

Recursive Linear and Differential Cryptanalysis of Ultralightweight Authentication Protocols

Zahra Ahmadian, Mahmoud Salmasizadeh, and Mohammad Reza Aref

Abstract—Privacy is faced to serious challenges in the ubiquitous computing world. In order to handle this problem, some researches in recent years have focused on design and analysis of privacy friendly ultralightweight authentication protocols. In less than a decade, many ultralightweight authentication protocols are proposed. Though, successful cryptanalyses are proposed for almost all of them, most of these attacks are based on ad-hoc methods that are not extensible to a large class of ultralightweight protocols. So this research area still suffers from the lack of structured cryptanalysis and evaluation methods.

In this paper, we introduce new frameworks for full disclosure attacks on ultralightweight authentication protocols based on new concepts of *recursive linear* and *recursive differential* cryptanalysis. Both of them exploit triangular functions in ultralightweight protocols and recover all secret data stored in the tag in a recursive manner. The recursive linear attack is applied to Yeh et al. and SLMAP authentication protocols. This attack is passive, deterministic (i.e. the attacker can retrieve all the secrets with probability of one), and requires only a single authentication session. The recursive differential attack is more powerful and can be applied to the protocols which linear attack may not work on. We show the effectiveness of this attack on LMAP⁺⁺ and SASI authentication protocols. This differential attack is probabilistic, active in the sense that the attacker suffices only to block some specific messages, and requires a few authentication sessions.

Index Terms—RFID technology, Ultralightweight authentication protocols, Triangular functions, system of linear equations.



1 INTRODUCTION

RADIO Frequency Identification (RFID) is the technology for automatically unique identification or tracking of the objects using wireless non contact systems. The target object is attached a *tag* which is a transponder with limited capability of computation, storage and communications. The element that identifies the tag is called the *reader* which is a transceiver that comparing the tag, has much more computational capabilities. The reader is securely connected to a back end system which keeps the data associated with the tag, in a database.

RFID technology, as the fundamental technology for ubiquitous services, has proliferated rapidly in recent years and its widespread applications can be seen everywhere, ranging from libraries to supply chains. Despite the advantages and promises, this technology has raised many potential aspects for abuse. Important security concerns associated with the future ubiquitous-computing world, are Threats to privacy of consumers. In recent years, the cryptographic community have made an extensive effort to recommend mechanisms to respond to or at least limit such misuses. roughly speaking, the aim of all researches in this area is to design a privacy friendly mutual authentication protocol to be consistent with the inherent limitations of lightweight

environments.

Therefore, the primal requirements for a mutual authentication protocol are resistance against traceability attacks, desynchronization attacks, and full disclosure attacks. The full disclosure attack i.e. the recovery of all the static and dynamic secrets stored on tag, is the strongest one which if is successfully applied to a protocol, implies all the other types of the attacks.

Although all RFID protocols are categorized as lightweight protocols in the general expression, Chien [2] introduced a more accurate classification for such protocols based on the computational cost and the operations supported on tag side. The RFID protocols are categorized in four classes *full-fledged*, *simple*, *lightweight*, and *ultralightweight*, in descending order.

The most lightweight category (ultralightweight protocols) which we focus on in this paper, refers to the protocols that only involve simple bitwise operations (like XOR, AND, OR, modular addition, etc.) on tags. Ultralightweight authentication protocols first introduced by Peris-Lopez et al. as UMAP family [5],[6], and [7], though they were named later by Chien [2] as Ultralightweight. Since that, lots of protocols in this category are proposed. Due to the not-so-long life of these protocols and considering many constrains that the designers of such protocols are faced to, almost all of them have been cryptanalysed successfully. Except a few cases [1] and [19], most of these attacks are based on ad-hoc methods that are not extensible to a large class of ultralightweight protocols. The approach of [1] is based on building progressive knowledge on the tag's *ID* given a series of observations and is applied successfully to

• Zahra Ahmadian and Mohammad Reza Aref are with the Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran. Mahmoud Salmasizadeh is with the Electronic Research Center, Sharif University of Technology, Tehran, Iran.
E-mail: ahmadian@ee.sharif.edu, salmasi@sharif.edu, aref@sharif.edu

SASI [2] and Yeh et al. protocol [14]. [19] introduces a method for full disclosure attack of all secrets of the protocol using Grobner basis where successful attacks on a number of ultralightweight protocols, including UMAP family [5], [6], [7], and SASI [2] is presented.

1.1 Our contributions

In this paper, we introduce new frameworks for evaluating ultralightweight authentication protocols. We call them *recursive linear* and *recursive differential* cryptanalysis which are very efficient and powerful techniques for full disclosure of all secrets of a large class of ultralightweight authentication protocols. Both of them exploit triangular functions in ultralightweight protocols and recover all secret data stored in the tag in a recursive manner.

We first introduce the outline of recursive linear cryptanalysis. Yeh et al. and SLMAP authentication protocols are two examples of ultralightweight protocols which are successfully cryptanalysed by this technique. This attack is passive, deterministic (i.e. the attacker can retrieve all the secrets with probability of one), and requires only a single authentication session.

In the following we introduce the recursive differential cryptanalysis. This attack is more powerful and can be applied to the protocols which linear attack may not work on. We show the effectiveness of this attack on $LMAP^{++}$ and SASI authentication protocols. The recursive differential attack is probabilistic, active in the sense that the attacker suffices only to block some specific messages, and requires a few authentication sessions.

1.2 Outline

The rest of the paper is organized as follows: In section 2 we briefly describe the most important features of ultralightweight authentication protocols. In section 3 we present the recursive linear cryptanalysis and apply it to two well known protocols: SLMAP and Yeh et al. protocol. In section 4 we describe the framework of recursive differential attack and apply it to two other important ultralightweight protocols: $LMAP^{++}$ and SASI. Finally we conclude our work in section 5.

2 ULTRALIGHTWEIGHT AUTHENTICATION PROTOCOLS.

Since the introduction of ultralightweight authentication protocols by Peris-Lopez et al. [5],[6], and [7], lots proposals have been presented in this field. Avoine et al. have made a comprehensive survey on the most important proposed ultralightweight protocols [1]:

Use of index pseudonym. In all ultralightweight authentication protocols, the tag uses a preshared temporary index pseudonym, namely IDS , to identify itself to the reader. Though, two parties have shared a static identifier, ID , but it is never sent in clear on the channel. Since its clear transmission results in compromising the

tag's owner privacy. IDS is updated in both sides at the end of each session.

Messages pattern. All the ultralightweight protocols share the same pattern of exchanged messages. The static ID has been preshared between two parties. In addition, The tag shares with the reader a dynamic session key K as well as the dynamic index pseudonym IDS allocated to each session. The pair (K, IDS) is called the *state* of the tag. The reader first sends a Hello message to the reader to which the tag answers with its current index pseudonym IDS . The reader searches its database indexed by its IDS to find the secrets associated with it in the database. Then, the reader generates a nonce, namely n . It then sends a message $M_R = f(IDS, ID, k, n)$ to the tag. M_R contains two parts, the first one wraps n with a key dependent quantity, and the second part is a parameter by which the tag can authenticate the reader. Upon authenticating the reader, the tag sends the message $M_T = g(IDS, ID, k, n)$ to the reader, which is used by reader to authenticate the tag. Both parties then update their state with $(k, IDS) \leftarrow h(IDS, ID, k, n)$.

Desynchronization attack prevention. The state update procedure is performed at the end of each session. i.e. when the last message is transmitted. The party that sends the last message, update its state immediately after sending the message without being assured that this message is received correctly by the other party or not. the party that receives the last message update its state upon correct receipt of this message. Therefore if the second party could not get the last message correctly, it does not update its state while the other one has done; resulting a desynchronization between two parties. In order to avoid this potential problem, the party that first updates its state (i.e. the transmitter of the last message), must keep a backup of its previous state.

In the case that the tag keeps the backup data, the tag first identifies itself by its fresh IDS at the beginning of next session. If the reader finds an entry indexed by IDS in its database, the protocol proceeds with this index pseudonym and associated dynamic secrets otherwise the reader sends to the tag another Hello message, to which the tag will reply with IDS^{old} . This old indexed-pseudonym will be definitely accepted by the reader. In the case that the reader keeps the backup data, the tag identifies itself by the only IDS that it has. The reader keeps both old and new IDS 's in its database and can extract associated dynamic secrets whether it is fresh or not. Then the protocol proceeds with the tag's IDS and relevant secrets.

Though this trick is very consistent with ultralightweight environments' constraints, But it gives the attacker the opportunity to force two parties to work in their old state. This property is exploited in recursive differential cryptanalysis, as will be described in section 4.

The other drawback of almost all of the ultralightweight protocols is extensive use of triangular functions (T-functions). T-functions, first introduced in [22],

are bijective functions with the property of $(f(x))_i = f_i(x_i, x_{i-1}, \dots, x_0)$ i.e. the i^{th} bit of output depends on $i = 0, \dots, i$ bits of input word and it does not depend on the more significant input bits. This feature is widely exploited in both recursive linear and differential attacks. As we will see in next two sections, this property gives the attacker the possibility to recover secret variables one by one, starting from the LSB.

2.1 Notations and Definitions

In the rest of the paper, the logical operations OR, AND, XOR and left rotation are denoted by \vee , \cdot , \oplus and \lll respectively. The addition and subtraction in modulo 2^n is denoted by $+$ and $-$ respectively, where $n = 96$ is length of all variables. The i^{th} bit of n bit word X is shown as X_i where $0 \leq i \leq n - 1$. In this notation, the least significant bit (LSB) of X is X_0 and the most significant bit (MSB) is X_{n-1} . The index i is always taken modulo n but the term "mod n " is eliminated for convenience.

The binary majority function, $Maj(a, b, c)$ is defined as follows:

$$Maj(a, b, c) = a \cdot b \oplus a \cdot c \oplus b \cdot c$$

For modular addition we adopted the following notations:

$$\begin{aligned} Z &= X + Y \pmod{2^n} \\ Z_i &= X_i \oplus Y_i \oplus car_i \end{aligned}$$

for $i = 0, \dots, n - 1$. where car_i is the carry that results from the $(i - 1)^{th}$ bits and sums up in the i^{th} position.

$$\begin{aligned} car_0 &= 0, \\ car_i &= Maj(X_{i-1}, Y_{i-1}, car_{i-1}) \end{aligned}$$

Similar notation is used for modular subtraction:

$$\begin{aligned} Z &= X - Y \pmod{2^n} \\ Z_i &= X_i \oplus Y_i \oplus bar_i \end{aligned}$$

for $i = 0, \dots, n - 1$. where bar_i is the barrow that results from the $(i - 1)^{th}$ bits and subtracts from the i^{th} position.

$$\begin{aligned} bar_0 &= 0, \\ bar_i &= Y_{i-1} \oplus bar_{i-1} \oplus Maj(X_{i-1}, Y_{i-1}, bar_{i-1}) \end{aligned}$$

Table 1 shows the list of notations and definitions.

3 RECURSIVE LINEAR CRYPTANALYSIS

Recursive linear cryptanalysis directly exploits T-functions in ultralightweight protocols. In this attack, the attacker creates a system of independent linear equations for each bit of secret variables. Then she starts solving this system of equations from LSB and retrieves all secret data bits recursively. The steps of the attack is described in more detail below.

TABLE 1: list of notations and definitions

Symbol	Definition
\vee	bitwise OR
\cdot	bitwise AND
\oplus	bitwise XOR
\lll	left rotation
n	Length of all variables, $n = 96$
$+$	addition in modulo 2^n
$-$	subtraction in modulo 2^n
X_i	the i^{th} bit of the variable X , $0 < i < n - 1$
Maj	Majority function, $Maj(a, b, c) = a \cdot b \oplus a \cdot c \oplus b \cdot c$
car	carry for modular addition, $Z = X + Y \pmod{2^n} \Rightarrow Z_i = X_i \oplus Y_i \oplus car_i$
bar	barrow for modular subtraction, $Z = X - Y \pmod{2^n} \Rightarrow Z_i = X_i \oplus Y_i \oplus bar_i$

- 1) Determine all the unknown variables (static and dynamic secrets and nonces) for a single session of the protocol.
- 2) Write a *linear* representation for the i^{th} bit of each message, involving the i^{th} bit of known and unknown variables. You may need to define *intermediate variables* such as carries for modular additions or barrows for modular subtractions. Try to find enough independent linear equations for the i^{th} bit involving all unknown variables or a subset of them. In other words, for each bit you have created a system of linear equations in unknown variables with nonsingular coefficient matrix. (the linear property of the attack)
- 3) Solve these systems of equations, Starting from the LSB. The LSB of carries and borrows are zero so the unique solution of the system directly yields the LSB of unknown variables. Now you can compute the next bits of intermediate variables which should be known when you solve the system of equations of next bit. In general, when solving the i^{th} system of equations, the value of i^{th} bit of intermediate variables are needed which have been calculated in the previous step $i - 1$. (the recursive property of the attack)

Therefore, the linear cryptanalysis recovers all secret data bits one by one with probability of one, provided the existence of enough independent linear equations.

It should be noted that the recursive linear cryptanalysis is a deterministic method i.e. it recovers the secret data with success probability of one. This terminology is adopted due to the linear essence of the attack as well as its recursive nature in recovering the secret data bits. However, the term "linear attack" [20] in symmetric cryptography refers to a well known technique in cryptanalysis of symmetric primitives. This technique is based on finding linear *approximations* for nonlinear relations of the cipher with maximum possible bias. This method is probabilistic in the sense that the attacker success

ratio depends on the amounts of available data. Anyway, the recursive linear cryptanalysis of Ultralightweight protocols presented in this paper should not be mixed up with the linear cryptanalysis of symmetric primitives.

In next subsections the performance of recursive linear cryptanalysis for the full disclosure attacks on two ultralightweight protocols is shown.

3.1 Recursive linear cryptanalysis of SLMAP

3.1.1 History

In 2007, Li and Wang proposed the ultralightweight mutual authentication protocol SLMAP intended for very low-cost RFID tags [12]. This protocol is analysed by Hernandez-Castro et al. in [13] by a traceability attack.

3.1.2 Protocol specifications

Each tag shares a unique static identifier ID , a public index-pseudonym IDS and two session keys $K1$ and $K2$ with the reader. The length of each of ID , IDS , $K1$ and $K2$ is $n = 96$ bits. The protocol works as follows:

1. *Reader* \rightarrow *Tag* : *Hello*
2. *Tag* \rightarrow *Reader* : IDS

The reader uses the received IDS as a search index to extract the secret information linked to the tag i.e. $(ID, K1, K2)$.

3. *Reader* \rightarrow *Tag* : A, B

The reader generates a n -bit nonces, r , and computes two messages A and B as follows.

$$\begin{aligned} A &= (IDS \oplus K1) + r \\ B &= IDS + (K2 \oplus r) \end{aligned}$$

4. *Reader* \rightarrow *Tag* : C

Upon receiving the messages A and B , the tag first extracts r from A then checks the correctness of B . If the reader is authenticated, the tag computes the response value C as follows.

$$C = (IDS + (ID \oplus r)) \oplus (K1 + r) \oplus (K2 + r)$$

After sending C , the tag updates its pseudonym and secret keys.

Upon receiving C , the reader uses its local values to verify it. If the tag is authenticated the reader updates its pseudonym and secret keys. The pseudonym and keys updating are proceed as follows in both parties.

$$\begin{aligned} IDS_n &= ((IDS + (K1 \oplus r) + ID + K2) \oplus r \\ K1_n &= (K1 \oplus r) + (IDS_n + K2 + ID) \\ K2_n &= (K2 \oplus r) + (IDS_n + K1 + ID) \\ (IDS, K1, K2) &= (IDS_n, K1_n, K2_n) \end{aligned}$$

3.1.3 Cryptanalysis

Here we apply the three stages of our linear cryptanalysis on the SLMAP protocol.

- 1) For a single session all the secret data include $K1, K2, ID$ and r .
- 2) Each message in SLMAP protocol, i.e. A, B, C, IDS_n and D , provides a linear equation. Among them, equations given by A, B, C , and IDS_n are enough for constructing a system of independent linear equations in secret data bits.

$$\begin{aligned} A &= (IDS \oplus K1) + r \\ B &= IDS + (K2 \oplus r) \\ C &= (IDS + (ID \oplus r)) \oplus (K1 + r) \\ &\quad \oplus (K2 + r) \\ IDS_n &= ((IDS + (K1 \oplus r) + ID + K2) \oplus r \end{aligned}$$

Resulting in the following linear equation for bit $i = 0, \dots, n - 1$: (For more clarity, unknown variables are in bold)

$$\begin{aligned} A_i &= IDS_i \oplus \mathbf{K1}_i \oplus \mathbf{r}_i \oplus car1_i \\ B_i &= IDS_i \oplus \mathbf{K2}_i \oplus \mathbf{r}_i \oplus car2_i \\ C_i &= IDS_i \oplus \mathbf{ID}_i \oplus \mathbf{r}_i \oplus \mathbf{K1}_i \oplus \mathbf{K2}_i \oplus car3_i \\ &\quad \oplus car4_i \oplus car5_i \\ IDS_{ni} &= IDS_i \oplus \mathbf{K1}_i \oplus \mathbf{K2}_i \oplus \mathbf{ID}_i \oplus car6_i \\ &\quad \oplus car7_i \oplus car8_i \end{aligned} \quad (1)$$

Where $car1_0 = car2_0 = \dots = car8_0 = 0$, and for $i = 1, \dots, n - 1$:

$$\begin{aligned} car1_i &= Maj(IDS_{i-1} \oplus K1_{i-1}, r_{i-1}, car1_{i-1}) \\ car2_i &= Maj(IDS_{i-1}, K2_{i-1} \oplus r_{i-1}, car2_{i-1}) \\ car3_i &= Maj(IDS_{i-1}, ID_{i-1} \oplus r_{i-1}, car3_{i-1}) \\ car4_i &= Maj(K1_{i-1}, r_{i-1}, car4_{i-1}) \\ car5_i &= Maj(K2_{i-1}, r_{i-1}, car5_{i-1}) \\ car6_i &= Maj(IDS_{i-1}, K1_{i-1} \oplus r_{i-1}, car6_{i-1}) \\ car7_i &= Maj(IDS_{i-1} \oplus K1_{i-1} \oplus r_{i-1} \oplus car6_{i-1}, \\ &\quad ID_{i-1}, car7_{i-1}) \\ car8_i &= Maj(IDS_{i-1} \oplus K1_{i-1} \oplus r_{i-1} \oplus car6_{i-1} \\ &\quad \oplus ID_{i-1} \oplus car7_{i-1}, K2_{i-1}, car8_{i-1}) \end{aligned}$$

- 3) Equations given by (1) create a system of linear equations with the following matrix representation:

$$M \cdot \begin{pmatrix} \mathbf{K1}_i \\ \mathbf{K2}_i \\ \mathbf{r}_i \\ \mathbf{ID}_i \end{pmatrix} = V \quad (2)$$

Where

$$M = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix}$$

$$V = \begin{pmatrix} A_i \oplus IDS_i \oplus car1_i \\ B_i \oplus IDS_i \oplus car2_i \\ C_i \oplus IDS_i \oplus car3_i \oplus car4_i \oplus car5_i \\ IDS_{n_i} \oplus IDS_i \oplus car6_i \oplus car7_i \oplus car8_i \end{pmatrix}$$

The coefficient matrix in (2), M , is nonsingular and therefore invertible in $GF(2)$.

$$M^{-1} = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix}$$

In each step i one can obtain the missing bits, recalling that the intermediate variables (carries) in the right side of equation have been calculated in the previous step $i - 1$.

Attack summery. The attack is *passive*, retrieves all secret data of the tag, and eavesdrops only *one* genuine authentication session. We have simulated this attack and verified its soundness.

Exclusive use of T-functions in this protocol, makes it vulnerable against recursive linear cryptanalysis. One way to avoid the exclusive use of T-functions in ultralightweight environments, first introduced by Chien [2], is use of rotations which are non triangular ultralightweight operations. We show in next subsection that use of this operation does not necessarily frustrate the effectiveness of recursive linear attack.

3.2 Recursive linear cryptanalysis of Yeh et al. Protocol

3.2.1 History

Yeh et al. proposed a process-oriented ultralightweight protocol in RFIDSec Asia 2010 [14]. The first attacks presented by Peris-Lopez et al. [15] in 2010, were traceability and passive full disclosure of ID , which requires eavesdropping of an average of 250 sessions. In 2012, Avoine et al. presented a passive full disclosure of ID on this protocol which requires eavesdropping of an average of 25 sessions [1].

3.2.2 Protocol specifications

Each tag shares a unique static identifier ID , a public index-pseudonym IDS and a session key K with the reader. The length of each of ID , IDS and K is $n = 96$ bits. To overcome de-synchronization attack, the reader keeps IDS^{old} too, which is the pseudonym of the last successful session. The protocol considers a status bit $flag$ indicating the success or fail of completion of the previous protocol session (respectively $flag = 0$ and $flag = 1$). The protocol works as follows:

1. *Reader* \rightarrow *Tag* : Hello
2. *Tag* \rightarrow *Reader* : IDS
3. *Reader* \rightarrow *Tag* : $A, B, C, flag$

The reader uses the received IDS as a search index to find out if the received IDS is new or it is an old pseudonym. The session key depends on the freshness of IDS . In other words:

- $K = K_t$ and $flag = 0$ if IDS is a new pseudonym.
- $K = ID$ and $flag = 1$ if IDS is an old pseudonym.

Then the reader generates two n -bit nonces, $n1$ and $n2$, and computes three messages A, B and C as follows.

$$\begin{aligned} A &= IDS \oplus K \oplus n1 \\ B &= (IDS \vee K) \oplus n2 \\ \bar{K} &= (K \oplus n2) \lll (n1 \bmod n) \\ C &= (\bar{K} \oplus n1) + n2 \end{aligned}$$

4. *Reader* \rightarrow *Tag* : D

Upon receiving the messages A, B, C , the tag first extracts $n1$ from A , extracts $n2$ from B , computes \bar{K} and then verifies the value of C . If the verification succeeds, the reader is authenticated and the tag computes the response value D as follows.

$$\begin{aligned} \bar{K}' &= (K \oplus n1) \lll (n2 \bmod n) \\ D &= (\bar{K}' \oplus n2) + n1 \end{aligned}$$

Upon receiving D , the reader uses its local values to verify its correctness. If the tag is authenticated, the reader updates its pseudonym and secret key. The tag and the reader compute the next IDS , namely IDS_n . The Pseudonym and key updating are proceed as follows:

$$IDS_n = (IDS + (ID \oplus \bar{K}')) \oplus n1 \oplus n2$$

Then, the reader updates old and next versions of (IDS, K) as follows:

$$\begin{aligned} (IDS, K) &= (IDS_n, \bar{K}) \\ IDS^{old} &= IDS \end{aligned} \quad (3)$$

After updating, the reader sends an *update command* to the tag.

5. *Reader* \rightarrow *Tag* : *update command*

Upon receiving it, the tag updates (IDS, K) in the same vein as (3) for the next session.

3.2.3 Cryptanalysis

This attack works for $f = 1$ where due to the equation $K = ID$, number of unknown variables is one less.

- 1) For a single session all the secret data include $n1, n2, K, \bar{K}, \bar{K}'$ and ID . In next step, We create a system of equations involving three unknown variables $n1, n2, ID$ only.

- 2) The attacker waits for a session where $f = 1$ and saves the associated messages A, B, D and IDS_n . Given $K = ID$,

$$A = IDS \oplus ID \oplus n1 \quad (4)$$

$$B = (IDS \vee ID) \oplus n2 \quad (5)$$

$$D = (\overline{K'} \oplus n2) + n1 \quad (6)$$

$$IDS_n = (IDS + (ID \oplus \overline{K'})) \oplus n1 \oplus n2 \quad (7)$$

Eq. (6) and (7) hold:

$$IDS_n = (IDS + (ID \oplus n2 \oplus (D - n1)) \oplus n1 \oplus n2 \quad (8)$$

The following linear representations for the i^{th} bit of (4), (5), and (8) are sufficient for constructing a system of linear equations with nonsingular coefficient matrix:

$$A_i = IDS_i \oplus \mathbf{ID}_i \oplus \mathbf{n1}_i$$

$$B_i = (IDS_i \vee \mathbf{ID}_i) \oplus \mathbf{n2}_i$$

$$= (IDS_i \oplus 1) \cdot \mathbf{ID}_i \oplus IDS_i \oplus \mathbf{n2}_i$$

$$IDS_{ni} = IDS_i \oplus \mathbf{ID}_i \oplus D_i \oplus car_i \oplus bar_i \quad (9)$$

Where

$$car_0 = 0$$

$$bar_0 = 0$$

$$bar_i = n1_{i-1} \oplus bar_{i-1}$$

$$\oplus Maj(D_{i-1}, n1_{i-1}, bar_{i-1})$$

$$car_i = Maj(IDS_{i-1}, ID_{i-1} \oplus n2_{i-1} \oplus D_{i-1}$$

$$\oplus n1_{i-1} \oplus bar_{i-1}, car_{i-1})$$

- 3) Equations given by (9) hold a system of linear equations with the following matrix representation:

$$\mathbf{M} \cdot \begin{pmatrix} \mathbf{ID}_i \\ \mathbf{n1}_i \\ \mathbf{n2}_i \end{pmatrix} = \mathbf{V} \quad (10)$$

Where

$$\mathbf{M} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ IDS_i \oplus 1 & 0 & 1 \end{pmatrix}$$

$$\mathbf{V} = \begin{pmatrix} IDS_{ni} \oplus IDS_i \oplus D_i \oplus car_i \oplus bar_i \\ A_i \oplus IDS_i \\ B_i \oplus IDS_i \end{pmatrix}$$

The coefficient matrix \mathbf{M} is nonsingular, whether $IDS_i = 0$ or $IDS_i = 1$. Therefore it is invertible in $GF(2)$ in any case.

$$\mathbf{M}^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \text{ when } IDS_i = 0.$$

$$\mathbf{M}^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ when } IDS_i = 1.$$

In step i the solution of the associated system gives directly the i^{th} bit of unknown variables $ID, n1, n2$. once again, note that the values of car_i and bar_i in the right side of (9) have been calculated in the previous step $i - 1$.

After recovering all bits of $ID, n1, n2$, the remaining unknown variables, $\overline{K}, \overline{K'}$ can be computed straightforwardly.

Attack summary. This attack is *passive*, retrieves all *secret data* of the tag, and eavesdrops only *one* genuine authentication session with $flag = 1$. The correctness of this attack have been also verified by simulation.

4 RECURSIVE DIFFERENTIAL CRYPTANALYSIS

As described in section 3 recursive linear cryptanalysis is a straightforward an efficient manner for full disclosure attack on some ultralightweight protocols. This attack is deterministic and its success essentially depends on the existence of enough number of independent equations in unknown variables or a subset of them.

However, in some protocols, the attacker may not be so lucky to easily find such system of equations. Often in these cases, the number of equations of a session is less than the number of unknown variables. Use of messages of one or more new sessions is neither an effective solution since new sessions bring new unknown variables as many as or even more than new equations. In such scenarios, a more powerful attack that can generate enough independent equations is required.

Recursive differential cryptanalysis, presented in this section is a more effective technique for cryptanalysis of such protocols. Roughly speaking, in this attack, the attacker forces two parties to run new sessions in their *previous state*. This trick limits the introduction of new variables, while giving new equations. more precisely, when the protocol runs in the same state, all the dynamic secret variables stay the same and only new nonces are generated in each protocol run. Moreover, new nonces have a clear *differential* relation (XOR difference or modular addition difference) with the old ones that can be efficiently used for creating new independent equations. This differential relation often can be find because the wrapped nonces generated by reader are usually delivered to the tag in the initial messages of the protocols in such way that it could be extractable for the tag. Therefore it is often xored or added with a secret dependent quantity. Though the attacker does not know this quantity, but she can obtain differences of two nonces provided that they both wrapped with the same secrets. The name of the attack is derived from the important role of differences.

Of course, enforcement of two parties to work in their previous state demands a kind of *active* attacker. To see

how this process is possible, we recall from section 2 that in order to avoid possible desynchronization, the party that first updates its state (i.e. the transmitter of the last message), must keep a backup of its previous state. In the case that the tag keeps the backup data, the tag first identifies itself by its fresh *IDS* at the beginning of each session. If the reader finds an entry indexed by *IDS* in its database, the protocol proceeds with this indexed-pseudonym and associated dynamic secrets otherwise the reader sends to the tag another Hello message, to which the tag will reply with IDS^{old} . This old indexed-pseudonym will be definitely accepted by the reader. In the case that the reader keeps the backup data, the tag identifies itself by the only *IDS* that it has. The reader keeps both old and new *IDS*'s in its database and can extract associated dynamic secrets whether it is fresh or not. Then the protocol proceeds with the tag's *IDS* and relevant secrets.

Therefore, to force the protocol to stay in the same state, the attacker can easily block the last message of the protocol. (In the case that the tag keeps the backup data, the attacker can do this by blocking the *IDS* message sent by the tag at the beginning of the protocol, too. Thereupon, the reader asks the tag to use its old *IDS* and new session will be run with IDS^{old} and k^{old} .) furthermore, the attacker can repeat this scenario as many times as required to force the parties to work on the same state for any arbitrary number of sessions.

Thus, from this perspective, this attack is a kind of active attack. However, the attacker is a *weak* active attacker in the sense that she only blocks some specific messages rather than modifying or generating new messages (this weak attacker should not be confused with weak adversary defined in privacy models [17] i.e. the adversary who is not allowed to corrupt tags).

Recursive differential cryptanalysis is outlined as follows:

Phase 1. Data gathering

Allow two parties to run $s + 1$ genuine consecutive sessions $S^{(j)}, j = 0, \dots, s$, and block the last message of each of them to enforce parties to run all sessions in the same state as $S^{(0)}$. Save all message corresponding to these sessions.

Phase 2. Secret Recovery

- 1) Determine all the unknown variables (static and dynamic secrets and nonces) for the first session $S^{(0)}$. Determine also new unknown variables (nonces only) for the next sessions $S^{(j)}, j = 1, \dots, s$. Express clearly the (XOR or modular addition) differential relation of the nonces of $S^{(0)}$ and their counterparts in $S^{(j)}$.
- 2) Write the linear expansion of a proper message of $S^{(0)}$ for bit i possibly containing some intermediate variables (carries for modular additions or barrows for modular subtractions). Write the linear expansion for the corresponding message of $S^{(j)}$, too. The first one contains a subset of unknown variables, while the latter contains the same unknown

variables as well as differences of the nonces. The differences of these two linear equation results in a linear equation in the $(i - 1)^{th}$ bit of secrets with *random coefficients*. (the differential property of the attack)

- 3) Each differential pair corresponding to sessions $(S^{(0)}, S^{(j)}), j = 1, \dots, s$ provides such a linear equation in bit $i - 1$. Therefore there are s linear equations for each bit $i = 0, \dots, n - 2$. With a sufficient number of equations, and due to the randomness of coefficients, there will be an over defined system of linear equations for each bit that yields unknown variables uniquely. Solve the systems of equations starting from the LSB where intermediate variables (carries and borrows) are all zero. Then, compute the intermediate variables for next bit position. In general, in order to obtain the i^{th} bit of secret data, the i^{th} bit of intermediate variables need to be known, which have already computed in previous step. (the recursive property of the attack)

Therefore, the linear cryptanalysis recovers all secret data bits one by one with probability close to one, provided that there are sufficient number of successive sessions whose last messages is blocked.

It should be mentioned that this terminology is adopted due to the differential essence of the attack as well as its recursive nature in recovering the secret data bits. However, the term "differential attack" [21] in symmetric cryptography refers to a well known technique in cryptanalysis of symmetric primitives. This technique is based on finding *differential characteristics* with maximum possible probability and then combining them in an appropriate manner. This method is probabilistic in the sense that the attacker success ratio depends on the amounts of available data. Anyway, the recursive differential cryptanalysis of Ultralightweight protocols presented in this paper should not be mixed up with the differential cryptanalysis of symmetric primitives.

4.1 Recursive differential cryptanalysis of LMAP⁺⁺ Protocol

4.1.1 History

In 2008, Li proposed a new ultralightweight authentication protocol and called it *LMAP⁺⁺* as an improved of previous proposals [10]. In 2011, it received two cryptanalysis: the first one proposed by Safkhani et al. [9] that described a traceability an a desynchronization attack on *LMAP⁺⁺*, and the second one was a passive full disclosure of all secrets presented by Wang and Zhang [11] that requires eavesdropping about 480 genuine sessions.

4.1.2 Specification

Each tag shares a unique static identifier *ID*, a public index-pseudonym *PID* and two session keys *K1* and

$K2$ with the reader. The length of each of $ID, IDS, K1$ and $K2$ is $n = 96$ bits. The protocol works as follows:

1. *Reader* \rightarrow *Tag* : *Hello*
2. *Tag* \rightarrow *Reader* : *PID*

The reader uses the received *PID* as a search index to extract the secret information linked to the tag i.e. $(ID, K1, K2)$.

3. *Reader* \rightarrow *Tag* : A, B

The reader generates a n -bit nonces, r , and computes two messages A and B as follows.

$$\begin{aligned} A &= (PID \oplus K1) + r \\ B &= PID + (K2 \oplus r) \end{aligned} \quad (11)$$

4. *Tag* \rightarrow *Reader* : C

Upon receiving the messages A and B , the tag first extracts r from A then checks the correctness of B . If the reader is authenticated, the tag computes the response value C as follows.

$$C = (PID + (ID \oplus r)) \oplus (K1 + K2 + r) \quad (12)$$

Upon receiving C , the reader uses its local values to verify it. If the tag is authenticated, the reader updates its pseudonym and secret keys as follows:

$$\begin{aligned} PIDn &= ((IDS + K1) \oplus r) + ((ID + K2) \oplus r) \\ K1n &= (K1 \oplus r) + (PIDn + K2 + ID) \\ K2n &= (K2 \oplus r) + (PIDn + K1 + ID) \\ (PID, K1, K2) &= (PIDn, K1n, K2n) \end{aligned} \quad (13)$$

and to avoid desynchronisation, it keeps the previous state, too.

$$(PID^{old}, K1^{old}, K2^{old}) = (PID, K1, K2)$$

Then the reader sends a *Completion message* to the tag.

5. *Reader* \rightarrow *Tag* : *Completion message* (14)

Upon receiving *Completion message*, the tag updates its pseudonym and secret keys in the same vein as (13).

4.1.3 Cryptanalysis

In this section we show the effectiveness of our recursive differential cryptanalysis on $LMAP^{++}$ protocol.

Phase 1. Data gathering

Allow two parties to run $s + 1$ genuine consecutive sessions $\mathcal{S}^{(j)}$, $j = 0, \dots, s$, and block the fifth message (14) of each of them. Save all message corresponding to these sessions.

Phase 2. Secret recovery

- 1) For a clear description of this attack, we first consider the first two consecutive sessions, $\mathcal{S}^{(0)}$ and $\mathcal{S}^{(1)}$. The variables associated with $\mathcal{S}^{(1)}$ is denoted by X' where X is its counterpart in $\mathcal{S}^{(0)}$. For session $\mathcal{S}^{(0)}$ all the unknown data include $r, K1, K2$ and

ID . In session $\mathcal{S}^{(1)}$, the variables $K1, K2$ and ID stay the same (i.e. $K1' = K1, K2' = K2, ID' = ID$) but r' is new generated. At first, we describe the attack basis for a single differential pair corresponding to sessions $(\mathcal{S}^{(0)}, \mathcal{S}^{(1)})$, then extend it to more differential pairs of $(\mathcal{S}^{(0)}, \mathcal{S}^{(j)})$, $j = 1, \dots, m$ to complete the description of the attack scenario. The modular addition difference of two nonces r and r' is obtained from equations given by A and A' .

$$\begin{aligned} A &= (PID \oplus K1) + r \\ A' &= (PID \oplus K1) + r' \end{aligned}$$

Hence the following difference:

$$\begin{aligned} r' &= (A' - (PID \oplus K1)) \\ &= A' - (A - r) \\ &= r + \alpha \end{aligned} \quad (15)$$

Where $\alpha = A' - A$ is a known variable. Bit representation of (15) is as follows. (For more clarity, unknown variables are in bold):

$$\mathbf{r}'_i = \mathbf{r}_i \oplus \alpha_i \oplus \mathbf{car}_i \quad (16)$$

Where $\mathbf{car}_0 = 0$ and $\mathbf{car}_i = \mathit{Maj}(r_{i-1}, \alpha_{i-1}, \mathbf{car}_{i-1})$ for $i = 1, \dots, n - 1$.

- 2) We use messages B and B' to construct the linear equation.

$$\begin{aligned} B &= PID + (K2 \oplus r) \\ B' &= PID + (K2 \oplus r') \end{aligned}$$

With the following bit representations:

$$X_i = \mathbf{K2}_i \oplus \mathbf{r}_i \quad (17)$$

$$\begin{aligned} X'_i &= \mathbf{K2}_i \oplus \mathbf{r}'_i \\ &= \mathbf{K2}_i \oplus \mathbf{r}_i \oplus \alpha_i \oplus \mathbf{car}_i \end{aligned} \quad (18)$$

Where $X = B - PID$ and $X' = B' - PID$ are known. The XOR difference of (17) and (18) eliminates the unknown variables $\mathbf{K2}_i$ and r_i and yields the following one variable linear equation in r_{i-1} :

$$\Delta X_i = \mathbf{car}_i \oplus \alpha_i$$

where

$$\mathbf{car}_i = \mathbf{r}_{i-1} \cdot (\alpha_{i-1} \oplus \mathbf{car}_{i-1}) \oplus \alpha_{i-1} \cdot \mathbf{car}_{i-1}$$

Hence:

$$p_i \cdot \mathbf{r}_i = \alpha_i \cdot \mathbf{car}_i \oplus \Delta X_{i+1} \oplus \alpha_{i+1} \quad (19)$$

for $i = 0, \dots, n - 2$. Where the coefficient $p_i = \alpha_i \oplus \mathbf{car}_i$ is known in step i . This linear equation yields the unknown variable r_i if $p_i = 1$.

The coefficient p_i is a uniform binary random variable since α_i is uniform. Therefore, for a single differential pair of sessions $(\mathcal{S}^{(0)}, \mathcal{S}^{(1)})$, the probability of $\Pr\{p_i = 1\} = \frac{1}{2}$. In order to have a reliable

full disclosure attack, we need a sufficient number of such differential pair of sessions $(\mathcal{S}^{(0)}, \mathcal{S}^{(j)})$, $j = 1, \dots, s$, each one gives a new independent equation as follows.

$$p_i^{(j)} \cdot \mathbf{r}_i = \alpha_i^{(j)} \cdot \text{car}_i^{(j)} \oplus \Delta X_{i+1}^{(j)} \oplus \alpha_{i+1}^{(j)} \quad (20)$$

where the superscript (j) denotes the index of differential pair. $p_i^{(j)}, \alpha_i^{(j)}, \text{car}_i^{(j)}$, and $\Delta X_{i+1}^{(j)}$ are the variables corresponding to differential pair of sessions $(\mathcal{S}^{(0)}, \mathcal{S}^{(j)})$ which are defined in a similar way of sessions $(\mathcal{S}^{(0)}, \mathcal{S}^{(1)})$.

- 3) With a sufficient number of equations (20), there will be at least one equation that with a probability close to one $p_i^{(j)} = 1$, for all $i = 1, \dots, n$. So, except for the MSB, all bits of r can be retrieved recursively starting from the LSB. Again note that in each step i the carry bit car_i , has been already calculated in the previous step $i-1$. Having recovered $n-1$ bits of r , there will be two answer for r which differ only in the MSB. For each one, the other unknown variables $K1, K2$, and ID can be calculated from (11) and (12) respectively. Finally, another additional equation, e.g. PID_n transmitted on the channel after the final session, can be used to verify the correct answer.

probability analysis. Here, we discuss that for a reliable full disclosure attack, how many differential pair of sessions $(\mathcal{S}^{(0)}, \mathcal{S}^{(j)})$, $j = 1, \dots, s$, is required.

The coefficients p_i 's are independent uniform binary random variables. Hence the probability of failure for retrieving a single bit, given s differential pairs is $P_f = (\frac{1}{2})^s$. Therefore, the probability of success for all $n-2$ bits is.

$$\begin{aligned} P_{succ} &= (1 - (P_f)^s)^{n-2} \\ &= \left(1 - \left(\frac{1}{2}\right)^s\right)^{n-2} \end{aligned}$$

Which implies that for a success probability of more than 0.95, about $s = 11$ differential pairs $(\mathcal{S}^{(0)}, \mathcal{S}^{(j)})$ is required. Totally, including $\mathcal{S}^{(0)}$, This attack requires twelve sessions. The theoretical success probability of the attack as well as experimental results is depicted in Fig.1 for $n = 96$. The experimental results are based on about 300 times running the simulated $LMAP^{++}$ protocol.

Attack summary. This attack is *active* (though the attacker only blocks some messages), retrieves *all secret data* of the tag, and requires only *twelve* consecutive genuine authentication sessions for probability of success more than 0.95.

4.2 Recursive differential cryptanalysis of SASI protocol

4.2.1 History

In 2007, Chien proposed one of the most important ultra-lightweight protocols called SASI (Strong Authentication and Strong Integrity) as an improvement of the UMAP

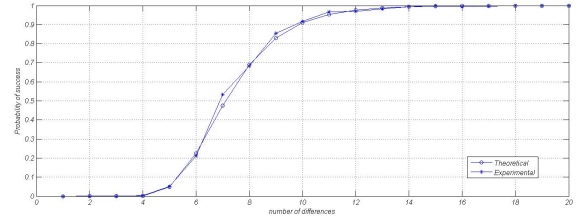


Fig. 1: Recursive differential cryptanalysis of $LMAP^{++}$. comparison of theoretical and experimental probabilities of success

protocols [2] to provide authenticity and integrity and withstand all the possible attacks to which the UMAP protocols are subject. The main contributions of this proposal were introduction of rotations as non triangular operations and desynchronization prevention by storing the previous state in the tag.

SASI received some successful cryptanalysis since its introduction. In 2008, Two traceability attacks proposed on SASI by Cao et al. [3] and Phan [4], separately. One year later, Hernandez Castro et al. [18] presented a passive full disclosure of ID that recovered the $O(\log n)$ least significant bits of ID by eavesdropping n genuine successive sessions that was impractical due to its exponential data complexity. In 2009, Sun et al. presented two desynchronization attacks, one of which is similar to D' Acro and De Santis desynchronization attack in [16] in which the authors proposed an active full disclosure of ID and active full disclosure of all secrets, too. This attack requires an average of 309 communications with the tag. The only successful passive full disclosure of ID is given by Avoine et al. in 2011 [1], which recovered the ID by eavesdropping an average of 2^{17} genuine sessions.

4.2.2 Specifications

Each tag shares a unique static identifier (ID), a public index-pseudonym (IDS) and two session keys $K1$ and $K2$ with the reader. The length of each of $ID, IDS, K1$ and $K2$ is $n = 96$ bits. To resist the possible de-synchronization attack, each tag keeps $(IDS^{old}, K1^{old}, K2^{old})$ also which are the parameters associated with the last successful session. The protocol works as follows:

1. Reader \rightarrow Tag : Hello
2. Tag \rightarrow Reader : IDS

The reader uses the received IDS as a search index to to extract the secret information linked to the tag. If the reader could find a matched entry in the database, it proceeds the next step. Otherwise, it probes again and the tag responds with its old IDS .

3. Reader \rightarrow Tag : A, B, C

The reader generates two n -bit nonces, $n1$ and $n2$, and computes three messages A, B and C as follows.

$$A = IDS \oplus K1 \oplus n1 \quad (21)$$

$$B = (IDS \vee K2) + n2 \quad (22)$$

$$\overline{K1} = (K1 \oplus n2) \lll k1$$

$$\overline{K2} = (K2 \oplus n1) \lll k2$$

$$C = (K1 \oplus \overline{K2}) + (\overline{K1} \oplus K2) \quad (23)$$

where $k1 = \mathcal{H}(K1)$ and $k2 = \mathcal{H}(K2)$ and \mathcal{H} denotes the hamming weight function.

4. Tag \rightarrow Reader : D

Upon receiving the messages A, B, C , the tag first extracts $n1$ from A , extracts $n2$ from B , computes $\overline{K1}$ and $\overline{K2}$ and then verifies the value of C . If the verification succeeds, the reader is authenticated and the tag computes the response value D as follows.

$$D = (\overline{K2} + ID) \oplus ((K1 \oplus K2) \vee \overline{K1}) \quad (24)$$

After sending D , the tag updates its pseudonym and secret keys.

Upon receiving D , the reader uses its local values to verify D . If the tag is authenticated the reader updates its pseudonym and secret keys. Pseudonym and keys updating are proceed as follows in both parties. The tag and the reader compute the next IDS , namely IDS_n .

$$IDS_n = (IDS + ID) \oplus (n2 \oplus \overline{K1})$$

Then, the tag updates the old and next versions of $(IDS, K1, K2)$ as follows:

$$\begin{aligned} (IDS, K1, K2) &= (IDS_n, \overline{K1}, \overline{K2}) \\ (IDS^{old}, K1^{old}, K2^{old}) &= (IDS, K1, K2) \end{aligned}$$

While the reader updates only $(IDS, K1, K2)$ for the next session.

4.2.3 Cryptanalysis

Though recursive linear and differential attacks relies on use of T-functions in Ultralightweight protocols, recursive linear cryptanalysis of Yeh et al. protocol shows that use of rotations as non triangular operations does not necessarily cancel out the effectiveness of recursive linear attack. And now, cryptanalysis of SASI is an example of successful applying of recursive differential cryptanalysis to a protocol that uses rotations as non triangular operations.

Phase 1. Data gathering

Allow two parties to run $s + 1$ genuine consecutive sessions $\mathcal{S}^{(j)}, j = 0, \dots, s$, and block the forth message D transmitted from the reader for each session. Save all message corresponding to these sessions.

Phase 2. Secret recovery

1) We first consider the first two consecutive sessions, $\mathcal{S}^{(0)}$ and $\mathcal{S}^{(1)}$. The variables associated with $\mathcal{S}^{(1)}$ is denoted as X' where X is its counterpart in

$\mathcal{S}^{(0)}$. For session $\mathcal{S}^{(0)}$ all the unknown data include $K1, K2, n1, n2$ and ID . In session $\mathcal{S}^{(1)}$, the variables $K1, K2$ and ID stay the same (i.e. $K1' = K1, K2' = K2, ID' = ID$) but $n1'$ and $n2'$ are new generated. At first, we describe the attack basis for a single differential pair corresponding to sessions $(\mathcal{S}^{(0)}, \mathcal{S}^{(1)})$, then extend it to more differential pairs of $(\mathcal{S}^{(0)}, \mathcal{S}^{(j)})$, $j = 1, \dots, m$.

We get the XOR-differential of $n1$ and $n1'$ from (21) and modular-differential of $n2$ and $n2'$ from (22):

$$\begin{aligned} n1' &= A' \oplus IDS \oplus K1 \\ &= A' \oplus A \oplus n1 \\ &= n1 \oplus \tilde{\alpha} \end{aligned} \quad (25)$$

$$\begin{aligned} n2' &= B - (IDS \vee K2) \\ &= B - (B' - n2) \\ &= n2 + \beta \end{aligned} \quad (26)$$

Where $\tilde{\alpha} = A \oplus A'$ and $\beta = B' - B$ are known.

2) Utilizing (23) for two sessions, linear equations of some secrets can be obtained. By expanding (23):

$$\begin{aligned} C &= (K1 \lll k1 \oplus K2 \oplus n2 \lll k1) + \\ &\quad (K2 \lll k2 \oplus K1 \oplus n1 \lll k2) \\ C' &= (K1 \lll k1 \oplus K2 \oplus n2' \lll k1) + \\ &\quad (K2 \lll k2 \oplus K1 \oplus n1' \lll k2) \end{aligned}$$

Let us define new variables X, Y and α as follows:

$$\begin{aligned} X &= K1 \lll k1 \oplus K2 \\ Y &= K2 \lll k2 \oplus K1 \oplus n1 \lll k2 \\ \alpha &= \tilde{\alpha} \lll k2 \end{aligned} \quad (27)$$

Thus,

$$\begin{aligned} C &= (X \oplus n2 \lll k1) + Y \\ C' &= (X \oplus ((n2 + \beta) \lll k1) + (Y \oplus \alpha)) \end{aligned} \quad (28)$$

We first try to find enough equations to recover unknown variables X, Y and $n2$. First, assume that the correct values of $k1$ and $k2$ are known. Bit representations of equation 28 are as follows:

$$C_i = \mathbf{X}_i \oplus \mathbf{n2}_{i-k1} \oplus \mathbf{Y}_i \oplus \text{car}_i \quad (29)$$

$$\begin{aligned} C'_i &= \mathbf{X}_i \oplus \mathbf{n2}_{i-k1} \oplus \beta_{i-k1} \oplus \text{car}''_{i-k1} \\ &\quad \oplus \mathbf{Y}_i \oplus \alpha_i \oplus \text{car}'_i \end{aligned} \quad (30)$$

Where $\text{car}_0 = \text{car}'_0 = \text{car}''_0 = 0$, and

$$\begin{aligned} \text{car}_i &= \text{Maj}(X_{i-1} \oplus n2_{i-k1-1}, Y_{i-1}, \text{car}_{i-1}) \\ \text{car}''_i &= \text{Maj}(n2_{i-1}, \beta_{i-1}, \text{car}''_{i-1}) \\ \text{car}'_i &= \text{Maj}(X_{i-1} \oplus n2_{i-k1-1} \oplus \beta_{i-k1-1} \oplus \text{car}''_{i-k1-1}, \\ &\quad Y_{i-1} \oplus \alpha_{i-1}, \text{car}'_{i-1}) \end{aligned}$$

for $i = 1, \dots, n - 1$. Eq. (29) yields one of the linear equations in three variables X_i, Y_i and $n2_{i-k1}$ which we are looking for. To generate more such equations we make use of differential equations.

From (29) and (30) bit representation of differential $\Delta C_i = C_i \oplus C'_i$ is:

$$\Delta C_i = \Delta car_i \oplus car''_{i-k1} \oplus \beta_{i-k1} \oplus \alpha_i \quad (31)$$

Where $\Delta car_0 = 0$. So, for $i = 0$, (31) immediately results in:

$$car''_{n-k1} = \Delta C_0 \oplus \beta_{n-k1} \oplus \alpha_0 \quad (32)$$

That gives the value of car''_{n-k1} explicitly which is required in next stage for retrieving unknown LSB bits. For $i = 1, \dots, n-1$:

$$\begin{aligned} \Delta car_i &= car_i \oplus car'_i \\ &= \mathbf{X}_{i-1} \cdot (\Delta car_{i-1} \oplus \alpha_{i-1}) \oplus \\ &\quad \mathbf{Y}_{i-1} \cdot (\Delta car_{i-1} \oplus car''_{i-k1-1} \oplus \beta_{i-k1-1}) \oplus \\ &\quad \mathbf{n2}_{i-k1-1} \cdot (\Delta car_{i-1} \oplus \alpha_{i-1}) \oplus \\ &\quad Maj(\beta_{i-k1-1} \oplus car''_{i-k1-1}, \alpha_{i-1}, car'_{i-1}) \end{aligned} \quad (33)$$

Substitution (33) in (31), then some simplifications yields:

$$\begin{aligned} \Delta C_i &= \mathbf{X}_{i-1} \cdot (\Delta car_{i-1} \oplus \alpha_{i-1}) \oplus \\ &\quad \mathbf{Y}_{i-1} \cdot (\Delta car_{i-1} \oplus car''_{i-k1-1} \oplus \beta_{i-k1-1}) \oplus \\ &\quad \mathbf{n2}_{i-k1-1} \cdot (\Delta car_{i-1} \oplus \alpha_{i-1} \oplus car''_{i-k1-1} \oplus \beta_{i-k1-1}) \\ &\quad \oplus Maj(\beta_{i-k1-1} \oplus car''_{i-k1-1}, \alpha_{i-1}, car'_{i-1}) \\ &\quad \oplus \beta_{i-k1} \oplus \alpha_i \oplus car''_{i-k1-1} \cdot \beta_{i-k1-1} \end{aligned}$$

For $i = 1, \dots, n-1, i \neq k1$.

For $i = k1$, $car''_{i-k1} = 0$ in (31). So the expansion of (31) is slightly different:

$$\begin{aligned} \Delta C_{k1} &= \mathbf{X}_{k1-1} \cdot (\Delta car_{k1-1} \oplus \alpha_{k1-1}) \oplus \\ &\quad \mathbf{Y}_{k1-1} \cdot (\Delta car_{k1-1} \oplus car''_{n-1} \oplus \beta_{n-1}) \oplus \\ &\quad \mathbf{n2}_{n-1} \cdot (\Delta car_{k1-1} \oplus \alpha_{k1-1}) \oplus \\ &\quad Maj(\beta_{n-1} \oplus car''_{n-1}, \alpha_{k1-1}, car'_{k1-1}) \\ &\quad \oplus \beta_0 \oplus \alpha_{k1} \end{aligned}$$

Summarizing these both cases in a single equation gives the following linear equation in three variables X_i, Y_i and $n2_{i-k1}$:

$$P_i \cdot \mathbf{X}_i \oplus Q_i \cdot \mathbf{Y}_i \oplus R_{i-k1} \cdot \mathbf{n2}_{i-k1} = M_i \quad (34)$$

Where for $i = 0, \dots, n-2, i \neq k1-1$:

$$\begin{aligned} P_i &= \Delta car_i \oplus \alpha_i \\ Q_i &= \Delta car_i \oplus car''_{i-k1} \oplus \beta_{i-k1} \\ R_{i-k1} &= \Delta car_i \oplus \alpha_i \oplus car''_{i-k1} \oplus \beta_{i-k1} \\ M_i &= Maj(\beta_{i-k1} \oplus car''_{i-k1}, \alpha_i, car'_i) \oplus \beta_{i-k1+1} \\ &\quad \oplus \alpha_{i+1} \oplus car''_{i-k1} \cdot \beta_{i-k1} \oplus \Delta C_{i+1} \end{aligned}$$

and for $i = k1-1$:

$$\begin{aligned} P_{k1-1} &= \Delta car_{k1-1} \oplus \alpha_{k1-1} \\ Q_{k1-1} &= \Delta car_{k1-1} \oplus car''_{n-1} \oplus \beta_{n-1} \\ R_{n-1} &= \Delta car_{k1-1} \oplus \alpha_{k1-1} \\ M_{k1-1} &= Maj(\beta_{n-1} \oplus car''_{n-1}, \alpha_{k1-1}, car'_{k1-1}) \oplus \\ &\quad \beta_0 \oplus \alpha_{k1} \oplus \Delta C_k. \end{aligned} \quad (35)$$

each differential pair of sessions $(S^{(0)}, S^{(j)})$, results in a new linear equation with random coefficients $P_i^{(j)}, Q_i^{(j)}$ and $R_{i-k1}^{(j)}$:

$$P_i^{(j)} \cdot \mathbf{X}_i \oplus Q_i^{(j)} \cdot \mathbf{Y}_i \oplus R_{i-k1}^{(j)} \cdot \mathbf{n2}_{i-k1} = M_i^{(j)} \quad (36)$$

3) In order to recursively recover all bits of X, Y and $n2$, we require enough number of independent linear equations in $X_i, Y_i, n2_{i-k1}$. One of them is (29). The two other required equations are given from (36) for two distinct values of j . So we can construct such a system of linear equations:

$$\mathbf{M} \cdot \begin{pmatrix} \mathbf{X}_i \\ \mathbf{Y}_i \\ \mathbf{n2}_{i-k1} \end{pmatrix} = \mathbf{V} \quad (37)$$

Where

$$\mathbf{M} = \begin{pmatrix} P_i^{(j_1)} & Q_i^{(j_1)} & R_{i-k1}^{(j_1)} \\ P_i^{(j_2)} & Q_i^{(j_2)} & R_{i-k1}^{(j_2)} \\ 1 & 1 & 1 \end{pmatrix} \quad (38)$$

$$\mathbf{V} = \begin{pmatrix} M_i^{(j_1)} \\ M_i^{(j_2)} \\ C_i \oplus car_i \end{pmatrix} \quad (39)$$

Where $j_1 \neq j_2, 1 \leq j_1, j_2 \leq s$.

For s differential pair of sessions, we can build $\frac{s(s-1)}{2}$ distinct system of linear equations. For a full recovery of the unknown bits X_i, Y_i and $n2_{i-k1}$ there must be at least one system of equations with nonsingular matrix of coefficients. Let's examine the two cases of $i \neq k1-1$ and $i = k1-1$ separately. For the case $i = 0, \dots, n-2, i \neq k1-1$, all matrix entries in the first two rows are independent uniform random variables so having enough such system of equations, one can hope that at least coefficients matrix of one of them is nonsingular.

For the case $i = k1-1$, according to (35), the first and third columns of \mathbf{M} are the same for all differential pairs. In other words, for $i = k1-1$, \mathbf{M} is always nonsingular and X_{k1-1}, Y_{k1-1} and $n2_{n-1}$ could never be retrieved uniquely from (37). However we can still construct the following linear equations:

$$P_{k1-1}^{(j)} \cdot (\mathbf{X}_{k1-1} \oplus \mathbf{n2}_{n-1}) \oplus Q_{k1-1}^{(j)} \cdot \mathbf{Y}_{k1-1} = M_{k1-1}^{(j)} \quad (40)$$

Thus, the linear system of equation corresponding to $i = k1-1$ reduces to:

$$\mathbf{M} \cdot \begin{pmatrix} \mathbf{X}_{k1-1} \oplus \mathbf{n2}_{n-1} \\ \mathbf{Y}_{k1-1} \end{pmatrix} = \mathbf{V} \quad (41)$$

Where

$$\mathbf{M} = \begin{pmatrix} P_{k1-1}^{(j)} & Q_{k1-1}^{(j)} \\ 1 & 1 \end{pmatrix}$$

$$V = \begin{pmatrix} M_{k1-1}^{(j)} \\ C_{k1-1} \oplus car_{k1-1} \end{pmatrix}$$

Each differential pair of sessions yields a new system of linear equations thus there are s such matrices. The two entries in the first row of M are independent and uniformly distributed, So having enough such matrices, at least one of them is nonsingular. However, the unique values of $X_{k1-1} \oplus n2_{n-1}$ and Y_{k1-1} can be obtained from (41). Although we could not get X_{k1-1} and $n2_{n-1}$ separately, but $X_{k1-1} \oplus n2_{n-1}$ and Y_{k1-1} are solely sufficient to obtain intermediate variables for the next step $i = k1$ (i.e. car'_k and car''_k).

Hence, as long as the correct values of $k1$ and $k2$ are known, we can retrieve unknown bits of X_i, Y_i and $n2_{i-k1}$ one by one starting from the LSB except $X_{k1-1}, X_{n-1}, Y_{n-1}$, and $n2_{n-1}$. At each step the intermediate variables required for next step is calculated. For the LSB, car_0 and car'_0 are zero and car''_{n-k1} is driven from (32).

Anyway, the correct values of $k1$ and $k2$ are not known and we have to guess them from the most probable to the least probable one. $Pr(k1 = i) = Pr(k2 = i) = \frac{\binom{i}{n}}{2^n}$ Thus we guess the values of $k1$ and $k2$ according to the following order of priority:

$$k1, k2 = \frac{n}{2}, \frac{n}{2} \pm 1, \dots, \frac{n}{2} \pm \frac{n}{2}.$$

The correct guess, definitely recovers the correct values of X, Y and $n2$. But, what happens if we guess these values incorrectly?

As described, one can construct one equation of the form (36) for each difference. For the correct guess of $k1$ and $k2$, all of these s equations are valid and consistent with each other. For the wrong guesses of $k1$ or $k2$, these equations are not necessarily valid (more precisely each equation is valid with the probability of $\frac{1}{2}$) and one can find some contradictions between them which discovers the wrong guesses. For example, for bit $i = 0$ the j_1^{th} difference may result in, say $(P_0^{(j_1)}, Q_0^{(j_1)}, R_{n-k1}^{(j_1)}) = (1, 1, 0)$ and $M_0^{(j_1)} = 1$ while the j_2^{th} difference may result in $(P_0^{(j_2)}, Q_0^{(j_2)}, R_{n-k1}^{(j_2)}) = (1, 1, 0)$ and $M_0^{(j_2)} = 0$. Such contradictions happens with probability of $\frac{1}{2}$. In this problem all the wrong guesses are detected due to the high redundancy of equations. This claim will be discussed more precisely at the end of this session.

The recursive differential cryptanalysis ends here and yields X, Y and $n2$. In case of SASI, an additional stage should be done to extract the secrets $K1, K2, n1$ and ID from X, Y and $n2$. After obtaining almost all bits of X, Y and $n2$, we can find the values of $K1$ and $K2$ as follows.

TABLE 2: PMF of the rank of the matrix

Rank	Probability
32	2.8×10^{-7}
64	1.25×10^{-14}
80	9.6×10^{-4}
88	1.19×10^{-2}
92	4.26×10^{-2}
94	1.67×10^{-1}
96	7.78×10^{-1}

from (27) and (21):

$$\begin{aligned} X_i &= \mathbf{K1}_{i-k1} \oplus \mathbf{K2}_i \\ Y_i &= \mathbf{K2}_{i-k2} \oplus \mathbf{K1}_i \oplus \mathbf{n1}_{i-k2} \\ A_i &= IDS_i \oplus \mathbf{K1}_i \oplus \mathbf{n1}_i \end{aligned} \quad (42)$$

for $i = 0, \dots, n-1$. Hence, it holds that:

$$\mathbf{K1}_i \oplus \mathbf{K1}_{i-k2} \oplus \mathbf{K1}_{i-k1-k2} = V_i. \quad (43)$$

Where $V_i = A_{i-k2} \oplus IDS_{i-k2} \oplus X_{i-k2} \oplus Y_i$ is known for $i = 0, \dots, n-2, i \neq k2-1, k1+k2-1$. Therefore, set of equations given by (43) creates a system of $n-3$ linear equations in n variables.

Let's first assume that set of equations given by (43) holds for all $i = 0, \dots, n-1$. The rank of the coefficient matrix of such linear system equations, namely R , takes some limited number of values between $n/2$ and n depending on the values of nonuniformly distributed random variables $k1$ and $k2$. Table 2 shows the probability mass function of R for $n = 96$. In our problem, we have totally $n-3$ equations so (43) gives at least $R-3$ independent linear equations in $K1_i$'s. On the other hand, (22) and (42) holds:

$$IDS_i \vee (X_i \oplus \mathbf{K1}_{i-k1}) = (B - n2)_i \quad (44)$$

which also brings new linear equations in terms of $K1_i$ if $IDS_i = 0$ which has a probability of $\frac{1}{2}$. Therefore, in average we obtain from (44) $n/2$ more equations that along with $R-3$ independent equations in (43) holds an overdefined system of linear equations in K_i (except the case $R = 32$ which has the negligible probability of 2.8×10^{-7}). So the unique solution of this system, gives all bits of $K1$. The remaining secrets, $K2, n1$ and ID are also recovered respectively by (42) and (24).

probability of success. Here, we discuss that for a reliable full disclosure attack, how many differential pair of sessions $(S^{(0)}, S^{(j)})$, $j = 1, \dots, s$, is required.

First, two lemmas are stated:

Lemma 1. Assume s vectors $m^{(j)}, j = 1, \dots, s$ where $m^{(j)} = (m_1^{(j)}, m_2^{(j)})$. $m_i^{(j)}$'s ($i = 1, 2, j = 1, \dots, s$) are independent uniformly distributed binary random variables. Let 2×2 matrix $M^{(j)}$ be constructed as follows:

$$M^{(j)} = \begin{pmatrix} m_1^{(j)} & m_2^{(j)} \\ 1 & 1 \end{pmatrix}$$

for $j = 1, \dots, s$. The probability that all $M^{(j)}$'s are singular is:

$$P_{sing,2 \times 2} = \left(\frac{1}{2}\right)^s$$

Proof: Consider the following outcome:

$$\mathcal{A} : m^{(j)} \in \{(1, 1), (0, 0)\}, \forall j \in \{1, \dots, s\} \quad (45)$$

for $j = 1, \dots, s$. All $M^{(j)}$ are singular if the outcome \mathcal{A} occurs. Hence,

$$P_{sing,2 \times 2} = P(\mathcal{A}) = \left(\frac{1}{2}\right)^s. \quad \square$$

Lemma 2. Assume s vectors $m^{(j)}, j = 1, \dots, s$ where $m^{(j)} = (m_1^{(j)}, m_2^{(j)}, m_3^{(j)})$. $m_i^{(j)}$'s ($i = 1, 2, 3, j = 1, \dots, s$) are independent uniformly distributed binary random variables. Let 3×3 matrix $M^{(j_1, j_2)}$ be constructed as follows:

$$M^{(j_1, j_2)} = \begin{pmatrix} m^{(j_1)} & & \\ & m^{(j_2)} & \\ 1 & 1 & 1 \end{pmatrix}$$

where $j_1 \neq j_2$ and $j_1, j_2 \in \{1, \dots, s\}$. There are $\frac{s(s-1)}{2}$ such matrices. The probability that all $M^{(j_1, j_2)}$'s are singular is

$$P_{sing,3 \times 3} = 3 \cdot \left(\frac{1}{2}\right)^s - 2 \cdot \left(\frac{1}{4}\right)^s$$

Proof: Consider the following outcomes:

$$\begin{aligned} \mathcal{A} : m^{(j)} &\in \{(1, 0, 0), (0, 1, 1), (1, 1, 1), (0, 0, 0)\}, \\ &\forall j \in \{1, \dots, s\} \\ \mathcal{B} : m^{(j)} &\in \{(0, 1, 0), (1, 0, 1), (1, 1, 1), (0, 0, 0)\}, \\ &\forall j \in \{1, \dots, s\} \\ \mathcal{C} : m^{(j)} &\in \{(0, 0, 1), (1, 1, 0), (1, 1, 1), (0, 0, 0)\}, \\ &\forall j \in \{1, \dots, s\} \end{aligned}$$

All $M^{(j_1, j_2)}$'s are singular if the outcomes \mathcal{B} or \mathcal{C} or \mathcal{A} occur. Hence,

$$\begin{aligned} P_{sing,3 \times 3} &= P(\mathcal{A} \vee \mathcal{B} \vee \mathcal{C}) \\ &= P(\mathcal{A}) + P(\mathcal{B}) + P(\mathcal{C}) \\ &\quad - P(\mathcal{A} \wedge \mathcal{B}) - P(\mathcal{A} \wedge \mathcal{C}) - P(\mathcal{B} \wedge \mathcal{C}) + \\ &\quad P(\mathcal{A} \wedge \mathcal{B} \wedge \mathcal{C}) \end{aligned} \quad (46)$$

For a single j ,

$$P\left(m^{(j)} \in \{(1, 0, 0), (0, 1, 1), (1, 1, 1), (0, 0, 0)\}\right) = \frac{1}{2}$$

Hence,

$$P(\mathcal{A}) = P(\mathcal{B}) = P(\mathcal{C}) = \left(\frac{1}{2}\right)^s \quad (47)$$

Since $m^{(j)}$'s are independent.

$$\begin{aligned} P(\mathcal{A} \wedge \mathcal{B}) &= P(\mathcal{A} \wedge \mathcal{C}) = P(\mathcal{B} \wedge \mathcal{C}) \\ &= P\left(m^{(j)} \in \{(1, 1, 1), (0, 0, 0)\}, j = 1, \dots, s\right) \\ &= \left(\frac{1}{4}\right)^s \end{aligned} \quad (48)$$

$$\begin{aligned} P(\mathcal{A} \wedge \mathcal{B} \wedge \mathcal{C}) &= P\left(m^{(j)} \in \{(1, 1, 1), (0, 0, 0)\}, j = 1, \dots, s\right) \\ &= \left(\frac{1}{4}\right)^s \end{aligned} \quad (49)$$

Equations (46), (47), (48), and (49) holds:

$$P_{sing,3 \times 3} = 3 \cdot \left(\frac{1}{2}\right)^s - 2 \cdot \left(\frac{1}{4}\right)^s \quad (50)$$

□

Attack success depends on the existence of at least one nonsingular matrix M among all $\frac{s(s-1)}{2}$ matrices in (37) for all bits $i = 0, \dots, n-2, i \neq k1-1$ and at least one nonsingular matrix M among all s matrices in (41) for bit $i = k1-1$. Thus, by the mentioned lemmas:

$$\begin{aligned} P_{succ} &= (1 - P_{sing,3 \times 3})^{n-2} \times (1 - P_{sing,2 \times 2}) \\ &= \left(1 - \left(3 \cdot \left(\frac{1}{2}\right)^s - 2 \cdot \left(\frac{1}{4}\right)^s\right)\right)^{n-2} \\ &\quad \times \left(1 - \left(\frac{1}{2}\right)^s\right) \end{aligned} \quad (51)$$

According to (51) for $P_{succ} > 0.96$, it suffices that $s = 12$. Thus, for a reliable differential cryptanalysis of SASI, 13 consecutive genuine sessions are sufficient, provided that the last message of each one is blocked.

Fig. 2 shows the theoretical attack probability of success (51) in terms of number of differential pairs (s) as well as experimental results for $n = 96$. The experimental results is based on about 500 times running the simulated SASI protocol for $s = 2, \dots, 20$.

Probability of detecting wrong guesses of $k1$ and $k2$. For each bit i (except the MSB), there are at most eight distinct set of coefficients $(P_i^{(j)}, Q_i^{(j)}, R_{i-k1}^{(j)})$. Thus, there are at least $s - 8$ equations with the coefficients that are repetitive. since the probability of consistency of two equations with repetitive coefficients are $\frac{1}{2}$ (i.e. the probability of equality of their associated constants $M_i^{(j)}$), the probability of consistency of all the repetitive equations for a wrong guess of $k1$ and $k2$ is $P_{non-det} = \left(\frac{1}{2}\right)^{(s-8)(n-1)}$ which for $s = 12$ is completely negligible. Thus the probability of detection of the all $n^2 - 1$ wrong guesses is $P_{det} = (1 - P_{non-det})^{n^2-1} = \left(1 - \left(\frac{1}{2}\right)^{(s-8)(n-1)}\right)^{n^2-1}$ which is very close to one for $n = 96$ and $s = 12$.

Attack summary. This attack is *active* (though the attacker only blocks some messages), retrieves *all secret data* of the tag, and requires only *thirteen* consecutive genuine authentication sessions for probability of success more than 0.96.

5 CONCLUSIONS

We introduced new cryptanalysis techniques for full disclosure attack on ultralightweight authentication protocols. We called them recursive linear and recursive differential attacks. The former is based on constructing a systems of linear equations in i^{th} bit of secret variables and solve them in a recursive manner starting from the LSB. This attack, if can be applied successfully,

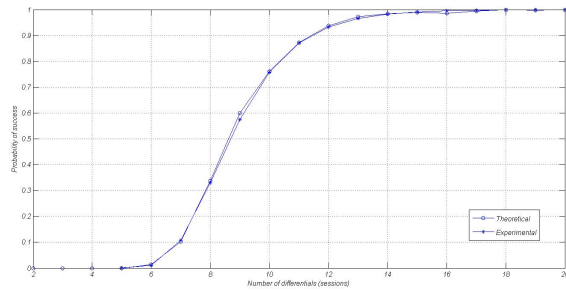


Fig. 2: Recursive differential cryptanalysis of SASI. Comparison of theoretical and experimental attack probability of success

is very efficient since it is passive, deterministic, and requires only one authentication session. We showed the effectiveness of this attack on SLMAP and Yeh et al. protocols.

In recursive differential attack, we again try to create a system of linear equations and solve them recursively. But these linear equations are derived from differential relations. This attack is more powerful than the first one and naturally has more requirements. The attacker is an active one who only blocks some specific messages of the protocols. This attack usually requires a couple of protocol sessions and reveals all secrets of the protocol with high probability of success. We applied this technique on *LMAP*⁺⁺ and SASI protocols successfully.

REFERENCES

- [1] G. Avoine, X. Carpent, and B. Martin, "Privacy-Friendly Synchronized Ultralightweight Authentication Protocols in the Storm," *Journal of Network and Computer Applications*, vol. 35, pp. 826-843, 2012.
- [2] H. Chien, "SASI: A New Ultralightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity," *IEEE Trans. Dependable and Secure Computing*, vol. 4, no. 4, pp. 337-340, Oct.-Dec. 2007.
- [3] T. Cao, E. Bertino, and H. Li, "Security Analysis of the SASI Protocol," *IEEE Trans. Dependable and Secure Computing*, vol. 6, no. 1, pp. 73-77, Jan.-Mar. 2009.
- [4] R.C.-W. Phan, "Cryptanalysis of a New Ultralightweight RFID Authentication Protocol SASI," *IEEE Trans. Dependable and Secure Computing*, vol. 6, no. 4, pp. 316-320, Oct.-Dec. 2009.
- [5] P. Peris-Lopez, J.C. Hernandez-Castro, J.M. Estevez-Tapiador, and A. Ribagorda, "LMAP: A Real Lightweight Mutual Authentication Protocol for Low-Cost RFID Tags," *Proc. Second Workshop RFID Security*, July 2006.
- [6] 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," *Proc. OTM 06 Workshop*, pp. 352-361, 2006.
- [7] P. Peris-Lopez, J.C. Hernandez-Castro, J.M. Estevez-Tapiador, and A. Ribagorda, "M2AP: A Minimalist Mutual-Authentication Protocol for Low-Cost RFID Tags," *Proc. Ubiquitous Intelligence and Computing*, pp. 912-923, 2006.
- [8] H. Sun, W. Ting, and K. Wang, "On the Security of Chiens Ultralightweight RFID Authentication Protocol," *IEEE Trans. Dependable and Secure Computing*, vol. 8, no. 2, pp. 315-317, Mar.-Apr. 2011.
- [9] M. Safkhani, Nasour Bagheri, Majid Naderi, and Somitra Kumar Sanadhya, "Security Analysis of *LMAP*⁺⁺, an RFID Authentication Protocol," *International Conference on Internet Technology and Secured Transactions (ICITST'11)*, Dec. 2011.
- [10] T. Li, "Employing lightweight primitives on low-cost rfid tags for authentication," *Proc. IEEE Vehicular Technology Conference, (VTC '08)*, pp. 1-5, Sept. 2008.
- [11] S. Wang and W. Zhang, "Passive Attack on RFID LMAP++ Authentication Protocol," *Cryptography and Network Security (CANS'11)*, LNCS 7092, pp. 185193, 2011.
- [12] T. Li, G. Wang. SLMAP - A Secure ultra-Lightweight RFID Mutual Authentication Protocol, In *Proc. Chinacrypt07*, pp.19-22, Oct. 2007.
- [13] J. C. Hernandez-Castro, J. E. Tapiador, P. Peris-Lopez, J. A. Clark, and E.-G. Talbi, "Metaheuristic traceability attack against SLMAP, an RFID lightweight authentication protocol," *International Journal of Foundations of Computer Science (IJFCS)*, vol. 23, no. 2, pp. 543-553, 2012.
- [14] K.H. Yeh, N.W. Lo, and E. Winata, "An Efficient Ultralightweight Authentication Protocol for RFID Systems," In *Proc. of RFIDSec Asia 2010, Cryptology and Information Security Series*, vol. 4, pp. 4960, 2010.
- [15] P. Peris-Lopez, J.C. Hernandez-Castro, R.C.W. Phan, M.E. Tapiador, T. Li, "Quasi-Linear Cryptanalysis of a Secure RFID Ultralightweight Authentication Protocol," In *6th China international conference on information security and cryptology (Inscrypt10)*, pp. 427-442, Oct. 2010.
- [16] P. D' Arco and A. De Santis, "On Ultralightweight RFID Authentication Protocols," *IEEE Trans. Dependable and Secure Computing*, vol. 8, no. 4, Jul.-Aug. 2011.
- [17] S. Vaudenay, "On privacy models for RFID," In *Proc. of ASIACRYPT 2007*. LNCS, vol. 4833, pp. 6887, 2007.
- [18] J.C. Hernandez-Castro, J.M.E. Tapiador, P. Peris-Lopez, and J.-J. Quisquater, "Cryptanalysis of the SASI Ultralightweight RFID Authentication Protocol," In *Proc. of Intl Workshop on Coding and Cryptography (WCC 09)*, May 2009.
- [19] D. Han, "Groebner Basis Attacks on Lightweight RFID Authentication Protocols," *Journal of Information Processing Systems*, Vol.7, No.4, pp. 691-706, 2011.
- [20] M. Matsui, "The First Experimental Cryptanalysis of the Data Encryption Standard," in the proceedings of CRYPTO 1994, *Lecture Notes in Computer Science*, vol. 839, pp. 1-11, 1994.
- [21] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," *Journal of Cryptology*, vol. 4, no.1, pp. 3-72, 1991.
- [22] A. Klimov, A. Shamir, "A new class of invertible mappings," In *proc. of Cryptographic hardware and embedded systems-CHES 2002*, *Lecture Notes in Computer Science*, vol. 2523, pp. 47083, 2003.