout the world in various forms of cryptography [4, 5, 6, 7, 8] to address the challenges outlined above. The following sections will take a brief look at the areas of research.

1.1 Research

There is an ongoing body of researched devoted to the provision of security services for low cost RFID systems [3, 9, 10, 11]. The research has evolved in two different directions. One is focussed on using existing, what we define here as, *heavyweight cryptographic mechanisms*. Another is focussed on employing simple techniques and circuits to achieve security. The resulting research from the second approach is what we refer to here as *lightweight cryptography* [2].

1.2 Heavyweight Cryptographic Mechanisms

The mainstream cryptographic primitives (based on large keys, or working with large numbers as in typical modular exponentiation, in RSA, and discrete-logarithm methods), require high computational power, as well as high memory to achieve a level of security and efficiency required for modern security services. Elliptic Curve Cryptography (ECC) [12, 13, 14] has emerged to reduce the key sizes, in relation to RSA, to provide the same comparable security, For instance, an ECC system requires a 163 bit key for the same level of security as a 1024 bit key in the RSA system [4, 5, 11]. However, ECC places a greater demand on computational power. While integer factorization, as in RSA, and discrete-logarithm in the Diffie-Hellman and Elgamal systems can be broken in sub-exponential times, breaking is infeasible if the key size used is large enough. However, there are no known

sub-exponential time algorithms for ECC [1, 2]. So if an adequate key size is used, the cryptosystems mentioned above have no effective methods for defeating them in polynomial time. The fact that these cryptographic systems have withstood years of public scrutiny is generally taken as validation of the level of security they provide. All of the cryptographic tools mentioned provide a level of security called computational security (or practical security) [4], but are heavyweight cryptographic methods.

However for devices, such as RFID transponders and WSN nodes, it is not possible to use the large key sizes required to make known cryptanalytic attacks infeasible. Present cost and performance constraints prevent the tags from performing even basic symmetric key cryptographic operations [1, 2, 15].

It is anticipated that obstacles like very restricted operating power and the requirement to keep tags as cheap as possible, and therefore save silicon area, restrict the existing implementations of most of the known cryptographic primitives. Efforts towards application in simple devices, for instance an ECC engine on RFID tags [12,13] have been reported in literature, but these methods still place a large computational burden, increase costs and severely hinder performance while ignoring implications of sudden power loss to passively power tags.

Clearly there are challenges in providing security to low cost RFID tags [1, 2] whose prices are likely to drop to $0.05 per unit over next several years. RFID tags will become the feasible alternative to barcodes in supply chains. Currently low cost tags use very weak authentication mechanisms such as passwords and simple forward channel only encryptions based on an implied security of the backward channel [16].

1.3 Lightweight Cryptographic Mechanisms

An alternative approach is provided through the approach outlined previously as lightweight cryptography. Once such alternative is to consider the simplicity of one-time codes and the theoretical level of perfect security they offer. However, one-time pads require the generation of keys that are of the same size as the length of the plain text that needs to be encrypted. The problem is further aggravated by the requirement to have the key generated from a purely random source since that is the only way to guarantee the level of security termed perfect security.

Such one-time codes are often generated in complex apparatus from galactic or quasar noise, or some thermal or electronic noise. The alternative is to compromise on the level of security by using a pseudorandom number generator whose output for all practical purposes is random to an observer armed with a set of statistical tests at his or her disposal.

Once a reasonably secure key z, is generated, the plaintext (message, m) can be encrypted by bit XORing with the keystream to produce the encrypted message y. The receiver then decrypts y using another XOR operation of the same keystream z, to obtain the plaintext (message, m). The process is illustrated in Figure 1 below.

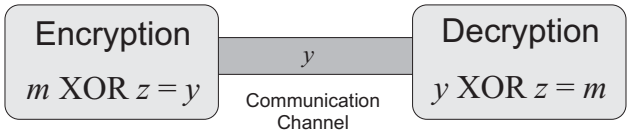

Figure 1. The encryption and the decryption process using a one-time pad.

Clearly this thinking leads us to Bit Oriented Generators (BOGs) based on GF(2) algebra, and simple operators such as XOR operators that are simple to implement on hardware [4, 5].

There has been much research into BOGs to achieve pseudorandom number generators (PRNG) to generate near random numbers for one-time codes [15]. A comprehensive survey and analysis of BOGs has been performed in [4]. The simplest of these PRNGs are implemented using Linear Feed Back Shift Registers (LFSRs). The linearity in the output of an LFSR is a weakness that can be exploited to know the internal connections, and hence predict the output of a system based on LFSRs. Solution of a set of linear equations from the observed bit stream output is a well known method of discovering the internal structure [6, 10].

Suggestions to enhance the security of such systems, desired properties of PRNGs, and use of non-linear functions have been well elucidated in [6]. A number of methods used for improving unpredictability and randomness of generators have been collected and expressed in [21]. These methods are described as:

- decimation (where every nth bit is selected from the output bit stream, while others are rejected);
- combining generators by addition of bit streams;
- combining bit streams by shuffling;
- and combining by shrinking bit streams.

The shrinking operation is less amenable to analysis based on linear algebra or generating functions, and hence, is preferable in applications where the sequence needs to be unpredictable, as per Brent [21].

In this paper we provide an overview of BOGs. Then we discuss some properties of the shrinking generator proposed in [18] which uses the technique of shrinking to achieve non-linearity, and its role as a lightweight cryptosystem for resource constrained devices such as low cost RFID tags and WSN nodes.

2. Types of Bit-Oriented Generators

There are many bit oriented stream generators [4, 6, 9, 15] 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 RSA generator; (e) the Alternating Step Generator; (f) the Shrinking Generator; and (g) the Self-Shrinking 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 a modular arithmetic. The Blum-Blum-Shub generator is the most used in practice. It is based on the hardness of quadratic residues and integer factorization over modular field arithmetic [9].

The Geffe Generator [4, 9] is defined by three maximum-length LFSRs. The lengths of the LFSRs *L1*, *L2*, *L3*, are pair wise relatively prime. This gives a high period as well as linear complexity [4, 6]. The combining function is given by (1) below [4].

$$f(x_1, x_2, x_3) = x_1 x_2 \ XOR \ x_2 x_3 \ XOR \ x_3 \qquad (1)$$

The output of the Geffe generator is balanced [4]. The number of 0's is roughly the same as the number of 1's. However the Geffe generator is prone to correlation attacks. This is because the probability the output keystream, *z*, follows either LFSR1 or LFSR3 is ¾, or, $P(z(t) = x_1(t) = P(z(t) = x_3(t)) = 0.75$.

The Geffe generator is a simple logic based generator, which is a combination of three LSFRs, connected by way of AND and XOR gates. However, this logic structure makes it prone to correlation attacks.

The RSA generator uses the underlying RSA arithmetic, which is based on the hardness of integer factorization over modular fields [5, 7, 9]. The Discrete Logarithm Generator uses the underlying discrete logarithm arithmetic over modular fields [9]. Both of these generators while cryptographically secure have the same implementation complexities discussed in Section 1.

There are several clock controlled generators which introduce non-linearity in the combination function. The motivation of the clock-controlled generator is to introduce non-linearity into the LFSR-based keystream generators [4, 17]. This is achieved by having the output of one LFSR control the clocking or stepping of the second LFSR. The second LFSR is clocked in an irregular manner. This helps in ensuring attacks based on the regular motion of LFSRs can be prevented. The Alternating Step Generator (ASG) and the Shrinking Generator (SG) are two such examples of clock-controlled generator designs [4].

The Alternating Step Generator (ASG) [4] uses a single clock and 3 LFSRs, *R1*, *R2*, *R3* connected by way of AND and NOT gates. The LFSRs are chosen to be maximum-length LFSRs. The lengths of these LFSRs, L_1, L_2, L_3 are pairwise relatively prime [4], that is,

- g.c.d.(L_1, L_2) =1,
- g.c.d. (L_2, L_3) = 1 and
- g.c.d. (L_1, L_3) = 1.

For the best security of the ASG, the lengths should be about the same. If *L1*, *L2*, *L3* are approximately *k* bits, the best known attacks on the ASG is a divide-and-conquer attack on the control register *R1*, which takes approximately 2^k steps. In [4] it is indicated that if $k = 128$ bits, the generator is secure against all previously known attacks. The Shrinking Generator (SG) [4, 18], appears to provide the simplest structure and the most secure output. In both these 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 a 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 [4]. 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. Nevertheless some cryptanalysis of the self-shrinking generator also consider the shrinking generator and hence provide valuable inputs for study of the Shrinking Generator [20]. The following sections will consider the shrinking generator as a BOG in more detail.

3. The Shrinking Generator

There are many candidate bit generators suitable for minimalist hardware. Of these the Shrinking Generator, a very simple system based on two LFSRs, has some excellent properties for use as a lightweight secure key stream source for resource starved devices. A literature review reveals that thus far the generator has withstood practical attacks if the tap polynomials and the initial seeds are kept secret and chosen with care [2]. Thus a scheme based on a shrinking generator can be thought of as a private key primitive where the private key forms the tag polynomials and the initialisation seed of the generator. The shrinking generator design first appeared in [18]. An illustration of this generator is shown in Figure 2. The generator is based on two LFSRs; the

selector *S*, and the data *A*, 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 and there is no output to the keystream [4, 9, 18]. 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 other schemes based on LFSRs.

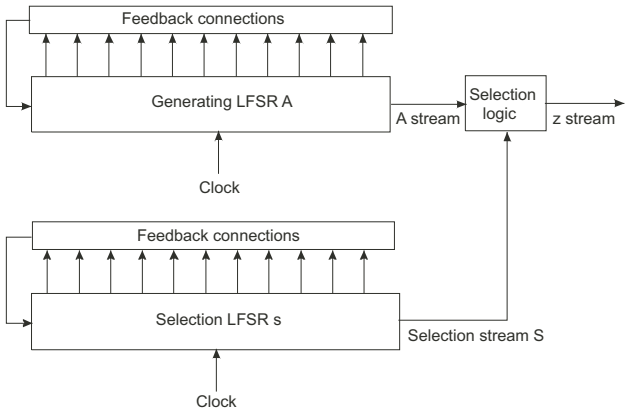

Figure 2. The Shrinking generator.

The current shortcoming is the gaps in the output keystream, *z*, when the output of the *S* register is 0. This problem and possible solutions are discussed in Section 6.

The Shrinking Generator has good random properties and important properties not present in LFSR sequences, such as exponential linear complexity. The standard tests for bit streams, such as randomness, and predictability as given by Yao's next bit test, are passed by the Shrinking Generator [4, 18]. In [4, 18] it is shown that the period of the keystream, the *z*-sequence, is exponential in both the lengths, *A* and *S*, of the *A* and the *S* registers. They show that the generator scheme passes the minimal tests for randomness, as indicated above.

If both LFSRs, *S* and *A* use primitive polynomials, then the periods of the LFSRs are [1, 23]: $L_S = 2^S - 1$, and $L_A = 2^A - 1$, where *S* denotes the size, in bits, of the LFSR *S*, and *A* the size of the LFSR *A*.

A Shrinking Generator, as outlined in Figure 2, will have the following useful properties [4, 18].

- If gcd (*S* , *A*) = 1, then the output sequence, z, has a period given by $(2^A - 1).\ 2^{S-1}$.
- The linear complexity L(*z*), of the output sequence *z*, satisfies: $A.2^{S-2} < L(z) \leq A.2^{S-1}$ and increases the complexity of attacks based on observing the generator outputs to given plaintext.

4. Security of the Shrinking Generator

While the implementation of the SG requires minimal hardware (shift registers and simple Boolean logic gates) and has the capacity to meet high throughput rates, ensuring the security of the keystream generated requires that the conditions outlined below are satisfied [4, 18]:

- The length of the two LFSRs must be co-prime, or gcd *(S, A)* = 1.
- Use of secret connection polynomials that are not sparse.
- Use of maximum length LFSRs for the LFSR *A* and *S* in the Shrinking Generator. (Using primitive polynomials for *A* and *S* ensures maximum length) [8, 23].

The security of the shrinking generator by way of an example, as illustrated in [4], is as follows. If secret connections are used, and $A = k$ bits, and $S = k$ bits, then the shrinking generator has a security level (the number of steps required to break the system) 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 provided the above condition are met.

4.1 Predictability

A novel approach to predict the output of a Shrinking Generator when the connections are known and the attacker is able to obtain a sufficient number of observations of the output sequence is presented in [22].

The approach presented in [22] used the imbalance of the observed output keystream bit sequence and the known connection polynomials to defeat the generator. Normally, the imbalance, in a bit stream, is defined as the difference between the number of 0's and the number of 1's in a period [23]. In the case of LFSRs based on primitive polynomials the imbalance is as close to zero as can be, consistent with the fact that the all zeros state of the register is excluded [23].

The approach used is similar to approaches used in [11] in attacking the algorithms used in GSM mobile communication networks. The method is not applicable to a shrinking generator with unknown feedback polynomials for the generating LFSRs.

4.2 Attacks

The general approaches to attacks on BOGs are based on several well articulated methods. Most of these will be applicable to several different types of generators, including the Shrinking Generator. They are listed below:

- Basic Divide and Conquer method which requires an exhaustive search through possible initial states and feedback polynomials of the selector LFSR as presented in [4, 7, 17].

- Use of Cellular Automata to attack the SG [2, 24]. This approach could be generalized to other bit oriented pseudo-random number generators.

- Correlation Attacks targeting the output of LFSRs [25, 26].
- A newer type of attack on the Shrinking Generator, known as the Fault Jumping Attack, seeking to break into the Selector Register (A) is reported in [27]. There is also the harder "dual" problem of breaking into the Data Register (A).

Since the inception of the Shrinking Generator there have been a number of reported attacks. However, the important point to note is the conditions under which these attacks have been reported. A summary of these attacks and the conditions under which such attacks are possible are outlined below [4, 18].

- If the connection polynomials (but not the seeds) for LFSRs, *S* and *A*, are known, then the best attack known for recovering the seeds and all future values of the key stream takes $O(2^S.A^3)$ steps. This attack is exponential in *S* and polynomial in *A*.
- If the connection polynomials and seeds are secret, the best attack known takes $O(2^{2S}.S.A)$ steps.

There is also an attack through the linear complexity of the shrinking generator which takes $O(2^S.A^2)$ steps when the seeds are secret irrespective of whether the connections are known or are secret. This attack however requires $(2^S.A)$ consecutive bits from the output sequence, *z*. This is infeasible for moderately large register bit sizes.

5. Application to RFID

It is thought that the shrinking generator can offer a practicable basis for securing low cost RFID systems in which the tags employ the shrinking generator as a source of apparently random numbers. The sequences generated by the tag can them be replicated by backend systems. Then, authorised access to a repository which securely indexes the tag's secrets (the connection polynomials and seeds) by its unique identifier can be used in providing security services such as mutual authentication of a tag and a reader, confidentiality of communication between a tag and reader or privacy services such as anonymity.

Such tags might enjoy the simplicity of not having to incorporate relatively silicon demanding circuits for generation of message authentication codes. Disguising a tag identity by XORing a static identity with a variable output of a shrinking generator becomes possible. While the hardware complexity and tag costs are limited to implementing shift registers to meet the lengths *S* and *A* of the shift registers and the Boolean logic.

6. Limitations of the Shrinking Generator and Future Work

The statistical properties, such as low correlations, distributions of 1's and 0's, and other metrics for measuring randomness depend on the choice of the size of the LFSRs and the initial seeds. There can be situations where the randomness is not close to near one-time codes, as desired for a pseudorandom bit generator. There is scope for research in identifying pattern cycles, of the *S* and *A* registers, in particular the properties of the *S* register.

In a LFSR based on a primitive polynomial, with the size of register *b*, the period is $2^b - 1$ [23]. If there is a stream of *t* ($1 \le t \le b$-2) consecutive 1's in a period there shall be *t* consecutive 0's in the period [23]. Thus, it is possible that 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 long periods of time. This is a severe limitation of the shrinking generator. The current methods to overcome this include buffering and intermittently running the two registers *S* and *A*, of the SG at twice the output data and keystream rate [2, 18] but this is provided at added costs to implementation.

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 analysis. This approach could be generalised to other generators such as the Geffe Generator, such that after *N* occurrences of output following a specific register, the sequence can be modified by adding an extra '0' or '1' .

In practical terms in the SG this function would be incorporated into the selection logic of Figure 2. The selection logic would also incorporate a function of buffering the output. Running the shrinking generator clock at an appropriate multiple of the rate at which key stream elements are needed will then ensure the buffer does not empty. The selection logic would also incorporate functions to ensure the buffer does not overflow, by stopping the bit generating clock when the buffer is full.

While the SG has a minimal silicon footprint there are issues of synchronisation of operations between the tags and the readers, recovery and security under a model of sudden power loss still need to be considered. Also, *A* and *S* may be generalised to any pair of pseudorandom sources [18]. It is open to analysis if any other sources can provide a secure source of pseudorandom bit sequences. Future work should consider addressing the above issues and exploring alternative pairs of pseudorandom sources for *A* and *S* registers.

7. Conclusion

Various aspects of Lightweight Cryptography have been presented. For resource limited devices that still require an adequate level of security, the focus should be on lightweight security mechanisms. We have illustrated

some avenues based on using BOGs to explore lightweight solutions.

A promising bit oriented generator, the Shrinking Generator, which provide the hardware simplicity, throughput and the level of security needed for low cost RFID tags and wireless sensor network nodes have been illustrated. We have quantified and outlined various forms of attacks on the Shrinking Generator. While the practical strength of the shrinking generator has become clear only after years of public scrutiny, there is scope for considerable research to more firmly establish the security of the Shrinking Generator approach, to remove the potential for the generator to have an unavailable output, and to apply it to other low computing power devices. There is also scope for further research into the theoretical basis of its security.

Acknowledgements

The authors are grateful to Mr. Manfred Jantscher and Mr Alfio Grasso for valuable discussions.

References

[1] A. Juels, RFID: Security and Privacy: A Research Survey, *Condensed version in IEEE Journal on Selected Areas in Communication, 2006. available from: http://www.rsasecurity.com/rsalabs/node.asp?id=2937 (6th Dec 2007)*

[2] D. C. Ranasinghe, Lightweight Cryptography for Low Cost RFID, in *Networked RFID systems and lightweight cryptography: raising barriers to product counterfeiting,* eds. P H. Cole & D. C. Ranasinghe (Berlin: Springer Verlag, 2008, ISBN: 978-3-540-71640-2).

[3] G. Avoine, Bibliography on Security and Privacy in RFID Systems, *MIT Cambridge, Massachusets, 2006 available from: http://lasecwww.epfl.ch/~gavoine/rfid/ (6th Dec 2006)*

[4] A. Menezes, P. Van Oorschot, & S. Vanstone, *Handbook of applied cryptography* (Boca-Raton, Florida: CRC Press, 1997, ISBN: 0-8493-8523-7)

[5] R. A. Mollin, *Introduction to cryptography* (Boca-Raton, Fl: Chapman & Hall/CRC, 2001, ISBN 1-58488-127-5)

[6] J. Seberry, Stream Clarke1, 2, *available from http://www.uow.edu.au/~jennie, (6th Dec 2007).*

[7] B. Schneier, *Applied cryptography: protocols, algorithms and source code in C* (New York: John-Wiley and Sons, 1994, ISBN: 0-471-5975602)

[8] D. R. Stinson, *Cryptography: theory and practice* (Boca-Raton, Florida: CRC Press, 1995, ISBN: 0-8493-8521-0)

[9] M. Aigner, Crypto Implementations for RFID Tags, Learning from the smart card industry. *PROACT, Crypto for RFID series*, Graz (Austria), 2006.

[10] S. Piramuthu, Protocols for RFID tag/reader Authentication, *Decision Support Systems, Elsevier, 43*(3), April 2007, 897-914.

[11] E. Barkan, E. Biham E., & N. Keller, Instant Ciphertext-Only Cryptoanalysis on GSM Encrypted Communications; *available from: http://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-get.cgi/2006/CS/CS-2006-07.pdf (6th Dec 2007).*

[12] R. Balasubramanian., Elliptic Curves and Cryptography, in *Elliptic curves, modular forms, and cryptography,* 325-345, eds. A. K. Bhandari, D. S. Nagraj, B. Ramakrishnan, & T. N. Venkataraman, (New Delhi: Hindustan Book Agency, 2003, ISBN 81-85931-42-9)

[13] J. Wolkerstorfer, J., Is Elliptic-Curve Cryptography Suitable to Secure RFID Tags? in *Handout of the Ecrypt Workshop on RFID and Lightweight Crypt*, Graz, *2005*.

[14] L. Batina, J. Guajardo, T. Kerins, N. Mentens, P. Tuyls, & I. Verbauwhede, An Elliptic Curve Processor Suitable For RFID-Tags, *Cryptology ePrint Archive: Report 2006/227.*

[15] D. C. Ranasinghe, D. Engels, & P. H. Cole, Low-Cost RFID Systems: Confronting Security and Privacy, *White Paper Series*, (Zurich: AutoID Lab, Edition 1; 2005).

[16] D. C. Ranasinghe & P. H. Cole, Addressing Insecurities and Violations of Privacy, in *Networked RFID systems and lightweight cryptography: raising barriers to product counterfeiting*, eds. P H. Cole & D. C. Ranasinghe, (Berlin: Springer Verlag, 2008, ISBN:978-3-540-71640-2)

[17] M. Jantscher, *Use of Random and Varying Codes in Object Identification*, Auto-ID Labs, School of Electrical and Electronics Engineering, University of Adelaide, Adelaide, Australia, 2006.

[18] D. Coppersmith, H. Krawczyk, & Y. Mansour, The Shrinking Generator, *Proc. Crypto-93*, LNCS Vol 773, 22-39, (Berlin: Springer-Verlag, 1994).

[19] Kanso Ali, Clock-Controlled Shrinking Generator of Feedback Shift Registers, *Proc. ACISP 2003, LNCS 2727,* eds. R.Safavi-Naini, and J. Seberrry, 443-451, (Berlin: Springer-Verlag, 2003).

[20] E. Zenner, M. Krause, & S. Lucks, Improved cryptoanalysis self shrinking generator, 2001. *available from: http://citeseer.ist.pst.edu/senner01improved.html (6th Dec 2007).*

[21] R. P. Brent., Fast and Reliable Random Number Generators for Scientific Computing, in *Applied Parallel Computing, PARA2004,* eds. J. Doagarra, K. Madsen, and J Wasniewski, LNCS 3732, 1-10, (Berlin: Springer Verlag, 2006, ISBN:978-3-540-29067-4)

[22] P. Ekdahl, W. Meier, & T. Johansson, Predicting the Shrinking Generator with Fixed Connections, *Proc. EuroCrypt 2003*, ed. E. Biham, LNCS 2656, 330-344 (Berlin: Springer-Verlag, 2003).

[23] D. Welsh, Dominic, *Codes and cryptography* (Oxford: Oxford University Press, 1990, ISBN 0-19-853287-3).

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

[25] B. Schneier, Dutch passport break in, RFID ISO 14443 protocol Basic Access Protocol, 2006, *available from: http://www.schneier.com/blog/archives/2005/08/rfid_passport_s_1.html (6th Dec 2007).*

[26] D. J. Golic, & R. Menicocci, A New Statistical Distinguisher for the Shrinking Generator, *available from: http://citeseer.ist.psu.edu/560671.html (6th Dec 2007).*

[27] M. Gomulkeiwicz, M. Kutylowski M, & P. Wlaz, Fault Jumping Attacks Against the Shrinking Generator, 2006, *available from: http://drops.dagstuhl.de/opus/volltexte/2006/611/pdf/06111.KutylowskiMiroslaw.Paper.611.pdf (6th Dec 2007).*