應用於低成本RFID系統之高效率多層雙向認證協定

學生：鄭依文　　指導教授：黃育綸 博士

國立交通大學 電機學院

電機與控制工程研究所

摘　　要

無線射頻識別（RFID）技術利用射頻信號，以非接觸的方式，自動辨識目標物件，並利用該物件之識別資訊，由後端資料庫系統取得相關資料。其低成本與非接觸等便利性使其得以應用於許多領域。然而，採用 RFID 技術的相關應用於拓展時所遇到的最大瓶頸在於其協定中缺乏隱私保護及資料安全等功能。這使得透過 RF 訊號傳遞的資料，容易遭受到攻擊者的監聽、掃瞄與追蹤。因此，在此篇論文中，我們針對 RFID 系統，終端標籤（TAG）低運算能力的特性，設計了一套雙向身份認證協定。此協定的特色有四項：一、提供標籤與讀取器、讀取器與資料庫、資料庫與標籤等元件之間的雙向認證；二、利用定時更新標籤與讀取器上之金鑰，解決目標物件受到監控追蹤的問題；三、資料庫系統可依據讀取器層級而提供不同的資料；四、在標籤上僅採用簡單的雜湊技術，完成身份認證並達到資料保護的目的。在這篇論文中，我們利用 GNY 邏輯分析，驗證此協定的正確性與資料的完整性，以確保這套雙向認證協定的安全性。其次，分析說明本協定能夠抵檔 RFID 應用中常見的幾種攻擊。最後，我們針對各種效能、儲存需求、運算成本、通訊量等因素，提供完整的分析，並與其他現有 RFID 安全協定做一比較。

An Efficient Multilevel Mutual Authentication Protocol of Low-cost RFID Systems

Student: I-Wen Cheng **Advisor：Dr. Yu-Lun Huang**

Department of Electrical and Control Engineering

National Chiao Tung University

Abstract

The technology of contactless identification of objects, RFID or radio-frequency identification, allows its readers to get application data from the back-end database by object identifications. The convenience resulting from its contactless and low-cost makes it possible to be widely adopted in many areas. However, due to the lack of data privacy and protection mechanism in RFID technology, the RF signals are easily monitored, scanned and traced. This makes the widespread adoption and deployment of RFID technology pose several privacy-related concerns for consumers. In this thesis, we propose a mutual authentication protocol for the low-cost RFID systems. The proposed protocol has four features: 1. it mutually authenticates tags, readers and database; 2. it periodically updates the keys in tags and readers to prevent RFID objects from being illegally traced; 3. it forces readers to get different data from the backend database according to its level; 4. it adopts hash function in designing tags for the authentication and data protection. In this thesis, we also formally prove the correctness and completeness of the proposed protocol by using GNY logics. In addition, we identify several attacks that can be applied to RFID systems and we demonstrate the security of the proposed protocol. Furthermore, we show compare the performances, including storage cost, computational cost and communication cost, with other existing protocols.

誌 謝

在新竹六年，度過了我最精采的求學生涯。感謝指導教授黃育綸老師的教導，老師親民的領導方式及振奮人心的能力都讓學生嘆為觀止，實為楷模。也謝謝口試委員曾文貴教授以及胡竹生教授提供了許多寶貴的意見，讓此篇論文能夠更完整地呈現。

在RTES實驗室裡，因為有同舟共濟的同學、學長姊們以及可愛的學弟妹們陪伴，這兩年的歡笑居多，謝謝大家。再來要感謝的是，男友楷祥的支持與包容，陪我度過低潮時期，甘苦與共，是我身邊的一大支柱。最後，當然要感謝我的媽媽我的弟弟與好友們，因為有你們的鼓勵與關懷，才讓我能衣食無憂心無旁騖地完成我的碩士論文，有著如此溫暖的避風港，十分感激。

謝謝這一路上陪著我的各位，我給予誠心地祝福與感謝。

僅以本論文

獻給我的家人及所有關愛我的人

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The evolution of Radio Frequency Identification (RFID) technique has been exceeding over sixty years. Over seventy percentages of global enterprises already institute RFID programs. This proves that RFID technique is noticed by all trades and professions of the society. Typically, an RFID system is composed of three major elements, tag, reader and back-end database, as shown in Figure 1.1.

An RFID tag is a read-only or read/write device with low computing power, even without batteries, based on the design and the different purposes and can be placed to the object and electronically programmed with unique identification for the object. An RFID reader sends a radio signal to tags . It contains a radio frequency module that is able to read/write data to/from RFID tags, when receiving information from tag, it sends the data to the back-end server within database. A back-end database stores and manages various records and authentication data for tags, such as object identification, expiration date, application data, utility rate, read time, etc. The back-end database can also be used to receive requests from readers and authenticate tags.

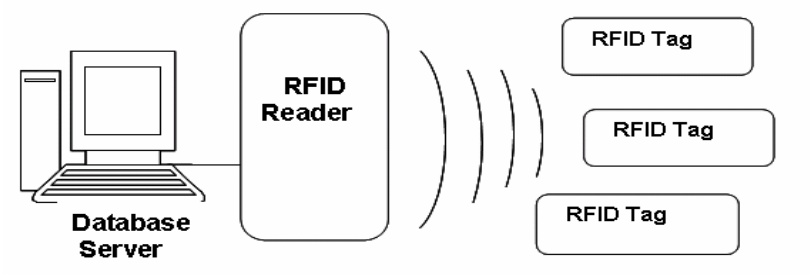


Figure 1. 1 Typical RFID Systems

With the characteristics of contact less, non directional demands ,wireless transmission, and long read range, RFID technique is widely applied to barcode-like applications and object identifications, such as supply chain management, inventory, manufacturing industry, object identification, location tracking, bank notes [1], e-passports [18] and so on. The convenience brought by RFID makes its popularity possible in the near future. However, due to the low computing power in RFID tags, the popularity of RFID technology is followed by security and privacy issues. The main problem is that tags do not have enough computing power to authenticate those readers sending query message to them. If customers are carrying products with these insecure tags, the location privacy is easily be exposed and traced by the evildoers.

Designing authentication protocols for RFID systems is one of the solutions to overcome the security and privacy issues mentioned above. In the past few years, researchers proposed many protocols [2-8] focusing on RFID security and privacy issues. These works try to make a balance between security and restricted resources in RFID systems. However, some of them are still vulnerable to attacks; some are requiring heavy computing power in RFID tags or backend database. Since the lower computation on tags, those secret keys kept by tags used for mutual authentication can be cracked by eavesdroppers in modern times. In our work, we not only present a mutual authentication protocol but also propose additional key update scheme to against varied attacks, we use one way hash function and XOR gate to achieve our objective.

The remainder of this paper is constructed as follows: Chapter 2 defines common attacks in RFID systems and then briefly introduces and analyzes previous works on the area in chapter 3. Several schemes are mostly focused on how to guarantee security and protect user privacy in low-cost RFID environment. In chapter 4, in order to satisfy low computational power on the existing RFID, the proposed authentication protocol adapts simple cryptographic primitives,

one-way hash function, and random number generators. Mutual authentication is the basis for the proposed protocol, and with an extra key update scheme. Chapter 5 gives a formal proof of the purposed protocol based on GYN logic [22] followed by the security analysis in chapter 6. Finally, chapter 7 depicts the comparison with other works. In the last chapter, we conclude this thesis.

Chapter 2

RFID Attacks

One of the major challenges in RFID systems is that it is vulnerable to several attacks if there is no reliable security mechanisms built into the systems. This chapter depicts the common attacks in RFID systems. According to the modification of RFID data, these attacks fall into two categories: passive and active.

Passive Attacks

A passive attack monitors or listens in on the communication between two parties, attempting to retrieve valuable information without modifying the data stream. Such an attack makes no effect to the forwarding progress. Eavesdropping and traceability attacks are classified as passive attacks.

- *Eavesdropping attack:* In the eavesdropping attack, an adversary captures a copy of the RF signal transmitted between a legitimate RFID reader and a tag. The RF signal is then analyzed by the eavesdropper to obtain sensitive data, such as anamnesis, personal privacy and so on. In RFID systems, two types of communication channels are defined: The Reader-to-Tag channel or forward channel is assumed for readers broadcast with strong RF signal to long distance, perhaps 100 meters. The Tag-to-Reader channel or backward channel is relatively much weaker, and may only be monitored by eavesdroppers within the tag's shorter operating range [11]. Readers will only collect information from tags nearby (e.g. three meters) as limitation for passive tags, so we assume this attack is much easier happened on forward channel then on

backward channel. This relationship is illustrated in Figure 2.1.

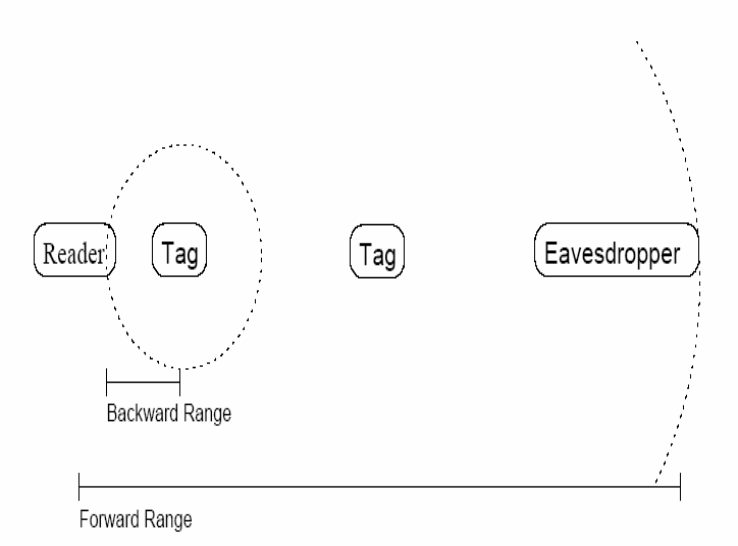


Figure 2.1 Forward vs. Backward Channel

- *Traceability attack:* Traditionally, the tag identification is sent to the reader upon responding the query signal. These constant responses make the moving tag easy to be traced, that is called the traceability attack. It means that people who carry these tags is also traceable. Therefore, adversary only needs to set up readers at different places, and read tags around then he can know the location of the bearer, privacy of the bearer is all gone. In other words, traceability of RFID tags reveals the privacy in personal routes.

Active Attacks

The active attacks are different from passive attacks, these kinds of attacks are attacks modifying, forging or replaying the RF signals between RFID readers and tags.
Man-in-the-middle, counterfeit, replay and denied-of-service attacks are classified as active attacks.

- *Man-in-the-Middle attack*: As the name indicates, an attacker impersonates one of the

communication parties. He tries to modify and replay the data to and from the counter parties. It means that the attacker is analyzing the weakness of the communication protocol and is able to get the negotiated key to decrypt the cipher data exchanged between the reader and the tag in an RFID system.

- *Counterfeit attack:* In such an attack, the attacker eavesdrops the communication channels between tag and reader to copy and masquerade the response messages. The attacker tries to involve into the forwarding process and retrieve the valuable information from the system.

- *Replay attack*: Replay attack indicating that an unauthorized third party attempt to copy and resend the RF signals and to cheat the reader or tag to get valuable information.

- *Denial-of- Service (DoS)*: The DoS attacks make RFID system resources unavailable and could not service tags. For example, the adversary may actively send RF signals to stuff up all the channels and make the reader busy with the unexpected signals. The reader then is not able to service to the validate tags.

Chapter 3

Related Works

Since fundamental RFID tags are limited in resources for cryptograph operations, it results in some privacy and security issues. Hence that many schemes are proposed to address the RFID security concerns. We now list various proposed cryptographic approaches to tags' privacy problem.

1) Hash Lock scheme

This scheme is low cost, since it only requires implementing a one way hash function on the tag and managing keys on the back-end server[11,12] as shown in Figure 3.1. From the difficulty of inverting a one way hash function, this scheme prevents unauthorized readers from reading tag contents. The verification process of the reader for each tag describe as follows. To lock a tag, the bearer of a tag stores the hash of a random key as the tag's metaID, i.e. metaID = hash (key) and the reader has key k for each tag. To unlock a tag, the bearer queries the metaID from the tag, and inquires the appropriate key in the back-end database and finally transmits the key to the tag. The tag hashes the key and compares it to the stored metaID. If the values match, it unlocks itself and offers its full functionality to any nearby readers. However, the tag has a fixed return value metaID, so this solution can not defeat traceability attack. On the other hand, while readers are sending the key on the forward channel, and the eavesdropper may easily get the key. Since the key is known by the eavesdropper, he can use it to spoof the tag to the reader.

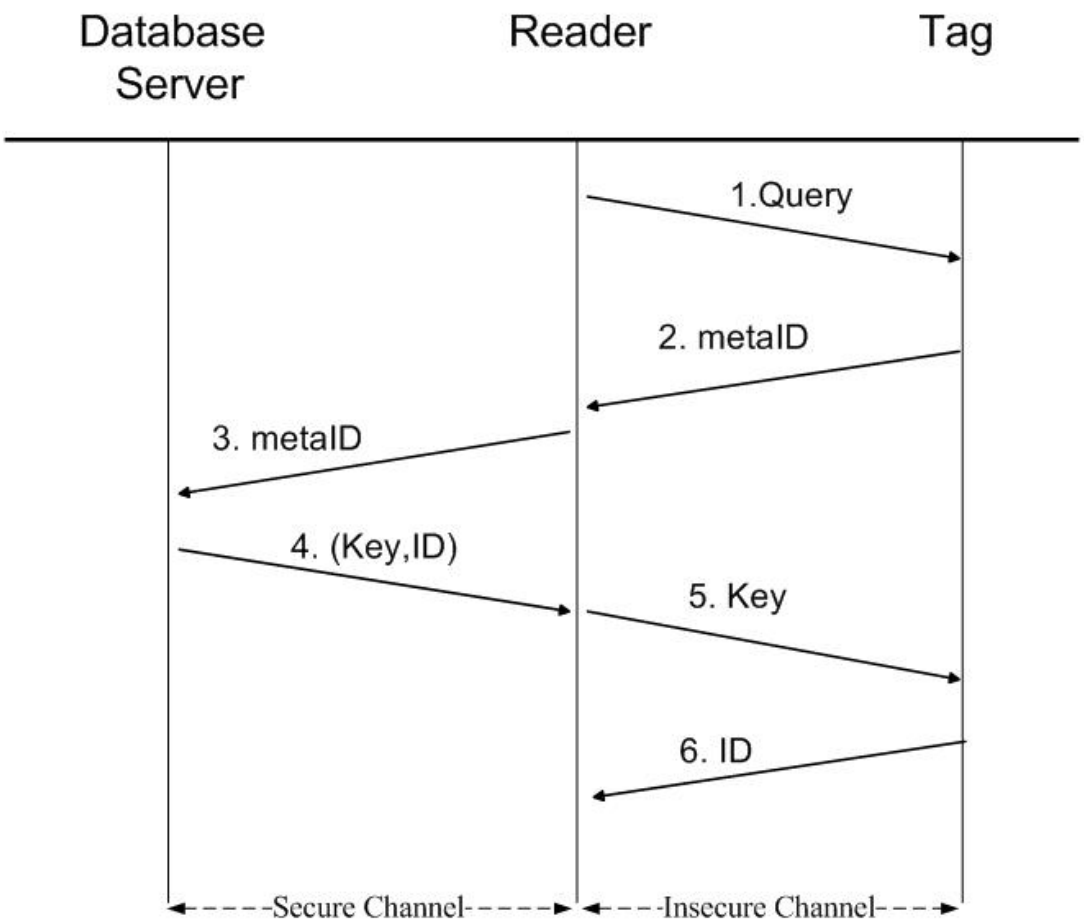


Figure 3. 1 Hash Lock Scheme

2) Random Hash Lock scheme

This is an extended scheme from Hash Lock scheme, tags in these schemes are provided with not only a hash function but also a random number generation [11]. Upon query, tag sends the different response every time. The response including a random number and a hash value of its ID_k and the random number r. Reader will direct sends the returned value to database server, then server calculates all ID_k with r to find the matching ID ,and transmits the ID back to the tag. This scheme improves the traceability in Hash Lock scheme, but may cause a heavy loading for database doing hash functions many times with a large number of tags ID. The procedure for the

scheme is illustrated in Figure 3.2.

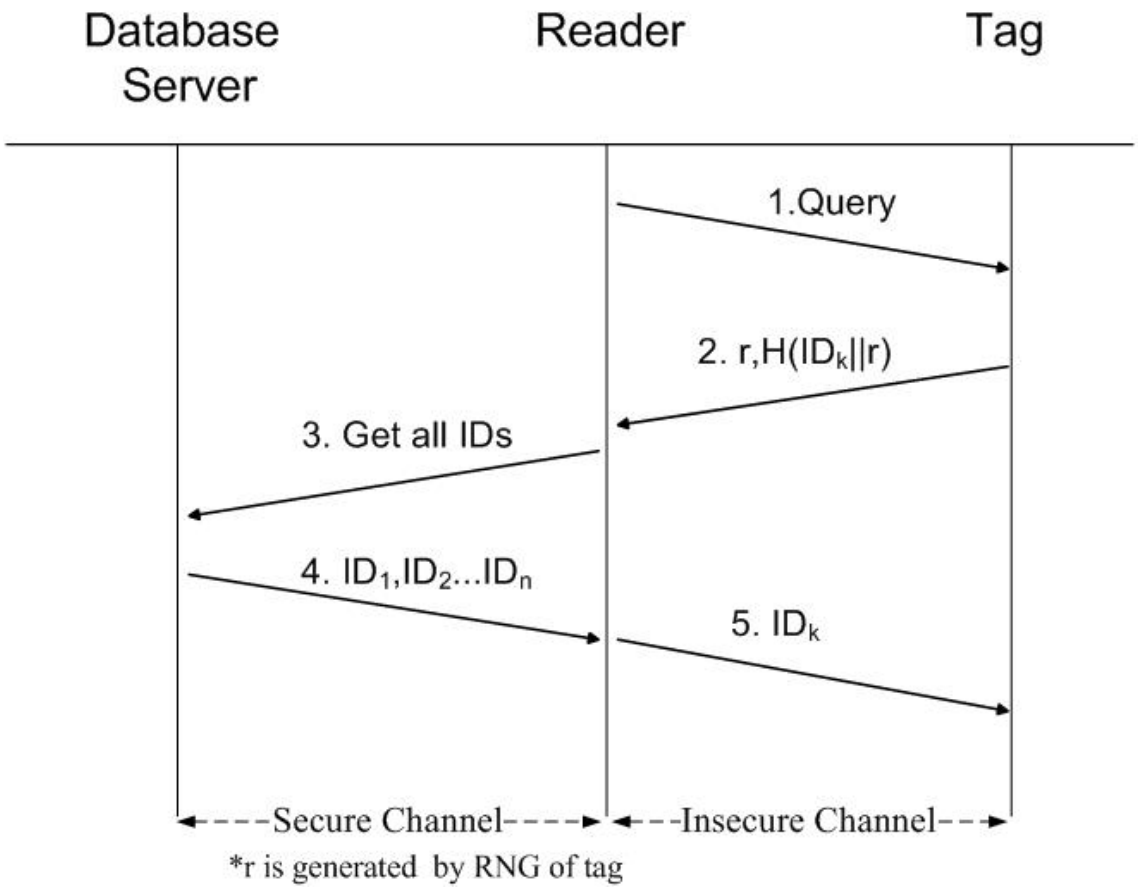


Figure 3. 2 Random Hash Lock Scheme

3) Hash Chain scheme

In this scheme, the authors design a scheme that uses only two hash functions to prevent backward tracking, so the privacy of each tag can be protected and the cost is also reduced [3]. In figure 2.2, H and G are hash functions. Initially, tag has a secret information s_1 .When the tag is i-th queried, it sends $a_i=G(s_i)$ to the reader and renews information $s_{i+1}=H(s_i)$. The reader sends a_i to the back-end database. The back-end database holds a list of pairs (ID, s_1), where s_1 is

different from each tag. After receiving tag's output a_i from reader, the back-end database calculates $a'_i = G(H^i(s_1))$ for each s_1 in the list, and find if $a'_i = a_i$,then return the corresponding ID. In this idea, the tags modify the secret information automatically via s querying so that the tag is recognized by authorized readers only. The protocol ensures the anonymity of the tag ID , satisfy the user privacy demand. The drawback of the protocol is that secrets asynchronous problem may occur between tags and the database server. In addition, it has the same problem as discussed in Random Hash Lock scheme before. The scheme is illustrated in Figure 3.3.

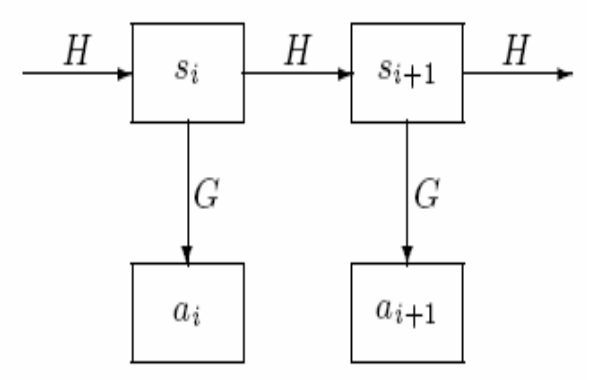


Figure 3. 3 Hash Chain Scheme

4) "Minimalist" Cryptography

Juels [14] proposes a minimalist system involves only an XOR operation, and relatively little storage. In this work, the tag and the valid verifier share the same pseudonyms which can strengthen privacy for RFID tags. An output of the tag is different by rotating pseudonyms with each access, thus only a valid reader can recognize as the same tag from diverse outputs . It is evident that an attacker can repeating query a tag to collect the secret list of all pseudonyms. To protect against that, Juels proposes some enhancements to the basic system. First, tags "throttle" their names only at a suitably slow rate. Second, the common secrets can be overwritten by valid readers. The advantage is each β_i and γ_i values in the tag are no longer valid for future rounds,

for this reason the scheme eschews the counterfeit attack .

5) Lightweight Authentication algorithms

I. Vajda and L. Buttyan proposed five lightweight tag authentication protocols [4] in 2003. These protocols are XOR、Subset、Squaring、RSA、Knapsack, the above protocols involving with several keys, and they said re-keying issue will be discussed in the future reports. The authors also declared theses lightweight protocols may suffer attacks by a powerful attacker, but their goal is to find out the best trade-off between security and performance. In 2004, Martin Feldhofer proposed a novel solution that uses a symmetric key encryption algorithm AES (128-bit keys) to implements the ISO/IEC 18000 protocol in FPGA [2]. However, it is not suitable for low-cost RFID system. Later in 2005 and 2006, mutual authentication protocols proposed in[10,20] solve the tracking problem by using update key every time so that identifier changes ,too. Even though both schemes defeat many attacks, they have a common problem, exhausted searching in database server. Then in [6], Tassos proposes a solution that can solve exhausted searching in database problem by maintaining the previous ID in database. Tags in [7, 19, 5, 15] are equipped with a random number generator or nonce generator even symmetric encryption functions in order to enhance the security and privacy of RFID application system, [5] also solves the de-synchronization problem and doesn't need exhaustive search in back-end database, but these schemes have much higher cost. [16], which is a strong and robust protocol provide both forward and backward intractability, but it's a pity it's not easy to implement in RFID tags under the existing standard. Moreover, Pedro proposed a protocol [21], they claims it can be implemented at low-cost (<1K logic gates), avoid many attacks, and have high efficiency in searching of the database.

Chapter 4

Proposed Lightweight Multi-level Authentication Protocol

The notations used in our protocol are presented in Table 4.1.

Table 4. 1 Notations

Notation	Meaning
T	RFID tag or transponder.
R	RFID reader or transceiver.
DB	Back-end database.
ID	Unique chip serial number that is embedded into T.
RID	Unique number for each R.
BID	Unique number for DB. Secret key shared between reader and tag.
$E_{Kr}(\ )$	Symmetric-key encryption functions with the key Kr.
$D_{Kr}(\ )$	Symmetric-key decryption functions with the key Kr.
$H_K()$	Keyed hash functions with the secret key K.
Kr	Secret key shared between reader and database.
$Kt1$	Secret key shared between tag and database
$Kt2$	Secret key shared between tag and database
RNG	Random Number Generator
N_R	Random number generated by RNG.
$\oplus$	Exclusive-or (XOR) function.
TID	Temporary ID index for searching in DB, $TID = H_{kt2}(ID \oplus Kt1)$
D_L	A field for all other application related data of tag (L=level).
D_0	If level=0, indicated that public information in the memory of tag.

4.1 Assumptions

We design the protocol used for passive tag that has a hash function and bitwise XOR function. We suppose readers are more powerful than tags, so readers in our system require a random number generator, and use the symmetric-key cryptosystem to encrypt and decrypt message, and each of them has dissimilar level to access data from database server. We assume that reader is not a trust third party (TTP) and consider the problem that attackers may access the information from database through wireless connection under insecure situation.

4.2 Protocol Design

The basic message flows is presented in Figure 4.1. The detailed procedures for each step are described.

- Secure Tag $(\mathcal{T})$: A secure tag consists of two parts: data memory and computing unit. The first part is a memory unit, which consists of read only memory (ROM) and read/write memory. Each tag has a unique chip serial number (*ID*) in the ROM, and non-secure data (D_0), a pair of keys ($Kt1$、$Kt2$) and ID of database (*BID*) in the read/write memory. The pair of keys is used for identifying the authenticated tags. Second part is computing unit, which is able to perform simple computation, such as performing keyed hash functions, exclusive-or (XOR) function and so on.

- Secure Reader $(\mathcal{R})$: A secure reader is the same as secure tag, consists of two parts. The memory unit has ID of an authenticated reader (*RID*)、*BID* and a number *Kr* in the read/write memory, and random number (N_R) in the random access memory (RAM). The number *Kr* is used for identifying the authenticated readers. However, the computing unit which is able to perform keyed hash functions, creating simple pseudo random numbers

(*RNG*) and so on.

- Secure Database *(DB)*: The secure back-end database consists of two tables: reader table and tag table. The reader table stores the *RID, Kr* and its *level*. The *level* of reader is assigned by database. The database is assigned to provide different level of information to different level of readers. Take the level 1 authorized reader for example, it can only read D_1 from database, but the level 2 authorized reader can read D_1 and D_2 from database. The tag table stores the *ID*, *TID, Kt1*, *Kt2* and $Data_L$. The $Data_L$ is the security information data of tag.

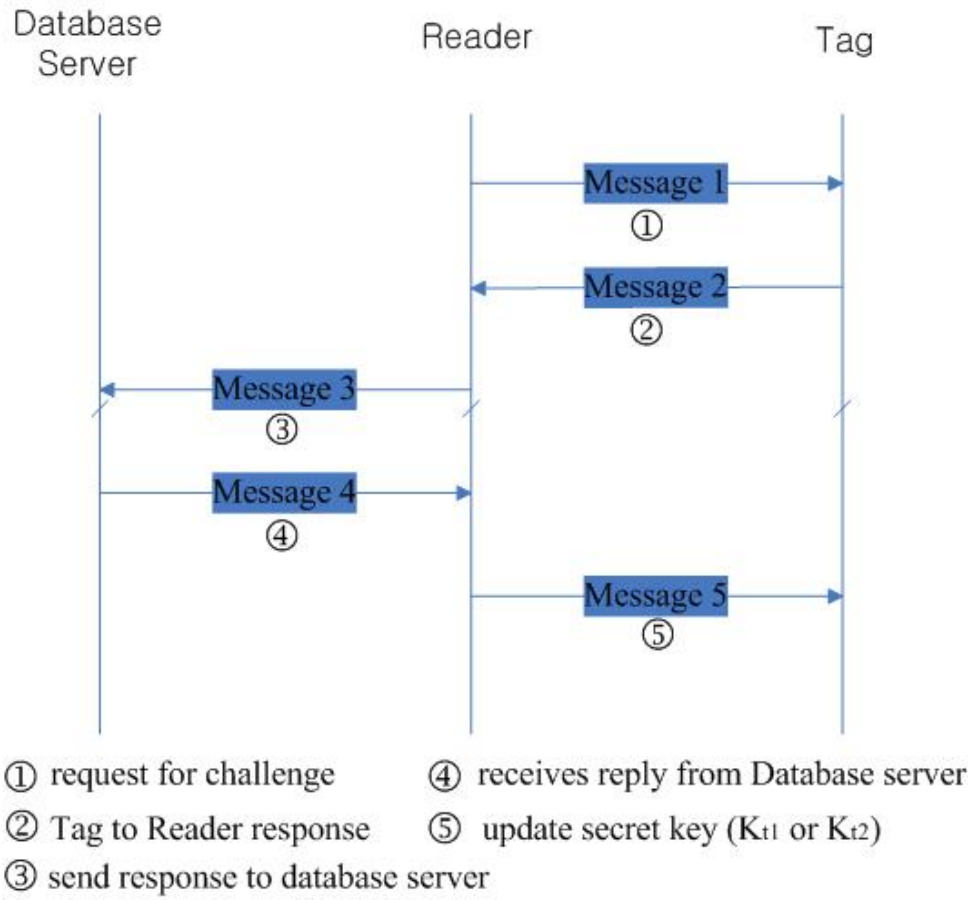


Figure 4. 1 Message Flows

4.3 Authentication Multi-level Protocol

In order to provide a safety environment between the communication among tag, reader and database, we design a secure multi-level authentication protocol. Our proposed protocol can be divided into three phases: initialization phase, communication phase and key update phase. We describe the three phases in individual sections.

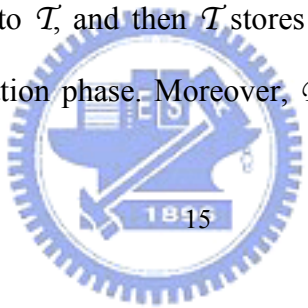
4.3.1 Initialization Phase

In initialization phase, the main work is that how a reader/tag registers to the database server to be a qualified reader/tag. In this phase, the communication among them are been executed under secure channel. We will introduce the registration of a reader at first:

For reader, $\mathcal{DB}$ generates a key *Kr*, a unique *RID* and the identifier of $\mathcal{DB}$, *BID,* then stores them in the reader table. Second, $\mathcal{DB}$ forwards *Kr*, *RID* and *BID* to $\mathcal{R}$, and when $\mathcal{R}$ receives them then stores them in memory to be the qualified authentication in communication phase. $\mathcal{DB}$ will also follow the *RID* to be the start blank of the reader table to search the qualified $\mathcal{R}$ in authentication process in the communication phase. Moreover, because our protocol is designed as multilevel type, we will define and record the level of the reader in the reader table.

The following part is the registration between tag and database:

Initially, $\mathcal{DB}$ generates a set including a unique *ID* ,two keys *Kt1*, *Kt2* and its hash value of *ID* with *Kt1* as *TID*: *[ID,Kt1, Kt2,* H_{Kt2} *(ID⊕Kt1)]* and store them in tag table. $\mathcal{DB}$ uses this *TID* to be the start blank of the tag table and register the $Data_L$ of the $\mathcal{T}$ within $\mathcal{DB}$. The *TID* will be the $Data_L$ indicator which is used for $\mathcal{DB}$ to search and restore $\mathcal{T}$ in the communication phase. The pair of keys, *BID*, and *ID* will be sent to $\mathcal{T}$, and then $\mathcal{T}$ stores them in the memory to be the qualified authentication in the communication phase. Moreover, $\mathcal{DB}$ will also write some non-secure data



into $\mathcal{T}$'s read/write memory.

4.3.2 Communication Phase

As shown in Figure 4.2, the proposed protocol is described in detail according to the sequence of messages exchange. A one session of multi-level authentication process is from step 1 to step 4.

Step 1: In this step, $\mathcal{R}$ usually applies a collision-avoidance protocol like the secure binary tree walking or any other anti-collision protocol. $\mathcal{R}$ generates a fresh random nonce, N_R, and sends N_R to query $\mathcal{T}$ nearby.

Step 2: When $\mathcal{T}$ is queried by $\mathcal{R}$, $\mathcal{T}$ yields *X* and *D* to $\mathcal{R}$. *X* is composed of *TID* and the output of keyed hash function, $H_{BID}(N_R)$. *TID* is used as the identification information to $\mathcal{DB}$ and $H_{BID}(N_R)$ value can be used to verify the legitimate $\mathcal{R}$ with N_R .The key, *Kt1*, shared by $\mathcal{T}$ and $\mathcal{DB}$ is aimed to authenticate the validity of $\mathcal{T}$. Moreover, *BID* is the shared secret between $\mathcal{R}$ and $\mathcal{T}$.

Step 3: *R* creates the keyed hash value with N_R by *BID* after step 1. Until receives response, $\mathcal{R}$ uses the value to do X-or function with *X*, and will get *TID*. $\mathcal{R}$ will also get the non-secure data D_0 from $\mathcal{T}$. After that, $\mathcal{R}$ encrypts *TID* and N_R together with the key *Kr* and then forwards it to $\mathcal{DB}$. At the same time, $\mathcal{R}$ also transmits *RID* to $\mathcal{DB}$ for identification. Within this step, $\mathcal{DB}$ authenticates $\mathcal{R}$ and $\mathcal{T}$ consequently with *RID* and *TID*, respectively.

- At first, $\mathcal{DB}$ judges whether the forwarded *RID* is legal or not by searching with reader information throughout reader database table. If *RID* is registered, $\mathcal{DB}$ takes the corresponding key *Kr* so it can decrypt the received data. *Kr* is the shared secret key only between $\mathcal{R}$ and $\mathcal{DB}$, so $\mathcal{DB}$ can detect the illegal $\mathcal{R}$ and discards the forwarded message.

- Furthermore, $\mathcal{DB}$ gets *TID* and verifies it from tag's information of database. Then, $\mathcal{DB}$ authenticates $\mathcal{T}$ with K_{t1} and K_{t2}. If the authentication is failed, $\mathcal{DB}$ will terminate in this session.

Since $\mathcal{DB}$ initially stores the unique chip serial number, *ID*, and temporary ID index *TID* . $\mathcal{DB}$ can evaluate the linkage between the forwarded authentication information $H_{kt2}(ID)$with K_{t1} and $\mathcal{T}$ itself in order to prevent forgery. Forgery can be detected and prevented by $\mathcal{DB}$ at this moment.

At the same time, $\mathcal{DB}$ detects and prevents the man-in-the-middle attack since N_R is used as the factor of the man-in-the-middle attack detection. Similarly, the replay attack can be also detected and prevented simultaneously.

Step 4 : $\mathcal{DB}$ encrypts the corresponding security data and N_R in accordance with *level* of reader. Then, $\mathcal{DB}$ replies the message $E_{Kr}(D_L, N_R)$. After this step, the corresponding decryption process, $D_{Kr}(D_L, N_R)$, is processed by $\mathcal{R}$ to get security data. Thus, security data of $\mathcal{T}$ is securely obtained only by the legitimate $\mathcal{R}$ although the adversary eavesdrops the reply messages on the insecure channel.

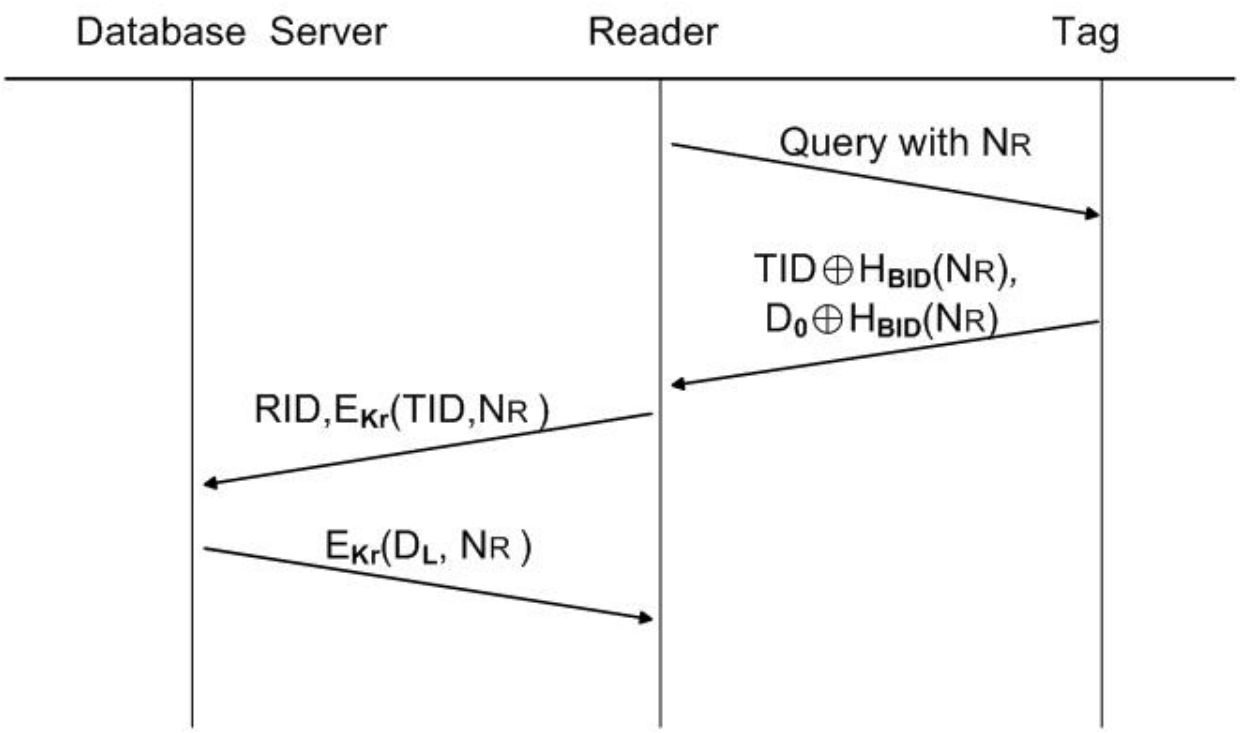


Figure 4. 2 Communication Phase

4.3.3 Key Update Phase

In [20], J. Yang et al. proposed a scheme with key changed in every session. In our opinion, changing key every session is a heavy computation cost for database server.

As shown in Figure 4.3, there are five steps in the process of updating Key *Kt1* and *Kt2*, and since step 1 to step 3 is the same as in communication phase, we don't explain again, and we focus on step 4 and step 5.

Step 4: After $\mathcal{DB}$ received the message 3, $\mathcal{DB}$ verifies $\mathcal{T}$ and detects the key *Kt1* is overdue. Then, $\mathcal{DB}$ generates a new key Kt1' substitute Kt1 for $\mathcal{T}$ and also alter the corresponding *TID* by using the new key Kt1'. Afterward $\mathcal{DB}$ calculates $K_{new}=Kt1' \oplus Kt1 \oplus Kt2$, $Y=H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1')$ and encrypts them with N_R and corresponding security data in accordance with *level* of reader.

Then, $\mathcal{DB}$ replies the message E_{Kr} (K_{new}, N_R, Y, D_1). After this step, the corresponding decryption process, D_{Kr} (K_{new}, N_R, Y, D_L), is processed by $\mathcal{R}$ to get security data.

Step 5: Since $\mathcal{R}$ meets the update message, $\mathcal{R}$ sends Y, K_{new}, to $\mathcal{T}$ to update Kt1 stored in $\mathcal{T}$. After that, $\mathcal{T}$ obtains the new key Kt1' by doing XOR function with Knew, Kt1 and Kt2. $\mathcal{T}$ also calculates H_{Kt1} $(H_{Kt2}$ $(ID) \oplus N_R \oplus Kt1')$ and compare with Y, then Kt1' substitutes Kt1. Up to now, update process of Kt1 is completed.

In other case, $\mathcal{DB}$ detects that the key Kt2 is overdue on Step 4, and then Kt2 will be updated .The update process of Kt2 is similar to Kt1.Use Kt2' instead of Kt1' from Step 4 to Step5 explained above. The most important change is $Y=H_{Kt2}(H_{Kt1}(ID) \oplus NR \oplus Kt2')$ in Kt2 update process.

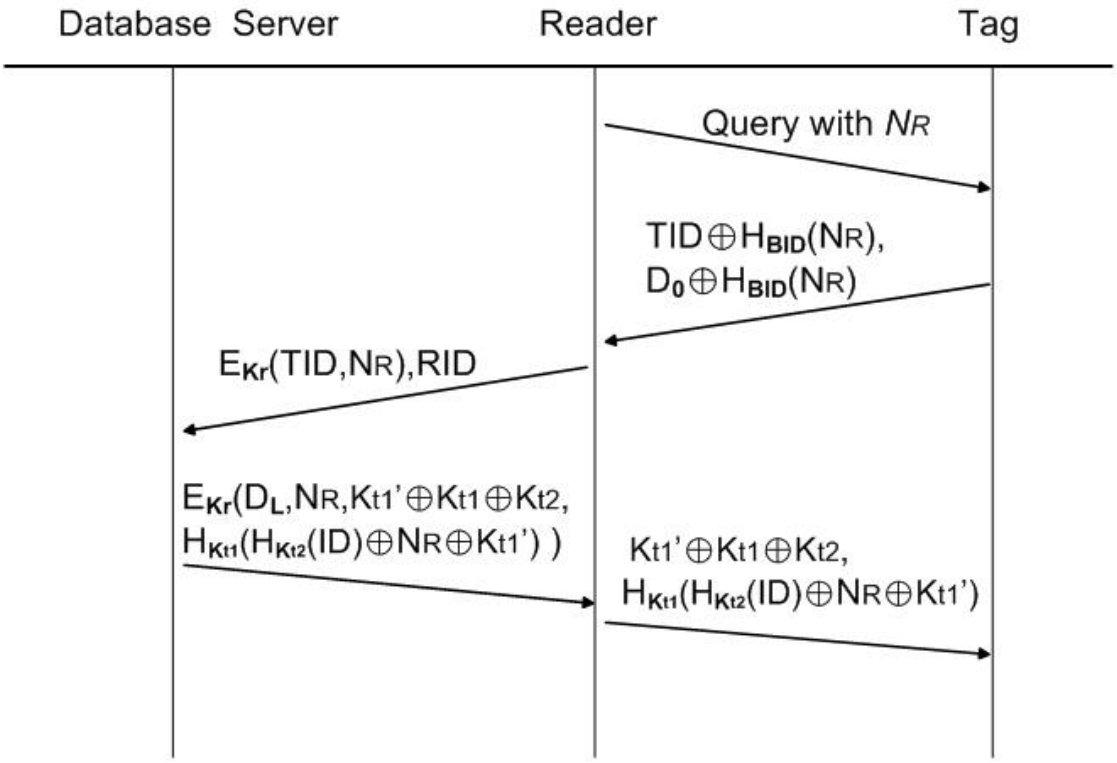


Figure 4. 3 Key Update Phase (for Kt1 Kt2)

Besides updating key in $\mathcal{T}$, the key Kr in $\mathcal{R}$ will be also updated by $\mathcal{DB}$. The key update protocol for Kr is shown in Figure 4.4. There are three steps described below.

Step 1 : In this step, $\mathcal{R}$ keeps the record of Kr, once Kr is overdue, $\mathcal{R}$ sends request to $\mathcal{DB}$ voluntarily. $\mathcal{R}$ generates a fresh random nonce, N_R, and $H_{BID}(N_R, Kr)$, an output of keyed one-way hash function. $\mathcal{R}$ sends them encrypted with Kr as update information and RID to $\mathcal{DB}$.

Step 2 : At first, $\mathcal{DB}$ searches the RID in reader table and find out the related key Kr to decrypt the received message. Then, $\mathcal{DB}$ verifies whether the forwarded N_R is valid by comparing the value $H_{BID}(N_R, Kr)$. BID and Kr is the shared secret key between $\mathcal{DB}$ and $\mathcal{R}$, so $\mathcal{DB}$ can recognize it legal $\mathcal{R}$ or not. If $\mathcal{R}$ is legal, $\mathcal{DB}$ generates a fresh random nonce, N_{DB}, and a new key Kr' for the $\mathcal{R}$, and hide Kr in $Z=Kr' \oplus BID \oplus Kr$. Afterward $\mathcal{DB}$ encrypts them together with a output of the keyed one-way hash function, $H_{BID}(N_{DB}, Kr')$,by Kr, and sends it to $\mathcal{R}$. At the same time, the nonce N_R obtained from $\mathcal{R}$ is also retransmitted to $\mathcal{R}$.

Step 3 : $\mathcal{R}$ checks the nonce N_R, and confirms it sent by $\mathcal{R}$ itself in step 1 before. After that, $\mathcal{R}$ decrypts the message and does X-or function with Z, BID, and Kr, so $\mathcal{R}$ obtains the new key Kr'.Kr' is used to verify the value $H_{BID}(N_{DB}, Kr')$. If $\mathcal{R}$ successfully finishes the verification, $\mathcal{R}$ uses Kr' replace Kr, and then encrypts N_{DB} by Kr' as a reply message to $\mathcal{DB}$. On $\mathcal{DB}$ side, $\mathcal{DB}$ checks the nonce N_{DB}, if it is admissible, $\mathcal{DB}$ will also replace Kr by Kr' in R table.

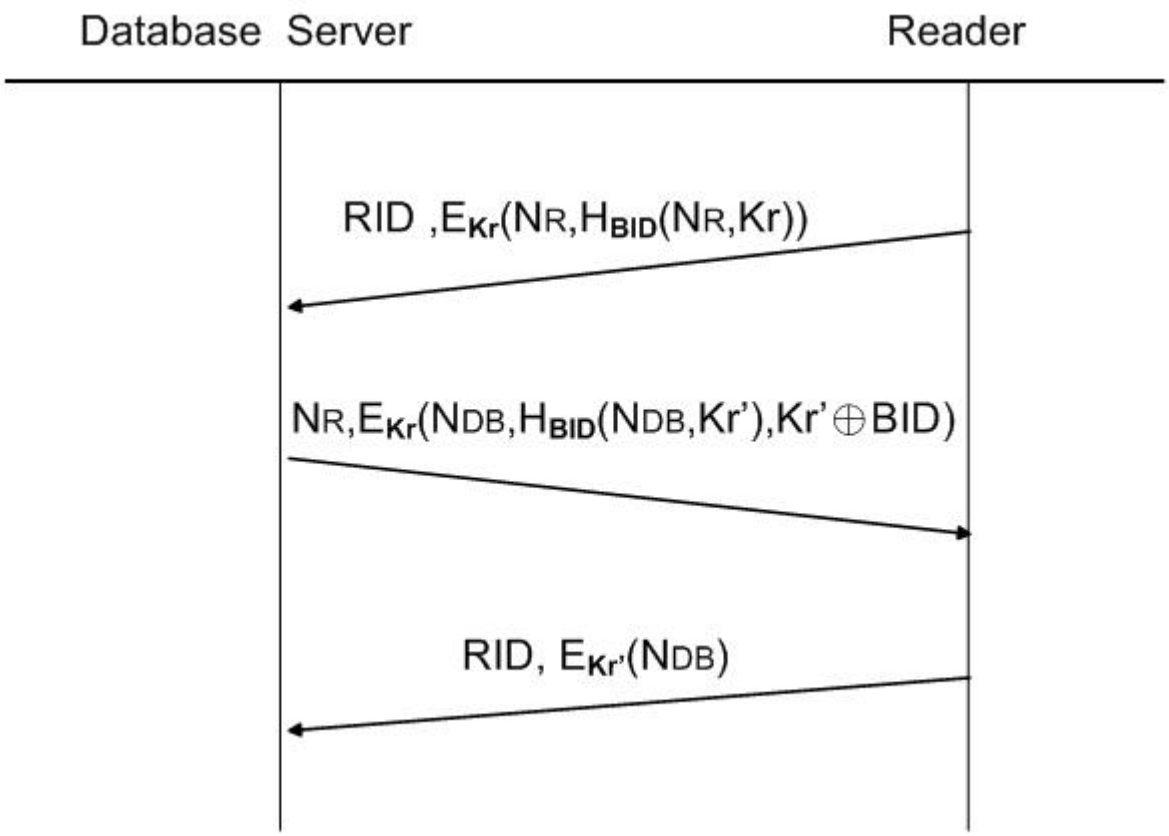


Figure 4. 4 Key Update Phase (for Kr)

BID is an identifier of a $\mathcal{DB}$, both authorized $\mathcal{R}$ and $\mathcal{T}$ from the same system has the same share secret BID in the beginning. Moreover, BID plays a major role in every phase, here, it is recommended that $\mathcal{DB}$ changes BID after a period of time avoid various attacks, and $\mathcal{DB}$ will also ask $\mathcal{R}/\mathcal{T}$ updating their BID in their memory. New BID is the hash value of old BID, $BID_{i+1}=H(BID_i)$. Every time when $\mathcal{DB}$ request $\mathcal{R}/\mathcal{T}$ to update BID, they hash BID in hand until get the same value with $\mathcal{DB}$'s.

Chapter 5

GNY Proof

The proposed protocol is proved for correctness and security using GNY cryptographic protocol. The GNY protocol is usually used for analysis of security of cryptographic protocol. The following notations are already defined in Table 1, and other symbols and rules can be referenced in [22].There are two parts in our protocol: communication phase and key-update phase, we will give the GNY proof for both two parts. In the proposed protocol, there are three participating principals, *DB* the Database server, *R* the Reader, and *T* the Tag. The goals and assumptions are list below.

Goals

Communication phase:

G1. $DB|\equiv T\ |\sim \#H_{kt2}(ID \oplus Kt1)$

G2. $DB|\equiv T|\equiv DB \xleftrightarrow{Kt1,Kt2} T$

G3. $T|\equiv DB|\equiv DB \xleftrightarrow{Kt1,Kt2} T$

G4. $DB|\equiv R|\equiv DB \xleftrightarrow{Kr} R$

G5. $R|\equiv DB|\equiv DB \xleftrightarrow{Kr} R$

Update phase for Kt1/Kt2:

G6. $T|\equiv DB \xleftrightarrow{Kt1',Kt2'} T$

G7. $T|\equiv DB|\equiv DB \xleftrightarrow{Kt1',Kt2'} T$

G8. $T|\equiv DB\ |\sim \#H_{kt1}(H_{kt2}(ID) \oplus N_R \oplus Kt1')$

G9. $T|\equiv DB\ |\sim \#(Kt1 \oplus Kt1' \oplus Kt2)$

Update phase for Kr:

G10. $R|\equiv DB \xleftrightarrow{Kr'} R$

G11. $DB|\equiv R|\equiv DB \xleftrightarrow{Kr'} R$

G12. $DB|\equiv R|\equiv DB \xleftrightarrow{BID} R$

G13. $R|\equiv DB|\equiv DB \xleftrightarrow{BID} R$

G14.DB|≡R∋Kr'

Assumptions

Communication phase:

A1. $T \ni H_k(X)$

A2. $T \ni (Kt1, Kt2, BID)$

A3. $T| \equiv \#(N_R)$

A4. $R \ni H_k(X)$

A5. $R \ni (N_R, Kr, BID)$

A6. $R| \equiv \#(N_R)$

A7. $R| \equiv \phi(N_R)$

A8. $DB \ni H_k(X)$

A9. $DB \ni (Kt1, Kt2, Kr, ID, BID)$

A10. $DB| \equiv \#(N_R)$

A11. $T| \equiv DB \xleftrightarrow{Kt1, Kt2} T$

A12. $T| \equiv R \xleftrightarrow{BID} T$

A13. $R| \equiv R \xleftrightarrow{BID} T$

A14. $R| \equiv DB \xleftrightarrow{Kr} R$

A15. $DB| \equiv DB \xleftrightarrow{Kt1, Kt2} T$

A16. $DB| \equiv DB \xleftrightarrow{Kr} R$

Update phase for Kt1/Kt2:

A17. $T| \equiv DB| \Rightarrow DB \xleftrightarrow{K1'} T$

A18. $DB| \equiv T \ni (Kt1, Kt2, ID)$

A19. $DB| \equiv DB \xleftrightarrow{K1'} T$

Update phase for Kr:

A20. $R| \equiv \#(N_{DB})$

A21. $R| \equiv DB| \Rightarrow DB \xleftrightarrow{Kr'} R$

A22. $DB| \equiv \#(Kr', N_{DB})$

A23. $DB \ni Kr'$

A24. $DB| \equiv DB \xleftrightarrow{Kr'} R$

A25. $R| \equiv DB \xleftrightarrow{BID} R$

A26. $DB| \equiv DB \xleftrightarrow{BID} R$

The four messages exchanged during the communication phase are written in the type of the GNY logic as follows:

Msg1 $R \rightarrow T$: $T \triangleleft *N_R$

Msg2 $T \rightarrow R$: $R \triangleleft * [H_{Kt2}(ID \oplus Kt1) \oplus H_{BID}(N_R), D_0 \oplus N_R]$

Msg3 $R \rightarrow DB$: $DB \triangleleft * [RID, \{H_{Kt2}(ID \oplus Kt1), N_R\}_{Kr}]$

Msg4 $DB \rightarrow R$: $R \triangleleft * \{N_R, D_L\}_{Kr}$

Analysis (1)

Message 1 : $T \triangleleft * N_R$

Applying T1,P1,A3($T \mid\equiv \#(N_R)$ and F1,we get $T \ni N_R$ .That is T possesses N_R.

---

1. $T \triangleleft N_R$
2. $T \ni N_R$

Message 2 : $R \triangleleft * [H_{Kt2}(ID \oplus Kt1) \oplus H_{BID}(N_R), D_0 \oplus N_R]$ ~> $T \mid\equiv R \xleftrightarrow{BID} T$, $T \mid\equiv DB \xleftrightarrow{Kt1,Kt2} T$

First the extensions to the message, $T \mid\equiv R \xleftrightarrow{BID} T$, $T \mid\equiv DB \xleftrightarrow{Kt1,Kt2} T$, are valid because they hold when the message is sent as is evident from the initial assumptions. Applying T1, P1, P4, P5, A5, F1 and F10. The rules F1 and F10 show that the message is fresh because R believes freshness of N_R. Since R can produce the hash output of BID and N_R, P4 and P5 reveal that R possesses the message contents, namely temporary ID index, and public information.

3. $R \triangleleft [H_{Kt2}(ID \oplus Kt1) \oplus H_{BID}(N_R), D_0 \oplus N_R]$

4. $R \ni [H_{Kt2}(ID \oplus Kt1) \oplus H_{BID}(N_R), D_0 \oplus N_R]$

5. $R|\equiv \#[H_{Kt2}(ID \oplus Kt1) \oplus H_{BID}(N_R), D_0 \oplus N_R]$

6. $R \ni [H_{Kt2}(ID \oplus Kt1), D_0]$

Message 3：$DB \triangleleft * [RID, \{H_{Kt2}(ID \oplus Kt1), N_R\}_{kr}] \sim> R|\equiv DB \overset{Kr}{\longleftrightarrow} R$

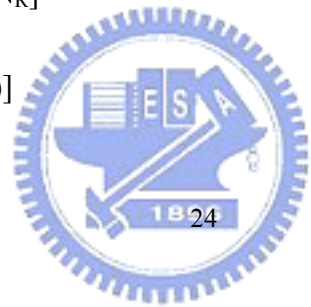
First the extension to the message, $R|\equiv DB \overset{Kr}{\longleftrightarrow} R$, is valid because it holds when the message is sent as is evident from the initial assumptions. Applying T1, T3, P1, R5, I1, I3, A9, A11,A15,A16 and J2.By T1,T3 and P1,DB finds out key Kr in accordance with RID and then possesses the components, namely temporary ID index, random nonce. Now DB checks if $H_{Kt2}(ID \oplus Kt1)$ exists. The rules F1, I3 and J2 give us $DB|\equiv T \sim \# [H_{Kt2}(ID \oplus Kt1)]$ and $DB|\equiv T|\equiv DB \overset{Kt1,Kt2}{\longleftrightarrow} T$ which mean DB believes that T sent the temporary ID index and DB believes that T also believes in the shared secrets between them.I1 and J2 confirm that DB believes R possesses Kr and DB also believes that R believes in the shared key Kr between them.

7. $DB \triangleleft [RID, \{H_{Kt2}(ID \oplus Kt1), N_R\}_{kr}]$

8. $DB \ni [RID, \{H_{Kt2}(ID \oplus Kt1), N_R\}_{kr}]$

9. $DB \ni [H_{Kt2}(ID \oplus Kt1), N_R]$

10. $DB|\equiv \phi [H_{Kt2}(ID \oplus Kt1)]$



11. $DB|\equiv \#[H_{Kt2}(ID \oplus Kt1)]$

12. $DB|\equiv T\sim\#[H_{Kt2}(ID \oplus Kt1)]$ (G1)

13. $DB|\equiv T|\equiv DB \xleftrightarrow{Kt1,Kt2} T$ (G2)

14. $DB|\equiv R \ni Kr$

15. $DB|\equiv R|\equiv DB \xleftrightarrow{Kr} R$ (G4)

Message 4：$R^{\triangleleft} * \{N_R, D_L\}_{kr} \sim> DB|\equiv DB \xleftrightarrow{Kr} R$

$DB|\equiv DB \xleftrightarrow{Kr} R$ is valid because it is an assumption. Applying T1, T3, P1, R1, I1 and J2 to show that R obtains user's data D_L and also believes the message is fresh and really sent by DB.

---

16. $R^{\triangleleft} \{N_R, D_L\}_{kr}$

17. $R \ni (N_R, D_L)$

18. $R|\equiv \phi(N_R, D_L)$

19. $R|\equiv \#(N_R, D_L)$

20. $R|\equiv DB \ni Kr$

21. $R|\equiv DB|\equiv DB \xleftrightarrow{Kr} R$ (G5)

In update phase for Kt1/Kt2, there are five messages transmitted between three principles.

Msg1 $R \rightarrow T$: $T \triangleleft * N_R$

Msg2 $T \rightarrow R$: $R \triangleleft * [H_{Kt2}(ID \oplus Kt1) \oplus H_{BID}(N_R), D_0 \oplus N_R]$

Msg3 $R \rightarrow DB$: $DB \triangleleft * [RID, \{H_{Kt2}(ID \oplus Kt1), N_R\}_{Kr}]$

Msg4 $BD \rightarrow R$: $R \triangleleft * \{N_R, D_L, H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1'), Kt1' \oplus Kt1 \oplus Kt2\}_{Kr}$

Msg5 $R \rightarrow T$: $T \triangleleft * [H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1'), Kt1' \oplus Kt1 \oplus Kt2]$

The first three messages are equal to those in communication phase. For the analyses of these messages, refer to prior analysis (1). The analyses of rest messages are described below.

Analysis (2)

Message 4：$R \triangleleft * \{N_R, D_L, H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1'), Kt1' \oplus Kt1 \oplus Kt2\}_{Kr}$

Applying T1, T3, P1, R1, F1 and I1, we can deduce from message 4: $R|\equiv DB|\sim (D_L)$, which indicates the database sever has ever sent D_L and the Data is not a replay message.

---

22. $R \triangleleft \{N_R, D_L, H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1'), Kt1' \oplus Kt1 \oplus Kt2\}_{Kr}$
23. $R \ni (N_R, D_L, H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1'), Kt1' \oplus Kt1 \oplus Kt2)$
24. $R|\equiv \#(N_R, D_L, H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1'), Kt1' \oplus Kt1 \oplus Kt2)$
25. $R|\equiv \phi(D_L, N_R, H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1'), Kt1' \oplus Kt1 \oplus Kt2)$
26. $R|\equiv DB|\sim (D_L)$

Message 5：$T \triangleleft * [F_1, H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1'), Kt1' \oplus Kt1 \oplus Kt2] \sim> DB|\equiv T \ni (Kt1, Kt2, ID)$,

$DB|\equiv DB \xleftrightarrow{Kt1'} T$

Applying T1, T3, P1, P5, R1, R6, F1, J1, J2 and I3. First, Tag possesses $Kt1$ and $Kt2$, so it can get $Kt1'$ and then checks the integrity of $H_{Kt2}(ID) \oplus N_R \oplus Kt1'$.If so, Tag will carry on update process. According to assumption A11 and the rule I3, Tags believes the database sever has ever sent the key-update information and it is not a replay message. Using J1 and J2, Tag believes $Kt1'/Kt2'$ is a valid key. When Tag receives the secret key, it will serve as database server.

---

27. $T \triangleleft [F_1, H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1'), Kt1' \oplus Kt1 \oplus Kt2]$
28. $T \ni (H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1'), Kt1' \oplus Kt1 \oplus Kt2)$
29. $T|\equiv \#(H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1'), Kt1' \oplus Kt1 \oplus Kt2)$
30. $T \ni Kt1'$
31. $T \ni \phi Kt1'$
32. $T|\equiv DB|\sim \#H_{Kt1}(H_{Kt2}(ID) \oplus N_R \oplus Kt1')$ (G8)
33. $T|\equiv DB|\sim \#(Kt1 \oplus Kt1' \oplus Kt2)$ (G9)
34. $T|\equiv DB|\equiv DB \xleftrightarrow{Kt1', Kt2'} T$ (G7)
35. $T|\equiv DB \xleftrightarrow{Kt1', Kt2'} T$ (G6)

In update phase for Kr, there are three messages transmitted among three principles. The security protocol can be idealized into GNY-logic:

Msg1 $R \rightarrow DB : DB \triangleleft * [RID, \{N_R, H_{BID}(N_R, Kr)\}_{Kr}]$

Msg2 $DB \rightarrow R : R \triangleleft * [N_R, \{(N_{DB}, H_{BID}(N_{DB}, Kr'), Kr \oplus BID \oplus Kr')\}_{Kr}]$

Msg3 $R \rightarrow DB : DB \triangleleft * [RID, \{N_{DB}\}_{Kr'}]$

Analysis (3)

Message 1：$DB \triangleleft * [RID, \{N_R, H_{BID}(N_R, Kr)\}_{Kr}] \sim> R|\equiv DB \overset{Kr}{\longleftrightarrow} R,\ R|\equiv DB \overset{BID}{\longleftrightarrow} R$

Applying T1, T3, P1, F1, I3 and J2. Database server believes reader has ever conveyed the key-update request, and he checks the correctness of $H_{BID}(N_R, Kr)$ .If the value is right, and then he believes reader believes that BID is a good key for both of them.

---

36. $DB \triangleleft [RID, \{N_R, H_{BID}(N_R, Kr)\}_{Kr}]$
37. $DB \ni [RID, N_R, H_{BID}(N_R, Kr)]$
38. $DB|\equiv \#[RID, N_R, H_{BID}(N_R, Kr)]$
39. $DB|\equiv R|\sim H_{BID}(N_R, Kr)$
40. $DB|\equiv R|\equiv DB \overset{BID}{\longleftrightarrow} R$ (G12)

Message 2：$R \triangleleft *[N_R, \{(N_{DB}, H_{BID}(N_{DB}, Kr'), Kr \oplus BID \oplus Kr')\}_{Kr}] \sim> DB|\equiv DB \xleftrightarrow{Kr'} R$,

$DB|\equiv DB \xleftrightarrow{BID} R$

Applying T1, T3, P1, P5, F1,R6, I3 ,J1 and J2. As reader gets $Kr \oplus BID \oplus Kr'$, and he owns Kr and BID, so he can obtain new key Kr' and verify it from $H_{BID}(N_{DB}, Kr')$.From I3, J2 and J1,we can prove our goal that reader believes database server believes that BID is a good key for both of them and reader believes that Kr' is a suitable key between reader and database server

---

41. $R \triangleleft [N_R, \{(N_{DB}, H_{BID}(N_{DB}, Kr'), Kr \oplus BID \oplus Kr')\}_{Kr}]$
42. $R \ni [(N_{DB}, H_{BID}(N_{DB}, Kr'), Kr \oplus BID \oplus Kr']$
43. $R|\equiv \#[(N_{DB}, H_{BID}(N_{DB}, Kr'), Kr \oplus BID \oplus Kr']$
44. $R \ni Kr'$
45. $R|\equiv \phi Kr'$
46. $R|\equiv DB|\sim H_{BID}(N_{DB}, Kr')$
47. $R|\equiv DB|\equiv DB \xleftrightarrow{BID} R$ (G13)
48. $R|\equiv DB|\equiv DB \xleftrightarrow{Kr'} R$
49. $R|\equiv DB \xleftrightarrow{Kr'} R$ (G10)

Message 3：$DB \triangleleft * [RID, \{ N_{DB} \}_{Kr'}] \sim> R|\equiv DB \xleftrightarrow{Kr'} R$

Applying T1, T3, P1, F1, I1 and J2. Database server decrypts the message by using the new key

Kr', and then he verifies N_{DB}. According to rules I1 and J2, it proves that database believes reader owns new key Kr' and also believes reader believes that Kr' is a good key for both of them

---

50. $DB \triangleleft [RID, \{ N_{DB} \}_{Kr'}]$

51. $DB \triangleleft N_{DB}$

52. $DB|\equiv R|\sim \{ N_{DB} \}_{Kr'}, DB|\equiv R \ni Kr'$ (G14)

53. $DB|\equiv R|\equiv DB \xleftrightarrow{Kr'} R$ (G11)

Chapter 6

Security and Privacy Analysis of Proposed Protocol

In this chapter, we evaluate our proposed mutual authentication protocol from RFID attacks point of view, and do the security analysis as same as Yang analyzes in [20].

1. *Eavesdropping attack*

   In the proposed protocol, data in the transmission process of R/T and R/DB are protected by one-way hashed function. Since hash is irreversible, even if the eavesdropper collects the outputs from T/R/DB, it is difficult to figure out the actual ID and encrypted data. In addition, data is different with every session a new nonce *N* is generated, raising the difficulty of brute force attack.

2. *Traceability attack*

   Tracking problem can be prevented employing the proposed protocol; we ensure that the output of a Tag is different, hence location privacy is guaranteed. Although [8] [20] [11] resolve tracking problem, they cause a cost in searching the database. Our protocol adopts a temporal ID index TID thus the searching is efficient and enhance the scalability of system.

3. *Man-in-the-Middle attack*

   As our protocol based on a mutual authentication, a man-in-the-middle attack is impossible. If R is an illegitimate reader, when sending message3 to DB, DB authenticates both T and R

by TID and Kr and then finds it a fake. Even if an illegitimate R gets the message4, it can't acquire the secret data since it encrypted by the secret key Kr between DB and R. Also in update phase, when updating key Kr, DB and T needs to authenticate each other; when updating key K_{t1}/K_{t2}, the illegitimate reader is detected from the message5 because of verification of DB and T.

4. *Counterfeit attack*

   The information of Tag, such as ID is embedded during the chip manufacturing, and the output of Tag is generated by XOR(⊕)、keyed hash and with random number N. Hence, it is difficult to copy the other one as the real tag. Even though some tags are compromised, the other tags have no relation with compromised information.

5. *Replay attack*

   Replay attacks cannot compromise the proposed protocol since every message is equipped with new nonce N every session. An adversary may pretend a reader to resend message3 to DB or message5 to T: in the former case, the adversary may get the message4, however the key Kr is changed after a period of time and he also can't decrypt it without Kr; in the latter case, it seems that it will cause the de-synchronous between DB and T, but it has no influence on the system because T still store the same key K_{t1}/K_{t2} as in the DB, and after a success key update process, the old message is inactive.

6. *Denial-of- Service (DoS)*

   In the update key phase, suppose DB server has updated the key yet the message5 is intercepted by an adversary, and therefore T doesn't get the update information. So in the next session, DB can't tell the tag, and the protocol is fail. To resolve this problem, our DB will

store the past secret information, and overwrite it until he receives the new secret information from T or it will still request T to update. The other kind of DoS attacks, the adversary resends RF signals to stuff up all the channels, we can't prevent this kind of attack and it is an issue to any RFID systems.

7. *Data confidentiality*

When T is queried by R, T sends the message $X(X=H_{kt2}(ID \oplus Kt1) \oplus H_{BID}(N_R))$ hiding his tag ID to R. With XOR and keyed hash functions, the tag ID is kept secret; and further, each new key hide with old key and another secret key, and T doesn't store any user privacy data, so the data confidentiality is guaranteed in our protocol.

8. *Tag anonymity*

As mentioned in traceability attack, we use the fresh nonce N every run and keys that are related to output of T also be updated and anonymous at short intervals. Although our tag is expected low-cost can't generate random numbers like [11] [15] [19], but we use the lightweight logic function to guarantee the tag anonymity. There is a possible scenario that an attacker keeps sending the same message1 to challenge T, from the proposed protocol, T responds the same output to R, and thus the attacker can track the bearer until tag has updated his keys. In order to prevent this kind of tracking, we assume that T keeps silent a certain time if he receives the same nonce N twice or more.

9. *Data integrity*

There is rewrite memory in our T, and tag stores the secret key Kt1/Kt2/BID in this kind of memory. To prevent the modification of data in such memory happen, thus before write data to memory, tag will verify the R with the authentication message. Moreover, we adopt the

data recovery mechanism on DB to against data loss by purposely DoS attacks.

10. *Mutual authentication*

Mutual authentication is designed for our protocol including T-to-DB authentication by message1 and DB-to-T authentication by message5. Besides, DB and R also authenticate each other by their shared secret key Kr.

11. *Forward security*

In our protocol, we guarantees the forward security that means privacy of messages sent today will be valid tomorrow [3]. Even if the attacker collects all transmitted messages from communication channels, they can't reason the past secret by present secret information because of the irreversible property of hash function and key update is completed within a legitimate time.

Table 6.1 shows the comparison of the security properties among different protocols made by Yang [20] and Pedro Peris-Lopez [21]. We add our protocol in the last column.

Table 6. 1 Comparison

Protocol	HLS [11]	EHLS [11]	HBVI [9]	MAP [20]	LMAP [21]	Our Protocol
User data confidentiality	×	△	△	○	○	○
Tag Anonymity	×	△	△	○	○	△
Data Integrity	△	△	○	○	△	○
Mutual Authentication	△	△	△	○	○	○
Reader Authentication	×	×	×	○	×	○
Forward security	△	△	○	○	○	○
Man-in-the-middle attack prevention	△	△	×	○	○	○
Replay attack prevention	△	△	○	○	○	○
Forgery Resistance	×	×	×	○	○	○

††Notation: ○ Satisfied △ Partially Satisfied × No Satisfied

Chapter 7

Performance Analysis

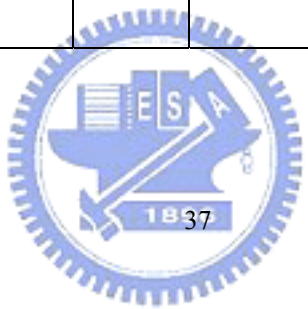
We analyze the overall performance of our proposed protocol in different aspects as Yang analyzes in [20] and determine their overheads: computation, storage, communication and, cost.

♦Computational Overhead

Owing to the restriction on RFID tags, T requires only a keyed hash operation and a basic bitwise XOR operation thus meets the claim for limited resource (only 2.5-5K logic gates available for security part [3]).Compare to previous works, as you can see in table 7.1, required computation of our protocol to find TID and related ID of T at DB is *O(1),* the computational cost of DB in our protocol is lower than others .This is because most hash-based solutions have a problem is that the load on DB side is proportional to the number of tags. We solve this problem by using TID and also achieve the security requirements. To protect secure data D_L transmitted on insecure channel between DB and T, we use the symmetric-key cryptology and both DB and R sides need to have capability of processing encryption and decryption function. Moreover, the velocity of symmetric encryption is very fast thus the proposed protocol is fast and suitable for numerous tags. Table 7.1 shows the comparison of computational for one session with previous works.

Table 7. 1 Comparison of Computational

Protocol	Entity	HLS [11]	EHLS [11]	HBVI [9]	MAP [20]	LMAP [21]	Our protocol	
							Communication Phase	Key update phase
No. of Hash Operation	T	1	2	3	2	¬	¬	¬
	R	¬	¬	¬	¬	¬	¬	¬
	B	¬	N	3	2N	¬	¬	¬
No. of Keyed Hash Operation	T	¬	¬	¬	¬	¬	2	4
	R	¬	¬	¬	1	¬	1	1
	B	¬	¬	¬	1	¬	1	2
No. of RNG Operation	T	¬	1	¬	¬	¬	¬	¬
	R	¬	¬	¬	1	¬	1	1
	B	¬	¬	1	¬	¬	¬	¬
No. of Basic Operation	T	¬	¬	¬	4	19	3	5
	R+B	¬	¬	¬	2(N+1)	21	1	4
No. of Encryption	R	¬	¬	¬	¬	¬	1	1
	B	¬	¬	¬	1	¬	1	1
No. of Decryption	R	¬	¬	¬	1	¬	1	1
	B	¬	¬	¬	¬	¬	1	1
Searching efficiency		*O(1)*	*O(N)*	*O(1)*	*O(N)*	*O(1)*	*O(1)*	*O(1)*



††Notation: N (Number of tags)

♦ Storage Overhead

We assume each component is L bits, and Table 7.2 shows the non-volatile memory comparison with previous works. In our protocol, T needs to store ID, BID, K_{t1}, and K_{t2}, so the needed memory is 4L bits. Thus, it is reasonable for implementing.

Table 7. 2 The Required Memory Size

Protocol	Entity	HLS [11]	EHLS [11]	HBVI [9]	MAP [20]	LMAP [21]	Our Protocol
Non-volatile memory size	T	2L	1L	4L	3L	6L	4L
	R	¬	¬	¬	L	¬	3L
	B	3L	1L	11L	9L	6L	7L

♦ Communication Overhead

The proposed protocol realizes mutual authentication between T/DB and R/DB. In general, it takes only four rounds through one session and needs one additional round while key-update. Normally, the power consuming of whole RFID system is partially relative to amount of messages. Compare with Yang's [20], they update the secret key every session, if there are plenty of tags and it will be a heavy load for whole system. We recommend reducing the frequency of updating tags and formalize the reasonable frequency for diverse RFID systems. Suppose the key update frequency of K_{t1} is higher than K_{t2}, K_{t1} is updated every n sessions and K_{t2} is updated every m sessions (m>n). After m sessions, there are 5m messages transmitted through Yang's, and there are only 4m+(m/n)+1 (if m=kn, k=1,2,3…,it will be

4m+(m/n)) messages through our protocol. An example is shown in Table 7.3.

Table 7. 3 Quantity of Transmitted Message

(n,m)	No. of rounds in m sessions		Reducing rate
	MAP[20]	Our protocol	
(2,6)	30	27	10%
(3,10)	50	44	12%
(4,17)	85	73	14%

We also hire the data recovery mechanism like [4], so the synchronization problem between Tag and Database can be resolved effectively.

♦ Cost Overhead

As stated by Ohkubo [3], they claimed that the number of gates available for security of passive tag cannot exceed 2.5 to 5 K gates. For our protocol, Tag only uses a one way hash function and XOR function, so it fits the capabilities of low-cost RFID.

Chapter 8

Future Work

There are two additional issues can be continued for discussion:

i. Key-update frequency

The previous works mentioned before update their secret key every round to protect user's privacy. We consider that the load of database server is quite heavy in case of updating every time. In our protocol, it doesn't need to update key every round but can still keep user's privacy. It is an issue to define key-update frequency and it changes by all kinds of application. For example, we assume all lengths of secret keys in our protocol are 32 bits and attackers possess a computer with CPU 1GHz. As attackers want to work out the secret keys, they can only use brute-force to break our system, and the computation is $2^{32}\times 2^{32}$. Time they need to spend is $2^{32*2-30}=2^{34}$, in theory, it is an acceptable result. Furthermore, attackers have more than one computer, if they have a thousand computers, they can break the system in half hour ($2^{32*2}/(2^{30}\times 1000)= 2^{24} \approx$ 32 minutes), and so the key needs to be updated in every half hour. This is a way to define the key-update frequency but needs of further studies.

ii. Hardware implement

We haven't implemented our protocol in practice, and we consider implementing it on

FPGA. From a practical of view, the most important question is how reduces a tag cost .The number of logic gates for implementing the tag can be a criterion, and should not exceed 10k gates and therefore accomplish the purpose desired.

Chapter 9

Conclusion

In this thesis, we presented an efficient and lightweight protocol that can resist most attacks. Our protocol addresses the problems of other existing protocols by using simple cryptographic tools. The proposed protocol has applied the symmetric cryptosystem between the reader and the database server and hash function within tag side. The resultant protocol achieves mutual authentication and key-update, also manages the data with different levels. It is efficient in that it only requires four messages to complete the protocol in communication phase and database searching. It is also flexible to do key-update in accordance with diverse requirements. We prove the correctness of our protocol by GNY model logic, and it shows that the protocol is secure and achieves the authentication.

References

[1] A. Juels and R. Pappu, “Squealing Euros: Privacy Protection in RFID-Enabled Banknotes,” Financial Cryptography, R. Wright, Ed., Springer-Verlag, 2003.

[2] M. Feldhofer, “An Authentication Protocol in a Security Layer for RFID Smart Tags,” IEEE MELECON, May, 2004.

[3] M. Ohkubo, K. Suzuki and S. Kinoshita, “Cryptographic Approach to “Privacy-Friendly” Tags,” RFID Privacy Workshop @ MIT, 2003.

[4] I. Vajda and L. Buttyan, “Lightweight Authentication Protocols for Low-Cost RFID Tags,” Security in Ubiquitous Computing, 2003.

[5] T. Dimitriou, “ A lightweight rfid protocol to protect against traceability and cloning attacks,” IEEE SECURECOMM, 2005.

[6] T. Dimitriou, “A secure and efficient rfid protocol that can make big brother obsolete,” International Conference on Pervasive Computing and Communications, 2006.

[7]K. Rhee, J. Kwak, S. Kim and D. Won, “Challenge-Response Based RFID Authentication Protocol,” International Conference on Security in Pervasive Computing, 2005.

[8] Z. Luo, T. Chan and J.S. Li, “A lightweight mutual authentication protocol for RFID networks,” e-Business Engineering IEEE International Conference, pp.620-625, October, 2005.

[9] D. Henrici and P. M¨uller , “Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers,” IEEE International Workshop on Pervasive Computing and Communication Security, pp. 149-153, March, 2004.

[10] S. Lee, H. Lee, T. Asano and K. Kim, “Enhanced RFID Mutual Authentication Scheme based on Synchronized Secret Information,” Auto-ID Labs White Paper WP-HARDWARE-032.

[11] S. Weis, S. Sarma, R. Rivest and D. Engels, “Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems,” Security in Pervasive Computing, March, 2003.

[12] X. Gao, Z. Xiang and H. Wang, “An Approach to Security and Privacy of RFID System for Supply Chain,” Proceedings of the IEEE International Conference on E-Commerce Technology for Dynamic E-Business 2004.

[13] J. Ma, A. Nakamura and R. Huang, “A Random ID Update Scheme to Protect Location Privacy in RFID-based Student Administration Systems,” in IEEE CS Proceedings of the 16th DEXA/NBiS, Copenhagen, Denmark, pp. 67-71,August, 2005.

[14] A. Juels, “Minimalist Cryptography for Low-Cost RFID Tags,” Security in Communication Networks, 2004.

[15] Y. K. Lee and I. Verbauwhede. “Secure and Low-cost RFID Authentication Protocols,” Adaptive Wireless Networks, November, 2005.

[16] C. H. Lim and T. Kwon, “Strong and Robust RFID Authentication Enabling Perfect Ownership Transfer, ” Conference on Information and Communications Security, 2006.

[17] G. Ateniese, J. Camenisch and B. de Madeiros, “Untraceable RFID tags via insubvertible encryption,” Proc. 12th ACM Conf. Comput. Commun. Security, 2005.

[18]A. Juels, D. Molnar and D. Wanger, “Security and privacy issues in e-passports,” IEEE SecureComm, September, 2005.

[19] L. Zhang, H. Zhou, R. Kong and F.Yang, “An Improved Approach to Security and Privacy of RFID, ” Proceedings of the IEEE International Conference on Wireless Communications, Networking and Mobile Computing, 2005.

[20] J. Yang, K. Ren and K. Kim, “Security and Privacy on Authentication Protocol for Low-Cost RFID,” Symposium on Cryptography and Information Security, 2005.

[21] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador and A. Ribagorda, “LMAP: A Real Lightweight Mutual Authentication Protocol for Low-cost RFID tags,” Workshop on RFID Security, July, 2006.

[22] L. Gong , R. Needham and R. Yahalom, “Reasoning about belief in cryptographic protocols. ” In: Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy, 1990.