# An Efficient and Private RFID Authentication Protocol Supporting Ownership Transfer

Süleyman Kardaş [*] [†], Atakan Arslan [*‡], Serkan Çelik [*] [†], and Albert Levi[†]

[*]TÜBİTAK BİLGEM UEKAE, Kocaeli, Turkey
[†]Sabancı University, Faculty of Engineering and Natural Sciences, İstanbul, Turkey
[‡] İstanbul Technical University, Dept. of Electronics and Communication Engineering, Turkey

***Abstract*—Radio Frequency IDentification based systems, which are the most famous example of ubiquitous networks, are getting pervasively deployed in many daily life applications where privacy sensitivity is entrusted to tag or server. In some applications, ownership transfer of RFID labels is another significant requirement. Specifically, the owner of RFID tags could be required to change several times during its lifetime. During the transfer, new owner first obtains necessary private information from the old owner, with these information he then takes over tag identification and authorization so as to have secure communication with tags. Security and privacy are major issue in the presence of malicious adversary. Therefore, the protocol used to identify tag should not only allow a legitimate reader to authenticate a tag but it should also protect the privacy of the tag against unauthorized tracing. Besides, after ownership transfer, the authentication protocol should also prevent the old owner to trace the tags and disallow the new owner to trace old transactions of the tags. On the other hand, while achieving privacy and security on tag and server side, the computation complexity is also very important.**

**In order to resolve these security and privacy problems, numerous authentication protocols have been proposed in the literature. Many of them are failed to provide security and privacy and the computation on the server side is also very high. Motivated by this need, in this paper, we first analyze an existing RFID authentication protocol and show that it does not resist against tag tracking attack. Then, we propose an RFID mutual authentication protocol which is also used to realize ownership transfer. In our protocol, the server needs only a constant-time complexity for identification when the tag and server are synchronized. In case of ownership transfer, our protocol preserves both old owner and new owner privacy. Our protocol also achieves backward untraceability against a strong adversary who compromise tag, and forward untraceability under the assumption that the adversary misses at least one subsequent successful session between the tag and the reader.**

***Index Terms*—RFID, Privacy, Security, Ownership Transfer Protocol.**

## 1. Introduction

Today, ubiquitous information and communication technology has been widely accepted by everyone that aspire to reach information anytime and anywhere. Radio-frequency identification (RFID) systems are one example of the ubiquitous information technology especially due to the fact that they are at the center of the "Internet-of Things". RFID technology aims to identify and track an item or a person by using radio waves. It has been pervasively deployed in several daily life applications such as contactless credit cards, e-passports, ticketing systems, and etc.

A RFID system basically consists of several tags *(transponders)*, a set of readers *(interrogator)* and a back-end receiver. A tag contains a microchip which carries data and antenna. It is interrogated by a reader with its modulated radio signals. A RFID reader that is a central part of the a RFID system, acquire the data of the tag and convey it to the back-end system for further processing. Moreover, RFID tags can be categorized in three groups by using energy source such as active, passive and semi-passive or battery assisted tags. Passive RFID tags do not have own internal energy source. Instead, they use the radio energy transmitted by the reader [8]. Furthermore, RFID systems can also be grouped into the three basic ranges by their using operating frequency: Low frequency (LF, 30-300 KHz), high fequency (HF 3-30 MHz) and ultra high frequency ( 300 MHz - 3 GHz ) / microwave ( >3 GHz) [7].

Nowadays, the number of RFID applications have been proliferating because of their productivity, efficiency, reliability and so on. Many companies also prefer low-cost tags with tiny sizes. This brings some computational and memory restrictions to RFID tags. On the other hand, RFID tags and readers communicate with each other over air interface. This unsecure channel and the limited capabilities of RFID tags cause security and privacy vulnerabilities. An adversary may do tag impersonating, tracking, eavesdropping, and denial of service (DoS) attack. Besides the vulnerabilities, a tag might be distinguishable in its life-span by an attacker. If it is once recognized by an adversary, it will be easily able to be traceable. At that situation , there might be two attacks. (i) An attacker might track the previous interactions of the tag or (ii) he may track the future ones. These two attacks are called backward traceability and forward traceability, respectively. The protocol used for RFID system should provide not only resistance against passive attacks, replay attacks, cloning attacks but also resistance against active attacks. There are public-key cryptography solutions in literature but none of them are convenient for the low-cost tags used in lots of applications because of their limitations. It needs to find much light-weight approaches. Therefore, many light-weight authentication protocols are proposed to have a win against the adversaries that deceive the capacity-restricted tags. But, designing light-weight cryptographic authentication protocols with basic cryptographic primitives (xor, hash function) is a challenging task [17].

Another significant related problem is the changing ownership of RFID tags several times during its life-cycle. For instance, tags are initially created and attached to objects by producers and labeled objects are then taken over to retailers, and finally consumers buy tagged objects in shopping malls [11]. The ownership of a labeled object may be frequently transferred from one party to another. At the moment of the transfer, both new and old owners have the same information about the tag for its authentication, and this situation may cause an breach of the tag privacy. Therefore, this transfer has to guarantee that as soon as ownership transfer occurs, the old owner should no longer be able to trace the future interactions of the tag and the new owner should not be able to trace old interactions between the tag and its previous owner.

Besides the passion of having a strong, secure authentication protocols with providing privacy and robustness against the known attack scenarios, entire system performance has become an important issue. Therefore, designing authentication protocol without compromising security concerns begets decreasing efficiency of whole system. However, achieving the security and privacy properties, the complexity in tag and server side can vary dramatically from one protocol to another. Hence, while handling security and privacy issues, it is also important to realize it with less computational complexity in the server and tag side.

In order to resolve these security and privacy issues, numerous RFID authentication protocols have been recently proposed [1], [3]–[6], [9], [10], [13]–[16]. However, some of them are not compliant to ownership transfer. Also, none of them achieves constant-time complexity for identification while providing forward untraceability against old-owner and backward untraceability against the new owner.

Recently, Yanfei Liu proposed an authentication protocol in order to provide security and privacy for RFID tags. This protocol only needs constant-time complexity to identify an RFID tag irrespective of the total number of the tags in the system. However, this scheme is not resistant against tracking attack, tag impersonation attack, and desynchronization attack, if the attacker has the possibility to tamper with only one RFID tag. In [5], Erguler and Anarim (EA) enhanced this protocol in order to handle these attacks, but in the following section, we show that the revised protocol is also not resistant against tag tracking attack.

Our contributions are two-fold. First, we analyze the EA's protocol and point out the vulnerabilities against tag tracking. Although the authors claimed that they remedy the Yanfei Liu (YL)'s [12] security weaknesses, they can't overcome the tag tracking attack [5]. Second, we propose an efficient, secure and private RFID mutual authentication protocol which needs constant-time complexity to identify a tag. Then, we utilize this protocol and achieves a secure and efficient ownership transfer. We prove that our protocol achieves backward untraceability against the new owner and forward untraceability against the old owner. Moreover, we also show that our protocol provides backward untraceability against a strong attack and forward untraceability under an assumption that the adversary misses one subsequent successful protocol between the reader and the compromised tag.

The outline of the paper is as follows. In Section 2, the attack scenario is explained. The vulnerabilities of EA's protocol is shown. In Section 3, security and threat model, security and privacy concerns are discussed in RFID systems for ubiquitous networks. Section 4 describes the protocol. In Section 5, analysis of our protocol is given in a detail. In Section 6, we finally conclude the paper.

## 2. THE ANALYSIS OF EA'S PROTOCOL

In this section, we first present the protocol proposed by [5] and then, introduce the tracking attack which is induced by an incorrect update mechanism in the protocol.

### A. The Protocol Description

The notations used in the protocol is defined in Table I.

| | |
|---|---|
| $\mathcal{T}$ | RFID tag or transponder |
| $\mathcal{R}$ | RFID reader or transceiver |
| $\mathcal{DB}$ | The back-end database |
| $h$ | One-way hash function |
| $\|\|$ | Concatenation operator |
| $\oplus$ | Bitwise XOR operation |

TABLE I
THE NOTATIONS [5]

The registration of the protocol is as follows. For each RFID tag $\mathcal{T}_i$ , a random number $x_i$ and a secret key $y_i$ are generated by $\mathcal{DB}$. Then, $\{x_i, y_i\}$ pair is assigned to $\mathcal{T}_i$. A secret $K$ is shared by the registered tags and $\mathcal{DB}$. The authors state that in order to provide resistance against the tracking attack, unique secret K should be assigned to each tag. On the other hand, $\mathcal{DB}$ stores $K$ and $\{x_i, y_i, x_i^{old}, y_i^{old}\}$ tuple with related information for each tag $\mathcal{T}_i$ . Initially, $x_i^{old} = xi$ and $y{*}old_i = y_i$. The description of the Erguler and Anarim's protocol is as follows.

- $\mathcal{R}$ generates a random nonce $r_1$ and sends it to $\mathcal{T}_i$ as a query.
- $\mathcal{T}_i$ produces another random nonce $r_2$ and computes $M_1 = x_i \oplus h(K \oplus r_2)$, $M_2 = h(y_i||r_1||r_2)$.
- $\mathcal{T}_i$ sends $r2, M_1, M_2$ to $\mathcal{R}$.
- $\mathcal{R}$ transmits $r_1, r_2, M_1,\ and\ M_2$ to $\mathcal{DB}$.
- $\mathcal{DB}$ computes $x_i = M_1 \oplus h(K \oplus r_2)$ and searches $x_i$ with the $x$ and $x^{old}$ values stored in the database. If the values exist in the database, then $\mathcal{DB}$ uses corresponding $y_i$ in the table and checks whether $M_2 = h(y_i||r_1||r_2)$. If it holds, $\mathcal{DB}$ authenticates $\mathcal{T}_i$. Otherwise, it sends an error to $\mathcal{R}$ and terminates the session.
- $\mathcal{DB}$ computes new key $y_i^* = h(1||x_i \oplus y_i||r_1 \oplus r_2)$.
- $\mathcal{DB}$ calculates $M_3 = x_i^* \leftarrow h(x_i \oplus y_i||r_1||r_2)$, sends $M_3$ with related data of $\mathcal{T}_i$ to $\mathcal{R}$ and sets $x_i^{old} = x_i, x_i = x_i^*$ , $y_i^{old} = y_i, y_i = y_i^*$.
- R forwards $M_3$ to $\mathcal{T}_i$.
- $T_i$ checks $M_3 = h(x_i \oplus y_i||r_1||r_2)$. If it holds, $\mathcal{T}_i$ authenticates the server and sets $x{*}i = M_3$ and new key

$y_i^* = h(1||x_i \oplus y_i||r_1 \oplus r_2)$. Next, it sets: $x_i = x_i^*$ , $y_i = y_i^*$. Otherwise, it keeps $x_i$, $y_i$ unchanged.

### *B. The Tag Tracing Attack*

The authors claim that their protocol is resistant against tracking when each tag shares a unique secret $K$ value with the server. However, we show that an adversary $\mathcal{A}$ can link old subsequent transactions of a tag whenever $\mathcal{A}$ compromises a tag and capture its secret $K$. This violates tag untraceability. The attack works as follows. Let $\mathcal{C}$ be the challenger.

*1) Preparation Phase:*

- $\mathcal{C}$ chooses a pair of tag ($\mathcal{T}_0$, $\mathcal{T}_1$).
- $\mathcal{R}$ run two successful protocol transcripts $(\pi_0^0, \pi_0^1)$ with $\mathcal{T}_0$, then $\mathcal{C}$ obtains $\pi_0^0[r_1, r_2, M_1, M_2]$ and $\pi_0^1[r_1, r_2, M_1, M_2]$ messages.
- $\mathcal{R}$ run two successful protocol transcripts $(\pi_1^0, \pi_1^1)$ with $\mathcal{T}_1$, then $\mathcal{C}$ obtains $\pi_1^0[r_1, r_2, M_1, M_2]$ and $\pi_1^1[r_1, r_2, M_1, M_2]$ messages.

*2) Learning Phase:*

- $\mathcal{A}$ randomly chooses one of tag ( let $\mathcal{T}_0$).
- $\mathcal{A}$ compromises $\mathcal{T}_0$ and captures its secret K.

*3) Challenging Phase:*

- $\mathcal{A}$ is given a random pair of protocol transcript $(\pi_j^i, \pi_l^k)$ where $i, j, k, l \in_R \{0, 1\}$.
- $\mathcal{A}$ computes $M_3' = h(K \oplus \pi_j^i[r_2])$. If $M_3'$ is equal to $\pi_l^k[M_3]$, then this pair is two subsequent protocol transcript of $\mathcal{T}_0$. Otherwise, go to next step.
- $\mathcal{A}$ computes $M_3' = h(K \oplus \pi_l^k[r_2])$. If $M_3'$ is equal to $\pi_j^i[M_3]$, then this pair is two subsequent protocol transcript of $\mathcal{T}_0$.

Notice that, by applying the preceding attack, $\mathcal{A}$ has non-negligible advantage in link-ability of two subsequent protocol transcripts. In other words, we show that Erguler and Anarim's protocol does not achieve the tag untraceability property.

## 3. THE PROPOSED PROTOCOL

In this section, we propose a novel scalable RFID authentication protocol whic is the enhanced version of the scheme presented in [10]. The notations used in the protocol are defined. Then, the initialization and the authentication phases are described in detail. The protocol is summarized in Figure 1.

### *A. The Notations*

- $\in_R$: The random choice operator that randomly selects an element from a finite set.
- $\oplus$, $||$ : XOR operator and concatenation operator, respectively.
- $h$, $H$ : A hash function s.t. $h : \{0,1\}^* \to \{0,1\}^n$, $H : \{0,1\}^* \to \{0,1\}^{2n}$.
- $N$ : The number of tags in the database.
- $N_a$, $N_b$ : $n$-bit nonce generated by the reader and the tag, respectively.
- $K$ : $n$-bit secret shared between the tag and the reader.
- $val_1$, $val_2$ : $n$-bit the server validator of the tag and the reader, respectively.
- $K^{old_1}$, $K^{old_2}$ : Previous $n$-bit secret shared between the tag and the reader.
- $val_1^{old}$, $val_2^{old}$ : Previous $n$-bit the server validator of the tag and the reader, respectively.
- $L$, $S$ : The seed value of $val_1$ and $val_2$, respectively.
- $r_1, r_2$ : $n$-bit random bit strings produced by $h(N_a)$, $h(N_b, K)$, respectively.
- $v_i$ : $n$-bit random bit strings produced by $h(K, r_1, r_2)$.
- $M_1, M_2$ : $M_1 = V_1 \oplus L$, $M_2 = V_2 \oplus S$.
- $DB$ : Server database.
- $\gamma$ : $n$-bit string.
- $state$ : 1-bit string is 0 or 1.

### *B. The Registration Phase*

For each tag $T_i$, the following steps have to be performed by the registrar (e.g. the tag manufacturer) before the authentication protocol:

1) The registrar generates three $n$-bit random nonce ($K$, $S$, $L$). It also computes $val_1 = h(L, K)$, $val_2 = h(S)$. Initially, $K^{old_1}$ and $K^{old_2}$ are both equal to $K$, $S^{old}$ is equal to $S$, and $val_1^{old}$ is equal to $val_1$. Finally, state is set to 0 and it computes hash of the shared secret key $K$, $\gamma = h(K)$.
2) The registrar creates an entry in its back-end database and stores ($K, S, val_1, K^{old_1}, K^{old_2}, S^{old}, val_1^{old}, h(K)$) in the entry.
3) The registrar assigns $(K, L, val_2, state)$ to the tag $T_i$.

### *C. The Authentication Phase*

In our protocol (see Figure 1) each tag stores its own triple values $K$, $L$, $val_2$, $\gamma$,and $state$ . The reader stores the $K$, $S$, $val_1$ for that tag. The steps are described below.

Step 1.A reader randomly generates an $n$-bit nonce $N_a$ and computes hash of it $r_1 = h(N_a)$. Then it sends $r_1$ to the tag $T_i$.

Step 2.The tag $T_i$ randomly generates a $n$-bit $N_b$ nonce and computes hash of it, $r_2 = h(N_b, K)$. Then, it checks the state. If its own state is 0, it computes hash of the shared secret key $K$. If it isn't, the tag randomly generates a $n$-bit $\gamma$ nonce. Later, the tag uses a pseudo-random function that digests $r_1$, $r_2$ messages with shared secret key $K$ to compute $v_1||v_2 = H(K, r_1, r_2)$. The length of each $v_1$ and $v_2$ are both equal to $n$. After that, the tag computes message $M_1$ by simply XORing $v_1$ with secret $L$. Finally, the tag sends $r_2$, $M_1$ and $\gamma$ messages to the reader.

Step 3.The reader transfers $N_a$, $r_1$, $r_2$, $M_1$, and $\gamma$ to the server.

Step 4.The server firstly searches in DB that there exists $h(K)$ equals to $\gamma$.
The server performs an exhaustive search among all tags in the database. It computes $v_1||v_2 = H(K, r_1, r_2)$ and $h(M_1 \oplus v_1, K)$. The server checks whether $h(M_1 \oplus v_1, K^{old_1})$ is equals to $val_1$. If one match is found, then the server computes $M_2$

message by XORing $v_2$ with $S$ and then sends $M_2$ to the reader. After that, it updates $K^{old_2} = K^{old_1}$, $K^{old_1} = K$, $S^{old} = S$, $val_1^{old} = val_1$, $K = v_2$, $S = N_a$, and $val_1 = r_2$. If no match is found, then the server performs another an exhaustive search among all tags in the database. In this time, it computes $v_1||v_2 = H(K^{old_1}, r_1, r_2)$ and it checks whether $h(M_1 \oplus v_1, K^{old_2})$ is equals to $val_1^{old}$. If one match is found, the server computes $M_2$ message by XORing $v_2$ with $S$ and sends $M_2$ to the tag. After that, it updates $K = v_2$, $S = N_a$, and $val_1 = r_1$. However, if there is no match, the server generates an $n$-bit random bit string and sends it to the reader. The reason behind sending random bit string is that this prevents any attacker to validate $M_1$ for random nonce $r_1$ and $r_2$.

Step 5.The reader forwards $M_2$ to the tag $T_i$. Upon receiving $M_2$ message, $T_i$ computes $h(M_2 \oplus v_2)$ and checks whether it is equal to $val_2$. If equal, then it updates $K = v_2$, $L = N_b$, and $val_2 = r_1$.

### *D. The Ownership Transfer*

Once the owner of the tags are required to change one party to another, the tags are first synchronized with the server and the server run at least two successful authentication protocols with tags in a secure environment where no adversary is allowed to do any passive/active attacks. Then, all the tag related information and tags are transferred to new owner. Once the new owner receives the information and tags, the new owner runs at least one successful protocol between readers and the tags in a secure environment where a malicious adversary is not allowed to any kind of attacks.

During the ownership transfer, the old owner does not need to transfer the secret value of $K^{old_2}$ and $S^{old}$ of the tags to the new owner because the remaining secrets are enough to communicate with the synchronized tags.

## 4. Security, Privacy, and Performance Analysis

In this section, we first describe the adversarial capabilities. Then, we analyze our ownership transfer protocol depicted in Figure 1 against passive and strong attacks.

In our model, we assume that each tag can perform cryptographic hash operations. The communication between server and readers are assumed to be secure because they have no restriction on using SSL/TLS protocol. However, the reader and tags communicate over an insecure wireless channel and so an attacker can intercept, modify and generate messages. Also, each tag memory is not tamper-proof.

### *A. The Security Against Passive Adversary*

A offline passive adversary may want to know the contents of the secrets K and L stored in the tag $\mathcal{T}_i$. Then, the adversary simply eavesdrops the channels between a legitimate reader and $\mathcal{T}_i$ in order to get $r_1$, $r_2$, $M_1$, $M_2$ and $\gamma$. With these information and publish hash function $H$, she cannot obtain the secret $K$ or $L$ because of one-wayness of the hash function.

Moreover, the protocol also resists against replay attack because a challenge-response scheme is used in the protocol. In addition, for each session of the protocol a new pair of random numbers $(r_1, r_2)$ are used. This prevents to use the same challenge-response values in other sessions.

Furthermore, our protocol is resistant against desynchronization in case of the last flow of the protocol drops. Normally, this causes desynchronization of the tag secrets and the back-end server. However, this issue is resolved by storing previous of tag secrets in the database. Hence the server can resynchronize with the tags in such a condition.

### *B. The Security Against Strong Adversary*

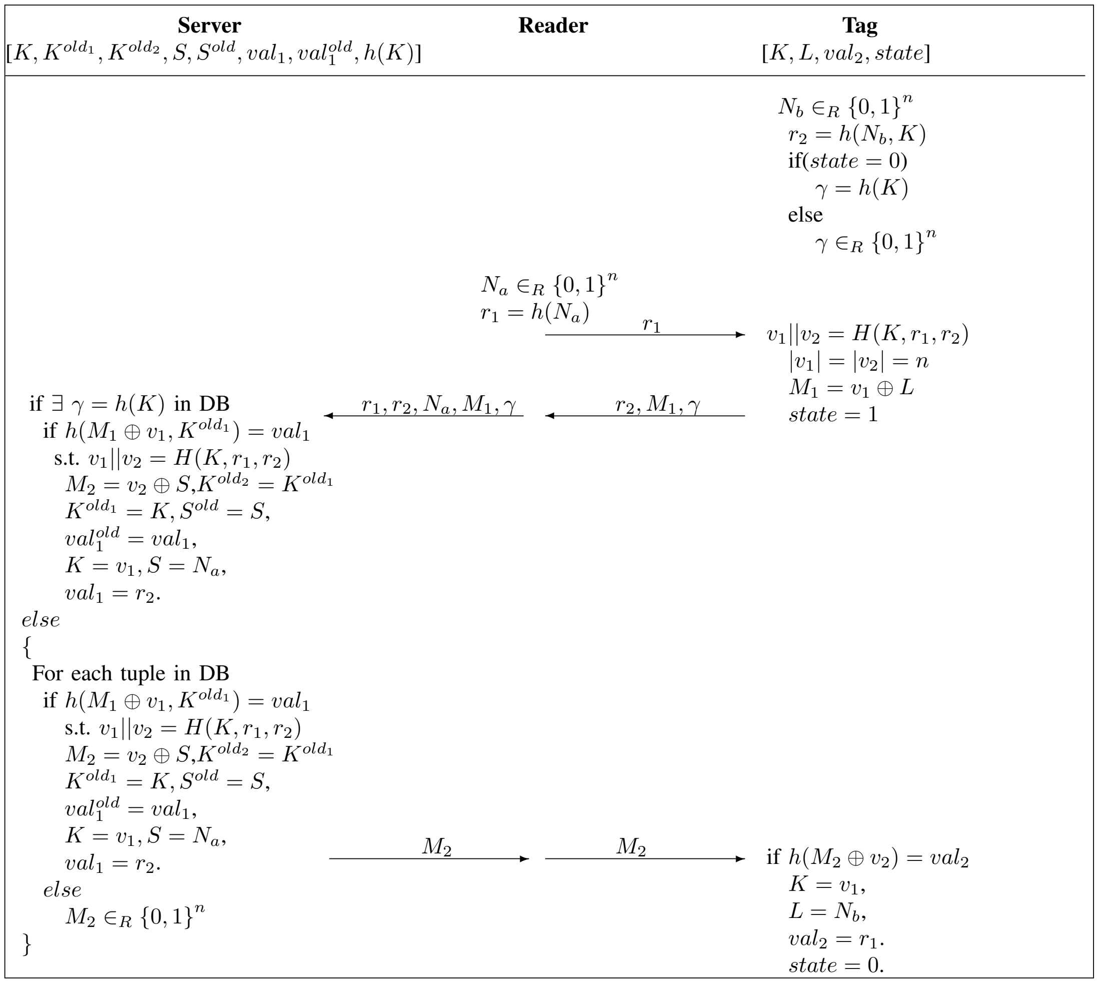
In this section, we will analyze the protocol depicted at Figure 1 in terms of backward and forward untraceability [2], [14], [18] against old owner, new owner, and a strong malicious adversary who can compromise a tag capture the secret in the tag. As a starting point, we assume that at time $t_i$, the owner of the system is changed. We test backward untraceability for the new owner, denoted by $\mathcal{A}_n$, with assumption that $\mathcal{A}_n$ has had control over communications between reader and tags made before time $t_i$. Note that, the number of these communications is finite. Similarly, we test forward untraceability against the old owner, denoted by $\mathcal{A}_o$. Also, we test these two privacy properties against a strong adversary $\mathcal{A}_s$ with assumption that $\mathcal{A}_s$ has ability of corrupting a tag and captures its secrets. Throughout the analysis, in order to make proofs more understandable, without loss of generality, we assume that there are only two tags in the system, namely $\mathcal{T}_0$ and $\mathcal{T}_1$. First of all, let us give the definitions of concepts mentioned above and the oracle that we use in the proofs of theorem given below.

**Definition 4.1.** *Backward Untraceability: An RFID scheme provides backward untraceability if $\mathcal{A}$ compromising $\mathcal{T}_i$ at time $t$ cannot trace the past interactions of $\mathcal{T}_i$ that occurred at time $t' < t$.*

**Definition 4.2.** *Forward Untraceability: An RFID scheme provides forward untraceability if $\mathcal{A}$ compromising $\mathcal{T}_i$ at time $t$ cannot trace the future interactions of $\mathcal{T}_i$ that occurred at time $t' > t$.*

**Definition 4.3.** *Oracle $\mathcal{O}^k$: The oracle chooses $b \in_R \{0,1\}$. If $b = 0$, $\mathcal{O}^k$ sends to the adversary the protocol transcript which was realized between tag $\mathcal{T}_0$ and the reader at time $t_k$. Similarly, if $b = 1$, the protocol transcript which was realized between tag $\mathcal{T}_1$ and the reader at time $t_k$ is sent to the adversary by the oracle. At the end, the adversary sends the bit $b'$ by after investigating the transcript sent. If $Pr[(b' = b) = 1] = \frac{1}{2} + \epsilon$, where $\epsilon$ is non-negligible, than the adversary wins.*

One can give simplified version of the oracle defined above as follows: At time $t_i$, $\mathcal{A}$ gets information of server and the tag $\mathcal{T}_0$. Then at time $t_k$, $\mathcal{O}^k$ chooses $b \in_R \{0,1\}$. The transcript sent to the adversary according to value of b same as above. Then, $\mathcal{A}$ returns $b' = 0$ if he thinks the transcript sent by oracle


**Server** $[K, K^{old_1}, K^{old_2}, S, S^{old}, val_1, val_1^{old}, h(K)]$

**Reader**

**Tag** $[K, L, val_2, state]$

Tag:
```
$N_b \in_R \{0,1\}^n$
  $r_2 = h(N_b, K)$
  if$(state = 0)$
    $\gamma = h(K)$
  else
    $\gamma \in_R \{0,1\}^n$
```

Reader:
```
$N_a \in_R \{0,1\}^n$
$r_1 = h(N_a)$
```

Reader $\xrightarrow{r_1}$ Tag

Tag:
```
$v_1||v_2 = H(K, r_1, r_2)$
  $|v_1| = |v_2| = n$
  $M_1 = v_1 \oplus L$
  $state = 1$
```

Server $\xleftarrow{r_1, r_2, N_a, M_1, \gamma}$ Reader $\xleftarrow{r_2, M_1, \gamma}$ Tag

Server:
```
if $\exists\ \gamma = h(K)$ in DB
  if $h(M_1 \oplus v_1, K^{old_1}) = val_1$
    s.t. $v_1||v_2 = H(K, r_1, r_2)$
      $M_2 = v_2 \oplus S$,$K^{old_2} = K^{old_1}$
      $K^{old_1} = K, S^{old} = S,$
      $val_1^{old} = val_1,$
      $K = v_1, S = N_a,$
      $val_1 = r_2.$
$else$
{
 For each tuple in DB
  if $h(M_1 \oplus v_1, K^{old_1}) = val_1$
    s.t. $v_1||v_2 = H(K, r_1, r_2)$
      $M_2 = v_2 \oplus S$,$K^{old_2} = K^{old_1}$
      $K^{old_1} = K, S^{old} = S,$
      $val_1^{old} = val_1,$
      $K = v_1, S = N_a,$
      $val_1 = r_2.$
  $else$
    $M_2 \in_R \{0,1\}^n$
}
```

Server $\xrightarrow{M_2}$ Reader $\xrightarrow{M_2}$ Tag

Tag:
```
if $h(M_2 \oplus v_2) = val_2$
  $K = v_1,$
  $L = N_b,$
  $val_2 = r_1.$
  $state = 0.$
```

Fig. 1. The Proposed RFID Authentication Protocol

realized between reader and tag $\mathcal{T}_0$. Otherwise the adversary returns $b' = 1$. If $Pr[(b' = b) = 1] = \frac{1}{2} + \epsilon$, where $\epsilon$ is non-negligible, than the adversary wins.

Throughout the proofs given to the corresponding theorem, four subsequent successful protocol transactions are enough. Thus, without loss of generality, we assume that $i = 4$ is the time where server owner changed, i.e. at time $t_4$. Moreover, addition to the notations given at protocol steps, we use left subscript part to denote the time that it was used.

In order to obtain traceability capability of $\mathcal{A}_n$, we start studying with more powerful adversary $\mathcal{A}_c$, who has had all secrets of the server and tags at time $t_i$ and observed all protocol transactions realized before given time.

**Theorem 4.1.** *The system has backward untraceability property for time $t_k$ satisfying $k < i - 3$ for the adversary $\mathcal{A}_c$*

*Proof:* Since at time $t_4$, $\mathcal{A}_c$ knows the value of ${}_4val_1$ and this value equals to ${}_3r_2$, then at time $t_3$, $\mathcal{A}_c$ can traces $\mathcal{T}_0$. Moreover, as $\mathcal{A}_c$ knows the value of ${}_4S^{old_1}$, then he knows the value of ${}_3S$. Thus, ${}_2N_a$ value is known. Therefore, at time $t_2$, $\mathcal{A}_c$ can traces $\mathcal{T}_0$ as he can figure out the value of ${}_2r_1$ from $h({}_2N_a)$. Note that, after that point, $\mathcal{A}_c$ knows ${}_2r_2$ and ${}_2M_2$ and since ${}_2K = {}_4K^{old_2}$, the values of ${}_2v_1$ and ${}_2v_2$ are known. Hence, ${}_2S$ is known. So, $\mathcal{A}_c$ learns the value of ${}_1N_a$. From this knowledge, $\mathcal{A}_c$ calculates ${}_1r_1$. Therefore, $\mathcal{A}_c$ can trace $\mathcal{T}_0$ at time $t_1$, which means $\mathcal{A}_c$ also learns the values of ${}_1r_2$, ${}_1M_1$, ${}_1M_2$. Apart from these values, ${}_1L$ is also known. Note that, the only thing $\mathcal{A}_c$ knows about the transaction happened at time $t_0$ is ${}_0N_b$. Thus, the probability of $\mathcal{A}_c$'s finding correct value of ${}_0r_2$ is $\frac{1}{2^n}$ since ${}_0K$ is not known and the range of hash function $h$ is $\{0,1\}^n$. Similarly, finding correct values of ${}_0r_1$,${}_0M_1$,${}_0M_2$ is $\frac{1}{2^n}$. Thus, the probability that $\mathcal{A}_c$ distinguishes the transcript that the oracle sent is $\frac{1}{2} + \frac{1}{2^n}$. However, $\frac{1}{2^n}$ is negligible.

Therefore, if $\mathcal{A}_c$ has all secrets of the server and tags at time $t_i$, then the system has backward untraceability property for time $t_k$ satisfying $k < i - 3$. ■

**Remark 1.** *The values of $K^{old_2}$ and $S^{old}$ of tags are stored in server database in order to overcome synchronisation problem. If at the time when ownership transfer is realised, the system is synchronised, then $K^{old_2}$ and $S^{old}$ values are not given to $\mathcal{A}_n$.*

At the next part, we give a backward traceability result for an adversary $\mathcal{A}_{cR}$, which is like $\mathcal{A}_c$ with exception indicated at Remark 1.

**Corollary 4.2.** *The system has backward untraceability property for time $t_k$ satisfying $k < i - 2$ for the adversary $\mathcal{A}_{cR}$.*

**Remark 2.** *The privacy is the main aim that should be reached. Therefore, just before ownership transfer, $\mathcal{A}_o$ completes two successful protocol transactions with tags such that no part of the protocol transcripts are seen by $\mathcal{A}_n$.*

Note that the adversary $\mathcal{A}_c$ with incapability explained at Remark 2 corresponds to the new owner, $\mathcal{A}_n$. Thus, we have the following corollary.

**Corollary 4.3.** *For the new owner, $\mathcal{A}_n$, the system has backward untraceability property for time $t_k$ satisfying $k < i$.*

**Theorem 4.4.** *If $\mathcal{A}_o$ has all secrets of the server and tags at time $t_i$, then the system has forward untraceability property for time $t_k$ satisfying $k > i$.*

*Proof:* Since ownership transfer occurs, $\mathcal{A}_o$ misses at least one of the subsequent successful protocol transactions between $\mathcal{A}_n$ and tags. We can get the best result if one subsequent successful transaction miss is assumed. In that case, $\mathcal{A}_o$ only knows values of ${}_5K^{old_1}$, ${}_4K^{old_2}$, ${}_5S^{old_1}$ and ${}_4val_1{}^{old}$. Since the attacker missed subsequent successful transaction, the other values are unknown. Note that, as the value of ${}_4N_b$ is not known, $\mathcal{A}_o$ can find the value of ${}_4r_2$ with possibility of $\frac{1}{2^n}$. By similar argument, $\mathcal{A}_o$ guess the value ${}_4r_2$ with possibility of $\frac{1}{2^n}$. Although $\mathcal{A}_o$ knows the values of ${}_4S$ and ${}_4L$, as ${}_4v_1$ and ${}_4v_2$ are not known, $\mathcal{A}_o$ can figure out the values of ${}_4M_1$ and ${}_4M_2$ with possibility of $\frac{1}{2^n}$. Hence, the probability that $\mathcal{A}_n$ distinguishes the transcript that the oracle sent is at most $\frac{1}{2} + \frac{1}{2^n}$. However, $\frac{1}{2^n}$ is negligible.

Therefore, if $\mathcal{A}_0$ has all secrets of the server and tags at time $t_i$, then the system has forward untraceability property for time $t_k$ satisfying $k > i$. ■

Our next result is about the adversary, $\mathcal{A}_s$, who can corrupts a tag and captures all secrets of the tag at any given time and follow all steps of the each successful protocol runs before and after the time corruption occurs.

**Corollary 4.5.** *If $\mathcal{A}_s$ corrupts a tag at time $t_j$ with $j \neq i$, then the system has backward untraceability for time $t_k$ satisfying $k < j - 1$ and forward untraceability for time $t_k$ satisfying $k > j+1$ under the assumption that $\mathcal{A}_s$ misses the transactions occurred at time $j + 1$ and $j - 1$.*

*Proof:* Forward secrecy part is direct result of Theorem 4.4. Moreover, the backward secrecy result is derived from Remark 3 ■

**Remark 3.** *If $\mathcal{A}_s$ does not miss the transaction at $j - 1$, then by knowledge of ${}_jval_2$, he deduces the value of ${}_{j-1}r_1$. Thus, the values of ${}_{j-1}r_2$, ${}_{j-1}M_1$, ${}_{j-1}M_2$ are known to him. Thus, in this case, $\mathcal{A}_s$ can trace the corrupted tag at time $t_{j-1}$. However, no more traces are possible, because $\mathcal{A}_s$ only knows the value of ${}_{j-2}N_b$ about the transaction realised at time $t_{j-2}$ and from the similar arguments given at proof of Theorem 4.1, the success probability that $\mathcal{A}_s$ traces the corrupted tag at time $t_{j-2}$ is $\frac{1}{2} + \frac{1}{2^n}$ and $\frac{1}{2^n}$ is negligible.*

**Remark 4.** *If $\mathcal{A}_s$ does not miss any the transaction after corruption occurs, then $\mathcal{A}_s$ can trace the corrupted tag forever.*

### C. Performance Issues

Considering memory storage for tag identifiers or keys and other information, our protocol requires $3n + 1$ bit ($3n$-bit for $K$, $L$, and $val_2$ and 1-bit for $state$) memory in tag side. Contrary to tags, server has no limited resource so we do not consider the server-side memory usage.

Concerning computational cost, our protocol requires at most 4 hash computation overhead for the tag. If the tags and the server are synchronized, the computational complexity at the server side is $\mathcal{O}(1)$. Otherwise, the complexity is at most $\mathcal{O}(N)$.

## 5. Conclusions

In this paper, we first analyze the security and privacy of Erguler and Anarim's RFID authentication protocol and we show that the protocol is not resistant against tag tracing attack. Then, we propose a secure and efficient an RFID mutual authentication protocol which is the revised version of the scheme peresented in [10]. With the use of the authentication protocol, we achieve ownership transfer. We prove that our protocol provides forward untraceability against the old owner of the tags and backward untraceability against the new owner of the tags. Also, we show that our protocol provides backward untraceability of a tag against an adversary who compromise the tag and forward untraceability under assumption that she misses at least one of the subsequent authentication protocol between the tag and the reader. Our protocol requires $\mathcal{O}(1)$ complexity to identify a synchronized tag.

## References

[1] B. Alomair, A. Clark, J. Cuellar, and R. Poovendran. Scalable rfid systems: a privacy-preserving protocol with constant-time identification. *Dependable Systems and Networks, International Conference on*, 0:1–10, 2010.

[2] G. Avoine. *Cryptography in Radio Frequency Identification and Fair Exchange Protocols.* PhD thesis, EPFL, Lausanne, Switzerland, December 2005.

[3] M. Burmester, B. de Medeiros, and R. Motta. Anonymous rfid authentication supporting constant-cost key-lookup against active adversaries. *IJACT*, 1(2):79–90, 2008.

[4] T. Dimitriou. A Lightweight RFID Protocol to protect against Traceability and Cloning attacks. In *SECURECOMM '05: Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, pages 59–66, Washington, DC, USA, 2005. IEEE Computer Society.

[5] I. Erguler and E. Anarim. Practical attacks and improvements to an efficient radio frequency identification authentication protocol. *Concurrency and Computation: Practice and Experience*, October 2011.

[6] A. Fernandez-Mir, R. Trujillo-Rasua, and J. Castella-Roca. Scalable RFID Authentication Protocol Supporting Ownership Transfer and Controlled Delegation. In *Workshop on RFID Security – RFIDSec'11*, Amherst, Massachusetts, USA, June 2011.

[7] K. Finkenzeller. *RFID Handbook.* John Wiley and Sons, 2003.

[8] S. Garfinkel and B. Rosenberg. *RFID: Applications, Security, and Privacy.* Addison-Wesley, 2005.

[9] J. Ha, S.-J. Moon, J. M. G. Nieto, and C. Boyd. Low-cost and strong-security rfid authentication protocol. In *EUC Workshops*, pages 795–807, 2007.

[10] S. Kardaş, A. Levi, and E. Murat. Providing Resistance against Server Information Leakage in RFID Systems. In *New Technologies, Mobility and Security – NTMS'11*, pages 1–7, Paris, France, February 2011. IEEE, IEEE Computer Society.

[11] C. H. Lim and T. Kwon. Strong and robust rfid authentication enabling perfect ownership transfer. In *ICICS*, pages 1–20, 2006.

[12] Y. Liu. An efficient rfid authentication protocol for low-cost tags. In *Proceedings of the 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing - Volume 02*, EUC '08, pages 180–185, Washington, DC, USA, 2008. IEEE Computer Society.

[13] D. Molnar and D. Wagner. Privacy and security in library RFID: issues, practices, and architectures. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, pages 210–219. ACM, 2004.

[14] M. Ohkubo, K. Suzuki, and S. Kinoshita. Cryptographic Approach to "Privacy-Friendly" Tags. In *RFID Privacy Workshop*, MIT, Massachusetts, USA, November 2003.

[15] B. Song and C. J. Mitchell. RFID authentication protocol for low-cost tags. In *WiSec '08: Proceedings of the first ACM conference on Wireless network security*, volume 0, pages 140–147. ACM, 2008.

[16] B. Song and C. J. Mitchell. Scalable RFID Security Protocols supporting Tag Ownership Transfer. *Computer Communication, Elsevier*, March 2010.

[17] I. Vajda and L. Buttyn. Lightweight authentication protocols for low-cost rfid tags. In *In Second Workshop on Security in Ubiquitous Computing Ubicomp 2003*, 2003.

[18] T. Van Le, M. Burmester, and B. de Medeiros. Universally Composable and Forward-secure RFID Authentication and Authenticated Key Exchange. In F. Bao and S. Miller, editors, *ACM Symposium on Information, Computer and Communications Security – ASIACCS 2007*, pages 242–252, Singapore, Republic of Singapore, March 2007. ACM, ACM Press.