Available online at www.sciencedirect.com
ScienceDirect
Computer Standards & Interfaces 31 (2009) 88–97
COMPUTER STANDARDS & INTERFACES
www.elsevier.com/locate/csi

LAMED — A PRNG for EPC Class-1 Generation-2 RFID specification

Pedro Peris-Lopez *, Julio Cesar Hernandez-Castro, Juan M. Estevez-Tapiador, Arturo Ribagorda

Computer Science Department, Carlos III University of Madrid, Avda. de la Universidad, 28911, Leganés (Madrid), Spain

Received 7 February 2007; accepted 18 November 2007
Available online 5 December 2007

Abstract

RFID is a relatively heterogenous radio technology, where it is necessary to put an extra effort on security and privacy-related issues. As early as 2004, some authors suggested the use of a PRNG for increasing security. This was later questioned because many thought a PRNG implementation may go well beyond the very limited computational capabilities of low-cost RFID tags. However, its use has been ratified by EPCGlobal (EPC Class-1 Generation-2) and ISO (ISO/IEC 18000-6C). This motivates our proposal of a new PRNG, named LAMED, which is compliant with the standards and successfully passes several batteries of very demanding randomness tests (ENT, DIEHARD, NIST, and SEXTON). A study of its hardware complexity shows that LAMED can be implemented with slightly less than 1.6 K gates, and that pseudo-random numbers can be generated each 1.8 ms. So we can affirm this is a realist proposal both conforming with the EPC-G1C2 standard, and suitable for low-cost RFID tags.

Keywords: RFID; EPC; Security; PRNG; Lightweight cryptography

1. Introduction

Which are the RFID standards? If you had to answer this question, you would need a vast period of time to analyze the great tangle of associated standards. This is due to the heterogeneousness of the technology, as well as its nearly sixty years of existence (one of the earliest papers exploring RFID is a landmark paper by Harry Stockman "Communications by means of Reflected Power").[1] Nowadays, EPCglobal [1] (a joint venture between EAN International and Uniform Code Council) and ISO [2] join forces to publicize and harmonize the use of RFID technology.

One of the most important standards proposed by EPCglobal is the EPCglobal Class-1 Gen-2 RFID specification (EPC-C1G2) [3]. This standard was adopted in 2004, and eighteen months later (March–April 2006) ratified by ISO and published as an amendment to its 18000-6 standard. In the following, we briefly summarize the most important properties of EPC-C1G2 specification:

- Tags are passive, so they receive all their operating energy from the reader's RF waveform.
- Tags operate on the UHF band (860–960 MHz). Generally, their effectiveness will be poor around metals and water. Their read range is up to 9 m.
- The very constrained resources and storage capabilities dictate that EPC-C1G2 tags cannot afford traditional cryptographic primitives.
- Tags include on chip a 16-bit Pseudo-Random Number Generator and a 16-bit Cyclic Redundancy Code (CRC) checksum.
- Tags have two 32-bit PINs:
  - Kill PIN: The kill password is a 32-bit value stored in reserved memory (00h to 1Fh). A reader shall use a tag's kill password once, to kill the tag and render it silent there after.
  - Access PIN: The access password is a 32-bit value stored in reserved memory (20h to 3Fh). Tags with a nonzero access password shall require a reader to issue this password before transitioning to the secured state, which will allow it to read or write in the password fields.

* Corresponding author.
E-mail addresses: pperis@inf.uc3m.es (P. Peris-Lopez), jcesar@inf.uc3m.es (J.C. Hernandez-Castro), jestevez@inf.uc3m.es (J.M. Estevez-Tapiador), arturo@inf.uc3m.es (A. Ribagorda).

[1] Proceedings of the Institute of Radio Engineers, pp. 1196 - 1204, October 1948.

0920-5489/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.csi.2007.11.013

The two main operations for managing tag populations (inventory and access) are shown in Fig. 1. A detailed description of the inventory operation is as follows:

(1) A reader sends a request message (query) to a tag. The query initiates an inventory round and decides which tags participate in the round.
(2) Each tag which receives the query picks a 16-bit random value using the PRNG, and shall load this value into a slot counter. When the slot counter becomes zero, the tag backscatters the random value (RN16) to the reader.
(3) The reader acknowledges the tag with an ACK containing this same RN16.
(4) The tag compares the random number in the ACK with the RN16 it sends. If it is right, the tag backscatters its PC (Protocol-Control), EPC (Electronic Product Code), and CRC-16.

After acknowledging a tag, a reader may choose to access it. The access command set comprises ReqRN, Read, Write, Kill, Lock, Access, BlockWrite, and BlockErase. A reader will access a tag as follows:

(1) The reader issues a ReqRN, containing the previous RN16, to the acknowledged tag.
(2) The tag compares the random number in the ReqRN with the RN16 in the tag. If these values coincide, the tag generates and stores a new RN16 (denoted handle), and backscatters the handle.
(3) The Write, Kill, and Access commands send 16-bit words (either data or half-passwords) from reader to tag. These commands use one-time-pad based link cover-coding to obscure the word being transmitted, as follows:
   (a) The reader issues a ReqRN, to which the tag responds by backscattering a new RN16.
   (b) The reader then generates a 16-bit ciphertext string comprising a bitwise xor of the 16-bit word to be transmitted with this new RN16.
   (c) The tag decrypts the received ciphertext string by performing a bit-wise xor of the received 16-bit ciphertext string with the original RN16.

Even though it is out of the scope of this document, we would like to mention some important security faults in the EPC-G1C2 specification. In the inventory command, the EPC is transmitted as plain text, something that generates privacy and spoofing problems. Tracking could be done in a very straightforward way, since the EPC is fixed. Moreover, the security of the access command is weak, so performing a passive attack is very simple. An attacker listening to the backward and forward channel can pick up the random number sent by the tag. Next, the attacker can decrypt the ciphertext sent by the reader by performing an xor with the previously collected random number. So the plaintext or PIN can be disclosed by this quite simple mechanism, which constitutes an important security pitfall. The following papers are recommended to readers interested in this topic [4–7].

The remainder of the paper is organized as follows. Section 2 motivates the need of pseudo-random number generators for

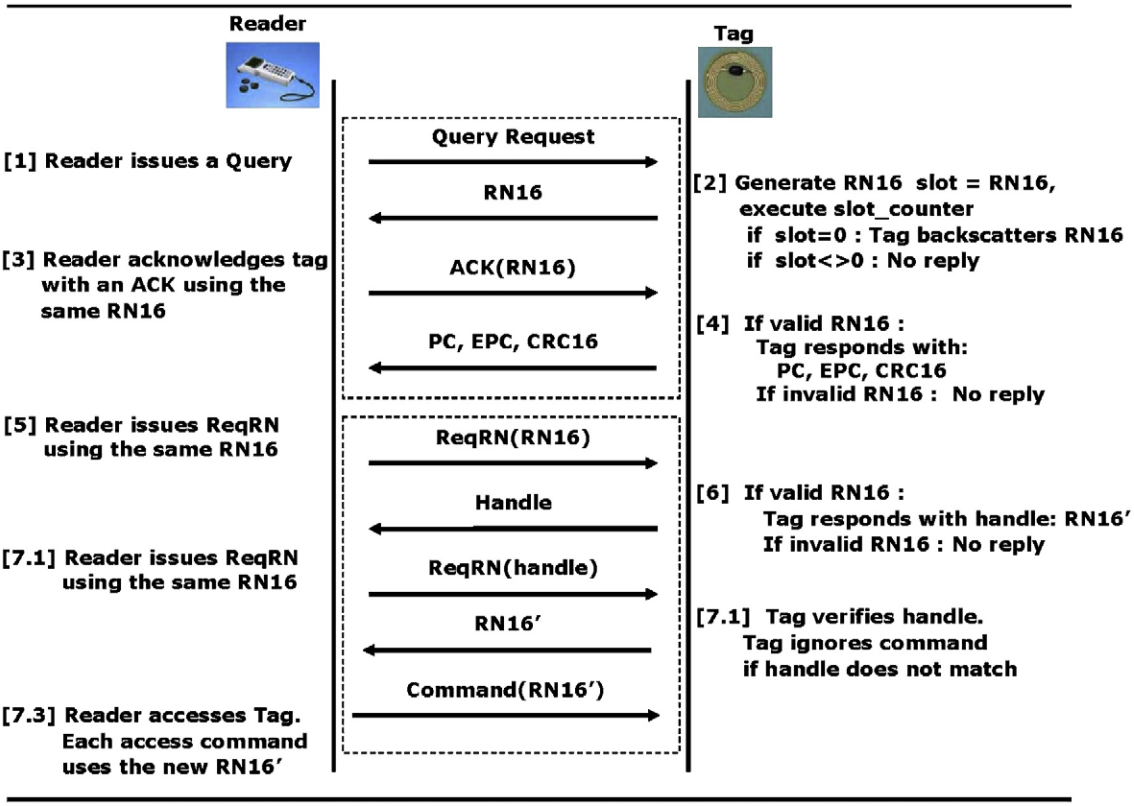

Fig. 1.

low-cost RFID tags. In Sections 3 and 4, a new PRNG conforming with the EPC-C1G2 is proposed. A security evaluation is presented in Section 5. In Section 6, the hardware complexity of our proposal is evaluated. Finally, some concluding remarks are presented in Section 7.

2. Pseudo random number generation

The need for random and pseudo-random numbers arises in many cryptographic applications. In fact, the usage of PRNGs in RFID systems has been proposed nearly since its beginning. In 2003, Weis et al. proposed the randomized hash-locking scheme, based on a hash function and a random number generator in order to prevent tracking, but limiting its applicability only to small tag populations [8]. Molnar et al. proposed a simple protocol for enhancing passwords in RFID tags [9]. There are others papers where the use of a PRNG has been proposed [10–14]. Nowadays, the used of a PRNG has been ratified by EPCGlobal (EPC-C1G2) and ISO (ISO/IEC 18000-6C). A generator conforming with these specifications [3,15], should meet the following randomness criteria:

- Probability of a single RN16: The probability that any RN16 drawn from the RNG has value RN16=j for any j, shall be bounded by:

$$\frac{0.8}{2^{16}} < P(\mathrm{RN16} = j) < \frac{1.25}{2^{16}} \tag{1}$$

- Probability of simultaneously identical sequences: For a tag population of up to 10,000 tags, the probability that any of two or more tags simultaneously generate the same sequence of RN16s shall be less than 0.1%, regardless of when the tags are energized.
- Probability of predicting an RN16: An RN16 drawn from a tag's RNG 10 ms after the end of Tr, shall not be predictable with a probability greater than 0.025% if the outcomes of prior draws from RNG, performed under identical conditions, are known.

Furthermore, when designing a PRNG conforming to the EPC-G1C2, we should take into account the severe hardware limitations of these systems. Low-cost RFID tags have roughly 5 K–10 K gates, and within this gate counting only from 400 to 4 K gates can be devoted to security-related tasks [16]. Additionally, we have important temporal requirements, which demand that a given number of tags should be read in a given amount of time. Compared to previous class-1 EPC tags (150 tags/sec), generation-2 readers should be able to read 450 tags/sec [3,17]. This puts a severe limitation on the maximum number of cycles (1/f) a tag can spend to generate a random number.

3. Experimentation issues

The methodology to obtain the core of our PRNG is based in the use of Genetic Programming (GP). GP is a stochastic population-based search method devised in 1992 by John R. Koza [18]. This technique evolves computer programs instead of just particular solutions to a specific problem as in GA. As PRNGs are designed and implemented as computers programs, the use of GP in the problem is justified.

We have used the lil-gp library [19] for our experimentation. Next, we briefly describe how its parameters have been adjusted to our particular problem.

- *Function Set*. These functions are the building blocks of the individual we will obtain. We decided to include only very efficient operations easy to implement in hardware: vrotd (one-bit right rotation), xor (addition mod2), and (bitwise and), or (bitwise or), and not (bitwise not). The sum (sum mod 2^{32}) operator is also necessary, in order to avoid linearity. We did not include multiplication mod 2^{32} because the multiplication of two 32-bit values could be a very costly operator [20].
- *Terminal Set.* The terminals will be represented by two 32-bit unsigned integers (a_0, a_1). We also included Ephemeral Random Constants (ERCs), which are constant values (in our problem, 32-bit random values) that GP uses to try to generate better individuals.
- *Fitness Function*. We use the Avalanche Effect to evaluate the nonlinearity of our generator. In fact, an even more demanding property will be used: the Strict Avalanche Criterion [21], which can be mathematically described by:

$$\forall x, y | H(x,y) = 1, \quad H(F(x), F(y)) \approx B\left(n, \frac{1}{2}\right) \tag{2}$$

To measure the proximity of the distribution of the computed Hamming distances to the sought theoretical binomial $B(n, 1/2)$, a χ^2 goodness-of-fit test statistic is employed. Concretely, the proposal fitness function is the following:

$$\mathrm{Fitness} = 10^6/\chi^2 \tag{3}$$

It was necessary to amplify the fitness function (multiplying by 10^6) because the initial values of the χ^2 statistic were extremely high, making the fitness negligible at the beginning of the evolution process.

In more detail, the fitness of each individual is calculated as follows: we use the Mersenne Twister generator [22] to randomly generate the pair (a_0, a_1). The output O_0 for this input is stored. Then, we randomly flip one single bit of this two 32-bit input and we obtain a new output O_1. Now, we store the Hamming distance between those two output values $H(O_0, O_1)$. This process is repeated a number of times ($2^{11}=2048$ was experimentally proved to be enough) and each time a Hamming chi-square statistic is obtained.

- *Tree Size Limitations*. The depth and/or the number of nodes of the individuals should be limited. We tried both limiting the depth and not limiting the number of nodes, and vice versa. The best results were consistently obtained by using the latter option. We allowed the PRNG to use up to 65 nodes for trying to ensure a high degree of Avalanche Effect and robustness without exceeding the processing and temporal requirements of a low-cost RFID tag.

When the parameters were adjusted, we ran 20 experiments with different seeds for generating the initial population (seed $_i$ = $(\pi * 100{,}000)^i$ (mod 1,000,000)), with a population size of 500 individuals, a crossover probability of 0.8, a reproduction probability of 0.2, and an ending condition of reaching 2000 generations. These parameters were experimentally found to be adequate for our purposes. The best individual found following the approach described above has an Avalanche Effect of 15.9707 (16.00 being the optimal value) and presents a χ^2 goodness-of-fit test statistic of 5.1175 for a χ^2 probability distribution with 32 degrees of freedom implying that, with probability 0.9999999853, the computed Hamming distances come from a Binomial distribution $B(32,1/2)$.

```
=== BEST-OF-RUN ===                    TREE:
          generation: 1172             (xor (sum a1 a0) (vrotd (vrotd (sum (xor a0 a1)
               nodes: 65               (vrotd (xor a1 (vrotd (vrotd (vrotd (vrotd (sum
               depth: 39               (sum a0 a1)) (vrotd (vrotd (sum (vrotd (vrotd
                hits: 1597070          (vrotd (xor (sum a1 a0) (vrotd (vrotd (sum (sum
TOP INDIVIDUAL: -- #1 --                a0 a1)) (vrotd (vrotd (sum (vrotd (vrotd (vrotd
                hits: 1597070          (vrotd (xor (sum a1 a0) (vrotd (vrotd (vrotd
           raw fitness: 195409.7232    (sum (xor a0 a1) (vrotd (vrotd (vrotd (vrotd
  standardized fitness: 195409.7232    (vrotd (sum a1 a0))))))))))))))) a1))))))))))
      adjusted fitness: 195409.7232     a1))))))))))))))))
```

4. Design specification

The seed of the PRNG will consist of an initialization vector (iv) and a key (s). The iv may be public, but it is very important that it is never reused together with the same key. It can also be kept secret, effectively extending the keylength up to 64-bits, depending on the security needs of the specific application. The key is a secret only known by an authorized reader and the tag. Usually, the secret (s) will be set at time of manufacture, and will be stored in the associate row of the back-end database. In Eqs. (4) and (5) we show the proposed update function for the internal state of LAMED.

$$a_0^{n+1} = \begin{cases} a_1^n + iv \text{ si } n \text{ is odd} \\ a_1^n \oplus iv \text{ si } n \text{ is even} \end{cases} \quad (4)$$

$$a_1^{n+1} = \begin{cases} 0\left(a_0^n, a_1^n\right) \oplus s \text{ si } n \text{ is odd} \\ 0\left(a_0^n, a_1^n\right) + s \text{ si } n \text{ is even} \end{cases} \quad (5)$$

The output length is 32 bits. As the specification EPC-C1G2 proposes the use of a 16-bit PRNG, we have designed a 16-bit version of our PRNG, named LAMED-EPC, with an additional xor operation before its final output. The 32-bit output is divided in two halves, $MSB_{31:16}$ and $LSB_{15:0}$. These two halves will then be xored in order to obtain a 16-bit output with higher entropy. In this way, our proposal is EPC-C1G2 compliant and has the additional advantage that a 32-bit PRNG is also supported, which could be relevant for certain applications and also increases its flexibility and, probably, its longevity, as mentioned in [7,5]. Furthermore, the access and kill PIN are 32-bit values. The use of 32-bit random numbers would avoid the complex multi-step procedure for using the access and kill command proposed in the standard. It is important to recall, however, that the security margin of a protocol using a 16-bit PRNG is usually bounded by $\frac{1}{2^{16}}$. Moreover, a generic time-memory-data trade-off attack costs $O\left(2^{\frac{n}{2}}\right)$, see [23], where n is the number of inner state variables in the PRNG. In LAMED-EPC, with a public iv, which is the weakest security configuration possible, the total of state variables is 32. Thus, the expected complexity of a time-memory is lower limited by $O(2^{16})$.

5. Security analysis

In this section we start by verifying the randomness properties of both the output of LAMED and LAMED-EPC. We also perform a more exhaustive analysis of LAMED-EPC, proving its conformity with the EPC-C1G2 specification.

5.1. Standard security analysis

We have performed an extensive security analysis of LAMED,[2] consisting of examining the statistical properties of the output over a random initialization of the initialization vector (iv) and the key (s) obtained from http://randomnumber.org. Unfortunately, it is not possible to prove randomness, because there is no efficient deterministic definition of this rather abstract concept. Instead, scientists usually limit themselves to using batteries of randomness tests to verify that the output of a given function 'seems' random, meaning the used tests cannot distinguish it from a truly (theoretical) random variable. In 2001, the National Institute of Standards and Technology (NIST) proposed a comprehensive suite of randomness tests suitable for the evaluation of PRNGs used in cryptographic applications [24]. Additionally, there is another very stringent set of randomness tests called Diehard, developed by Marsaglia [25,26]. We have also used a battery of tests named ENT [27], and a very recent set of randomness tests proposed by Sexton [28]. However, none of these test suites ensure, when successfully passed, that a given generator is useful for all kind of applications. On the other hand, systematically passing the NIST and Diehard batteries provides evidence in favour of a good degree of output randomness.

Two files of 300 MB and 4 GB have been generated with LAMED and LAMED-EPC, the latter only being used in Sexton's battery as it needs a huge amount of data to run. Results obtained with ENT, Diehard and David Sexton's battery are presented in Tables 1, 2, and 3, respectively. When several p-values were produced in the same test, we summarized them by a Kolmogorov–Smirnov p-value (marked with *), that should be greater than 0.05 to be considered successful. LAMED also passed the very demanding – because it is oriented to cryptographic applications – NIST statistical battery. We have computed 100 p-values for every test in the statistical suite; the proportion of successful ones is presented in Table 4. If this proportion is lower than 0.96 it is considered that the whole test failed. From these results, we can conclude that LAMED's output successfully passed all the randomness tests.

5.2. Compliance to EPC-C1G2 security requirements

In this section we will study the compliance of LAMED-EPC with EPC-C1G2. This study will be started analyzing the probability of a single 16-bit random number. The standard asks

[2] The whole report is available in http://163.117.149.208/lamed.

Table 1
Results obtained with ENT

Test	LAMED	LAMED-EPC
Entropy	7.999999 bits/byte	7.999999 bits/byte
Compression rate	0%	0%
χ^2 Statistic	256.90 (50%)	246.61 (50%)
Arithmetic mean	127.5024	127.4980
Monte Carlo π estimation	3.141474228 (0.00%)	3.141796646 (0.01%)
Serial correlation coefficient	−0.000023	0.000015

that "the probability that any RN16 drawn from the RNG has value RN16=j for any j, shall be bounded by $0.8/2^{16} < P(\mathrm{RN16}=j) < 1.25/2^{16}$". In order to verify this property, five files of 2^{30} bytes have been generated. These files have been obtained with different secret keys and initialization vectors, in every case taken from http://random.org/. From this analysis, we can conclude that the probability of any 16-bit random number drawn from LAMED-EPC is, in fact, bounded by:

$$\frac{0.96}{2^{16}} < P_{\mathrm{LAMED-EPC}}(\mathrm{RN16}=j) < \frac{1.05}{2^{16}} \quad (6)$$

Another interesting property asked for in the EPC-C1G2 standard is about the probability of predicting a random number. The specification in this context determines that "a RN16 could not be predicted with probability greater than 0.025% if the outcomes of prior draws from RNG, performed under identical conditions, are known". In order to check if this property holds, some tests have been completed:

- *Serial Correlation Test*: This quantity measures the extent to which each n-bit output depends upon the previous n-bit output. For random sequences, this value (which can be positive or negative) should be very close to zero. As an example, a non-random n-bit stream such as a counter will yield a serial correlation coefficient of about 0.5. Five files of 2^{25} bytes haven been generated (with different secret keys and initialization vectors). From each file, the serial correlation (bit, byte, 16-bit) has been obtained.Table 5, shows the obtained results.
- *Bit-Byte Prediction Test from David Sexton's battery*: Many algorithms are used to predict the value of each bit (respectively, byte) of the sequence from the beginning of the sequence to the end. In a random sequence the probability of success of any such algorithm should be 1/2 (resp. 1/256). The number of successes is counted. A chi-squared statistic with 1 degree of freedom is computed. The following tests have been carried out:
  - *Bit Prediction A Test*: the numbers of zeros and ones in all the previous bits are counted. If the ones outnumber the zeros, a zero is predicted; if the zeros outnumber the ones, a one is predicted. Otherwise the prediction is the same as for the previous bit.

Table 2
Results obtained with the Diehard suite

	LAMED	LAMED-EPC
Test	p-value	p-value
Birthday spacings	0.261	0.192
GCD and Gorilla	0.778*	0.608*
Overlapping Permutations	0.311*	0.564*
Ranks of 31×31 and 32×32 Matrices	0.699*	0.587*
Ranks of 6×8 Matrices	0.521	0.947
Monkey tests on 20-bit words	0.312*	0.758*
Monkey Test OPSO	0.436*	0.751*
Monkey Test OQSO	0.742*	0.835*
Monkey Test DNA	0.688*	0.231*
Count the 1's in a stream of bytes	0.664	0.789
Count the 1's in specific bytes	0.586*	0.680*
Parking lot test	0.433	0.117
Minimum distance test	0.411	0.03
Random spheres test	0.788	0.50
The squeeze test	0.841	0.449
Overlapping sums test	0.173	0.003
Runs up and down test	0.191	0.859
The craps test	0.443*	0.539*
Overall KS p-value	0.778	0.792

Table 3
Results obtained with David Sexton's battery

	LAMED	LAMED-EPC
Test	p-value	p-value
Bit runs test	0.925*	0.726*
Frequency test	0.016*	0.962*
Bit test	0.375*	0.748*
Sum test	0.841*	0.18*
Matrix test	0.432*	0.857*
Prediction test	0.119*	0.529*
And test	0.778*	0.856
Up/down test	0.699*	0.355*
Rect. distance test	0.018	0.798
Collision test	0.577*	0.362*
Offset xor test	0.865*	0.723*
Mod test	0.230*	0.637*

Table 4
Results obtained with the NIST suite

	LAMED	LAMED-EPC
Test	Proportion	Proportion
Frequency	0.98	0.98
Block-frequency	0.98	1.00
Cumulative-sums	0.98, 0.98	0.98, 0.98
Runs	1.00	0.99
Longest-run	1.00	0.99
Rank	0.98	0.99
Fft	0.99	0.98
Overlapping-templates	0.98	1.00
Universal	0.96	0.98
Apen	0.99	1.00
Serial	0.97, 1.00	0.99, 0.97
Linear-complexity	0.99	0.98
Random-excursions	0.97, 0.98	0.97, 1.00
	1.00, 0.97	0.97, 0.96
	1.00, 1.00	1.0, 0.98
	0.97, 1.00	0.97, 0.98
Random-excursions-variant	1.00, 1.00, 1.00	1.00, 1.00, 1.00
	0.98, 1.00, 1.00	0.98, 0.98, 1.00
	1.00, 1.00, 1.00	1.00, 0.98, 0.98
	1.00, 1.00, 1.00	1.00, 1.00, 1.00
	1.00, 0.98, 0.97	1.00, 0.98, 0.98
	0.98, 1.00, 0.97	1.00, 1.00, 1.00

Table 5
Serial correlation test

	LAMED-EPC		
Experiment	Bit	Byte	16-bit
1—Experiment	−0.000002	0.000154	0.000065
2—Experiment	−0.000015	0.000028	0.000028
3—Experiment	−0.000013	−0.000008	−0.000157
4—Experiment	−0.000006	0.000079	0.000074
5—Experiment	−0.000053	0.000088	0.000047

- *Bit Prediction B Test*: the numbers of zeros and ones in the previous 9 bits are counted. If the ones outnumber the zeros, a zero is predicted; if the zeros outnumber the ones, a one is predicted.
- *Bit Prediction C Test*: the numbers of zeros and ones in the previous 17 bits are counted. If the ones outnumber the zeros, a zero is predicted; if the zeros outnumber the ones, a one is predicted.
- *Bit Prediction D Test*: the numbers of zeros and ones in the previous 33 bits are counted. If the ones outnumber the zeros, a zero is predicted; if the zeros outnumber the ones, a one is predicted.
- *Bit Prediction E Test*: the numbers of zeros and ones in the previous 65 bits are counted. If the ones outnumber the zeros, a zero is predicted; if the zeros outnumber the ones, a one is predicted.
- *Byte Prediction A Test*: the next byte is predicted to be equal to all the previous bytes bitwise XORed together. The first byte of the sequence is predicted to equal zero.
- *Byte Prediction B Test*: the next byte is predicted to be equal to the sum of all the previous bytes, modulo 256. The first byte of the sequence is predicted to equal zero.
- *Byte Prediction C Test*: the next byte value is predicted to be zero until the first zero is found. From that point on, the next byte value is predicted to be the byte value whose last appearance was furthest back in the sequence.
- *Byte Prediction D Test*: a given byte value is predicted to be followed by the same byte value it was followed by the last time it appeared in the sequence. A byte value that has not previously appeared in the sequence is predicted to be followed by the byte value of the first byte in the sequence. The first byte of the sequence is predicted to equal zero.
- *Byte Repetition Test*: This test is equivalent to a byte prediction test where each byte is predicted to be equal to its preceding byte. The first byte of the sequence is predicted to equal the last byte of the sequence.

  A file of 2^{32} bytes has been generated to check these prediction tests, results obtained are summarized in Table 6.

- Lineal Predictor: another interesting approach for finding a good predictor for a given function is to try to approximate it by a linear relation, similarly to what is done in linear cryptanalysis for block ciphers [29,30]. In order to obtain the linear bias of LAMED-EPC, the following experiment has been accomplished: two 16-bit masks (A,B) have been randomly picked, and two consecutive outputs have been generated (O_i,O_{i+1}). With these two masks and outputs, the equality $A*O_i=B*O_{i+1}$ is evaluated. The process is repeated 2^n times, counting the numbers of successes (m). The * symbolizes scalar product, with a mod 2 operation being carried out after addition. The bias is defined as:

$$\text{bias} = \frac{1}{2^{-\log_2\left(\left|\frac{m}{2^n}-\frac{1}{2}\right|\right)}} \tag{7}$$

  We have randomly tested many pairs of different masks, A and B. For each pair, 2^{25} 16-bit outputs have been generated, and for consecutive outputs the previous expression ($A*O_i=B*O_i$) was evaluated. From the above results, we can gather that the bias of LAMED-EPC is limited by:

$$\text{BIAS}_{\text{LAMED-EPC}} < \frac{1}{2^{11.77}} \tag{8}$$

  which implies that the security margin, given by the number of observations needed for predicting the next output value with a good accuracy is around $(2^{11.77})^2$, see [29,30], which is well over the 2^{-16} limit that any protocol using this PRNG will have.
- *Differential Analysis*: A differential analysis is a form of attack in which the differences between consecutive values are used to attempt to gain additional knowledge about the system. Two different analysis have been proposed, where 0_i refers to the i^{th} output provided by LAMED-EPC:
  - The simpler and generally most useful is the XOR analysis where $0_i \oplus 0_{i+1}$ is studied.
  - Another standard analysis is that of the difference $(0_i - 0_{i+1})$ mod 2^{16}. For each of the two analysis, the following experiments have been carried out four times: First, the secret key and initialization vector have been randomly set. Next, a sequence consisting on 2^{25} outputs has been generated. Finally, the xor and difference values have been computed. In Table 7, the statistical properties are summarized.

Summarizing, the probabilities associated with LAMED output are well within the limits set by the specification. Our

Table 6
Bit–Byte Prediction tests as in David Sexton's battery

	LAMED-EPC
Test	p-value
Bit Prediction A Test	0.8421
Bit Prediction B Test	0.6966
Bit Prediction C Test	0.8499
Bit Prediction D Test	0.8081
Bit Prediction E Test	0.4742
Byte Prediction A Test	0.3263
Byte Prediction B Test	0.6074
Byte Prediction C Test	0.5686
Byte Prediction D Test	0.3254
Byte Repetition Test	0.4184

Table 7
Analysis of the xor and substraction

Experiment	xor ($O_i \oplus O_{i+1}$)	Difference ($(O_i - O_{i+1})$)
Experiment #1		
Entropy	7.999999 bits/byte	7.999999 bits/byte
Compression Rate	0%	0%
χ^2 Statistic (byte)	267.81 (50%)	259.79 (50%)
χ^2 Statistic (16-bit)	65370.5898 (67.55%)	65209.1562 (62.26%)
Arithmetic mean	127.5060	127.4997
Monte Carlo π estimation	3.141522675 (0.00%)	3.141581433 (0.00%)
Serial correlation coefficient	−0.000041	−0.000093
Experiment #2		
Entropy	7.999999 bits/byte	7.999999 bits/byte
Compression rate	0%	0%
χ^2 statistic (byte)	247.43 (50%)	267.88 (50%)
χ^2 statistic (16-bit)	65484.3281 (55.60%)	65607.109375 (42.14%)
Arithmetic mean	127.4969	127.4977
Monte Carlo π estimation	3.142054749 (0.01%)	3.141537534 (0.01%)
Serial correlation coefficient	0.000097	0.000049
Experiment #3		
Entropy	7.999999 bits/byte	7.999999 bits/byte
Compression rate	0%	0%
χ^2 statistic (byte)	240.76 (50%)	264.96 (50%)
χ^2 statistic (16-bit)	65673.964844 (35.09%)	65182.406250 (83.56%)
Arithmetic Mean	127.4910	127.4954
Monte Carlo π estimation	3.141983490 (0.01%)	3.141622471 (0.00%)
Serial correlation coefficient	−0.000078	0.000112
Experiment #4		
Entropy	7.999999 bits/byte	7.999999 bits/byte
Compression rate	0%	0%
χ^2 statistic (byte)	259.95 (50%)	259.79 (50%)
χ^2 statistic (16-bit)	65346.839844 (69.88%)	65614.957031 (41.29%)
Arithmetic mean	127.4997	127.5001
Monte Carlo π estimation	3.142095337 (0.02%)	3.141177521 (0.01%)
Serial correlation coefficient	0.000064	0.000044

prediction analysis does not suggest any evidence that the output could be predicted significantly better than just by chance by knowledge of prior outputs without knowing the secret key and the IV. Although the period of LAMED has not been exactly determined, in Section 5.1 a file of 2^{30} bytes (4 Gb) was analyzed without finding any evidence that those bytes behave differently from what would be expected from a random variable, thus proving that the period was larger and, in any case, sufficiently large for the intended application. Let's consider the following simple scenario: a tag having LAMED-EPC on chip, and a reader which interrogates this tag every 5 ms. Then, under these conditions, the reader could be continuously interrogating the tag at least during fifteen days, which clearly fulfills the needs of the vast majority of applications.

6. Hardware complexity

In this section, we explain in detail one architectural design for LAMED. Before starting, it is necessary to consider whether we need a parallel or serial architecture. As mentioned in Section 2, class-1 generation-2 tags have severe temporal requirements, for around 450 tags should be readable every second [6,21]. Power consumption is another important requirement. Tags are passive, so we should limit their power consumption as much as possible. One of the parameters with a major influence on this target is clock frequency. Following other authors in the RFID area [31], we assume that clock frequency must be in the range of KHz, at some value around 100 KHz, implying a clock cycle consumption of 0.01 ms. With these conditions, a tag can use up to 220 clock cycles (2.2 ms) for the whole random number generation phase. For this, we have decided to process 32-bit streams in parallel. Next, a code for the implementation of LAMED is included, where ^ is the xor operator and vrotdk(v,k) means rotations of v, k times.

```
#1 If n is odd
#2 a0=a1+iv
#3 a1=out^s
#4 If n is even
#5 a0=a1^iv
#6 a1=out+s
#1   aux1=a0+a1;              #12  aux3=aux3 ^ aux1;
#2   aux2=a0 ^ a1;            #13  aux3=vrotdk(aux3,3);
#3   aux3=vrotdk(aux1,5);     #14  aux33=aux3+a1;
#4   aux3=aux3+aux2;          #15  aux3=vrotdk(aux3,2);
#5   aux3=vrotdk(aux3,3);     #16  aux3=aux3+aux1;
#6   aux3=aux3 ^ aux1;        #17  aux3=vrotdk(aux3,4);
#7   aux3=vrotdk(aux3,4);     #18  aux3=aux3 ^ a1;
#8   aux3=a1+aux3;           #19  aux3=vrotd(aux3);
#9   aux3=vrotdk(aux3,2);     #20  aux3=aux3+aux2;
#10  aux3=aux3+aux1;          #21  aux3=vrotdk(aux3,2);
#11  aux3=vrotdk(aux3,2);     #22  out=aux1 ^ aux3;
```

The architecture of LAMED, see Fig. 2, can be divided into four main parts:

- *Input Selection Unit*. The PRNG will be initialized with a 32-bit initialization vector (*iv*). Furthermore, the tag has *s* stored, a 32-bit secret which is only known by the tag and authorized readers (possibly via a database query). After initialization, the state of the PRNG will be updated as in Expression 4 and 5.
- *Arithmetic Logic Unit (ALU)*. Due to the structure of the PRNG, we have only included the xor and sum operators.
- *Registers*. We have used three registers for the core of our PRNG. Two will be used to store a0+a1 (aux1) and a0^a1 (aux2) for the generation process of each 32-bit output. Additionally, once an output has been obtained, these registers will be used for the temporal storage of the new two inputs (updated state) to the PRNG. The third register will be used to execute the right rotations and to store the intermediate results. aux1 and aux2 must be updated after the generation of a new 32-bit stream. Moreover, two additional registers will be used to store the initialization vector (*iv*) and the secret (*s*).

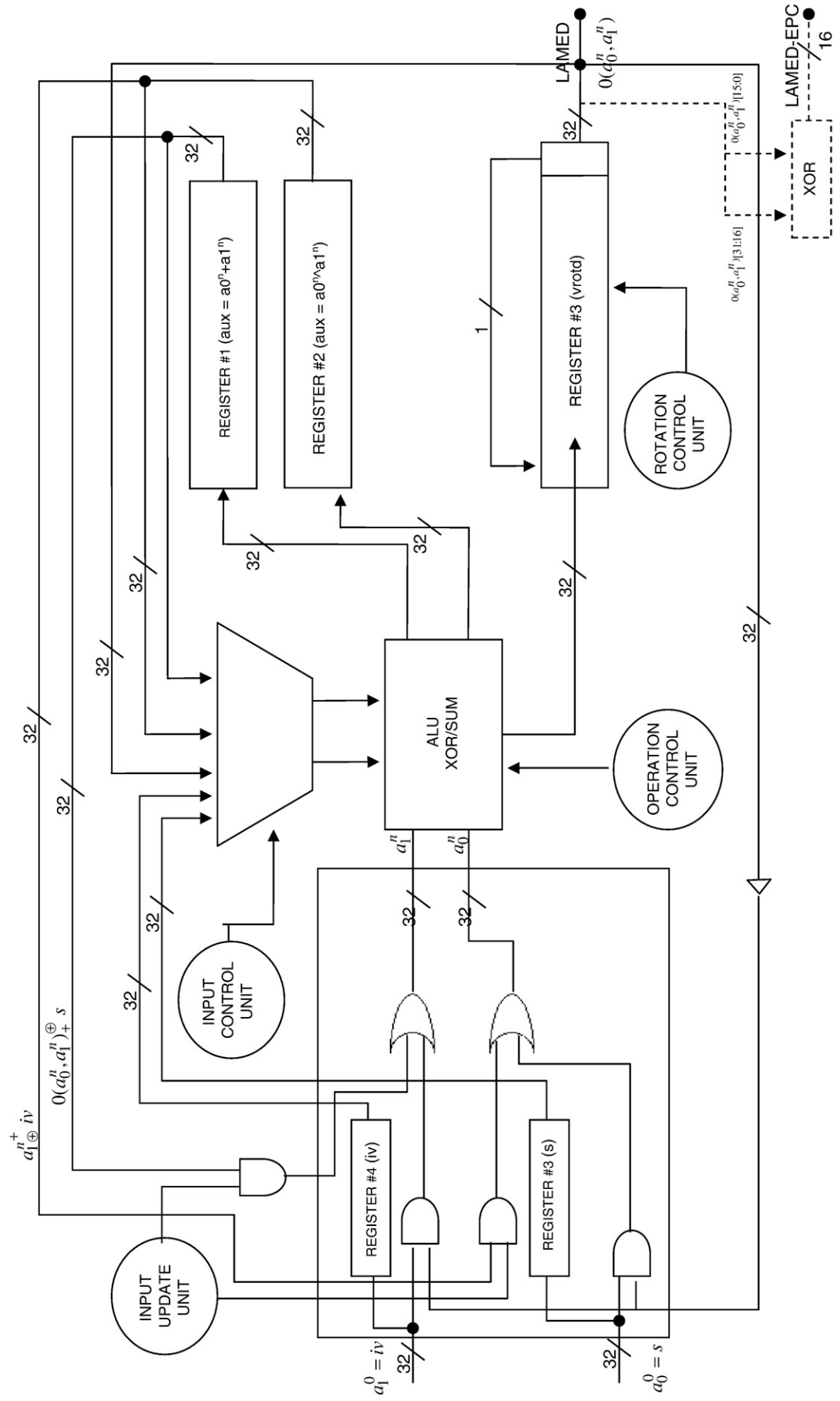
LAMED
LAMED-EPC
16
XOR
REGISTER #1 (aux = a0^a1^n)
REGISTER #2 (aux = a0^a1^n)
REGISTER #3 (vrotd)
ROTATION CONTROL UNIT
ALU XOR/SUM
OPERATION CONTROL UNIT
INPUT CONTROL UNIT
INPUT UPDATE UNIT
REGISTER #4 (iv)
REGISTER #3 (s)
32
1

Fig. 2.

Table 8
Number of logic gates

	LAMED		LAMED-EPC	
Architecture units	Gate counting		Gate counting	
Arithmetic logic unit	xor	32 LG	xor	32 LG
	sum	192 LG	sum	192 LG
16-bit unit	––		16 LG	
Update unit	10 LG		10 LG	
Registers[1]	Aux1–2	512 LG	Aux1–2	512 LG
	Aux3	264 LG	Aux3	264 LG
	Aux4–5	512 LG	Aux4–5	512 LG
Control (20%)	44 LG		50 LG	
Total	1566 Logic Gates		1585 Logic Gates	

- *16-bit Unit Output*. This unit performs a split xor operation. The 32-bit output is divided in two halves, $MSB_{31:16}$ and $LSB_{15:0}$, and the xor of these two halves will be outputted.

Directly derived from all of the above, we reckon that 186–194 clock cycles (1.86–1.94 ms) are needed for generating each 32–16 output with LAMED or LAMED-EPC. Besides, these results imply a throughput of between 17.2 and 8.2 kbps, respectively. We can then conclude that the temporal requirements are accomplished with enough margin in both cases.

Another important aspect we should consider is gate counting. An overestimation of this factor is presented in Table 8. In this calculation, an extra 20% of logic gates are added for control functions, and 8 additional gates are needed for implementing a flip flop[1] as in [32].

Finally, we compare LAMED with other recent proposals aimed at very restricted hardware environments. As already mentioned, and for comparison's sake, we assume that clock frequency is set to 100 KHz. We would like to emphasize the interesting work of Feldhofer et al., who have proposed a very efficient AES implementation (3400 LG, 12 Kbps) [33,34]. When comparing both we observe that LAMED gate count is 48% lower, while its throughput is in the same range. Another interesting work in this direction is the ECRYPT Stream Cipher Project (eSTREAM), which tries to identify new secure stream ciphers. eSTREAM profile 2 is focused on stream ciphers oriented for hardware with very restricted resources such as limited storage, very low gate count, or minimal power consumption. Two of the candidates that have been selected for the Phase 2 (in profile 2) are Grain and Trivium. Grain (2133 LG, 100 Kbps), and Trivium (3000 LG, 100 Kbps) are synchronous stream ciphers proposed by Hell et al. [35–37], and by De Cannire et al. [38,35]. These ciphers have superior throughput but an extra 90% (Trivium) and 34% (Grain) logic gates are needed in comparison with LAMED. Moreover, the security of these ciphers is under a dark cloud, as there are some recent and powerful cryptanalytic results [39,40].

7. Conclusions and future work

There are different ways to identify objects, animals and people. RFID technologies offer certain benefits that make them better suited than other technologies: item-level tracking, no need for alignment, high read rates, variety of form factors, rewritability, etc. The usage of RFID technology is steadily increasing, and the predicted "Internet of the things" may not be far away. [3] However, in order to became a reality it is necessary to avoid proprietary solutions, and one of the most important standards proposed is EPC Class-1 Generation-2, which has been ratified by both EPCglobal and ISO.

In the EPC-C1G2 specification the use of a PRNG as a basic primitive for building up security solutions and protocols has been proposed for low-cost RFID tags. For this reason, we propose a new PRNG compliant with this specification. We have shown that the proposed PRNG LAMED not only fulfils the requisites of the EPC-C1G2 standard, but also passes some very demanding randomness test batteries (ENT, DIEHARD, NIST, and SEXTON).

At the same time, its hardware complexity has been analyzed and we conclude that the gate count (around 1.6 K gates) is well below the 4 K gates which is the theoretical limit that can be devoted to security-related tasks [16] on low-cost RFID tags. Additionally, the temporal requirements are accomplished with an adequate margin, needing only around 1.9 ms to generate a 32-bit random number.

Finally, we should make a comment on the security of EPC-C1G2 specification. Analyzing the specification in detail, it is possible to detect that the 16-bit random numbers are not used for security but only to establish a new session. Indeed, this new specification has several important security limitations and pitfalls, as has been indicated in Section 1. To conclude, we hope that security will be considered more carefully in the coming generation 3 standard.

References

[1] EPCglobal, http://www.epcglobalinc.org/. [Online]. Available: http://www.epcglobalinc.org/.
[2] ISO—Internacional Organization for Standardization, http://www.iso.org/. [Online]. Available: http://www.iso.org/.
[3] Class-1 Generation-2 UHF air interface protocol standard version 1.0.9: "Gen 2", http://www.epcglobalinc.org/, January 2005. [Online]. Available: http://www.epcglobalinc.org/.
[4] D. Bailey, A. Juels, Shoehorning security into the EPC standard, Manuscript in submission, 2006.
[5] K. Hyun Kim, E. Young Choi, S. Mi Lee, D. Hoon Lee, Secure EPCglobal Class-1 Gen-2 RFID system against security and privacy problems, Proc. of OTM-IS'06, ser. LNC, vol. 4277, 2006, pp. 362–371.
[6] A. Juels, Strengthening epc tags against cloning, Manuscript, RSA Laboratories, March 2005.
[7] D. Nguyen Duc, J. Park, H. Lee, K. Kim, Enhancing security of epcglobal gen-2 RFID tag against traceability and cloning, Proc. of Symposium on Cryptography and Information Security, 2006.
[8] S. Weis, S. Sarma, R. Rivest, D. Engels, Security and privacy aspects of low-cost radio frequency identification systems, Proc. of Security in Pervasive Comp., ser. LNCS, vol. 2802, 2004, pp. 201–212.
[9] D. Molnar, D. Wagner, Privacy and security in library RFID: Issues, practices, and architectures, Proc. of ACM CCS'04, 2004, pp. 210–219.
[10] C. Chatmon, T. Van Le, M. Burmester, Secure anonymous RFID authentication protocols, Technical Report TR-060112, 2006.

[3] This term was originally attributed to the Auto-ID Center.

[11] T. Dimitriou, A lightweight RFID protocol to protect against traceability and cloning attacks, Proc. of PerCom'06, 2006.
[12] S. Lee, T. Asano, K. Kim, RFID mutual authentication scheme based on synchronized secret information, Proc. of Symposium on Cryptography and Information Security, Japan, 2006.
[13] K. Rhee, J. Kwak, S. Kim, D. Won, Challenge-response based RFID authentication protocol for distributed database environment, Proc. of SPC'05, ser. LNCS, vol. 3450, 2005, pp. 70–84.
[14] G. Tsudik, YA-TRAP: yet another trivial RFID authentication protocol, Proc. of PerCom'06, 2006.
[15] ISO/IEC 18000-6:2004/Amd:2006, http://www.iso.org/, 2006.
[16] D. Ranasinghe, D. Engels, P. Cole, Low-cost RFID systems: Confronting security and privacy, Auto-ID Labs Research Workshop, 2004.
[17] M. Roberti, K.A. Vice, Y. Marguire, M. Reynolds, R. Dunn, EPC Generation 2: Everything you need to know, RFID Journal Webinars, Technical Report, 2005.
[18] J. Koza, Evolving a computer program to generate random number using the genetic programming paradigm, Proc. of the 4th Int. Conference on Genetic Algorithms, Morgan Kaufmann, 1991, pp. 37–44.
[19] The lil-gp genetic programming system, http://garage.cps.msu.edu/software/lil-gp/lilgpindex.html.
[20] T. Lohmmann, M. Schneider, C. Ruland, Analysis of power constraints for cryptographic algorithms in mid-cost RFID tags, Proc. of Cardis'06, ser. LNCS, vol. 3928, 2006, pp. 278–288.
[21] R. Forré, The strict avalanche criterion: Spectral properties of boolean functions and an extended definition, Proc. of Crypto'88, ser. LNCS, 1990, pp. 450–468.
[22] M. Matsumoto, et al., Mersenne twister: A 623-dimensionally equidistributed uniform PRNG, ACM Trans. Model. Comput. Simul. 8 (1) (1998) 3–30.
[23] A. Biryukov, A. Shamir, Cryptanalytic time-memory-data tradeoffs for stream ciphers,, Proc. of Advances of Cryptology-Asiacrypt, vol. 1976, 2000, pp. 1–13.
[24] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, S. Vo, A statistical test suite for random and pseudorandom number generators for cryptographic applications, http://csrc.nist.gov/rng/, Technical Report, 2001.
[25] G. Marsaglia, The Marsaglia Random Number CDROM Including the DIEHARD Battery of Tests of Randomness, http://stat.fsu.edu/pub/diehard,1996.
[26] G. Marsaglia, W. Tsang, Some difficult-to-pass tests of randomness, J. Stat. Softw. 7 (3) (2002).
[27] J. Walker, Randomness Battery, http://www.fourmilab.ch/random/,1998.
[28] David Sexton's battery, http://www.geocities.com/da5id65536, 2005.
[29] M. Matsui, Linear cryptanalysis method for DES cipher, Proc. Of EUROCRYPT'93, 1994, pp. 386–397.
[30] F.-X. Standaert, G. Piret, J.-J. Quisquater, Cryptanalysis of block ciphers: A survey, Technical Report, 2003.
[31] M. Feldhofer, S. Dominikus, J. Wolkerstorfer, Strong authentication for RFID systems using the AES algorithm, Proc. of Ches'04, vol. 3156, 2004, pp. 357–370.
[32] M. Hell, T. Johansson, W. Meier, Grain —a stream cipher for constrained environments, Handout of the Ecrypt Workshop on RFID and Lightweight Crypto, 2005.
[33] M. Feldhofer, K. Lemke, E. Oswald, F. Standaert, T. Wollinger, J. Wolkerstorfer, State of the art in hardware architectures, Technical report, ECRYPT Network of Excellence in Cryptology, 2005.
[34] M. Feldhofer, J. Wolkerstorfer, V. Rijmen, AES implementation on a grain of sand, Proc. of IEEE on IS, vol. 152, 2005, pp. 13–20.
[35] T. Good, W. Chelton, M. Benaissa, Review of stream cipher candidates from a low resource hardware perspective, http://www.ecrypt.eu.org/stream/hw.html.
[36] M. Hell, T. Johansson, W. Meier, Grain: a stream cipher for constrained environments, http://www.ecrypt.eu.org/stream/.
[37] M. Hell, T. Johansson, W. Meier, A stream cipher proposal: Grain-128, http://www.ecrypt.eu.org/stream/.
[38] C. de Canniere, B. Preneel, Trivium specifications, http://www.ecrypt.eu.org/ stream/.
[39] B. Berbain, H. Gilbert, A. Maximov, Cryptanalysis of grain, http://www.ecrypt.eu.org/ stream/.
[40] O. Kucuk, Slide resynchronization attack on the initialization of grain 1.0, http://www.ecrypt.eu.org/ stream/.

Pedro Peris-Lopez is an Assistant Professor at the Computer Science Department of Carlos III University of Madrid. He has a M.Sc. in Telecommunications Engineering. His research interests are in the field of protocols design, authentication, privacy, lightweight cryptography, etc. Nowadays, his research is focused on Radio Frequency Identification Systems (RFID). In these fields, he has published a great number of papers in specialized journals and conference proceedings.

Julio C. Hernandez-Castro is an Associate Professor at the Computer Science Department of Carlos III University of Madrid. He has a B.Sc. in Mathematics, a M.Sc. in Coding Theory and Network Security, and a Ph.D. in Computer Science. His interests are mainly focused in cryptology, network security, steganography and evolutionary computation. He loves chess and dreams of becoming, one day, a professional chess player. He also loves Recreational Mathematics and has published some fun articles in Journals specialized in this area.

Juan M. Estevez-Tapiador is an Associate Professor at the Computer Science Department of Carlos III University of Madrid. He holds a M.Sc. in Computer Science from the University of Granada (2000), where he obtained the Best Student Academic Award, and a Ph.D. in Computer Science (2004) from the same university. His research is focused on cryptology and information security. In these fields, he has published around 40 papers in specialized journals and conference proceedings. He is a member of the program committee of several conferences related to information security and serves as a regular referee for various journals.

Arturo Ribagorda is a Full Professor at Carlos III University of Madrid, where he is also the Head of the Cryptography and Information Security Group and currently acts as the Director of the Computer Science Department. He has a M.Sc. in Telecommunications Engineering and a Ph.D. in Computer Science. He is one of the pioneers of computer security in Spain, having more than 25 years of research and development experience in this field. He has authored 4 books and more than 100 articles in several areas of information security. Additionally, he is a member of the program committee of several conferences related to cryptography and information security.