Securing Low-cost RFID Systems: an Unconditionally Secure Approach

Basel Alomair [a], Loukas Lazos [b], and Radha Poovendran [a]
[a] *Network Security Lab (NSL), University of Washington*
[b] *Electrical and Computer Engineering Dept., University of Arizona*
e-mail: {alomair,rp3}@u.washington.edu and llazos@ece.arizona.edu

Abstract. We explore a new direction towards solving the identity authentication problem in RFID systems. We break the RFID authentication process into two main problems: message authentication and random number generation. For parties equipped with a good source of randomness and a secure cryptographic primitive to authenticate messages, the literature of cryptography is rich with well-studied solutions for secure identity authentication. However, the two operations, random number generation and message authentication, can be expensive for low-cost RFID tags. In this paper, we lay down the foundations of a new direction towards solving these problems in RFID systems. We propose an unconditionally secure direction for authenticating RFID systems. We use the fact that RFID readers are computationally powerful devices to design a protocol that allows RFID readers to deliver random numbers to RFID tags in an unconditionally secure manner. Then, by taking advantage of the information-theoretic security of the transmitted messages, we develop a novel unconditionally secure message authentication code that is computed with a single multiplication operation. The goal of this work is to bring more research to the design of such unconditionally secure protocols, as opposed to the computationally secure protocols that have been proposed extensively, for the purpose of suiting the stringent computational capabilities of low-cost devices.

Keywords. RFID, unconditional security, authentication, secrecy

Introduction

While mutual authentication is a well-studied problem in the cryptographic literature, it becomes more challenging with the use of low-cost devices. Low-cost RFID tags, in particular, have limited computational capabilities that render them unable to perform sophisticated cryptographic operations. Hoping Moore's law will eventually render RFID tags computationally powerful, it might be tempting to consider the computational limitations of low-cost tags a temporary problem. The cost of tags, however, will remain a determining factor in the deployment of RFID systems in real life applications. When RFID technology is to replace barcodes to identify individual items, RFID tags will substantially contribute to the cost of these products. Even when the price of tags that can implement provably secure cryptography can be driven to 10 cents or less, it would still be impractical to attach them to low-cost items, e.g., 50-cent or cheaper products. When retailers are to choose between tags that can perform sophisticated cryptographic operations and cheaper tags that cannot, it seems inevitable that the cheaper tags will prevail.

The problem of mutual authentication in RFID systems has been studied under different constraints. Juels and Pappu proposed the use of a public key cryptosystem to solve the problem of consumer privacy in RFID banknotes [22]. Golle $et\ al.$ [19] proposed the universal re-encryption, where re-encryption of the ciphertext does not require the knowledge of the corresponding public key. As public key proposals might be suitable for some applications (e.g., banknotes [22], ePassports [4], credit cards [21], etc.), they are impractical for low-cost RFID tags. Consequently, proposals based on the hardness of breaking symmetric key primitives have been the most popular solution for RFID systems. Such solutions include the use of symmetric key encryption [13,14,12], pseudorandom functions (PRF) [26,24], cryptographic hash functions [34,5,7,10,27], or other NP-hard problems such as the linear parity with noise problem [23,16,9,8,28]. Although computationally secure symmetric key solutions are usually less expensive than asymmetric ones, they still require expensive operations and/or carefully designed iterations of complicated operations.

In order to come up with cheaper solutions, lightweight protocols that are not based on computational assumptions and require tags to perform only simple bitwise operations have been proposed, e.g., in [33,29,30]. Such proposals, however, were not based on rigorous security analysis and have been shown to have severe security flaws, see e.g., [25,17,1] for analysis of specific protocols. In fact, it has been shown in [2] that simple bitwise operations cannot lead to secure authentication protocols.

As can be observed from the above examples, previous secure RFID protocols are all *computationally secure*. Hoping to meet the limited computational capabilities of low-cost tags, we start the search for *unconditionally secure* protocols for RFIDs. Unconditional security relies on the freshness of the keys rather than the hardness of solving mathematical problems. Thus, an appropriately designed unconditionally secure protocol will normally require less computational effort.

Contributions. We propose the first UnConditionally Secure mutual authentication protocol for RFID systems (UCS-RFID). To minimize the computational effort on tags, we develop an unconditionally secure method for delivering random number from RFID readers to tags. Thus, allowing tags to benefit from the functionalities of random numbers without the hardware to generate them. Then, we take advantage of the secrecy of exchanged messages to develop a novel unconditionally secure technique for message authentication using only a single multiplication operation. Since modular multiplication is the most expensive operation tags are to perform, we show, for completeness, a simple modular multiplication algorithm that requires minimum circuitry.

The rest of the paper is organized as follows. In Section 1, we state our model assumptions. In Section 2, we describe our UCS-RFID protocol. In Section 3, we analyze the security of our UCS-RFID, and conclude the paper in Section 4. The modular multiplication algorithm is presented in the Appendix.

1. System Model

1.1. Computational Capabilities

We assume low-cost RFID tags identified via unique identifiers. Tags are assumed to be capable of performing bitwise (XOR) operations, in addition to modular multiplication and addition. RFID tags are not assumed to have the capability of performing traditional computationally-secure cryptographic primitives such as MACs, hash functions, random numbers generations, etc.

RFID readers are powerful devices capable of performing sophisticated cryptographic primitives. Readers are also assumed to be connected to the database via a secure link (whether by establishing a secure wireless channel or using wired connection) in order to retrieve information about tags. Addressing the security between RFID readers and the database is beyond the scope of this paper.

1.2. Adversarial Model

We assume an adversary with a complete control over the communication channel. The adversary can observe messages exchanged between readers and tags, initiate communication with a reader or a tag, modify message exchanged between the authorized reader and tags in the system, block messages and replayed them later, and generate messages of her own. We do not consider an adversary whose only goal is to jam the communication channel.

The adversary is modeled as a polynomial-time algorithm. Similar to the adversarial model proposed by Avoine in [3], given a tag T and a reader R, we assume that the adversary has access to the following oracles:

- *Send*(R, m_1, x_2, m_3): The adversary executes the protocol, acting as a tag. The adversary sends m_1 to identify itself to R; receives the reader's response, x_2; and authenticate itself with $m_3(x_2)$. This oracle models the adversary's ability to impersonate a tag in the system.
- *Query*(T, x_1, m_2, x_3): The adversary acts as the reader in an instance of the protocol. The adversary interrogates T; receives the tag's response, x_1; sends the message $m_2(x_1)$ to authenticate itself; and receives x_3. This oracle models the adversary's ability to impersonate valid readers.
- *Execute*(T, R): The tag T and the reader R execute an instance of the protocol. The adversary eavesdrops on the channel; this oracle models the adversary's ability to monitor the channel between tag and reader.

Observe that in practical RFID systems, unlike the *Send* and *Query* oracles, the adversary does not have complete control over the number of *Execute* oracles she can call on a particular tag. This is due to the fact that the *Execute* oracle simulates a complete protocol run between *authorized* reader-tag pairs. Thus, unless the adversary is physically capturing a tag, she does not have a complete control of when the tag is in the vicinity of an authorized reader.

1.3. Security Model

Inspired by the work of Bellare and Rogaway in [6], we define honest protocol runs as follows: A mutual authentication protocol run is said to be honest if the parties involved in the protocol run use their shared key to exchange messages, and the messages exchanged in the protocol run have been relayed faithfully (without modification).

Another term that will be used in the reminder of the paper is the definition of negligible functions. A function $\gamma : \mathbb{N} \rightarrow \mathbb{R}$ is said to be negligible if for any nonzero polynomial p, there exists N_0 such that for all $N > N_0$, $|\gamma(N)| < 1/|p(N)|$. That is, the function is said to be negligible if it converges to zero faster than the reciprocal of any polynomial function [18].

We now provide a formal definition of secure mutual authentication for RFID systems.

Definition 1 *A mutual authentication protocol for RFID systems is said to be secure if it satisfies the following conditions:*

1. *No information about the secret keys of the RFID tag is revealed by observing messages exchanged in protocol runs.*
2. ***Honest protocol*** $\Rightarrow$ ***Authentication:*** *if the protocol run is honest, the tag-reader pair must authenticate each other with probability one.*
3. ***Authentication*** $\Rightarrow$ ***Honest protocol:*** *the probability of authentication when the protocol is not honest is negligible in the security parameter.*

To model the adversary's attempt to authenticate herself to a reader (tag), we propose the following game between the challenger $\mathcal{C}$ (the RFID system) and an adversary $\mathcal{A}$.

1. $\mathcal{A}$ signals $\mathcal{C}$ to begin the game.
2. $\mathcal{C}$ chooses a tag, T, at random, and a reader, R, and gives them to $\mathcal{A}$.
3. $\mathcal{A}$ calls the oracles *Query, Send*, and *Execute* using T and R (this is the data collecting phase).
4. $\mathcal{A}$ decides to stop and signals $\mathcal{C}$ to move on to the next phase.
5. $\mathcal{A}$ *Send* (*Query*) $\mathcal{C}$ as if it is a tag (reader) in the system (this is the actual attempt to break the security of the system).
6. If $\mathcal{A}$ is authenticated as a valid tag (reader), $\mathcal{A}$ wins the game.

Definition 1 implies that the protocol achieves secure mutual authentication only if the adversary's probability of winning the game is negligible in the security parameter.

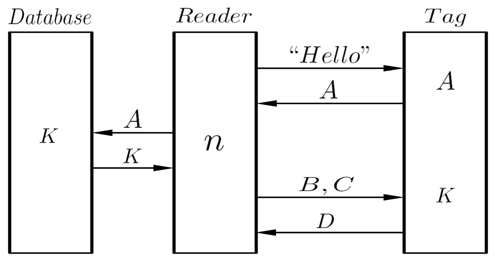
1.4. Preliminaries

The following notations will be adopted throughout the paper. For a finite integer p, $\mathbb{Z}_p$ will denote the finite integer ring with the usual addition and multiplication modulo p. $\mathbb{Z}_p^*$ will denote the multiplicative group modulo p; that is, the subset of $\mathbb{Z}_p$ with elements relatively prime to p. For the special case at which p is a prime integer, $\mathbb{Z}_p^*$ will contain all non-zero elements of $\mathbb{Z}_p$; that is, $\mathbb{Z}_p^* = \mathbb{Z}_p \backslash \{0\}$. $\mathbb{Z}_p^*$ and $\mathbb{Z}_p \backslash \{0\}$ will be used interchangeably to emphasize the multiplicative property or the exclusion of the zero element, respectively. Throughout the rest of the paper, random variables will be represented by bold font symbols, whereas the corresponding non-bold font symbols represent specific values that can be taken by these random variables.

The following are two important properties of the integer ring $\mathbb{Z}_p$ that will be used in the security analysis of UCS-RFID.

Lemma 1 *For any two integers α and β in $\mathbb{Z}_p$, if p is a prime integer and p divides $\alpha\beta$, then one of the integers α and β must be the zero element in $\mathbb{Z}_p$. Formally, if p is a prime integer, $\{\alpha\beta \equiv 0 \mod p\}$ implies that $\{\alpha \equiv 0 \text{ OR } \beta \equiv 0 \mod p\}$.*

Lemma 2 *Let p be a prime integer. Then, given an integer $k \in \mathbb{Z}_p^*$, for an r uniformly distributed over $\mathbb{Z}_p$, the value $\delta \equiv rk \mod p$ is uniformly distributed over $\mathbb{Z}_p$.*



$$
\begin{aligned}
A &\equiv n_\ell + k_a \mod 2^N\\
B &\equiv n + k_b \mod p\\
C &\equiv n \times k_c \mod p\\
D &= n_\ell \oplus k_d
\end{aligned}
$$

Figure 1. A schematic of one instance of the protocol.

Lemma 1 states that, for a prime integer p, the ring $\mathbb{Z}_p$ is an integral domain. Lemma 2 is a direct consequence of the fact that, for a prime integer p, the ring $\mathbb{Z}_p$ is a field. One more definition that is vital for this paper is the definition of Shannon's perfect secrecy [31].

Definition 2 (Perfect Secrecy [32]) *For a plaintext m and its corresponding ciphertext φ, the cipher is said to achieve perfect secrecy if* $\Pr(\boldsymbol{m} = m|\boldsymbol{\varphi} = \varphi) = \Pr(\boldsymbol{m} = m)$ *for all plaintext m and all ciphertext φ. That is, the a posteriori probability that the plaintext is m, given that the ciphertext φ is observed, is identical to the a priori probability that the plaintext is m.*

2. The proposed UCS-RFID Protocol

Based on pre-defined security requirements, a security parameter, N, is specified and a $2N$-bit prime integer, p, is chosen. Initially, each tag is loaded with an N-bit long identifier, $A^{(0)}$, and a secret key composed of five subkeys, i.e., $K^{(0)} = (k_a^{(0)}, k_b^{(0)}, k_c^{(0)}, k_d^{(0)}, k_u^{(0)})$. The length of k_a and k_d is N bits, while k_b, k_c, and k_u are $2N$-bit long. The subkeys, $k_a^{(0)}$ and $k_d^{(0)}$, and the identifier, $A^{(0)}$, are drawn independently and uniformly from $\mathbb{Z}_{2^N}$; $k_b^{(0)}$ is drawn uniformly from $\mathbb{Z}_p$; while $k_c^{(0)}$ and $k_u^{(0)}$ are drawn independently and uniformly from $\mathbb{Z}_p^*$. The subkeys k_a, k_b, k_c, and k_d will be used to generate messages exchanged in protocol runs, while the sole purpose of k_u is for updating the secret keys to maintain certain properties (details are discussed later).

The security of the protocol relies on the reader's ability to convey a random nonce to the tag in an *authenticated* and *secret* manner. When an RFID reader interrogates a tag within its communication range, the tag responds with its identifier, A. Once the tag has been identified, the reader generates a $2N$-bit long random nonce, n, and delivers it to the tag. If the reader is authenticated successfully, the received n will be used by the tag to authenticate itself to the valid reader.

For the rest of the paper, quantities involved in the generation of exchanged messages in different protocol runs will be differentiated by superscripts. When differentiation between protocol runs is unnecessary, superscripts will be dropped for ease of notation.The proposed UCS-RFID enables the mutual authentication between an RFID reader and a tag by executing four phases: a tag identification phase, a reader authentication phase, a tag authentication phase, and a key updating phase. Figure 1 depicts a single protocol run of the proposed UCS-RFID.

2.1. Tag Identification Phase

In order to carry out the authentication process, the reader must identify the tag it is communicating with to access its key information.
Step 1. The reader announces its presence by broadcasting a *"Hello"* message.
Step 2. The tag responds to the *"Hello"* message by sending its current identifier, A.
Step 3. The reader looks up the database for the key $K = (k_a, k_b, k_c, k_d, k_u)$ corresponding to the tag's current identifier, A.[1] If A is not recognized as a valid identifier, the tag is rejected.

2.2. Reader Authentication Phase

This is one of the most important phases in the proposed protocol. In this phase, the RFID reader authenticates itself to the tag by proving its knowledge of the tag's subkeys k_b and k_c. More importantly, the reader delivers a nonce, n, to the tag in an authenticated and perfectly secret manner.
Step 4. The reader generates a $2N$-bit random nonce, n, drawn uniformly from the *multiplicative group* $\mathbb{Z}_p^*$. We emphasize that n must be an unpredictable nonce; predictable nonces such as time stamps do not induce the required randomness.
Step 5. With k_b, k_c, and n, the reader broadcasts two messages, B and C, generated according to the following formulas:

$$B \equiv n + k_b \mod p, \tag{1}$$

$$C \equiv n \times k_c \mod p. \tag{2}$$

Step 6. Upon receiving B and C, the tag extracts n from message B and verifies its integrity using message C. The reader is authenticated if and only if the following integrity check is satisfied,

$$(B - k_b) \times k_c \equiv C \mod p. \tag{3}$$

2.3. Tag Authentication Phase

In the tag authentication phase, the tag is authenticated by its ability to extract the correct nonce, n, and its knowledge of the secret key k_d.
Step 7. If the reader failed the authentication process, the tag aborts the protocol. Otherwise, the tag broadcasts message D, given by

$$D = n_\ell \oplus k_d, \tag{4}$$

where n_ℓ denotes the N most significant bits of n.
Step 8. Upon receiving D, the reader authenticates the tag by verifying that the received D is equal to $n_\ell \oplus k_d$. Otherwise, the tag is rejected.

[1] Database management is beyond of the scope of this paper.

2.4. Key Update Phase

After a mutual authentication between the RFID reader and the tag is achieved, the parameters are updated at the database and the tag for the next mutual authentication run.
Step 9. The reader and the tag update the key, K, and the tag identifier, A. Let $A^{(m)}$, $k_i^{(m)}$, and $n^{(m)}$ denote the identifier A, k_i, and n used to execute the m^{th} protocol run; let n_r denotes the N least significant bits of n. Then, the parameters are updated as follows,

$$k_a^{(m+1)} = n_r^{(m)} \oplus k_a^{(m)}, \tag{5}$$

$$k_b^{(m+1)} \equiv k_u^{(m)} + (n^{(m)} \oplus k_b^{(m)}) \mod p, \tag{6}$$

$$k_c^{(m+1)} \equiv k_u^{(m)} \times (n^{(m)} \oplus k_c^{(m)}) \mod p, \tag{7}$$

$$k_d^{(m+1)} = n_r^{(m)} \oplus k_d^{(m)}, \tag{8}$$

$$k_u^{(m+1)} \equiv k_u^{(m)} \times n^{(m)} \mod p, \tag{9}$$

$$A^{(m+1)} \equiv n_\ell^{(m)} + k_a^{(m+1)} \mod 2^N. \tag{10}$$

It is vital for the security of the protocol that the updated $k_b^{(m+1)}$ and $k_c^{(m+1)}$ remain uniformly distributed over $\mathbb{Z}_p$ and $\mathbb{Z}_p \backslash \{0\}$, respectively. Here is where the updating key, k_u, comes to play. In addition to inducing a desired independence between message $B^{(m)}$ and the updated $k_b^{(m+1)}$, and between message $C^{(m)}$ and the updated $k_c^{(m+1)}$, observe that, by Property 2, $k_u^{(m)}$ will always be uniformly distributed over $\mathbb{Z}_p^*$ (since the initial $k_u^{(0)}$ is drawn uniformly from $\mathbb{Z}_p^*$ and every generated nonce is a random element of $\mathbb{Z}_p^*$). Therefore, $k_b^{(m+1)}$ is uniformly distributed over $\mathbb{Z}_p$. However, there is a possibility that $k_c^{(m+1)}$ will be equal to zero; which will occur, *with negligible probability*, when $n^{(m)} \oplus k_c^{(m)}$ is congruent to zero modulo p. In this case, $n^{(m)} \oplus k_c^{(m)}$ in equation (7) is replaced with $n^{(m)} \times k_c^{(m)}$. Now, $n^{(m)} \times k_c^{(m)}$ is guaranteed not to be congruent to zero (by Property 1), which guarantees that $k_c^{(m+1)}$ is not zero. The reason for not starting with $n^{(m)} \times k_c^{(m)}$ in the update equation of $k_c^{(m+1)}$ is that this is equal to $C^{(m)}$, which will lead to revealing information about the nonce with the observation of multiple consecutive protocol runs. With the update procedure described above, $n^{(m)} \times k_c^{(m)}$ will be used for updating $k_c^{(m+1)}$ with negligible probability, and even when it is used, the adversary can never know that it is being used. Therefore, without loss of generality, we will assume for the rest of the paper that equation (7) always results in a $k_c^{(m+1)}$ that is uniformly distributed over $\mathbb{Z}_p \backslash \{0\}$.

3. Security Analysis

Before we show the security of our UCS-RFID, we will first prove our claims that, under our adversarial model, the integrity of the delivered nonce, n, can be verified using a single modular multiplication, and show that the random nonce is delivered to tags in an unconditionally secure manner.

3.1. Integrity of the Delivered Nonce

In this section, we will show how the integrity of the nonce, n, is preserved without resorting to computationally secure cryptographic primitives. The integrity of the delivered nonce in our UCS-RFID is accomplished in a novel way, by taking advantage of the properties of the integer field $\mathbb{Z}_p$, with only a single multiplication operation.

There are two cases to consider: modifying message B alone and modifying both B and C in order to make the tag authenticate a false nonce. Modifying message C alone, since its main purpose is to authenticate the received nonce, does not lead to the extraction of a modified nonce.

Lemma 3 *Given that k_c is uniformly distributed over $\mathbb{Z}_p\backslash\{0\}$, the probability of accepting a modified nonce by a valid tag is at most $1/(p-1)$.*
Proof: Assume that message B has been modified to B'. This modification will lead to the extraction of a nonce, n', different than the authentic n generated by the reader; that is, $n' \equiv B' - k_b \mod p$. Message C, however, is used to verify the integrity of the extracted n'. Let $n' \equiv n + \epsilon \mod p$; for some $\epsilon \in \mathbb{Z}_p\backslash\{0\}$. To be accepted by the tag, n' must satisfy the integrity check of equation (3). That is,

$$n' \times k_c \equiv (n+\epsilon) \times k_c \equiv (n \times k_c) + (\epsilon \times k_c) \overset{?}{\equiv} C \equiv n \times k_c \mod p. \tag{11}$$

Clearly, the congruence in equation (11) will be satisfied only if $\epsilon \times k_c \equiv 0 \mod p$. However, since k_c is a nonzero element by design, and $\epsilon \not\equiv 0$ (since $\epsilon \equiv 0$ implies that $n' \equiv n \mod p$), Property 1 guarantees that $\epsilon \times k_c \not\equiv 0 \mod p$. Therefore, the congruence of equation (11) can never be satisfied, and any modification of message B *alone* will be detected with probability *one*.

The second case to consider here is when both messages B and C are corrupted simultaneously. Assume that message B has been modified so that the extracted nonce becomes $n' \equiv n + \epsilon \mod p$; for some $\epsilon \in \mathbb{Z}_p\backslash\{0\}$. Also, assume that message C has been modified to $C' \equiv C + \delta \mod p$, for some $\delta \in \mathbb{Z}_p\backslash\{0\}$. The integrity of the extracted n' is verified using the received C' as follows:

$$\begin{aligned} C + \delta \equiv C' \overset{?}{\equiv} n' \times k_c &\equiv (n+\epsilon) \times k_c \\ &\equiv (n \times k_c) + (\epsilon \times k_c) \equiv C + (\epsilon \times k_c) \mod p. \end{aligned} \tag{12}$$

Equivalently, the false n' is accepted only if $\delta \equiv \epsilon \times k_c$. Since k_c is unknown to the adversary, for any fixed δ, by Property 2, there exists a unique $\epsilon \in \mathbb{Z}_p\backslash\{0\}$ that satisfies (12). Therefore, the probability of modifying both B and C in a way undetected by the tag is at most $1/(p-1)$ (equivalently, guessing the value of k_c). ■

3.2. Secrecy of the Delivered Nonce

Before we show that the nonce is delivered to the tag in an unconditionally secure manner, we need the following lemma.

Lemma 4 *Given that the preloaded subkeys $k_a^{(0)}$, $k_b^{(0)}$, $k_c^{(0)}$, and $k_d^{(0)}$ are mutually independent, the subkeys $k_a^{(m)}$, $k_b^{(m)}$, $k_c^{(m)}$, and $k_d^{(m)}$ at the m^{th} protocol run are mutually independent, for any $m \in \mathbb{N}$.*

Proof: Let $\boldsymbol{k_a^{(m)}}, \boldsymbol{k_b^{(m)}}, \boldsymbol{k_c^{(m)}}$, and $\boldsymbol{k_d^{(m)}}$ be the random variables representing the subkeys involved in the generation of the messages exchanged between an authorized RFID pair during the m^{th} protocol run of our UCS-RFID. Then, for any $k_a^{(1)}, k_b^{(1)}, k_c^{(1)}$, and $k_d^{(1)}$,

$$\begin{aligned}
&\Pr\left(\boldsymbol{k_a^{(1)}} = k_a^{(1)}, \boldsymbol{k_b^{(1)}} = k_b^{(1)}, \boldsymbol{k_c^{(1)}} = k_c^{(1)}, \boldsymbol{k_d^{(1)}} = k_d^{(1)}\right) \\
&= \sum_{n,k_u} \Pr\left(\boldsymbol{k_a^{(1)}} = k_a^{(1)}, \boldsymbol{k_b^{(1)}} = k_b^{(1)}, \boldsymbol{k_c^{(1)}} = k_c^{(1)}, \boldsymbol{k_d^{(1)}} = k_d^{(1)} | \boldsymbol{n} = n, \boldsymbol{k_u} = k_u\right) \\
&\qquad\qquad \cdot \Pr\left(\boldsymbol{n} = n, \boldsymbol{k_u} = k_u\right) \qquad (13) \\
&= \sum_{n,k_u} \Pr\left(\boldsymbol{k_a^{(0)}} = k_a^{(1)} \oplus n_r, \boldsymbol{k_b^{(0)}} = (k_b^{(1)} - k_u^{(0)}) \oplus n, \boldsymbol{k_c^{(0)}} = (k_c^{(1)} \times k_u^{(0)^{-1}}) \oplus n\right. \\
&\qquad\qquad \left., \boldsymbol{k_d^{(0)}} = k_d^{(1)} \oplus n_r\right) \cdot \Pr\left(\boldsymbol{n} = n, \boldsymbol{k_u} = k_u\right) \qquad (14) \\
&= \sum_{n,k_u} \Pr\left(\boldsymbol{k_a^{(0)}} = k_a^{(1)} \oplus n_r\right) \cdot \Pr\left(\boldsymbol{k_b^{(0)}} = (k_b^{(1)} - k_u^{(0)}) \oplus n\right) \\
&\qquad \cdot \Pr\left(\boldsymbol{k_c^{(0)}} = (k_c^{(1)} \times k_u^{(0)^{-1}}) \oplus n\right) \cdot \Pr\left(\boldsymbol{k_d^{(0)}} = k_d^{(1)} \oplus n_r\right) \cdot \Pr\left(\boldsymbol{n} = n, \boldsymbol{k_u} = k_u\right) \qquad (15) \\
&= \sum_{n,k_u} \frac{1}{2^N} \cdot \frac{1}{p} \cdot \frac{1}{p-1} \cdot \frac{1}{2^N} \cdot \Pr\left(\boldsymbol{n} = n, \boldsymbol{k_u} = k_u\right) \qquad (16) \\
&= \Pr\left(\boldsymbol{k_a^{(1)}} = k_a^{(1)}\right) \cdot \Pr\left(\boldsymbol{k_b^{(1)}} = k_b^{(1)}\right) \cdot \Pr\left(\boldsymbol{k_c^{(1)}} = k_c^{(1)}\right) \cdot \Pr\left(\boldsymbol{k_d^{(1)}} = k_d^{(1)}\right). \qquad (17)
\end{aligned}$$

Equations (15) and (16) hold due to the independence and the uniform distribution of the initial subkeys $(\boldsymbol{k_a^{(0)}}, \boldsymbol{k_b^{(0)}}, \boldsymbol{k_c^{(0)}}, \boldsymbol{k_d^{(0)}})$, respectively; while equation (17) holds due to the uniform distribution of the updated subkeys $(\boldsymbol{k_a^{(1)}}, \boldsymbol{k_b^{(1)}}, \boldsymbol{k_c^{(1)}}, \boldsymbol{k_d^{(1)}})$. The existence of $k_u^{(0)^{-1}}$, the multiplicative inverse of $k_u^{(0)}$ in $\mathbb{Z}_p$, is a direct consequence of the fact that $k_u^{(0)} \in \mathbb{Z}_p^*$. The proof of the lemma follows by induction. ■

Lemma 5 *At each instance of the protocol, the random nonce generated by the authorized reader in an instance of our UCS-RFID protocol is delivered to the tag in a perfectly secret manner.*

Proof: Fix k_b, k_c, and let $\boldsymbol{n}$ be uniformly distributed over $\mathbb{Z}_p \backslash \{0\}$. (Recall that, by Lemma 4, $\boldsymbol{k_b}$ and $\boldsymbol{k_c}$ are statistically independent in every protocol run; so, the superscript will be dropped for ease of notation.) Then the resulting $\boldsymbol{B}$ and $\boldsymbol{C}$ will be uniformly distributed over $\mathbb{Z}_p$ and $\mathbb{Z}_p \backslash \{0\}$ (by Property 2), respectively. Consequently, for any arbitrary $b \in \mathbb{Z}_p$ and $c \in \mathbb{Z}_p \backslash \{0\}$, the probability of $\boldsymbol{B}$ and $\boldsymbol{C}$ taking these specific values are $\Pr(\boldsymbol{B} = b) = 1/p$ and $\Pr(\boldsymbol{C} = c) = 1/(p-1)$.

Now, given a specific value of the random nonce $\boldsymbol{n} = n$, the probability that $\boldsymbol{B}$ takes a value b is

$$\Pr(\boldsymbol{B} = b | \boldsymbol{n} = n) = \Pr(\boldsymbol{k_b} = b - n) = 1/p. \qquad (18)$$

Similarly, given a specific value of the random nonce $\boldsymbol{n} = n$, the probability that $\boldsymbol{C}$ takes a value c is

$$\Pr(\boldsymbol{C} = c|\boldsymbol{n} = n) = \Pr(\boldsymbol{k_c} = c \times n^{-1}) = 1/(p-1). \tag{19}$$

Equations (18) and (19) hold since, by design, $\boldsymbol{k_b}$ and $\boldsymbol{k_c}$ are uniformly distributed over $\mathbb{Z}_p$ and $\mathbb{Z}_p\backslash\{0\}$, respectively. The existence of n^{-1} is a direct consequence of the fact that $n \in \mathbb{Z}_p^*$.

Therefore, for any nonce n and any values of b and c, Bayes' theorem [20] can be used to show that $\Pr(\boldsymbol{n} = n|\boldsymbol{B} = b) = \Pr(\boldsymbol{n} = n) = \Pr(\boldsymbol{n} = n|\boldsymbol{C} = c)$. That is, the a priori probabilities that the random nonce is n are the same as the a posteriori probabilities that the random nonce is n given the corresponding B and C. Hence, both B and C "individually" provided perfect secrecy. However, since they are both functions of the same variable, there might be information leakage about n revealed by the *combination* of B and C. One way of measuring how much information is learned by the observation of two quantities is the notion of mutual information. Consider an arbitrary $b \in \mathbb{Z}_p$ and arbitrary $c, n \in \mathbb{Z}_p^*$. Then, for independent $\boldsymbol{k_b}$ and $\boldsymbol{k_c}$ uniformly distributed over $\mathbb{Z}_p$ and $\mathbb{Z}_p^*$, respectively, we get:

$$\begin{aligned}
\Pr(\boldsymbol{B} = b, \boldsymbol{C} = c) &= \sum_n \Pr(\boldsymbol{B} = b, \boldsymbol{C} = c|\boldsymbol{n} = n)\Pr(\boldsymbol{n} = n) && (20)\\
&= \sum_n \Pr(\boldsymbol{k_b} = b - n, \boldsymbol{k_c} = c \times n^{-1})\Pr(\boldsymbol{n} = n) && (21)\\
&= \sum_n \Pr(\boldsymbol{k_b} = b - n)\Pr(\boldsymbol{k_c} = c \times n^{-1})\Pr(\boldsymbol{n} = n) && (22)\\
&= \sum_n \frac{1}{p} \cdot \frac{1}{p-1} \cdot \Pr(\boldsymbol{n} = n) && (23)\\
&= \Pr(\boldsymbol{B} = b) \cdot \Pr(\boldsymbol{C} = c). && (24)
\end{aligned}$$

Equation (22) holds by the independence of $\boldsymbol{k_b}$ and $\boldsymbol{k_c}$, while equations (23) and (24) hold by the uniform distribution of $\boldsymbol{k_b}$, $\boldsymbol{k_c}$, $\boldsymbol{B}$, and $\boldsymbol{C}$. Consequently, $\boldsymbol{B}$ and $\boldsymbol{C}$ are independent and, thus, their mutual information is *zero* [11]. In other words, observing both messages B and C gives no extra information about n than what they give individually. ■

Remark 1 Lemma 5 does not hold for an adversary who has observed multiple *consecutive* protocol runs between authorized reader-tag pairs. Consider observing three consecutive B messages, say $B^{(0)}, B^{(1)}, B^{(2)}$. The fundamental problem is that only $k_b^{(0)} \in \mathbb{Z}_p$ and $k_u^{(0)} \in \mathbb{Z}_p^*$ are involved in the update equation of the subkey k_b. Therefore, out of the total $(p-1)^3$ possible sequences of $\{n^{(0)}, n^{(1)}, n^{(2)}\}$, to an adversary who has observed $\{B^{(0)}, B^{(1)}, B^{(2)}\}$, there are only $p(p-1)$ possible $\{n^{(0)}, n^{(1)}, n^{(2)}\}$ sequences that could have generated the observed B's. A violation to the definition of perfect secrecy. Obviously, one can include more variables in the update equation of k_b but that will only increase the number of consecutive protocol runs an adversary is allowed to observe to a certain number.

However, breaking perfect secrecy does not imply breaking the system. In what follows, we provide an example to further illustrate the remark; then, we give detailed probabilistic analysis to show that the system can still provide unconditional security given some practical assumptions about the RFID system.

Example 1 This example illustrates the effect of observing consecutive protocol runs between authorized reader-tag pairs. For simplicity, assume that the used prime number is $p = 7$. Assume further that the initial keys $k_b^{(0)} = 2$ and $k_u^{(0)} = 5$ are preloaded into the tag. Consider the first three protocol runs as follows.
First run: let the generated nonce be $n_0 = 1$. Then by equation (1) the adversary can observe $B_0 = 3$ broadcasted by the reader. The tag and the reader will then update the keys according to equations (6) and (9) to $k_b^{(1)} = 1$ and $k_u^{(1)} = 5$.
Second run: let the generated nonce be $n_1 = 6$. Then by equation (1) the adversary can observe $B_1 = 0$. The tag and the reader will then update the keys according to equations (6) and (9) to $k_b^{(2)} = 5$ and $k_u^{(2)} = 2$.
Third run: let the generated nonce be $n_2 = 2$. Then the adversary can observe $B_2 = 0$. Now, with some algebra the adversary can construct the following system of equations:

$$B_0 = k_b^{(0)} + n_0, \tag{25}$$

$$B_1 = k_u^{(0)} + (n_0 \oplus k_b^{(0)}) + n_1, \tag{26}$$

$$B_2 = (k_u^{(0)} \times n_0) + \Big(n_1 \oplus \big(k_u^{(0)} + (n_0 \oplus k_b^{(0)})\big)\Big) + n_2. \tag{27}$$

Consider now the sequence $\{n_0 = 1, n_1 = 1, n_2 = 1\}$. Given the observed B's, by checking equations (25), (26), and (27), one can see that the sequence $\{n_0 = 1, n_1 = 1, n_2 = 1\}$ cannot satisfy the three equations simultaneously. Moreover, by checking all possible $6\times6\times6$ sequences, one can find that only 7×6 of them can satisfy all three equations simultaneously. The fundamental problem is that only $k_b^{(0)} \in \{0, 1, 2, 3, 4, 5, 6\}$ and $k_u^{(0)} \in \{1, 2, 3, 4, 5, 6\}$ are involved in the three equations.

Observe, however, that this does not imply anything more than that the sequence $\{n_0 = 1, n_1 = 1, n_2 = 1\}$ cannot generate the observed B's. That is, it does not imply that $n_0 \neq 1$, nor that $n_1 \neq 1$, nor that $n_2 \neq 1$. In other words, individually, any one of the n's can be equal to one (indeed, $n_0 = 1$ in the above example).

Therefore, for the adversary to obtain meaningful information, she must know the exact value of at least one of the nonces (so that possible values of other nonces can be eliminated). This can only occur if for at least one nonce n_i, only one value in $\mathbb{Z}_p^*$ is possible. That is, all the possible values n_i is allowed to take can be eliminated, except for exactly one value.

We will now provide probabilistic analysis of the number of consecutive protocol runs an adversary must observe in order to learn the value of at least one of the transmitted nonces.

By the randomness nature of the generated nonces, the total number of possible sequences is uniformly distributed over the nonces. That is, given there are $p(p-1)$ possible sequences, if the adversary has observed m consecutive protocol runs, each of the m nonces is expected to have $\sqrt[m]{p(p-1)}$ possible values. Therefore, for m consecutive protocol runs, the total number of possible values distributed over the m nonces is $m \sqrt[m]{p(p-1)}$.

To give a lower bound on the number of consecutive protocol runs an adversary must observe in order to infer at least one nonce with a certain probability, we use the well-known "balls in bins without capacity" problem in probability theory. Given r balls thrown uniformly at random at m bins, the probability that at least one bin remains empty is given by [15]:

$$\Pr(\text{at least one bin remains empty}) = \frac{\binom{r-1}{m-1}}{\binom{m+r-1}{m-1}}. \tag{28}$$

Given that each nonce will take at least one value, the problem reduces to distributing $(m \sqrt[m]{p(p-1)} - m)$ values uniformly at random at m nonces and finding the probability that at least one nonce does not receive another possible value. Substituting $r = m \sqrt[m]{p(p-1)} - m$ in equation (28), we plot the results in Figure 2. Each plot shows the number of consecutive protocol runs an adversary must observe in order to infer at least one nonce with a certain probability. In the top left plot, the security parameter N is 128-bit long, with only k_b and k_u are involved in the update equation of k_b. The plot in the top right shows the result when all secret keys are involved in the update equation of k_b. The two bottom plots shows the result when the used security parameter is 256-bit long.

As can be seen in Figure 2, the number of consecutive protocol runs an adversary has to observe to learn the value of at least one nonce is much higher than the number of protocol runs needed to break perfect secrecy. Depending on how many secret keys are used in the update equations and the length of the security parameter, for an adversary to have a 50% chance of exposing a secret nonce value, the number of *consecutive* protocol runs needed to be observed can be as high as 240 complete runs.

Remark 2 Observe that, unlike general computer communications, that many consecutive protocol runs can be sufficiently high for RFID systems. Consider, for example, an RFID tag used for a pay-at-the-pump application. If the user goes to the same gas station every single time, this implies that for an adversary to extract secret tag information, she must be in a close proximity to the user for about 240 consecutive gas pumping. In the case in which the user goes to different gas stations, this implies that the adversary is following the user everywhere. Both scenarios are highly unlikely to occur in real life applications. In a different example, consider low-cost tags replacing barcodes for identifying grocery items. In such applications, that many authorized protocol runs are unlikely to occur during the entire life time of a low-cost tag. The following corollary is a direct consequence of this remark.

Corollary 1 *In order to expose secret tag information, the adversary must observe a sufficiently high number of honest protocol runs between authorized reader-tag pairs.*

This implies that adversaries, regardless of their computational power, must rely on authorized reader-tag interactions to have a chance of inferring secret information.

Assumption 1 *For the rest of the paper, we will adopt the assumption that observing enough protocol runs to expose the value of a nonce is impractical in low-cost RFID systems.*

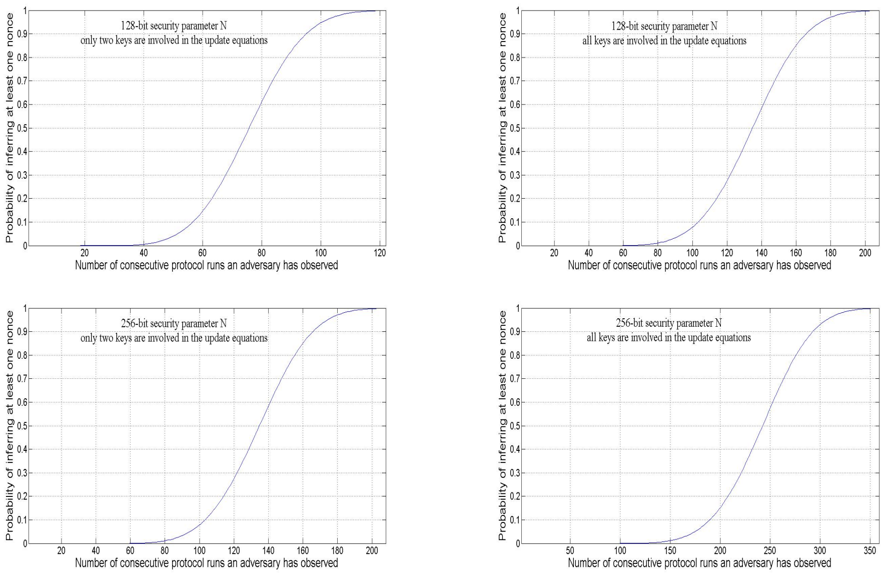


Figure 2. The probability of exposing at least one nonce as a function of the number of consecutive protocol runs observed by the adversary, for different size of security parameter and different number of parameters involved in the update equation.

3.3. Security of Mutual Authentication

Before we can state our main theorem regarding the security of mutual authentication in our protocol, we need two more lemmas.

Lemma 6 *Under Assumptions 1, given that the reader generates random nonces, no information about the secret key, K, is revealed by observing protocol runs of the proposed protocol.*

Proof: We start with the basic assumption that the key is loaded to the tag secretly; that is $k_a^{(0)}$, $k_b^{(0)}$, $k_c^{(0)}$, and $k_d^{(0)}$ are secret. By Lemma 5, messages $B^{(0)}$ and $C^{(0)}$ provide perfect secrecy. That is, no information about the nonce, $n^{(0)}$, nor the keys, $k_b^{(0)}$ and $k_c^{(0)}$, will be leaked by $B^{(0)}$ and $C^{(0)}$. Now, $n^{(0)}$ will be used to generate $D^{(0)} = n_\ell^{(0)} \oplus k_d^{(0)}$ and $A^{(1)} = n_\ell^{(0)} + (n_r^{(0)} \oplus k_a^{(0)})$. Since $n^{(0)}$ is delivered in a perfectly secret manner, and $k_d^{(0)}$ and $k_a^{(0)}$ are secret, no information will be revealed by the observation of $D^{(0)}$ and $A^{(1)}$. (The proof is very similar to the proof of Lemma 5; it is based on the fact that $k_a^{(0)}$ and $k_d^{(0)}$ are random and independent.)

So far, no secret information about the initial key $K^{(0)}$ nor the nonce $n^{(0)}$ has been revealed. Therefore, there is no information leakage about the updated subkeys $k_a^{(1)} = n_r^{(0)} \oplus k_a^{(0)}$, $k_b^{(1)} = k_u^{(0)} + (n^{(0)} \oplus k_b^{(0)})$, $k_c^{(1)} = k_u^{(0)} \times (n^{(0)} \oplus k_c^{(0)})$, and $k_d^{(1)} = n_r^{(0)} \oplus k_d^{(0)}$. Given that the keys are updated to remain independent and to have the same distribution as the outdated keys, and that $n^{(1)}$ is random and independent from the previous nonce and from the secret keys, the proof follows by induction (given Assumption 1). ■

Lemma 7 *Under Assumption 1, an adversary making q_q Query oracles, q_s Send oracles will succeed with probability at most*

$$\max\{\frac{q_q}{p-1}, \frac{q_s}{2^N}\}. \tag{29}$$

Proof: By Lemma 6 and Corollary 1, calling the *Execute* oracle a practical number of consecutive times is of no help to the adversary, since no information is leaked by observing messages exchanged between authorized RFID pairs.

On the other hand, an adversary calling the *Query* oracle will receive A as the tag's response. Depending on the adversary's response, the tag will respond with message D with probability $1/(p-1)$ (the probability of successful forgery by Lemma 3), or abort the protocol with probability $(p-2)/(p-1)$. If the tag does respond, the protocol is considered broken. However, upon unsuccessful forgery, the tag will abort, and responds to the next *Query* with the same identifier, A. Therefore, no information about the tag's secret key is revealed by multiple *Query* calls.

Finally, an adversary calling the *Send* oracle to impersonate a valid tag will be successful with probability at most $1/2^N$. This is due to the fact that A might or might not be a valid tag identifier. If it is not, the reader will abort the protocol. Assume, however, that A is a valid identifier (the adversary can obtain a valid one by interrogating a tag in the system). An authorized reader, responding with B and C, will accept D if and only if $D = n_\ell \oplus k_d$. To extract the correct n_ℓ, however, the adversary must know k_b or k_c, which are kept secret by Lemma 6. Moreover, k_d is unknown to the adversary (also by Lemma 6). Hence, the adversary's probability of success is $1/2^N$, and the lemma follows. ■

Given that p is a $2N$-bit prime integer, the adversary's probability of falsely authenticating herself to a valid tag is at most $1/2^{2N-1}$, and the adversary's probability of authenticating herself to a valid reader is $1/2^N$. That is, the probability of mutual authentication when the protocol is not honest is negligible in the security parameter N. We can now state our main theorem.

Theorem 1 *Under Assumption 1, the proposed UCS-RFID is a secure mutual authentication protocol for RFID systems.*

Proof: Lemma 6 implies that the first condition of Definition 1 is satisfied. The second condition of Definition 1 can be easily verified; it merely means that if the messages exchanged between legitimate RFID pairs are relayed faithfully to one another, mutual authentication is achieved. The third condition of Definition 1 is shown to be satisfied in Lemma 7. Thus, all three conditions of Definition 1 are satisfied. ■

3.4. Desynchronization Attacks and the Update Procedure

It is critical to point out the importance of the key update procedure to the security of our protocol. Both the tag and the reader must update their parameters for the security to hold.

Consider an adversary blocking message A and replaying it to the reader. Since the adversary does not know k_b, k_c nor k_d, she cannot extract the correct n and generate a valid D with a non-negligible probability.

Consider an adversary blocking messages B and C, and replaying them to the tag. Of course, the adversary will be authenticated. However, this is considered as faithfully relaying messages, which does not affect the honesty of the protocol. This makes sense, because the tag will respond with a message D which does not reveal extra information about the tag that has not been revealed by A.

Consider an adversary blocking message D sent to the reader. The reader will assume that the tag has not updated its parameters while, in fact, it has. Consequently, the secret keys at the tag's side will be different than the secret keys stored at the database, causing a possible desynchronization between the tag and the reader. A solution to this problem is that the reader updates the parameters even if it does not receive message D from the tag. The reader, however, must store both the updated and the outdated parameter values at the database to count for the possible scenario that the tag has not updated its parameters.

A more dangerous attack can be launched by blocking messages B and C sent to the tag, or message D sent to the reader, if the *same* keys, with *different* nonces are used. Let $B^{(1)} \equiv n^{(1)} + k_b^{(1)} \mod p$ and $C^{(1)} \equiv n^{(1)} \times k_c^{(1)} \mod p$ be blocked by an active adversary. If the same keys $k_b^{(1)}$ and $k_c^{(1)}$ are used to generate $B^{(2)} \equiv n^{(2)} + k_b^{(1)} \mod p$ and $C^{(2)} \equiv n^{(2)} \times k_c^{(1)} \mod p$, the difference between the two nonces, $n^{(1)}$ and $n^{(2)}$, is simply the difference between $B^{(1)}$ and $B^{(2)}$. It can be easily seen that:

$$\begin{aligned} C^{(2)} &\equiv n^{(2)} \times k_c^{(1)} \equiv (n^{(1)} + \delta) \times k_c^{(1)} \\ &\equiv (n^{(1)} \times k_c^{(1)}) + (\delta \times k_c^{(1)}) \equiv C^{(1)} + (\delta \times k_c^{(1)}) \mod p. \end{aligned} \tag{30}$$

Hence, with the knowledge that $n^{(2)} \equiv n^{(1)} + \delta \mod p$, where $\delta \equiv B^{(2)} - B^{(1)} \mod p$, the value of $k_c^{(1)}$ can be easily computed as $k_c^{(1)} \equiv (C^{(2)} - C^{(1)}) \times \delta^{-1} \mod p$. Thus, we emphasize that whenever the reader receives an outdated identifier, the reader retransmits the same messages $B^{(1)}$ and $C^{(1)}$, as opposed to generating a new nonce and transmitting $B^{(2)}$ and $C^{(2)}$ as above.

The requirement that the reader responds with the same B and C when receiving an outdated A, however, introduces a vulnerability to a man-in-the-middle (MITM) attack. Consider an adversary observing messages $A^{(1)}, B^{(1)}, C^{(1)}$, and then intercepting message $D^{(1)}$. The reader will assume that the tag has not updated its parameter. Hence, the adversary can impersonate the tag by sending its $A^{(1)}$ and, upon receiving the same $B^{(1)}$ and $C^{(1)}$, she can replay the intercepted $D^{(1)}$, which will be accepted by the reader.

Fortunately, there is an easy fix for this vulnerability. Whenever a valid reader receives an outdated identifier $A^{(1)}$, it responds with the same $B^{(1)}$ and $C^{(1)}$ to avoid key exposure (as discussed above). But the tag does *not* get authenticated upon the reception of $D^{(1)}$ (to avoid the man-in-the-middle attack described above). The reader continues by carrying out another protocol run with the tag (with updated keys this time), and *only* if the second authentication run is passed, with updated parameters to generate $A^{(2)}, B^{(2)}, C^{(2)}$, and $D^{(2)}$, the tag is authenticated.

4. Conclusion and Future Work

In this paper, a new direction into the problem of authenticating low-cost RFID systems is proposed. The aim of this paper was to investigate the possibilities of unconditional security in the design of RFID protocols. An instance of such protocols was proposed. Under a restriction on the number of consecutive protocol runs an adversary is assumed to observe, the proposed protocol is shown to achieve unconditional secrecy and unconditional integrity.

References

[1] B. Alomair, L. Lazos, and R. Poovendran. Passive attacks on a class of authentication protocols for RFID. *International Conference on Information Security and Cryptology – ICISC'07*, 2007.

[2] B. Alomair and R. Poovendran. On the Authentication of RFID Systems with Bitwise Operations. *New Technologies, Mobility and Security, 2008. NTMS'08.*, 2008.

[3] G. Avoine. Adversary model for radio frequency identification. Technical report, Swiss Federal Institute of Technology (EPFL), Security and Cryptography Laboratory (LASEC), 2005.

[4] G. Avoine, K. Kalach, and J.-J. Quisquater. ePassport: Securing international contacts with contactless chips. In *Financial Cryptography and Data Security – FC'08*, 2008.

[5] G. Avoine and P. Oechslin. A Scalable and Provably Secure Hash Based RFID Protocol. In *International Workshop on Pervasive Computing and Communication Security – PerSec 2005*, 2005.

[6] M. Bellare and P. Rogaway. Entity authentication and key distribution. In *Advances in Cryptology-CRYPTO'93*, 1993.

[7] A. Bogdanov, G. Leander, C. Paar, A. Poschmann, M. J. Robshaw, and Y. Seurin. Hash Functions and RFID Tags : Mind The Gap. In *Proceedings of the 10th International Workshop Cryptographic Hardware and Embedded Systems, CHES'08*, 2008.

[8] J. Bringer and H. Chabanne. Trusted-HB: A Low-Cost Version of HB$^+$ Secure Against Man-in-the-Middle Attacks. *IEEE Transactions on Information Theory*, 2008.

[9] J. Bringer, H. Chabanne, and D. Emmanuelle. HB^{++}: a Lightweight Authentication Protocol Secure against Some Attacks. In *Security, Privacy and Trust in Pervasive and Ubiquitous Computing-SecPerU'06*, 2006.

[10] J. Bringer, H. Chabanne, and T. Icart. Improved Privacy of the Tree-Based Hash Protocols Using Physically Unclonable Function. In *Security and Cryptography for Networks-SCN'08*, 2008.

[11] T. Cover and J. Thomas. *Elements of Information Theory*. Wiley-Interscience New York, 2006.

[12] S. Dominikus, E. Oswald, and M. Feldhofer. Symmetric Authentication for RFID Systems in Practice. Handout of the Ecrypt Workshop on RFID and Lightweight Crypto, 2005.

[13] M. Feldhofer, S. Dominikus, and J. Wolkerstorfer. Strong authentication for RFID systems using the AES algorithm. In *Workshop on Cryptographic Hardware and Embedded Systems – CHES 2004*, 2004.

[14] M. Feldhofer, J. Wolkerstorfer, and V. Rijmen. AES Implementation on a Grain of Sand. *IEE Proceedings - Information Security*, 2005.

[15] W. Feller. *An Introduction to Probability Theory and its Applications*. Wiley India Pvt. Ltd., 2008.

[16] H. Gilbert, M. Robshaw, and Y. Seurin. HB#: Increasing the Security and Efficiency of HB. 2008.

[17] H. Gilbert, M. Robshaw, and H. Sibert. An active attack against HB$^+$ – a provably secure lightweight authentication protocol. Manuscript, 2005.

[18] O. Goldreich. *Foundations of Cryptography*. Cambridge University Press, 2001.

[19] P. Golle, M. Jakobsson, A. Juels, and P. Syverson. Universal re-encryption for mixnets. In *The Cryptographers' Track at the RSA Conference – CT-RSA*, 2004.

[20] J. Gubner. *Probability and Random Processes for Electrical and Computer Engineers*. Cambridge University Press, 2006.

[21] T. S. Heydt-Benjamin, D. V. Bailey, K. Fu, A. Juels, and T. O'Hare. Vulnerabilities in First-Generation RFID-Enabled Credit Cards. Manuscript, 2006.

[22] A. Juels and R. Pappu. Squealing euros: Privacy protection in RFID-enabled banknotes. In *Financial Cryptography – FC'03*, 2003.

[23] A. Juels and S. Weis. Authenticating pervasive devices with human protocols. In *Advances in Cryptology – CRYPTO'05*, 2005.

[24] J. Lee and Y. Yeom. Efficient RFID Authentication Protocols Based on Pseudorandom Sequence Generators. Cryptology ePrint Archive, Report 2008/343, 2008.

[25] T. Li and R. H. Deng. Vulnerability analysis of EMAP - an efficient RFID mutual authentication protocol. In *Second International Conference on Availability, Reliability and Security – AReS 2007*, 2007.

[26] D. Molnar and D. Wagner. Privacy and Security in Library RFID: Issues, Practices, and Architectures. In *Conference on Computer and Communications Security – ACM CCS*, 2004.

[27] M. O'Neill (McLoone). Low-Cost SHA-1 Hash Function Architecture for RFID Tags. In *Workshop on RFID Security – RFIDSec'08*, 2008.

[28] K. Ouafi, R. Overbeck, and S. Vaudenay. On the Security of HB# against a Man-in-the-Middle Attack. In *Advances in Cryptology - Asiacrypt 2008*, 2008.

[29] P. Peris-Lopez, J. C. Hernandez-Castro, J. Estevez-Tapiador, and A. Ribagorda. LMAP: A real lightweight mutual authentication protocol for low-cost RFID tags. Workshop on RFID Security – RFIDSec'06, 2006.

[30] P. Peris-Lopez, J. C. Hernandez-Castro, J. Estevez-Tapiador, and A. Ribagorda. M2AP: A minimalist mutual-authentication protocol for low-cost RFID tags. In *International Conference on Ubiquitous Intelligence and Computing – UIC06*, 2006.

[31] C. Shannon. *Communication Theory and Secrecy Systems*. Bell Telephone Laboratories, 1949.

[32] D. Stinson. *Cryptography: Theory and Practice*. CRC Press, 2002.

[33] I. Vajda and L. Buttyán. Lightweight authentication protocols for low-cost RFID tags. In *Second Workshop on Security in Ubiquitous Computing – Ubicomp 2003*, 2003.

[34] S. Weis, S. Sarma, R. Rivest, and D. Engels. Security and privacy aspects of low-cost radio frequency identification systems. In *International Conference on Security in Pervasive Computing-SPC'03*, 2003.

A. An Example of Multiplication with Cheap Circuitry

For completeness of presentation, we show here a simple algorithm that multiplies two numbers with minimum circuitry, in expense of time efficiency.

To multiply two N-bit integers, the two operands are stored in two registers R_1 and R_2, while a third register R_3, used to store a temporary partial sum, is initialized to zero. The algorithm starts by examining the least significant bit of R_2; if it is one, then R_1 is added to R_3. The register R_1 is then shifted one bit to the left, which corresponds to multiplying the operand stored in R_1 by two. If the second least significant bit of R_2 is one, then R_1 is added to R_3 and then R_1 is shifted. If it is zero, then the bits in R_1 are shifted one bit to the left without addition. After the $(i-1)^{th}$ shift of R_1, the i^{th} least significant bit of R_2 is examined; if it is one, then the value in the register R_1 is added to the temporary sum in register R_3, and so forth, until the bits in R_1 are shifted N times. The value stored in R_3 is the result of multiplying the two integers modulo p.

The addition algorithm `ADD` can be any modular adder. The multiplication algorithm `MULT` is shown in Algorithm 1.

Algorithm 1 `MULT`(a, b)

```
$R_1 \leftarrow a$;
$R_2 \leftarrow b$;
$R_3 \leftarrow 0$;
for $i = 0, ..., N-1$ do
  if $R_2[i] = 1$ then
    $R_3 \leftarrow$ ADD$(R_3, R_1)$;
  end if
  $R_1 \leftarrow$ ShiftLeft$(R_1)$;
end for
return $R_3$
```

Therefore, by implementing algorithm `MULT` described above, multiplication of two elements of the field $\mathbb{Z}_p$ can be performed using only three N-bit registers and one modular adder.