# 國立交通大學

## 資訊工程學系

## 碩 士 論 文

以動態身份為基礎來保護低成本被動式無線射頻辨識標籤的使用者隱私

Protecting User Privacy with Dynamic Identity-Based Scheme

for Low-cost Passive RFID Tags

研 究 生：李禮安

指導教授：謝續平 博士

中華民國九十五年六月

以動態身份為基礎來保護低成本被動式無線射
頻辨識標籤的使用者隱私

Protecting User Privacy with Dynamic Identity-Based Scheme

for Low-cost Passive RFID Tags

研 究 生：李禮安　　　　　Student: Li-an Lee

指導教授：謝續平 博士　　　Advisor: Dr. Shiuh-Pyng Shieh

國 立 交 通 大 學
資 訊 工 程 學 系
碩 士 論 文

A Thesis
Submitted to
Department of Computer Science and Information Engineering
College of Computer Science
National Chiao Tung University
in Partial Fulfillment of the Requirements
for the Degree of
Master
in
Computer Science and Information Engineering

June 2005
Hsinchu, Taiwan, Republic of China

中 華 民 國 九 十 五 年 六 月

# 以動態身份為基礎來保護低成本被動式無線射頻辨識標籤的使用者隱私

研究生：李禮安　　　　　　　　指導教授：謝續平 博士

國立交通大學資訊工程學系

摘要

RFID 被稱為「下個世代的 barcode」，具備了不需要視覺接觸就能夠遠距離辨識身份的能力，將這項技術運用在各種應用上能帶來前所未有的便利。不過非視覺接觸的特性之下卻隱藏著 RFID 標籤上的資料隨著空氣中的電磁波在無形中被發送出來，透過收集這些資料，產品使用者的隱私便會暴露無疑。

使用者隱私主要分成資料隱私和位置隱私。欲同時保護這兩種隱私，RFID 標籤的輸出必須要加密且動態改變，否則攻擊者可以預測標籤輸出的話仍然能夠追蹤該該標籤。另外一個議題是低成本的 RFID 標籤成本應該在五分錢美金，在這樣的成本之下，標籤能夠運用在保護隱私及安全上的資源相當有限，一般對稱式和非對稱式的加密演算法都因為成本限制而無法運用在低成本的 RFID 上。

目前嘗試解決隱私問題的論文主要可以分成經由認證、加密和動態改變身份三種方式。有效的認證可以防止未授權的 RFID 讀取器取得 RFID 標籤上的資料。可惜目前低成本的 RFID 標籤很容易遭到物理性的解析，進而取得 RFID 標籤上的資料以及認證用的金鑰，因此認證方式很容易因為系統中的一個標籤被破解而喪失保護的功能。以加密為基礎的方式雖然可以保護資料的隱私，但是由於密文是固定的所以並不能夠保護使用者免於被追蹤。動態身份則會遇到搜尋 RFID 標籤身份的效率不足以及更新週期中仍然會被追蹤的問題。

　　在這篇論文之中，我們提出了一個可實行於低成本被動式標籤的機制。每個標籤都有一個動態改變的身份，能夠輸出動態的資料。另外我們也證明這個機制可以抵擋重送攻擊、竊聽、偽造、和封包遺失等攻擊。

# Protecting User Privacy with Dynamic Identity-Based Scheme

# for Low-cost Passive RFID Tags

Student: Li-an Lee                    Advisor: Shiuh-Pyng Shieh

Department of Computer Science and Information Engineering

National Chaio Tung University

**Abstract**

Radio Frequency Identification (RFID) is said to be the next generation bar code, which features contactless identification without visibility. We benefit greatly by adopting RFID in a variety of daily applications such as warehouse management, toll collection, library management, etc. However, RFID transmits data through radio frequency signals; therefore, attackers could analyze the radio frequency signals to acquire private data from users. If user privacy is not protected, users will be susceptible to personal identification and tracking by an adversary.

User privacy may include data privacy and location privacy. To protect both of them, the output of tags must be encrypted and unpredictable. Furthermore, the acceptable cost of a passive RFID tag, which is no more than five cents, severely restricts the resources available for security.

Schemes that protect user privacy in RFID applications are classified into three main categories: authentication, encryption, and dynamic identity. However, authentication-based schemes are easily broken. Because low-cost RFID tags do not contain tamper-resistant mechanisms, an adversary can steal the key for the authentication protocol. Encryption-based schemes can protect data privacy, but location privacy is still vulnerable since the ciphertext remains the same. Dynamic identity schemes are limited by exhaustive search problem, and the tag is still traceable in the period between identity updates.

In this thesis, we proposed a feasible scheme based on one-way hash function for low-cost passive RFID tags. Each tag has a dynamic identity. Therefore the output of tag changes each time. We also proved that the scheme can protect both data privacy and location privacy against threats of replay attacks, eavesdropping, spoofing, man-in-the-middle attack, and message loss.

誌謝

　　這份論文得以完成，需要感謝很多人的幫忙。首先感謝指導老師謝續平教授給予的指導，讓我在做研究的過程中能夠有明確的方向和目標。另外感謝楊明豪學長、黃士一學長和楊子逸學長每星期個別討論時間給予的經驗、建議以及鼓勵。還有葉羅堯學長給予認證機制上的建議。

　　另外感謝大的、野狗、Jelly 和 Warren，在作研究的路上互相鼓勵，這些熬夜吃宵夜甚至見日出的日子裡一定是將來難忘的回憶。碩一的學弟妹也幫忙分擔了實驗室的重擔讓我們得以專心在研究上。之涯幫忙系計中網頁上的事務、阿波對於實驗室主機的維護、仁倩幫忙管理meeting 相關事項、阿挫安排實驗室出遊、浩洋和朝彥負責採買實驗室零食。

　　最後感謝臭北鼻子欣一直以來的陪伴和支持。感謝在作研究的路上因為有你們，讓我得以堅持到最後。

# Table of contents

# List of Figures

# 1. Introduction

Radio Frequency Identification (RFID) is a wireless identification technology. Compared with optical bar codes, it has many characteristics such as contactless identification, mass identification, identification at longer distance, larger data size, changeable data content, and being difficult to counterfeit. Due to those features optical car codes can't provide, RFID becomes a good approach for automated identification of products and offers powerful benefits for businesses and consumers.

## 1.1. RFID System

The basic components of RFID system are tags, readers, and the back-end database. Tags are affixed to items need to be identified. Readers search tags in its transition range, read information on tags and forward it back to the back-end database.

### 1.1.1. RFID Tags

A tag is composed of an integrated chip and antenna, it stores the information associated with the tagged item such as manufacture, product type, and product identity on the integrated chip. A tag responses its information while queried by RFID readers.

Tags can be classified into *active, passive, and semi-passive*. An active tag has its own battery to provide the power, usually has stronger computation power and longer transmission range, but the life of tag is limited by the battery and the price of cost is

high. A passive tag doesn't have its own battery. It is powered by the radio signal of reader. Compared with active tags, it has limited computation power and shorter transmission range, but their life is not limited and price of cost is low. A semi-passive uses a battery for IC chip computation, and reader's radio signal for communication.

Most applications use passive tags for cost issues. The acceptable cost of passive tag should be no more than 5 cents. With such cost, the number of gates for security is limited from 2,500 to 5,000 gates [1], most cryptographies can not be implement. For example: Data Encryption Standard (DES) requires 10,000 gates and Advanced Encryption Standard (AES) implementation requires 5,000 gates [19].

### 1.1.2. Readers

A reader interrogates RFID tags nearby, reads the information on them, and forwards the information to the back-end database for further applications. It might also update the information on tags.

### 1.1.3. The Back-end Database

A database stores associated information of tags, like object name, price, location, manufacturer, and owner…etc. According to applications, the database changes the record of the tag. For instance, the database for library management changes the records when users check out books.

## 1.2. RFID Applications

Adopting RFID in variety of applications offers convenient services. For

example, RFID tags can be attached to merchandises in a supermarket. To check whether the merchandises are expired, missed, or misshelved can be done by 'smart shelves' equipped with RFID readers [10]. The branded products can prevent counterfeit by embedding tags inside, and Prada put RFID tags on its shoes, handbag, and dress for better costumer service, by reading the RFID on the clothes, the screen shows more information related to the clothes [20]. Automatic toll collection can be done by using tags linked to debit accounts, an RFID credit card is affixed to a car's license plate or windshield, the RFID card sends the account information to the toll collection which equipped with RFID readers [21]. Using RFID can make the library management more efficient, speed up the procedure of check-in, check-out, and use air scan for missing books [16].
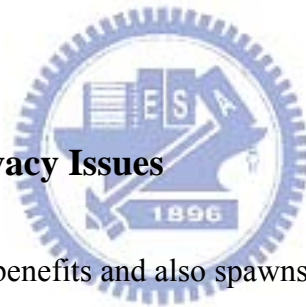
## 1.3.  RFID Personal Privacy Issues

RFID provides us many benefits and also spawns many issues regarding privacy. Since RFID signal is transited over air, attackers can always sniff the messages between readers and tags to get private information on tags. Privacy issues are mainly classified into *data privacy* and *location privacy*.

### 1.3.1.  Data Privacy

Adversaries can use reader to scan individuals for RFID tagged items without their knowledge remotely. If adversaries can associate the output of the tag with the item the tag affixed to, then adversaries can get the shopping list and even the preference list of individuals without their consent.

Worse, if adversaries associate the output of the tag with the individual who

carries it, personal identification becomes another issue.

And adversaries might scan individuals to know private item they carry. Theft might also use reader to choose a rich victim by scanning individual who carries high-value items.

### 1.3.2. Location Privacy

People who carry RFID tags and vehicles with RFID are under the threat of tracking. If adversaries can predict the output of some tag or a tag always output the same message, adversaries can associate the tag and its owner. By scanning tagged items, adversaries can track the individual by RFID. Since people can't sense the radio frequency signal and RFID provides contactless identification, individuals can hardly find he or she is being tracked remotely by someone.

In this paper we proposed a feasible scheme for low-cost RFID to solve both data privacy and location privacy problems. We give the previous work on RFID privacy issues in section 2. The proposed scheme is introduced in section 3. Section 4 is the security analysis about our work and section 5 is the conclusion.

## 2. Related Work

There are many researches proposed different schemes to protect user privacy for RFID. This section introduces those previous works on RFID privacy issues. We also discuss the advantages and disadvantages of those previous works.

To evaluate those previous work, there are some security requirements should be sufficed. Such as encrypted information for data confidentiality, dynamic output of tag against tracking, forward security, and mechanisms to distinguish spoofed and replay messages. Furthermore, because low-cost RFID tags are not tamper-resistant, the scheme should be secure even if attackers compromise some tags in the system. And the other legitimate tags should still work under the protection of all security requirements talked previously.

The previous work on RFID privacy issues can be classified into hash-based authentication schemes, key-based authentication schemes, encryption-based schemes, dynamic identity schemes, and deactivation approaches.

### 2.1. Hash-Based Authentication Schemes

In these schemes, the authors assume that the channel from reader to tag is easily eavesdropped, and the channel from tag to reader is hard eavesdropped due to the different power of radio frequency signal. However, attackers within the transmission range of tag still can sniff the message between tags and readers.

### 2.1.1. Hash Lock Scheme

The authors proposed an access control mechanism for RFID [12]. Each tag stores a hash value of a random key called metaID. The back-end database stores random keys and corresponding metaID of each tag in this system. A tag responses its metaID as a challenge, and the reader asks the back-end database for the appropriate key for response, the tag then make the hash value of this key and compares it to its metaID, if the values are the same, the tag thinks the reader is legitimate and replies its identity to the reader.

This scheme is feasible for low-cost RFID tag, since only one hash function needed. However, it can not resist replay attacks if attackers sniff the message between tags and readers. Furthermore, due to the fixed metaID, the location privacy is not protected. Attackers can easily recognize the output from the same tag and track the tag. And there is no mechanism to protect the RFID system against spoofing.
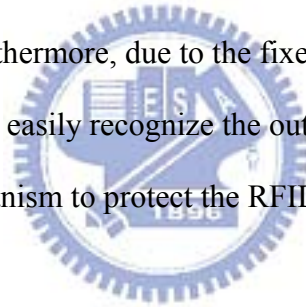
### 2.1.2. Randomized Hash Lock Scheme

When a tag is queried by a reader, the tag responses a random number and the hashed value of its identity concatenated with the random number [12]. After the reader forwards this message to the backend database, the database makes the hash value of identity of each tag concatenated with the random number to find the appropriate identity of the tag.

This scheme improved the tracking problem of hash lock scheme, but due to the exhaustive search in the backend database, the scalability of this scheme is limited. And there is no mechanism to distinguish spoofing.

## 2.2. Key-Based Authentication Schemes

These schemes assume all tags in same system share a symmetric secrete key with the back-end database [4] [9]. Before reading, tags make challenge messages to the reader. Only those who have the shared key can make a valid response message and pass the authentication. However, Low-cost RFID tags are not tamper-resistant, attackers can get the shared key on the tag by physical analysis. Beside, there is no key management mechanism for RFID system. If the shared key is compromised by attackers, change the shared key for tags is not practical. Thus attackers can break the whole system by compromising only one tag.

## 2.3. Encryption-Based Schemes

Encrypting the identity of tag by symmetric or asymmetric cryptography protects the data confidentiality. However, the low-cost tag can not afford encryption algorithms such as DES, AES, and RSA. Those schemes use external devices to compute ciphertexts and write back to tags.

### 2.3.1. Anonymous ID Scheme

To protect the data privacy, the tags of this scheme store encrypted identity instead of real one [13] [14]. The tags response its encrypted identity to the reader while being read. To get the real identity of a tag, the back-end database decrypts the encrypted identity.

The encrypted identity can protect the data privacy. But the encrypted identity is fixed, adversaries can still track the individual, and the location privacy cannot be

protected.

### 2.3.2. External Re-encryption Scheme

In this scheme [17], the tag stores the identity encrypted by public key cryptography. After each read operation, the reader re-encrypts the real identity of the tag into different ciphertext and writes it back to the tag. This scheme protects data privacy and makes stronger protection of location privacy. But the encrypted identity is still fixed. Before the identity of the tag is re-encrypted, adversaries can track the individual.

## 2.4. Dynamic Identity Schemes

It is very hard to protect the location privacy if a tag always sends fixed output to readers. Some people proposed schemes to make the output of tags dynamic. Hash functions are practical for low-cost RFID tags. They require fewer gates to implement than general cryptographies. There are two kinds of hash based schemes. One requires the information on tags and on the database to be synchronous, while the other does not.
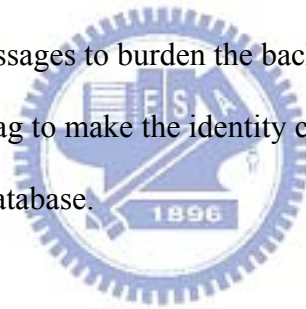
### 2.4.1. Asynchronous Identity Scheme

In this scheme, two different hash functions G and H are used [1]. Every time being queried by a reader, the tag puts its original identity as the input of the hash function G to generate its new identity, and then puts its new identity as the input of the hash function H to generate the output which sent to the reader. And finally stores

the new identity as the original identity.

To recognize the identity of the tag, the database stores the initial identity of each tag in the system. After the reader sends an output to the database, the database hashes every initial identity with hash function G and H iteratively to find the initial identity of the tag.

The scheme resolves the tracking problem and also protects the data privacy. However, to recognize the identity of tags, the database has to perform an exhaustive search, thus the scope of this scheme is limited. To enlarge the scope of this scheme, some people used time-memory trade-off to reduce the search complexity on the database side [3]. But when the scope keeps growing, the benefit from time-memory trade-off is still limited. Moreover, there is no mechanism to verify spoofing message, attackers could sends fake messages to burden the back-end database, and attackers could keep reading the same tag to make the identity change, which makes the tag hard to be recognized in the database.

### 2.4.2. Synchronous Identity Schemes

To void the search overhead on the database side, these schemes require the identity of tags and the database to be synchronous [6] [8] [9]. The tag of this scheme sends the hash value of its real identity to the reader instead of its real identity. After each reading operation, the database sends an update message to the tag. The tag updates its identity according to the message.

The hash value protects the database confidentiality. However, if the update message is missed, the identity of the tag will not be updated, which might result in the asynchronous status between the tag and the database. And before next identity update, fixed hash value of identity fail to protect the tag against tracking.

## 2.5. Other Approaches

There are other approaches to resolve privacy issues. We show them here

### 2.5.1. Kill Command Feature

Propose by EPCglobal and has been ratified [2]. The tag has a kill password to make itself permanent disable, and afterward the tag won't response to any query. Although the privacy problem can be resolved completely by this way, but since users cannot benefit from RFID after killing the tag, this method is not suggested.

### 2.5.2. Blocker Tags Scheme

This scheme doesn't enhance the communication process between tags and readers [11]. The main idea is to interfere with the communication if protected tags are being read. A blocker tag is such a device to interfere with the tree-walking protocol.

The serial numbers of protected tags are given to the blocker tag in advanced. When the reader makes an interrogation to singular the protected tags via tree-walking protocol, the blocker tag also responses to interfere with the output of protected tags. In this way, the reader cannot read the protected tags.

However, this scheme is limited, because blocker tags cannot protect protected tags while out of the transmission rang.

# 3. Proposed Scheme

To protect the data confidentiality for low-cost RFID tags, cryptography such as DES, AES, and RSA is not feasible due to the cost issue. The computation power needed is beyond low-cost tags. In stead of general cryptography, exclusive-OR operation and hash functions are much more practical for low-cost RFID tags.

If the output of tags contains fixed segments or is predictable, attackers can easily track tags. To protect the location privacy, dynamic identity should make the output of tag dynamic and unpredictable. There are two update strategies, one has to keep the identity of tag and database record to be synchronous, and the other does not.

The advantage of synchronization approaches is to reduce the search overhead of the database. But the drawback is, before next identity update, the fixed identity might be the vulnerability to be tracked. On the other hand, synchronization approaches provide the stronger protection against location privacy. However, it also burdens the database due exhaustive search.

Here we introduce a dynamic identity-based scheme. This synchronous approach protects data privacy and location privacy for low-cost RFID tags under the threats of replay attacks, spoofing, eavesdropping, and message loss. First we describe the task of the back-end database and readers, then the computation power requirement of tag, final the initial setup and communication protocol.

## 3.1. The Back-end Database

The back-end database is in charge of some access control mechanism to authenticate legitimate readers. Only legitimate readers can communicate with the

back-end database and establish secure channels. The database also stores the following detailed information for each tag in the system:

- **Current ID (CID)**: the current identity of a tag, it changes after each legitimate reading operation.

- **Hashed ID (HID)**: the hash value of the CID of a tag, it also has to be recalculated after each identity update. It plays as a primary index in the database.

- **Serial Number (SN)**: a unique serial number assigned to the tag. This number is fixed, if two tags have the same dynamic identity, database can distinguish them by this value.

- **Transaction ID (TID)**: a number associated with a reading operation, it accumulates by one while receiving the reading request of readers.

- **Last Transaction ID (LST)**: the TID of the last legitimate reading operation.

- **Reference (REF)**: an entry point to another data row in the database. Initially there is only one data row for a new tag, thus the initial REF is set to N/A. In our approaches, we keep the recent two dynamic identity information of a tag. After the tag is read, a new row is created to record the new identity information. The two rows of the same tag are set to point to each other, and afterward, the new identity information is updated by turns.

## 3.2. The Readers

The task of reader is to play a forwarder between the back-end database and RFID tags. In our scheme the reader cannot recognize the identity of the tag. To recognize the identity, the back-end database is in charge of decrypting the output of the tag and returning the identity to the reader if necessary.

Further, there is some access control mechanism between the back-end database and readers. We assumed only legitimate readers can communicate with the back-end database via secure channels.

## 3.3. Computation Power Requirement of Tags

In our approach, the tag has a random number generator, and the computation power to compute exclusive-OR operation and a hash function H. Hash functions are thought lightweight and practical for RFID tags [1] [3] [6] [7] [8] [9] [12]. The storages of the tag are its CID, SN, TID and LST. We use the following notations in our protocol:

- **H(m)**: the hash value of message m using hash function H

- **$m_1 \oplus m_2$:** the exclusive-OR value of message $m_1$ and $m_2$.
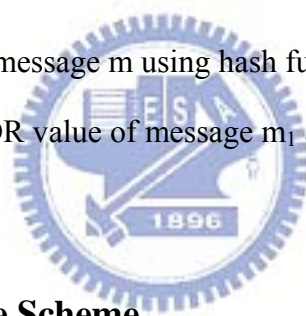
## 3.4. Tag Identity Update Scheme

The identity update scheme is based on a one-way hash function H. After each legitimate reading operation, the tag and the back-end database compute the hash value of the exclusive-OR value of the CID with a random number R, and take the hash value as the new CID. **Figure 1** shows the identity update scheme. The hash identity $HID_i = H(CID_i)$ and new ideneity $CID_{i+1} = H(CID_i \oplus R_i)$, where $R_i$ is a random number generated by the back-end database.
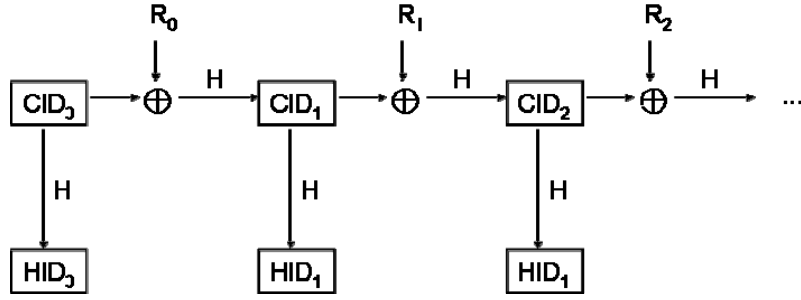
**Figure 1: The identity update mechanism.**

Adopting dynamic identity makes RFID tags hard to track. However, if the reading operation is not complete, the identity of a tag and the database might become inconsistent, which results in the identity loss in the database. To make our scheme resist against asynchronous status between tags and the database, we keep the recent two reading records in the database. We illustrate how it works against asynchronous condition in section 4.

## 3.5. Initial Setup

While issuing a new tag, the tag is assigned a random CID and a unique SN. The TID and LST are set to the same random value. In the database, a row is created for the tag. The CID, TID, and LST are set to the same value with the tag, and the HID is pre-computed. The REF is not set because there's no recent record initially.

## 3.6. Communication Protocol

This protocol contains three messages, the first message is a reading request made by the reader. The second message contains the identity and authentication information of the tag. The third message contains the update parameter and

authentication information of the back-end database. **Figure 2** shows the

communication process, where N is a random nonce generated by the tag.



| Database | | | | | | | Reader | | RFID tag | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| HID | CID | SN | TID | LST | RFF | | | | CID | SN | TID | LST |

**Request** →

1. Generate random nonce N
2. Compute TID = TID+1
3. Compute △TID = TID-LST
4. Compute HID = H(CID)

← N, H(SN⊕HID⊕N), △TID⊕H(SN⊕N), H(CID⊕TID)

1. Search SN in the database
2. Restore the tag's TID by computing LST+△TID
3. Verify H(CID⊕TID)
4. Generate random number R to update the tag's CID
5. Update the database record

R⊕H(SN⊕(N+1)), H(R⊕CID⊕TID) →

1. Get the random number R
2. Verify H(R⊕CID⊕TID)
3. Update CID = H(R⊕CID)
4. Set LST = TID

**Figure 2: The proposed protocol.**

Before the communication process, the reader might need to perform singulation

protocol before sending a reading request to a tag. The tag does the following tasks

after receiving a request message:

1. Generate a random number for nonce N.

2. Increase TID by one.

3. Compute $\Delta$TID by TID-LST.

4. Compute HID by H(CID).

Then it sends N, H(SN$\oplus$HID$\oplus$N), $\Delta$TID$\oplus$H(SN$\oplus$N), and H(CID$\oplus$TID) as

the reply message to the reader. Note that there are four segments in this message. The

first segment is the nonce of the session, the nonce participates in hash values in this

message and make the message looks dynamic. The second segment is hashed

identity information. By the nonce and the database information, the back-end

15

database can know the identity of the tag. The third segment is encrypted ΔTID. The back-end database can restore the TID of the tag by ΔTID+LST. And the last segment is authentication information of the tag. The back-end database can verify it to check if the tag is a counterfeiting.

The back-end database does the following tasks after receiving the message of the tag:

1. According to N, compute the hash value $H(SN \oplus HID \oplus N)$ of tags in the database to find the database record of the tag.

2. After knowing the identity of the tag, compute $H(SN \oplus N)$ to decrypt the encrypted ΔTID. Then restore the TID of the tag by computing LST+ΔTID. And check the message is fresh. Normally the TID should be greater than the database, otherwise it is a replay.

3. Verify $H(CID \oplus TID)$ is valid. Correct $H(CID \oplus TID)$ means the message comes from a legitimate tag, since only the tag knows its own CID and TID.

4. Generate a random number R to update the CID of the tag by $H(CID \oplus R)$.

5. Update the TID of the original data row found in step 1 to the tag's TID. If the REF is N/A, create a new row to store new status of the tag or use the row pointed by REF otherwise. The new row stores the new status like HID, CID, TID and LST of the tag. The two rows are set to point to each other.

After updating the database, the back-end database sends $R \oplus H(SN \oplus (N+1))$, $H(R \oplus CID \oplus TID)$ to the tag. The first segment of the message is an encrypted update parameter. The tag uses the parameter to update the identity. The second segment is authentication information. Since only the back-end database knows the TID and CID of the tag, the tag can verify whether the message is valid from the back-end database.

The tag does these tasks after seeing the third message:

1. Verify the H(R⊕CID⊕TID) by recalculating it. If they are not the same means the message does not come from the back-end database, the tag drops it and does nothing.

2. Decrypt R⊕H(SN⊕(N+1)) to get R, and update its CID to the new identity H(R⊕CID), finally set LST to TID.

Now the communication process finishes. The status of the tag and the database is consistent. If the tag did not update its identity successfully, the tag will send the elder identity next time. Note that the database keeps recent two data rows for the tag, one records the older identity, and another records the new identity. The database can still find the identity by keeping the older identity.
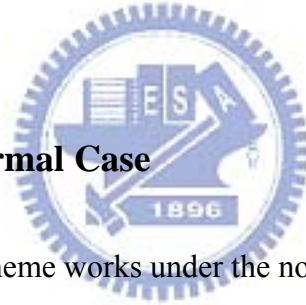
## 3.7. Example under Normal Case

We illustrate how our scheme works under the normal operation. Assume the CID of a new tag is 11, HID is H(11), TID and LST are 51. A row is created for the tag in the database. **Figure 3** shows the initial status of the tag and database. Note that the REF is N/A initially.

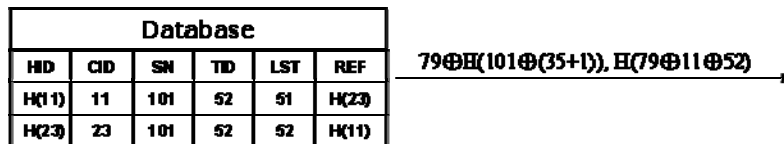| Database | | | | | | | RFID tag | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| HID | CID | SN | TID | LST | REF | | CID | SN | TID | LST |
| H(11) | 11 | 101 | 51 | 51 | N/A | | 11 | 101 | 51 | 51 |

**Figure 3: The initial status of the tag and the database.**

As the reading session starts, the tag picks a random number for nonce N (i.e. 35), increases its TID by one (i.e. 52= 51+1), computes ΔTID by TID – LST (i.e. 1=52-51) and HID H(11), sends N, H(SN⊕HID⊕N),ΔTID⊕H(SN⊕N), and H(CID⊕TID) to

the reader. **Figure 4** shows the message and the status of the tag afterward. By this time, the TID of the tag is greater than the database.



35, H(101⊕H(11)⊕35),
1⊕H(101⊕35), H(11⊕52)

| RFID tag | | | |
|---|---|---|---|
| CID | SN | TID | LST |
| 11 | 101 | 52 | 51 |
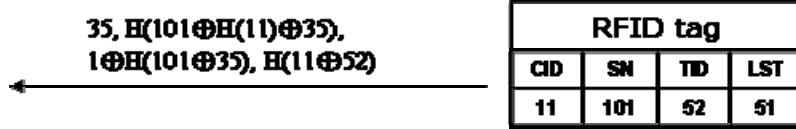
**Figure 4: The message from the tag and the tag status.**

Seeing the message from the tag, the back-end database finds the identity by computing H(SN⊕HID⊕N) of each data row, then gets the encrypted ΔTID from the third segment of the message and restores the TID of the tag by LST+ΔTID (i.e. 52=51+1). The TID of the tag is greater than the database (i.e. 52 > 51), means the message is still fresh, not a replay. The back-end database then checks H(CID⊕TID) is valid and generates a random number R to update the identity of the tag (i.e. R=79, 23=H(11⊕79). Because the REF of the tag is N/A, a new row is created for the tag to record new identity. The older identity is kept, too. The two rows are set to point to each other. Note that the TID of the older row is updated to check replay attacks (i.e. TID is set to 52). Then database sends a update message R⊕H(SN⊕(N+1)), H(R⊕CID⊕TID) to the tag. **Figure 5** shows the message from the database and the database status afterward.



| Database | | | | | |
|---|---|---|---|---|---|
| HID | CID | SN | TID | LST | REF |
| H(11) | 11 | 101 | 52 | 51 | H(23) |
| H(23) | 23 | 101 | 52 | 52 | H(11) |

79⊕H(101⊕(35+1)), H(79⊕11⊕52)

**Figure 5: The message from the database and the database status.**

After receiving the update message, the tag decrypts R and verifies H(R⊕CID⊕TID) is valid, then updates its new identity to H(R⊕CID). **Figure 6**

shows the final status in the end of the reading process. Now the status of the tag and the database are consistent.

| Database | | | | | | | RFID tag | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| HID | CID | SN | TID | LST | REF | | CID | SN | TID | LST |
| H(11) | 11 | 101 | 52 | 51 | H(23) | | 23 | 101 | 52 | 52 |
| H(23) | 23 | 101 | 52 | 52 | H(11) | | | | | |

**Figure 6: The ended status of the tag and the database.**

Note that next time the same tag is read, the new status will be recorded in the older row (i.e. the row which HID is H(11)). We keep the recent two records of each tag to void identity loss.

# 4. Security Analysis

We show that our scheme can protect data privacy and location privacy under the threats of eavesdropping, spoofing, man-in-the-middle attack, message loss, and replay attacks. We use a one-way hash function to protect the messages. And we define the one-way hash function as following.

**Definition 1:** A function H that maps an arbitrary length message m to a fixed length digest h is a one-way hash function if it provides such properties:

- Preimage resistant: given h, it is computationally infeasible to find some input m such that $h = H(m)$.
- Second preimage resistant: given an input $m_1$, it is computationally infeasible to find another input $m_2 \neq m_1$ such that $H(m_1) = H(m_2)$.
- Collision-resistant: it should be hard to find two different message $m_1$ and $m_2$ such that $H(m_1) = H(m_2)$.

## 4.1. Data Privacy

There are three messages in a reading session. The first message is the reading request from readers. It has no private information related to tags. The second message contains private information such as the identity, and corresponding authentication data. We use a one-way hash function to encrypt private information into message digest, only the nonce N is transmitted in plain text format. Since the one-way hash function is computationally infeasible to invert. Attackers can not decrypt the message to get privacy by the brute force way.

The third message contains identity update parameter and the authentication data of the back-end database. The message is encrypted by the one-way hash function, too. By the preimage resistance, attackers can not invert the message to know any information.

**Definition 2:** We define the data privacy in this scheme. Given any tag $T_i$, any adversary $A_d$ can adaptively query $T_i$ ,gets the output $M_1$, $M_2$, $M_3$, … from $T_i$ and the corresponding update message $U_j$ from the back-end database, where j = 1, 2, 3... $A_d$ can break the data privacy if $A_d$ can know one of the following:

● HID, CID, SN, and TID of $T_i$ in some message $M_j$, says $HID_j$, $CID_j$, $SN_j$, $TID_j$, where j = 1, 2, 3…

● R in the corresponding update message $U_j$, says $R_j$, where j = 1, 2, 3…

***Corollary 1:*** *By the preimage resistant in Definition 1, except $N_j$, $A_d$ cannot know $HID_j$, $CID_j$, $SN_j$, $TID_j$ in $M_j$, where j = 1, 2, 3...*

***Lemma 1:*** *$A_d$ cannot know $R_j$ in $U_j$, where j = 1, 2, 3...*

***Proof:***

To knows $R_j$ in $U_j$, $A_d$ has to know $H(SN_j \oplus (N_j+1))$. However, $A_d$ only has knowledge of $N_j$. By the preimage resistant in Definition 1, $SN_j$ is infeasible to compute. Therefore, $A_d$ cannot know $R_j$.

## 4.2. Location Privacy

Attackers can track tags if the tags' output contains fixed segments or can be predictable. In our scheme, tags change identity after each legitimate reading session.

Therefore, the output of the same tag changes every time. However, the update message from the back-end database might get lost due to the unreliable channel between reader and tags. If the tag did not update new identity successively, it uses the same HID and CID for next reading session, which might be vulnerable against tracking.

To fix this problem, we should let the tag sends the message contains the same HID and CID in a dynamic form. In the message from tag, we do not send HID or CID alone. In stead, we make exclusive-OR value of HID and CID with dynamic N and TID, and hash the value to looks more dynamic. But by the accumulated TID and randomly picked nonce N, output of the same tag changes dynamically even if the identity did not update successively.

**Definition 3:** Given an arbitrary set of tags $T_1$, $T_2$, …, $T_n$, any adversary $A_d$ can adaptively access the set of tags as many times as they want, each time $A_d$ accesses the tags, $A_d$ gets an message $M_{i,k}$ from a random picked tag $T_i$ of the set, where $1 \leq i \leq n$, k= 1, 2, 3…($A_d$ does not know what value i and k are). $A_d$ can track tags if $A_d$ can distinguish whether there are outputs $M_{j,p}$ and $M_{j,q}$ from the same tag $T_j$, where $p \neq q$, $1 \leq j \leq n$ (Tags will not renew their identity since the reading session is not complete).

*Lemma 2: $A_d$ cannot track tags*

*Proof:*

To distinguish the outputs from the same tag $T_j$, the adversary $A_d$ must know some relationship between message $M_{j,p}$ and $M_{j,q}$. Since $T_j$ does not renew its identity, $A_d$ could use SN, HID, and CID of $T_j$, says $SN_j$, $HID_j$, and $CID_j$ to distinguish if $M_{j,p}$ and $M_{j,q}$ comes from the same tag $T_j$, $A_d$ may verify the equations for both case 1 and case 2:

**Case 1:** check $H(SN \oplus HID \oplus N)$

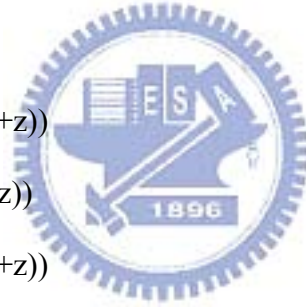$$H(SN_j \oplus HID_j \oplus N_{j,p}) = H(SN_{j,p} \oplus HID_{j,p} \oplus N_{j,p})$$

and

$$H(SN_j \oplus HID_j \oplus N_{j,q}) = H(SN_{j,q} \oplus HID_{j,q} \oplus N_{j,q})$$

for some $SN_j$, $HID_j$, where $1 \leq j \leq n$, and $p \neq q$

However, to know $SN_j$ and $HID_j$ of $T_j$ from $M_{j,k}$, where $k = 1, 2, 3 \ldots$ is computationally infeasible due to the preimage resistant in Definition 1.

**Case 2:** check $H(CID \oplus TID)$

$$H(CID_{j,p} \oplus TID_{j,p})$$
$$= H(CID_{j,p} \oplus (TID_{j,q}+z))$$
$$= H(CID_j \oplus (TID_{j,q}+z))$$
$$= H(CID_{j,q} \oplus (TID_{j,q}+z))$$

that is $TID_{j,p} = TID_{j,q} + z$, for some $z \in \mathbf{Z}$

However, to know $TID_{j,p}$ and $TID_{j,q}$ is computationally infeasible due to preimage resistant in Definition 1. Therefore, $A_d$ can not track tags, and our scheme protects the RFID location privacy.

Besides dynamic information, we show that the output of tag is not predictable. Attackers can not find relationship between successive $HID_k$ and $HID_{k+1}$ of the same tag since the one-way hash function is irreversible. It is infeasible to compute the $CID_k$ where $HID_k = H(CID_k)$ and $HID_{k+1} = H(H(CID_k \oplus R_k))$. The proposed identity

update scheme also provides forward security. If an attack compromises a tag whose CID is $CID_t$ at time t, the attacker can not find the past messages sent with $CID_{t'}$ by the same tag where t'<t even if the attacker records all message before time t.

Since HID, CID, and the one-way hash function is irreversible. Attackers can not trace the past event related to the tag.

***Corollary 2:*** *Our identity update scheme provides forward security due to the preimage resistant of one-way hash function.*

## 4.3. Eavesdropping

Since radio frequency signal is transmitted over air, attackers can always eavesdrop to know messages between tags and readers. We have showed that any active attacker cannot violate the data privacy and the location privacy. It is trivial that passive attackers such as eavesdropper cannot, either.

## 4.4. Spoofing

If there is no mechanism to check whether the messages from tags are valid, the attacker could pretend to be a valid tag and spoof the back-end database. In the proposed scheme, the message from a tag contains the authentication data of the tag. Without knowing the information such as CID, SN, TID, and LST on the tag, the attacker can not make authentication of the tag. And since that information on tag never is transmitted in plan text form, the attacker can not get information on the tag.

**Definition 4:** Adversaries can spoof the back-end database if they can win the

following game. Any adversary $A_d$ can adaptively access some tag $T_i$. It means $A_d$ can get the outputs $M_1$, $M_2$, ..., $M_n$ from $T_i$. $A_d$ wins if $A_d$ can make a valid output $M_a$ from $T_i$ where $M_a \neq M_1$, $M_2$, ..., $M_n$.

***Lemma 3:*** *$A_d$ cannot spoof the back-end database*

***Proof:***

A valid message $M_a$ means the back-end database will response an update message after seeing $M_a$. In message $M_a$, $A_d$ can not change the SN, HID, and CID of $T_i$, or the back-end database can not find the corresponding data row in the database. $A_d$ can only change the N and TID. But due to the hash function is preimage resistant, $A_d$ cannot make

$H(SN_i \oplus HID_i \oplus N_a)$

$\neq H(SN_i \oplus HID_i \oplus N_i)$ where $N_a \neq N_i$ and $1 \leq i \leq n$.
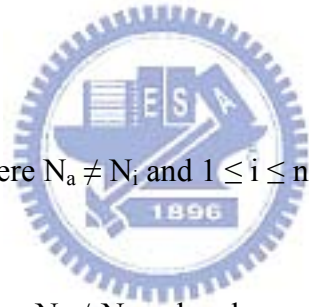
Therefore, to make a nonce $N_a \neq N_i$ and make corresponding $H(SN_i \oplus HID_i \oplus N_a)$ is computationally infeasible. The only way is to make $TID_a = TID_n + n'$ where $n' \in$ **N** and corresponding $M_a$ from $M_n$ to deceive the back-end database by the following way:

$M_a$

$= N_a, H(SN_i \oplus HID_i \oplus N_a), \Delta TID_a \oplus H(SN_i \oplus N_a), H(CID_i \oplus TID_a)$

$= N_n, H(SN_n \oplus HID_n \oplus N_n), (\Delta TID_n + n') \oplus H(SN_n \oplus N_n), H(CID_n \oplus (TID_n + n'))$

$\neq N_n, H(SN_n \oplus HID_n \oplus N_n), \Delta TID_n \oplus H(SN_n \oplus N_n), H(CID_n \oplus TID_n)$

$= M_n$

However due to the hash function is preimage resistant in Definition 1, $A_d$ can not find $CID_i$, $TID_i$, $SN_i$ and $\Delta TID_i$ from $M_i$ where $1 \leq i \leq n$. Thus it is computationally infeasible to make:

$$\Delta TID_a \oplus H(SN_i \oplus N_a)$$
$$= (\Delta TID_n + n') \oplus H(SN_n \oplus N_n)$$
$$\text{and}$$
$$H(CID_i \oplus TID_a)$$
$$= H(CID_n \oplus (TID_n + n'))$$

Where $(\Delta TID_n + n') \oplus H(SN_n \oplus N_n)$ and $H(CID_n \oplus (TID_n + n'))$ is the third and four segments in $M_a$. So, our scheme do resist against spoofing.

## 4.5. Man-in-the-middle Attack

Here we define man-in-the-middle attack in this paper. If attackers can intercept the message from the back-end database, and change the message content to deceive tags. For example, if the attacker can change the update parameter $R_i$ in the update message to another value $R_j$ where $j \neq i$, then the tag will update identity by $H(R_j \oplus CID)$ instead of $H(R_i \oplus CID)$, results in the asynchronous condition between the tag and the back-end database. Worse, attackers could let the tag identity get lost in the database.

To deceive the tag, the attacker must know the SN, CID, and TID of the tag, due to the first segment and the second segment of the update message are chained by the value R. Furthermore, to make the valid segments of the message, attackers must know the SN, CID, and TID of the tag. Due to irreversibility, attacker can not invert the hash value of the original message to know tag's information. Thus man-in-the

middle attack can not work to break the proposed scheme.

**Definition 5:** Adversaries is said to make a successful man-in-the-middle attack if they can win the following game. Given an arbitrary tag $T_i$, Any adversary $A_d$ can adaptively access $T_i$ and get outputs $M_1$, $M_2$, ..., $M_n$ from $T_i$. $A_d$ also can adaptively query the back-end database with $M_i$, where $1 \le i \le n$, and get corresponding update message $U_i$. $A_d$ wins if $A_d$ can make a valid update message $U_a$ for some $M_i$ where $U_a \ne U_i$.

*Lemma 4: $A_d$ cannot make man-in-the-middle attacks.*

*Proof:*

To make a valid $U_a$ for some $M_i$, A chooses some $R_a = R_i \oplus z \ne R_i$ where $z \in \mathbf{Z}$ and make segments $R_a \oplus H(SN_i \oplus (N_i + 1))$ and $H(R_a \oplus CID_i \oplus TID_i)$. The first segment can be made by:

$R_a \oplus H(SN_i \oplus (N_i + 1))$

$= (R_i \oplus z) \oplus H(SN_i \oplus (N_i + 1))$

$= (z \oplus R_i) \oplus H(SN_i \oplus (N_i + 1))$

$= z \oplus R_i \oplus H(SN_i \oplus (N_i + 1))$

$= z \oplus (R_i \oplus H(SN_i \oplus (N_i + 1)))$

But to make the second segment:

$H(R_a \oplus CID_i \oplus TID_i)$

$= H((R_i \oplus z) \oplus CID_i \oplus TID_i)$

$= H(R_i \oplus z \oplus CID_i \oplus TID_i)$

$= H(z \oplus (R_i \oplus CID_i \oplus TID_i))$

However, though $A_d$ knows the $H(R_i \oplus CID_i \oplus TID_i)$. But due to one-way hash function is preimage resistant, A can not get $R_i \oplus CID_i \oplus TID_i$. Therefore it is no way $A_d$ can make such message $U_a \neq U_i$.

## 4.6. Message Loss

Since the unreliable channel between readers and tags, the message transited through the air might be lost. If the last message is missing, the status of a tag and database record might be inconsistent. And if the database record is newer than the actual identity of the tag, it might result in the identity loss in the database. To solve this problem, the back-end database keeps the latest two status of each tag.

If a tag did not update its identity successfully, it uses the same HID and CID next time being read. The back-end database will find a elder row matched the message, by checking the TID of the tag is newer than the database, the back-end database knows the identity did not update successfully last time and uses the old CID to update the identity again. If the identity updates successfully this time then the status of the tag and the database becomes synchronous again.

*Corollary 3: We have proved the proposed scheme resists against spoofing and man-in-the-middle attacks. If an update message to some tag gets lost, adversaries can not change the CID of the tag, nor the database status. Since the database stores the older status of the tag, eventually the status of the tag and the database will be synchronous again after a legitimate reading session finishes.*

## 4.7. Replay

Since attackers can always eavesdrop in the air, record those messages, and replay those messages at any time, the database and tags have to check the freshness of message.

Each Tag in this system has its own TID, and the TID plays as a transaction number. While being read, the TID increases by one. The database record the TID of the tag after each reading session. Thus replay an old message can not work since the database will find out the TID is not newer than the database.

**Definition 6:** Adversaries can do a successful replay attack if they win the following game. Given an arbitrary tag $T_i$, any adversary $A_d$ can eavesdrop and record the message $M_j$ from $T_i$ where $j = 1, 2, 3…$ $M_k$ is the message database most recently sees. $A_d$ wins if $A_d$ sends $M_a$ to the back-end database and get corresponding update message $U_a$ where $1 \leq a \leq k$.

*Corollary 4: The proposed scheme resists against replay attack since the database rejects such $M_a$ where $TID_a \leq TID_k$.*

## 4.8. Compromising Tags

The low-cost passive RFID tags are not tamper-resistant. Attackers could compromise tags and get the information in the memory. Realize that low-cost RFID tags is vulnerable, we should void storing critical information on tags, or attackers can compromise only one tag to break the system.

In our scheme, the tag stores only its own status information like CID, TID, SN, and LST. There is no other global secret on the tag. Compromising a tag only exposes

the information of the tag. Other tags in the same system can still work under the

protection of user privacy.

# 5. Conclusion

RFID provides contactless identification, mass identification, and longer transmission range. We benefit from those features. However, those characteristics might also make attackers to violate user privacy without users' consent.

Due to the price of cost, low-cost RFID tags do not have many resources for security. Symmetric or asymmetric cryptography are not practical to protect private information on the tag. And since low-cost RFID is not tamper-resistant, the private information on tag might be stolen by attackers via physical analysis.

In this paper, we proposed a feasible privacy protection scheme based on dynamic identity for low-cost RFID passive tags. By one-way hash functions, tags store dynamically changeable identity to resist against the tracking problem. And even though a tag is compromised by attackers, it only exposes the status of the tag. Attackers still can not get the real identity of the tag. And other tags can work well under the protection against eavesdropping, replay attack, man-in-the-middle attack, and message loss. We also give the proof of those security properties.

# Reference

[1] Miyako Ohkubo, Koutarou Suzuki, and Shingo Kinoshita. Cryptographic Approach to Privacy-Friendly Tags. *RFID Privacy Workshop,* November 2003.

[2] EPC global Inc. http://www.epcglobalinc.org/

[3] Gildas Avoine and Philippe Oechslin. A Scalable and Provably Secure Hash-Based RFID Protocol. *Proceedings of the 3rd International Conference on Pervasive Computing and Communications Workshops (PerCom 2005 Workshops),* March 2005.

[4] Martin Feldhofer. An Authentication Protocol in a Security Layer for RFID Smart Tags. *Proceedings of the 12th IEEE Mediterranean Electrotechnical Conference (MELECON 2004),* May 2004.

[5] István Vajda and Levente Buttyán. Lightweight Authentication Protocols for Low-Cost RFID Tags. *Workshop on Security in Ubiquitous Computing (UBICOMP),* 2003.

[6] Tassos Dimitriou. A Lightweight RFID Protocol to protect against Traceability and Cloning attacks. *Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm ),* September 2005.

[7] Gwo-Ching Chang. A Feasible Security Mechanism for Low Cost RFID Tags. *Proceedings of the International Conference on Mobile Business (ICMB'05),* 2005.

[8] Dirk Henrici and Paul Müller. Hash-based Enhancement of Location Privacy for Radio-Frequency Identification Devices using Varying Identifiers. *Proceedings of the 2nd IEEE Annul Conference on Pervasive Computing and Communications Workshops (PERCOMW'04)*, 2004.

[9] Dirk Henrici and Paul Müller. Tackling Security and Privacy Issues in Radio Frequency Identification Devices. *2nd International Conference on Pervasive Computing (Pervasive 2004)*, April 2004.

[10] Simson L. Garfinkel, Aril Juels and Ravi Pappu. RFID Privacy: An Overview of Problems and Proposed Solutions. IEEE Security and Privacy, Volume 3, Issue 3, Pages: 34 -43, May 2005.

[11] Aril Juels, Ronald L. Rivest and Michael Szydlo. The Blocker Tag: Selective Blocking of RFID Tags for Consumer Privacy. *In Proceedings of 10th ACM Conference on Computer and Communications Security (CCS'03)*, October 2003.

[12] Stephen A. Weis, Sanjay E. Sarma, Ronald L. Rivest, and Daniel W. Engels. Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems. *First International Conference on Security in Pervasive Computing (SPC 2003)*, 2003.

[13] Shingo Kinoshita, Fumitaka Hoshino, Tomoyuki Komuro, Akiko Fujimura, and Miyako Ohkubo. Non-identifiable Anonymous-ID Scheme for RFID Privacy Protection. *Computer Security Symposium 2003 (CSS2003)*, October 2003.

[14] Shingo Kinoshita, Fumitaka Hoshino, Tomoyuki Komuro, Akiko Fujimura, and Miyako Ohkubo. Low-cost RFID Privacy Protection Scheme. IPSJ Journal, Volume 45, No.8, pp.2007-2021, 2004.

[15] Miyako Ohkubo, Koutarou Suzuki, and Shingo Kinoshita. RFID Privacy Issues and Technical Challenges. Communications of the ACM, Volume 48, No.9, September 2005.

[16] David Molnar and David Wagner. Privacy and Security in Library RFID Issues, Practices, and Architechtures. *11th ACM Conference on Computer and Communications Security (CSS'04)*, October, 2004.

[17] Ari Juels and Ravikanth Pappu, Squealing Euros: Privacy Protection in

RFID-Enabled Banknotes. *Financial Cryptography'03*, 2003.]

[18] RSA Laboratories. http://www.rsasecurity.com/rsalabs/node.asp?id=2176/

[19] Ted Phillips, Tom Karygiannis, and Rick Kuhn. Security Standards for the RFID

Market. IEEE Security and Privacy, November/December, 2005.

[20] RFID Journal. http://www.rfidjournal.com/article/articleview/272/1/79/

[21] Melanie R. Rieback, Bruno Crispo, and Andrew S. Tanenbaum. The Evolution of

RFID Security. IEEE Pervasive Computing, Vol. 5, No.1, pp. 62-69,

January-March 2006.