A Secure ECC-based Mobile RFID Mutual Authentication Protocol and Its Application

Shin-Yan Chiou, Wen-Tsai Ko, and Erl-Huei Lu
(Corresponding author: Shin-Yan Chiou)

Department of Electrical Engineering, Chang Gung University
259, Wen-Hwa 1st Road, Kwei-Shan, Tao-Yuan, Taiwan
(Email: corresponding_ansel@mail.cgu.edu.tw)
(Received Feb. 15, 2017; revised and accepted June 25, 2017)

Abstract

Mobile RFID applications combine RFID technologies and mobile device to create a new convenient application area. However, most of the applications suffer from the security issues due to insecure communication channels among tags, readers and servers. In 2012, Zhou et al. proposed an ECC-based mutual authentication protocol to promote mobile RFID applications security. However, we found their protocol faces to OTRUTS (one time reading and unlimited times service) problem, which means once a reader read the data of a certain tags from a server successfully, the reader can read it unlimited times without reading those tags again. Therefore, their protocol cannot securely support some mobile RFID applications such as the security patrolling application. In this paper, we propose a new secure ECC-based mobile RFID mutual authentication protocol for the safety of mobile RFID applications such as security patrolling.

Keywords: ECC; Mutual Authentication; RFID

1 Introduction

Recently, many researches have concentrated on mobile RFID-based applications [1, 2, 4, 12, 19, 23, 24, 26, 27, 29, 30] as it is believed that this type of applications have advantages of both RFID technology and mobile smart device. In most of the traditional RFID applications, it is assumed that the communication between reader and backend server is wired and secure, while it between tag and reader is wireless and insecure. This is because readers are usually installed at a fixed location but the tags are mobile.

However, in mobile RFID applications, both tag-reader and reader-backend server communication channels are in wireless transmission mode and therefore considered to be insecure. In the mobile RFID based telecommunication service, the tags (different from those of traditional RFID applications) are designed to be stationary and the readers (installed in a mobile device such as a cell phone) are movable. The mobile RFID telecommunication services provide tag information which is stored and maintained in backend database over a reader embedded mobile network to support many applications such as mobile payment [5, 11, 18, 21], emergency response [6, 15, 20, 23], marketing [2], advertisements promotion [14], security patrolling, position reporting, etc. Therefore, the mobile device, with an embedded reader, could be used by a potential customer, a consumer, a security patrolman, etc. That means the holder of the mobile device could also be an adversary to the mobile RFID system.

ECC is proved to be suitable for RFID applications [3, 7, 8, 10, 13, 17, 22, 25]. In 2012, Zhou et al. [28] proposed a mutual authentication protocol based on public-key cryptography using ECC for mobile RFID application.

However, their protocol has OTRUTS (one time reading and unlimited times service) problem, which means once a reader read the data of a certain tags from a server successfully, the reader can read it unlimited times without reading those tags again. Therefore, their protocol cannot securely support some mobile RFID applications such as the security patrolling application. In this paper, we propose a new secure ECC-based mobile RFID mutual authentication protocol for the safety of mobile RFID applications such as security patrolling.

The rest of this paper is organized as follows. Second section provides a brief background of ECC. In the third and forth section, we review and analyze Zhou et al.’s protocol. The proposed scheme is demonstrated in fifth section. Sixth section provides security analysis. Finally, we draw conclusions in seventh section.

2 Preliminaries

For ECC application, a non-singular elliptic curve should be chosen. All points in a non-singular elliptic curve ($y^2 = x^3 + ax + b(\text{mod } p)$) have tangent lines except
one point at infinity, where \( p > 3 \) and \( 4a^3 + 27b^2 \neq 0 \). The security of ECC is based on the intractability of the following problems.

### 2.1 Elliptic Curve Discrete Logarithm Problem (ECDLP)

Given an elliptic curve \( E \) defined over a finite field \( F_q \), denoted by \( E(F_q) \). There is a point \( P \in E(F_q) \) with order \( m \) and a point \( Q \in < P > \). Then the problem of finding the integer \( k \in [0, m - 1] \) from given \( P \) and \( Q \) such that \( Q = kP \) is defined as ECDLP, where \( k \) is the discrete logarithm of \( Q \) to the base \( P \), denoted \( k = \log_P Q \) [9].

### 2.2 Elliptic Curve Computational Diffie-Hellman Problem (ECCDHP)

From three given points \( P, xP \) and \( yP \) over \( E(F_q) \), it is hard to compute \( xyP \) over \( E(F_q) \).

### 2.3 Elliptic Curve Factorization Problem (ECFP)

From two given points \( P \) and \( Q \) over \( E(F_q) \), where \( Q = xP + yP \), it is hard to find two points \( xP \) and \( yP \) over \( E(F_q) \) [16].

### 3 Zhou et al.’s Protocol

In this section, we provide a brief introduction to the notations and Zhou et al.’s protocol. Table 1 shows the notations used in our and Zhou et al.’s scheme. In table 1, \( h() \) is an one-way hash function, where \( h : \{0, 1\}^* \rightarrow \{0, 1\}^{2m}, \) \( m \) is the bit length of the coordinate \( x \) or \( y \) of a point over the elliptic curve \( E(F_q) \).

<table>
<thead>
<tr>
<th>Notation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P )</td>
<td>a base point of a subgroup on elliptic curve ( E(F_q) )</td>
</tr>
<tr>
<td>ID1</td>
<td>the identity of Tagi</td>
</tr>
<tr>
<td>( x(T) )</td>
<td>the x-coordinate of a point ( T = (x_i, y_i) )</td>
</tr>
<tr>
<td>( y(T) )</td>
<td>the y-coordinate of a point ( T = (x_i, y_i) )</td>
</tr>
<tr>
<td>( t_R )</td>
<td>time</td>
</tr>
<tr>
<td>( k_1, k_2 )</td>
<td>secrets of Server</td>
</tr>
<tr>
<td>( k_3 )</td>
<td>private key of Server</td>
</tr>
<tr>
<td>( K )</td>
<td>public key of Server</td>
</tr>
<tr>
<td>( T_P )</td>
<td>pseudo-id of Tagi</td>
</tr>
<tr>
<td>( R_P )</td>
<td>public key of Reader</td>
</tr>
<tr>
<td>( r )</td>
<td>private key of Reader</td>
</tr>
<tr>
<td>( h() )</td>
<td>one-way hash function</td>
</tr>
<tr>
<td>( (S)_L )</td>
<td>the left half bits of a binary sequence ( S )</td>
</tr>
<tr>
<td>( (S)_R )</td>
<td>the right half bits of a binary sequence ( S )</td>
</tr>
</tbody>
</table>

Zhou et al.’s protocol (Figure 1) has four phases: (1) **Initialization Phase**, (2) **Mobile Reader Authentication Phase**, (3) **Tag Authentication Phase** and (4) **Tag Information Sending Phase**. Those phases are described as follows.

#### 3.1 Initialization Phase

In this phase, **Server** chooses an elliptic curve \( E(F_q) \) and a base point \( P \) over \( E(F_q) \) with order \( n \), where \( n \) is a large prime number. **Server** randomly chooses his secrets \( k_1 \) and \( k_2 \in_R Z_q^* \) and his private key \( k_3 \in_R Z_q^* \), and computes his public key \( K = k_3P \). Next, **Server** computes pseudo-ids \( T_P = k_{1}^{-1}ID_i + k_3P \) for each tag and writes \( T_P \) into Tagi’s memory. On the other hand, **Reader** randomly chooses his private key \( r \in_R Z_q^* \) and computes his public key \( RP = rP \).

#### 3.2 Mobile Reader Authentication Phase

In this phase, **Reader** randomly chooses \( s \in_R Z_q^* \), computes \( Q = sP \), and sends a request \( Q \) to Tagi. After Tagi receives \( Q \), it chooses a random number \( t \in_R Z_q^* \) and sends \( t \) to Reader. Next, **Reader** computes \( v = rt - s \) and sends \( v \) to Tagi. **Tag** checks whether \( vP + Q \) is equal to \( tRP \). If it does, Tagi authenticates the **Reader** successfully. Otherwise, Tagi aborts this communication.

#### 3.3 Tag Authentication Phase

In this phase, the Tagi first chooses a random number \( c \in_R Z_q^* \), computes \( T_1 = cP, T_2 = cQ, T_3 = cK, T_4 = T_P + T_3 \) and \( u = h(\overline{\pi(T_2)}, \overline{\gamma(T_3)}) \), and sends \( T_1, T_4 \) and \( u \) to **Reader**. After **Reader** receives them, it computes \( R_1 = sT_1 \) and \( w = h(\overline{x(R_2)}, \overline{\gamma(T_4)}) \), and checks whether \( w \) is equal to \( u \). If it does not, **Reader** aborts this session. Otherwise, **Reader** considers \( T_1, T_4 \) and \( u \) as valid parameters. Next, **Reader** chooses a random number \( q \), extracts time \( t_R \), computes \( R_2 = qP, R_3 = (r + g)K \) and \( d_R = h(\overline{y(R_3)}, \overline{x(T_1)}, \overline{\gamma(T_4)}, t_R) \), and sends \( T_1, T_4, R_2, t_R \) and \( d_R \) to **Server**. After **Server** receives these messages, it checks whether \( t_R \) is valid. If it does, **Server** computes \( B_1 = k_3(R_P + R_2) \) and \( d_B = h(\overline{y(B_1)}, \overline{x(T_1)}, \overline{\gamma(T_4)}, t_R) \). Otherwise, **Server** aborts this session. **Server** checks whether \( d_B = d_R \) holds, and considers \( T_1, T_4, R_2, t_R \) and \( d_R \) as valid parameters and authenticates **Reader** successfully. Next, **Server** computes \( ID_i = k_3(B_2 - k_3P) \) and checks whether \( ID_i \) exists in the database. If it does, **Server** authenticates Tagi successfully. Otherwise, **Server** aborts this session.

#### 3.4 Tag Information Sending Phase

In this phase, **Server** fetches the related \( DATA_i \) of \( ID_i \) from the database, encrypts it, and sends the encrypted data to **Reader**. **Server** first chooses a random number \( l \in_R Z_q^* \), computes \( B_3 = lP, B_4 = lR_P, B_5 = k_3RP \), \( d_1 = \overline{y(B_4)} \oplus (DATA_i)_L \| \overline{x(B_5)} \oplus (DATA_i)_R \) and \( d_2 =\)
4 Analysis on Zhou et al.'s Protocol

In Zhou et al.'s protocol, we find that once Reader has read Tag's data, then the Reader can get Tag's data from Server without reading Tag again. We name this problem as "One Time Reading, Unlimited Times Service (OTRUTS)." The detail of this problem is described as follows and shown in Figure 2.

Assume Reader read Tag, once get the data of Tag from Server successfully. Reader have valid T1 and T4. As Reader wants Tag's new DATAi, from Server, he can assign T'1 = T1, T'2 = T2, extract a new time t'R', generate a random number g'εR'Z_q, compute R'_2 = g'P, R'_3 = (r + g')K and d'R' = h(g(R'_3), x(T'_1), x(T'_2), t'R'), and send T'_1, T'_4, R'_2, t'R', d'R' to Server to request the DATAi of Tag. As Server received those messages from Reader, if Server authenticates t'R' successfully (with no doubt), computes B'_1 = k3(RP + R'_2) and d'_R = h(g(B'_1), x(T'_1), x(T'_2), t'_R), finds out d'_R = d'R', holds, and computes ID1 = k1(B'_2 - k2P). Thus Server can successfully find ID in database because B'_2 has the pseudo-id information T1. Therefore, Server can fetch the DATAi and process the "TagInformationSendingPhase" to encrypt DATAi for Reader's request. Server then generates a random number l'RZ_q, computes B'_1 = l'P, B'_4 = l'RP, B'_5 = k3RP, d'_1 = y(B'_1) ⊕ (DATAi), d'_2 = h(x(B'_3), d'_1, t'_R + 1), and sends B'_2, d'_1 and d'_2 to Reader. After Reader receives those message, it computes R'_4 = rB'_3 and R'_5 = rK. Thus, Reader can recover DATAi1 = (d'_1) ⊕ y(R'_3)) ⊕ (d'_2) ⊕ x(R'_3). Therefore, in Zhou et al.'s protocol, Reader just needs to read Tag's DATAi one time, then he can read Tag's DATAi from Server with unlimited times without reading Tag again.

5 Proposed Protocol

In this section, we propose a mobile RFID-based mutual authentication protocol using elliptic curve cryptography for security patrolling application. In our protocol, we fix the OTRUTS problem of Zhou et al.'s protocol and make our protocol suitable to secure applications such as security patrolling.

We take a security patrolling scenario as an instance. In the security patrolling scenario, there are three roles: (1) Server as the Security Management Center (SMC), (2) Reader as the patrolman's Reader (PMR), and (3)Tagi as the sentry post's Tag (SPTi). Our protocol has four phases: (1) Initialization Phase, (2) SPTi to PMR Authentication Phase, (3) SMC to PMR and SPTi Authentication Phase and (4) DATAi Sending Phase. These phases are described as follows and shown in Figure 3.

5.1 Initialization Phase

The initialization phase is same as Zhou et al.'s protocol. SMC chooses an elliptic curve E(Fq) and a base point P over E(Fq) with order n, where n is a large prime number. SMC chooses two secrets k1, k2 ε R Z_q and one private

<table>
<thead>
<tr>
<th>Tag</th>
<th>Reader</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>T_i = k_i + k_iP</td>
<td>R_i = k_P</td>
<td>k_i, k_i, k_i = k_iP</td>
</tr>
<tr>
<td>t is a random number</td>
<td>Q is a random number</td>
<td>ID = k_1(R_i + R_p)</td>
</tr>
<tr>
<td>If vt + Q = tR_p then</td>
<td>v is a random number</td>
<td>Does tR_p is valid?</td>
</tr>
<tr>
<td>c is a random number</td>
<td>w is a random number</td>
<td>B_i = T_i - k_iT_p</td>
</tr>
<tr>
<td>T_i = cP, T_i = cQ</td>
<td>x = f(T_i, f(T_i), f(T_i, t_i))</td>
<td>ID = k_2(B_i - k_iP)</td>
</tr>
</tbody>
</table>

Figure 1: Zhou et al.'s protocol
Figure 2: OTRUTS problem

Figure 3: Proposed protocol
key \( k_3 \in R \mathbb{Z}_q^* \), and computes his public key \( K = k_3 P \). Each tag has pseudo-id \( T_P = k_1^{-1} D_1 + k_2 P \). On the other hand, \( PMR \) chooses his private key \( r \in R \mathbb{Z}_q^* \) and computes his public key \( R_P = r P \).

### 5.2 SPT\(_i\) to PMR Authentication Phase

In this phase, \( PMR \) randomly chooses \( s \in R \mathbb{Z}_q^* \), extracts times \( t_R \) and computes \( Q = s P \). Then \( PMR \) sends a request, \( Q \) and \( t_R \), to SPT\(_i\). After SPT\(_i\) receives \( Q \) and \( t_R \), it randomly chooses \( t \in R \mathbb{Z}_q^* \) and replies \( t \) to \( PMR \). After \( PMR \) receives \( t \), it computes \( v = rt - s \) and sends \( v \) to SPT\(_i\). SPT\(_i\) checks whether \( vP + Q = tR_P \) holds. If it does, SPT\(_i\) authenticates the \( PMR \) successfully. Otherwise, it aborts the communication.

### 5.3 SMC to PMR and SPT\(_i\) Authentication Phase

SPT\(_i\) randomly chooses \( c \in R \mathbb{Z}_q^* \), computes \( T_1 = cP \), \( T_2 = cQ \), \( T_3 = c t_R K \), \( T_4 = T_P + T_3 \) and \( u = h(\tilde{x}(T_3), \tilde{y}(T_3)) \), and sends \( T_1, T_4 \) and \( u \) to \( PMR \). After \( PMR \) receives \( T_1, T_4 \) and \( u \), it computes \( R_1 = sT_1 \) and \( w = h(\tilde{x}(R_2), \tilde{y}(T_4)) \), and checks whether \( w = u \). If it does, \( PMR \) authenticates the messages \( T_1, T_4 \) and \( u \) successfully. Otherwise, it aborts this session. Then \( PMR \) chooses a random number \( g \in R \mathbb{Z}_q^* \), computes \( R_2 = gP \), \( R_3 = (r + g)K \) and \( d_R = h(\tilde{y}(R_3), \tilde{x}(T_3), \tilde{x}(T_4), t_R) \), and sends \( T_1, T_4, R_2, t_R \) and \( d_R \) to \( Server \). Then \( Server \) checks whether \( t_R \) is valid. If it does not, \( Server \) aborts this session. Otherwise, \( Server \) computes \( B_1 = k_3 (R_P + R_2) \) and \( d_B = h(\tilde{y}(B_1), \tilde{x}(T_1), \tilde{x}(T_4), t_R) \), and checks whether \( d_B = d_R \) holds. If it does, \( Server \) considers \( T_1, T_4, R_2, t_R \) and \( d_R \) as valid parameters and authenticates \( Reader \) successfully. Next, \( Server \) computes \( B_2 = T_4 - k_3 t_R T_1 \) and \( ID_i = k_1 (B_2 - k_2 P) \), and checks whether \( ID_i \) exists in the database. If it does, \( Server \) authenticates \( Tag_i \) successfully. Otherwise, \( Server \) aborts this session.

### 5.4 DATA\(_i\) Sending Phase

In this phase, \( Server \) fetches the related \( DATA_i \) of \( ID_i \) from the database, encrypts it, and sends the encrypted data to \( PMR \). First, \( SMC \) randomly chooses \( l \in R \mathbb{Z}_q^* \), computes \( B_3 = lP \), \( B_4 = lR_P \), \( B_5 = k_3 R_P \), \( d_1 = \tilde{y}(B_4) \oplus (DATA_i)_L \parallel \tilde{x}(B_5) \oplus (DATA_i)_R \) and \( d_2 = h(\tilde{x}(B_5), d_1, t_R + 1) \), and sends \( B_3, d_1 \) and \( d_2 \) to \( PMR \). After \( PMR \) receives those messages, it computes \( R_4 = rB_3 \), \( R_5 = rK \), \( d_3 = h(\tilde{x}(R_5), d_1, t_R + 1) \), and checks whether \( d_3 = d_2 \) holds. If it does, \( PMR \) believes the parameters \( B_3, d_1 \) and \( d_2 \) comes from a valid \( SMC \), and recovers \( DATA_i = (d_1)_L \oplus \tilde{y}(R_4) \parallel (d_1)_R \oplus \tilde{x}(R_5) \). Otherwise, it aborts this session.

### 6 Security Analysis

In the security patrolling scenario, \( PMR \) is supposed to visit the assigned \( SPT \) in person, read the \( SPT \) and send proof back to the \( SMC \) for verification in a valid time interval. If a protocol has the OTRUTS problem (described in section 3), \( PMR \) just needs to visit \( SPT \) only one time then he can sit on the chair in the security office and complete the patrolling report without visiting the same \( SPT \) again. Therefore, a security patrolling application should avoid the OTRUTS problem in the RFID mobile mutual authentication protocol.

In our protocol, we rearranged \( T_3 = ct_R K \) to solve this OTRUTS problem. Thus, we have \( T_4 = T_P + ct_R K \). If \( PMR \) tries to read \( SPT_i \)'s data from \( SMC \) without reading \( SPT_i \) again, shown as Figure 4, he assigns \( T_1 = T_1 \) and \( T_2 = T_2 \), extracts a new time \( t'_R \), generates a random number \( g' \in R \mathbb{Z}_q^* \), computes \( R'_2 = g'P \), \( R'_3 = (r + g')K \) and \( d'_R = h(\tilde{y}(R'_3), \tilde{x}(T'_1), \tilde{x}(T'_2), t'_R) \), and sends \( T'_1, T'_4, R'_2, t'_R, d'_R \) to \( SMC \). After \( SMC \) receives these messages, it authenticates \( t'_R \) successfully (with no doubt), computes \( B'_1 = k_3 (R_P + R'_2) \) and \( d'_B = h(\tilde{y}(B'_1), \tilde{x}(T'_1), \tilde{x}(T'_4), t'_R) \), finds \( d_B = d_R \) holds, and compute \( B'_2 = T'_4 - k_3 t'_R T'_1 \) = \( T_P + (t_R - t'_R) c k_3 P \). Now SMC tries to recover \( ID_i \) by computing \( ID'_i = k_1 (B'_2 - k_2 P) = k_1 (T_P + (t_R - t'_R) c k_3 P - k_2 P) = ID_i + (t_R - t'_R) c k_3 P \neq ID_i \). However, \( SMC \) finds out \( ID'_i \) is not in the database and aborts the session. Therefore, our protocol not only provides the security properties of Zhou et al.'s protocol, but also resists to OTRUTS problem which make our protocol more suitable for security patrolling application.

### 7 Conclusions

This paper discusses the Zhou et al.’s mutual authentication protocol and points out their protocol is faces OTRUTS problem and therefore cannot securely support some mobile RFID applications such as the security patrolling application. This paper proposes a new mutual authentication using ECC and proved the proposed protocol is resistant to OTRUTS problem.

### Acknowledgments

This work is partially supported by the Ministry of Science and Technology under Grant MOST 104-2221-E-182-012 and by the CGMH project under Grant BMRPB46. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

### References

Figure 4: The resistance of OTRUTS problem in our protocol

<table>
<thead>
<tr>
<th>Tag</th>
<th>Reader</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_k = (k, ID_k, P)$</td>
<td>$r \in_k F'_k, R_k = (rP)$</td>
<td>$k, k_1, k_2, k_3, k_4 = (k, P)$</td>
</tr>
<tr>
<td>$T'_k = T_k, T_k = T_k$</td>
<td>$t_k$ (timestamp),</td>
<td>Does $t_k$ is valid?</td>
</tr>
<tr>
<td>$g \in_k F'_k$</td>
<td>$R_k = gP,$</td>
<td>$R_k = k(R_k + R_r)$</td>
</tr>
<tr>
<td>$R_k = (r + g)k,$</td>
<td>$d_k = h(f(T_k), f(T_k), T_k, t_k).$</td>
<td>$d_k = d_k,$ then</td>
</tr>
<tr>
<td>$d_k = h(f(T_k), f(T_k), T_k, t_k).$</td>
<td>$T'_k = T_k - kyT'_r,</td>
<td>ID_k = k(ID_k - kyP)</td>
</tr>
<tr>
<td>$T'_k, R_k, d_k,$</td>
<td>$ID_k = k(ID_k - kyP)</td>
<td>= ID_k.$</td>
</tr>
<tr>
<td>$d_k,$</td>
<td>$= ID_k.$</td>
<td>Does $ID_k$ in DB?</td>
</tr>
</tbody>
</table>

Session Aborted.

Figure 4: The resistance of OTRUTS problem in our protocol


**Biography**

**Shin-Yan Chiou** received the PhD degree in Electrical Engineering from National Cheng Kung University, Taiwan, in 2004. From 2004 to 2009, he worked at Industrial Technology Research Institute as a RD Engineer. Since 2009, he joined the faculty of the Department of Electrical Engineering, Chang Gung University, Taoyuan, Taiwan, where he is currently an Associate Professor. His research interests include information security, cryptography, social network security, and secure applications between mobile devices.

**Wen-Tsai Ko** received the B.S. degree in Applied Mathematics from Chung Cheng Institute of Technology in 1986, the M.B.A. degree in Defense Information from National Defense Management College in 1998, and the PhD degree in Electrical Engineering from Chang Gung University in 2014. His research interests include information security, visual cryptography, and RFID security.

**Erl-Huei Lu** received the B.S. and M.S. degrees in electrical engineering from Chung Cheng Institute of Technology, Taiwan, in 1974 and 1980, respectively, and the Ph.D. degree electrical engineering from National Cheng Kung University, Taiwan, in 1988. Lu is a professor in the Department of Electrical Engineering, Chang Gung University, Taiwan. His research interests include error-control coding, network security, and systolic architectures.