# A Scalable and Untraceable Authentication Protocol for RFID

Youngjoon Seo, Hyunrok Lee and Kwangjo Kim

International Research center for Information Security (IRIS)
Information and Communications University (ICU)
103-6 Munji-dong, Yuseong-gu, Daejeon, 305-732, Korea.
{golbeat,tank,kkj}@icu.ac.kr

**Abstract.** RFID (Radio Frequency Identification) is recently becoming popular, promising and widespread. In contrast, RFID tags can bring about traceability that causes user privacy and reduces scalability of RFID. Guaranteeing untraceability and scalability at the same time is so critical in order to deploy RFID widely since user privacy should be guaranteed. A large number of RFID protocols were designed in the open literature, but any known protocols do not satisfy untraceability and scalability at the same time to the best of our knowledge. In this paper, we suggest a RFID authentication protocol that guarantees untraceability and scalability together; needless to say preventing several known attacks: replay, spoofing, desyncronization, and cloning by eavesdropping. Our protocol supports ownership transfer and considers multi-tag-reader environment; a reader receives messages from the tags what a reader wants in our protocol. In addition, we address the reason why the item privacy is important, and a way to keep it securely.

## 1 Introduction

RFID is recently becoming popular all over the world due to its convenience and economical efficiency; furthermore, RFID nowadays comes into the spotlight as a technology to substitute the bar code [10, 11].

On the other hand, RFID is jeopardized from various attacks and problems as an obstacle of widespread RFID deployment: replay, spoofing, traceability, desyncronization, unscalability, and tag cloning. We focus ourselves on untraceability and scalability in this paper. To prevent attacker from tracing a tagged item is most important in RFID system since it infringes personal privacy. For example, Albrecht[18] who organized a Benetton boycott claimed RFID tags "spy chips" due to the traceability of tags. And moreover, tags with unique $ID$ can be associated with a person's identity. Garfinkel *et al.* discussed personal privacy threats in [13].

However, we have to keep the constant computational time in back-end server regardless of the number of tags when designing an untraceable protocol. In other words, there must be a trade-off between scalability and untraceability. If a response from a tag, as an example, does not include information about its $ID$, which is dynamic or incomputable, these protocols are likely to be unscalable since readers are supposed to exhaustively search in database to find tag's $ID$. If a response from a tag, on the contrary, includes information about its $ID$, which is static or computable, tagged items are likely to be traceable because an adversary also can find its $ID$ as an authorized one does.

The previous protocols[2, 6, 7, 15, 16] and hash lock scheme[14] are scalable, but traceable. Rhee *et al.*[5], Ohkubo *et al.*[9] and randomized hash lock[14] schemes are untraceable, but unscalable. Therefore, we try to design a scalable and untraceable protocol that any other literatures have not dealt with before.

## 1.1 Our Contribution

Our contribution in this work is twofold: firstly, we propose the reason why we write pseudo-EPC into tag's memory not a code itself. Writing EPC itself into tag's memory brings about infringing item privacy after an adversary eavesdrops EPC or tampering a tag. Shortly after an adversary finds out what EPC of the particular tag is, he/she can learn what kinds of items and whether tagged items are expensive or cheap. In other words, item privacy can be violated. It is clear that item privacy brings about user privacy and incentive to steal valuable items. On the other hand, writing pseudo-EPC into tag's memory guarantees item privacy even after an adversary comprises tags. It doesn't matter as long as back-end server converts pseudo-EPC into a valid EPC and points to a right entry for retrieving relevant product information.

Secondly, our contribution is to design a scalable and untraceable protocol which is more secure than Dimitrious protocol[15](denoted by "TD"); we use only four hash operations while TD uses five and more hash operations. We make it using a shared secret $k$; when a reader sends a query, a shared secret $k$ needs to be authenticated by a tag. This is totally different approach in comparison with the previous literatures. The only tags stored with the same secret $k$ respond to reader's query; a reader gets the message from particular tag what the reader wants. It reduces computational time in tags and back-end sever, especially in multi-tag-reader environment.

## 1.2 Notations

We use the notations for entities and operations as summarized in Table 1 throughout the paper.

**Table 1.** Notations

| | |
|---|---|
| $R$ | RFID tag reader, or transceiver. |
| $T$ | RFID tag, or transponder. |
| $\mathcal{T}_k$ | A set of $T$ which has same secret $k$. |
| $\mathcal{T}'_k$ | A set of $T$ which has secret $k'$ where $k' \neq k$. |
| $\mathcal{B}$ | A back-end Server. |
| $\mathcal{A}$ | An adversary. |
| $h()$ | One-way hash function. |
| $ID_i$ | Pseudo-EPC of $T$ at $i$-th query($i$=0,1,$\cdots$). |
| $k$ | Shared secret key between $R$ and $\mathcal{T}_k$. |
| $TS$ | Timestamp. |
| $TS_{last}$ | last $TS$ sent by an authorized $R$. |
| $t$ | Temporal storage. |
| $\oplus$ | Exclusive-or (XOR) function. |
| $M_1, M_2$ | Concatenation of messages $M_1$ and $M_2$. |
| $PIN$ | Access PIN written into a reserved $T$ memory. |
| $l \leftarrow r$ | Operator which updates $l$ with $r$. |
| $\overset{?}{=}$ | Verification operator to check whether the left hand side is same with the right hand side or not. |
| $\overset{?}{>}$ | Comparison operator to check whether the left hand side is greater than the right hand side or not. |
| $m$ | Number of read operations. |
| $n$ | Number of tags. |
| $\gamma$ | Number of tags within an operating range. |
| $\beta$ | Number of tags that have same $k$ within an operating range. |

## 1.3 Organization

The rest of the paper is organized as follows: In Section 2, we briefly introduce the previous work. In Section 3, we describe how to design a protocol that provides forward secrecy, untraceablility, scalability, synchronization, anti-spoofing, and anti-cloning. In Section 4, we propose our RFID authentication protocol which is a scalable and untraceable protocol based on hash function. In Section 5, we describe the analysis of our protocol. We finally conclude our results in Section 6.

## 2 Previous Work

There have been many papers which are hash-based[2, 4, 3, 5, 7, 9, 14, 15, 17], pseudonym-based[1, 12], zero knowledge-based[16] using PUF(Physical Unclonable Function), and tree-based protocol[8] using pseudonym generator that attempts to address the security concerns raised as using RFID tags, but it is believed that there is no perfect protocol that avoids all of the threats with reasonably low cost until now.

*Hash Lock Scheme*[14](denoted by "HLS") is based on one-way hash function; HLS is traceable. *Randomized Hash Lock scheme*[14](denoted by "RHLS") is an extended version of HLS to remove traceability, but RHLS is unscalable. Henrici *et al.*[2], Lee *et al.*[7](denoted by "LACP"), and TD are scalable, but traceable during a valid session. Ohkubo *et al.* protocol[9](denoted by "OSK") is untraceable, but unscalable. Wong *et al.*[6] and Tuyls *et al.* protocol[16] can be traceable; and also, pseudonym-based protocols[1, 12] can be traceable after $\mathcal{A}$ collects all of the pseudonyms.

## 3 Security Requirements

When designing a RFID authentication protocol, the following properties should be guaranteed together as our goal: forward secrecy, untraceablility, scalability, synchronization, item privacy, anti-cloning and preventing spoofing. In each subsection, we suggest how to improve unsatisfying security requirements in the previous five protocols: HLS, RHLS, OSK, LACP, and TD.

### 3.1 Forward Secrecy

OSK and TD can guarantee forward secrecy. OSK and TD employ a hash function to update an identifier while HLS and RHLS do not refresh an identifier; that is, upon compromising an identifier, $\mathcal{A}$ learns all the previous transactions in HLS and RHLS. LACP uses XOR operation to update an identifier; consequently, LACP fails to guarantee forward secrecy. Hash function has a one-wayness property, while XOR operation does not. $\mathcal{A}$ can collect all pseudonyms from the response of tags in pseudonym-based protocol[1, 12] which can not guarantee forward secrecy; more seriously, pseudonym-based protocol can not guarantee untraceablility. In order to design a protocol that guarantees forward secrecy, we have to use a hash function when updating secret keys as long as there is no alternative. When updating $ID_i$, finding a lightweight function or scheme that guarantees forward secrecy is still an open problem.

## 3.2 Untraceability and Scalability

In Table 2, forward secrecy($FS$), untraceability($UNT$), and untraceability during a valid session($UNT\text{-}DVS$) are viewed as one categorization. $FS$ and $UNT\text{-}DVS$ are classified into $UNT$; Guaranteeing $UNT$ means satisfying $FS$ and $UNT\text{-}DVS$. OSK is successful in designing $UNT$, but OSK causes the worst result in terms of scalability. The number of tags is going to increase sharply in the near future; furthermore, $T$ recognition rate is not perfect so far, which increases the number of read operation; the complexity of OSK, $O(2mn^2)$, definitely suffers from too much in multi-tag-reader environments since all of the tags within the operating range of reader are supposed to respond a query. That's why scalability also can not be overlooked. We introduce $\gamma$ as the number of tags within an operating range since all tags, which are stored in $\mathcal{B}$, are not likely to be within a range of $R$. After applying $\gamma$ to complexity of OSK, it becomes $O(2mn\gamma)$. In this paper, we define scalability as that the computational complexity is quite suitable for multi-tag-reader environment in the $\mathcal{B}$.

## 3.3 Synchronization

HLS and RHLS don't need to synchronize a shared key because the shared secret is fixed; however, TD, OSK and LACP have to synchronize secret information since they update key only by an authorized $R$. OSK can lose synchronization due to resilience. If desynchronization occurs, $\mathcal{B}$ can not recognize the $T$; this $T$ becomes useless in this case.

## 3.4 Spoofing and Cloning

HLS and RHLS send message in the clear; so, $\mathcal{A}$ can learn the shared secret keys by eavesdropping, and then can spoof $R$ and $T$. In OSK, spoofing $R$ is possible by replay attack. In LACP, $\mathcal{A}$ can spoof the $T$ if $R$ and $T$ send message carelessly.

Cloning is divided into two groups: by eavesdropping or by tampering. Cloning by eavesdropping has the same significance with spoofing the $R$ in terms of security; preventing $\mathcal{A}$ from cloning by tampering is hard to prevent since $\mathcal{A}$ learns all information of storage. However, Tuyls *et al.* protocol[16] discussed how to prevent $\mathcal{A}$ from cloning by tampering using PUF(Physical Unclonable Function), but it's too costful.

## 3.5 Item Privacy

*Item privacy* can be stated verbally as: active $\mathcal{A}$ can not find out the contents or price of a tagged item even though EPC is revealed.

Violation of item privacy gives $\mathcal{A}$ the seduction to steal tagged items after $\mathcal{A}$ eavesdrops EPC; in other words, item privacy should be guaranteed although $\mathcal{A}$ knows what kind of product after tampering $T$. For example, $\mathcal{A}$ tampers tiny jewelry such that the general public can not tell genuine from imitation; in this case, $\mathcal{A}$ is difficult to decide to counterfeit or not if pseudo-EPC is used in RFID tag.

## 4 Our Protocol

In this section, we propose a scalable and untraceable RFID authentication protocol based on hash function.

### 4.1 Initialization

Any $T$ has four non-volatile memories $ID_0$, $k$, access PIN and $TS_{last}$ which are initialized into $T$'s memory during manufacturing process; $ID_0$, pseudo-EPC, which is produced by hash function or the other encoding schemes, is written into $T$'s memory; access PIN is written into $T$'s reserved memory; $k$ is written into $T$'s memory; $TS_{last}$ is set by 0 while initializing. $TS_{last}$ is updated with $TS$ sent by an authorized $R$ to prevent replay attack after successful mutual authentication. $R$ only has $k$ which is stored during manufacturing process or ownership transfer. $\mathcal{B}$ keeps four fields: EPC, $h(ID_i)$, $ID_i$, and access PIN; $ID_i$ and access PIN are shared between $T$ and $\mathcal{B}$, while EPC and $h(ID_i)$ are not.

In our protocol, we assume that $\mathcal{B}$ can tell an authorized $R$ from an unauthorized one; time clock which is built in $R$ is tightly synchronized like the mobile phone in multi-tag-reader environment.

### 4.2 A Scalable and Untraceable protocol

Our protocol is illustrated in Figure 1. TD does not guarantee *UNT-DVS*; and so, we suggest a protocol which removes the weakness of TD. In addition, we propose how $R$ communicates with $T$ using timestamp to prevent replay attack without implementing time clock in $T$ unlike TD.

**Operation**

1. $R$ gets $TS$ from its timestamp information. $R$ computes $h(k,TS)$, and then transmits $h(k,TS), TS$ to $T$. $T$ compares $TS$ and $TS_{last}$. If $TS$ is greater than $TS_{last}$, then $T$ generates $h(k,TS)$ using $TS$ and $k$. Otherwise, $T$ considers it as an unauthorized request. If the value

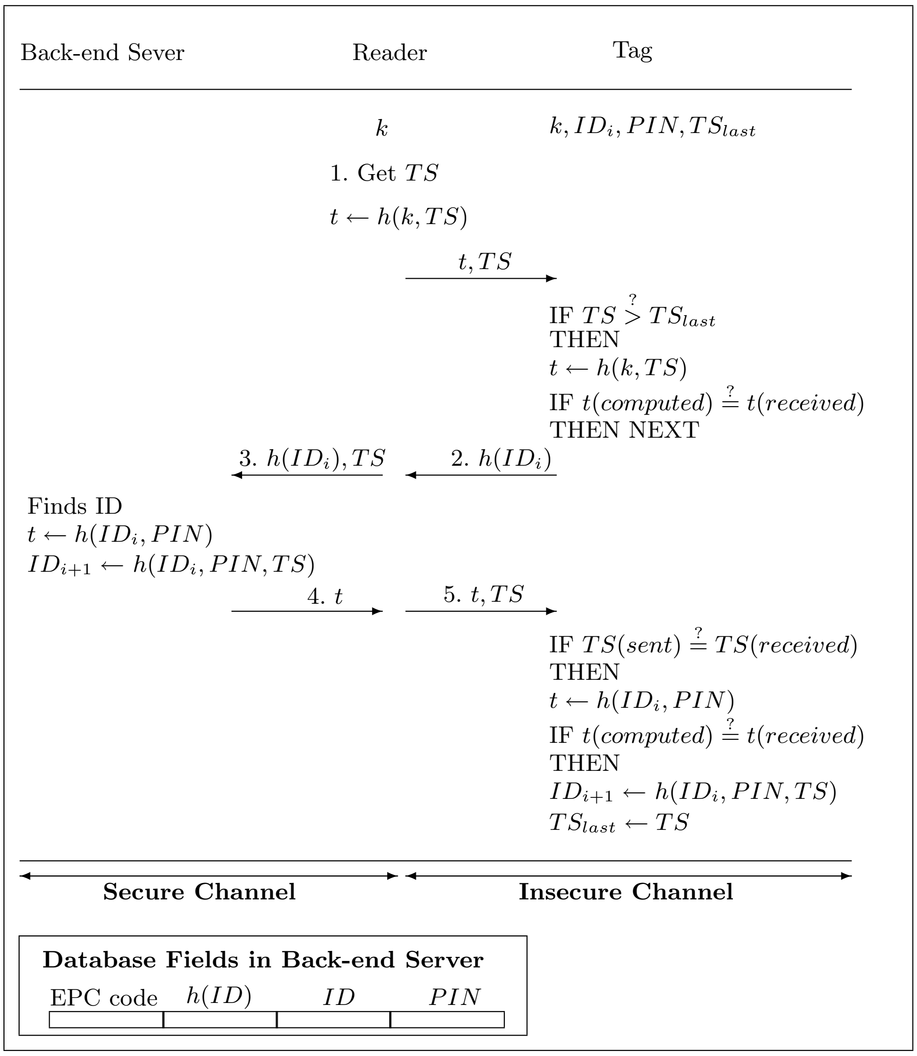
Back-end Sever
Reader
Tag
$k$
$k, ID_i, PIN, TS_{last}$
1. Get $TS$
$t \leftarrow h(k, TS)$
$t, TS$
IF $TS \overset{?}{>} TS_{last}$
THEN
$t \leftarrow h(k, TS)$
IF $t(computed) \overset{?}{=} t(received)$
THEN NEXT
3. $h(ID_i), TS$
2. $h(ID_i)$
Finds ID
$t \leftarrow h(ID_i, PIN)$
$ID_{i+1} \leftarrow h(ID_i, PIN, TS)$
4. $t$
5. $t, TS$
IF $TS(sent) \overset{?}{=} TS(received)$
THEN
$t \leftarrow h(ID_i, PIN)$
IF $t(computed) \overset{?}{=} t(received)$
THEN
$ID_{i+1} \leftarrow h(ID_i, PIN, TS)$
$TS_{last} \leftarrow TS$
Secure Channel
Insecure Channel
Database Fields in Back-end Server
EPC code
$h(ID)$
$ID$
$PIN$

**Fig. 1.** Our Protocol

received is the same as the value computed, they authenticate the $R$ as an authorized one. The step 1 is quite different from the other protocols: the other protocols authenticate $R$ at the last steps(4 - 5) while our protocol authenticates $R$ at the step 1. In other words, $\mathcal{T}_k$ responds to $R$ while $\mathcal{T}'_k$ does not respond.

2. $T$ sends $h(ID_i)$ to the $R$, which reduces time complexity to $O(\beta)$ in multi-tag-reader environment because all of the tags respond to $R$'s

query in the previous protocols at all time while only $\mathcal{T}_k$ responds in our protocol.
3. $R$ forwards $h(ID_i)$ and $TS$ to $\mathcal{B}$. $\mathcal{B}$ finds $ID_i$; $\mathcal{B}$ computes $h(ID_i, PIN)$ using $ID_i$ and $PIN$; $\mathcal{B}$ updates $ID_i$ to $ID_{i+1}$ where $ID_{i+1} = h(ID_i, PIN, TS)$. Otherwise, $\mathcal{B}$ stops the procedure.
4. $\mathcal{B}$ sends $h(ID_i, PIN)$ to the $R$.
5. $R$ forwards $h(ID_i, PIN)$ and $TS$ to $T$. $T$ compares received and sent $TS$. If two values equal, $T$ also computes $h(ID_i, PIN)$ and compare the received and computed values. If all comparisons are successful, $T$ updates $ID_i$ to $ID_{i+1}$ like $\mathcal{B}$ does; $T$ also updates $TS_{last}$. Otherwise, $T$ stops the procedure.
   The main difference between the previous protocols and ours is that $T$ authenticates the $R$ two times at the steps 1 and 5 while the $R$ authenticates $T$ just one time in the previous protocols.

Our main idea is to use a shared secret key $k$; $k$ is written as a new value when enrolling tags in the system or doing ownership transfer while $ID_i$ is updated as $ID_{i+1}$ when a successful mutual authentication happens with only an authorized $R$.

## 5 Security and Performance Analysis

In this section, we analyze security of our protocol against all aspects in Table 2.

- **Synchronization.** Simplified TD protocol happens to desyncronization problem. TD protects desyncronization between $\mathcal{B}$ and tags at the last step in enhanced TD protocol. We, however, don't need the last step to avoid desyncronization since our protocol emits a query with shared secret $k$ which is used to authenticate $R$. On the other hand, Although the memory channel is read by $\mathcal{A}$ once; we guarantees synchronization between tags and $\mathcal{B}$ even though $\mathcal{A}$ knows $k$ and $h(ID_i)$. The reason why we should use $TS$ is discussed in [15].
- **Forward Secrecy.** Our protocol updates $ID_i$ to $ID_{i+1}$ using a one-way function $h()$ like OSK and TD. As long as there is no alternative, we have to use one-way function to guarantee forward secrecy.
- **Untraceability during a valid session.** Tags authenticate the $R$ after receiving the first message, and then tags respond to only an authorized $R$'s query. Therefore, tags do not respond to $R$ with different $k$. As a result, tags are untraceable during a valid session since $\mathcal{A}$ doesn't impersonate even in the step 1.

**Table 2.** Comparison with others

| Protocol | HLS [14] | RHLS [14] | OSK [9] | TD [15] | LACP [7] | Our Protocol |
|---|---|---|---|---|---|---|
| Forward Secrecy | $\times$ | $\times$ | ○ | ○ | $\times$ | ○ |
| Untraceability during a valid session | $\times$ | ○ | ○ | $\times$ | $\times$ | $*$ |
| Untraceability | $\times$ | $\triangle$ | ○ | $\triangle$ | $\triangle$ | $\star$ |
| Scalability | $O(1)$ | $O(n)$ | $O(2mn)$ | $O(1)$ | $O(1)$ | $O(1)$ |
| Scalability in multi-tag-reader environment | $O(\gamma)$ | $O(n\gamma)$ | $O(2mn\gamma)$ | $O(\gamma)$ | $O(\gamma)$ | $O(\beta)$ |
| Hash operations | 0 | 1 | 2 | 5+$\alpha$ | 2 | 4 |
| Prevent spoofing $R$ | $\times$ | $\times$ | ○ | ○ | ○ | ○ |
| Prevent spoofing $T$ | $\times$ | $\times$ | $\times$ | ○ | $\times$ | ○ |
| Synchronization | NA | NA | $\triangle$ | ○ | ○ | ○ |

Notations

| | | | |
|---|---|---|---|
| ○ | satisfied | $\triangle$ | partially satisfied |
| $\times$ | not satisfied | $*$ | if k is revealed, $\times$. Otherwise, ○ |
| | | $\star$ | if k is revealed, $\triangle$. Otherwise, ○ |

- **Untraceability.** Tags authenticate the $R$ after receiving the first message; $R$ authenticates the tags after receiving the second message. In each step, tags and $R$ authenticate counterpart to remove traceability. In addition, although $\mathcal{A}$ knows $k$, $\mathcal{A}$ can not trace a particular tag since tag responses to query is always different at the valid session.
- **Scalability.** This is most big contribution in our work. $\mathcal{B}$ has time complexity $O(\beta)$ to find a tag in multi-tag-reader environment. This result is the best complexity in comparison with the previous protocols. Time complexity of each protocol changes in multi-tag-reader environment(See Table 2); from $O(1)$ to $O(\gamma)$ in most cases, from $O(1)$ to $O(\beta)$ in ours where $\beta < \gamma < n$.
- **Spoofing the tag.** As long as $\mathcal{A}$ doesn't know the value of $k$, $\mathcal{A}$ can not spoof the tags in our protocol. If $\mathcal{A}$ tampers with a tag, then $\mathcal{A}$ can spoof the tags at the step 1. However, $\mathcal{B}$ finds out that $\mathcal{A}$ is not an authorized $R$ in the end. There is no way to spoof the a tag unless $\mathcal{A}$ knows $k$ and $ID_i$.
- **Spoofing the reader.** As long as $\mathcal{A}$ doesn't know the $ID_i$, $\mathcal{A}$ can not spoof the $R$ since tag response to $R$'s query is different at all time.
- **Item Privacy.** The party who has EPC is only $\mathcal{B}$ in our protocol; that is, we guarantee item privacy as long as $\mathcal{B}$ is not compromised. The

other previous protocols are also possible to guarantee item privacy if HLS, RHLS, OSK, TD, and LACP assume that those satisfy three conditions: only $\mathcal{B}$ has EPC, $T$ doesn't have EPC, $ID$ is not a EPC itself.
- **Performance Analysis.** Our protocol is more secure than TD in terms of traceability aspects even though ours reduces hash operations five and more to four. In our protocol, tag needs four hash operations to take care of communicating with $R$ with quite good security performance. Under the assumption that tags can not be tampered, we don't need to send last message.
- **Ownership Transfer.** We supports ownership transfer using $k$. As far as we know, ownership transfer issue is dealt with only in [8] so far. For example, Alice has $R$ that has $k$ which is also stored in tagged items of Alice. When Alice gets some tagged item from Bob, Alice can write her own $k$ which is changeable into tagged item received from Bob.

## 6 Concluding Remarks

We deal with what item privacy is, why item privacy is important and how the way guaranteeing item privacy can be applied to our protocol.

There is a trade-off between scalability and untraceablility in RFID authentication protocol; therefore, many literatures did not suggest a protocol which guarantees scalability and untraceability together. However, in this paper, we propose a scalable and untraceable protocol. In addition, $R$ gets response from tags what $R$ wants. As future work, we will propose scalable and untraceable RFID authentication protocol with specific ownership transfer.

## References

1. Ari Juels, "Minimalist Cryptography for Low-cost RFID Tags", In C. Blundo and S. Cimato, editors, *The Fourth International Conference on Security in Communication Networks – SCN 2004, LNCS 3352*, pp.149-164, Sep. 2004, Springer-Verlag, Amalfi, Italia.
2. Dirk Henrici and Paul Müller, "Hash-based Enhancement of Location Privacy for Radio-Frequency Identification Devices using Varying Identifiers", *International Workshop on Pervasive Computing and Communication Security – PerSec 2004*, pp.149-153, Mar. 2004, IEEE Computer Society, Orlando, Florida, USA.
3. Gene Tsudik, "YA-TRAP: Yet Another Trivial RFID Authentication Protocol", *International Conference on Pervasive Computing and Communications – PerCom 2006*, Mar. 2006, IEEE Computer Society Press, Pisa, Italy. To appear.

4. Gildas Avoine and Philippe Oechslin. "A Scalable and Provably Secure Hash based RFID Protocol", *In International Workshop on Pervasive Computing and Communication Security – PerSec 2005*, pp.110-114, Mar. 2005, IEEE Computer Society Press, Kauai Island, Hawaii, USA.
5. Keunwoo Rhee, Jin Kwak, Seungjoo Kim and Dongho Won, "Challenge-Response based RFID Authentication Protocol for Distributed Database Environment", *International Conference on Security in Pervasive Computing – SPC 2005, LNCS 3450*, pp.70-84, Apr. 2005, Springer-Verlag, Boppard, Germany.
6. Kirk Wong, Patrick Hui and Allan Chan, "Cryptography and Authentication on RFID Passive Tags for Apparel Products", *Computers in Industry*, Nov. 2006, Elsevier Science, Article In press.
7. Su-Mi Lee, Young Ju Hwang, Dong Hoon Lee and Jong In Lim, "Efficient Authentication for Low-Cost RFID Systems", *International Conference on Computational Science and its Applications - ICCSA 2005, LNCS 3480*, pp.619-627, May 2005, Springer-Verlag, Singapore.
8. David Molnar, Andrea Soppera and David Wagner, "A Scalable, Delegatable Pseudonym Protocol Enabling Ownership Transfer of RFID Tags", *Selected Areas in Cryptography – SAC 2005, LNCS 3897*, pp.276-290, Aug. 2005, Springer-Verlag, Kingston, Canada.
9. Miyako Ohkubo, Koutarou Suzuki and Shingo Kinoshita, "Cryptographic Approach to Privacy-friendly Tags", *In RFID Privacy Workshop*, 2003, MIT, USA.
10. "Navigating the New Era of RFID", Article in EPCglobal Canada Inc.
11. Nigel Wood, "Global Supply Chain GTIN & RFID Standards II", *EPC Global Standards Development*, EPCglobal Canada, October 14, 2004.
12. Philippe Golle, Markus Jakobsson, Ari Juels and Paul Syverson. "Universal Re-encryption for Mixnets", *The Cryptographers' Track at the RSA Conference – CT-RSA, LNCS 2964*, pp.163-178, Feb. 2004, Springer- Verlag, San Francisco, California, USA.
13. Simson L. Garfinkel, Ari Juels and Ravi Pappu, "RFID Privacy: An Overview of Problems and Proposed Solutions", *IEEE SECURITY and Privacy*, pp.34-43, May-Jun. 2005.
14. Stephen Weis, Sanjay Sarma, Ronald Rivest and Daniel Engels, "Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems", *Conference on Security in Pervasive Computing – SPC 2003, LNCS 2802*, pp.454-469, Mar. 2003, Springer-Verlag, Boppard, Germany.
15. Tassos Dimitriou, "A Lightweight RFID Protocol to protect against Traceability and Cloning attacks", *Conference on Security and Privacy for Emerging Areas in Communication Networks – SecureComm'05*, pp.59-66, Sep. 2005, Athens, Greece.
16. Pim Tuyls and Lejla Batina, Lejla, "RFID-Tags for Anti-Counterfeiting", *Topics in Cryptology – CT-RSA 2006, LNCS 3860*, pp.115-131, Feb. 2006, Springer-Verlag, San Jose, CA, USA.
17. Jeongkyu Yang, Jaemin Park, Hyunrok Lee, Kui Ren and Kwangjo Kim, "Mutual Authentication Protocol for Low-cost RFID", *Ecrypt Workshop on RFID and Lightweight Crypto*, pp.17-24, Jul. 2005, Graz, Austria.
18. http://www.spychips.com/what-is-rfid.html.