A Mobile RFID Authentication Scheme Based on the COMP-128 Algorithm

Jia-Ning Luo[1] and Ming Hour Yang[2]
[1]*Information and Telecommunication, Ming Chuan University*
[2]*Information Computer Science, Chung Yuan Christian University*
Taoyuan, Taiwan

1. Introduction

Radio frequency identification (RFID), based on the MIT Auto-ID project [1], is a technology that uses wireless transmission to identify an object. RFID is seeing increased use in various industries as an alternative to the bar code. An RFID system consists of three components: the reader, the tag, and the back-end database. Some advantages of an RFID system are that it does not require direct contact with the tag, and can scan multiple tags simultaneously. However, because the reader uses wireless technology to communicate with the tag and the EPC Class 1 Gen 2 protocol [2] does not have a well-designed access mechanism to protect the tag data privacy and location privacy, a malicious attacker is able to retrieve the tag's information by listening to the traffic between the reader and the tag [3]. To protect the information stored on a tag, Juels [5] and Weis [6] proposed methods for a tag to lock or destroy itself when attacked. However, these methods are an inconvenience to normal users. Many studies [7] propose authentication mechanisms in RFID systems, in which only authorized readers can read the correct information storing on the tag. However, due to hardware limitations, an RFID tag cannot perform complex operations, such as traditional symmetric and asymmetric encryption algorithms.

Previous research proposes using the simple XOR operation to encrypt messages in RFID authentication protocols. Some studies use the RFID tag's built-in CRC function to achieve message authentication [8]. Other studies [4][9][10][11] use the one-way hash function to enhance authentication protocol security. This study briefly explains these authentication mechanisms and analyzes existing security issues.

Karthikeyan [12] proposed a mutual authentication scheme that uses two matrices and the corresponding anti-matrix. In this approach, the multiplication of a vector key and the matrix serves as an authentication index for the tag. However, in Karthikeyan's scheme, the tag does not verify reader's return value; that is, the attacker can re-send the message to track tag's location.

Duc [8] used the built-in CRC function of an RFID tag to generate a message authentication code (MAC) consisting of a random number and a secret previously shared between the tag and the reader. Duc uses the MAC to authenticate the tag and update the pre-shared secret. However, Duc's scheme cannot prevent the forge attack and it does not have forward security. To enhance Karthikeyan and Duc's scheme, Chien [13] proposed a synchronization

Source: Radio Frequency Identification Fundamentals and Applications, Bringing Research to Practice, Book edited by: Cristina Turcu, ISBN 978-953-7619-73-2, pp. 278, February 2010, INTECH, Croatia, downloaded from SCIYO.COM

authentication protocol based on CRC. However, because CRC is a linear function, no protocols based on CRC can resist the forge attack.

Other studies use a one-way hash function in the RFID authentication mechanism [4][9][10][11][14]. Henrici proposed a *hash based scheme* [9] in which the tag sends h(ID) instead of its unique ID to the reader. Henrici's scheme protects the tag's location privacy because the attacker cannot derive the tag's ID from h(ID).

In Henrici's scheme, if the message between reader and tag is lost, the tag will be out-of-sync. To improve Henrici's scheme, Yang proposed a novel mutual authentication mechanism [10] that uses index-pseudonyms and XOR method. In this case, the tag generates a hash value for a random number sent by the reader. This hash value is used as the tag's pseudonym. In Yang's protocol, an attacker can trace the tag's location because the current authentication message sent from tag to the reader can be derived from the last authentication message.

Ohkubo proposed an authentication scheme that uses the hash chain technique to renew the secret information stored in the tag [4]. The tag's ID is derived from two hash functions, *G* and *H*. However, in Ohkubo's scheme, the database must perform an exhaustive search to find the matching tag ID, which creates a computing burden in the database. Further, Ohkubo's scheme cannot avoid replay attacks.

Chan [3] proposed an authentication scheme that uses the Chameleon Hash algorithm to update the tag's ID and protect the tag's location privacy. In Chen's algorithm, the database uses the authentication information from the previous session to derive the tag's current ID, which means an exhaustive search of the database is not required. Lee [16] proposed an authentication scheme based on a hash function to protect communication between the tag and the reader. In this approach, the tag's ID is updated concurrently in the database and the tag. Lee's scheme is resistant to replay attacks and man-in-the-middle attacks, and provides location privacy.

Other studies discuss how to embed the RFID reader into a mobile phone, which then serves as a mobile RFID reader [18][19]. In the mobile RFID environment, any user that holds a mobile RFID reader can retrieve any tag's information. As a result, RFID security problems become even more serious in the Mobile RFID environment [20].

In the Mobile RFID environment, a mobile RFID reader is able to move freely and read any tags nearby. The database must determine the reader's identity before providing it with tag information. Therefore, authentication schemes must be modified to accommodate this feature. For example, in Lee's and Chan's schemes, the reader forwards the authentication message between the database and the tag, and the database always trusts the reader. In the Mobile RFID environment, however, the reader cannot be trusted, and the communication channel between the reader and the database is not secure [3][16].

This paper proposes an authentication mechanism based on the COMP-128 algorithm [21][22], called *COMP-128 in Mobile RFID Authentication Protocol* (C-MRAP), for use in Mobile RFID environments. C-MRAP uses the A3 algorithm in COMP-128 to encrypt messages, and uses the A8 algorithm in COMP-128 to update the authentication key and session key between the database and the tag. In C-MRAP, the database, the mobile reader, and the tag authenticate each other, and the transmission messages between them are encrypted to provide robust security.

This paper is organized as follows. The second section discusses related studies. The third section presents the C-MRAP algorithm. The fourth section performs security analysis and performance analysis, while the fifth section draws conclusions.

2. Related works

The previous section briefly discusses some RFID authentication protocols and their security issues. This section describes the Mobile RFID architecture, the authentication protocols used in Chan [3] and Lee [16], and the COMP-128 algorithm used in the current GSM architecture.

2.1 Mobile RFID architecture

Figure 1 illustrates the typical Mobile RFID environment, in which each user can read the product information of RFID tags through a combination of mobile phone and RFID reader devices. For example, a consumer using a mobile phone's RFID reader can read the tag on a movie poster, and then link to the RFID database to download movie-related information and release dates, and reserve tickets online.

In Figure 1, the communication between the authentication server (AS) and back-end database is secure. But the channel between the tag and the mobile reader, and the channel between the mobile reader and the database are insecure.

The operations of a Mobile RFID system are as follows:

1. The mobile reader sends a request to the tag. The tag generates a message containing the authentication message, and sends it to the reader. The reader then forwards the message to the authentication server to validate the tag's identity.
2. If the tag is valid, the authentication server sends the key updated messages through the reader to the tag.
3. The tag replies with a successful update message through the reader to the authentication server.
4. The authentication server sends the tag's information to the reader as soon as it receives the acknowledgement message from the reader.
5. The reader connects to the back-end database through the AS to get extra services, e.g., booking a ticket.

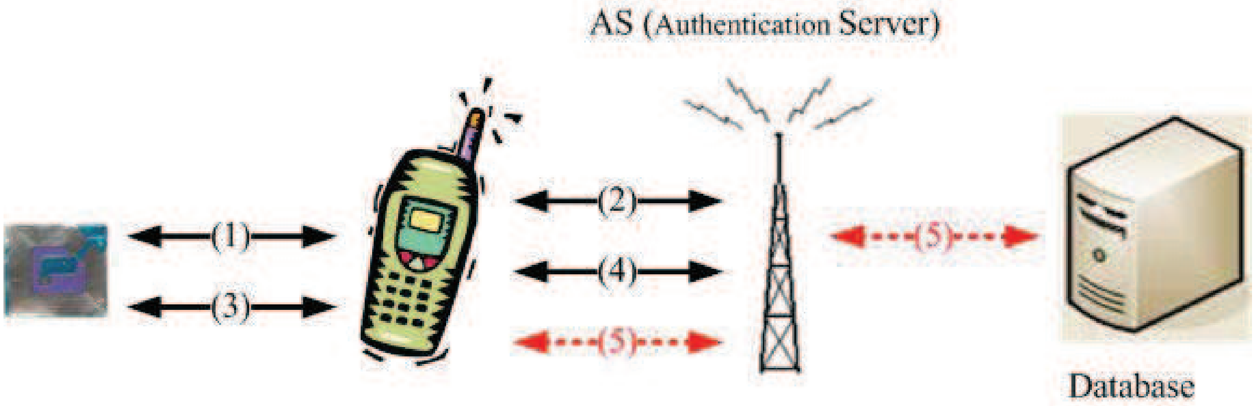


Fig. 1. The Mobile RFID architecture

2.2 Chan's protocol

Chan [3] proposed an RFID authentication protocol based on the Chameleon hash algorithm [26]. A chameleon hash function is associated with a pair of public and private keys. A user R generates a key pair, a public key HK_R and a private key CK_R, according to a given generation function. The chameleon hash functon, denoted $CH_R(m_1,r_1)$, can be computed easily by using R's public key HK_R, where m_1 and r_1 are two strings. The chameleon hash function has two important properties: collision resistant and trapdoor collisions. For two messages m_1 and m_2, where $m_1 \neq m_2$, it's hard to find a collision that $CH_R(m_1,r_1) = CH_R(m_2,r_2)$ by using R's public key HK_R (the collision resistant property). However, it is easily to find a collision that $CH_R(m_1,r_1) = CH_R(m_2,r_2)$ by using R's private key CK_R (the trapdoor collisions property).

Table 1 shows the terminology of Chan's protocol. The database and every tags shares five variables: a unique serial number *CID*, a transaction counter *TID*, the last transaction counter *LST*, and two random numbers SN_1, and SN_2. The *TID* increases in each transaction, and the *LST* will be set to the current *TID* if the authentication procedure is done successfully.

CID	Tag's identification
TID	The transaction counter of a tag
LST	The previous TID that authenticated successfully
SN_1, SN_2	Random numbers
REF	A pointer stored i-1th authentication data

Table 1. Terminology of Chan's protocol

Figure 2 shows Chan's protocol. When a reader sends a read request to a tag, the tag generates three random numbers, r_1, r_2, and r_3, and sends them to the reader (step 1). The reader forwards them to the database (step 2). The database uses the trapdoor property of Chameleon hash to calculate r_4 that satisfy $CH_R(r_1,r_2) = CH_R(r_3,r_4)$. Database sends r_4 to the tag (step 3). The tag checks if $CH_R(r_1,r_2) = CH_R(r_3,r_4)$. The tag then performs the following operations:

1. Increases TID_i by 1.
2. Calculates $\Delta TID=TID_i-LST_{i-1}$
3. Generates three chameleon hash values: $K_i=CH_R(r_1,r_2)$, $HID_{i-1}=CH_R(CID_{i-1},SN_2)$, and $CH_R(CID_{i-1}, TID_{i-1})$.
4. Uses A5/1 algorithm [21] to encrypts the two variables ($HID_{i-1} || \Delta$ TID) and $CH_R(CID_{i-1}, TID_{i-1})$ by the key K_i to construct $M_1=E_{Ki}((HID_{i-1} || \Delta \text{ TID}) || CH_R(CID_{i-1}, TID_{i-1}))$.
5. Sends M_1 to the database (step 4).

After the database receives M_1, the database uses K_i to decrypt the message and gets ($HID_{i-1} || \Delta$ TID) and $CH_R(CID_{i-1}, TID_{i-1})$. Because the database does not know tag's identity, it searches *HID* by calculating $CH_R(CID, SN_2)$ for all tags. If there is a match, the database performs the following operations:

1. Updates $TID_i = HID_{i-1} || \Delta$ TID
2. Verifies $CH_R(CID_{i-1}, TID_{i-1})$
3. Calculates $M_2= CH_R(TID_{i-1},CID_{i-1})$.
4. Sends M_2 to the tag (step 5).

When the tags receives M_2, it verifies whether $M_2= CH_R(TID,CID)$. Finally, both the database and the tag update CID_i and LST_i.

In a transaction of Chan's protocol, a tag should do six Chameleon hash operations and one *A5* encryption. The database should do *2n+5* Chameleon hash operations, one *A5* decryption, and a collision finding of Chameleon hash.

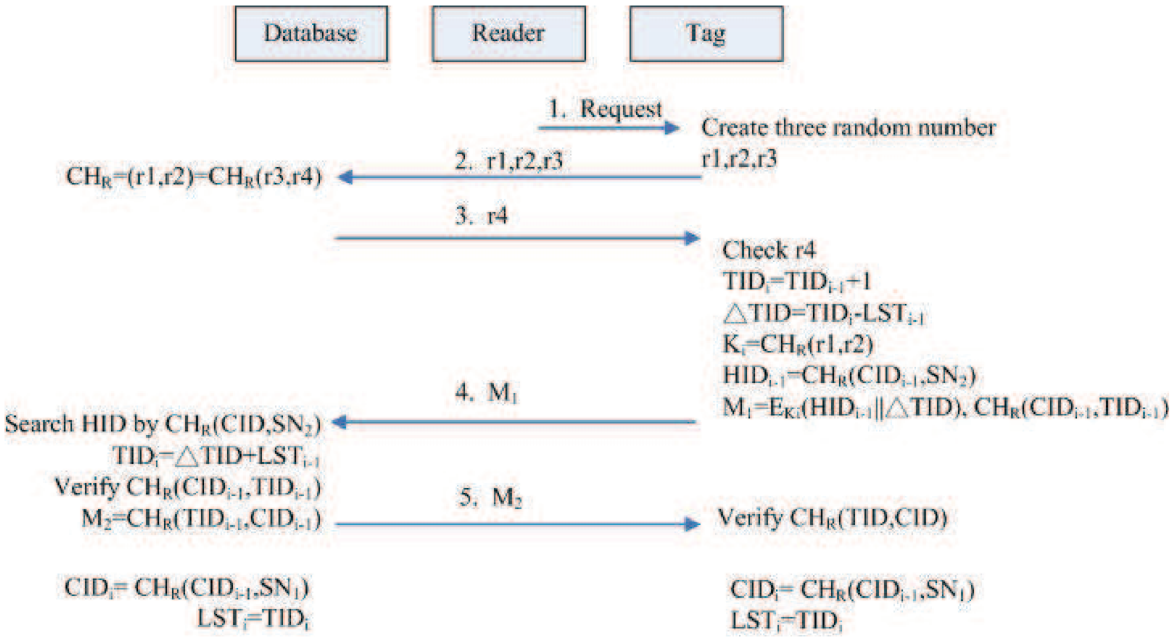

Database
Reader
Tag
1. Request
Create three random number r1,r2,r3
2. r1,r2,r3
CH_R=(r1,r2)=CH_R(r3,r4)
3. r4
Check r4
TID_i=TID_{i-1}+1
△TID=TID_i-LST_{i-1}
K_i=CH_R(r1,r2)
HID_{i-1}=CH_R(CID_{i-1},SN_2)
M_1=E_{Ki}(HID_{i-1}||△TID), CH_R(CID_{i-1},TID_{i-1})
4. M_1
Search HID by CH_R(CID,SN_2)
TID_i=△TID+LST_{i-1}
Verify CH_R(CID_{i-1},TID_{i-1})
M_2=CH_R(TID_{i-1},CID_{i-1})
5. M_2
Verify CH_R(TID,CID)
CID_i= CH_R(CID_{i-1},SN_1)
LST_i=TID_i
CID_i= CH_R(CID_{i-1},SN_1)
LST_i=TID_i

Fig. 2. Chan’s Protocol

2.3 Lee’s protocol

Lee proposed another RFID authentication protocol by using one-way hash function [16]. In Lee’s protocol, the database and every tags shares four variables: a unique serial number *CID*, a transaction counter *TID*, the last transaction counter *LST*, and a random number *SN*. The *TID* increases in each transaction, and the *LST* will be set to the current *TID* if the authentication procedure is done successfully.

Figure 3 shows Lee’s protocol. When a reader sends a read request to a tag, the tag performs the following operations:

1. Generates a random number *N*.
2. Increases *TID* by 1.
3. Calculates $\Delta \mathrm{TID}=\mathrm{TID}_i\text{-}\mathrm{LST}_{i\text{-}1}$.
4. Calculates the hash value of CID_{i-1}, where $HID_{i-1}=H(CID_{i-1})$.
5. Calculates another three hash values: $H(SN\oplus HID_{i-1}\oplus N)$, $H(SN\oplus N)$, and $H(CID_{i-1}\oplus TID_{i-1})$.
6. Constructs $M_1 = N||H(SN\oplus HID_{i-1}\oplus N)||\ \Delta TID\oplus H(SN\oplus N)||H(CID_{i-1}\oplus TID_{i-1})$, and sends M_1 to the database through the reader (step 2).

The database performs the following operations:

1. Searches *SN* by $H(SN\oplus HID\oplus N)$
2. Checks if the condition $LST+\Delta TID > TID_{i-1}$ holds
3. Updates $TID_i = LST+\Delta TID$
4. Verifies the tag’s identity by checking $H(CID_{i-1}\oplus TID_{i-1})$
5. Generates a random number *R* and updates *HID*, *CID*, *TID* and *LST* in the database if the tag is valid.
6. Constructs $M2= R\oplus H(SN\oplus (N+1))\ ||\ H(R\oplus CID_{i-1}\oplus TID_{i-1})$, and sends M_2 to the tag (step 3).

The tag then verifies $H(R\oplus CID_{i-1}\oplus TID_{i-1})$ by using *R*. If the value is correct, the tag updates CID_i and LST_i: $CID_i=H(R\oplus CID_{i-1})$ and $LST_i=TID$.

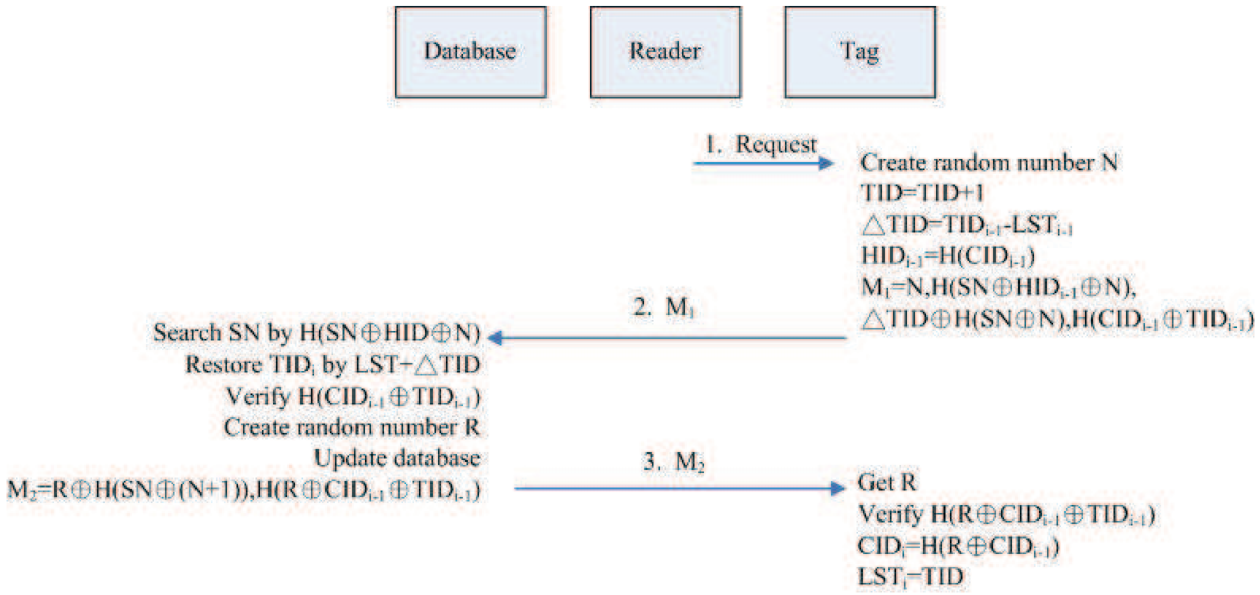

Database
Reader
Tag
1. Request
Create random number N
TID=TID+1
△TID=TID$_{i-1}$-LST$_{i-1}$
HID$_{i-1}$=H(CID$_{i-1}$)
M$_1$=N,H(SN⊕HID$_{i-1}$⊕N),
△TID⊕H(SN⊕N),H(CID$_{i-1}$⊕TID$_{i-1}$)
2. M$_1$
Search SN by H(SN⊕HID⊕N)
Restore TID$_i$ by LST+△TID
Verify H(CID$_{i-1}$⊕TID$_{i-1}$)
Create random number R
Update database
M$_2$=R⊕H(SN⊕(N+1)),H(R⊕CID$_{i-1}$⊕TID$_{i-1}$)
3. M$_2$
Get R
Verify H(R⊕CID$_{i-1}$⊕TID$_{i-1}$)
CID$_i$=H(R⊕CID$_{i-1}$)
LST$_i$=TID

Fig. 3. Lee's protocol

2.4 COMP-128 algorithm

The GSM authentication architecture uses the A3 algorithm of COMP-128 for authentication, the A8 algorithm for generating session keys, and the A5 algorithm for encryption. Table 2 shows the components used by the COMP-128 algorithm in the GSM network.

MS (Mobile station)	The mobile phone
SIM(Subscriber Identity Module)	The smartcard put into the mobile phone, to store the session key and perform simple operations
AuC (Authenticaton center)	The *AuC* authenticates each SIM card
BS(Base station)	The station communicates with the mobile phone
K_i	The key used for authenticatoin

Table 2. COMP-128 Terminology

In the GSM network, each mobile station shares a key K_i with the authentication center. A malicious attacker cannot get the K_i by sniffing all the packets in the air. Figure 4 shows the operation flow of the COMP-128 algorithms (A3, A5, and A8).

When the AuC wants to authenticate a SIM card in a mobile station, it generates a 128-bit random number (RAND) and delivers it to the MS through the BS. The MS then forwards the random number to the SIM card module. The SIM module computes $SRES = A3(K_i, RAND)$. The SIM module forwards SRES to the AuC to authenticate itself. If the SRES is correct, both the AuC and SIM module generate a session key $K_c = A8(K_i, RAND)$, which is used to encrypt all the messages between the MS and the BS.

3. COMP-128 in Mobile RFID Authentication Protocol (C-MRAP)

To improve Lee and Chan's schemes, this paper proposes a mutual authentication scheme, called the COMP-128 in Mobile RFID Authentication Protocol (C-MRAP), for the Mobile

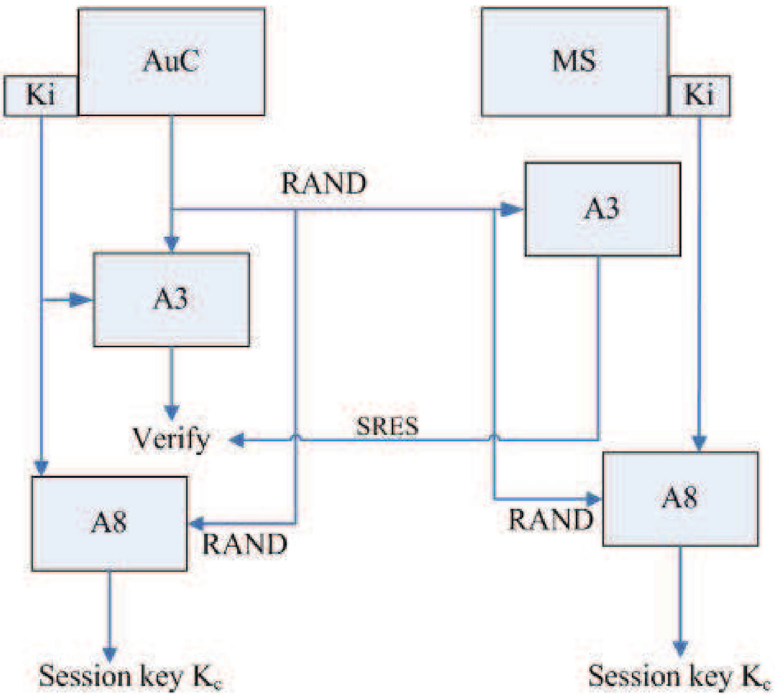


Fig. 4. COMP-128 Algorithms

RFID environment. There are three phases in the C-MRAP protocol. In the first phase, the database authenticates the mobile reader and the tag. The reader queries the tag and then sends a read request to the database by forwarding the tag's identity message. The database uses the session key shared with the reader to authenticate the reader's identity. The database then uses the information sent from the tag to verify the tag's identity. In the second phase, the database updates the authentication key with the tag after the database successfully authenticates the tag. The third phase is used to confirm the key update. The tag sends an update confirmation message to the database, and the database then sends the tag's information to the reader. Table 3 shows the information stored in the database, the reader, and the tag. In this approach, the tag shares four secrets with the database: *SN*, *Kc*$_i$, *UN*, and *PIN*. These variables are used to authenticate the tag and to perform key update. Table 4 lists the terminology used in C-MRAP scheme.

Shared information between the database and the tag	
SN	The unique serial number of the tag.
PIN	The access password of the tag.
Kc$_i$	The key used to authenticate the tag in the ith round.
UN	A parameter used in the key update process
Shared information between the database and the reader	
RID	The unique serial number of the reader.
The extra information stored in the database	
Kc$_{i-1}$	The key used in the i-1 round.
Nr, *Nt*	Random numbers generated by the reader and the tag in the previous authentication message. They are used to foil replay attacks.
DATA$_x$	The detailed information of a tag$_x$

Table 3. The variables stored in the database, the reader, and the tag

r_1 r_2	The random numbers generated by the reader.
r_3	The random number generated by the database.
N	The random number generated by the tag.
Kc_i	The tag's authentication key used in the i^{th} round
Auth	The tag's authentication information, which is derived from the A8 algorithm. The database uses this variable to search for the tag in its memory.
M_1	The message generated by the tag.
M_2	The message generated by the database.
m1 || m2	The message combines m1 and m2.
$A_3(m_1,m_2)$	Encrypt m_1 and m_2 using the A3 algorithm
$A_8(m_1,m_2)$	Encrypt m_1 and m_2 using the A8 algorithm
f(.)	The pseudo random number generator
H()	A one-way hash function
$Cert_{RID}$	Reader's certificate

Table 4. C-MRAP terminology

3.1 The C-MRAP protocol

Figure 5 depicts the C-MRAP protocol, which includes three phases. The following section describes each message in detail.

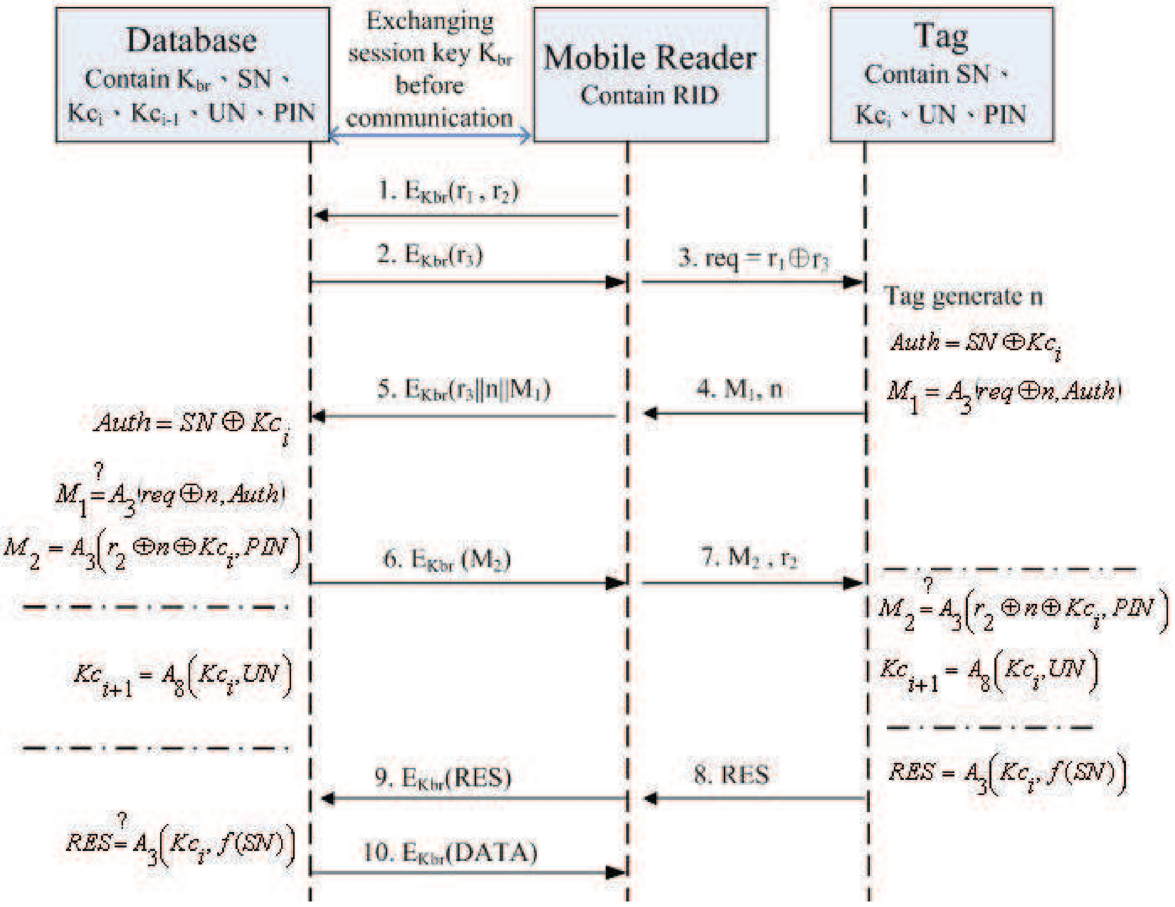

Database
Contain K_{br} 、SN 、Kc_i 、Kc_{i-1} 、UN 、PIN
Exchanging session key K_{br} before communication
Mobile Reader
Contain RID
Tag
Contain SN 、Kc_i 、UN 、PIN
1. $E_{Kbr}(r_1, r_2)$
2. $E_{Kbr}(r_3)$
3. $req = r_1 \oplus r_3$
Tag generate n
$Auth = SN \oplus Kc_i$
$M_1 = A_3(req \oplus n, Auth)$
4. M_1, n
5. $E_{Kbr}(r_3 \| n \| M_1)$
$Auth = SN \oplus Kc_i$
$M_1 \overset{?}{=} A_3(req \oplus n, Auth)$
$M_2 = A_3(r_2 \oplus n \oplus Kc_i, PIN)$
6. $E_{Kbr}(M_2)$
7. M_2, r_2
$M_2 \overset{?}{=} A_3(r_2 \oplus n \oplus Kc_i, PIN)$
$Kc_{i+1} = A_8(Kc_i, UN)$
$Kc_{i+1} = A_8(Kc_i, UN)$
$RES = A_3(Kc_i, f(SN))$
8. RES
9. $E_{Kbr}(RES)$
$RES \overset{?}{=} A_3(Kc_i, f(SN))$
10. $E_{Kbr}(DATA)$

Fig. 5. The C-MRAP protocol

Phase 1: Tag authentication

In the first phase, the reader and the database exchange their own certificates and establish a shared session key K_{br}. When the reader sends a read request to the tag, the tag sends the authenticated messages to the reader. The reader then forwards the message and two random numbers shared with the database to the database. The database searches its records to verify if the tag is valid.

1. When a reader sends a read request to a tag, the reader first generates two random numbers r_1 and $r_{2,}$, and encrypts them using the session key K_{br} shared by the reader and the database. The reader then sends the encrypted values to the database.
2. When the database receives the request, it generates another random number r_3, encrypts it with K_{br}, and sends the encrypted value back to the reader.
3. The reader calculates req=$r_1 \oplus r_3$, and sends req to the tag.
4. When the tag receives req, the tag generates a random number n, and calculates two variables: $Auth$ = $SN \oplus Kc_i$ and M_1=A$_3$($req \oplus n$, $Auth$) The tag then forwards M_1 and n to the reader.
5. The reader combines r_3, n, and M_1, encrypts it with the session key K_{br}, and sends the encrypted message to the database.
6. The database decrypts the message using the session key K_{br}. The database perform an exhaustive search of all the tags by calculating M_1'=A$_3$($r \oplus n$, $SN \oplus Kc_i$). If there is a matched M_1, the database uses the A3 algorithm to calculate M_2=A_3($r_2 \oplus Kc_i$, PIN), encrypts M_2 with K_{br}, and sends it to the reader.
   Furthermore, the database calculates Kc_{i+1}=A$_8$(Kc_i , UN), and backs up the Kc_i to Kc_{i-1}, and Kc_{i+1} to the Kc_i, as Figure 6 shows. If the database cannot find a matching tag, the database searches its records by calculating $Auth'$=A$_8$(SN , Kc_{i-1}) and $M_{1'}$=A$_3$($r \oplus n$, $Auth'$). If a match is found, the database and the tag are out of sync. At this time, the database checks req and n with the random variables Nr and Nt. If the variables are not the same, the database updates the tag's key because it is not a replayed message.

Phase 2: Synchronized Key Update

The reader decrypts M_2, and sends M_2 and r_2 to the tag.

The tag computes M_2'=A_3($r_2 \oplus Kc_i$, PIN), and compares it with M_2. If they are equal, the authentication process is complete. The tag generates its new key by calculating Kc_{i+1}= A$_8$(Kc_i , UN).

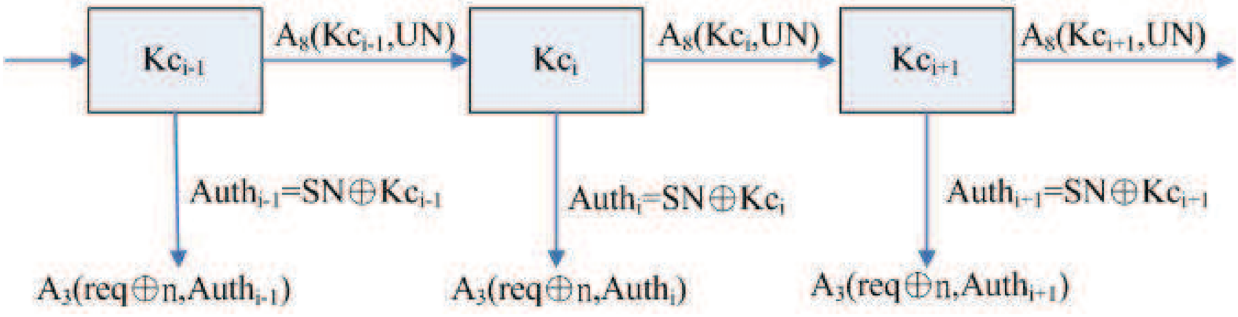

Kc_{i-1}
$A_8(Kc_{i-1},UN)$
Kc_i
$A_8(Kc_i,UN)$
Kc_{i+1}
$A_8(Kc_{i+1},UN)$
$Auth_{i-1}=SN \oplus Kc_{i-1}$
$Auth_i=SN \oplus Kc_i$
$Auth_{i-1}=SN \oplus Kc_{i-1}$
$A_3(req \oplus n,Auth_{i-1})$
$A_3(req \oplus n,Auth_i)$
$A_3(req \oplus n,Auth_{i+1})$

Fig. 6. Key Update

Phase 3: Key Update confirm

The tag calculates RES = $A_3(Kc_i , f(SN))$ and sends it back to the reader.

The reader encrypts RES using the session key K_{br}, and sends the encrypted message to the database.

The database compares *RES* with $A_3(Kc_i , f(SN))$. If the value is correct, the database sends the tag's information to the reader.

Using secrets shared between the database, the reader, and the tag, the proposed protocol updates the tag's authentication key during each session to protect the tag's privacy in a mobile RFID environment. Because Kc_i is not transmitted on the air, the protocol is secure and the reader can rapidly obtain a tag's information from the database.

4. Security analysis and performance evaluation

This section analyzes all the transmitted messages in the proposed protocol, and explains why this protocol is resistant to security attacks and can continue its operations without falling out of sync. Possible attacks include packet sniffing attacks, replay attacks, the man-in-the-middle attacks, and message dropping attacks. This section also compares the proposed protocol with other methods.

Section 4.2 implement three algorithms, including the Chameleon hash, COMP-128, and SHA1 algorithms, and evaluates the performance of Chan's, Lee's, and the proposed protocols. Results show that our protocol decreases the computation time of a tag and the database.

4.1 Security analysis

Message sniffing attacks

Assume a malicious attacker collects message 3 (*req*), message 4 (M_1 , n), message 7 (M_2 , r_2), or message 8 (*RES*) which are sent between the reader and the tag, and attempts to perform a guessing attack to retrieve tag information. The attacker cannot succeed in this attempt because he cannot guess the *Auth* value after obtaining *req* and n in message 3 and 4 because $Auth = SN \oplus Kc_i$, and the *SN* and Kc_i are only known by the database and the tag. The attacker must perform a brute force attack to guess these two values. Because Kc_i is updated in every session, it is hard to guess both *SN* and Kc_i at the same time. In addition, the attacker cannot retrieve messages transfered between the database and the reader because these messages are encrypted by the session key K_{br}.

Replay attacks

Using several random numbers, an attacker can attempt to replay message 3 (*req*) and message 7 (M_2, r_2). However, this is not possible because the messages are different in each session. For example, an attacker cannot replay message 5 ($E_{Kbr}(r_3 \mid\mid n \mid\mid M_1)$) because the database will verify it with the previous *req'* and n'.

Message dropping attacks

Next, consider the situation if the authentication message between the reader and the tag is lost during the transmission. In the proposed protocol, if message 3, message 4, or message 7 is lost, the reader waits for a timeout period and performs another reading request.

Man-in-the-middle attacks

If an attacker plays a role between the tag and the reader, and attempts to modify the value of *req* or n, the authentication process will fail because the *req* value is generated from the original reader, and not by an attacker.

If an attacker collects message 3 and message 4, and attempts to generate a new message 5 and send it to the database, this attack will fail because the attacker does not know r_1 and r_3.

If an attacker attempts to modify message 7 (M_2 , r_2), or message 8 (*RES*), the attack will fail because the attacker does not know the value of *PIN* and Kc_i.

Data privacy

The previous analysis indicates that an attacker cannot retrieve any valid information from data transmissions between the reader and the tag. In this protocol, only the variable *n* is not encrypted. Furthermore, all the transmission messages between the database and the reader are protected by the session key E_{kbr}.

Location privacy

An attacker can trace a tag's location if he can send the tag a specified value, and the tag returns a predicted value to the attacker. This type of attack will fail because the value of message 4 (*n* and M_1) changes every time. If an attacker wants to trace the tag by re-sending message 7, he will fail because the Kc_i is different.

Forward security

The proposed protocol satisfies the forward security because the COMP-128 algorithms it uses are one-way functions. Thus, an attacker cannot derive the previous messages using current messages.

Table 5 compares our protocol with other RFID security protocols.

	Anonymity	Location Privacy	Resisted to Replay attack	Resisted to man-in-the-middle attack	Forward security	Mobile RFID
Karthikeyan [12]	X	X	X	X	X	X
Duc [8]	O	O	X	X	X	X
Chien [13]	O	O	O	X	O	X
Henrici [9]	X	X	X	X	X	X
Yang [10]	X	X	O	O	X	X
Ohkubo [4]	O	O	X	X	O	X
Chan [3]	O	O	O	X	O	X
Lee [16]	O	O	O	X	O	X
Our scheme	O	O	O	O	O	O

Table 5. The Security Analysis

According to Table 5, Karthikeyan and Henrici's protocols cannot protect the location privacy, and are not resistant to replay attacks or middleman attacks. Chien's protocol cannot resist replay attacks because it uses the CRC function. Yang's protocol is resistant to replay attacks and man-in-the-middle attacks but it cannot protect the location privacy [15]. Chen and Lee's protocols are not suitable for mobile RFID environments because they trust all readers. Our protocol performs mutual authentication between the tag, the reader and the database, and is therefore suitable for use in mobile RFID environments.

4.2 Performance evaluation

In the Ohkubo protocol [4], the database must perform an exhaustive search to retrieve tag information. If there are *n* tags, the database complexity is *O(mn)* after *m* operations. The database complexity of Lee, Chan, and our protocol are the same, which is *O(n)*. However, Chan's scheme executes many Chameleon hash operations in the database and the tag, which decreasing overall performance.

Table 6 and Table 7 compare the operations of the three protocols. These tables assumes that the database is operated on a P4-2GHz personal computer, and the reader is a PDA equipped with a StrongARM SA-110 32-bit 233MHz CPU. We also assume that the CPU inside the tag is a 8-bit 12MHz processor. Table 6 lists all the operations required by the three protocols. In Chan and Lee's protocols, the reader only forwards messages between the tag and the database. We assume the packets transmitted between the reader and the database are encrypted by AES algorithms. Table 7 lists the average processing times for the tag and the reader to perform a single transaction in various approaches.

Table 8 lists the average execution times that it takes the database to search for a tag and perform key updates. Table 7 shows that the proposed protocol has better performance than Chan's scheme. Lee's scheme offers better performance than our protocol, but in Lee's scheme, the reader is trusted, rendering this scheme unsuitable for a mobile RFID environment. In the mobile RFID environment, the identity of a mobile reader must be authenticated to ensure protocol security. Finally, the proposed protocol provides better protection than Lee or Chan's protocols.

Protocols	Database operations	Reader operations	Tag operations
Chan	2n*CH_R search + 5*CH_R + 1*collision + 2*AES encrypt + 3*AES decrypt	2*AES encrypt + 2*AES decrypt	6*CH_R
Lee	2n*SHA1 search + 3*SHA1 + 1*AES encrypt + 1*AES decrypt	1*AES encrypt + 1*AES decrypt	7*SHA1
Our scheme	2n*COMP-128 search + 4*COMP-128+ 3*AES encrypt + 3*AES decrypt	3*AES encrypt + 3*AES decrypt	4*COMP-128

Table 6. The operations required by the three protocols

Protocol	Tag execution time	Reader execution time
Chan	0.0034	0.000945
Lee	0.0001	0.00047
Our scheme	0.0004	0.00141

Table 7. A comparison of execution times
(Unit: second)

Protocols	100 tags	1000 tags	10000 tags	100000 tags
Chan	0.0300	0.2368	2.1127	24.4873
Lee	0.0008	0.0064	0.0552	0.6451
Our scheme	0.0053	0.0420	0.3725	4.4791

Table 8. The average database execution times for various schemes
(Unit: second)

5. Conclusion

Mobile RFID technology offers more advantages than traditional RFID. However, because mobile RFID technology uses wireless communication, a secure authentication protocol is required to protect user privacy.

The proposed protocol is resistant to forge tag attacks, man-in-the-middleman attacks, packet sniffing attacks, replay attacks, packet dropping attacks, and out-of-sync attacks. In addition, the shared private key stored in the tag and the database is updated after each successful transaction. Compared to other protocols, this method provides a more secure protocol that enables all users to use a more secure Mobile RFID environment.

6. References

[1] MIT Auto-ID, retrieved Sep. 10, 2009 from World Wide Web http://autoidlabs.mit.edu.

[2] EPCGlobal, Class 1 Generation 2 UHF Air Interface Protocol Standard, retrieved Sep. 10, 2009 from World Wide Web http://www.epcglobalinc.org/standards/uhfc1g2.

[3] M. Chan, "Protect Mobile RFID Location Privacy Using Dynamic Identity," in *Proceedings of the National Computer Symposium*, Taiwan, 2007.

[4] M. Ohkubo, K. Suzuki, and S. Kinoshita, "Cryptographic approach to privacy-friendly tags," in *Proceedings of the RFID Privacy Workshop*, 2003, pp. 624-654.

[5] A. Juels, R. L. Rivest, and M. Szydlo, "The blocker tag: selective blocking of RFID tags for consumer privacy," in *Proceedings of the 10th ACM conference on Computer and communications security*, 2003, pp. 103-111.

[6] S. A. Weis, S. E. Sarma, R. L. Rivest, and D. W. Engels, "Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems," in *Proceedings of the Security in Pervasive Computing*, 2003, pp. 201–212.

[7] M. Lehtonen, T. Staake, F. Michahelles, and E. Fleisch, "From identification to authentication—a review of RFID product authentication techniques," in *Printed handout of Workshop on RFID Security – RFIDSec*, vol. 2006, 2006.

[8] D. N. Duc, J. Park, H. Lee, and K. Kim, "Enhancing Security of EPCglobal Gen-2 RFID Tag against Traceability and Cloning," in *Proceedings of the 2006 Symposium on Cryptography and Information Security*, 2006.

[9] D. Henrici and P. Muller, "Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers," in *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, 2004, pp. 149-153.

[10] J. Yang, J. Park, H. Lee, K. Ren, and K. Kim, "*Mutual authentication protocol for low-cost RFID,*" in *Proceedings of the Ecrypt Workshop on RFID and Lightweight Crypto*, 2005, pp. 17-24.

[11] I. J. Kim, E. Y. Choi, and D. H. Lee, "Secure Mobile RFID system against privacy and security problems," in *Proceedings of the Third International Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing (SECPerU 2007)*, 2007, pp. 67-72.

[12] S. Karthikeyan and M. Nesterenko, "RFID security without extensive cryptography," in *Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks*, 2005, pp. 63-67.

[13] H. Y. Chien and J.H. Chen, “A secure authentication mechanism suitable for EPC Class 1 Generation 2 RFID standard,” in *Proceedings of the 16th Information Security Conference (ISC2006)*, 2006, Taiwan, pp. 206-213.

[14] C.-L. Lin and K. C. Chang, “Security Analysis of the EPC Class 1 Generation 2 RFID authentication protocol,” in *Proceedings of the 17th Information Security Conference (ISC2007)*, 2007, Taiwan, pp. 600-606.

[15] G. Avoine, “Cryptography in Radio Frequency Identification and Fair Exchange Protocols,” *PhD Thesis of Swiss Federal Institute of Technology (EPFL)*, Lausanne Switzerland, 2005.

[16] L.-a. Lee and S. Shieh, “Protecting User Privacy with Dynamic Identity-Based Scheme for Low-cost Passive RFID Tags,” in *Proceedings of the CISC 2008*, Taiwan, 2008, pp. 206-218.

[17] S. Dominikus, E. Oswald, and M. Feldhofer, “Symmetric authentication for RFID systems in practice,” in *Proceedings of the ECRYPT Workshop on RFID and Lightweight Crypto*, Graz, Austria, July, 2005, pp. 14-15.

[18] N. Park, H. Kim, K. Chung, and S. Sohn, “Design of an Extended Architecture for Secure Low-Cost 900MHz UHF Mobile RFID Systems,” in *Proceedings of the IEEE 10th International Symposium on Consumer Electronics (ISCE’06)*, 2006, pp. 1-6.

[19] Nokia Co., “Nokia Unveils RFID Phone Reader,” *RFID Journal*, retrieved Sep.10, 2009, from World Wide Web http://www.rfidjournal.com/article/articleview/834/1/1/.

[20] K. Rhee, J. Kwak, S. Kim, and D. Won, “Challenge-Response Based RFID Authentication Protocol for Distributed Database Environment,” in *Proceedings of the Second International Conference on Security in Pervasive Computing (SPC 2005)*, Boppard, Germany, 2005, pp. 70-84.

[21] ETSI/GSM, Recommendation GSM 11.11, version 3.16.0, 1994.

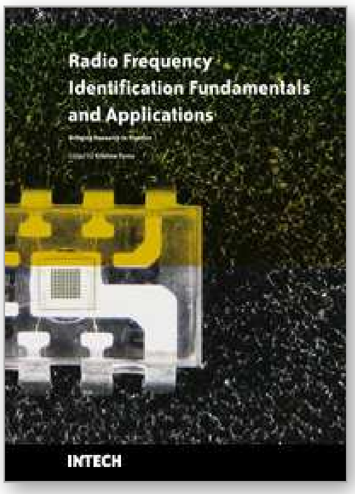
[22] ETSI/TC and SMG, Recommendation GSM 03.20. Security Related Network Function, Version 3.3.2, 1991.

[23] L. Gong, R. Needham, and R. Yahalom, “Reasoning about belief in cryptographic protocols,” in *Proceedings of the IEEE Symposium on Research in Security and Privacy*, 1990, pp. 234-248.

[24] R. Housley, W. Ford, W. Polk, and D. Solo, “Internet X. 509 Public Key Infrastructure Certificate and CRL Profile,” *IETF RFC 2459*, January 1999.

[25] H. Lee and J. Kim, “Privacy threats and issues in mobile RFID,” in *Proceedings of the First International Conference on Availability, Reliability and Security (ARES'06)*, 2006, pp. 510-514.

[26] H. Krawczyk and T. Rabin, “Chameleon signatures,” in *Proceedings of Network & Distributed System Security Conference (NDSS 2000)*, 2000, pp. 143-154.



Radio Frequency Identification Fundamentals and Applications Bringing Research to Practice
Edited by Cristina Turcu

ISBN 978-953-7619-73-2
Hard cover, 278 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

The number of different applications for RFID systems is increasing each year and various research directions have been developed to improve the performance of these systems. With this book InTech continues a series of publications dedicated to the latest research results in the RFID field, supporting the further development of RFID. One of the best ways of documenting within the domain of RFID technology is to analyze and learn from those who have trodden the RFID path. This book is a very rich collection of articles written by researchers, teachers, engineers, and professionals with a strong background in the RFID area.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821