# An Improved Ownership Transfer for RFID Protocol

Rui Xie[1], Bi-yuan Jian[2], and Dao-wei Liu[2]
*(Corresponding author: Bi-yuan Jian)*
School of Automation, Guangdong University of Technology[1]
Guangzhou 510006, China
School of Electronic and Computer Engineering, Guangzhou Vocational College of Science and Technology[2]
No. 1038, Guangcongroad Zhong Latan Baiyun District Guangzhou, Guangzhou 510006, China
(Email: jianbiyuan1983@126.com)
*(Received Mar. 6, 2017; Revised and Accepted May 16 & Jun. 21, 2017)*

## Abstract

The ownership transfer problems occur during using the RFID tag. In view of the problems of the RFID tag ownership transfer protocol, such as security defects and high computational cost, an improved lightweight RFID tag ownership transfer protocol is proposed in this paper. The improved protocol does not depend on the trusted third party, so that the improved protocol has a wider application space. Using the challenge-response mechanism, the new owner of the tag introduces the counter count and performs the corresponding operation according to the value of count to solve the desynchronization attack problems. The analysis results show that the improved protocol not only satisfies the security requirements of the tag transfer, but also overcomes the security defects of desynchronization attack. Compared with the existing RFID tag ownership transfer protocols, this improved protocol has larger promotion in the aspect of security and efficiency.

*Keywords: Internet of Things; Ownership Transfer Protocol; Rabin Algorithm; RFID, The Synchronization Attack*

## 1 Introduction

Radio frequency identification (RFID) is a kind of non-contact information transmission with a use of radio frequency signals, in order to achieve the purpose of identification by the transmitting information. RFID technology is widely used in production, logistics, national defense, transportation and other fields owning to the advantages of small size, easy portability, low cost, long life and so on [2, 7, 9, 16]. Because the RFID tag resources are limited and are running in the open wireless environment, the RFID system communication is vulnerable to various security threats such as eavesdropping attack and replay attack. So it is essential to ensure the security of the running protocols [3, 8, 12, 15, 17].

The ownership of the entity often changes during the practical application process. For example, after the producer sells the commodities to the wholesaler, the wholesaler has the ownership of the commodities physically, but it doesn't mean that the wholesaler completely controls the ownership of the commodities [4]. If there is no change in the ownership of the tag, the producers can still scan and obtain the tag's information; thereby the wholesaler's privacy may be exposed. When the wholesaler retails the commodities to the retailer, there will be a problem whether the ownership transfers completely or not, and likewise there will be a hidden danger of exposure of the tag's private information [1, 10, 18].

Reference [11] firstly proposes the RFID tag ownership transfer protocol, which uses trusted centers that are co-trusted by the old and new owners of tags to control all of the tags' information, but it limits the range of use of tags. In [14], the security and privacy requirements of the RFID tag ownership transfer protocol are defined, and three sub-protocols are proposed to achieve the transfer of RFID tag ownership with no trusted centers. However, several scholars have pointed out that the protocol has a lot of security problem. Reference [15] gives an improvement to the protocol in [14], and it proposes an extensible RFID authentication protocol that supports the tag ownership transfer, but the improved protocol still cannot protect the backward privacy and is vulnerable to de-synchronization attacks. In [5], two transfer schemes are proposed based on the quadratic residue, but both schemes require the exchange of information between the tags and the old and new owners over and over again, which results in the low computational efficiency of the tags. The ownership transfer protocol proposed in [6] cannot resist counterfeiting attacks, and the tags are easy to be tracked because the private keys Kp and Ku used in the tags are not updated every time. The ownership transfer

protocol proposed in [19] cannot resist DoS attacks, and replaying messages will make the tag updated repeatedly so that the protocol cannot resist replay attacks of the tags as well.

The ownership transfer protocol proposed in [13] can not resist the de-synchronization attacks. The attacker can obtain the message Q by listening to a complete communication process; then, by replaying the message Q during the process of blocking the associated communication, the tag's original owner continually updates the shared key so that the shared key between the tag's owner and the tag is different, which ultimately makes the both shared information out of sync. Aiming at the security flaws in the protocol of [13], an improved RFID tag ownership transfer protocol is proposed. In this protocol, the counter count is introduced on the tag's original owner, and the value of the counter is used to solve the problem of de-synchronization attack defects in the original protocol.

The remainder of this paper is organized as follows. The second part of the paper is to conduct a security analysis of the protocol in [13]. The third part is to put forward my own ownership transfer protocol. The fourth part is to carry out security analysis of the proposed protocol. The fifth part is using the BAN logic to formally verify the proposed protocol. The sixth part is the performance comparison between the proposed protocol and other protocols. The seventh part is the summary and concluding remarks.

## 2 Jin-wei Shen et al.'s Protocol and Its Drawbacks

The protocol can not resist de-synchronization attacks. In [13], an improved ultra-lightweight RFID ownership transfer protocol is proposed, which claims to be resistant to de-synchronization attacks. However, the study in this paper found that the ownership transfer protocol in [13] can not resist the de-synchronization attacks. The specific attack process is as follows.

The attacker can obtain all the information? such as IDS, M, N, P, Q, X, Y –in the complete communication process of [13] by using some monitoring methods. After obtaining the above information, the attacker can immediately block the communication process of the previous five steps, so that the shared key between $D_j$ and $T$ can be out of sync by continually replaying the message $Q$.

The first replay is as follows. The attacker disguises as $D_i$ to send the intercepted Q message to $D_j$. Since the authentication of $Q$ is passed before, the replay information $Q$ can also be authenticated. Before the message is replayed, the information stored in $D_i$ is $s_i$, $t_i$, $X$, $Y$, $IDSold = IDS$, and $IDSnew = IDS \oplus NT \oplus NR$. After the message is replayed, $D_i$ generates the random number $S_{i+1}$, calculates $t_{i+1}$, $X1$, $Y1$, and updates the data $IDSold = IDS$, $IDSnew = IDS \oplus NT \oplus NR$. Let $IDSnew = IDS1$, $u_i = s_i$, $v_i = t_i$, $s_i = s_{i+1}$, $t_i = t_{i+1}$. After $D_i$ is updated, $X1$ and $Y1$ will be sent to the tag, and the attacker will block the information transmission between the both.

The second replay is as follows. After the first replay, the attacker intercepts the $X1$, $Y1$ that are transmitted to the tag by $D_i$. At this time the attacker prevents the information from being transmitted to the tag, and meanwhile replays the message $Q$ again. Because $D_i$ stores the shared keys of this and the last authentication, so $Q$ can still be authenticated. After $Q$ is replayed again, $D_i$ will be set as follows.

$D_i$ generates random number $S_{i+2}$ and calculates $t_{i+2}$, $X2$, $Y2$ ; then updates data $IDSold = IDSnew = IDS1$, $IDSnew = IDS1 \oplus NT \oplus NR$. Let $IDSnew = IDS2$, $u_i = s_{i+1}$, $v_i = t_{i+1}$, $s_i = s_{i+2}$, $t_i = t_{i+2}$. After $D_i$ is updated, $X2$ and $Y2$ will be sent to the tag, and the attacker will block the information transmission between the both.

The third replay is as follows. After the second replay, the attacker intercepts the $X2$, $Y2$ that are transmitted to the tag by $D_i$. At this time the attacker prevents the information from being transmitted to the tag, and meanwhile replays the message $Q$ again. Because $D_i$ stores the shared keys of this and the last authentication, so $Q$ can still be authenticated. After $Q$ is replayed again, $D_i$ will be set as follows.

$D_i$ generates random number $S_{i+3}$ and calculates $t_{i+3}$, $X3$, $Y3$ ; then updates data $IDSold = IDSnew = IDS2$, $IDSnew = IDS2 \oplus NT \oplus NR$. Let $IDSnew = IDS3$, $u_i = s_{i+2}$, $v_i = t_{i+2}$, $s_i = s_{i+3}$, $t_i = t_{i+3}$. After $D_i$ is updated, $X3$ and $Y3$ will be sent to the tag, and the attacker will block the information transmission between the both.

After the above three replay attacks are completed, the attacker will transmit the original intercepted message $X$, $Y$ to the tag. Because the tag has never updated the shared key during the previous three replay attacks, $X$ and $Y$ certainly can be authenticated. After the authentication, the tag updates the shared key, $IDS = IDS \oplus NT \oplus NR$, i.e. $IDS = IDS1$; the shared key is $t_{i+1}$.

When analyzing the tag and the shared key ultimately stored in $D_i$, we can find that there is no synchronization between them. The information stored in the tag is $IDS1$, $t_{i+1}$, but the information stored in $D_i$ is $IDS3$, $t_{i+3}$. At this time, the attacker successfully makes the shared key between $D_i$ and the tag no longer the same by the replay attacks, so that the subsequent authentication fails. We can draw a conclusion that the original protocol can not resist the de-synchronization attacks.

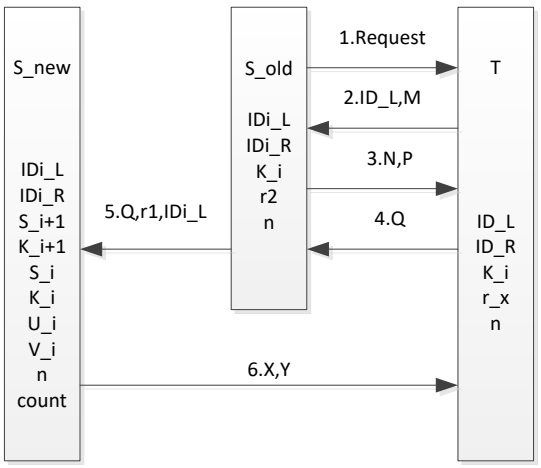


Figure 1: The improved protocol

## 3 Improved RFID Tag Ownership Transfer Protocol

The ownership transfer protocol proposed in [13] can not resist the replay attacks and de-synchronization attacks, so that this paper propose an improved RFID tag ownership transfer protocol. In this improved protocol, the message counter count is introduced to the tag's original owner S_old, and the replay attacks are resisted by the value of the counter count. The counter count is to record how many times the message $Q$ is replayed. The value of count does not exist or is 0, indicating that the message $Q$ is transmitted for the first time. If the value of count is not 0, the message $Q$ may be a replayed message. Since the above two situations are different, the operations of the tag's original owner S_old are different as well, which not only is able to resist the replay attacks, but also avoids the asynchronous problems between the tag's original owner S_old and the tag.

As is the same to other authentication protocols, we assume that the transmission channels between the tag's original owner S_old and the tag's new owner S_new are secure. We also suppose the transmission channels between the tag's original owner S_old and the tag are insecure, and that the transmission channels between the tag's new owner S_new and the tag are insecure as well.

### 3.1 Symbol Description

Firstly, the meaning of each symbol in this protocol is given in Table 1.

### 3.2 Protocol Description

The process of the improved RFID tag ownership transfer protocol is presented in Figure 1.

The descriptions of the symbols $M$, $N$, $P$, $Q$, $X$, $Y$ are

Table 1: The symbols used in the paper

| Symbols | Meaning |
|---|---|
| $S_{old}$ | The tag's original owner. |
| $S_{new}$ | The tag's new owner. |
| $T$ | A tag. |
| $T_i$ | The $i$-th tag. |
| $ID_i$_L | The left half of the $i$-th tag identifier ID. (its length is L bits) |
| $ID_i$_R | The right half of the $i$-th tag identifier ID. (its length is $L$ bits) |
| $ID\_L$ | The left half of the tag identifier ID. (its length is L bits) |
| $ID\_R$ | The right half of the tag identifier ID. (its length is L bits) |
| $r\_x$ | The tag data saved at the beginning. (its length is L bits) |
| $r1$ | The random number generated by the tag. (its length is L bits) |
| $r2$ | The random number generated by the tag's original owner. (its length is $L$ bits) |
| $n$ | Mersenne number, where the value is $n$=$2^L$-1. |
| $L$ | The length of the key. |
| $S_{-i}$ | The private key of the tag $T_i$. (its length is $L$ bits) |
| $K_{-i}$ | The public key of the tag $T_i$, where $K_{-i}=(S_{-i})^2 \bmod n$. (its length is $L$ bits) |
| $U_{-i}$ | The private key of the tag $T_i$ on the last round. (its length is $L$ bits) |
| $V\_i$ | The public key of the tag $T_i$ on the last round, where $U_{-i}=(V_{-i})^2 \bmod n$. (its length is $L$ bits) |
| $S_{-i+1}$ | The random number generated by the tag's new owner, used as the private key of the authentication on the current round. (its length is $L$ bits) |
| $K_{-i+1}$ | The public key of the authentication on the current round, where $K_{-i+1} = (S_{-i})^2 \bmod n$. (its length is $L$ bits) |
| $count$ | The counter for the message $Q$ of the tag's new owner. |
| $M$, $N$, $P$, $Q$, $X$, $Y$ | The communication data in this protocol. (each length is $L$ bits) |
| $MIXBITS(a, b)$ | The new random number obtained by computing (a, b). (the output length is $L$ bits) |
| $\oplus$ | XOR operation. |
| & | AND operation. |
| $[X]_L$ | Take the first L bits of the result of the operation []. |

as follows.

$$\begin{aligned} M &= K\_i \oplus r1. \\ N &= r2 \oplus ID_i\_R. \\ P &= [(r1 \oplus r2 \oplus K\_i)^2 \bmod n]_L. \\ Q &= [(r1 \oplus r2 \oplus D\_R)^2 \bmod n]_L. \\ X &= S\_{i+1} \oplus r1 \oplus ID_i\_R. \\ Y &= K\_{i+1} \& r1 \& ID_i\_R. \end{aligned}$$

$[\cdot]_L$ means taking the first L bits of the result of the operation [].

The complete execution steps of this ownership transfer protocol are described below.

**Step 1:** The tag's original owner S_old sends a request command Request for transferring tag ownership to the tag T, and opens ownership transfer session.

**Step 2:** T receives the messages from S_old; then calculates r1=r_x, M=$K\_i \oplus$ r1, and sends the values of ID_L and M to S_old.

**Step 3:** *S_old* receives the messages from *T*; then searches the database for the result of whether $ID_i\_L$ is equal to *ID_L*. If the result does not exist, the tag is forged, and the protocol terminates immediately. If it exists, *S_old* generates an *L*-bit random number *r2*, and uses K$_i$ (corresponding to $ID_i\_L$) to calculate $K\_i \oplus M$ and obtain a random number *r1*.Then it uses *r1*, *r2*,$ID_i\_R$ (corresponding to $ID_i\_L$) and K$_i$ to calculate $N = r2 \oplus ID_i\_R$ and $P=[(r1 \oplus 2 \oplus K\_i)^2 \bmod n]_L$ . Finally the values of *N* and *P* are sent to *T*.

**Step 4:** *T* receives messages *N* and *P* from *S_old*. The tag uses its own *ID_R* to calculate $N \oplus$ *ID_R* and obtain random number *r2*. Then the tag uses random number *r1* generated by itself, random number *r2* and its own public key $K\_i$ to verify the correctness of *P*, i.e.

$$P' = [(r1 \oplus (N \oplus ID\_R) \oplus K\_i)^2 \bmod n]_L.$$

If $P'$ is unequal to *P*, then *S_old* is forged, and the protocol terminates immediately. If $P'$ is equal to *P*, the tag correctly verifies *S_old*. Next the tag begins to update data *r_x=MIXBITS(r1, r2)*, and uses random number *r1* generated by itself, random number *r2* and its own *ID_R* to calculate the value of *Q*. Finally the value of *Q* is sent to *S_old*.

**Step 5:** *S_old* receives the message *Q* from the tag. *S_old* uses random number *r2* generated by itself, random number *r1* and its own $ID_i\_R$ to verify the correctness of *Q*, i.e.

$$Q = [((K\_i \oplus M) \oplus r2 \oplus ID_i\_R)2 \bmod n]_L.$$

If $Q'$ is unequal to *Q*, then the tag is forged, and the protocol terminates immediately. If $Q'$ is equal to *Q*, the tag correctly verifies *S_old*. Then *S_old* send all the values of *Q*, *r1*, *ID_L* to the new tag's owner *S_new* through secure channels.

**Step 6:** *S_new* receives the messages from *S_old*. Then *S_new* searches the database for the result of whether $Q'$ is equal to *Q*. If the result exists and the value of the corresponding counter count is not 0, it indicates that the message *Q* has been transmitted. In order to resist replay attacks, *S_new* executes Step 7. If the result does not exist, *S_new* executes Step 8.

**Step 7:** *S_new* searches the database for the result of whether $ID_i\_L$ is equal to *ID_L*. If the result does not exist, the tag is forged, and the protocol terminates immediately. If the result exists, *S_new* does not make any updates, and the values of *X* and *Y* which is calculated during the last authentication are transmitted directly to the tag.

**Step 8:** *S_new* stores the value of *Q* into its own database, allocates a corresponding counter, and sets the counter count to 1. Then *S_new* searches the database for the result of whether $ID_i\_L$ is equal to *ID_L*. If the result does not exist, then the tag is forged, and the protocol terminates immediately. If the result exists, then *S_new* generates a *L*-bit random number $S\_{i+1}$, uses it as the new private key in the current authentication, and calculates $K\_{i+1} = (S\_{i+1})^2 \bmod n$. After the calculation is finished, *S_new* begins to update data $U\_i = S\_i$ , $V\_i = K\_i$ , $S\_i = S\_{i+1}$ , $K\_i = K\_{i+1}$ , and uses random number $S\_{i+1}$ generated by itself, *r1* transmitted from *S_old*, $K\_{i+1}$ and $ID\_i\_R$ (corresponding to $ID\_i\_L$) to calculate $X=S\_{i+1} \oplus r1 \oplus ID_i\_R$ , $Y = K\_{i+1}$ & *r1* & $ID_i\_R$. Finally the values of *X* and *Y* are sent to the tag.

**Step 9:** *T* receives messages *X* and *Y* from *S_new*. Next the tag uses the random number *r1* generated by itself, its own *ID_R* and *X* sent from *S_new* to calculate $X \oplus r1 \oplus ID\_R$ and obtain the private key $S\_{i+1}$. Then it uses the private key S$_{i+1}$, random number *r1* generated by itself and its own *ID_R* to verify the correctness of *Y*, i.e.

$$Y' = ([(S\_{i+1})^2 \bmod n]_L) \& r1 \& ID\_R.$$

If $Y'$ is unequal to *Y*, then *S_new* is forged, and the protocol terminates immediately. If $Y'$ is equal to *Y*, the tag correctly verifies *S_new*, and then the tag begins to update data $K\_i = Y \oplus r2$. The tag ownership transfers successfully.

# 4 Security Analysis

## 4.1 Valid Target Transfer

Valid target (abbr. VT) transfer means it is the valid target that is transferred, instead of other tags in the

system. In the improved protocol, the tag's original owner *S_old* verifies the authenticity of the tag for the first time in Step 3. The tag firstly verifies the authenticity of the tag's original owner *S_old* in Step 4, and then *S_old* will verify the tag's authenticity again in Step 5. It makes the authentication security improve greatly between the tag's owner and the tag, and after mutual authentication it can make sure that the current authenticated tag is certainly the tag that belongs to *S_old*.

*S_new* verifies the tag's authenticity in Step 6. The tag verifies the authenticity of *S_new* in step 9. During the entire authentication process, the security of the protocol improves greatly because there is a mutual authenticity verification between *S_new* and the tag. Through the above process to achieve mutual authentication, it can ensure that the tag is certainly the target that will be transferred to *S_new*. The target tag has been authenticated several times to complete the ownership transfer from *S_old* to *S_new*. As a result, the improved protocol ensures that the transfer tag is certainly the target, not the other tags in the system.

### 4.2 Impersonation Attack

We assume that the attacker impersonates the tag's original owner *S_old*. Because the attacker does not know the shared key $K_{-i}$ and $ID_i_R$between *S_old* and the tag $T_i$, the attacker can not correctly calculate the values of $N$ and $P$. In Step 4 the tag will promptly find that *S_old* is forged, and the protocol terminates immediately.

Then we assume that the attacker impersonates the tag's new owner *S_new*. Because the attacker does not know the shared key $K_{-i}$ and $ID_i\_R$ between *S_new* and the tag $K_{-i}$, the attacker can not correctly calculate the values of $X$ and $Y$. In Step 4 the tag will promptly find that *S_new* is forged, and the protocol terminates immediately.

Next we assume that the attacker impersonates the tag. The attacker knows neither the shared key $K_{-i}$ and $ID_i\_R$ between *S_old* and the tag $T_i$, nor the shared key $K_{-i}$ and $ID_i\_R$ between *S_new* and the tag $T_i$, so there is no way at all to correctly calculate the value of $M$ and $Q$. Both *S_old* in Step 3 and *S_new* in Steps 6, 7, 8, will find that the tag is forged, and the protocol terminates immediately. Above all, the improved protocol can resist resist various impersonation attacks.

### 4.3 Brute Force Attack

By listening to a complete communication process, the attacker can obtain the values of $M$, $N$, $P$, $Q$, $X$ and $Y$. In the improved protocol, the random numbers *r1*, *r2* are no longer transmitted in plain text, but simply encrypted with other information firstly. For instance, the attacker is unaware of the values of the shared private key $K_{-i}$ and $ID_i\_R$ between *S_old* and the tag $T_i$. Moreover, the number *r2* is randomly generated by *S_old*, and the number *r1* is randomly generated by the tag as well. From the attacker's perspective, although the values of $N$ and $P$ are intercepted, it is impossible to make an exhaustion of any useful privacy information.Aiming at the formulas $P = [(r1 \oplus r2 \oplus K_{-i})^2 \bmod n]_L$ and $N = r2 \oplus ID_i\_R$, the attacker knows nothing about *r1*, *r2*, $K_{-i}$, $ID_i\_R$. As a result, it is impossible for an attacker to analyze a specific $K_{-i}$. Simply knowing the values of $N$ and $P$, there is no way to make an exhaustion of the specific values. For the same reason, the attacker is unable to make an exhaustion of any useful information by intercepting the values of $M$, $N$, $P$, $Q$, $X$, $Y$. Based on the above descriptions, the improved protocol can resist brute force attacks.

### 4.4 Replay Attack

During each execution of the improved protocol, *S_old* uses a random number *r2* generated by itself to keep the messages fresh, and similarly *S_new* uses a random number $s_i$ generated by itself to keep the messages fresh. The tag uses a random number *r1* generated by function *MIXBIT(a, b)* to keep the messages fresh. Based on the above descriptions, the values of $M$, $N$, $P$, $Q$, $X$, $Y$ are various in each step. Therefore, although the attacker replays the messages, any useful information won't be available. So the improved protocol can resist replay attacks.

### 4.5 De-synchronization Attack

In the improved protocol, *S_new* introduces the counter count. It's no use that the attacker replays message $Q$, because only when the value of count does not exist or is 0 will *S_new* generate a new random number $s_i$ as the new shared private key, and then execute the subsequent update steps. If the value of count is not 0, it indicates that the message $Q$ has existed before, and this $Q$ is possibly the information which the attacker replays. In order to resist de-synchronization attack, *S_new* will not generate new random numbers, but directly use current private key to verify the correctness of $Q$. Then it uses current private key to update related data. During this process, according to the different values of count, the update operation also uses different mechanisms, so it avoids the differences of the shared private key between the tag and *S_new* caused by replaying the message $Q$. Based on the above descriptions, the improved protocol can resist de-synchronization attacks.

Table 2 shows the security comparison between this protocol and several other RFID tag ownership transfer protocols. $\surd$ indicates resistance, $\times$ indicates irresistance.

## 5 BAN Logic Formal Analysis

In this paper, BAN logic formalization method is used to prove the security of the improved protocol. BAN logic is proposed by Burrows et al. Using BAN logic, the proving process of the protocol is shown as follows. Because both

Table 2: Protocol security comparison

| Attack Types | Ref.[11] | Ref.[13] | Ref.[14] | Ref.[15] | Our protocol |
|---|---|---|---|---|---|
| *VT* | √ | √ | √ | √ | √ |
| *Impersonate Attack* | √ | × | √ | √ | √ |
| *Brute Force Attack* | √ | √ | √ | √ | √ |
| *Replay Attack* | √ | √ | × | √ | √ |
| *Replay Attack* | × | √ | √ | × | √ |

*S_new* and *S_old* have part of readers, so we can see the both as a whole, as a large reader, represented with R.

## 5.1 Idealized Model of the Protocol

**Message 1:** $R \longrightarrow T$: *Q*uery;

**Message 2:** $T \longrightarrow R$: ID_L, $M$;

**Message 3:** $R \longrightarrow T$: $N$, $P5$;

**Message 4:** $T \longrightarrow R$: $Q$;

**Message 5:** $R \longrightarrow T$: X, Y.

## 5.2 Expected Target of the Protocol

The main proving target of the protocol's correctness is G1, G2, G3, G4, G5 and G6, that is, mutual authentication entity's belief in the freshness of interactive information.

**G1:** $R \mid\equiv Q$, $R$ believes $Q$.

**G2:** $T \mid\equiv N$, $T$ believes $N$.

**G3:** $T \mid\equiv P$, $T$ believes $P$.

**G4:** $T \mid\equiv X$, $T$ believes $X$.

**G5:** $T \mid\equiv Y$, $T$ believes $Y$.

**G6:** $R \mid\equiv M$, $R$ believes $M$.

## 5.3 Initial Assumptions of the Protocol

**P1:** $R \mid\equiv R \overset{K_{-i}}{\longleftrightarrow} T$, $R$ believes $R$ and $T$ share the public key $K_{-i}$.

**P2:** $T \mid\equiv R \overset{K_{-i}}{\longleftrightarrow} T$, $T$ believes $R$ and $T$ share the public key $K_{-i}$.

**P3:** $R \mid\equiv R \overset{n}{\Longleftrightarrow} T$, $R$ believes $R$ and $T$ share the Mersenne number $n$.

**P4:** $T \mid\equiv R \overset{n}{\Longleftrightarrow} T$, $T$ believes $R$ and $T$ share the Mersenne number $n$.

**P5:** $R \mid\equiv R \overset{ID\_R}{\Longleftrightarrow} T$, $R$ believes $R$ and $T$ share the identifier *ID_R*.

**P6:** $T \mid\equiv R \overset{ID\_R}{\Longleftrightarrow} T$, $T$ believes $R$ and $T$ share the identifier *ID_R*.

**P7:** $R \mid\equiv \sharp\,(r1)$, $R$ believes the freshness of the random number *r1*.

**P8:** $T \mid\equiv \sharp\,(r1)$, $T$ believes the freshness of the random number *r1*.

**P9:** $R \mid\equiv \sharp\,(r2)$, $R$ believes the freshness of the random number *r2*.

**P10:** $T \mid\equiv \sharp\,(r2)$, $T$ believes the freshness of the random number *r2*.

**P11:** $R \mid\equiv \sharp\,(S\_i{+}1)$, $R$ believes the freshness of the random number *S_i+1*.

**P12:** $T \mid\equiv \sharp\,(S\_i{+}1)$, $T$ believes the freshness of the random number *S_i+1*.

**P13:** $R \mid\equiv \sharp\,(K\_i{+}1)$, $R$ believes the freshness of the random number *K_i+1*.

**P14:** $T \mid\equiv \sharp\,(K\_i{+}1)$, $T$ believes the freshness of the random number *K_i+1*.

**P15:** $T \mid\equiv R \mid\Rightarrow N$, $T$ believes $R$ has jurisdiction over $N$.

**P16:** $T \mid\equiv R \mid\Rightarrow P$, $T$ believes $R$ has jurisdiction over $P$.

**P17:** $T \mid\equiv R \mid\Rightarrow X$, $T$ believes $R$ has jurisdiction over $X$.

**P18:** $T \mid\equiv R \mid\Rightarrow Y$, $T$ believes $R$ has jurisdiction over $Y$.

**P19:** $R \mid\equiv T \mid\Rightarrow M$, $R$ believes $T$ has jurisdiction over $M$.

**P20:** $R \mid\equiv T \mid\Rightarrow Q$, $R$ believes $T$ has jurisdiction over $Q$.

## 5.4 The Proving Process of the Protocol

From Message 4 we have that $R \lhd \{D\}$, which means $R$ had receive the message D. According to the initial assumptions P2, P5 and the message-meaning rule $\dfrac{R \mid\equiv R \overset{K}{\longleftrightarrow} T, P \lhd \{X\}_K}{P \mid\equiv Q \mid\sim X}$, it follows $R \mid\equiv T \mid\sim D$.

Table 3: Performance comparison of protocols

| Operation | Ref.[11] | Ref.[13] | Ref.[14] | Ref.[15] | Our protocol |
|---|---|---|---|---|---|
| *Operation A* | 0T1 | 1T1 | 0T1 | 3T1 | 2T1 |
| *Operation B* | 2T2 | 1T2 | 19T2 | 8T2 | 9T2 |
| *Operation C* | 1T3 | 0T3 | 0T3 | 5T3 | 3T3 |
| *Operation D* | 0T4 | 0T4 | 0T4 | 1T4 | 1T4 |
| *Operation E* | 0T5 | 4T5 | 5T5 | 0T5 | 0T5 |
| *Operation F* | 0T6 | 0T6 | 3T6 | 3T6 | 3T6 |
| *Operation G* | 0T7 | 0T7 | 1T7 | 0T7 | 0T7 |
| *Operation H* | 0T8 | 3T8 | 5T8 | 0T8 | 0T8 |
| *Operation I* | 2T9 | 1T9 | 2T9 | 2T9 | 2T9 |
| *Storage capacity* | 1L | 3L | 4L | 3L | 3L |

Then by the initial assumptions P7, P9 and the freshness-concatenation rule $\frac{P \mid\equiv \sharp(X)}{P \mid\equiv \sharp(X,Y)}$, it follows $R \mid\equiv \sharp\ (D)$.

Because of the conclusions $R \mid\equiv T \mid\sim D$, $R \mid\equiv \sharp\ (D)$, that we have proved above and the nonce-verification rule $\frac{P \mid\equiv \sharp(X), P \mid\equiv Q \mid\sim X}{P \mid\equiv Q \mid\equiv X}$, we have that $R \mid\equiv T \mid\equiv D$.

Finally, according to the corollary $R \mid\equiv T \mid\equiv D$, the initial assumption P20 and the jurisdiction rule $\frac{R \mid\equiv T \mid\Rightarrow Q, P \mid\equiv Q \mid\equiv X}{P \mid\equiv X}$, it can be proved that $R \mid\equiv D$. Thus, the proof of target G1 is now completed.

The target G2, G3, G4, G5 and G6 can be proved in a similar way as shown above.

## 6 Performance Analysis

The tag calculation complexity, the tag storage space and other several aspects are used for performance analysis.

As shown in Table 3, in [15] the tag stores $t_i$, with storage capacity of 1L. In [5] the tag stores *ik*, *uk* and *id*, with storage capacity of 3L. In [6] the tag stores *h(TID)*, *KTID*, *r* and *n*, with storage capacity of 4L. In this paper the tag stores *IDS*, $t_i$, and *Nx*, with storage capacity of 3L.

Operation A represents '+' operation, the operation time of which is represented by T1. Operation B represents '$\oplus$' operation, the operation time of which is represented by T2. Operation C represents Rabin encryption, the operation time of which is represented by T3. Operation D represents the function *MIXBITS(x, y)*, the operation time of which is represented by T4. Operation E represents Hash function, the operation time of which is represented by T5. Operation F represents *mod* operation, the operation time of which is represented by T6. Operation G represents CRC function, the operation time of which is represented by T7. Operation H represents PENG operation, the operation time of which is represented by T8. Operation I represents comparison operation, the operation time of which is represented by T9. Moreover, the time represented by T1 to T9 is different, some operations to spend a long time, some operations to take a short time. To sum up, the overhead cost of the protocol presented in this paper is acceptable.

Compared the improved protocol in this paper and the protocol in [13], both of the tag storage space are similar. In terms of the tag calculation, the improved protocol has twice less square operations than the original protocol. Although calculation complexity is not much reduced, it is found that the improved protocol solves the security flaws in the original protocol without increasing the calculation complexity of the tag. The original protocol can not resist the replay attacks and can not resist the de-synchronization attacks, however, the improved protocol can resist them. Compared with the protocols in [5, 6, 15], the tag storage space of the improved protocol is similar to theirs. What's more, it reduces the total calculation complexity of the tag, and meanwhile compensates for the security flaws in the protocols above.

## 7 Conclusion

An improved lightweight RFID tag ownership transfer protocol is proposed for the security problems of current ownership transfer protocol in [13]. Aiming at the problem that the tag's new owner in the original protocol can not resist the de-synchronization attacks caused by the replay messages, the improved protocol introduces the concept of the counter count for message $Q$. According to the value of *count*, different operations are used so as to solve the de-synchronization problems. If the value of *count* does not exist or is 0, the tag's new owner will generate new random numbers, otherwise won't, which makes it possible to avoid the problem that the shared private key between the both is not synchronized because the random number is generated after the message $Q$ is received multiple times. Finally, a comprehensive security analysis shows that the improved protocol meets the security requirements of the tag ownership transfer.

## Acknowledgments

This work was supported in part by the Natural Science Foundation of China under Grant 61472090, Grant 61472089 and Grant 61672169, in part by the Science and Technology Project of Guangdong Province under Grant 2015B010128014 and Grant2016B010107002,in part by the Science and Technology Planning Project of Guangzhou under Grant 201707010492.Grant 201604016003,Grant 201604016067 and Grant 201604016041.

## References

[1] M. A. Chang-She, "Low cost RFID authentication protocol with forward privacy," *Chinese Journal of Computers*, vol. 34, no. 8, pp. 1387–1398, 2011.

[2] C. L. Chen, Y. L. Lai, C. C. Chen, Y. Y. Deng, and Yu C. Hwang, "RFID ownership transfer authorization systems conforming epcglobal class-1 generation-2 standards," *International Journal of Network Security*, vol. 13, no. 1, pp. 41–48, 2011.

[3] Y. C. Chen, W. L. Wang, M. S. Hwang, "RFID authentication protocol for anti-counterfeiting and privacy protection", in *The 9th International Conference on Advanced Communication Technology*, pp. 255–259, 2007.

[4] P. Y. Cui, "An improved ownership transfer and mutual authentication for lightweight RFID protocols," *International Journal of Network Security*, vol. 18, no. 6, pp. 1173–1179, 2016.

[5] R. Doss, W. Zhou, and S. Yu, "Secure RFID tag ownership transfer based on quadratic residues," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 2, pp. 390–401, 2013.

[6] S. Fouladgar and H. Afifi, "An efficient delegation and transfer of ownership protocol for RFID tags," *The First International Eurasip Workshop on RFID Technology*, 2007.

[7] M. S. Hwang, C. H. Wei, C. Y. Lee, "Privacy and security requirements for RFID applications", *Journal of Computers*, vol. 20, no. 3, pp. 55–60, Oct. 2009.

[8] Y. Jin, Q. Wu, Z. Shi, X. Lu, and L. Sun, "RFID lightweight authentication protocol based on PRF," *Journal of Computer Research and Development*, vol. 51, no. 7, pp. 1506–1512, 2014.

[9] G. Kapoor and S. Piramuthu, "Single RFID tag ownership transfer protocols," *IEEE Transactions on Systems Man and Cybernetics Part C*, vol. 42, no. 2, pp. 164–173, 2012.

[10] L. Lu, "Wireless key generation for RFID systems," *Chinese Journal of Computers*, vol. 38, no. 4, pp. 822–832, 2015.

[11] D. Molnar, A. Soppera, and D. Wagner, "A scalable, delegatable pseudonym protocol enabling ownership transfer of RFID tags," in *International Conference on Selected Areas in Cryptography*, pp. 276–290, 2005.

[12] Q. Qian, Y. L. Jia, R. Zhang, "A lightweight RFID security protocol based on elliptic curve Cryptography," *International Journal of Network Security*, vol. 18, no. 2, pp. 354–361, 2016.

[13] J. W. Shen and J. Ling, "Improved ultra-lightweight authentication of ownership transfer protocol for RFID tag," *Computer Science*, vol. 41, no. 12, pp. 125–128, 2014.

[14] B. Song and C. J. Mitchell, "RFID authentication protocol for low-cost tags," in *ACM Conference on Wireless Network Security (WISEC'08)*, pp. 140–147, 2008.

[15] B. Song and C. J. Mitchell, "Scalable RFID security protocols supporting tag ownership transfer," *Computer Communications*, vol. 34, no. 4, pp. 556–566, 2011.

[16] S. Wang, S. Liu, and D. Chen, "Scalable RFID mutual authentication protocol with backward privacy," *Journal of Computer Research and Development*, vol. 50, no. 6, pp. 1276–1284, 2013.

[17] C. H. Wei, M. S. Hwang, A. Y. H. Chin, "A mutual authentication protocol for RFID", *IEEE IT Professional*, vol. 13, no. 2, pp. 20–24, Mar. 2011.

[18] H. U. Wei, L. I. Yong-Zhong, and L. I. Zheng-Jie, "New defending RFID authentication protocol against dos attacks," *Application Research of Computers*, vol. 29, no. 2, pp. 676–675, 2012.

[19] M. H. Yang, "Secure multiple group ownership transfer protocol for mobile RFID," *Electronic Commerce Research and Applications*, vol. 11, no. 4, pp. 361–373, 2012.

## Biography

**Rui Xie** received his B.S. in electrical engineering and automation from Dalian Maritime University in 2000, and the M.Sc. in Computer Science from Guangdong University of Technology in 2003. He is currently a Ph.D Candidates in Guangdong University of Technology. His research interests cover a variety of different topics including network security, machine learning, cloud computing, data mining and their applications.

**Bi-yuan Jian** received a master's degree in School of Computer Science and Engineering from South China University of Technology (China) in June 2011. He is a lecturer in School of Electronic and Computer Engineering in Guangzhou Vocational College of Science and Technology. His current research interest fields include information security and computer application.

**Dao-wei Liu** received a master's degree in School of Computers from Guangdong University of Technology (China) in June 2016. He is a lecturer in School of Electronic and Computer Engineering in Guangzhou Vocational College of Science and Technology. His current research interest fields include information security.