Tree-Based RFID Authentication Protocols Are Definitively Not Privacy-Friendly

Gildas Avoine, Benjamin Martin, and Tania Martin

Université catholique de Louvain
Information Security Group
B-1348 Louvain-La-Neuve, Belgium
{gildas.avoine, benjamin.martin, tania.martin}@uclouvain.be

Abstract. Authentication for low-cost Radio-Frequency IDentification (RFID) is a booming research topic. The challenge is to develop secure protocols using lightweight cryptography, yet ensuring privacy. A current trend is to design such protocols upon the *Learning Parity from Noise* (LPN) problem. The first who introduced this solution were Hopper and Blum in 2001. Since then, many protocols have been designed, especially the protocol of Halevi, Saxena, and Halevi (HSH) [15] that combines LPN and the tree-based key infrastructure suggested by Molnar and Wagner [24]. In this paper, we introduce a new RFID authentication protocol that is less resource consuming than HSH, relying on the same adversary model and security level, though. Afterwards, we show that, if an adversary can tamper with some tags, the privacy claimed in HSH is defeated. In other words, either tags are tamper-resistant, then we suggest a protocol more efficient than HSH, or they are not, then we suggest a significative attack against the untraceability property of HSH.

Key words: RFID, Security, Traceability, Authentication, HB, Tree-based, LPN.

1 Introduction

Radio Frequency IDentification (RFID) is a wireless technology that allows to identify/authenticate items without physical contact. An RFID interaction is proceeded between: (i) RFID tags, or transponders jointed with an antenna, embedded into objects such as access cards [33], books [24] or even electronic passports [27] and, (ii) RFID readers composed of a transreceiver securely connected to a back-end system.

RFID tags are divided into two categories, passive or active devices: active tags have their own power source (a battery), whereas passive tags draw their energy from the electromagnetic field of the reader. The latter tags so suffer from very limited resources, especially in terms of computation and memory. In secure applications where cryptography is required, tags (even passive) can embed a microprocessor. This solution being expensive, researchers got interested in building *lightweight* cryptographic building blocks that can be implemented with wired logic only. Many such protocols have been published so far, but many attacks have been put forward as well. For many protocols [11, 18, 28–30], active and passive attacks were published [1, 5, 6, 10, 21, 22, 32, 34] while for other protocols [19, 31] only active attacks were found [7, 12].

In 2000, Hopper and Blum [17, 18] took benefit of the *Learning Parity from Noise* (LPN) problem to design a human-to-computer authentication protocol, today known as HB. Juels and Weis [19] then noticed the link between the human-to-computer and tag-to-reader authentication paradigms: the computation capabilities of the provers are quite restricted in both cases. They also stress that HB presents the noticeable particularity that it does not identify the prover who is involved in the protocol. This may be an interesting feature to protect the privacy of the prover, but this becomes a drawback when considering radio frequency *identification*. The idea of using the LPN problem to design lightweight authentication protocols was then taken up in several papers [13, 15, 19] leading to the HB-saga.

During RFIDSec 2009, Halevi, Saxena, and Halevi [15] presented a lightweight privacy-friendly authentication protocol that aims to reduce the reader computational load. The protocol consists of two phases. The first phase identifies the tag using a tree traversal, as suggested by Molnar and Wagner [24] at ACM CCS 2004. This technique allows the reader to retrieve in the database the key associated to the tag with a computation complexity $O(\log(n))$ instead of $O(n)$, where n is the number of tags in the system. In the second phase, the tag is authenticated using the HB+ protocol proposed by Juels and Weis [19] at Crypto 2005.

Our first contribution is a new LPN-based authentication protocol where the computation complexity of the reader is lower than those implied by HSH [15]. Nevertheless, our protocol complies with the privacy threat model considered in [15] and reaches the same security level. Another attractive property is that our protocol also reduces the memory requirement for the tag.

Our second contribution is related to the privacy model considered in [15]. This model considers that tags are tamper-resistant, which is quite a strong assumption. We demonstrate that if this assumption is relaxed, as it is commonly admitted in the literature [2, 3], tampering with one or few tags threatens all the tags belonging to the system. Our attack generalizes the one presented by Avoine, Dysli, and Oechslin [4] at SAC 2005 against the protocol of Molnar and Wagner [24].

The paper is organized as follows: Section 2 provides the background and introduces HSH. Section 3 presents our protocol, and compares it with HSH. Section 4 points out a privacy attack against HSH. Finally, Section 5 concludes.

2 From LPN to HSH

2.1 The LPN Problem

The *Learn Parity with Noise* problem is one of the well-known problems in cryptography. Given that:

- x is a secret k-bit vector,
- a is a random known k-bit vector,
- $\epsilon \in]0, \frac{1}{2}[$ is a noise parameter,
- and η is a bit noise where $\Pr(\{\eta = 1\}) = \epsilon$,

then it is hard to recover x from the result $r = a \cdot x \oplus \eta$ (the scalar product of a and x, XORed with η).

Many attempts on identification and authentication protocols relying on the LPN problem have been proposed so far, such as all HB-family protocols [7, 8, 13, 15–20, 23, 25, 35], or the LPN-C protocol of Gilbert, Robshaw and Seurin [14]. During the sequel, we consider that the RFID systems are composed of n tags.

2.2 The HB+ Protocol

The HB+ protocol has been proposed by Juels and Weis in 2005 [19] to improve the original HB protocol [18] against active attacks.

At the system setup, each tag T has a unique pair of secret keys (x_T, y_T) known by every reader R, where $|x_T| = |y_T| = k$. T is also given a random noise parameter $\epsilon \in]0, \frac{1}{2}[$.

Then s rounds of challenge/response are required by the reader to authenticate the tag T (see Fig. 1), where s is a security parameter. For each round, R selects a random k-bit vector a and sends it to T. The latter also chooses a random k-bit vector b and noise bit η, and sends to R its answer $r = (a \cdot x_T) \oplus (b \cdot y_T) \oplus \eta$. R accepts the round if $(a \cdot x_T) \oplus (b \cdot y_T) = r$. Finally, the reader authenticates the tag T after s rounds if T's answers are correct in more than $s\epsilon$ rounds.

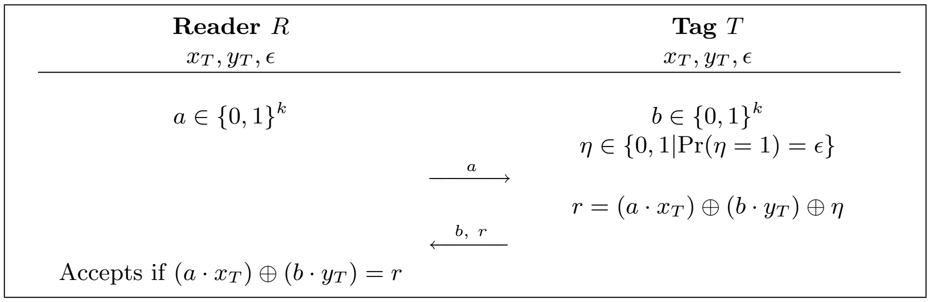
Reader R
x_T, y_T, ϵ
Tag T
x_T, y_T, ϵ
$a \in \{0,1\}^k$
$b \in \{0,1\}^k$
$\eta \in \{0, 1 | \Pr(\eta = 1) = \epsilon\}$
$\xrightarrow{a}$
$r = (a \cdot x_T) \oplus (b \cdot y_T) \oplus \eta$
$\xleftarrow{b,\ r}$
Accepts if $(a \cdot x_T) \oplus (b \cdot y_T) = r$

Fig. 1. A single round of the HB+ protocol.

2.3 The HSH Protocol

In this section, we present the authentication protocol proposed by Halevi, Saxena, and Halevi at RFIDSec 2009 [15] designed for radio-frequency applications.

During the rest of the paper, it will be denoted by HSH. It is claimed to be light and fast, and to preserve tag privacy under the model provided below.

The heart of HSH is that all its design relies on the HB-family protocols combined with the tree-based key infrastructure proposed by Molnar and Wagner in [24] (called here MW). HSH is divided in two stages: the tree traversal and the authentication. The following table gives the notations and recommended values for HSH:

Notation	Meaning	Recommended Values [15]
d	depth of the tree	$d \in \{2, 3\}$
β	tree branching factor	$\beta \in \{100, 1000\}$
k_x	length of the key x_T	$k_x = 80$
k_y	length of the key y_T	$k_y \in [224, 512]$
s	size of T's answer	$s \in [80, 212]$
ϵ	noise level	$\epsilon \in [\frac{1}{8}, \frac{1}{4}]$

The choice of HB-family comes from the fact that these kinds of protocols fit perfectly in low-cost RFID tags. Here we present HSH using HB+ (this choice is given by the HSH authors). Then the idea of MW is to consider the n tags of the system as leaves in a tree of branching factor β. Thus MW associates each edge in the tree with a secret key. The readers are assumed to know all the keys. Each tag stores the $\lceil \log n \rceil$ keys corresponding to its path from the root to the leaf. HSH builds the key infrastructure of the tree traversal stage on MW because it reduces significantly the complexity of the reader during the identification process: $\beta \lceil \log_\beta n \rceil$ operations in the worst case, instead of n. Thus the use of MW makes the protocol lighter on the reader side.

The HSH privacy threat model. Since all HB-like protocols are not resistant against active adversaries, [15] considers adversaries who can eavesdrop all the communications and who can interact with both R and T, but disregarding man-in-the-middle attacks. Also, since the MW key infrastructure does not resist to privacy traceability when several tags are compromised, [15] considers that the tags are tamper-resistant.

Initialization. Given a system with n tags, the parameters β and d are chosen at the system setup, such that they define a tree with $\beta^d \geq n$ leaves. Each leaf is associated randomly to a tag of the system. During the setup of the system, each tag T is assigned to a unique pair of secret keys (x_T, y_T), known by every reader R involved in the system.

Let $p_0, p_1, p_2, \ldots, p_d$ be the path in the tree from the root (denoted p_0) to the leaf that is associated to the tag T (denoted p_d). For each node p_i (except the root), T knows a corresponding k_y-bit key y_{p_i}. R knows the entire tree arrangement, and thus all the keys associated to the nodes.

Tree traversal stage. First of all, R must recover the right pair of secret keys to authenticate correctly the tag T. To do so, T chooses an $s \times k_y$ random binary matrix B and a s-bit random noise vector ν^i for every level i in the tree. Then T computes $z^i = B \cdot y_{p_i} \oplus \nu^i$; and it sends B and z^i for every level i of the tree to R.

Upon reception of these data, R goes down into the tree, node by node, using the z^i's. Clearly, R computes for every child c of the root: $z^c = B \cdot y_c$, where y_c is a child key. And R goes down to the child whose answer is the closest to the data z^1 sent by T. The same procedure is iterated for every level i in the tree.

At the end, R reaches one of the leaves of the tree, and uses the pair of corresponding secret keys (x, y) for the authentication stage.

Authentication. At the end of the tree traversal stage, R is convinced that the pair of keys found is the correct pair for the tag T.

Then R processes the HB+ protocol on this pair to confirm this result and to authenticate definitely T. Thus, the reader sends an $s \times k_x$ challenge binary matrix A to the tag T. The latter then chooses a s-bit random noise vector ν, and sends back the result $z = A \cdot x_T \oplus B \cdot y_T \oplus \nu$. R computes $z' = A \cdot x \oplus B \cdot y$ with the pair found at the end of the tree traversal stage, and computes the Hamming distance between z' and T's answer z. If this value is under the threshold $\tau = s\epsilon$, then R accepts T; otherwise it rejects it.

3 Our Protocol

In this section, we present an authentication protocol relying on the LPN problem. The goal of this protocol is to be as secure as HSH in the same threat model (see Section 2.3 for its definition). In such a case, we want to prove that, in the same weak threat model, it is possible to create a protocol with less needed tag memory and less computational complexity, especially on the reader side.

3.1 Problem Statement

As explained before, HSH is tree-based which leads to a $O(\log n)$ reader complexity, nevertheless better than a classical challenge/response protocol whose reader complexity is in $O(n)$. Since tags are tamper-resistant, we decide to put a unique pair of symmetric keys (x, y) shared between all readers and tags, in order to decrease R's complexity. Thus, having a common pair of keys for the whole system is better for R's computation search, rather than n pairs (one unique per tag in classical cryptography): R's complexity search will be in $O(1)$.

In a classical HB-family protocol, each tag T has a unique symmetric secret key (or pair of keys) to authenticate itself to the reader R. During the protocol execution, T adds some noise to its answer (with some probability). Then R tries every tags' secret key and, when it finds a result enough close to T's answer (with respect to the noise probability), the authentication succeeds. Basically, we consider that R scans its database of all tags' secret keys and stops when it finds such a match: it is a problem since R does not try all the secret keys to find the one whose computation will be the closest to T's answer, but the first one which is close to T's answer under the probability parameter ϵ. So the HB-family protocols provide tag authentication, but R will not be sure of the real tag's identity. That is the reason we associate a unique secret identifier Id_T per tag, which is sent into the tag's answer to be identified by R. The latter knows all the tags' identifiers.

Thus, in order to merge all these properties, we present a variant of the LPN-C protocol proposed by Gilbert, Robshaw and Seurin in [14]. Actually, since the latter is vulnerable to replay attacks, our authentication protocol has to thwart such attacks. In our proposal, the tag's answer is built in the same way as for HB+ [19] (see Section 2.2). In comparison with LPN-C, we first decide to add a challenge sent by the reader to the tag: this is to avoid the problem of replay attacks that are inherent in LPN-C. Then in our protocol, the challenges are matrix whereas the secrets are vectors, and these latters are defined as in HB+, i.e. two secret keys instead of one: this choice is to store less information on the tag, to reduce the tag memory needed by the protocol and thus to reduce the potential price of the tag (*i.e.* less memory means lower costly tag). The final achievement of our protocol is to allow a reader to authenticate and identify a tag correctly, based on the following hypothesis :

- all tags and readers share a common pair of keys,
- every tag has a unique secret identifier.

3.2 The Protocol Description

Initialization. When the system is set up, every reader and tag share a pair (x, y) of secret keys. Each tag T is assigned with a unique secret identifier Id_T known by R. The notations and values that will be used in the protocol are given below:

Notation	Meaning	Usual choices
k_x	length of the key x	$k_x = 80$
k_y	length of the key y	$k_y \in [224, 512]$
s	length of T's identifier Id_T	$s \in [80, 212]$
ϵ	noise level	$\epsilon \in [\frac{1}{8}, \frac{1}{4}]$

We define C as the code of all the tags' identifiers of our system. For a given codeword $Id_T \in C$, we consider the ball $\mathcal{B}_{Id_T}$ of radius $t = \lceil s\epsilon \rceil$ around Id_T. Each ball represents all the codewords c such that $\mathrm{d}_{\mathcal{H}}(Id_T, c) \leq t$, where $\mathrm{d}_{\mathcal{H}}$ denotes the Hamming distance. The volume of $\mathcal{B}_{Id_T}$ is the number of all these codewords c, defined as: $\mathrm{Vol}(\mathcal{B}_{Id_T}) = \displaystyle\sum_{i=0}^{t} \binom{s}{i}$. To make viable the tag identification, we distribute the identifiers such that all the balls are pairwise disjoint.

Authentication. The authentication protocol consists of three steps (see Fig. 2):

(1) The reader sends a random binary $s \times k_x$ matrix A.

(2) The tag chooses a random binary $s \times k_y$ matrix B, a s-bit random noise vector $\nu = (\nu_1, \nu_2, \ldots, \nu_s)$.
$\forall i \in \{1, \ldots, s\} : \nu_i \in \{0, 1 | \Pr(\{\nu_i = 1\}) = \epsilon\}$.
Then it computes $r = (A \cdot x) \oplus (B \cdot y) \oplus Id_T \oplus \nu$, where $|r| = s$.
Finally, it sends B and r to R.

(3) The reader computes $r' = (A \cdot x) \oplus (B \cdot y)$, and recovers instantaneously $D = r \oplus r' = Id_T \oplus \nu$.
Then for each tag identifier I, R computes the Hamming distance between D and I. Since all the tags identifiers are well-distributed, when this distance is lower than t, that means D only belongs to $\mathcal{B}_I$. Thus R retrieves the real identifier $I = Id_T$.

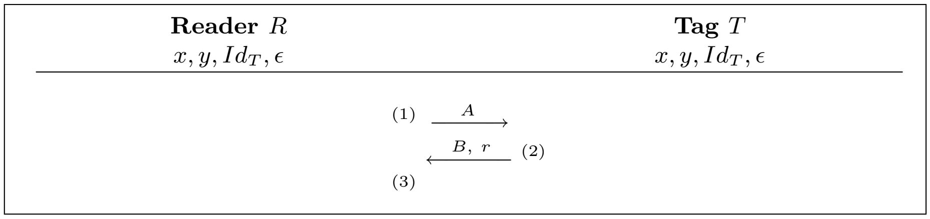

Fig. 2. Authentication protocol.

Remark 1. The step (3) of the authentication process can be improved. R must recover T's identifier from D. Clearly, D is the tag's identifier XORed with some noise vector ν, i.e. Id_T containing at most t error bits (t being the Hamming weight of ν). Instead of computing naively the Hamming distance between D and I, R can use an appropriate error-correcting code to recover Id_T without the t errors. This extension is out of the scope of this paper, though.

3.3 Analysis

Besides the assumption that all the balls $\mathcal{B}_{Id_T}$ are pairwise disjoint, we assume that (i) the identifiers space is large enough and, (ii) the tags' identifiers are uniformly distributed for security reasons. First, the distance between two identifiers must be at least two times the radius of a ball, i.e. $2t$. This will allow the reader to identify without mistakes every tag, since every D result will belong to a unique ball. But if the identifiers space is too small and if all the balls cover exactly the space, the security is nonexistent: an adversary can send a value at random and be sure to be identified by the reader. We thus want that the success probability of an adversary to send a random value that could match a result into a ball is negligible. That is why we choose an identifiers space large enough.

We compare our results to the ones given by the HSH protocol. If we take $n = 10^6$ tags, we have the following results when ϵ, d, β, k_x and k_y are fixed:

	FAR	s	Tag mem	Comm	$\mathbb{C}_T$	$\mathbb{C}_R$
HSH	$2^{-41.3}$	86	1400	44978	$2^{16.9}$	$2^{26.2}$
Our protocol	$2^{-41.5}$	128	648	66688	2^{16}	

- $n = \beta^d = 10^6$ = total number of tags in the system,
- $d = 2$ = tree depth for the HSH protocol,
- $\beta = 1000$ = tree branching factor for the HSH protocol,
- $\epsilon = 0.125$ = noise level,
- $k_x = 80$ = length of the key x,
- $k_y = 440$ = length of the key y,
- s = length of T's identifier/response,
- FAR = False Accept Rate = probability of guessing a tag authentication reply at random,
  - $\text{FAR} = n\dfrac{\text{Vol}(\mathcal{B}(0,t))}{2^s}$ for our protocol,
  - $\text{FAR} = \dfrac{\text{Vol}(\mathcal{B}(0,t))}{2^s}$ for HSH,
- Tag mem = the memory needed on the tag,
  - $k_x + k_y + s$ bits for our protocol,
  - $k_x + k_y(d + 1)$ bits for HSH,
- Comm = total number of bits sent during the whole protocol,
  - $s(k_x + k_y + 1)$ bits for our protocol,
  - $s(k_x + k_y + d + 1)$ bits for HSH,
- $\mathbb{C}_T$ = tag computation complexity,
  - $s(k_x + k_y)$ bit operations for our protocol,
  - $sdk_y + s(k_x + k_y)$ bit operations for HSH,
- $\mathbb{C}_R$ = reader computation complexity,
  - $s(k_x + k_y)$ bit operations for our protocol (+ decoding $D = Id_T \oplus \nu$),
  - $\beta sdk_y + s(k_x + k_y)$ bit operations for HSH.

Here, we augment s to reach the same security level of HSH for our protocol (i.e. FAR $\approx 2^{-41}$). Thus we notice that our protocol needs less memory in the tag to achieve the same security level (around half less). The complexity for the reader and for the tag to process the protocol is also much lower than for HSH. This conclusion is further observable at the reader side: the tree traversal stage of HSH done by the reader to find the right pair of secret keys increases consequently the reader's time search.

4 Attack on HSH

In this section, we introduce an attack conducted on the HSH protocol to damage the tag privacy. First, let remind that HSH is a combination of HB+ and the tree-based key infrastructure of Molnar and Wagner (MW). Several attacks have already demonstrated that MW does not provide tag privacy: Avoine, Dysli, and Oechslin in [4] (called here ADO), then Buttyán, Holczer and Vajda in [9], and finally Nohl and Evans in [26]. HSH naturally inherits from MW's weaknesses.

In what follows, we consider ADO being the first published attack on MW, and show that it can be adapted to break the untraceability of HSH.

In Section 3, we assumed that tags are tamper-resistant, and we provided a protocol better than HSH under this assumption. In this section, we consider that tags are not tamper-resistant, and we show that compromising a few tags smashes the whole system when based on HSH. The details of all the probabilities computations done in this section are given in the appendix.

4.1 Adversary game

We explain here how the tree technique chosen in HSH is predisposed for tag tracing when the adversary $\mathcal{A}$ can tamper with one tag. We then show that the situation is still worse when $\mathcal{A}$ can tamper with several tags. Following ADO [4], we consider a *Challenger* that supplies two tags to the adversary, one of them being the target tag. The attack is done as follows:

1. $\mathcal{A}$ requests the *Challenger* and receives one tag T_0 (respectively several tags) that she can tamper with. Thus $\mathcal{A}$ can obtain all T_0 (respectively the tampered tags) keys. Since the number of tags in the system is large enough, we consider for the sake of simplicity that $\mathcal{A}$ is allowed to put T_0 (respectively all the tampered tags) back into circulation.
2. Then $\mathcal{A}$ requests the *Challenger* and receives a target tag T that she can query as much as she wants, but cannot tamper with it. Next $\mathcal{A}$ puts T back into circulation.
3. $\mathcal{A}$ requests the *Challenger* and receives two tags T_1 and T_2 such that $T \in \{T_1, T_2\}$. She can query T_1 and T_2 as much as she wants, without tampering with them.
4. Finally, $\mathcal{A}$ outputs $T = T_1$ or T_2.

$\mathcal{A}$ succeeds if she can guess correctly which one of T_1 and T_2 is the target tag T.

Notice that in our attack, $\mathcal{A}$ always provides an answer. The goal of our study is to compute $\mathcal{A}$'s advantage.

4.2 Tampering with one tag

The purpose of this section is to evaluate the probability that the attack described above succeeds. To formalize the analysis, we denote the keys of T, T_0, T_1 and T_2 by $[y_1, \ldots, y_d]$, $[y_1^0, \ldots, y_d^0]$, $[y_1^1, \ldots, y_d^1]$ and $[y_1^2, \ldots, y_d^2]$ respectively. At a given level i in the tree ($1 \leq i \leq d$), the ADO attack considers four possibilities:

- $C_i^1 = \{y_i^0 = y_i^1\} \wedge \{y_i^0 \neq y_i^2\}$,
- $C_i^2 = \{y_i^0 \neq y_i^1\} \wedge \{y_i^0 = y_i^2\}$,
- $C_i^3 = \{y_i^0 \neq y_i^1\} \wedge \{y_i^0 \neq y_i^2\}$,
- $C_i^4 = \{y_i^0 = y_i^1 = y_i^2\}$.

MW is based on a classical challenge/response protocol using pseudo-random functions. But HSH is based on a challenge/response protocol using HB+. The noise inherent to HB+ does not allow to apply the ADO attack directly.

We first define $z_i^\ell = B^\ell y_i^\ell \oplus \nu_i^\ell$ being the answer of the tag T_ℓ at level i of the HSH tree traversal stage. Then, we define $\mathcal{B}_i^\ell$ being the ball of radius $t = \lceil s\epsilon \rceil$ (as defined in Section 3.3) around z_i^ℓ. A direct consequence of HSH is that the adversary can only evaluate the possibility that T_0's key y_i^0 was used to compute z_i^ℓ, i.e. $B^\ell y_i^0 \in \mathcal{B}_i^\ell$ or not. With these notations, we consider for our attack four possibilities related to the ADO ones :

- $A_i^1 = \{B^1 y_i^0 \in \mathcal{B}_i^1\} \wedge \{B^2 y_i^0 \notin \mathcal{B}_i^2\}$,
- $A_i^2 = \{B^1 y_i^0 \notin \mathcal{B}_i^1\} \wedge \{B^2 y_i^0 \in \mathcal{B}_i^2\}$,
- $A_i^3 = \{B^1 y_i^0 \notin \mathcal{B}_i^1\} \wedge \{B^2 y_i^0 \notin \mathcal{B}_i^2\}$,
- $A_i^4 = \{B^1 y_i^0 \in \mathcal{B}_i^1\} \wedge \{B^2 y_i^0 \in \mathcal{B}_i^2\}$.

Clearly here, the events that are taken into account for this attack are:

- $E_i^1 = C_i^1 \wedge A_i^1$ then the attack succeeds,
- $E_i^2 = C_i^2 \wedge A_i^2$ then the attack succeeds,
- $E_i^3 = C_i^3 \wedge A_i^3$ then the attack definitely fails,
- $E_i^4 = C_i^4 \wedge A_i^4$ then the attack fails at level i but can move to level $i+1$,

where all the E_i^j events are pairwise disjoint. The fact that the attack succeeds means that the adversary has been able to distinguish T_1 from T_2.

In order to simplify the notation in the following, we give and denote explicitly two probabilities to compare T_0's key y_i^0 and a given T_ℓ's key y_i^ℓ at level i:

- $\Pr(\{B^\ell y_i^0 \in \mathcal{B}_i^\ell\} | \{y_i^0 = y_i^\ell\}) = \sum_{j=0}^{t} \binom{s}{j} \epsilon^j (1-\epsilon)^{s-j} = S_t$,
- $\Pr(\{B^\ell y_i^0 \in \mathcal{B}_i^\ell\} | \{y_i^0 \neq y_i^\ell\}) = \Pr(\mathrm{d}_{\mathcal{H}}(B^\ell y_i^0, z_i^\ell) \leq t) = \frac{\mathrm{Vol}(\mathcal{B}(0,t))}{2^s} = V_t$.

We compute the probabilities of the events E_i^1, E_i^2, E_i^3, and E_i^4. The final results are:

$$\Pr(E_i^1) = \Pr(E_i^2) = S_t(1 - V_t)\left(\frac{\beta - 1}{\beta^2}\right)$$
$$\Pr(E_i^3) = (1 - V_t)^2 \left(\frac{\beta - 1}{\beta}\right)^2$$
$$\Pr(E_i^4) = \left(\frac{S_t}{\beta}\right)^2$$

Following ADO attack, the overall probability P_{succ} that the whole attack succeeds when the adversary tampers with one tag is:

$$P_{\text{succ}} = 2S_t(1 - V_t)\left(\frac{\beta - 1}{\beta^2}\right)\left(\frac{1 - (\frac{S_t}{\beta})^{2d}}{1 - (\frac{S_t}{\beta})^2}\right)$$

4.3 Adversary probability P_{succ} when tampering with several tags

We now analyze the adversary success probability of tracing a tag when she tampers with c_0 tags ($c_0 \geq 1$). As before, we denote the keys of T, T_1 and T_2 by $[y_1, \ldots, y_d]$, $[y_1^1, \ldots, y_d^1]$ and $[y_1^2, \ldots, y_d^2]$ respectively. Then at a given level i of the tree, we consider $\mathcal{K}_i = \{k_{i,1}, k_{i,2}, \ldots, k_{i,c_i}\}$ the set of keys known by the adversary (with $\mathrm{Card}(\mathcal{K}_i) = c_i$). The ADO attack considers five possibilities:

- $C_i^1 = \{y_i^1 \in \mathcal{K}_i\} \wedge \{y_i^2 \notin \mathcal{K}_i\}$,
- $C_i^2 = \{y_i^1 \notin \mathcal{K}_i\} \wedge \{y_i^2 \in \mathcal{K}_i\}$,
- $C_i^3 = \{y_i^1 \in \mathcal{K}_i\} \wedge \{y_i^2 \in \mathcal{K}_i\} \wedge \{y_i^1 \neq y_i^2\}$,
- $C_i^4 = \{y_i^1 \notin \mathcal{K}_i\} \wedge \{y_i^2 \notin \mathcal{K}_i\}$,
- $C_i^5 = \{y_i^1 \in \mathcal{K}_i\} \wedge \{y_i^2 \in \mathcal{K}_i\} \wedge \{y_i^1 = y_i^2\}$.

In the same vein as the previous section, we define five possibilities related to ADO ones:

- $A_i^1 = \{\exists k_{i,m} \in \mathcal{K}_i : B^1 k_{i,m} \in \mathcal{B}_i^1\} \wedge \{\forall k_{i,m} \in \mathcal{K}_i : B^2 k_{i,m} \notin \mathcal{B}_i^2\}$,
- $A_i^2 = \{\forall k_{i,m} \in \mathcal{K}_i : B^1 k_{i,m} \notin \mathcal{B}_i^1\} \wedge \{\exists k_{i,m} \in \mathcal{K}_i : B^2 k_{i,m} \in \mathcal{B}_i^2\}$,
- $A_i^3 = \{\exists k_{i,m} \in \mathcal{K}_i : B^1 k_{i,m} \in \mathcal{B}_i^1\} \wedge \{\exists k_{i,m'} \in \mathcal{K}_i : B^2 k_{i,m'} \in \mathcal{B}_i^2\} \wedge \{k_{i,m} \neq k_{i,m'}\}$,
- $A_i^4 = \{\forall k_{i,m} \in \mathcal{K}_i : B^1 k_{i,m} \notin \mathcal{B}_i^1\} \wedge \{\forall k_{i,m} \in \mathcal{K}_i : B^2 k_{i,m} \notin \mathcal{B}_i^2\}$,
- $A_i^5 = \{\exists k_{i,m} \in \mathcal{K}_i : B^1 k_{i,m} \in \mathcal{B}_i^1\} \wedge \{\exists k_{i,m'} \in \mathcal{K}_i : B^2 k_{i,m'} \in \mathcal{B}_i^2\} \wedge \{k_{i,m} = k_{i,m'}\}$.

Then the events that are taken into account for this attack are:

- $E_i^1 = C_i^1 \wedge A_i^1$ then the attack succeeds,
- $E_i^2 = C_i^2 \wedge A_i^2$ then the attack succeeds,
- $E_i^3 = C_i^3 \wedge A_i^3$ then the attack succeeds,
- $E_i^4 = C_i^4 \wedge A_i^4$ then the attack definitely fails,
- $E_i^5 = C_i^5 \wedge A_i^5$ then the attack fails at level i but can move to level $i+1$.

We compute the probabilities of the events $E_i^1, E_i^2, E_i^3, E_i^4$, and E_i^5 at level i:

$$\Pr(E_i^1) = \Pr(E_i^2) = \left(\frac{c_i(\beta - c_i)}{\beta^2}\right)\left(1 - (1 - S_t)^{c_i}\right)(1 - V_t)^{c_i}$$

$$\Pr(E_i^3) = \left(\frac{c_i(c_i - 1)}{\beta^2}\right)\left(1 - (1 - S_t)^{c_i}\right)^2$$

$$\Pr(E_i^4) = \left(\frac{\beta - c_i}{\beta}\right)^2 (1 - V_t)^{2c_i}$$

$$\Pr(E_i^5) = \left(\frac{c_i}{\beta^2}\right)\left(1 - (1 - S_t)^{c_i}\right)^2$$

Following ADO attack, the overall probability P_{succ} that the whole attack succeeds when the adversary tampers with c_0 tags is:

$$P_{\text{succ}} = \Pr(E_1^1 \vee E_1^2 \vee E_1^3) + \sum_{i=2}^{d}\left(\Pr(E_i^1 \vee E_i^2 \vee E_i^3) \times \prod_{j=1}^{i-1} \Pr(E_j^5)\right) \qquad (1)$$

where c_i, the number of different keys known by the adversary at level i, is given by the ADO attack:

$$c_1 = \beta\left(1 - \left(\frac{\beta-1}{\beta}\right)^{c_0}\right) \quad \text{and} \quad c_i = \beta\left(1 - \left(\frac{\beta-1}{\beta}\right)^{g(c_i)}\right) \quad \forall i,\ 2 \leq i \leq d$$
$$\text{where} \quad g(c_i) = c_0 \prod_{\ell=1}^{i-1} \frac{1}{c_\ell}$$

Remark 2. When $\epsilon = 0$, there is a perfect match between the ADO attack and ours. In such a case, there is not wanted noise added in the tag's answer, which influences the values of S_t and V_t ($S_t = 1$ and $V_t = 0$).

Table 1 gives numerical values of Eq. 1 when the parameters s and ϵ are fixed ($s = 86$ and $\epsilon = 0.125$).

Table 1. Numerical values of the probability P_{succ} of tracing tag T according to branching factor β when the adversary tampers with c_0 tags. The system contains 2^{20} tags, $s = 86$ and $\epsilon = 0.125$.

β c_0	2	20	100	500	1000
1	33.7%	5.8%	1.2%	0.2%	0.1%
20	56.1%	84.3%	32.9%	7.7%	3.9%
50	56.3%	95.8%	63.0%	18.1%	9.5%
100	56.3%	96.9%	86.0%	33.0%	18.1%
200	56.3%	98.1%	97.4%	55.0%	33.0%

4.4 Adversary probability P_{luck} when tampering with several tags

Contrary to MW where the adversary can always determine with probability 1 that a given key has been used to generate a tag's answer, HSH does not provide such a deterministic verification procedure. In other words, the adversary can be unlucky, meaning that she checks the right key but concludes that the key is wrong due to the noise; but she can also be lucky, meaning that the noise makes her observe something wrong but, nevertheless, she provides the correct result.

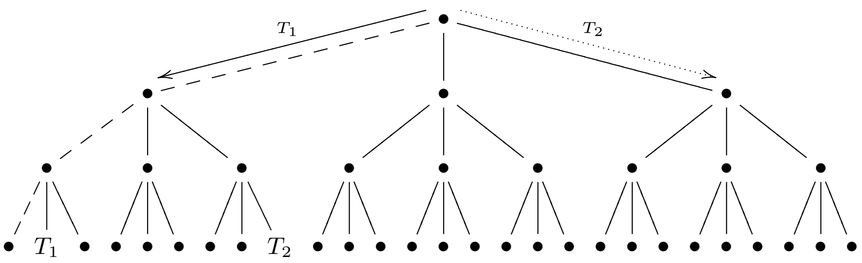

Fig. 3. An example of the event B.

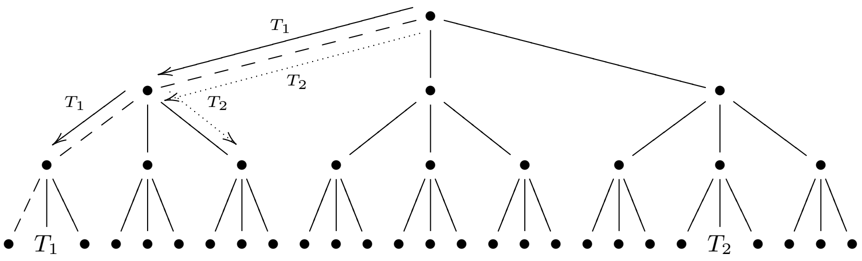

Fig. 4. An example of the event L.

P_{luck} reflects the fact that an adversary $\mathcal{A}$ is lucky and finds the right answer even if she makes a mistake. It is divided in two events during the tree traversal:

- B = $\{\mathcal{A}$ separates too soon T_1 and T_2, while this can be done later$\}$,
- L = $\{\mathcal{A}$ separates too late T_1 and $T_2\}$,

which define the adversary probability of luck as:

$$P_{\text{luck}} = \Pr(\text{B}) + \Pr(\text{L}) \qquad \text{where } B \wedge L = \emptyset \tag{2}$$

Figures 3 and 4 are illustrations of the events B and L. As legend, the branches – – – represent the paths compromised; $\xleftarrow{T_1}$ and $\xleftarrow{T_2}$ represent the paths supposed by $\mathcal{A}$ for T_1 and T_2, respectively.

Below are the formulas for Pr(B) and Pr(L). Notice that $\Pr(E_0^5) = 1$.

$$\Pr(\text{B}) = \sum_{i=1}^{d-2}\left(\Pr(\text{B-Separation}_i) \times \prod_{j=0}^{i-1} \Pr(E_j^5)\right)$$

$$\Pr(\text{L}) = \sum_{i=2}^{d}\left(\Pr(\text{L-Separation}_i) \times \sum_{k=1}^{i-1}\left(\prod_{j=0}^{k-1} \Pr(E_j^5) \times \prod_{\substack{\ell=i-1\\ \ell=\ell-1}}^{k} \Pr(\text{L-Follow}_\ell)\right)\right)$$

Table 2 gives numerical values of Eq. 2 when the parameters s and ϵ are fixed ($s = 86$ and $\epsilon = 0.125$). We decide to only give values for $\beta = 2, 20$ and 100, because they are not significant for a larger β.

β c_0	2	20	100
1	13.10%	1.189%	0.238%
20	7.33%	0.023%	$1.2 * 10^{-10}$%
50	7.25%	0.012%	$5.9 * 10^{-11}$%
100	7.23%	$2.7 * 10^{-7}$%	$3.9 * 10^{-11}$%
200	7.22%	$1.0 * 10^{-7}$%	$2.9 * 10^{-11}$%

Table 2. Numerical values of the probability P_{luck} of tracing tag T according to branching factor β when the adversary tampers with c_0 tags. The system contains 2^{20} tags, $s = 86$ and $\epsilon = 0.125$.

4.5 Adversary probability P_{fail} when tampering with several tags

This probability represents the fact that $\mathcal{A}$, at some level in the tree, cannot take any rational decision given her observations. Thus since our attack definition forces $\mathcal{A}$ to give an answer, the latter will be randomly chosen.

$$P_{\text{fail}} = \Pr(E_1^4) + \sum_{i=2}^{d-1} \Big(\Pr(E_i^4) \times \Pr(\text{Continue}_{i-1}) \Big) \tag{3}$$

where Continue_i is the event that $\mathcal{A}$ continues the attack until level i.

Table 3 gives numerical values of Eq. 3 when the parameters s and ϵ are fixed ($s = 86$ and $\epsilon = 0.125$). Like with the table of P_{luck}, we decide to only give values for $\beta = 2, 20$ and 100, because they are not significant for a larger β.

Table 3. Numerical values of the probability P_{fail} of tracing tag T according to branching factor β when the adversary tampers with c_0 tags. The system contains 2^{20} tags, $s = 86$ and $\epsilon = 0.125$.

β c_0	2	20	100
1	27.57%	90.33%	98.00%
20	0.31%	15.59%	66.89%
50	0.08%	4.11%	36.60%
100	0.02%	3.02%	13.39%
200	0.01%	1.88%	2.56%

4.6 The adversary advantage when tampering with several tags

The advantage is defined as:

$$Adv_{\mathcal{A}} = 2\Big(P_{\text{succ}} + P_{\text{luck}} + P_{\text{fail}} \cdot \frac{1}{2} \Big) - 1 \tag{4}$$

Table 4 gives numerical values of Eq. 4 when the parameters s and ϵ are fixed ($s = 86$ and $\epsilon = 0.125$). The advantage $Adv_{\mathcal{A}}$ is plotted in Figure 5.

When c_0 is different from 1 and the branching factor β is less than 100, this advantage grows up to reach its maximum which is greater than 0.9. Then $Adv_{\mathcal{A}}$ reduces increasingly slower when c_0 augments. This outcome seams reasonable: clearly, the more tags an adversary $\mathcal{A}$ opens, the more secret keys she knows, which implies that the higher probability the attack success will be.

5 Conclusion

In this paper, we analyzed HSH which is an HB-like protocol using the MW tree-based key infrastructure.

First, we presented a new authentication protocol based on HB+ and on the LPN problem. We demonstrated that our proposal is as secure as HSH, in

β c_0	2	20	100	500	1000
1	0.2122	0.0434	0.0091	0.0018	0.0004
20	0.2716	0.8422	0.3274	0.0768	0.0392
50	0.2712	0.9571	0.6262	0.1810	0.0951
100	0.2711	0.9692	0.8536	0.3292	0.1811
200	0.2711	0.9811	0.9654	0.5497	0.3294

Table 4. Numerical values of the adversary advantage of tracing tag T according to branching factor β when the adversary tampers with c_0 tags. The system contains 2^{20} tags, $s = 86$ and $\epsilon = 0.125$.

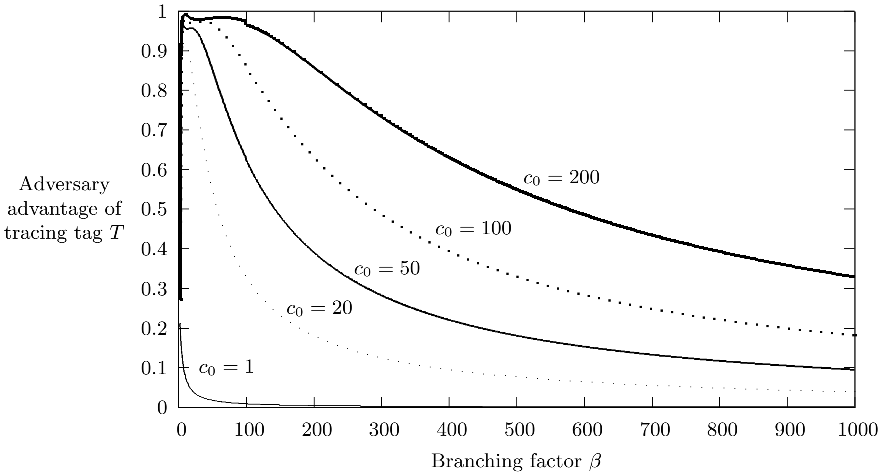

Fig. 5. Adversary advantage of tracing a tag T when she tampers with c_0 tags, $s = 86$, and $\epsilon = 0.125$.

the same privacy threat model (i.e. the tags are considered tamper-resistant, and the adversaries are supposed active, but man-in-the-middle attacks are not allowed). We showed that it is possible to build an authentication protocol with better properties than HSH: tag memory, and reader computational complexity are reduced.

Second we proved that, in a more realistic threat model where the tags are not tamper-resistant, the HSH protocol is defeated by an attack based on the weaknesses of the tree-based key infrastructure. We redesigned the original attack given by Avoine, Dysli and Oechslin to strike the Molnar and Wagner protocol: the main difficulty of this adaptation was to take into account the LPN problem present in HSH, since it relies on the HB-family protocols. In this real-life privacy threat model, our results showed that the adversary has a significant advantage to trace a tag.

Acknowledgements. This work was partially funded by the Walloon Region Marshall plan through the SPW DG06 Project TRASILUX.

References

1. B. Alomair, L. Lazos, and R. Poovendran. Passive Attacks on a Class of Authentication Protocols for RFID. In *International Conference on Information Security and Cryptology – ICISC'07*, volume 4817 of *Lecture Notes in Computer Science*, pages 102–115, Seoul, Korea, November 2007. Springer.
2. R. Anderson and M. Kuhn. Tamper Resistance - a Cautionary Note. In *2nd USENIX Workshop on Electronic Commerce*, pages 1–11, Oakland, CA, USA, November 1996.
3. R. Anderson and M. Kuhn. Low Cost Attacks on Tamper Resistant Devices. In *5th International Workshop on Security Protocols*, volume 1361 of *Lecture Notes in Computer Science*, pages 125–136, Paris, France, April 1997. Springer.
4. G. Avoine, E. Dysli, and P. Oechslin. Reducing Time Complexity in RFID Systems. In *Selected Areas in Cryptography – SAC'05*, volume 3897 of *Lecture Notes in Computer Science*, pages 291–306, Kingston, Canada, August 2005. Springer.
5. M. Barasz, B. Boros, P. Ligeti, K. Loja, and D. Nagy. Breaking LMAP. In *Workshop on RFID Security – RFIDSec'07*, Malaga, Spain, July 2007.
6. M. Bárász, B. Boros, P. Ligeti, K. Lója, and D. Nagy. Passive Attack Against the M2AP Mutual Authentication Protocol for RFID Tags. In *First International EURASIP Workshop on RFID Technology*, Vienna, Austria, September 2007.
7. 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*, 54(9):4339–4342, 2008.
8. J. Bringer, H. Chabanne, and D. Emmanuelle. HB++: a Lightweight Authentication Protocol Secure against Some Attacks. In *IEEE International Conference on Pervasive Services, Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing – SecPerU'06*, Lyon, France, June 2006. IEEE.
9. L. Buttyán, T. Holczer, and I. Vajda. Optimal Key-Trees for Tree-Based Private Authentication. In *Workshop on Privacy Enhancing Technologies - PET'06*, Cambridge, UK, June 2006.
10. T. Cao, E. Bertino, and H. Lei. Security analysis of the SASI protocol. *IEEE Transactions on Dependable and Secure Computing*, 6:73–77, 2008.
11. H.-Y. Chien. SASI: A New Ultralightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity. *IEEE Transactions on Dependable and Secure Computing*, 4(4):337–340, December 2007.
12. H. Gilbert, M. Robshaw, and H. Sibert. An Active Attack Against HB+ – A provably Secure Lightweight Authentication Protocol. Manuscript, July 2005.
13. H. Gilbert, M. J. Robshaw, and Y. Seurin. HB#: Increasing the Security and Efficiency of HB+. In *Advances in Cryptology – EUROCRYPT'08*, volume 4965 of *Lecture Notes in Computer Science*, pages 361–378, Istanbul, Turkey, April 2008. Springer.
14. H. Gilbert, M. J. Robshaw, and Y. Seurin. How to Encrypt with the LPN Problem. In *35th International Colloquium on Automata, Languages and Programming – ICALP'08*, volume 5126 of *Lecture Notes in Computer Science*, pages 679–690, Reykjavik, Iceland, July 2008. Springer.
15. T. Halevi, N. Saxena, and S. Halevi. Using HB Family of Protocols for Privacy-Preserving Authentication of RFID Tags in a Population. In *Workshop on RFID Security – RFIDSec'09*, Leuven, Belgium, July 2009.
16. G. Hammouri and B. Sunar. PUF-HB: A Tamper-Resilient HB Based Authentication Protocol. In *Applied Cryptography and Network Security – ACNS'08*, volume

5037 of *Lecture Notes in Computer Science*, pages 346–365, New York, NY, USA, May 2008. Springer.

17. N. J. Hopper and M. Blum. A Secure Human-Computer Authentication Scheme. Technical report, Computer Science Department, School of Computer Science, Carnegie Mellon University, May 2000.
18. N. J. Hopper and M. Blum. Secure Human Identification Protocols. In *Advances in Cryptology – ASIACRYPT'01*, volume 2248 of *Lecture Notes in Computer Science*, pages 52–66, Gold Coast, Australia, December 2001. Springer.
19. A. Juels and S. A. Weis. Authenticating Pervasive Devices with Human Protocols. In *Advances in Cryptology – CRYPTO'05*, volume 3621 of *Lecture Notes in Computer Science*, pages 293–308, Santa Barbara, CA, USA, August 2005. Springer.
20. X. Leng, K. Mayes, and K. Markantonakis. HB-MP+ Protocol: An Improvement on the HB-MP Protocol. In *IEEE International Conference on RFID*, pages 118–124, April 2008.
21. 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'07*, Vienna, Austria, April 2007.
22. T. Li and G. Wang. Security Analysis of Two Ultra-Lightweight RFID Authentication Protocols. In *IFIP SEC'07*, Sandton, Gauteng, South Africa, May 2007.
23. M. Madhavan, A. Thangaraj, Y. Sankarasubramaniam, and K. Viswanathan. NLHB : A Non-Linear Hopper Blum Protocol. arXiv.org, February 2010.
24. D. Molnar and D. Wagner. Privacy and Security in Library RFID: Issues, Practices, and Architectures. In *ACM Conference on Computer and Communications Security – ACM CCS'04*, pages 210–219, Washington, DC, USA, October 2004. ACM.
25. J. Munilla and A. Peinado. HB-MP: A Further Step in the HB-Family of Lightweight Authentication Protocols. *Computer Networks*, 51(9):2262–2267, 2007.
26. K. Nohl and D. Evans. Quantifying Information Leakage in Tree-Based Hash Protocols. In *Conference on Information and Communications Security – ICICS'06*, volume 4307 of *Lecture Notes in Computer Science*, pages 228–237, Raleigh, NC, USA, December 2006. Springer.
27. I. C. A. Organization. Machine Readable Travel Documents, Doc 9303, Part 1, Machine Readable Passports, Fifth Edition (2003).
28. 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. In *Workshop on RFID Security – RFIDSec'06*, Graz, Austria, July 2006.
29. 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 – UIC'06*, volume 4159 of *Lecture Notes in Computer Science*, pages 912–923, Wuhan and Three Gorges, China, September 2006. Springer.
30. P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador, and A. Ribagorda. EMAP: An Efficient Mutual Authentication Protocol for Low-Cost RFID Tags. In *OTM Federated Conferences and Workshop: IS Workshop – IS'06*, volume 4277 of *Lecture Notes in Computer Science*, pages 352–361, Montpellier, France, November 2006. Springer.
31. P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador, and A. Ribagorda. Advances in Ultralightweight Cryptography for Low-cost RFID Tags: Gossamer Protocol. In *Workshop on Information Security Applications – WISA'08*, volume 5379 of *Lecture Notes in Computer Science*, pages 56–68, Jeju Island, Korea, September 2008. Springer.

32. R. C.-W. Phan. Cryptanalysis of a New Ultralightweight RFID Authentication Protocol - SASI. *IEEE Transactions on Dependable and Secure Computing*, 6:316–320, 2008.
33. N. Semiconductors. MIFARE Smartcards ICs. `http://www.nxp.com/products/identification/card_ics/mifare`.
34. H.-M. Sun, W.-C. Ting, and K.-H. Wang. On the Security of Chien's Ultra-Lightweight RFID Authentication Protocol. *IEEE Transactions on Dependable and Secure Computing*, 99, 2009.
35. B. Yoon. HB-MP++ Protocol: An Ultra Light-weight Authentication Protocol for RFID System. In *IEEE International Conference on RFID*, Orlando, FL, USA, April 2009.

A The Details of our Probabilities Computations

During the sequel, we compute the probability of $\Pr(E_i^1)$. Then $\Pr(E_i^2)$, $\Pr(E_i^3)$, $\Pr(E_i^4)$, $\Pr(E_i^5)$ are computed in the same way. We will also denote "$\exists! x$" as "there exists an *unique* x".

A.1 When the adversary $\boldsymbol{\mathcal{A}}$ tampers with one tag

The probability $\Pr(E_i^1)$ is computed as follows:

$$\begin{aligned}\Pr(E_i^1) &= \Pr(C_i^1 \wedge A_i^1)\\ &= \Pr(\{y_i^0 = y_i^1\} \wedge \{B^1 y_i^0 \in \mathcal{B}_i^1\}) \times \Pr(\{y_i^0 \neq y_i^2\} \wedge \{B^2 y_i^0 \notin \mathcal{B}_i^2\})\\ &= \frac{1}{\beta} \times S_t \times \frac{\beta - 1}{\beta} \times (1 - V_t) \quad \text{(Bayes' law)}\end{aligned}$$

The overall probability P_{succ} is:

$$\begin{aligned}P_{\text{succ}} &= 2\Pr(E_1^1) + \sum_{i=2}^{d}\Big(2\Pr(E_i^1) \times \prod_{j=1}^{i-1}\Pr(E_j^4)\Big)\\ &= 2S_t(1 - V_t)\Big(\frac{\beta - 1}{\beta^2}\Big)\Big(\frac{1 - (\frac{S_t}{\beta})^{2d}}{1 - (\frac{S_t}{\beta})^2}\Big)\end{aligned}$$

A.2 When the adversary $\boldsymbol{\mathcal{A}}$ tampers with $\boldsymbol{c_0}$ tags

$\boldsymbol{P_{\text{succ}}}$ computation. The probability $\Pr(E_i^1)$ is computed as follows:

$$\begin{aligned}\Pr(E_i^1) &= \Pr(C_i^1 \wedge A_i^1)\\ &= \Pr(\{y_i^1 \in \mathcal{K}_i\} \wedge \{\exists k_{i,m} \in \mathcal{K}_i : B^1 k_{i,m} \in \mathcal{B}_i^1\})\\ &\quad \times \Pr(\{y_i^2 \notin \mathcal{K}_i\} \wedge \{\forall k_{i,m} \in \mathcal{K}_i : B^2 k_{i,m} \notin \mathcal{B}_i^2\})\\ &= \frac{c_i}{\beta} \times (1 - (1 - S_t)^{c_i}) \times \frac{\beta - c_i}{\beta}\\ &\quad \times (1 - \Pr(\{\exists! k_{i,m} \in \mathcal{K}_i : B^2 k_{i,m} \in \mathcal{B}_i^2\} | \{y_i^2 \notin \mathcal{K}_i\}))^{c_i}\\ &= \frac{c_i}{\beta} \times (1 - (1 - S_t)^{c_i}) \times \frac{\beta - c_i}{\beta} \times (1 - V_t)^{c_i}\end{aligned}$$

P_{luck} computation. The event B-Separation$_i$ divided in four cases:

- $M_i^1 = \{T_1$ is identified by its real key $y_i^1 \in \mathcal{K}_i\} \wedge \{T_2$ is not identified at all (even if it should)$\} = \text{Normal}^{\oplus}_{M_i}(T_1) \wedge \text{False}_{M_i}(T_2)$,
- $M_i^2 = \{T_1$ is identified by its real key $y_i^1 \in \mathcal{K}_i\} \wedge \{T_2$ is identified by a wrong known key$\} = \text{Normal}^{\oplus}_{M_i}(T_1) \wedge \text{Normal}^{\ominus}_{M_i^2}(T_2)$,
- $M_i^3 = \{T_1$ is identified by a wrong known key$\} \wedge \{T_2$ is not identified at all (even if it should)$\} = \text{Normal}^{\ominus}_{M_i}(T_1) \wedge \text{False}_{M_i}(T_2)$,
- $M_i^4 = \{T_1$ is identified by a wrong known key$\} \wedge \{T_2$ is identified by a wrong known key$\} = \text{Normal}^{\ominus}_{M_i}(T_1) \wedge \text{Normal}^{\ominus}_{M_i^4}(T_2)$.

$$\begin{aligned}
\Pr(\text{Normal}^{\oplus}_{M_i}(T_1)) &= \Pr(\{\exists ! k_{i,m} \in \mathcal{K}_i : B^1 k_{i,m} \in \mathcal{B}_i^1\} \wedge \{y_i^1 = k_{i,m} \in \mathcal{K}_i\}) = \frac{S_t}{c_i} \\
\Pr(\text{False}_{M_i}(T_2)) &= \Pr(\{\nexists k_{i,m} \in \mathcal{K}_i : B^2 k_{i,m} \in \mathcal{B}_i^2\} \wedge \{y_i^2 \in \mathcal{K}_i\}) = \frac{c_i(1-S_t)^{c_i}}{\beta} \\
\Pr(\text{Normal}^{\ominus}_{M_i^2}(T_2)) &= \Pr(\{\exists ! k'_{i,m} \in \mathcal{K}_i : B^2 k_{i,m} \in \mathcal{B}_i^2\} \wedge \{y_i^2 \in \mathcal{K}_i \setminus \{k'_{i,m},\ y_i^1\}\}) \\
&= \frac{(c_i-2)V_t}{c_i} \\
\Pr(\text{Normal}^{\ominus}_{M_i}(T_1)) &= \Pr(\{\exists ! k_{i,m} \in \mathcal{K}_i : B^1 k_{i,m} \in \mathcal{B}_i^1\} \wedge \{y_i^1 \in \mathcal{K}_i \setminus \{k_{i,m}\}\}) \\
&= \frac{(c_i-1)V_t}{c_i} \\
\Pr(\text{Normal}^{\ominus}_{M_i^4}(T_2)) &= \Pr(\text{Normal}^{\ominus}_{M_i}(T_1)) + \Pr(\text{Normal}^{\ominus}_{M_i^2}(T_2)) = \frac{(2c_i-3)V_t}{c_i} \\
\Pr(\text{B-Separation}_i) &= \Pr(M_i^1) + \Pr(M_i^2) + \Pr(M_i^3) + \Pr(M_i^4)
\end{aligned}$$

The event L-Separation$_i$ is divided in four cases where T_2 is no longer part of the current sub-tree:

- $N_i^1 = \{T_1$ is identified by its real key $y_i^1 \in \mathcal{K}_i\} \wedge \{T_2$ is not identified at all$\} = \text{Normal}^{\oplus}_{N_i}(T_1) \wedge \text{False}_{N_i}(T_2)$,
- $N_i^2 = \{T_1$ is identified by its real key $y_i^1 \in \mathcal{K}_i\} \wedge \{T_2$ is identified by a wrong known key $k'_{i,m}\} = \text{Normal}^{\oplus}_{N_i}(T_1) \wedge \text{Normal}^{\ominus}_{N_i}(T_2) \wedge \{y_i^1 \neq k'_{i,m}\}$,
- $N_i^3 = \{T_1$ is identified by a wrong known key$\} \wedge \{T_2$ is not identified at all$\} = \text{Normal}^{\ominus}_{N_i}(T_1) \wedge \text{False}_{N_i}(T_2)$,
- $N_i^4 = \{T_1$ is identified by a wrong known key $k_{i,m}\} \wedge \{T_2$ is identified by a wrong known key $k'_{i,m}\} = \text{Normal}^{\ominus}_{N_i}(T_1) \wedge \text{Normal}^{\ominus}_{N_i}(T_2) \wedge \{k_{i,m} \neq k'_{i,m}\}$.

The event L-Follow$_\ell$ is defined as $\{T_1$ is identified by its real key $y_\ell^1 \in \mathcal{K}_\ell\} \wedge \{T_2$ follows the same branch as T_1 at level ℓ (which is a false branch for T_2)$\} =$

$\text{Normal}^{\oplus}_{N_\ell}(T_1) \wedge \text{Fol}_\ell(T_2) \wedge \{y_i^1 = k'_{\ell,m}\}$. Thus we have:

$$
\begin{aligned}
\Pr(\text{L-Separation}_i) &= \Pr(N_i^1) + \Pr(N_i^2) + \Pr(N_i^3) + \Pr(N_i^4) \\
\Pr(\text{Normal}^{\oplus}_{N_i}(T_1)) &= \Pr(\text{Normal}^{\oplus}_{M_i}(T_1)) = \frac{S_t}{c_i} \\
\Pr(\text{Normal}^{\ominus}_{N_i}(T_2)) &= \Pr(\{\exists! k'_{i,m} \in \mathcal{K}_i : B^2 k'_{i,m} \in \mathcal{B}_i^2\} \wedge \{y_i^2 \notin \mathcal{K}_i\}) = \frac{(\beta - c_i)V_t}{\beta} \\
\Pr(\text{Normal}^{\ominus}_{N_i}(T_1)) &= \Pr(\text{Normal}^{\ominus}_{M_i}(T_1)) = \frac{(c_i - 1)V_t}{c_i}
\end{aligned}
$$

$$
\begin{aligned}
\Pr(\text{False}_{N_i}(T_2)) &= \Pr(\{\nexists k_{i,m} \in \mathcal{K}_i : B^2 k_{i,m} \in \mathcal{B}_i^2\} \wedge \{y_i^2 \notin \mathcal{K}_i\}) = \frac{(\beta - c_i)(1 - V_t)^{c_i}}{\beta} \\
\Pr(\text{Fol}_\ell(T_2)) &= \Pr(\text{Normal}^{\ominus}_{N_\ell}(T_1)) + \Pr(\text{Normal}^{\ominus}_{N_\ell}(T_2)) = V_t\left(\frac{c_\ell - 1}{c_\ell} + \frac{\beta - c_\ell}{\beta}\right) \\
\Pr(\text{L-Follow}_\ell) &= \frac{S_t\ V_t}{c_\ell^2}\left(\frac{c_\ell - 1}{c_\ell} + \frac{\beta - c_\ell}{\beta}\right)
\end{aligned}
$$

$\boldsymbol{P}_{\textbf{fail}}$ **computation.** Continue$_i$ is composed of the following three main cases:

- for the i-th levels, T_1 and T_2 are identified by their real key: $\{y_i^1 \in \mathcal{K}_i\} \wedge \{y_i^2 \in \mathcal{K}_i\} \wedge \{y_i^1 = y_i^2\} = E_i^5$,
- for the i-th levels, T_1 and T_2 are identified by the same key, which is the real one only for $T_1 = $ L-Follow$_i$,
- for the i-th levels, T_1 and T_2 are identified by the same wrong key $= Q_i$, $Q_i = \text{Fol}_i(T_1) \wedge \text{Fol}_i(T_2) \wedge \{k_{i,m} = k'_{i,m}\}$,

and of the ordered combinations of theses cases: for $1 \le \ell < k < j \le i$, {L-Follow$_k$} can be only preceded by $\{E_\ell^5\}$, and $\{Q_j\}$ can be preceded by {L-Follow$_k$} or $\{E_\ell^5\}$ or {L-Follow$_k$ and E_ℓ^5}. Thus, we have:

$$
\begin{aligned}
\Pr(\text{Continue}_i) = &\prod_{j=1}^{i}\Big(\Pr(E_j^5) + \Pr(\text{L-Follow}_j) + \Pr(Q_j)\Big) \\
&+ \sum_{j=1}^{i-1}\left(\prod_{k=1}^{j}\Pr(\text{L-Follow}_k) \times \prod_{\ell=j+1}^{i}\Pr(Q_\ell)\right) \\
&+ \sum_{j=1}^{i-1}\left(\prod_{k=1}^{j}\Pr(E_k^5) \times \prod_{\ell=j+1}^{i}\Big(\Pr(\text{L-Follow}_\ell) + \Pr(Q_\ell)\Big)\right) \\
&+ \sum_{j=1}^{i-2}\left(\prod_{k=1}^{j}\Pr(E_k^5) \times \sum_{\ell=j+1}^{i-1}\Big(\prod_{m=j+1}^{\ell}\Pr(\text{L-Follow}_m) \times \prod_{p=\ell+1}^{i}\Pr(Q_p)\Big)\right)
\end{aligned}
$$