Engineering Applications of Artificial Intelligence 24 (2011) 1061–1069

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

Cryptanalysis of an EPC Class-1 Generation-2 standard compliant authentication protocol

Pedro Peris-Lopez [a,*], Julio C. Hernandez-Castro [b], Juan M.E. Tapiador [c], Jan C.A. van der Lubbe [a]

[a] Delft University of Technology (TU-Delft), Faculty of Electrical Engineering, Mathematics, and Computer Science (EEMCS), Information Security and Privacy Lab, P.O. Box 5031 2600 GA, Delft, The Netherlands

[b] School of Computing, Buckingham Building, Lion Terrace, Portsmouth PO1 3HE, United Kingdom

[c] Department of Computer Science, University of York, Heslington, York, YO10 5DD, United Kingdom

ARTICLE INFO

Article history:
Received 16 April 2009
Received in revised form
24 January 2011
Accepted 3 April 2011
Available online 19 May 2011

Keywords:
RFID
EPC
Security
Authentication
Cryptanalysis

ABSTRACT

Recently, Chen and Deng (2009) proposed an interesting new mutual authentication protocol. Their scheme is based on a cyclic redundancy code (CRC) and a pseudo-random number generator in accordance with the EPC Class-1 Generation-2 specification. The authors claimed that the proposed protocol is secure against all classical attacks against RFID systems, and that it has better security and performance than its predecessors. However, in this paper we show that the protocol fails short of its security objectives, and in fact offers the same security level than the EPC standard it tried to correct. An attacker, following our suggested approach, will be able to impersonate readers and tags. Untraceability is also not guaranteed, since it is easy to link a tag to its future broadcast responses with a very high probability. Furthermore, readers are vulnerable to denial of service attacks (DoS), by obtaining an incorrect EPC identifier after a successful authentication of the tag. Moreover, from the implementation point of view, the length of the variables is not compatible with those proposed in the standard, thus further discouraging the wide deployment of the analyzed protocol. Finally, we propose a new EPC-friendly protocol, named `Azumi`, which may be considered a significant step toward the security of Gen-2 compliant tags.

1. Introduction

A Radio Frequency Identification (RFID) system is a set of automated identification technologies in which a small transponder (tag), attached to an object (i.e. a human, animal or product), receives and responds to radio-frequency queries from a transceiver (reader). Barcodes are currently the most extended identification systems, but this technology may be replaced by RFID in the near future. RFID has two main advantages over barcodes. First, the technological advantages: 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 at a distance of several meters. Secondly, the unequivocal identification provided by RFID technology: an RFID tag assigns a unique identifier to each tagged item, while a barcode only specifies the type of the labeled product. Despite these benefits, security and privacy concerns are holding up the rapid and widespread adoption of this promising technology.

Due to the heterogeneity of RFID systems, there is a great number of interconnected standards. ISO and EPCglobal have played an important role in harmonization. In 2004, the Electronic Product Code Class-1 Generation-2 specification (EPC-C1G2 in short) was adopted by EPCGlobal (Class-1 generation 2 UHF, 2008). A few months later, it was ratified by ISO and published as an amendment to its ISO/IEC 18000-6 (Information technology—Radio, 2005). This standard is an important milestone for the standardization of low-cost RFID tags. However, the different security analysis carried out on the EPC-C1G2 specification have revealed important security flaws (Bailey and Juels, 2006; Peris-Lopez et al., 2008). Some researchers have later proposed EPC-friendly schemes trying to correct these weaknesses. One of the most recent proposals following this approach is Chen and Deng's scheme, which is our concern in this paper.

The rest of the paper is organized as follows. The related work is reviewed in Section 2. In Section 3, we present Chen and Deng's protocol. The properties of CRC functions are studied in Section 4. We present our attacks and the non-conformity with the standard in Section 5. In Section 6, we introduce a new protocol compliant with EPC-C1G2 standard and highly inspired by ISO/IEC 9798. Finally, we extract some conclusions in Section 7.

2. Related work

Motivated by the low security level of EPC-C1G2 specification, some recent proposals have attempted to rectify its deficiencies

* Corresponding author.
E-mail addresses: P.PerisLopez@tudelft.nl (P. Peris-Lopez), Julio.Hernandez-Castro@port.ac.uk (J.C. Hernandez-Castro), jet@cs.york.ac.uk (J.M.E. Tapiador), J.C.A.vanderLubbe@tudelft.nl (J.C.A. van der Lubbe).

0952-1976/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.engappai.2011.04.001

whilst still conforming to the standard. We now summarizes the most important proposals in this area.

Bailey and Juels (2006) examined various ways for RFID tags to perform cryptographic functions while remaining EPC-C1G2 compliant. Their main idea is to take an expansive view of EPC tag memory. Instead of considering memory merely as a storage medium, they use it as an input/output way of interfacing with a cryptographic module within the tag. Read/write commands may, therefore, involve 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 modules is unrealistic, at least at present time.

Karthikeyan and Nesterenko (2005) proposed an efficient tag identification and reader authentication protocol based on a simple XOR and matrix operations. Two matrices 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 messages Y, Z. The first is used to authenticate the tag and the second to update the key. However, an attacker can substitute the original Z by a random Z'. Upon receiving Y, Z', the tag will be authenticated and will wrongly update the key. So the legitimate reader and the tag will not be able to authenticate each other any more. Additionally, the protocol is vulnerable to replay attacks, and privacy location is not guaranteed (Chien and Chen, 2007).

Nguyen Duc et al. (2006) later proposed a tag-to-back-end database authentication protocol. The security of Nguyen Duc et al.'s protocol is based on key synchronization between tags and the back-end database. The last message of the protocol consists 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. The tag and the reader will then be no longer able to authenticate, which is an extremely serious problem. The protocol also presents backward secrecy problems, as compromise of the EPC allows an attacker to trace back all previous communications.

Chien et al. pointed out certain weaknesses in the schemes (Karthikeyan and Nesterenko, 2005) and (Nguyen Duc et al., 2006), and then proposed a new EPC-C1G2 compliant mutual authentication protocol (Chien and Chen, 2007). However, Peris-Lopez et al. (2009) showed how none of the objectives are met, as it is vulnerable to attacks including identity impersonation, non-forward security and tracking. Execution of the protocol itself even produces desynchronization between the tags and the back-end database.

Konidala and Kim (2007) 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 using a simple XOR operation, in contrast to what happens in the specification. However, Lim and Li (2007) showed how a passive attacker can recover the tag's password by eavesdropping over a single run of the protocol and performing correlation analysis on the captured information. Konidala and Kim then proposed a new version of the TRMA scheme (TRMA+) in which the tag access and kill passwords are used for authentication. This new version still contains important security flaws, as the key and access password can be acquired by an adversary, with non-negligible probability (Peris-Lopez et al., 2008).

We summarize here some of the most relevant works in which artificial intelligence (AI) techniques were successfully used to solve cryptographic problems. We urge the interesting reader to consult the original papers for details. Spillman et al. (1993) showed that simple substitution ciphers and knapsack ciphers (Spillman, 1993) could be solved using genetic algorithms. Giddy and Safavi-Naini (1994) used simulated annealing for the automatic cryptanalysis of transposition ciphers. Clark (1994) summarized the use of simulated annealing, genetic algorithms and tabu search in the cryptanalysis of simple substitution and transposition ciphers. Knudsen and Meier (1999) introduced an improved simulated annealing search to attack an identification scheme based on the permuted perceptron problem (PPP). Hernandez Castro and Isasi Viuela (2005) introduced a technique capable of breaking reduced-round versions of the block cipher TEA and XTEA with the aid of genetic algorithms. A particle swarm optimization method and a genetic algorithm were able to cryptanalyze a four-round version of DES in Laskari et al. (2005) and Song et al. (2007), respectively. More recently, and in the context of RFID, a quite efficient simulated annealing-based traceability attack against an ultralightweight authentication protocol was presented in Hernandez-Castro et al. (2009).

3. Chen and Deng's protocol

Chen and Deng (2009) proposed an EPC-friendly scheme (CD-EPC in short) based on the use of a pseudo-random number generator (PRNG) and a cyclic redundancy code (CRC), as recommended by the EPC-C1G2 standard. We will outline the CD-EPC protocol in brief, which consists of two phases: a registration and an initialization. The following notation is used thorough the paper:

$x_{\mathcal{T}_i}(j)$	value x of tag $\mathcal{T}_i$ registered in jth database
$(N_{\mathcal{T}_i}(j),K_{\mathcal{T}_i}(j))$	$N_{\mathcal{T}_i}(j)$ is nonce word and $K_{\mathcal{T}_i}(j)$ is a key of tag $\mathcal{T}_i$
CRC()	a Cyclic Redundancy Code (CRC) function
$EPC_{\mathcal{T}_i}$	*EPC* identification number of tag $\mathcal{T}_i$
ID_{R_i}	identification number of the ith reader
RND	random number
$\oplus$	exclusive-OR operation
M_{req}	reader's request message
M_{resp}	reader's response message

3.1. Registration phase

Tags and readers must register in the database separately in a secure environment. Tags send their unique *EPCs* to the database. The database then responds with $N_{\mathcal{T}_i}(j)$ and $K_{\mathcal{T}_i}(j)$ to each tag that ask for registering. Each $N_{\mathcal{T}_i}(j)$ corresponds to only one $K_{\mathcal{T}_i}(j)$. Generally, each tag may register in several databases obtaining a set of $\{N_{\mathcal{T}_i}(j),K_{\mathcal{T}_i}(j)\}_{j=1}^{n}$. Readers are registered in the database under their unique identification ID_{R_i}. After registration, the database responds with the tuples $(N_{\mathcal{T}_i}(j),K_{\mathcal{T}_i}(j))$ of all the assigned tags that can be accessed by reader ID_{R_i}. The authors also believe that readers may be registered in several databases at the same time. A common scenario depicting tags and readers registration is illustrated in Fig. 1.

3.2. Communication phase

Through the registration phase, tags and readers can communicate mutually. Only random numbers, exclusive-OR operations and the lightweight CRC operation are employed to compose the exchanged messages, as required by the EPC-C1G2 standard. The proposed scheme is split into five steps (see Fig. 2):

Step 1: When the reader wants to access a tag, it sends a request message M_{req}, $CRC(N_{\mathcal{T}_i}(j)\oplus RND_1)$ and RND_1 to the tag.

Step 2: Upon receiving $CRC(N_{\mathcal{T}_i}(j)\oplus RND_1)$ and RND_1, the tag uses the stored $N'_{\mathcal{T}_i}(j)$ to compute $CRC(N'_{\mathcal{T}_i}(j)\oplus RND_1)$. Thus, the tag can authenticate the reader via the following verification:

$$CRC(N'_{\mathcal{T}_i}(j)\oplus RND_1)\stackrel{?}{=}CRC(N_{\mathcal{T}_i}(j)\oplus RND_1) \quad (1)$$

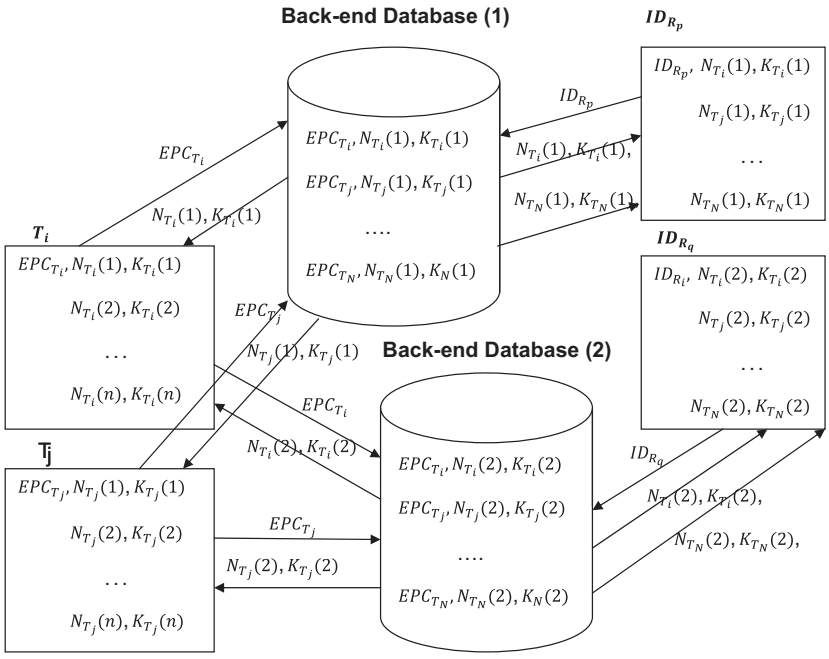

Fig. 1. CD-EPC initialization: tags & readers.

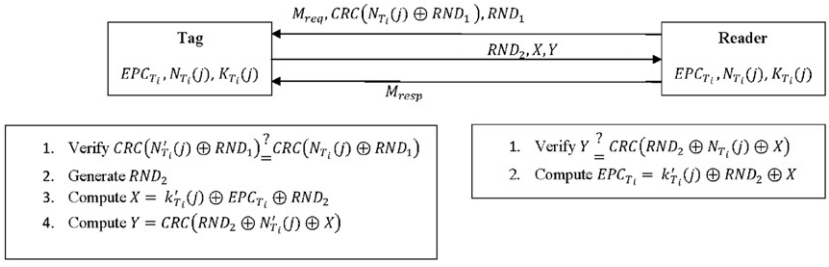

Fig. 2. CD-EPC mutual authentication protocol.

If this equality does not hold, the tag will not perform any further calculations or responses; the request is assumed to be sent from an attacker or from a forbidden list. If it holds, a tag will generate a new random number RND_2 and compute:

$$X = K'_{T_i}(j) \oplus EPC_{T_i} \oplus RND_2 \tag{2}$$

$$Y = CRC(RND_2 \oplus N'_{T_i}(j) \oplus X) \tag{3}$$

Step 3: Tag sends {RND_2,X,Y} to the reader.

Step 4: Upon receiving the tag's response message, the reader computes its local version of $Y = CRC(RND_2 \oplus N_{T_i}(j) \oplus X)$, using RND_2 and X as obtained from the tag. If the equality does not hold, the response may have been sent by an attacker, and the reader will not perform any further calculations or responses. If it holds, the reader uses $K_{T_i}(j)$ (linked to $N_{T_i}(j)$), and (RND_2,X) to obtain the static identifier of tag $\mathcal{T}_i$:

$$EPC_{T_i} = K_i \oplus RND_2 \oplus X \tag{4}$$

Step 5: When a reader obtains a tag's EPC_{T_i} and the authenticity of the tag has been confirmed, the reader sends a response M_{resp} to the tag.

4. 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 linear, so they should not be used in cryptographic or security related applications as they cannot detect malicious changes by a knowledgeable attacker (Anarchriz, 1999; Ranasinghe, 2007; Stigge et al., 2006; Westerbaan, 2005). To illustrate this property, the hamming distance (*HD*) can be used. The *HD* of a CRC polynomial is the minimum number of error bits that can pass undetected by the CRC. For example, if a CRC has a *HD* of 3, any combinations of 1 or 2 error bits will be detected, but there is at least one combination of three error bits that will pass unnoticed. Cryptographic hash functions, that have very high HD values, should, therefore, be used for any security related purpose instead.

4.1. Definitions and notations

Let A be an m-bit string in $\{0,1\}^m$, $A = A_{m-1}||A_{m-2}\| \ldots \|A_0$. We define $A_{\ll n}$ as the $m+n$ bit string A' resulting from left-shift of A by n-bits:

$$A'_i = \begin{cases} 0 & \text{for } 0 \le i \le n-1 \\ A_{i-n} & \text{for } n \le i \le n+m-1 \end{cases} \tag{5}$$

Let B be an n-bit string in $\{0,1\}^n$, where $n \leq m$. We define the exclusive-OR operation $A \oplus B = B \oplus A$ as follows:

$$(A \oplus B)_i = \begin{cases} A_i \oplus B_i & \text{for } 0 \leq i \leq n-1 \\ A_i & \text{for } n \leq i \leq m-1 \end{cases} \quad (6)$$

The set of bit strings $\{0,1\}^\infty$ forms a group under the exclusive-OR operation. F_2 and $F_2[x]$ symbolize the binary field and the ring of polynomials over F_2, respectively. For a m-bit string A, we define a map $\phi : \{0,1\}^\infty \rightarrow F_2[x]$:

$$\phi(A) = \sum_{i=0}^{m-1} A_i x^i \quad (7)$$

As ϕ is an group isomorphic, the inverse exists (ψ). That is,

$$\psi\left(\sum_{i=0}^{m-1} A_i x^i\right) = A_{m-1} \| A_{m-2} \| \ldots \| A_0 \quad (8)$$

4.2. CRC properties

CRC functions are based on polynomial arithmetic in F_2. Computing a crc value for a given binary stream is essentially performed by dividing the polynomial associated with this stream by another fixed polynomial (generator polynomial) and obtaining a remainder. Let G be the generator polynomial used for calculating CRC. The CRC for any bit-string A is then computed by

$$CRC(A) = \psi(\phi(A) \bmod G) \quad (9)$$

Due to the linearity, CRCs have the following properties (Han and Kwon, 2009; Peris-Lopez et al., 2009):

Theorem 1. *For any CRC (independent of its generator polynomial), and for any A n-bit and B m-bit string, it holds that*

$$CRC(A \oplus B) = CRC(A) \oplus CRC(B) \quad (10)$$

$$CRC(A \| B) = CRC(A_{\ll n}) \oplus CRC(B) \quad (11)$$

Proof. From the definition in Eq. (9), one can write

$$CRC(A \oplus B) = \psi(\phi(A \oplus B) \bmod G) \quad (12)$$

Since modular operations, ψ, and ϕ are homomorphic, the above equation can be rewritten

$$\begin{aligned} \psi(\phi(A \oplus B) \bmod G) &= \psi((\phi(A) \bmod G) \oplus (\phi(B) \bmod G)) \\ &= \psi(\phi(A) \bmod G) \oplus \psi(\phi(B) \bmod G) \\ &= CRC(A) \oplus CRC(B) \end{aligned} \quad (13)$$

The concatenation of any two bit strings ($A \| B$), can be viewed as the exclusive-OR between the n-bit shift of the left variable ($A_{\ll n}$) and the right value (B). Thus, applying Eq. (13)

$$CRC(A||B) = CRC(A_{\ll n} \oplus B) = CRC(A_{\ll n}) \oplus CRC(B) \quad \square \quad (14)$$

5. Vulnerabilities of Chen and Deng's protocol

In this section we analyze the most relevant weaknesses of the CD-EPC protocol.

5.1. Reader impersonation

Each tag shares some private information with the reader: $N_{T_i}(j)$ and $K_{T_i}(j)$. This information is used to build the exchanged messages between these two devices in order to proof their authenticity. Specifically, the reader is authenticated by checking the following equation: $CRC(N'_{T_i}(j) \oplus RND_1) \stackrel{?}{=} CRC(N_{T_i}(j) \oplus RND_1)$.

Theorem 2. *In the CD-EPC protocol, a passive attacker is able to supplant a legitimate reader after eavesdropping one authentication session by sending message* M_{req}, $CRC(N_{T_i}(j) \oplus RND_1) \oplus CRC(\Delta)$, RND'_1, *where* $\Delta = RND_1 \oplus RND'_1$.

Proof. *Step*1: The attacker eavesdrops an authentication session between the reader and the tag.

(1) $R \rightarrow T : M_{req}, CRC(N_{T_i}(j) \oplus RND_1), RND_1$
(2) $T \rightarrow R : RND_2, X, Y$
(3) $R \rightarrow T : M_{resp}$,

*Step*2: The attacker can supplant the reader by sending the following message:

(1) $A \rightarrow T : M_{req}, CRC(N_{T_i}(j) \oplus RND_1) \oplus CRC(\Delta), RND'_1$
where $\Delta = RND_1 \oplus RND'_1$
(2) ...

On receiving $CRC(N_{T_i}(j) \oplus RND_1) \oplus CRC(\Delta)$ and $RND1'$, the tag uses $N'_{T_i}(j)$ to compute its local value and compare this with the received value.

$$CRC(N_{T_i}(j) \oplus RND_1) \oplus CRC(\Delta) \stackrel{?}{=} CRC(N'_{T_i}(j) \oplus RND'_1) \quad (15)$$

From Eq. (10), it holds that

$$\begin{aligned} CRC(N_{T_i}(j) \oplus RND_1) \oplus CRC(\Delta) &= CRC(N'_{T_i}(j) \oplus RND_1 \oplus \Delta) \\ &= CRC(N'_{T_i}(j) \oplus RND_1 \oplus RND_1 \oplus RND'_1) \\ &= CRC(N_{T_i}(j) \oplus RND'_1) \quad \square \end{aligned} \quad (16)$$

The message sent by the attacker is, therefore, accepted as valid. The attack described is quite harmful because once an authentication session is eavesdropped, the attacker is able to supplant the reader indefinitely. This issue could be mitigated by refreshing internal values ($N_{T_i}(j)$,$K_{T_i}(j)$), but the authors curiously did not choose to do this in their protocol design.

5.2. Tracking or private location

Untraceability can be viewed as a game $\mathcal{G}$ played between an adversary ($\mathcal{A}$) and a collection of readers ($\mathcal{R}_i$) and tags instances ($\mathcal{T}_i$). The success of $\mathcal{A}$ winning $\mathcal{G}$, therefore, translates to its success in breaking untraceability (Phan, 2009):

$$Adv_{\mathcal{A}}^{(UNT)} = |Pr(\mathcal{A}wins) - \tfrac{1}{2}| \quad (17)$$

where k is a security parameter (i.e. bit length of secret values). An RFID protocol achieves untraceability if $Adv_{\mathcal{A}}^{(UNT)} < \varepsilon(k)$, $\varepsilon(.)$ being some negligible function.

Theorem 3. *The CD-EPC protocol does not protect against privacy location*; *an adversary wins the untraceability* ($\mathcal{G}$) *gain with a significant probability*: $Adv_{\mathcal{A}}^{(UNT)} \simeq 0.499985$.

Two different approaches can be used by the adversary. First, the reader impersonation attack presented in the Section 5.1 may be employed (Proof-A). A passive attacker can trace any given RFID after observing one single authentication session because the secret $N_{T_i}(i)$ is kept constant and the *CRC* function does not disguise it well enough. Secondly, the attacker may focus on the responses provided by tags (Proof-B). In this case, we take advantage of the authors' misuse of bitwise operations which result in the inclusion of a constant value in tag responses.

Proof-A. Specifically, the adversary $\mathcal{A}$ performs the following steps:

Learning: Eavesdrops on an authentication session between the reader and the tag $\mathcal{T}_0$.

Challenge: Some time later, the adversary chooses two fresh tags $\mathcal{T}_0$ and $\mathcal{T}_1$. Then, one of these tags is randomly selected ($\mathcal{T}_i \in \{\mathcal{T}_0,\mathcal{T}_1\}$) and presented to the adversary. The adversary tries to impersonate it by the procedure described in Section 5.1.

Guessing: If an answer message is obtained, $\mathcal{A}$ conjectures it is tag $\mathcal{T}_0$. Otherwise, it supposes it to be $\mathcal{T}_1$. There is only negligible probability that by chance the procedure described in Section 5.1 will work for tag $\mathcal{T}_1$. As EPC-compliant tags support 16-bit CRC function on-board (Class-1 generation 2 UHF, 2008), and assuming independence and uniformity in the random number generation and in the CRC output, this probability has a value of 2^{-16}.

So, the $Adv_{\mathcal{A}}^{(UNT)}(k)$ is non-negligible and the proposed protocol does not achieve untraceability:

$$Adv_{\mathcal{A}}^{(UNT)} \simeq \left|\frac{1}{2}-\frac{1}{2^{16}}\right| \qquad \square \tag{18}$$

Secondly, the attacker may analyze tag responses. These answers would have to be anonymized to protect privacy location. However, the use of fresh random numbers is not enough to guarantee this required and important property. Messages have to be carefully designed when they are constructed, using modular operations (Alomair and Poovendran, 2008). As in the above case, we can show how privacy location is compromised by means of a game $\mathcal{G}$.

Proof B. We start observing a bad property of tag responses, which is expressed by the following Lemma.

Lemma 1. *In CD-EPC protocol, tags respond with a constant* $Y = CRC(N'_{T_i}(j) \oplus K'_{T_i}(j) \oplus EPC_{T_i})$ *value despite using of different nonces in each authentication session.*

The verification of the above Lemma is straightforward. After reader authentication, the tag answers the reader by sending values RND_2, X, *and* Y:

$$X = K'_{T_i}(j) \oplus EPC_{T_i} \oplus RND_2 \tag{19}$$

$$Y = CRC(RND_2 \oplus N'_{T_i}(j) \oplus X) \tag{20}$$

Combining the above equations, a constant value is obtained:

$$\begin{aligned} Y &= CRC(RND_2 \oplus N'_{T_i}(j) \oplus k'_{T_i}(j) \oplus EPC_{T_i} \oplus RND_2) \\ &= CRC(N'_{T_i}(j) \oplus k'_{T_i}(j) \oplus EPC_{T_i}) \end{aligned} \tag{21}$$

Lemma 1 is used in the untraceability ($\mathcal{G}$) game. Specifically, the adversary $\mathcal{A}$ performs the following steps:

Learning: Eavesdrops on an authentication session between the reader and tag $\mathcal{T}_0$ and stores value Y.

Challenge: Some time later, the adversary chooses two fresh tags T_0 and T_1. Then, one of these tags is randomly selected ($\mathcal{T}_i \in \{\mathcal{T}_0,\mathcal{T}_1\}$) and presented to the adversary. The adversary eavesdrops on an authentication session between this unknown tag and a legitimate reader.

Guessing: If the same Y value is captured, $\mathcal{A}$ conjectures that the unknown tag is in fact $\mathcal{T}_0$. Otherwise, it supposes it to be $\mathcal{T}_1$. Note that the probability that tag $\mathcal{T}_1$ and $\mathcal{T}_0$ lead to the same value for $Y=CRC(.)$ value is very low: $1/2^{16}$.

So, the proposed protocol does not offer protection against traceability because the $Adv_{\mathcal{A}}^{(UNT)}(k)$ is significant:

$$Adv_{\mathcal{A}}^{(UNT)} \simeq \left|\frac{1}{2}-\frac{1}{2^{16}}\right| \qquad \square \tag{22}$$

Summarizing, we have shown in this section, by means of two different (but related) passive attacks, how privacy location is not protected in the CD-EPC protocol. An attacker may exploit this vulnerability to associate a tag with its holder, and track him or her as he/she passes through different readers in, for example, different places in a city.

5.3. Tag impersonation

In this section we show how an attacker is able to impersonate a legitimate tag just by eavesdropping one authentication session between the reader and the tag. The exploitation of this attack can extend for an indefinite period of time, as the internal values of the tag tuple $\{EPC, K_{T_i}(j), N_{T_i}(j)\}$ remain constant throughout its life. In Burmester et al. (2009), a similar attack is suggested but its proof is not included.

Theorem 4. *In the CD-EPC protocol, on eavesdropping one authentic session, an adversary is able to respond to the reader's queries correctly – with the consequence of tag impersonation – by sending* $RND2'$, $X' = X \oplus \Delta$, $Y' = Y$ *message, where* $\Delta = RND2' \oplus RND2$.

Proof. *Step* 1: The attacker eavesdrops one authentication session between the reader and the tag.

(1) $R \rightarrow T : M_{req}, CRC(N_{T_i}(j) \oplus RND1), RND1$
(2) $T \rightarrow R : RND2, X, Y$
(3) $R \rightarrow T : M_{resp}$,

Step 2: The attacker can impersonate the tag by sending the following message:

(1) $R \rightarrow T : M_{req}, CRC(N_{T_i}(j) \oplus RND1'), RND1'$
(2) $T \rightarrow R : RND2', X', Y'$
where $X' = X \oplus \Delta, Y' = Y$
and $\Delta = RND2' \oplus RND2$

Upon receiving X', Y' and $RND2'$, the reader uses $N_{T_i}(j)$ to compute its local Y' value and compare this with the received value:

$$\begin{aligned} Y' &= CRC(RND'_2 \oplus N_{T_i}(j) \oplus X') \\ &= CRC(RND'_2 \oplus N_{T_i}(j) \oplus X \oplus \Delta) \\ &= CRC(RND'_2 \oplus N_{T_i}(j) \oplus X \oplus RND'_2 \oplus RND_2) \\ &= CRC(RND_2 \oplus N_{T_i}(j) \oplus X) = Y \end{aligned} \tag{23}$$

The reader, therefore, accepts RND'_2, X', Y' as a valid message and authenticates the adversary. Finally, the reader obtains the static identifier of the impersonated tag:

$$\begin{aligned} EPC_{T_i}{}' &= K_{T_i}(j) \oplus RND'_2 \oplus X' \\ &= K_{T_i}(j) \oplus RND'_2 \oplus X \oplus RND_2 \oplus RND'_2 \\ &= K_{T_i}(j) \oplus RND_2 \oplus X = EPC_{T_i} \qquad \square \end{aligned} \tag{24}$$

5.4. Denial of service

The attack described in the last section can be generalized to perform a denial of service (DoS) attack. This exploits the linearity of the *CRC* function and its lack of resistance against active attacks due to the use of the exclusive-OR operation (Alomair and Poovendran, 2008).

First, we briefly explain the weaknesses of the exclusive-OR operation. Suppose a tag and a reader share an unique identifier (EPC_{T_i}) and a secret key (K_{T_i}). Fresh random numbers are denoted as *RND*. Let $\mathcal{P}$ be the following simple authentication protocol:

$$\begin{aligned} R \rightarrow T: \quad & m_1 \| m_2 \\ & m_1 = EPC_{T_i} \oplus RND \\ & m_2 = MAC_{K_{T_i}} \oplus (RND) = K_{T_i} \oplus RND \end{aligned}$$

where *MAC* symbolizes a message authenticate code.

Upon receiving message m_1 and its MAC m_2, the reader performs a bitwise XOR between m_1 and the tag's *EPC* to extract RND'. Then, the result of computing a bitwise XOR between the obtained RND' value and its version of the secret key is compared with m_2. If this comparison holds, authentication process is successful.

Attack: Considering modifying m_1 to $m'_1 = m_1 \oplus b$, b being any non-zero bit string. Upon receiving m_1, the extracted RND' is $RND \oplus b$. The attacker will have to modify m_2 to $m'_2 = m_2 \oplus b = K_{T_i} \oplus (RND \oplus b)$ in order to pass unnoticed. Therefore, all the adversary has to do is to disturb both messages m_1 and m_2 by the same non-zero string.

Theorem 5. *In the CD-EPC protocol, after eavesdropping one authentic session, an adversary is able to respond to the reader's queries correctly and lead the reader to an incorrect EPC value* ($EPC_{T_i}'' = EPC_{T_i} \oplus \delta$) *by sending* $X' = X \oplus RND_3$, $Y' = Y \oplus CRC(RND_3) \oplus CRC(\Delta)$ *message, where* $\Delta = RND2' \oplus RND2$ *and* $\delta = \Delta \oplus RND_3$.

Proof. *Step* 1: The attacker eavesdrops one authentication session and blocks or alters the tag's response.

(1) $R \rightarrow T : M_{req}, CRC(N'_{T_i}(j) \oplus RND_1),\ RND_1$

(2) $T \rightarrow R : (blocked)\ RND_2, X, Y$

Step 2: The attacker can supplant the tag by sending the following message:

(2′) $A \rightarrow R : RND_2', X', Y'$
where $X' = X \oplus RND_3$,
$Y' = Y \oplus CRC(RND_3) \oplus CRC(\Delta)$,
and $\Delta = RND2' \oplus RND2$

Upon receiving X', Y' and $RND2'$, the reader uses $N_{T_i}(j)$ to compute its local Y'' value and compare this with the received value:

$$Y'' = CRC(RND'_2 \oplus N_{T_i}(j) \oplus X') = CRC(RND'_2 \oplus N_{T_i}(j) \oplus X \oplus RND_3) \quad (25)$$

As $b \oplus b = 0$,

$$Y'' = CRC(RND'_2 \oplus N_{T_i}(j) \oplus X \oplus RND_3 \oplus RND_2 \oplus RND_2) = CRC(\Delta \oplus N_{T_i}(j) \oplus X \oplus RND_2 \oplus RND_3) \quad (26)$$

Applying Eq. (10),

$$Y'' = CRC(\Delta) \oplus CRC(N_{T_i}(j) \oplus X \oplus RND_2) \oplus CRC(RND_3) = CRC(\Delta) \oplus Y \oplus CRC(RND_3) = Y' \quad (27)$$

The tag is authenticated and the reader obtains the static identifier by computing the following equation:

$$EPC_{T_i}'' = X' \oplus K_{T_i}(j) \oplus RND'_2 = X \oplus RND_3 \oplus K_{T_i}(j) \oplus RND'_2 \quad (28)$$

Applying Eq. (2),

$$EPC_{T_i}'' = K_{T_i}(j) \oplus EPC_{T_i} \oplus RND_2 \oplus RND_3 \oplus K_{T_i}(j) \oplus RND'_2 = EPC_{T_i} \oplus RND_3 \oplus \Delta \quad \square \quad (29)$$

An incorrect EPC_{T_i} is obtained, but the reader is not able to detect the trick. The reader will associate an incorrect identifier ($EPC_{T_i}'' = EPC_{T_i} \oplus \delta$, where $\delta = \Delta \oplus RND_3$) with the tuple ($N_{T_i}(j)$, $K_{T_i}(j)$). So, the proposed protocol is vulnerable to a DoS through a man-in-the middle attack.

5.5. Standard compatibility

The authors claim that the protocol conforms to the EPC-C1G2 specification. However, important technical aspects necessary to its successful implementation were ignored. We now summarize the most important properties of tags that comply with this standard:

- Tags are passive, so they receive all their operating energy from the readers' 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.
- Their very constrained resources and storage capabilities dictate that EPC-C1G2 tags cannot afford traditional cryptographic primitives.
- Tags include a 16-bit PRNG and 16-bit CRC function on-chip.
- Tags have two 32-bit PINs:
  - Kill PIN: The kill password is a 32-bit value stored in reserved memory. A reader will use a tag's kill password just once, to kill the tag and render it silent thereafter.
  - Access PIN: The access password is a 32-bit value stored in reserved memory. Tags with a non-zero access password will require a reader to issue this password before transitioning to the secure state, which will allow reading and writing in password protected fields.

The CD-EPC protocol is described by the following three equations:

$$CRC(N_{T_i}(j) \oplus RND_1) \quad (30)$$

$$X = K_{T_i}(j) \oplus EPC_{T_i} \oplus RND_2 \quad (31)$$

$$Y = CRC(RND_2 \oplus N_{T_i}(j) \oplus X) \quad (32)$$

Eq. (31) sets $K_{T_i}(j)$, EPC_{T_i} and RND_2 to the same l-bit length. As X is a component of Eq. (32), RND_2 and $N_{T_i}(j)$ are l-bit length just as the above mentioned variables. Finally, as $N_{T_i}(j)$ is l-bit length, RND_1 is forced to the same length by Eq. (30). Summarizing, all the variables must have the same length to run the CD-EPC protocol. As the *EPC* unique identifier must have a length of 96 or 198 bits for compatibility with all encoding schemes (i.e. GID, SGTIN, SSCC) defined by EPCGlobal (EPC Tag, 2008), the value l would have to be fixed to one of these two values. However, the lengths of RND_i, $K_{T_i}(j)$ and $N_{T_i}(j)$ do not conform with the EPC-C1G2 standard. Although it is not explicitly mentioned by the authors, we can assume that $N_{T_i}(j)$ and $K_{T_i}(j)$ are equivalent to the access and kill passwords as proposed in the standard. A comparison between the length of the variables defined in the standard and the variables used in the proposed protocol is shown in the following table:

Variable	EPC-C1G2 standard	CD-EPC scheme
EPC_{T_i}	96 or 198 bits	96 or 198 bits
RND_i	16 bits	96 or 198 bits
$N_{T_i}(j)$	32 bits	96 or 198 bits
$K_{T_i}(j)$	32 bits	96 or 198 bits

So, CD-EPC protocols triples or sextuples the memory requirements ($password-length = 32m$), where $m = 3$ or 6) compared with the standard. Additionally, the 16-bit PRNG has to be invoked 12 or 6 times each time a new nonce is necessary, which results in a significant reduction in the number of answers/s that the tags can provide.

6. Azumi: a mutual authentication protocol compliant with the EPC-C1G2 standard

Gen-2 tags are very limited devices, specially regarding its computing and storage capabilities. Only very simple operations (i.e. bitwise operations), a 16-bit CRC function and a 16-bit PRNG are supported on-board on these transponders. Many previous proposals attempted to increase the security offered by the EPC-C1G2 standard, but none of these were successful. More precisely, the security of these schemes heavily relied on the abuse of the CRC function, in many cases asking the CRC properties only to be found on a cryptographic hash function. However, CRC functions are linear (see Section 4.2) and should not be used for any cryptographic purpose—only for detection of random errors in the channel. Our proposed protocol, named `Azumi`, does not make the same mistake, and will only use the PRNG supported on-chip, and the two passwords that each tag owns.

In some scenarios only tags are authenticated (unilateral authentication), but mutual authentication (reader and tag) is the most commonly adopted solution. We roughly base our proposal on the ISO/IEC 9798 standard, instead of starting from scratch, with some modifications that are needed for the intended environments. In this standard, three different mechanisms for entity authentication are described: (1) entity authentication using symmetric techniques; (2) entity authentication using public key; (3) entity authentication using a cryptographic check function. We opted for the last option due to the severe computational restrictions of Gen-2 tags, and the fact that this is the most easily implementable. Generally, in these schemes, the two entities (prover/verifier) share a secret authentication key. An entity corroborates its identity by demonstrating knowledge of the shared key. This is accomplished by using a secret key with a cryptographic check function applied to specific data to obtain a cryptographic check value. This value can be recalculated by the verifier and compared with the received value. As Gen-2 tags are passive and have to be energized by readers, a three-pass mutual authentication protocol seems the most advisable option. The messages exchanged between the entities A and B are described below. Each entity has an identifier (ID_A and ID_B, respectively) and these share a key (K_{AB}). $f_k(x\|y)$ denotes a one-way function on k and $x\|y$, where $\|$ is concatenation. Finally, RND_x represents a random number generated by entity x and Text_i denotes any optional message field.

$$B \rightarrow A : RND_B \| \text{Text}_1$$

$$A \rightarrow B : RND_A \| \text{Text}_3 \| f_{K_{AB}}(RND_A \| RND_B \| ID_B \| \text{Text}_2)$$

$$B \rightarrow A : \text{Text}_5 \| f_{K_{AB}}(RND_B \| RND_A \| ID_A \| \text{Text}_4)$$

We introduce in the following the notation used in the `Azumi` protocol. We emphasize that the bit length of all variables is defined in conformity with the standard.

$N_{\mathcal{T}_i}$	32-bit access password of tag $\mathcal{T}_i$
$N16_{\mathcal{T}_i}$	the 32 bits of the access password are divided into two 16-bit blocks, and then the two blocks are XORred to get $N16_{\mathcal{T}_i}$
$K_{\mathcal{T}_i}$	32-bit kill password of tag $\mathcal{T}_i$
$K16_{\mathcal{T}_i}$	the 32 bits of the kill password are divided into two 16-bit blocks, and then the two blocks are XORred to get $K16_{\mathcal{T}_i}$
$EPC_{\mathcal{T}_i}$	96-bit *EPC* identification number of tag $\mathcal{T}_i$
$EPC16_{\mathcal{T}_i}$	the 96 bits of the EPC identifier are divided into six 16-bit blocks, and then the six blocks are XORred to get $EPC16_{\mathcal{T}_i}$
$ID16_{R_i}$	a 16-bit identification number of the ith reader
CRC()	16-bit cyclic redundancy code (CRC) function
PRNG()	16-bit pseudo-random number generator
RND	16-bit random number
$\oplus$	exclusive-OR operation
M_{req}	reader's request message
M_{resp}	reader's response message

Each tag stores the tuple $\{EPC16_{\mathcal{T}_i}, N16_{\mathcal{T}_i}, K16_{\mathcal{T}_i}\}$ in its memory. The back-end database has to store the old and the new values of the updated variables to combat desynchronization attacks. More precisely, in each row of the database, the tuple $\{EPC16_{\mathcal{T}_i}, N16^{old}_{\mathcal{T}_i}, K16^{old}_{\mathcal{T}_i}, N16^{new}_{\mathcal{T}_i}, K16^{new}_{\mathcal{T}_i}\}$ is stored. Then registration phase is conducted in the same way as in CD-EPC, but there are two main differences. We operate with a 16-bit version of variables and in the back-end database the old and new variables are initially set to the same value ($N16^{old}_{\mathcal{T}_i} = N16^{new}_{\mathcal{T}_i}$ and $K16^{old}_{\mathcal{T}_i} = K16^{new}_{\mathcal{T}_i}$).

Upon registration, tags and readers can mutually communicate. The mutual authentication process is depicted in Fig. 3. The details of the messages exchanged are presented below:

Step 1: When the reader wants to access a tag, it sends a request message M_{req} – the reader can be easily identified (with $ID16_{R_i}$) by this message – which includes the random number RND_{R_i} which is also sent to the tag.

Step 2: After receiving a reader request, the tag generates two random numbers (RND_{T_i}, RND'_{T_i}), then computes an authentication token A, an anonymous version B of its static identifier, and

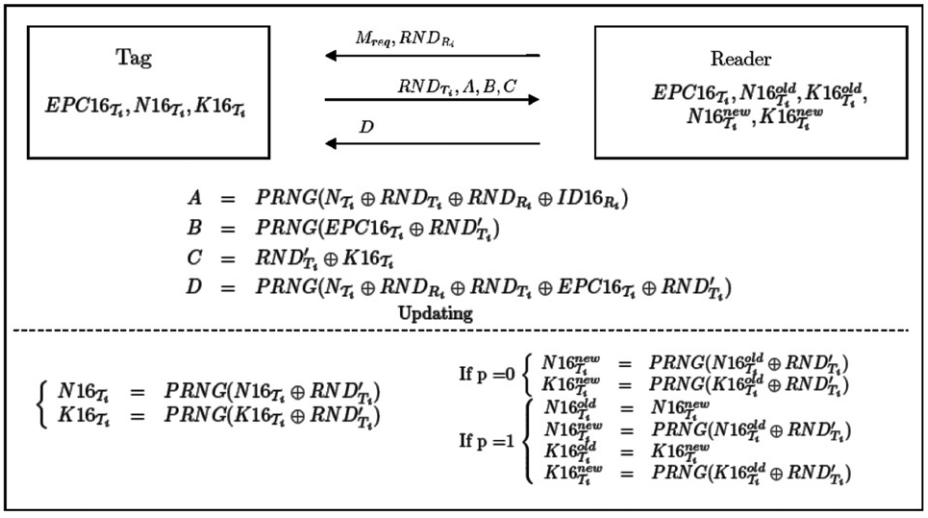

Fig. 3. Azumi protocol.

an encrypted value C of the second random number generated by the tag:

$$A = PRNG(N_{T_i} \oplus RND_{T_i} \oplus RND_{R_i} \oplus ID16_{R_i})$$

$$B = PRNG(EPC16_{T_i} \oplus RND'_{T_i})$$

$$C = RND'_{T_i} \oplus K16_{T_i}$$

Finally, the tag sends $\{RND_{T_i},A,B,C\}$ to the reader.

Step 3: Upon receiving $\{RND_{T_i},A,B,C\}$, the reader performs the followings operations:

Tag identification: The reader initiates a search process in the database. For each row, it picks up $K16_{T_i} \in \{K16^{old}_{T_i},K16^{new}_{T_i}\}$ and $EPC16_{T_i}$. Then, RND'_{T_i} is disclosed by XORing C and $K16_{T_i}$. Finally, a local version of B is computed. If the received and computed values match, the tag is successfully identified. If not, the reader repeats the procedure with the data of the next row. The protocol is aborted in case the end of the database is reached.

Tag authentication: Once the tag is identified, the reader has access to all the private information linked with the tag. The reader computes a local version A^* of A. If $A^*=A$, the tag is authenticated. Otherwise, the protocol is aborted. We emphasize here that the old/new value of N_{T_i} is used depending on wether the old/new value of $K16_{T_i}$ was used in the tag identification phase. Upon a successful authentication of the tag, the reader calculates its authentication token D:

$$D = PRNG(N_{T_i} \oplus RND_{R_i} \oplus RND_{T_i} \oplus EPC16_{T_i} \oplus RND'_{T_i})$$

Updating phase: The updating of internal values is a crucial step to stop attacks and to provide forward security. The updating phase is determined by the usage of old ($p=0$) or new values ($p=1$) in the identification/authentication phases. Specifically, the following equations define the updating phase:

$$\text{If } p=0 \begin{cases} N16^{new}_{T_i} = PRNG(N16^{old}_{T_i} \oplus RND'_{T_i}) \\ K16^{new}_{T_i} = PRNG(K16^{old}_{T_i} \oplus RND'_{T_i}) \end{cases}$$

$$\text{If } p=1 \begin{cases} N16^{old}_{T_i} = N16^{new}_{T_i} \\ N16^{new}_{T_i} = PRNG(N16^{old}_{T_i} \oplus RND'_{T_i}) \\ K16^{old}_{T_i} = K16^{new}_{T_i} \\ K16^{new}_{T_i} = PRNG(K16^{old}_{T_i} \oplus RND'_{T_i}) \end{cases}$$

Finally, the reader sends D to the tag.

Step 4: After receiving D, the tag computes a local version D^* of the authentication token. If $D^*=D$, the reader is authenticated and the mutual authentication process is completed. Moreover, the tag has confirmation that the reader received and correctly deciphered the second nonce—used in the updating phase. That is, the reader and the tag have confirmation of being in a synchronized state and share the same fresh updating material (RND'_{T_i}). Finally, the tag updates its internal values:

$$\begin{cases} N16_{T_i} = PRNG(N16_{T_i} \oplus RND'_{T_i}) \\ K16_{T_i} = PRNG(K16_{T_i} \oplus RND'_{T_i}) \end{cases}$$

Summarizing, in this Section we have introduced a new protocol compliant with the EPC-C1G2 standard, inspired in the time-tested ISO/IEC 9798. Nevertheless, some minor changes were introduced to ensure the privacy of tag's holders. The static identifier of a tag is anonymized and securely transmitted in an identification token (B). In fact, three random numbers are linked to each session and at least one of these nonces is used in each message computation ($\{A,B,C,D\}$). Replay attacks and traceability attacks are stopped by guaranteeing the freshness and unlinkability of the transmitted messages. The key updating phase provides forward security and makes the cryptanalysis of `Azumi` much harder. Finally, and as a countermeasure against DoS attacks, the old and potential new values of $N16_{T_i}$ and $K16_{T_i}$ are both stored in the reader.

We emphasize here that the security of `Azumi` resides in the primitive used as checking function. We use here a 16-bit PRNG because it is the only cryptographic primitive supported on Gen-2 compliant tags—standard cryptographic is far above the capabilities of this kind of tags. So the tokens exchanged in the protocol are of length 16 bits. In a brute force attack or exhaustive key search, the attacker repeatedly checks all possible keys until a success is detected (i.e. the correct key is found). The resources required for a brute force attack scale exponentially with increasing key size, not linearly (e.g. a {8,16,32}-bit key implies a key space of $2^{\{8,16,32\}}$, respectively). In our case, tokens are of length 16-bits, so the security level offered by a token – considering that the attacker does not know any weaknesses in the checking function and uses a brute force attack – is upper bounded by $(1/2^{16})$. In `Azumi`, the tag sends an authentication token (A) and an anonymized version of its static identifier (B). Assuming independence between the authentication and the identification token, and taking into account that the bit length of both tokens is 16, the security level against tag impersonation is thus upper bounded by $1/2^{32}$ (i.e. $1/2^{16} \cdot 1/2^{16}$). Regarding reader impersonation, the scheme offers a lower security of $(1/2^{16})$ because the reader only sends D as an authentication token.

The reader should note here that the security of the `Azumi` protocol is inherently limited by the conformity with the standard and not by the design proposed. We are aware that using a 16-bit key length – as in the tokens – is far away from the requirements of standard cryptography (ECRYPT, 2010), where a key length greater than 80-bits is generally demanded. Nevertheless, in this protocol the compliance with EPC-C1G2 standard is an essential requirement.

7. Conclusions

EPC-C1G2 is one of the most important standards related to RFID technology. In fact, it seems to have become the de facto standard for low-cost RFID tags. Because of its weak security, many authors have proposed enhanced schemes within the EPC framework. However, all these schemes have proven to be as insecure as the standard (Chien and Chen, 2007; Han and Kwon, 2009; Lim and Li, 2007; Peris-Lopez et al., 2009, 2008). In 2009, a new mutual authentication protocol was proposed by Chen and Deng that claimed to offer better security margins. In this paper, the security of this scheme is scrutinized and we show how an attacker is able to impersonate a tag or a reader, to trace a tag, and even to launch a DoS attack. These security vulnerabilities are all due to the use of the CRC function and take advantage of its linearity. Furthermore, the DoS attack is possible due to the lack of resistance against active attacks of the exclusive-OR operation. Finally, we additionally show how the protocol has non-trivial implementation difficulties because of the length of the involved variables, which is not compliant with that of the standard.

After the analysis of CD-EPC, we propose a new scheme attempting to avoid the security deficiencies of its predecessors. The `Azumi` protocol is based on well-known cryptographic techniques (ISO/IEC 9798). Moreover, the compatibility with the standard is guaranteed as only the 16-bit PRNG and XOR operations are involved. Therefore, `Azumi` offers a better security level and may be integrated in any EPC-C1G2 compliant tag without any modification.

Acknowledgments

We want to thank the anonymous reviewers that helped a lot on improving the quality of this paper.

References

Alomair, B., Poovendran, R., 2008. On the authentication of RFID systems with bitwise operations. In: Proceedings of the Second IFIP Conference on New Technologies Mobility and Security—NTMS'08, pp. 1–6.

Anarchriz. 1999. CRC and how to reverse it. http://www.woodmann.com/fravia/crctut1.htm.

Bailey, D., Juels, A., 2006. Shoehorning security into the EPC standard. International Conference on Security in Communication Networks—SCN'06, Lecture Notes in Computer Science, vol. 4116. Springer-Verlag, pp. 303–320.

Burmester, M., de Medeiros, B., Munilla, J., Peinado, A., 2009. Secure EPC Gen2 compliant radio frequency. Cryptology ePrint Archive, Report 2009/149, 2009. http://eprint.iacr.org/.

Chen, C.-L., Deng, Y.-Y., 2009. Conformation of EPC class 1 generation 2 standards RFID system with mutual authentication and privacy protection. Engineering Applications of Artificial Intelligence 22 (8), 1284–1291.

Chien, H.Y., Chen, C.H., 2007. Mutual authentication protocol for RFID conforming to EPC class-1 generation-2 standards. Computer Standards and Interfaces, Elsevier Science Publishers 29 (2), 254–259.

Clark, A., 1994. Modern optimisation algorithms for cryptanalysis. In: Proceedings of the Second Australian and New Zealand Conference on Intelligent Information Systems, pp. 258–262.

Class-1 generation 2 UHF air interface protocol standard version 1.2.0: "Gen 2". 2008. http://www.epcglobalinc.org/standards/.

EPC Tag data standard version 1.4. 2008. http://www.epcglobalinc.org/standards/.

Yearly report on algorithms and keysizes, Technical Report D.SPA.13 Rev. 1.0, ICT-2007-216676, ECRYPT, 2010.

GS1—EPCglobal. 2009. http://www.epcglobalinc.org/.

Giddy, J.P., Safavi-Naini, R., 1994. Automated cryptanalysis of transposition ciphers. The Computer Journal 37 (5), 429–436.

Han, D., Kwon, D., 2009. Vulnerability of an RFID authentication protocol conforming to EPC class 1 generation 2 standards. Computer Standards and Interfaces 31 (4), 648–652.

Hernandez Castro, J., Isasi Viuela, P., 2005. New results on the genetic cryptanalysis of tea and reduced-round versions of xtea. New Generation Computing 23, 233–243.

Hernandez-Castro, J.C., Tapiador, J.E., Peris-Lopez, P., Clark, J.A., Talbi, E.-G., 2009. Metaheuristic traceability attack against slmap an rfid lightweight authentication protocol. In: IEEE International Symposium on Parallel Distributed Processing, pp. 1–5.

Information technology—Radio frequency identification for item management—Part 6: parameters for air interface communications at 860 MHz to 960 MHz. http://www.iso.org, 2005.

ISO—International Organization for Standardization. http://www.iso.org/, 2009.

Karthikeyan, S., Nesterenko, M., 2005. RFID security without extensive cryptography. In: Proceedings of the SASN'05.

Knudsen, L., Meier, W., 1999. Cryptanalysis of an identification scheme based on the permuted perceptron problem. Advances in Cryptology EU—ROCRYPT'99, Lecture Notes in Computer Science, vol. 1592. Springer, Berlin/Heidelberg, pp. 363–374.

Konidala, D.M., Kim, K., January 2007. RFID tag-reader mutual authentication scheme utilizing tag's access password. Auto-ID Labs White Paper WP-HARDWARE-033.

Laskari, E.C., Meletiou, G.C., Stamatiou, Y.C., Vrahatis, M.N., 2005. Evolutionary computation based cryptanalysis: a first study. Nonlinear Analysis 63 (5–7), e823–e830.

Lim, T.L., Li, T., 2007. Addressing the weakness in a lightweight RFID tag-reader mutual authentication scheme. In: Proceedings of the IEEE International Global Telecommunications Conference—GLOBECOM'07. IEEE Computer Society Press, pp. 59–63.

Nguyen Duc, D., Park, J., Lee, H., Kwangjo, K., 2006. Enhancing security of epcglobal gen-2 RFID tag against traceability and cloning. In: Proceedings of the Symposium on Cryptography and Information Security.

Peris-Lopez, P., Hernandez-Castro, J.C., Estevez-Tapiador, J.M., Ribagorda, A., 2008. RFID Specification Revisited. In: The internet of Things: From RFID to The Next-Generation Pervasive Networked Systems. Auerbach Publications, Taylor & Francis Group, pp. 311–346.

Peris-Lopez, P., Hernandez-Castro, J.C., Tapiador, J.M.E., Ribagorda, A., 2009. Cryptanalysis of a novel authentication protocol conforming to EPC-C1G2 standard. Computer Standards and Interfaces 31 (2), 372–380.

Peris-Lopez, P., Li, T., Lim, T.L., Hernandez-Castro, J.C., Estevez-Tapiador, J.M., 2008. Vulnerability analysis of a mutual authentication scheme under the EPC class-1 generation-2 standard. In: Hand. of Conference on RFID Security.

Phan, R.,, 2009. Cryptanalysis of a new ultralightweight RFID authentication protocol—SASI. EEE Transactions on Dependable and Secure Computing 6 (4), 313–320.

Ranasinghe, D.C., 2007. Lightweight cryptography for low cost RFID. In: Networked RFID Systems and Lightweight Cryptography, pp. 311–346.

Song, J., Zhang, H., Meng, Q., Wang, Z., 2007. Cryptanalysis of four-round DES based on genetic algorithm. In: International Conference on Wireless Communications Networking and Mobile Computing, pp. 2326–2329.

Spillman, R., 1993. Cryptanalysis of knapsack ciphers using genetic algorithms. Cryptologia 17 (4), 367–377.

Spillman, R., Janssen, M., Nelson, B., Kepner, M., 1993. Use of a genetic algorithm in the cryptanalysis of simple substitution ciphers. Cryptologia 17 (1), 31–44.

Stigge, M., Pltz, Mller, W., Redlich, J.-P., 2006. Reversing CRC—theory and practice. Technical Report SAR-PR-2006-05, Humboldt-Universitat Berlin.

Westerbaan, B., 2005. Reversing CRC. http://blog.w-nz.com/archives/2005/07/15/reversing-crc/.