Computer Standards & Interfaces 31 (2009) 372–380

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Cryptanalysis of a novel authentication protocol conforming to EPC-C1G2 standard

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

Computer Science Department, Carlos III University of Madrid, Spain

ARTICLE INFO

Article history:
Received 16 May 2007
Received in revised form 7 March 2008
Accepted 4 May 2008
Available online 18 May 2008

Keywords:
RFID
EPC-C1G2
Security
Authentication
Cryptanalysis

ABSTRACT

In 2006, the standard EPC Class-1 Generation-2 (EPC-C1G2) was ratified both by EPCglobal and ISO. This standard can be considered as a "universal" specification for low-cost RFID tags. Although it represents a great advance for the consolidation of RFID technology, it does not pay due attention to security and, as expected, its security level is very low. In 2007, Chien et al. published a mutual authentication protocol conforming to EPC-C1G2 which tried to correct all its security shortcomings. In this article, we point out various major security flaws in Chien et al.'s proposal. We show that none of the authentication protocol objectives are met. Unequivocal identification of tagged items is not guaranteed because of possible birthday attacks. Furthermore, an attacker can impersonate not only legitimate tags, but also the back-end database. The protocol does not provide forward security either. Location privacy is easily jeopardized by a straightforward tracking attack. Finally, we show how a successful auto-desynchronization (DoS attack) can be accomplished in the back-end database despite the security measures taken against it.

1. Introduction

RFID technology today is employed in a great number of applications. However, security aspects do not play an important role in the introduction of this promising technology. We should have learned from past errors such as those related to bluetooth or WiFi technology. However, the security level offered by commercial solutions is very low (e.g. Texas Instruments DST tags [4], Philips Mifare cards [14]). The two main problems related to RFID technology are privacy and tracking:

Privacy: Tag content, which may include sensitive information, is revealed when insecure tags are interrogated by readers. Tags and readers should be authenticated to correct this problem. However, readers are frequently not authenticated, and tags usually answer in a completely transparent way.

Tracking: A problem closely related to privacy is tracking, or violations of location privacy. Even if access of tag content were only allowed to authorized readers, tracking still might not be guaranteed. The answer provided by tags is usually a constant value (i.e. a static identifier). Under this assumption, an attacker will be able to establish an association between tags and its owners. Additionally, we can relax our conditions and assume that tags only contain product codes rather than a unique identifier. In spite of this, Weis et al. claims that tracking will still be possible by using an assembly of tags (a constellation) [29].

In addition to the previous threats, there are some other aspects that must be considered: eavesdropping, physical attacks, counterfeiting, active attacks, denial of service, etc. For depth in all these matters we recommend reading of [12,21,22] which provide surveys of the most important advances in RFID technology.

Each time a new protocol is defined, the class of tag for which the proposed protocol is appropriate should also be specified. In general terms, a tag contains a microchip with some computational and storage capabilities, and a coupling element, such as an antenna coil for communication. Tags can be classified according to two main criteria:

The type of memory: The memory element serves as writable and non-writable data storage. Tags can be programmed to be read-only, write-once read-many, or fully rewritable. Depending on the kind of tag, tag programming can take place at the manufacturing level or at the application level.

The source of power: A tag can obtain power from the signal received from the reader, or it can have its own internal power source. The way the tag gets its power generally defines the category of the tag: 1. Passive tags do not have internal source of power. They harvest their power from the reader that sends out electromagnetic waves. These kind of tags are restricted in their read/write range as they rely on RF electromagnetic energy from the reader for both power and communication. 2. Semi-passive tags use a battery to run the microchip's circuitry but communicate by harvesting power from the reader signal. 3. Active tags possess a power source that is used to run the microchip's circuitry and to broadcast a signal to the reader.

Another relevant parameter is tag price, in which we mainly distinguish between high-cost and low-cost RFID tags. We note that depending on the class of tag, the security level that can be supported will also be different. For example, the security level of a tag used in e-passports should not be the same as that of a low-cost tag employed in the supply chain (e.g. tags compliant to EPC Class-1 Generation-2

* 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).

0920-5489/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.csi.2008.05.012

Table 1
Specifications for low-cost and high-cost RFID tags

	Low-cost RFID tag	High-cost RFID tag
Standards	EPC Class-1 Generation-2 ISO/IEC 18000-6C	ISO/IEC 14443 A/B
Power Source	Passively powered	Passively powered
Storage	32–1 K bits	32 KB–70 KB
Circuitry (security processing)	250 K–4 K gates Standard cryptographic primitives cannot be supported	Microprocessor Implement 3DES, SHA-1, RSA
Reading distance (commercial devices)	Up to 3 m	Around 10 cm
Price	0.05–0.1 €	Several euros
Physical attacks	Not resistant	Tamper resistance EAL 5+ security level

specification). To clarify the kind of systems we refer to as low-cost/high-cost RFID tags, Table 1 summarizes their specifications, these being relevant to current-commercial RFID tags.

2. Motivation

RFID is a relatively heterogeneous technology with a significant number of connected standards. As in [21], standards can be classified according to five main categories: contactless integrated circuit cards, RFID in animals, item management, near field communication (NFC) and EPC. Fig. 1 summarizes the most important of those. Within these standards, one of the most relevant is the EPCglobal Class-1 Gen-2 RFID specification (EPC-C1G2) [8]. It was adopted in 2004, and eighteen months later was ratified by ISO and published as an amendment to its ISO/IEC 18000-6 standard.

EPC-C1G2 tags are passive, so they receive their energy from the reader's RF waveform. The very constrained computational and storage capabilities dictates that these tags cannot afford the use of traditional cryptographic primitives. Following the standard, tags only support on chip a 16-bit Pseudo-Random Number Generator (PRNG) and a 16-bit Cyclic Redundancy Code (CRC). Tag memory is insecure, and susceptible to physical attacks. A 32-bit kill command is used to permanently disable the tag. A 32-bit access PIN is required to trigger it into secure mode.

Despite the great advance that EPC-C1G2 represents in terms of communication compatibility and performance between tags, and the major implications it could have for the widespread introduction of this technology, the security level of the standard is extremely weak. The two most relevant operations for managing tag populations are inventory and access. These two operations present serious security flaws, as described below:

- Inventory command: the private information stored in the tag is compromised by any attacker with access to the radio channel, because the EPC is transmitted as plain text. Additionally, an adversary can easily impersonate a legitimate tag: the attacker can obtain the EPC of any tag by simply eavesdropping the air channel, as this EPC will be emitted by the tag when the reader sends any request. After obtaining this value, the attacker can use it to impersonate the tag. Finally, as tags always transmit a fixed EPC value, this could be associated with its holder allowing the easy tracking of user movements and behaviors.
- Access command: the security of the access command is extremely weak, so performing a passive attack is very simple. An attacker listening in to the backward and forward channel (a very realistic assumption when using the air channel) can pick up the random number sent by the tag. Then the attacker can decrypt the ciphertext sent by the reader by performing an XOR (addition modulo 2) with

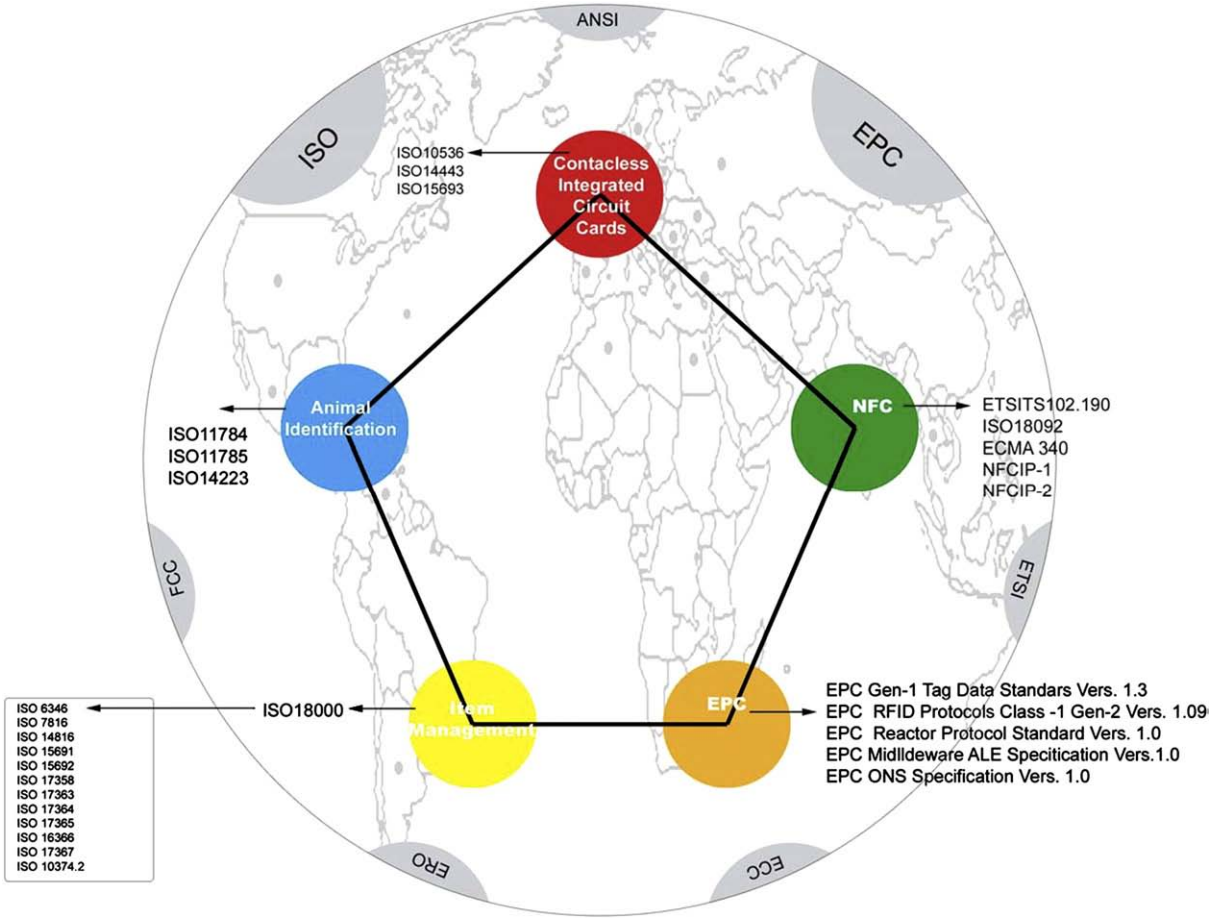

Fig. 1. RFID standards.

the previous eavesdropped random number. So the plaintext or PIN can be obtained by this quite simple mechanism, which constitutes an important security pitfall.

In spite of the serious security failures of EPC-C1G2, this standard could already be considered a great success having been adopted by many RFID manufacturers [1]. This is why great efforts have lately been made to develop new security features (e.g. authentication or key exchange protocols) compliant with the EPC-C1G2 standard [3,13,15,16,18].

3. Related work

The vast majority of designs for security protocols for RFID either do not conform to the EPC-C1G2 [6,7,10,11,25,31] specification or they suffer from major security flaws. In this section, we briefly present some recent attempts to raise the security level of low-cost RFID tags, whilst still conforming to EPC-C1G2.

In [13], Juels shows that EPC tags are vulnerable to elementary cloning and counterfeiting attacks. The proposals he makes for solving these problems, while resistant to skimming attacks, are vulnerable to eavesdropping and active attacks, etc.

In 2006, Juels et al. examine various ways for RFID tags to perform cryptographic functions while remaining compliant with EPC-C1G2 [3]. Their main idea is to take an expansive view of EPC tag memory. Instead of considering memory merely as an storage media, they use it as an input/output way of interfacing with a cryptographic module within the tag. Read/write commands may therefore carry out cryptographic values, such as messages in a challenge–response protocol. Their work clearly shows the need for mutual authentication between readers and tags. However, the assumption that a low-cost tag might support on-board cryptographic module is not realistic, at least at the present time.

Karthikeyan and Nesterenko [15] proposed an efficient tag identification and reader authentication protocol based on simple XOR and matrix operation. Two matrixes and a key are stored in both the tag (K, M_1, M_2^{-1}) and the back-end database (K, M_1^{-1}, M_2). Once the tag is identified, the reader sends to the tag the submessages Y, Z. The first is used to authenticate the tag and the second to update the key. However, an attacker can replay Z with a random Z'. Upon receiving Y, Z', the tag will be authenticated and will update the key wrongly. So the legitimate reader and the tag will not be able to authenticate each other any more. Additionally, the protocol is also vulnerable to replay attacks and privacy location is not guaranteed [5].

Duc et al. propose a tag-to-back-end database authentication protocol [18]. The security of Duc et al.'s protocol is based on key synchronization between tags and back-end database. The last message of the protocol is comprised of an EndSession command, which is sent to both tags and readers. Interception of one of these messages will cause a synchronization loss between the tag and the server. So the tag and the reader will not be able to authenticate any more, which is extremely serious. This protocol also presents backward secrecy problems, as compromise of the EPC allows an attacker to trace back all past communications.

Konidala and Kim [16] produced an interesting paper which tried to correct some of the security shortcomings of the EPC-C1G2 specification. The authors hold that the proposed scheme frustrates the access password acquisition by a simple XOR operation, against what happened in the specification. However, Lim and Li [17] show how a passive attacker can recover the password of the tag by eavesdropping over a single run of the protocol and performing some correlation analysis on the captured information.

4. Chien et al. protocol

In [5], Chien et al. propose a mutual authentication protocol for improving the security performance of EPC-C1G2. Their scheme consists of two phases: an initialization phase and authentication phase.

4.1. Initialization phase

For each tag denoted as T_i, the server randomly selects an initial authentication key K_{i_o} and a initial access key P_{i_o}. These two values, joined with the EPC (EPC_i), are stored in the tag. The authentication and access key will be updated after each successful authentication. For each tag, the server S (back-end database) maintains a record of six values: (1) EPC_i; (2) the old authentication key for this tag (K_{old}), which is initially set to K_{i_o}; (3) P_{old} denotes the old access key for this tag, which is initially set to P_{i_o}; (4) K_{new} denotes the new authentication key, which is initially set to K_{i_o}; (5) P_{new} denotes the new authentication key, which is initially set to P_{i_o}; (6) Data denotes all the information about the tagged object.

4.2. The (n+-1) authentication phase

$R \rightarrow Ti$:	N_1 The reader sends a random nonce N_1 as a challenge to the tag.
$T_i \rightarrow R \rightarrow S$:	M_1, N_1, N_2 The tag generates a random number N_2, computes $M_1 = CRC(EPC_i \| \| N_1 \| \| N_2) \oplus K_{in}$, and sends the value back to the reader, which will forward these values to the server. The reader interactively selects an entry (EPC_i, K_{old}, K_{new}, P_{old}, P_{new}, Data) from its database, computes $I_{old} = M_1 \oplus K_{old}$ and $I_{new} = M_1 \oplus K_{new}$, and checks whether any of these two equations hold $I_{old} = CRC(EPC_i \| \| N_1 \| \| N_2)$ $I_{new} = CRC(EPC_i \| \| N_1 \| \| N_2)$. This is designed to be a way of avoiding desynchronization attacks. The process is repeated until a match is found in the database, thus implying a successful authentication of the tag. If no match is found, a failure message is sent to the reader, and the authentication process is stopped.
$S \rightarrow R$:	M_2, Data After a successful authentication, the server computes $M_2 = CRC(EPC_i \| \| N_2) \oplus P_{old}$ or $M_2 = CRC(EPC_i \| \| N_2) \oplus P_{new}$, depending on which value (K_{old}, K_{new}) satisfies the equation in the previous step. It also updates $K_{old} = K_{new}$, $P_{old} = P_{new}$, $K_{new} = PRNG(K_{new})$ and $P_{new} = PRNG(P_{new})$. The server sends M_2, Data to the reader.
$R \rightarrow T_i$:	M_2 Upon receiving M_2, the tag verifies whether the equation $M_2 \oplus P_{in} = CRC(EPC_i \| \| N_2)$ holds. If so, it updates its keys $K_{in+1} = PRNG(K_{in})$ and $P_{in+1} = PRNG(P_{in})$.

5. Cyclic Redundancy Codes — CRCs

A Cyclic Redundancy Code (CRC) is a checksum algorithm that can be used to detect transmission errors (typically one or two bit flips, or bursts) in a very efficient way. CRCs operate by interpreting input binary sequences as polynomial coefficients that they divide over a prefixed polynomial in order to obtain a remainder, which, in its binary expression, constitutes the crc value.

CRCs are completely linear, so they shouldn't be used in cryptographic applications as they cannot detect malicious changes by a knowledgeable attacker [2,23,27,30]. To illustrate this property, the Hamming Distance (HD) can be used. The HD of a CRC polynomial is the minimum possible number of bit errors that is undetected by computing the CRC. For example, if a CRC has a HD of 3, any combinations of 1 or 2 bit errors will be detected, but there is at least one combination of 3 bit errors that will pass undetected. Cryptographic hash functions should therefore be used for this purpose.

Computing a crc value for a given binary stream is essentially dividing the polynomial associated with this stream by another fixed polynomial (that depends on the particular CRC implementation) and computing a remainder. The stream should be multiplied by x^N (being N the degree of the crc polynomial) prior to division. That is to say, computing the crc of a polynomial $i(x)$ is basically finding a remainder $r(x)$ so that,

$$i(x) \cdot x^N = d(x) \cdot p(x) + r(x) \quad \text{with} \quad |r(x)| < |p(x)| \tag{1}$$

The reader is referred to [28] where a detailed explanation about the selection of the $p(x)$ polynomial is included. Some popular polynomials are the followings:

$$CRC-8 = x^8 + x^5 + x^4 + 1$$
$$CRC-12 = x^{12} + x^{11} + x^3 + x^2 + x + 1$$
$$\text{CRC-16} = x^{16} + x^{15} + x^2 + 1$$
$$CRC-CCITT = x^{16} + x^{12} + x^5 + 1$$
$$CRC-32 = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$
$$CRC-64 = x^{64} + x^4 + x^3 + x + 1$$

The EPC-C1G2 specification proposes the use of CRC-CCITT which detects all single and double errors, all errors with an odd number of bits, all burst errors of length 16 or less, 99.997% of 17-bit error bursts, and 99.998% of 18-bit and longer bursts.

The hardware requirements required by a CRC generator are not very demanding. Specifically, an n-bit CRC consists of an n-bit shift register with some XOR gates, as illustrated in Fig. 2. To compute the CRC:

1 Load the register with the Preset value.
2 Augment the message by appending N zeros to the end of it $(i(x)\cdot x^N)$.
3 While (more message bits)
  3.1 Shift the register left by one bit, reading the next bit of the augmented message into register bit position 0.
  3.2 If (a 1 bit popped out of the register during the above step)
   Register = Register XOR $p(x)$
  End While
4 The register contains the remainder

5.1. CRC properties

Due to their linearity, CRCs have some properties that, from the security point of view, one can label as bad. In fact, we will show that one of these "bad" properties (derived from their linear structure) will be enough to successfully attack Chien et al.'s mutual authentication protocol in various ways.

Theorem 1. *For any CRC (independent of its divider polynomial) and for any values a, b, c and d* $\in F_2[x]$*, it holds that:*

$$\text{CRC}(a||b)\oplus\text{CRC}(c||d) = \text{CRC}(a\oplus c||b\oplus d) \quad (2)$$

Proof. From the definition in Eq. (1) above, one can write:

$$\text{CRC}(a||b) = \left(a\cdot x^N\oplus b\right)\cdot x^N\oplus d_1(x)\cdot p(x) \quad (3)$$

$$\text{CRC}(c||d) = \left(c\cdot x^N\oplus d\right)\cdot x^N\oplus d_2(x)\cdot p(x) \quad (4)$$

for certain polynomials $d_1(x)$ and $d_2(x) \in F_2[x]$. Substituting these values in the left side of Eq. (2) we obtain the following:

$$\left(a\cdot x^N\oplus b\right)\cdot x^N\oplus d_1(x)\cdot p(x)\oplus\left(c\cdot x^N\oplus d\right)\cdot x^N\oplus d_2(x)\cdot(x). \quad (5)$$

Rearranging terms in this expression we get:

$$\left((a\oplus c)\cdot x^N\oplus(b\oplus d)\right)\cdot x^N\oplus(d_1(x)\oplus d_2(x))\cdot p(x) \quad (6)$$

that is the corresponding expression for $\text{CRC}(a\oplus c||b\oplus d)$ (analogously to Eqs. (3) and (4). □

Corollary 1. *In particular, if in Eq. (1) we have a=c, then,*

$$\text{crc}(a||b)\oplus\text{crc}(a||d) = \text{crc}(a\oplus a||b\oplus d) = \text{crc}(0||b\oplus d) = \text{crc}(b\oplus d) \quad (7)$$

because $0\cdot x^N \equiv 0\cdot p(x)$.

This is the property we will use to our advantage in attacking Chien et al.'s protocol (and, for that matter, any other protocol relying on the use of a CRC as a means of concealing secrets). It is important to point out that this holds for every CRC implementation, independently of its length and crc polynomial (CRC-8, CRC-16, CRC-32, CRC-64, etc).

6. Vulnerabilities of Chien's protocol

In this section we will analyze the most important vulnerabilities in Chien et al.'s protocol.

6.1. Unequivocal identification

The use of RFID tags offers several advantages over barcodes: data can be read automatically, without line of sight, and through a non-conducting material such as cardboard or paper, at a rate of hundreds of times per second, and from a distance of several meters [12,21]. But there is a fundamental difference between barcode technology and RFID. Barcodes use Universal Product Codes (UPC) to identify the class of items. RFID technology replaces UPC with the Electronic Product Code (EPC) that allows the unequivocal identification of tagged items.

The Tag Data Specification [9] does not provide any specific guidance for using EPCs in UHF Class-1 Generation-2 tags. So in the following we assume that EPCs will be managed in the same way as they were in the EPC-C1G1 standard. So the EPC is composed of the following fields (identical to those of the General Identifier, GID-96)

- *Header* is set to the fixed hexadecimal value 0x35 (8-bits).
- *General manager* identifies a company, manager or organization (28-bits).
- *Object class* is used by an EPC managing entity to identify class or "type" of thing (24-bits).
- *Serial number* is unique within each object class (36-bits).

Static identifiers (EPC-96) represent valuable information that should be transmitted on the channel guaranteeing confidentiality and, at the same time, avoiding the tracking of its holders. To solve these two connected problems, researchers have proposed a number of pseudonym-based solutions. Generally speaking, a pseudonym is a fictitious name that disguises the real EPC-96 value, allowing only authorized parties to link it to its real value. A pseudonym can be interpreted as an anonymized static identifier. However, if only pseudonyms were used, privacy could be guaranteed. To ensure adequate protection against tracking (location privacy), it is necessary to update pseudonyms each time the tag is interrogated. The most commonly used solution in the literature for pseudonym updating consists of repeatedly applying a hash function to the static identifier (i.e. $\text{pseudonym}_i = \text{hash}^i(\text{EP C})$). However, hash functions have not been ratified by the EPC-C1G2 specification because of the inherent computational limitations of low-cost RFID tags. As we saw in Section 1, tags conforming to EPC-C1G2 only support on-board a 16-bit Pseudo-Random Number Generator and a 16-bit Cyclic Redundancy Code (CRC) checksum.

In the inventory command, as described in the EPC-C1G2 specification, tags transmit their EPC as plain text. Chien et al. propose that tags transmit $M_1 = \text{CRC}(\text{EPC}||n_1||n_2)\oplus K_{in}$ instead, where the nonce n_1 is generated by the reader and the nonce n_2 is generated by the tag. Message M_1, concatenated with these two nonces n_1 and n_2 is sent to the back-end database. This scheme presents a serious security failure.

An EPC has the first 8 bits of the header fixed, while the remaining 88 bits are variable. So, there are 2^{88} possible identifiers. However, tags support onboard a 16-bit CRC (ISO/IEC 13239, $p(x) = x^{16} + x^{12} + x^5 + 1$, Preset = 0xFFFF, Residue = 0x1D0F). So the 2^{88} possible EPC values reduce to only 2^{16} possible values when the CRC is applied to the EPC ($M_1 = \text{CRC}(\text{EPC}||n_1||n_2)\oplus K_{in}$).

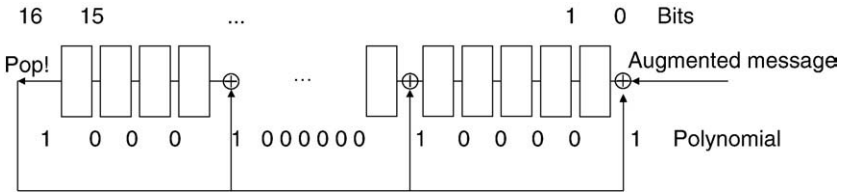

Fig. 2. CRC-CCITT implementation.

Weakness 1. Chien et al.'s protocol does not guarantee the unequivocal identification of tagged items, which is an essential property in authentication protocols.

We have simulated a population of N tags. For each tag, the values of EPC, K, P were randomly initialized. These values will be stored both in each tag and at the back-end database. Upon initialization, we simulate the reading of these N tags. For each reading, the following process is repeated:

```
(1) Reading of tag_x
(2) M_1 = CRC(EPC_x||n_1||n_2) ⊕ K_x
(3) Send M_1, n_1, n_2 to the back-end database
(4) for (x´==1, x´<N, x´++)
      M_1´ = CRC(EPC_x´ || n_1 || n_2 ) ⊕ K_x´
      if ((x´ !=x) && (M_1´==M_1)) collision++;
(5) if collision>0 "Failed unequivocal identification"
```

I.E. Failed Unequivocal Identification

EPC_x = 0xe48862a92b704993e0698583	$EPC_{x'}$ = 0xf1af12caee0319f564f89098
K_x = 0x9cf5	$K_{x'}$ = 0x2336
n_1 = 0xbdc5	n_1 = 0xbdc5
n_2 = 0xa6f4	n_2 = 0xa6f4
$M_1 = CRC(EPC_x \| n1 \| n_2) \oplus K_x$	$M_{1'} = CRC(EPC_{x'} \| n1 \| n_2) \oplus K_{x'}$
$M_1 = M_{1'} =$ 0xa2b2	

The above process is repeated T times ($T=10^4$) in order to obtain an estimation of the non-unequivocal identification probability (P_{NUI}). We have simulated the above experiment with eight different values (N=118, 226, 301, 397, 549, 626, 769, 800). The values obtained are summarized in Fig. 3.a, and fit perfectly with the values obtained for the birthday paradox with a group of N tags and $d=2^{16}$ boxes:

$$p(N;d) = \begin{cases} 1-\Pi_{k=1}^{N-1}\left(1-\frac{k}{d}\right) & N \le d \\ 1 & N > d \end{cases} \quad (8)$$

These results are hardly surprising, since each time a tag (tag_x) is read, we search if the equality $M_1 = CRC(EPC_x||n_1||n_2) \oplus K_x = CRC(EPC_{x'}||n_1||n_2) \oplus K_{x'} = M_1'$ holds for $x' \neq x$. As specified in EPC-C1G2, the CRC is 16-bits length. The kill and access passwords are 32-bits length in the specification. However, only one half (MSB or LSB) is included in each message. Chien et al. do not state the key length of the authentication key (k_x) or that of the access key (P_x). From the above, we can assume that these keys are 16-bit length. Moreover, the keys are xored with a CRC of 16-bit length, so the previous assumption seems consistent with their usage. Finally, if we assume that the CRC() and K_i are uniformly distributed (which may well not be the case, specially in the case of CRCs), the probability that at least two of N randomly selected tags have the same index ($M_1 = M_1'$), is exactly that of the birthday paradox with parameters N and $d=2^{16}$. Therefore, we have demonstrated that tags cannot be unequivocally identified under these conditions. For example, with a population of tags greater than 300, we have at least one non-unequivocal identification with a probability over 0.5 ($P_{NUI}>0.5$). Similarly, even for a relatively low number of tags (600), the probability of non-unequivocal identification rises to more than 90% (see Fig. 3.a).

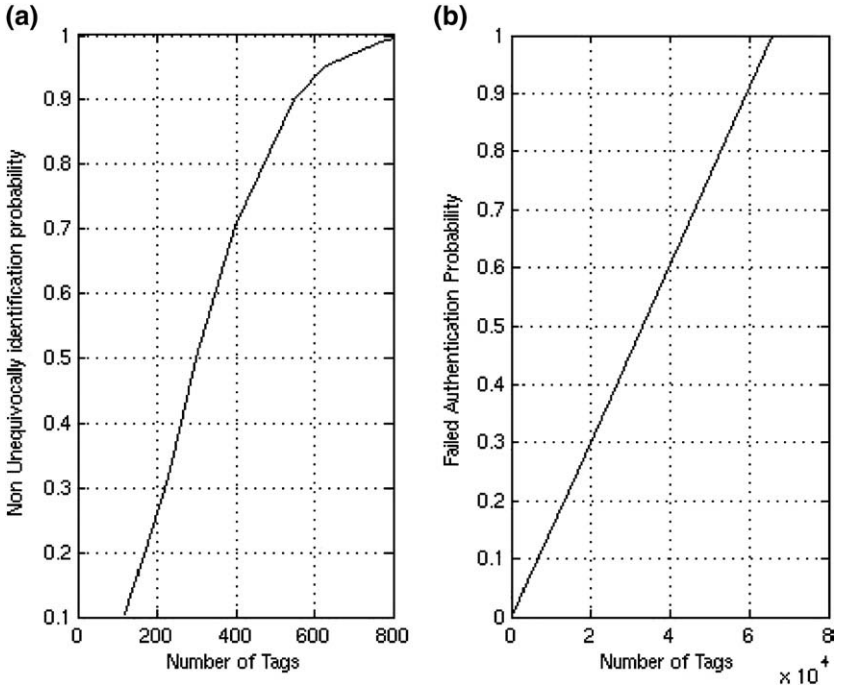

Fig. 3. Non-unequivocal identification and failed authentication.

Finally, even if we considerably relax the requirements of our RFID system even allowing the existence of collisions in the identification process (although if the collision probability is low, this seems a very unusual decision), Chien's protocol could present serious operational problems as the number of tags increases, because the probability of failed authentication rises quickly. To verify this, we carried out an experiment similar to that described above, although in this case each time a tag is read the absolute number of collisions in the database is computed. The experiment has been simulated with six different values ($N=2^7, 2^{10}, 2^{12}, 2^{14}, 2^{15}, 2^{16}$), and repeated T times ($T=10^4$). The obtained results are displayed in Fig. 3.b. So, the probability of failed identifications in a population of N tags, is described by the following equation:

$$P_{FI}(N) = \begin{cases} 2^{\log_2(N)-16} & N \leq 2^{16} \\ 1 & N > 2^{16} \end{cases} \quad (9)$$

I.E. Several Failed Identification of $Tag_{0x29a51b66cf66663d2dc9f16f}$

	MessagesTransmitted			
Reader ↔ $Tag_{0x38304699082162af23df77a1}$:	M_1	n_1	n_2	M_2
Reader ↔ $Tag_{0x9f52dc22daa678fe85d68b23}$:	M_1'	n_1'	n_2'	M_2'
Reader ↔ $Tag_{0xdb58b949c3a24a24484999a8}$:	M_1''	n_1''	n_2''	M_2''

EPC_x = 0x38304699082162af23df77a1 EPC_x''' = 0x29a51b66cf66663d2dc9f16f
K_x = 0xabd8 K_x''' = 0xa27b
n_1 = 0x1b17 n_2 = 0x2b72 n_1 = 0x1b17 n_2 = 0x2b72

$M_1 = CRC(EPC_x \| n_1 \| n_2) \oplus K_x$ $M_1''' = CRC(EPC_x' \| n_1 \| n_2) \oplus K_x'''$

$M_1 = M_1''' = 0xe5ce$

EPC_x' = 0x9f52dc22daa678fe85d68b23 EPC_x''' = 0x29a51b66cf66663d2dc9f16f
K_x' = 0x61b8 K_x''' = 0xa27b
n_1' = 0x8eac n_2' = 0xfb81 n_1' = 0x8eac n_2' = 0xfb81

$M_1' = CRC(EPC_x' \| n_1' \| n_2') \oplus K_x'$ $M_1''' = CRC(EPC_x''' \| n_1' \| n_2') \oplus K_x'''$

$M_1' = M_1''' = 0x5b41$

EPC_x'' = 0xdb58b949c3a24a24484999a8 EPC_x''' = 0x29a51b66cf66663d2dc9f16f
K_x'' = 0xc14e K_x''' = 0xa27b
n_1'' = 0x8017 n_2'' = 0xf1c3 n_1'' = 0x8017 n_2'' = 0xf1c3

$M_1'' = CRC(EPC_x'' \| n_1'' \| n_2'') \oplus K_x''$ $M_1''' = CRC(EPC_x''' \| n_1'' \| n_2'') \oplus K_x'''$

$M_1'' = M_1''' = 0x7058$

6.2. Tag impersonation and forward secrecy

Each tag shares with the reader some private information: EPC, authentication key (k_x) and the access key (P_x). This information is used to build messages M_1 and M_2 in order to prove its authenticity. However, a passive attacker eavesdropping the backward and forward channel (see [24] for a eavesdropping range classification) will be able to supplant a legitimate tag as described below:

Weakness 2. *Chien et al's protocol does not guarantee the nonimpersonation of legitimate tags.*

Proof. In order to accomplish this attack, an adversary only needs to listen to an iteration between the reader and the legitimate tag.

(1) $R \rightarrow T$: n_1
(2) $T \rightarrow R$: $M_1 = CRC(EPC_x || n_1 || n_2) \oplus K_x, n_2$

At this point, the attacker isolates the legitimate tag, preventing it from operating. He has the following information: M_1, n_1, and n_2. With this, the attacker should be able to build message $M_1' = CRC(EPC_x || n_1' || n_2')$ when queried by the reader. Although the attacker does not know the private information stored in the tag (EPC, K_x, and P_x), message M_1' can be easily computed as described below. Corollary 1 states:

$$crc(a||b) \oplus crc(a||d) = crc(b \oplus d) \quad (10)$$

As this holds for every $a, b, d \in F_2[x]$, if b and d are the concatenation of some other variables ($b = b1||b2$, $d = d1||d2$), the above expression also holds and can be rewritten as:

$$\begin{aligned} crc(a||b) \oplus crc(a||d) &= crc(a||b1||b2) \oplus crc(a||d1||d2) \\ &= crc((b1||b2)) \oplus (d1||d2) = crc(b1 \oplus d1 || b2 \oplus d2). \end{aligned} \quad (11)$$

So the difference between the known value M_1 and the new challenge M_1' is exactly $M_1 \oplus M_1' = CRC(EPC||n_1||n_2) \oplus CRC(EPC||n_1'||n_2')$ and, substituting in Eq. (11), we get

$$CRC(EPC||n_1||n_2) \oplus CRC(EPC||n_1'||n_2') = CRC(n_1 \oplus n_1' || n_2 \oplus n_2') \quad (12)$$

So, message M_1' can be obtained doing an XOR between message M_1 and the easily computable value $CRC(n_1 \oplus n_1' || n_2 \oplus n_2')$ (because all nonces are transmitted in clear and the CRC function is public). Therefore, the identity of a legitimate tag could be easily impersonated. An ANSI-C code with the implementation of this attack is available in http://163.117.149.208/rfid/chien/attack2.c.

I.E. Tag impersonation

EPC_x = 0xe34f5cdd919f4f2f9211678fe K_x = 0xb224
Reader → Tag: n_1 = 0xb3e2
Tag → Reader: M_1 = 0x21b4, n_2 = 0x5fa4
(Isolate the tag)
Reader → Attacker: n_1' = 0x77d8
Attacker → Reader: $M_1' = M_1 \oplus CRC(n_1 \oplus n_1' || n_2 \oplus n_2') = 0x21b4 \oplus 0x5e73 = 0x7fc7$
n_2' = 0xf0e2
Database: Check $M_1' = CRC(EPC||n_1'||n_2') \oplus K_x == 0xcde3 \oplus 0xb224 = 0x7fc7$
Tag is impersonated!

Additionally, the scheme does not provide forward secrecy protection. Suppose that an adversary listens to an iteration between a legitimate reader and a legitimate tag (M_1, n_1, n_2, M_2) and stores these values. Then, the tag which is not resistant to physical attacks is compromised, the EPC being obtained by the attacker. At this point, the attacker will be able to obtain the secret keys (K_x and P_x) and to generate future M_1', M_1'', etc. A detailed explanation of this attack is described below:

I.E. Forward secrecy

EPC_x = 0x4d3174f00cf844e4ce5fb064 K_x = 0x1479 P_x = 0xe04d
Reader → Tag: n_1 = 0x119b
Tag → Reader: M_1 = 0x1b36, n_2 = 0x8a4b
Reader → Tag: M_2 = 0xc57a
...

Attacker:	Tag is compromised obtaining its EPC
	M_1, M_2, n_1, n_2, EPC_x are known
	Obtaining the keys:
	$K_x = CRC(EPC||n_1||n_2) \oplus M_1 = 0x1479$
	$P_x = CRC(EPC||n_2) \oplus M_2 = 0xe04d$
	$K_x' = PRNG(K_x) = 0xa586$
	$M_1' = CRC(EPC||n_1'||n_2') \oplus K_x'$
	The attacker is able to generate future M_1 messages
	i.e. n_1' = 0xfa4b n_2' = 0x4b88
	$M_1' = CRC(EPC||n_1'||n_2') \oplus K_x' =$
	$= 0x3c64 \oplus 0xa586 = 0x99e2$

6.3. Back-end database impersonation

In Section 6.2 we focused on message M_1 sent by the tag when queried by the reader. In this case we concentrate on message M_2, generated by the back-end database. The attacker should be able to generate this message in order to impersonate a legitimate back-end database.

Weakness 3. *Chien et al.'s protocol is vulnerable to back-end database impersonation.*

Proof. For the attacker, it is enough to listen to an iteration between a legitimate tag and a reader-database in order to exploit this vulnerability:

(1) $R \rightarrow T$: n_1
(2) $T \rightarrow R$: $M_1 = \mathrm{CRC}(\mathrm{EPC}_x||n_1||n_2) \oplus K_x$, n_2
(3) $R \rightarrow Database$: M_1, n_1, n_2
(4) $Database \rightarrow R$: $M_2 = \mathrm{CRC}(\mathrm{EPC}_x||n_2) \oplus P_x$
(5) $R \rightarrow T$: M_2

The attacker has to block or disrupt the radio channel to obstruct the correct reception of message 5. The objective of this is to prevent the legitimate tag from updating its key. At this point, the attacker could supplant the back-end database without knowing all its private information (the six fields described in Section 6.1). In the next tag reading, the database will receive M_1', n_1', n_2'. The fraudulent database has to compute the message M_2'. But from Corollary 1, the following expression can be derived:

$$M_2 \oplus M_2' - \mathrm{CRC}(\mathrm{EPC}||n_2) \oplus \mathrm{CRC}(\mathrm{EPC}||n_2')\,\mathrm{CRC}(\mathrm{EPC}||n_2) \oplus \mathrm{CRC}(\mathrm{EPC}||n_2') = \mathrm{CRC}(n_2 \oplus n_2') \tag{13}$$

So, message M_2' can be obtained by means of an XOR between the previous M_2 message listened to in the air channel and the easily computed value $\mathrm{CRC}(n_2 \oplus n_2')$. Message M_2' will be sent to the tag, which will authenticate the fraudulent back-end database and update its keys. An ANSI-C code with the implementation of this attack is available in http://163.117.149.208/rfid/chien/attack3.c.

I.E. Database impersonation

EPC_x = 0x52c3e4175b97de07f22f9db0 K_x = 0xf6dd P_x = 0xca39
Reader → Tag: n_1 = 0x04a6
Tag → Reader: M_1 = 0x7a98, n_2 = 0xa833
Reader → Tag: M_2 = 0x25f6 (blocked!)
...
Attacker → Tag: n_1' = 0xf556
Tag → Attacker: M_1' = 0x47dc, n_2' = 0xae5c
Attacker → Tag: $M_2' = M_2 \oplus \mathrm{CRC}(n_2 \oplus n_2')$ = 0x1219
Tag: Check $M_2' = \mathrm{CRC}(\mathrm{EPC}||n_2') \oplus P_x$ =
= 0xd820 ⊕ 0xca39 = 0x 1219
Back-end database is impersonated!

6.4. Tracking or private location

Protection against tracking is not guaranteed when tags answer reader queries with the same identifier. In Chien et al.'s protocol, nonces n_1 and n_2 are employed in each session to ensure freshness. With this, it seems that the tag's private location is assured. This is not the case, as explained below:

Weakness 4. Chien et al.'s protocol does not guarantee the location privacy of tags.

Proof. The success of this attack depends on preventing tag key updating. Moreover, if the population of tags is greater than 2^{16}, the normal operation of the protocol will hamper the key update operation (see Section 6.1). In the back-end database a pair of keys (new, old) are stored for each tag key. Chien et al. claim that the storage of these two keys frustrates DoS attacks. To provide this property, the fact that a tag may sometimes use the same key (message M_2 was incorrectly received) to compute the message M_1 is considered as a normal operation. In fact, Chien does not specify the maximum number of times that a tag can be authenticated with the same authentication key. Imagine that the reader captures two non-consecutive iterations, upon non-updating key condition (an ANSI-C code with the implementation of this attack is available in http://163.117.149.208/chien/rfid/attack4.c):

(1) $R \rightarrow T$: n_1
(2) $T \rightarrow R$: $M_1 = \mathrm{CRC}(\mathrm{EPC}_x||n_1||n_2) \oplus K_x^n$, n_2
....
(1) $R \rightarrow T$: n_1'
(2) $T \rightarrow R$: $M_1' = \mathrm{CRC}(\mathrm{EPC}_x||n_1'||n_2') \oplus K_x^n$, n_2'

Now, the attacker computes the XOR of messages M_1 and M_1v. If messages M_1 and M_1' came from the same tag, the key K_x^n is cancelled when the XOR is computed. By means of Eq. (11), the attacker can verify if answers arise from the same tag:

$$M_1 \oplus M_1' = \mathrm{CRC}(n_1 \oplus n_1'||n_2 \oplus n_2') \tag{14}$$

I.E. Private location jeopardized

EPC_x = 0x26d4c59d93afbeaf871fb35c K_x = 0x650b
Upon non-updating key condition ...
Reader → Tag: n_1 = 0x1305
Tag → Reader: M_1 = 0x6b3c, n_2 = 0xb642
...
Reader → Tag: n_1' = 0x1ea4
Tag → Reader: M_1' = 0x1e33, n_2' = 0xf4a7
...
Attacker: A = $M_1 \oplus M_1'$ = 0x750f
B = CRC ($n_1 \oplus n_1'||n_2 \oplus n_2'$) = 0x750f
A==B ⇒ Tag answers provide for the same tag!

A similar attack can be accomplished using messages sent by the database (M_2). Suppose that a crooked reader interrogates a tag: reader sends the message M_1 to the database. The database authenticates the tag, and sends back message M_2. M_2 is stored by the reader, and a wrong M_2 message is sent to the tag avoiding its key updating. Then the above process is repeated, obtaining messages M_1', and M_2' with nonces n_1' and n_2'. At this point, the attacker computes the XOR of messages M_2 and M_2' to check if they came from the same tag. From Corollary 1, the following equality has to be fulfilled:

$$M_2 \oplus M_2' = \mathrm{CRC}(n_2 \oplus n_2') \tag{15}$$

6.5. Back-end database auto-desynchronization

To defend against a DoS attack, Chien et al. propose that the back-end database maintains a pair of keys (*new*, *old*) for each tag key. This assumption allows the server to authenticate tags and re-synchronize these each time they suffer a DoS attack. However, the normal operation of the protocol results in synchronization loss between the database and the tags due to the non-unequivocal identification property shown in Section 6.1.

Weakness 5. Chien et al.'s protocol is vulnerable to auto-desynchronization attacks.

We have simulated a population of N tags. For each tag the EPC, K, and P values are randomly initialized. These values will be stored both in the tag and in the back-end database. Upon initialization, we simulate the reading of N tags. For each reading, the following process is repeated:

(1) Reading of tag_x
(2) $M = \mathrm{CRC}(\mathrm{EPC}_x||n_1||n_2) \oplus K_x$
(3) Send M, n_1, n_2 to the back-end database
(4) while $((x' < N)\&\&(\mathrm{output}==0))$
$\{M' = \mathrm{CRC}(\mathrm{EPC}_x'||n_1||n_2) \oplus K_x'$
if $(M'==M)$ autodesyn[x']++; output = 1
x′++}

Upon the reading of the N tags, we compute the number of times that auto-desynchronization occurred. After the reading of a tag, its keys are updated both in the database and in the tag. An additionally wrong update, during the reading of a different tag, will cause a loss of synchronization for that tag. Therefore, the number of tags whose

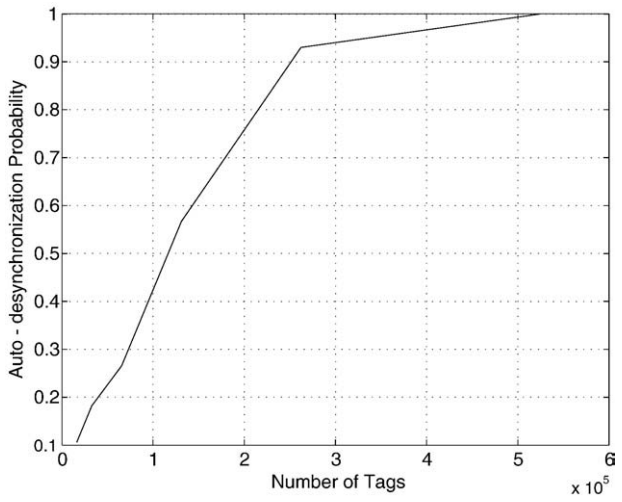

Fig. 4. Auto-desynchronization probability.

keys have been updated two or more times constitute the number of de-synchronized tags. So the probability of auto-desynchronization (P_{ADS}) can be defined as:

$$P_{ADS} = \frac{1}{N}\sum_{x=1}^{N} (\text{autodesyn}[x]-1). \quad (16)$$

The above process is repeated T times ($T=10^3$) in order to obtain an estimation of the P_{ADS}. We have simulated the experiment with different values ($N=2^{14}$, 2^{15}, 2^{16} … 2^{18}), as displayed in Fig. 4. The results indicate that if we have a population of $N\geq 2^{17}$ tags, the probability of auto-desynchronization is greater than 0.5. This probability increases to 0.93 if the population is $N\geq 2^{18}$.

7. Conclusions

Due to the security faults both of EPC-C1G2 and of the previous proposals conforming to this standard, in 2007 Chien et al. proposed a new mutual authentication protocol that tried to solve these problems. After briefly presenting the Chien scheme, the security of his protocol was analyzed, showing some important security failures: non-unequivocal-identification, identity impersonation (both of tags and, importantly, the back-end database), non-forward security, tracking, and auto-desynchronization.

We have also shown that all these security weaknesses are related to the use of the CRC. Some of them (non-unequivocal identification and auto-desynchronization) could have been solved simply by using larger CRCs (well above the 16-bit CRC proposed in the standard). The rest of the security problems highlighted in this article are due to the bad (linear) properties of CRCs and will not be solved by changing the CRC length. We must conclude that these results should be taken into account in future proposals. Furthermore, we doubt that CRCs should be used in any security protocol at all, and its use should be confined to guarantee an error-free communication channel. A cryptographic function such as a lightweight hash-function (i.e. PHF [19], Tav-128 [20]), or some kind of MAC (i.e. Squash [26]) should be used instead.

References

[1] Philips and Texas Instruments join forces to accelerate EPC Gen-2 RFID deployment, 2005 http://www.nxp.com/news/content/file1171.html.
[2] Anarchriz, CRC and how to reverse it, 1999 http://www.woodmann.com/fravia/crctut1.htm.
[3] D. Bailey, A. Juels, Shoehorning security into the EPC standard, 2006 Manuscript in submission.
[4] S. Bono, M. Greem, A. Stubblefield, A. Juels, A. Rubin, M. Syzdlo, Security analysis of a cryptographically-enabled device, Proc. of SSYM'05, Usenix Association, 2005.
[5] Hung-Yu Chien, Che-Hao Chen, Mutual authentication protocol for RFID conforming to EPC class-1 generation-2 standards, Computer Standards & Interfaces, vol. 29, Elsevier Science Publishers, February 2007, pp. 254–259, (2).
[6] E.Y. Choi, S.M. Lee, D.H. Lee, Efficient RFID authentication protocol for ubiquitous computing environment, Proc. of SECUBIQ'05, LNCS, 2005.
[7] T. Dimitriou, A lightweight RFID protocol to protect against traceability and cloning attacks, In Proc. of SECURECOMM'05, 2005.
[8] Class-1 Generation-2 UHF air interface protocol standard version 1.0.9: "Gen 2", January 2005 http://www.epcglobalinc.org/.
[9] EPC Generation-1 tag data standards version 1.1, May 2005 http://www.epcglobalinc.org/.
[10] J. Ha, S.J. Moon, J.M. Gonzlez Nieto, C. Boyd, Low-cost and strong- security RFID authentication protocol, Proc. of EUC Workshops, LNCS, vol. 4809, Springer, 2007, pp. 795–807.
[11] D. Henrici, P. Müller, Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers, Proc. of PERSEC'04, IEEE Computer Society, 2004, pp. 149–153.
[12] A. Juels, RFID security and privacy: a research survey. Manuscript, September 2005.
[13] A. Juels, Strengthening EPC tags against cloning. Manuscript, March 2005.
[14] N. Karten and H. Pltz. Mifare little security, despite obscurity. http://events.ccc.de/congress/2007/Fahrplan/events/2378.en.html.
[15] S. Karthikeyan, M. Nesterenko, RFID security without extensive cryptography, Proc. of SASN '05, ACM, 2005, pp. 63–67.
[16] D.M. Konidala, K. Kim, RFID tag-reader mutual authentication scheme utilizing tag's access, Auto-ID Labs White Paper WP-HARDWARE-033, 2007.
[17] T.L. Lim, T. Li, Addressing the weakness in a lightweight RFID tag-reader mutual authentication scheme, Proc. of GLOBECOM'07, 2007.
[18] 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.
[19] K. Nohl, D. Evans, Feasible privacy for lightweight RFID systems, 2007 http://www.cs.virginia.edu/evans/talks/spar07.
[20] P. Peris-Lopez, J.C. Hernandez-Castro, J.M. Estevez-Tapiador, A. Ribagorda, An efficient authentication protocol in RFID systems resistant to active attacks, Proc. of SecUbiq'06, LNCS, vol. 4809, Springer-Verlag, 2007, pp. 781–794.
[21] P. Peris-Lopez, J.C. Hernandez-Castro, J.M. Estevez-Tapiador, A. Rib- agorda, RFID systems: A survey on security threats and proposed solutions, Proc. of PWC06, LNCS, vol. 4217, 2006, pp. 159–170.
[22] S. Piramuthu, Protocols for RFID tag/reader authentication, Decis. support syst. 43 (3) (2007) 897–914.
[23] D.C. Ranasinghe, Networked RFID Systems and Lightweight Cryptography, chapter Lightweight Cryptography for Low Cost RFID, Springer, 2007, pp. 311–346.
[24] D.C. Ranasinghe, P.H. Cole, Confronting security and privacy threats in modern RFID systems, Proc. of ACSSC '06, 2006, pp. 2058–2064.
[25] S. Sarma, S. Weis, D. Engels, RFID systems and security and privacy implications, Proc. of CHES'02, LNCS, vol. 2523, 2002, pp. 454–470.
[26] A. Shamir. SQUASH — a new MAC with provable security properties for highly constrained devices such as RFID tags. In Proc. of FSE'08, volume In Press of LNCS. Springer-Verlag, in press.
[27] M. Stigge, H. Pltz, W. Mller, J.-P. Redlich, Reversing CRC theory and practice, Technical Report SAR-PR-2006-05, Humboldt-Universitat, Berlin, 2006.
[28] A.S. Tanenbaum, Computer Networks, 3rd edition, Prentice-Hall International, Inc, 1996.
[29] S. Weis, S. Sarma, R. Rivest, D. Engels, Security and privacy aspects of low-cost radio frequency identification systems, Proc. of SPC'03, LNCS, vol. 2802, 2003, pp. 454–469.
[30] B. Westerbaan, Reversing CRC, 2005 http://blog.w-nz.com/archives/2005/07/15/reversing-crc/.
[31] J.Yang, J.Park, H.Lee, K.Ren, K.Kim, Mutual authentication protocol for low-cost RFID, Ecrypt Workshop on RFID and Lightweight Crypto, 2005.

Pedro Peris-Lopez is 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, cryptanalysis, 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 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 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 regular referee for various journals.

Arturo Ribagorda is 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.