情報保護學會論文誌
第 21 卷 第 5 號, 2011. 10

# 서버와 태그 비동기시에도 효율적으로 검색이 가능한 해시기반 RFID 인증 프로토콜*

권 혜 진,[1†] 김 해 문,[1] 정 선 영,[2] 김 순 자[1‡]
[1]경북대학교, [2]경운대학교

# Hash-based Authentication Protocol for RFID Applicable to Desynchronization between the Server and Tag with efficient searching method*

Hyejin Kwon,[1†] Haemun Kim,[1] Seonyeong Jeong,[2] Soonja Kim[1‡]
[1]Kyungpook National University, [2]Kyungwoon University

## 요 약

RFID 시스템은 여러 장점을 가지며 다양한 곳에서 이용되고 있다. 그러나 현재의 RFID 시스템은 무선통신과 저가 태그의 사용으로 인해 도청, 재전송 공격, 메시지 탈취, 태그에 대한 물리적 공격에 취약하다. 이런 공격들은 태그와 리더 사칭, 태그 무능화를 통한 서비스 거부공격, 태그 소지자의 위치 추적등을 유발한다. 이런 취약점을 해결하기 위해 다수의 RFID 인증 프로토콜이 제안되었다. 특히 Weis, Sarma, Rivest, Engels이 해시 기반 RFID 인증 프로토콜을 제안한 이래로, 많은 해시 기반 RFID 인증 프로토콜이 제안되었고 최근의 해시 기반 프로토콜은 보안성과 만족할 만한 수준의 프라이버시를 제공한다. 그러나 보안성과 만족할 만한 수준의 프라이버시를 제공하는 동시에 서버에서의 태그 인식 시간을 줄이는 것은 여전히 풀리지 않는 문제다. 이에 우리는 태그가 이전 통신 상태에 따라 응답 메시지를 생성하는 프로토콜을 제안한다. 그 결과 보안성과 프라이버시를 보장하는 동시에 서버에서 합리적인 시간안에 태그를 인식할 수 있다. 구체적으로 제안하는 프로토콜은 세션키와 롱텀 고정키를 모두 사용해서 이전 세션이 비정상 종료 된 경우에 비슷한 수준의 안전성을 제공하는 기존의 연구결과에 비해 태그 인식시간이 50%로 절감한 효과를 제공한다.

## ABSTRACT

The RFID system provides undeniable advantages so that it is used for various applications. However recent RFID system is vulnerable to some attacks as eavesdropping, replay attack, message hijacking, and tag tampering, because the messages are transmitted through the wireless channel and the tags are cheap. Above attacks cause the tag and reader impersonation, denial of service by invalidating tag, and the location tracking concerning bearer of tags. A lot of RFID authentication protocol has been proposed to solve the vulnerability. Since Weis, Sarma, Rivest, and Engels proposed the hash-based RFID authentication protocol, many researchers have improved hash-based authentication protocol and recent hash-based authentication protocols provide security and desirable privacy. However, it remains open problem to reduce the tag identification time as long as privacy and security are still guaranteed. Here we propose a new protocol in which the tags generate the message depending on the state of previous communitions between tag and reader. In consequence, our protocol allows a server to identify a tag in a reasonable amount of time while ensuring security and privacy. To be specific, we reduced the time for the server to identify a tag when the last session finished abnormally by at least 50% compared with other hash-based schemes that ensure levels of security and privacy similar to ours.

**Keywords**: RFID, Mutual authentication, security, privacy

접수일(2010년 10월 29일), 수정일(2011년 4월 19일), 게재확정일(2011년 9월 30일)

* This Research was supported by Kyungpook National University Research Fund, 2009

† 주저자, hjkwon@knu.ac.kr
‡ 교신저자, snjkim@knu.ac.kr

## I. Introduction

The radio frequency identification (RFID) system identifies a tagged item through a wireless channel. Currently, it is used for various purposes and provides advantages in various applications, including supply chain management, access control systems, library systems, inventory control, and identification and tracking of livestock and pets [1]-[3]. Thus, RFID technology is expected to play an important role in a ubiquitous environment. Therefore, tagged items will become widely used in the near future.

However, like other telecommunication systems, the RFID system has security problems. For example, it is open to impersonation and denial of service (DoS) [4]-[6]. Moreover, several RFID systems are not privacy-friendly because some tags emit their naive, unique identifier which might represent the character of the items, although the bearer of the tag may not be aware of the lack of privacy. Thus, an adversary who eavesdrops on communications between tags and readers can learn what the tag holder owns and where he stayed which is an invasion of privacy [5]-[7].

In order for the system to guarantee security and privacy against adversarial threats, numerous schemes have been proposed. The schemes are based on the public key cryptosystem, the symmetric key cryptosystem, the hash operation, or the bit operation [8]. In this paper, we focus on authentication schemes based on the hash operation, rather than other operations. Though Weis et al. introduced a hash function in RFID authentication, their scheme cannot ensure both security and privacy. However, their proposal has inspired many researchers to study methods that guarantee security and privacy using a hash function.

Among the various hash-based schemes, hash chain protocol [7] has inspired studies of the RFID server scalability because of its poor scalabilty; the server identifies a tag after executing the hash operation $ni$ times, where $n$ and $i$ are the number of tags and the number of renewed IDs, respectively (the next section provides more details). As a result, the protocol which seemed efficient as well as secure and private was proposed in 2006 [6]. In that protocol, a server can usually identify a tag by hashing three times. However, a server must operate the hash function $n+3$ times on average to identify a tag if the last session is incomplete.

Thus, we study the threats to an RFID system and the RFID protocol requirements in detail. We review the RFID hash-based authentication protocols to achieve security, privacy, and efficiency in section II. Next, analyses of existing protocols are presented. Then, we propose our new efficient protocol, which is secure and private, in Section III. Finally, we present our conclusions after analyzing the security, privacy, and performance of our protocol.

## II. The requirements for RFID system

To identify a tag successfully, an RFID system must meet some requirements. These requirements protect the RFID system from adversaries so that the server can identify a tag smoothly. Here, the adversary is characterized by his goals of and his tools for attack [3]. An adversary attacks in order to achieve his attack goal using his tools for attack. In this section, we study the requirements for an RFID system to successfully fend off an adversary. Then, we investigate the existing hash-based authentication protocols to determine whether or not they meet the requirements.

### 2.1 Requirements of RFID protocol

The requirements for an RFID system have been presented and the general consensus is that the RFID protocol must achieve security, privacy, and efficiency [3], [6], [7], [9]. The details of each requirements are presented below.

#### 2.1.1 Security

An RFID system is secure if it perfectly frustrates the goals of an adversary, such as impersonating a tag and DoS. An adversary whose goal is tag impersonation may carry out attacks, including eavesdropping, message hijacking, replay attack. Since the tag communicates with the reader over the wireless channel, above attacks are relatively easy. Tag responds automatically when reader asks a query to identify a tag. If tag responds with a its naive identifier, an adversary can obtain information about tag through eavesdropping. It causes an invasion of privacy and other attacks including replay attack against security. An adversary sends eavesdropped message to a reader(respectively, a tag), then an adversary can impersonate a tag(respectively, a reader). Above attacks threaten the security of RFID system, Hence we design a secure RFID protocol, with an emphasis on preventing above attacks. However we do not treat the interleaving attacks as an adverserial tool for impersonating a tag. An adversary who use interleaving attacks in order to impersonate a legal entity takes part in the current session, and he uses the combined messages obtained from communicating with the legal tag and reader simultaneously [4]. Furthermore, in this paper we exclude attacks that send excessive queries to a tag and server in order to cause DoS.

#### 2.1.2 Privacy

Because the tags are attached to personal items that represent the individuality of the tag holder, noting information about the tag is the same as recognizing the private information of tag holder, which may be a severe invasion of privacy. An RFID system guarantees privacy if the system completely thwarts the adversarial goal of obtaining personal information about a tag holder. At this point, personal information is defined as the information about the belongings of the tag holder and where the tag holder has stayed. When the former is revealed, the item privacy is infringed(or simply information leakage is happened). Moreover, location privacy is invaded if the latter is revealed. There are two types of location privacy, anonymity and forward privacy, according to the adversary's ability to corrupt the tag. If an attacker cannot trace the previous location of a tag holder when he does not corrupt the tag, then the system ensures anonymity. If he cannot trace the previous location even though the tag is compromised, the system ensures forward privacy.

#### 2.1.3 Efficiency

If the server's process for identification of certain protocol has a high time complexity, the protocol is unlikely to be acceptable for a real system, even if the protocol ensures strong security and privacy. Thus, an RFID protocol must also be efficient. In particular, hashbased RFID protocols in which tags emit a fresh message every session have trouble identifying a tag when the previous session between that tag and the server ends abnormally, i.e., when the communication message was lost. To increase efficiency, some methods have been pro-

posed: utilizing distributed servers or adopting a group ID [4], [10]-[14]. The former, delegation, can cause serious privacy infringement when any one distributed server is compromised. Molnar et al. represented delegation as too coarse-grained to defeat such a problem [15]. The latter decreases the server workload, but it is not a fundamental solution; rather, it is an assistant solution that can be applied to any protocol.

### 2.2 Review of existing protocols

As cited Subsection 2.2., it is undesirable for a tag to emit its naive ID in response to a reader query because it allows someone who has an RFID reader to easily obtain private information regarding the tag bearer, i.e., the tag holders list of possessions and location information. Additionally, forwarding a naive ID can allow impersonation of a legal tag.

To solve these problems, Weis et al. proposed hash-locking and randomized hash-locking schemes [1]. In the former, the tag responds to a query with $h(ID)$, where $h$ is a hash function, while in the latter, the tag responds to a query with $h(ID\|r)$. However, this does not completely solve the problem because the tag sends its $ID$ through an insecure channel in the last flow.

The identification protocol called hash chain protocol(OSK) was proposed by Ohkubo, Suzuki, and Kinoshita [7]. In this scheme, a tag updates its $ID$ every session using the hash function $h$, while the server does not. The tag transmits $g(ID)$ where $g$ is a different hash function. As a result, this protocol can ensure forward privacy, as well as item privacy and indistinguishability. However, it has trouble with security and scalability. Because the server does not renew each tag $ID$ and the reader does not generate a random number, the adversary can easily impersonate a tag by resending a $g(ID)$ that was used once before. Moreover the server can identify a tag after performing hash function $ni$ times on average, where $n$ is the number of tags and $i$ is the number of tag $ID$ renewals. This causes server overload. The server for this protocol cannot identify a tag in a reasonable amount of time in a real RFID system.

The mutual authentication protocol, which is more scalable than OSK, was proposed by Henrici and Muller [16]. Unlike OSK, in this protocol(HM), the server also updates tag's $ID$ so that it only requires three hashes to authenticate a tag. However, an adversary can still impersonate a tag. This impersonation results from the fact that the reader does not generate random number. An attack may proceed as follows: the adversary blocks the servers authentication message and collects the tag responses; then, he forwards the collected message whenever the reader sends a query; thus, he impersonates a genuine tag. Moreover, this protocol cannot guarantee anonymity and cannot defend the system against DoS. An adversary can hijack the server's legal message $\{r, h(r\oplus tid\oplus ID)\}$, and he alters it into $\{0, h(0\oplus tid\oplus ID)\}$. Because $h(0\oplus tid\oplus ID)$ is exactly the same as $h(tid\oplus ID)$, he cans alter the message. Then the tag updates the $ID$ and $LST$(last successful transaction number) while the server does not. Because the tag updates the $LST$, the server will never authenticate the tag. Therefore, the tag will never renew its $ID$. As a result, the adversary can cause DoS and identify that tag because it always emits same $h(ID)$ [3].

Lee et al. proposed an authentication protocol(LHLL) that reduced the length of transmitted messages and the number of calculation of both tag and server [17]. By

using the extraction function, lengths of authentication messages of both the tag and server can be reduced by half. In detail, both the tag and server generate authentication messages. The tag sends the left half of its message using the extraction function. After receiving the tags message, the server verifies it. If the message is correct, the server forwards sends the right half of the already computed message without additional computation. Thus, they not only reduce the length of message but also the number of calculations. However, they still cannot solve the location tracking problem because the tag always emits the identifier hash value, even if the tag it was not updated.

Ha et al. proposed an authentication pro-

| *System* $\{\cdots, (ID, HID, PID), \cdots\}$ | | *Tag* $(k_1, k_2, c)$ |
|---|---|---|
| Pick $r \in \{0,1\}^l$ | $Query, r$ → | |
| | | Pick $t \in \{0,1\}^{l/2}$<br>If $sync=0$ : $P=h(ID)$ |
| Find a $(ID, HID, IID)$ :<br>If $P=HID$ : $PID \leftarrow ID$<br>Else if $P=h(ID\|t)$, $PID \leftarrow ID$<br>Else if $P=h(PID\|t)$, $PID \leftarrow PID$<br>Else, halt | ← $P, L(Q), t$ | Else : $P=h(ID\|t)$<br>$Q=h(ID\|t\|r)$<br>$sync \leftarrow 1$ |
| $Q'=h(PID\|t\|r)$<br>If $L(Q')=L(Q)$,<br>$\{ID=h(PID\|r),\ HID=h(ID)\}$ | $R(Q')$ → | If $R(Q)=R(Q')$,<br>$\{ID=h(ID\|r),\ sync \leftarrow 0\}$ |

〔Figure 1〕 HHMB〔6〕

| *System* $\{\cdots, (H, k_1, k'_1, k_2), \cdots\}$ | | *Tag* $(k_1, k_2, c)$ |
|---|---|---|
| Pick $r \in \{0,1\}^l$ | $Query, r$ → | |
| | | Pick $t \in \{0,1\}^{l/2}$<br>If $c=0$ : $m=m_1\|m_2$, where<br>$m_1=h(k_1)$,<br>$m_2=L(h(r\|k_1\|k_2))$. |
| Find a $(H, k_1, k'_1, k_2)$ :<br>If $H=m_1$ :<br>$\{k'_1 \leftarrow k_1$, verify $m_2$, $v=(h(r\|k'_1\|k_2)\}$<br>Else if $L(h(k_2\|t\|r)))=m_a$,<br>{ if $L(h(k_1\|t\|r)))=m_b$,<br>$k'_1 \leftarrow k_1$, $v=h(k'_1\|t\|r)$<br>else if $L(h(k'_1\|t\|r)))=m_b$,<br>$k'_1 \leftarrow k'_1$, $v=h(k'_1\|t\|r)$<br>else, find other $k_2$ }<br>Else, halt | ← $m$ | Else : $m=m_a\|m_b\|t$, where<br>$m_a=L(h(k_2\|t\|r))$,<br>$m_b=L(h(k_1\|t\|r))$.<br>$c \leftarrow 1$ |
| | $R(v)$ → | |
| When it finds match,<br>$\{k_1 \leftarrow h(k'_1\|r),\ H \leftarrow h(k_1)\}$ | | If $R(v)=R(h(k_1\|t\|r))$<br>: $k_1 \leftarrow h(k_1\|r)$, $c \leftarrow 0$ |

〔Figure 2〕 The proposed protocol

tocol(HHMB) in which the tag generates a message according to the state of the previous session [6]. The tag generates message by just hashing *ID*, if tag updated its *ID* normally in last session as shown in the [Figure 1]. Otherwise, tag generates message using not *ID* but also random number $t$. In spite of tag generates according to the state of previous session, their protocol can not assure anonymity when adversary blocked the previous message [9]. After adversary intercepted the previous message of either tag or server, the adversary receives the random number $r$ from reader by impersonating a tag. Then he sends the $r$ to tag and receives the respond, $h(ID\|t)$ and left half of $h(ID\|t\|r)$. He forwards the message to legal reader after exchanging $h(ID\|t)$ for $h(ID)$. Provided that he is authenticated, he can link the tag. Furthermore, this protocol has trouble with scalability: the server is able to search the tag after hashing $n+3$ times on average, when the previous session finishes abnormally. Here, we present a novel mutual authentication protocol. We focus on reducing the search time as well as enhancing security and privacy.

## III. Prpposed protocol

Before describing our protocol in detail, we will provide some assumptions and notations. Generally, an RFID system is composed of three entities: the server, the readers, and the tags. The communication channel between the server and reader is wired so that this channel is often considered to be secure. Thus, we consider these two entities as one unit, system.

In our protocol, every tag keeps two secret keys: an $l$-bit session key $k_1$ and long-term key $k_2$. Moreover, each tag has a one-bit counter value $c$, that is, $c$ is either 0 or 1. Each tag can generate $l/2$-bit random number, denoted as $t$. It also can carry out the hash function $h: 2^* \rightarrow 2^l$ and the extraction function $L$(resp. $R$): $2^m \rightarrow 2^{m/2}$ such that extracts the left(resp. right) half of the input. Each reader can generate the $l$-bit random number, denoted as $r$.

We assume that the $n$ tags are enrolled in a server. The server has a list of $H, k_1, k'_1, k_2$, and the *information* for each tag. $H$ represents the hash value of $k_1$, and $k_2$ is exactly the same for that tag. However, $k_1$ of server is not always the same as the tag's $k_1$. The server's $k_1$ equals to $k_1$ of tag when the last session finished normally so that both tag and server updated $k_1$ or when the tag message was lost in the last session and neither the server nor the tag renewed their $k_1$. However, the $k_1$ values differ when the server message was lost in the last session. In this case, the tag's session key $k_1$ is same to $k'_1$ of server. This discordance results from the fact that the server renewed the session key $k_1$ before the tag does.

Now, we introduce our mutual authentication protocol.

1. The reader broadcasts a *Query* with its $l$ -bit random number $r$ to the tag.
2. After receiving the *Query*, the tag generates an $l/2$-bit random number $t$ and a message $m$ according to the counter $c$.
   a. If $c=0$, then the tag computes $m$ as $m_1 \| m_2$, where $m_1$ is $h(k_1)$ and $m_2$ is $L(h(r \| k_1 \| k_2))$, that is, $m_2$ is the left half of $h(r \| k_1 \| k_2)$.
   b. Else, the tag generates $m$ as $m_a \| m_b \| t$, where $m_a$ and $m_b$ are, respectively, $L(h(k_2 \| t \| r))$, $L(h(k_1 \| t \| r))$.

   After sending $m$, the tag sets its counter value $c$ as 1.
3. The reader forwards the message transmitted from the tag with $r$.
4. After receiving a message from the

reader, the server has to find an appropriate quadruplet $(H, k_1, k'_1, k_2)$ in the list.

· It finds a quadruplet such that $H$ equals to two-thirds of $m$.

a. If there is such a quadruplet, it generates a verification message $v = h(r \| k_1 \| k_2)$ and confirms whether or not $L(v)$ equals the rightmost one-third of $m$, $m_2 = L(h(r \| k_1 \| k_2))$. Provided that the two values are same, server renews $k'_1$ as $k_1$.

b. Else, the server computes $L(h(k_2 \| t \| r)))$ for each $k_2$ and compares it with one-third of the received message, $m_a$, until it matches. If there exists a quadruplet containing such $k_2$, the server must determine which value between $k_1$ and $k'_1$ equals to $k_1$ of tag. The details are described below.

i. Let $v$ be $h(k_1 \| t \| r)$. If the midmost one-third bits of $m$, known as $m_b$, are the same as the left half of $v$, $L(v)$, then the second message $m$ was intercepted in the last session; thus, the server and tag did not update their $k_1$. As a result, $k_1$ of both the tag and server are the same. In this case, the server renews $k'_1$ as $k_1$.

ii. Otherwise, the server switches $v$ to $h(k'_1 \| t \| r)$. If the midmost one-third bits of $m$ equals to $L(v)$, then it implies that the final message $R(v)$ in the previous session was lost; thus, the tag did not update its $k_1$ while server did. In this case, the tag's secret $k_1$ is same to the previous secret $k'_1$ of server; therefore, the server does not update $k'_1$, unlike in the above case.

iii. Provided that neither $k_1$ nor $k'_1$ matches the $k_1$ of tag, the server repeats searching the appropriate row.

· The server finishes the session if it cannot find a quadruplet in the list that meets the criteria. Otherwise, it sends reader $R(v)$ and updates $k_1$ as $h(k'_1 \| r)$ and $H$ as $h(k_1)$.

5. The reader forwards $R(v)$.

6. After receiving $R(v)$, the tag confirms its correctness. If $R(v)$ is correct, the tag also renews $k_1$ and sets the counter value $c$ to 0.

## IV. Analysis

Depending on Subsection 2.1, we will analyze the security and privacy informally. The notations used in this section are the same as those used in Section Ⅲ. Without loss of generality, we assume that both the reader and tag generate the same random numbers in a row with negligible probability, and $2^{-l/2}$ is negligible.

### 4.1 Security analysis

#### 4.1.1 Denial of services

As cited in Subsection 2.1, we do not treat DoS caused by excessive queries. Because our protocol uses changeable key $k_1$, the server's key may not match that of the tag. In this section, we treat DoS resulting from this discordance between the server and tag keys. This discordance results from the interception of a transmitted message and impersonating a legal reader. We show that our protocol prevents the system from giving a DoS caused by differences in $k_1$.

4.1.1.1 By message interception

One way to accomplish DoS is for the opponent to intercept the messages transmitted through the wireless channel. For

〔Table 1〕 A comparison of the five protocols

| | OSK | HM | LHLL | HHMB | Proposed |
|---|---|---|---|---|---|
| Storage of tag | $l$ | $3l$ | $l$ | $l+1$ | $2l+1$ |
| Storage of server | $2ln$ | $10ln$ | $6ln$ | $3ln$ | $4ln$ |
| Length of tag's message | $l$ | $3l$ | $3l/2$ | $5l/2$ | $3l/2$ |
| Hash # of tag | 2 | 3 | 2 | 3 | 3 |
| Hash # of server(Normal) | $ni$ | 3 | 2 | 3 | 3 |
| Hash # of server(Abnormal) | - | 3 | - | $n+3$ | $n/2+4$ |
| Possible to recover to desync of ID | - | × | ○ | ○ | ○ |
| Secure against tag impersonating | × | × | ○ | ○ | ○ |
| Anonymity | ○ | × | × | × | ○ |
| Forward privacy | ○ | × | × | △ | △ |

※ $l$ is the length of *ID*. $n$ is the number of tags. $i$ is the number of tag *ID* renewals

protocols that employ a static state, this attack is futile. However, in a protocol that uses changeable tag states, message interception may be fatal because message interception causes desynchronization of the state between the server and tag. However, our protocol can resist message interception, even though it uses a variable secret key, $k_1$.

If we assume that the tag's message $m$ is intercepted, the secret key $k_1$ of the server and tag are still same. Moreover, the tag will send a message in the next session using the static secret key $k_2$, and $k_1$ that equals that of the server. Thus, server will be able to identify tag normally. Provided that the server message $v$ is intercepted. Then, the server updates $k_1$, while the tag does not. As a result, the $k_1$ of server and $k_1$ of tag are not equal. However, the server will be still able to authenticate the tag at the next session because the server keeps the tag's $k_1$ as $k'_1$.

4.1.1.2 By reader interception

In our protocol, impersonating a reader is not a goal in itself but is just a tool for causing a secret key mismatch that result in DoS. Spoofing a legal reader enables the adversary to renew the only tags secret arbitrarily so that the tag will be never authenticated. In order to imitate a legal reader, he must create a valid message $v$, either $R(h(r\|k'_1\|k_2))$ or $R(h(k'_1\|t\|r))$, without knowing $k'_1$. The probability of generating correct $v$ without the information concerning $k_1$ is at most $2^{-l/2}$, which is negligible.

#### 4.1.2 Tag impersonation

If someone pretends to hold a legal tag, he must generate a valid $m$. Namely, he must compute correct $h(k_1)\|L(h(r\|k_1\|k_2))$ or $L(h(k_2\|t\|r))\|L(h(k_1\|t\|r))$ when he knows the $r$ and $t$ with no information on $k_1$ and $k_2$. We assume that the adversary has collected the tag message. In this situation, he can be identified as a legitimate tag if a reader sends him a random number $r$ which is the same to that used in last session. We already assume that the reader creates the same random number in succession with negligible probability. Thus, he is only successful with negligible probability. If he does not wait for the same random number of reader, he must create $m$ randomly. In this case, the probability that he pretends to have a legal tag is at most $n2^{-l}$. The

probability of assuming a specific tag is at most $2^{-l}$. Because both probabilities are insignificant, we assert that our protocol is secure against tag impersonation.

### 4.2 Privacy analysis

#### 4.2.1 Information leakage

If we assume that an adversary wants to know what a tag holder possesses, he eavesdrops on the communication between the tag and reader. In our protocol, tags emit the hashed value of a mixture of random bits and the secret key of itself. Thus, it is difficult for the adversary to know who holds what, if he just collects the tag response. Furthermore, tags do not save their identifier, i.e., they keep two secrets that are independent of its identifier. Moreover, the tag information is only communicated through a secure channel between the server and reader.

#### 4.2.2 Location privacy

4.2.2.1 Anonymity

Because each tag generates a message by hashing its secrets and random numbers every session, the adversary cannot tell whether or not the messages are generated by the same tag. Though two messages are generated by the same tag, they are generated in a completely different way according to the counter value $c$ and random numbers $r,t$. Thus, an attack that invades the anonymity the previously proposed protocol [6] cannot invade the anonymity of our model.

4.2.2.2 Forward privacy

Our protocol provides restrictive forward privacy. Generally, the protocols in which the tags employ a static identifier cannot provide location privacy if the attacker compromises the tag. Because our protocol uses a static identifier, our protocol cannot provide forward privacy perfectly. That is, the attacker who compromised a certain tag and holds the entire transcript of previous communication can grasp the previous location information about the tag whenever the tag generates message using static key $k_2$. However the tags in our protocol employ not only static key $k_2$, but also variable key $k_1$. So the attacker cannot trace the tag seamlessly.

### 4.3 Performance analysis

In our protocol, tag authentication phase is composed of three steps; the identification step, the verification step, and the key renewal step. If the previous session ended completely, the server can identify a tag with no hash function. And the server verifies the tag identity with just one hash operation, and it performs hash operation twice for renewing the $k_1$ and $H$. Thus the server has to perform hash operation three times during the tag authentication phase usually. This is not too impressive because previous research [6], [16] has obtained the same result. However our protocol is superior when the previous session finished abnormally, that is, the communication between the entities was blocked. In this case, the server executes the hash function $n/2+4$ times on average, where $n$ is the number of tags contained in the server. As represented by [Table 1], this value is the half of HHMB.

## V. Conclusion

It is expected that RFID systems will be widely used in the near future. Thus, it is important to ensure that the system pre-

serves security, privacy, and scalability. Here, we used two secret keys, $k_1$ and $k_2$, for each tag; only $k_1$ is renewed each session, whereas $k_2$ is static. The server also keeps $k_1$, $k_2$, and $k_1$'s previous key $k'_1$ in order to recover from a desynchronization caused by loss of message if necessary. Thus, our protocol is secure against attacks to incite irrecoverable mismatch between tag and server keys.

Both the tag and reader generate messages using the secret keys and a newly picked random number; thus, the message changes irregularly every session, even though the tag does not update the secret key. Therefore, an adversary cannot impersonate both the tag and reader. In addition, he cannot violate tag anonymity. Unfortunately, our protocol provides limited forward privacy because of the static key $k_2$.

Moreover, we reduced the server workload by 50% when the previous communication finishes abnormally because we employed both variable and static secrets rather than one variable secret. As a result, our protocol ensures security against impersonation, a decrease in DoS, and tag holder privacy. Furthermore, the server can identify the tag in a reasonable amount of time, even if the messages from the previous session were lost.

## 참고문헌


[1] S. Weis, S. Sarma, R. Rivest, and D. Engels, "Security and privacy aspects of Low-Cost radio frequency identification systems," International Conference on Security in Pervasive Computing, pp. 454-469, March 2003.

[2] A. Juels and S. Weis, "Defining strong privacy for RFID," International Conference on Pervasive Computing and Communications, pp. 342-347, March 2007.

[3] G. Avoine, "Adversary model for radio frequency identification," Swiss Federal Institute of Technology (EPFL), Security and Cryptography Laboratory (LASEC), Lausanne, Switzerland, Technical Report LASEC-REPORT-2005-001, September 2005.

[4] M. Burmester, B. de Medeiros, and R. Motta, "Provably secure grouping-proofs for RFID tags," Proceeding of the 8th Smart Card Research and Advanced Applications, pp. 176-190, September 2008.

[5] K. Osaka, T. Takagi, K. Yamazaki, and O. Takahashi, "An efficient and secure RFID security method with ownership transfer," Computational Intelligence and Security, 2006 International Conference on, vol. 2, pp. 1090-1095, November 2006.

[6] J. Ha, J. Ha, S. Moon, and C. Boyd, "LRMAP: Lightweight and resynchronous mutual authentication protocol for RFID system," in ICUCT, pp. 80-89, December 2006.

[7] M. Ohkubo, K. Suzuki, and S. Kinoshita, "Cryptographic approach to Privacy Friendly Tags," in RFID Privacy Workshop, http://simson.net/ref/2004/rfidprivacy.us/2003/agenda.php. November 2003.

[8] H.Y. Chien, "Sasi: A new ultralightweight rfid authentication protocol providing strong authentication and strong integrity," IEEE Transactions on Dependable and Secure Computing, vol. 4, no. 4, pp. 337-340, Oct.-Dec. 2007.

[9] S. Vaudenay, "On privacy models for RFID," Advances in Cryptology - Asiacrypt 2007, pp. 68-87, December 2007.

[10] E.Y. Choi, S.M. Lee, and D.H. Lee, "Efficient RFID authentication protocol for ubiquitous computing environment," Proc. of SECUBIQ05, pp. 945-954, Decem-

ber 2005.

[11] J. Ha, S.J. Moon, J.M.G. Nieto, and C. Boyd, "Security analysis and enhancement of one-way hash based low-cost authentication protocol (OHLCAP)," PAKDD Workshops, pp. 574-583, May 2007.

[12] J. Ha, H. Kim, J. Park, S.J. Moon, J.M.G. Nieto, and C. Boyd, "HGLAP - hierarchical group-index based lightweight authentication protocol for distributed RFID system," EUC Workshops, pp. 557-567, December 2007.

[13] Y.K. Lee, L. Batina, and I. Verbauwhede, "EC-RAC: provably secure RFID authentication protocol," IEEE International Conference on RFID 2008, pp. 97-104, April 2008.

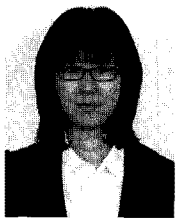
[14] 권혜진, 이재욱, 전동호, 김순자, "데이터베이스에서의 태그 검색이 쉽고 안전한 RFID 상호인증 프로토콜," 정보보호학회논문지, 18(5), pp. 125-134, 2008년 10월.

[15] D. Molnar, A. Soppera, and D. Wagner, "A scalable, delegatable pseudonym protocol enabling ownership transfer of RFID tags," Selected Areas in Cryptography, pp. 276-290, August 2005.

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

[17] S.M. Lee, Y.J. Hwang, D.H. Lee, and J.I. Lim, "Efficient authentication for Low-Cost RFID Systems," International Conference on Computational Science and its Applications, pp. 619-627, May 2005.

## 〈著者紹介〉



권 혜 진 (Hyejin Kwon) 학생회원
2007년 2월: 경북대학교 수학과 학사 졸업
2009년 2월: 경북대학교 정보보호학과 석사
2009년 3월~현재: 경북대학교 전자공학과 박사과정
〈관심분야〉 정보보호, 암호이론, 네트워크 보안

김 해 문 (Haemun Kim) 학생회원
2002년 2월: 경북대학교 전자공학과 학사 졸업
2004년 8월: 경북대학교 전자공학과 석사
2005년 2월~현재: 경북대학교 전자공학과 박사과정
〈관심분야〉 정보보호, 암호이론, 네트워크 보안, 스테가노그래피

정 선 영 (Seonyeong Jeong) 정회원
1987년 2월: 경북대학교 전자공학과 학사 졸업
2003년 2월: 경북대학교 정보통신학과 석사
2003년 3월~현재: 경북대학교 전자공학과 박사과정
2009년 3월~현재: 경운대학교 디지털전자공학과 전임강사
〈관심분야〉 유비쿼터스, 정보보호 응용기술

김 순 자 (Soonja Kim) 종신회원
1975년 2월: 경북대학교 수학 교육학과 학사
1977년 2월: 경북대학교 수학과 석사
1988년 2월: 계명대학교 수학과 박사
1993년 4월~현재: 경북대학교 전자·전기 공학부 교수
〈관심분야〉 정보보호 및 보안기술, 정보보호 응용기술