# LIGHTWEIGHT AUTHENTICATION PROTOCOLS BASED ON ROTATIONS AND THE LPN PROBLEM

Alberto Peinado and Jorge Munilla
*ETSI Telecomunicación, University of Málaga, Málaga, Spain*

Keywords: Lightweight Authentication, Cryptanalysis, RFID Authentication, LPN, Rotations.

Abstract: Many lightweight authentication protocols based on the LPN (Learning Parity with Noise) problem have been proposed, the first of which is the HB protocol. In 2007, the HB-MP protocol was presented to overcome the vulnerabilities by means of internal rotations. Since then, new protocols have been presented to improve the HB-MP. In this paper, we present a general analysis of the HB-MP related protocols, including the cryptanalysis of HB-MP++, define design guidelines and propose a new protocol following the model.



## 1 INTRODUCTION

Learning Parity in the presence of Noise (LPN) problem (Fossorier, 2006) constitutes one of the foundations for the development of many lightweight authentication protocols specially designed to be implemented in devices with computational constraints, such as RFID systems.

In 2001, Hopper and Blum (Hopper, 2001) propose the first RFID authentication protocol based on the LPN problem, known as HB protocol. Later, Juels and Weis (Juels, 2005) modify the HB protocol to be secure against active attacks. This modification, known as HB+ protocol, introduces an additional *k*-bit secret key shared by the tag and the reader, and establishes that the protocol is initiated by the tag instead of by the reader.

Gilbert, Robshaw and Seurin show in 2005 (Gilbert, 2005) that HB+ protocol is vulnerable to active attacks where the adversary, apart from eavesdropping on the communication between the parties, is able to modify the challenges going from the reader to the tag, and check whether this manipulation results (or not) in a successful authentication. In the HB framework, this kind of Man-In-The-Middle (MITM) adversary is accounted by the GRS security model.

Bringer, Chabanne and Dottax propose the HB++ protocol (Bringer, 2006), defined to be secure against the GRS attack at the price of making rather more computations. The main idea behind HB++ protocol is to generate a second noisy parity bit, by running the protocol twice under independent secrets but with correlated challenges. However, Gilbert et *al*. describe an attack in (Gilbert, 2008).

Later, many others protocols were proposed with different objectives such as to improve the efficiency; reduce the operation complexity; reduce the number of messages, but all of them trying to overcome known attacks.

HB++ is the first protocol, belonging to the HB family that uses rotations to improve the security. The next proposal including rotations is the HB-MP protocol (Munilla, 2007) that tries to improve the security maintaining the simplicity of the original HB protocol. A flaw, reported in (Gilbert, 2008), was detected in its design. As a consequence several improvements has been presented, such as HB-MP+ (Leng, 2008), HB-MP++ (Yoon, 2009) and CL-HB (Ya-Fen, 2009). However, all of them make use of others operations decreasing the level of simplicity stated in the original HB protocol, and allowing to break the system, as it is the case of HB-MP++.

The objective of this paper is to complete the analysis of the authentication protocols based on LPN problem and rotations, presenting the cryptanalysis of the HB-MP++, and to provide a general model to design this kind of protocols, giving an example, named as HB-ROT1.

Next section describes the LPN problem and the original HB protocol. Section III describes in detail the HB-MP related protocols. Section IV deals with the cryptanalysis of HB-MP++. In Section V, several design guidelines are stated to develop

Peinado A. and Munilla J..
LIGHTWEIGHT AUTHENTICATION PROTOCOLS BASED ON ROTATIONS AND THE LPN PROBLEM.
DOI: 10.5220/0003827901270134
In *Proceedings of the 1st International Conference on Sensor Networks* (SENSORNETS-2012), pages 127-134
ISBN: 978-989-8565-01-3

authentication protocols using rotations in the context of HB protocols.

## 2 NOTATIONS

LPN problem can be defined as follows. Let $k$ be a security parameter, let $x$ and $a^1$, ..., $a^r$ be binary vectors of length $k$, and let $z^i = a^i \cdot x$ denote the dot product of $a^i$ and $x$ (modulo 2). If the values $a^1$, $z^1$; $a^2$, $z^2$;...; $a^r$, $z^r$ are given for randomly-chosen $\{a^i\}$, it is possible to find $x$ efficiently by using standard linear-algebraic techniques. Note that it can be expressed in matrix notation as $Z = A \cdot x$, where $Z$ is a column vector containing the dot products $z^i$, and $A$ is a matrix whose rows correspond to vectors $a^i$.

However, solving for $x$ in the presence of noise where each $z^i$ is flipped randomly (and independently) with probability η, becomes much more difficult; *i.e.* $z^i = a^i \cdot x \oplus v^i$, where $v^i$ follows a Bernoulli distribution of parameter η, with $\eta \in (0, 1/2)$. So, let |H| denote the Hamming weight of the binary string H, the LPN problem involves finding a $k$-bit vector $x'$ that is functionally close to $x$ such that $|(A \cdot x') \oplus Z| \leq r\eta$. Formally, it is as follows:

**Definition 1** (LPN problem) The LPN problem with security parameters $r$, $k$ and η, with $\eta \in (0, 1/2)$, is defined as follows: let $A$ be a random $r \times k$ binary matrix, let $x$ be a random $k$-bit vector, and let $V$ be a random $r$-bit vector with noise parameter η such that $|V| \leq \eta r$. Given $A$, η, and the product $Z = A \cdot x \oplus V$, find a $k$-bit vector $x'$ such that

$$|A \cdot x' \oplus Z| \leq \eta r \quad (1)$$

We introduce the notation which will be used throughout the rest of the paper –which is directly connected with the notation previously used to describe the LPN problem:

- $x$, $y$ random secret keys shared by the parties.
- $k$ length of the secret key $x$.
- $t$ number of wrong responses tolerated.
- $a$, $b$ random $k$-bit vectors.
- $A$,$B$ $n \times k$ binary matrices (of the vectors $a$, $b$).
- L|$w$| length of the binary vector $w$.
- $w[i]$ denote the $i$th bit of the binary vector $w$.
- η noise parameter with $\eta \in (0, 1/2)$.
- $v$ noise bit; $v = 1$ with probability η.
- ⊕ denotes XOR operation.
- $w\ p$ denotes the scalar/dot product of the vectors $w$ and $p$.
- rot($w$, $p$) the bitwise left rotate operator. The operand $w$ is rotated $p$ positions.
- $\lfloor w \rfloor_p$ the $p$ less significant bits of the vector $w$.
- trun($w$, $p$) truncate operator. The operand $w$ is truncated $p$-LSB.
- Pr[$W$] denotes the probability of an event $W$.

### 2.1 HB Protocol

Hoper and Blum's LPN-based authentication protocol (Hopper, 2001) consists of $r$ rounds, where $r$ is a security parameter, which can be described as follows in the RFID setting (see Fig.1):

**Step 1.** The reader generates a bitstring $a$ (challenge), and sends it to the tag.

**Step 2.** The tag generates a noise bit $v$, and computes $z = a \cdot x \oplus v$, by computing the parity bit and adding the bit noise $v$ to the result. The tag sends $z$ (response) to the reader.

**Step 3.** The reader checks the parity bit $z = a \cdot x$

When the $r$ rounds finish, the reader will accept the tag as valid if less than (a threshold) $t = r\eta$ rounds are incorrect. Otherwise, the tag is rejected.

Only AND and XOR operations have to be implemented to perform the protocol, and the inner product $a \cdot x$ can be computed on the fly as each bit of $a$ is received; *i.e.* there is no need for the tag to store the entire vector $a$. Besides the number of rounds $r$ and the length of the keys $k$, which can be easily observed, the noise parameter η is also assumed to be public.

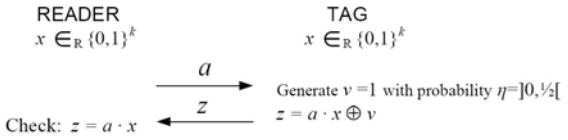


Figure 1: A single round of the HB protocol.

## 3 HB-MP PROTOCOLS

In this section, the most relevant protocols belonging to the HB-MP family are described, in order to design the cryptanalysis of the HB-MP++ and establish the guidance to overcome the vulnerabilities and weaknesses.

### 3.1 HB-MP Protocol

In order to overcome the active attacks reported on the HB protocol, but maintaining the original

simplicity, the HB-MP protocol (Munilla, 2007) incorporates and additional secret key $y$, as others HB related protocols.

The secret key $x$ will change each round by rotating it one bit ($y_i = 1$), or not ($y_i = 0$), according to the value of the $i$th bit of the secret key $y$. The initial value of $x$ will be restored at the beginning of each authentication session since it is stored in non-volatile memory. The answer $z$ is computed by using only the first $m$ bits ($\lfloor x \rfloor_m$) of the rotated $x$. The protocol consists of $r$ rounds, which can be described as follows (see Fig. 2).

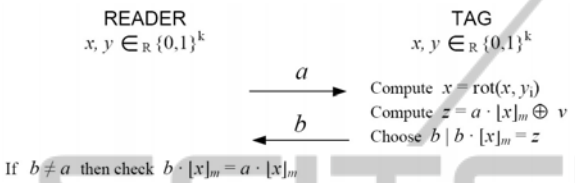


Figure 2: A single round of the HB-MP protocol.

**Step 1.** The reader chooses at random an $m$-bit vector $a$, and sends it to the tag.

**Step 2.** Reader and tag compute $x = \text{rot}(x, y_i)$ .

**Step 3.** The tag generates a noise bit $v$, computes $z = a \cdot \lfloor x \rfloor_m \oplus v$, and chooses an $m$-bit vector $b$ such that $b \cdot \lfloor x \rfloor_m = z$. The tag sends the response $b$ to the reader.

**Step 4.** If $a = b$, the answer is considered wrong. Else, the reader checks if $a \cdot \lfloor x \rfloor_m = b \cdot \lfloor x \rfloor_m$

When the $r$ rounds finish, the reader will accept the tag as valid if the number of failures is lower than $t$. Otherwise, the tag is rejected.

To pick the vector $b$, the authors suggest an easy algorithm for $\eta = 1/4$, which does not resort to any noise generator –in contrast to ordinary based-HB protocols.

The initial values of $x$ and $y$ are always the same at the beginning of every authentication. This avoids synchronization problems, which are difficult to deal with when complexity of the tags is very low, but represents, as pointed out in (Leng, 2008), the main weakness of HB-MP. This weakness stems from the fact that the rotations of $x$ are identical for all the authentication sessions.

## 3.2 HB-MP+ Protocol

Leng, Mayes and Markantonakis (Leng, 2008) state that the "weak" rotations of HB-MP also may compromise the security of the protocol against Man-In-The-Middle attacks (GRS), and propose an improved HM-MP protocol: HB-MP+.

This improved version incorporates a one-way function $f(\cdot)$, and an intermediate value $u = f(a, y)$, so that the bits of $x$ used in each round are unpredictable (see Fig. 3):

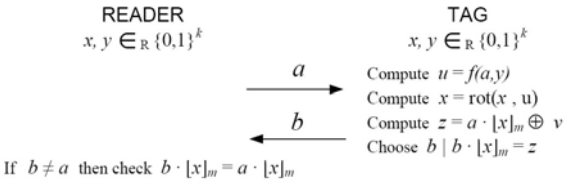


Figure 3: A single round of the HB-MP+ protocol.

**Step 1.** The reader picks at random an $m$-bit vector $a$, and sends it to the tag.

**Step 2.** Reader and tag compute $u = \text{f}(a, y)$ and $x = \text{rot}(x, u)$.

**Step 3.** The tag generates a noise bit $v$, computes $z = a \cdot \lfloor x \rfloor_m \oplus v$, and chooses an $m$-bit vector $b$ such that $b \cdot \lfloor x \rfloor_m = z$. The tag sends $b$ to the reader.

**Step 4.** The reader checks if $a \cdot \lfloor x \rfloor_m = b \cdot \lfloor x \rfloor_m$

When the $r$ rounds finish, the reader will accept the tag as valid if the number of failures is lower than $t$. Otherwise, the tag is rejected.

This protocol thwarts the passive attack of Gilbert (Gilbert, 2008) because $\lfloor x \rfloor_m$ used in the authentication 1 change for the authentication 2.

The authors extend the idea of using a different random key in each round (round key), and so, they describe an abstract form of the HB-MP+ protocol where the rotation operator is substituted for a one-way function, so that the round key $xr_i$ is computed as follows: $xr_i = \text{f}(a, x)$. As $x$ is not changed, there are not synchronization problems between the reader and the tag, and the shared secret y is not required anymore.

Unfortunately, these one-way functions are not concreted by the authors, and no information about the cost of implementation is provided.

## 3.3 HB-MP++ Protocol

HB-MP++ (Yoon, 2009) goes further than HB+, and proposes the use of a $k$-stage Linear Feedback Shifter Register (LFSR). This LFSR is used to generate pseudo-random noise sequences PNsequence, and the randomness is extracted by counting "runs" of these sequences.

A "run" is defined as a sequence of a single type of binary digits; *e.g.* for the 16-bit PNsequence = 0001001101011110, the following "run lengths" are calculated:

$run_1 = 3$, $run_2 = 1$, $run_3 = 2$, $run_4 = 2$, $run_5 = 1$, $run_6 = 1$, $run_7 = 1$, $run_8 = 4$, $run_9 = 1$, corresponding to

$$000 - 1 - 00 - 11 - 0 - 1 - 0 - 1111 - 0.$$

This protocol consists of $r$ rounds and the parties only need to share one $k$-bit secret key $x$. The round $i$th of the protocol is illustrated in Fig. 4, and described as follows:

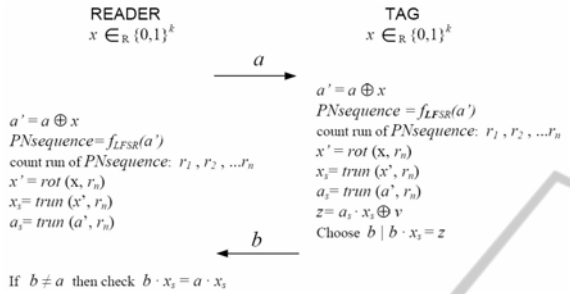


Figure 4: A single round of the HB-MP++ protocol.

**Step 1.** The reader chooses a random $k$-bit vector $a$, and sends it to the tag.

**Step 2.** The parties compute $a' = a \oplus x$, and generate a pseudo-random sequence from the LFSR by using $a'$ as the initial value; i.e. PNsequence = $f_{LFSR}(a')$.

Then, the parties calculate the lengths of the "runs" of the PNsequence, and use $run_n$ to compute the round key: $x_s$ = trun($x'$, $run_n$) where $x'$ = rot($x$, $run_n$). The length of the round key is $k-run_n$ and therefore the $run_n$ LSB of the challenge must also be truncated $a_s$ = trun($a'$, $run_n$).

**Step 3.** The tag computes $z = a_s \cdot x_s \oplus v$, and picks a random ($k - run_n$)-bit vector $b$ such that $z = b \cdot x_s$ (the original paper says that "the tag looks for a random-bit (< k) binary vector", but the dot product requires that the two operators have the same length, and thus $L|b| = L|x_s| = k-run_n$). The tag sends $z$ to the reader.

**Step 4.** The reader checks if $as \cdot x_s = b \cdot x_s$

When the $r$ rounds finish, the reader will accept the tag as valid if the number of failures is lower than t. Otherwise, the tag is rejected.

The length ($k-run_s$) of the response vector $b$ changes in each round $s$, which –according to the authors– increases the resistance against traceability. However, this could represent a weakness and questions the utility of the PNsequence, since the adversary can know how many positions the key $x$ (always the same) was rotated.

The utilisation of a LFSR, and more precisely, the way in which the PNsequence is generated and applied to the HB-MP++ protocol allows an attacker to recover easily the secret key $x$ of the tag, thus breaking completely the security of the system.

The recovery process is described in section IV.

### 3.4 CL-HB Protocol

The essential idea of CL-HB (Ya-Fen, 2009) is to provide mutual authentication by repeating the HB-MP protocol in both ways. Thus, this protocol is subject to the same passive attack as the original HB-MP.

## 4 CRYPTANALYSIS OF HB-MP++

The original description of the protocol HB-MP++ presented in (Yoon, 2009) does not provide enough details on the operations, data formats and functions. This facts makes the cryptanalysis more complex because some assumptions must be made. However, we have considered all possibilities when incomplete description affects to the analysis. In this way, we consider the following cases classified following two main criteria: the way in which the runs are applied, and the convention employed to represent data (keys, challenges and responses).

C1: We consider that $run_n$ at $i$-th round always corresponds to $n$-th run of the PN sequence generated from the seed $a'$, where $n$ is a security parameter of the protocol. Note that $run_n$ takes different values at each round because the PN sequence depends on the challenge $a'$, which is different at each round.

C2: We consider that $run_n$ at $i$-th round corresponds to $i$-th run of the PN sequence generated from the seed $a'$. In this way, the first round employs the value $run_1$ (length of fisrt run in the PN sequence); the second round employs $run_2$ (length of second run in the PN sequence); and so on.

C3. We consider that the least significant bit (LSB) corresponds to the rightmost bit of every data (keys, challenges, responses...). Since rot() operation is defined as "left rotation" in (Yoon, 2009), the bits will be shifted to the most significant locations. Although this is not the usual meaning of "left rotation", the description in (Yoon, 2009) points out to this possibility.

C4. We consider that the least significant bit (LSB) corresponds to the leftmost bit of every data (keys, challenges, responses,...). In this case, the rot() operation works in the usual way shifting the bits to the least significant locations (left rotations).

## 4.1 Cryptanalysis of case C1-C3

We consider simultaneously the assumptions C1 and C3. As one can observe in Fig. 5, the round key $x_s$ depends only on the value of $run_n$ (named $u$ for simplicity), in such a way that $x_s = \text{trun}(x,u)$, where $x_s$ corresponds to the $u$ least significant bits of $x$.

This fact implies that $x_0 = x_s[0]$; $x_1 = x_s[1]$,...; that is, the bits location of the secret key $x$ is known.

Since the challenge $a$ determines the PN sequence, and hence the runs, it is not convenient to applied the GRS attack on $a$. Instead, we apply a man in the middle attack, based on the GRS model (Gilbert, 2005), to the response $b$. It is as follows (see Fig. 6).

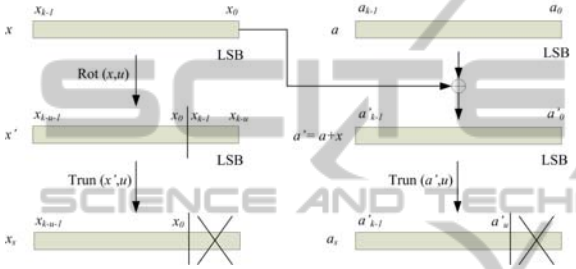


Figure 5: Round key $x_s$ generation when LSB corresponds to the rightmost bit of every data in the HB-MP++ protocol.

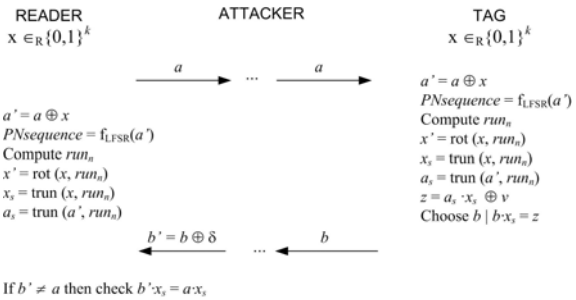


Figure 6: Man in the middle attack applied to the case C1-C3 of the HB-MP++ protocol.

**Step 1.** The attacker retransmits the challenge $a$ without modification.

**Step 2.** The attacker receives the response b and sends to the reader the modified response $b' = b \oplus \delta$, where $\delta$ takes the constant value $\delta = 000\cdots01$.

When all rounds finish, if the tag has been successfully authenticated, the attacker concludes that $\delta \cdot x_s = 0$, and thus $x_0 = 0$. If the tag is rejected, the attacker concludes $\delta \cdot x_s = 1$, determining that $x_0 = 1$.

If the attacker repeats the scheme using different values for $\delta$ ($000\cdots010$, $00\cdots0100$, ...) he can recover other bits of the secret key $x$.

The attack is possible because the bits of secret key $x$ appear always in the same location.

## 4.2 Cryptanalysis of case C1-C4

We consider simultaneously the assumptions C1 and C4. As one can observe in Fig. 7, the round key $x_s$ depends only on the value of $run_n$ (named $u$ for simplicity), in such a way that the bits of secret key $x$ always appears in $x_s$ following a known pattern.

More precisely, $x_0$ (the LSB of $x$) corresponds to ($k$-$u$) location in round key $x_s$; $x_{k-1}$ corresponds to ($k$-$u$-1) location in $x_s$; etc. Since the value $u$ can be obtained from the difference between challenge and response lengths, all locations are known.

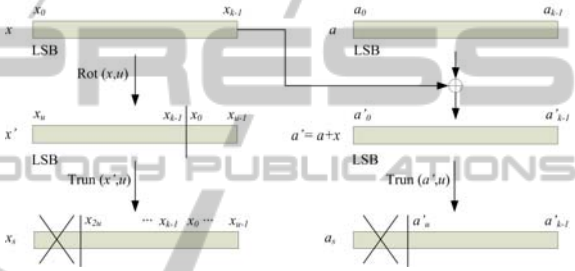


Figure 7: Round key $x_s$ generation when LSB corresponds to the rightmost bit of every data in the HB-MP++ protocol.

In this case, the attack follows the same model as previous case, with the difference that the attacker needs to compute the value $u$ for each response in order to generate the parameter $\delta$ in such a way that the "1" is in the location that corresponds to the secret key bit to be recovered. Hence, $\delta$ will take different values at each round. The attack is as follows.

**Step 1.** The attacker retransmits the challenge $a$ without modification.

**Step 2.** When the attacker receives the response $b$, he computes $u = L|a| - L|b| = k - run_n$.

**Step 3.** The attacker generates the vector $\delta$ with a "1" in ($k$-$u$) location (the rest components are all zeros) if the bit to be discovered is $x_0$. Then, the attacker sends $b' = b \oplus \delta$ to the reader.

When all rounds finish, if the tag has been successfully authenticated, the attacker concludes that $\delta \cdot x_s = 0$, and thus $x_0 = 0$. If the tag is rejected, the attacker concludes $\delta \cdot x_s = 1$, determining that $x_0 = 1$.

The attacker can repeat the scheme to discover other bits of the key using different values of $\delta$.

### 4.3 Cryptanalysis of case C2

We consider the assumption C2. In this case, the previous attacks can also be applied when C3 and C4 are considered. However, a different attack is now possible, independently of C3/C4 assumption, if we focus on the information leaked by the length of the responses.

The attacker impersonates the server sending always the same challenge to the tag. In this way, the tag generates always the same PN sequences and will employ the $run_i$ of the same PN sequence at the round $i$ of the protocol.

Hence, if the attacker obtains the successive runs of a sequence, he can reconstruct the sequence itself. The process is as follows.

**Step 1.** The forge server sends the challenge $a$ (always the same challenge at each round) to the tag.

**Step 2.** The tag computes $run_i$ and sends the response $b$ of length ($k$-$run_i$)

**Step 3.** The forge server compute $run_i$ = L|$a$| - L|$b$|.

When all rounds finish, the attacker reconstructs the PN sequence as follows. Since the attacker only knows the run lengths, he has to consider two possibilities on the first run: it is composed by zeros or by ones.

An example: Suppose that the successive run lengths are $run_1 = 3$, $run_2 = 1$, $run_3 = 2$, $run_4 = 2$, ... Hence, the PN sequence is one of these two sequences:

000-1-00-11-... or 111-0-11-00-...

Once the sequence is reconstructed, it is possible to recover the initial state because the PN sequence is generated by a LFSR (in the original paper (Yoon, 2009) the authors does not provide much information about this step).

The initial state is $a' = a \oplus x$. Hence, if the attacker reconstructs $a'$, the secret key $x$ can be obtained because $a$ is known.

Some consideration must be taken into account.

If we consider that the LFSR feedback polynomial is not known, the attacker needs to reconstruct at least $2k$ bits of the sequence to obtain the polynomial ($k$ being the bit length of $a$ and $x$). To do this are not necessary $2k$ rounds. Instead, $k$ rounds may be sufficient because each round produce more than one bit on average.

If the LFSR feedback polynomial is known, as it occurs in most LFSR applications, then only $k$ bits must be reconstructed.

## 5 DESIGN OF PROTOCOLS BASED ON ROTATIONS AND LPN PROBLEM

As one can observe in the previous sections, the main idea behind the HB-MP protocol resides on the increasing of robustness by means of rotations, but maintaining the advantages and simplicity of the original HB protocol. Despite of the flaw in its design, the simplicity of HB-MP has originated several improvements which include others operations that move away from the initial target. As a consequence, the global complexity increases.

The analysis of the HB-MP protocol and its derivatives (HB-MP+, HB-MP++ and CL-HB) allows to establish clear design objectives to assure a reasonable level of security and simplicity.

The identifying features of the HB-MP protocol are:

a) Utilization of rotations to derive round keys from a master secret key $x$.

b) Utilization of the LPN problem as the main foundation to assure a good security level, as the rest of HB-familiy members.

c) Utilization of only two messages between the parties.

The improvements of HB protocol do not overcome the limitations and weaknesses. The limitation of the HB-MP+ protocol resides on the hash function. In (Munilla and Peinado, 2007), the authors do not provide enough details to analyze a real implementation. The HB-MP++ has been designed to avoid traceability, as it is claimed by the authors in (Yoon, 2009). However, the modifications applied allows an attacker to recover the key $x$.

As a consequence, we establish the following guidelines to design new lightweight authentication protocols combining rotations and LPN problem.

a) The number of messages interchanged between the parties must be minimum. It is recommended not greater than two, following the model of HB and HB-MP protocols.

b) Rotations must be the main operation to derive all round keys from a master secret key $x$.

c) The keys (the round keys) must be different at each authentication session.

d) The round keys must be different. Each round will use a different round key.

e) The round key generation algorithm must allow the synchronization between reader

and tag.

f) The key generation algorithm will depend on the challenge sent by the reader, but not in a direct nor exclusive way.

g) No fixed pattern must exist between round and master key bits location.

h) The challenge must not be directly used in the scalar product; that is, z must not depend directly on challenge.

i) Checking equation at reader must avoid passive attacks.

### 5.1 HB-ROT1: A Proposal

As an example, we present in this subsection a protocol that follows the design guidelines of previous section. We call it HB-Rot1.

In this protocol, reader and tag share two secret keys, $x$ with length $k$ and $y$ with length $m < k$. This protocol consists of $r$ rounds. The round $i$-th of the protocol is illustrated in Fig. 8, and described as follows:

**Step 1.** The reader chooses a random $m$-bit vector $a$, and sends it to the tag.

**Step 2.** The parties compute the key $x_i$ using $x$ and $y$ as follows:

$$x_i = \text{rot}(x_{i-1}, y[i]+1) \text{ where } x_0 = x \quad (2)$$

This algorithm produces the key for round $i$, by rotation of the key employed in previous round $i$-1. The rotation is controlled by the $i$-th bit of the secret key $y$, in such a way that if $y[i] = 0$ the key is rotated one position, and if $y[i] = 1$ the key is rotated two positions. This algorithm warrants that the keys are different at each round.

**Step 3.** The parties compute a modified challenge as follows

$$a' = \text{rot}(a \oplus y, \text{trun}(x_i, p)) \quad (3)$$

where $p$ is a system parameter. This value will be a small integer in order to bound the number of rotations to be applied.

**Step 4.** The parties compute the key $x'_i$

$$x'_i = \text{rot}(x_i, \lfloor a' \rfloor_p)) \quad (4)$$

**Step 5.** The parties compute the round key $x_s$

$$x_s = \lfloor x'_i \rfloor_m \quad (5)$$

**Step 6.** The tag computes $z = a' \cdot x_s \oplus v$, and picks a random $m$-bit vector $b$ such that $z = b \cdot x_s$

**Step 7.** The reader checks if $a' \cdot x_s = b \cdot x_s$

When the $r$ rounds finish, the reader will accept the tag as valid if the number of failures is lower than $t$. Otherwise, the tag is rejected.

This protocol avoids the GRS attack applied to HB-MP++, because there is no fixed pattern between the master key and the round key bits locations. Furthermore, these locations are determined by the challenge $a'$, and hence they cannot be computed previously.

The key generation algorithm allows the synchronization between the parties. This fact, however, does not imply the re-utilization of keys at each authentication session, because the rounds keys also depend on the challenge and the secondary master key $y$.

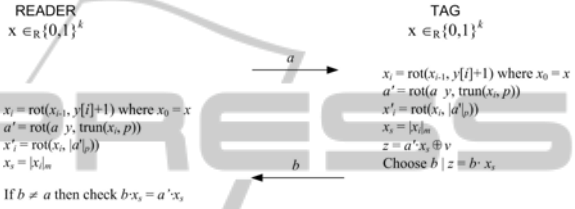


Figure 8: A single round of the HB-Rot1 protocol.

## 6 CONCLUSIONS

In this paper we have revised the security weaknesses of the HB-MP related protocols, that is, the HB-MP itself, the HB-MP+, the HB-MP++ and the CL-HB, providing a complete cryptanalysis of the HB-MP++, proving that it is not secure.

This cryptanalysis, although no much information is provided in the original proposal of HB-MP++, shows different ways to recover the master secret key $x$ of the tag.

Furthermore, the results of this work allow the establishment of a set of design guidelines oriented to the generation of new authentication protocols based on rotations and the LPN problem that overcome the limitations and weaknesses reported. We also propose the protocol HB-Rot1, as a sample of lightweight authentication protocol based on rotations and LPN problem.

## ACKNOWLEDGEMENTS

This work has been partly supported by the Spanish Ministry of Science and Innovation and the European FEDER funds under project TIN 2008-02236/TSI and TIN2011-25452.

## REFERENCES


Bringer, J., Chabanne, H., Dottax, E., 2006. HB++: a lightweight authentication protocol secure against some attacks, *IEEE International Conference on Pervasive Services, Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous computing - SecPerU*.

Fossorier, M. P. C., Mihaljevic, M. J., Imai, H., Cui, Y., Matsuura, K., 2006. A Novel Algorithm for Solving the LPN Problem and its Application to Security Evaluation of the HB protocol for RFID Authentication. *In Rana Barua and Tanja Lange, editors, INDOCRYPT*, LNCS 4329, pp 48–62. Springer.

Gilbert, H., Robshaw, M., Sibert, H., 2005. An Active Attack Against HB+. A Provably Secure Lightweight Authentication Protocol. *IEE Electronic Letters,* 41(21):1169–1170.

Gilbert, H., Robshaw, M., Seurin, Y., 2008. Good Variants of HB+ Are Hard to Find. *In Financial Cryptography, pages 156–170*.

Hopper, N. J., Blum, M., 2001. Secure Human Identification Protocols. *In ASIACRYPT '01: Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security,* pp 52–66, London, UK. Springer-Verlag.

Juels, A., Weis, S., 2005. Authenticating Pervasive Devices with Human Protocols. *In Advances in Cryptology – CRYPTO'05, LNCS 3126,* pp 293–308, Santa Barbara, California, USA. Springer-Verlag.

Leng, X., Mayes, K., Markantonakis, K., 2008. HB-MP+ Protocol: an improvement on the HB-MP protocol. *In IEEE International Conference on RFID,* pages 118–124, Las Vegas, Nevada, USA.

Munilla, J., Peinado, A., 2007. HB-MP: A further step in the HB-family of lightweight authentication protocols. *Computer Networks, 51(9):2262–2267*.

Ya-Fen, C., Yen-Cheng, L., 2009. An LPN-problem based Lightweight Authentication Protocol for Wireless Communication. *In IEEE International Conference on Availability, Reliability and Security,* pages 130–134, Krakow, Poland.

Yoon, B., Sung, M. Y., Yeon, S., Hyun, S. O., Kwon, Y., Kim, C., Kim K., 2009. HB-MP++ Protocol: An Ultra Light-weight Authentication Protocol for RFID System. *In IEEE International Conference on RFID,* pages 186–191, Orlando, Florida, USA.