A Secure and Efficient Authentication Protocol for Passive RFID Tags

Constantinos Kolias[#1], Vasilis Kolias[*2], Georgios Kambourakis[#3]

[#] *Department of Information and Communications Systems Engineering*
, University of the Aegean
Karlovassi, Samos, Greece
[1] kkolias@aegean.gr
[3] gkamb@aegean.gr

[*] *Department of Electrical and Computer Engineering*
National Technical University of Athens
[2] vkolias@medialab.ntua.gr

***Abstract*— At the onset of the ubiquitous computing era, systems need to respond to a variety of challenges, in order to capitalize on the benefits of pervasiveness. One of the pivotal enablers of pervasive computing is the RFID technology which can be successfully applied in numerous applications. However, the interaction of such applications with sensitive personal data renders the need for assuring confidentiality a *sine qua non*. The native limitations in computing resources, i.e., computational power, memory etc, that characterize nearly all classes of RFID tags make the development of custom-tailored RFID security protocols a troublesome yet challenging task. In this paper we propose a mutual authentication protocol for low cost RFID tags and readers. We also demonstrate that our scheme is more efficient in terms of resource utilization on the backend server, and under identical conditions, more secure when compared with existing congruent protocols.**

I. Introduction

Today, the advances in wireless communication technologies and the proliferation of mobile devices have enabled the realization of pervasive and intelligent environments for users (and devices) to communicate with each other, interact with information processing nodes, and acquire ubiquitously a plethora of mobile wireless services through various types of access networks. As communications further improve, combined with solid technological advances in miniaturization and nanotechnology, a new perspective in the information and communications world has emerged; not only will *anyone* be able to access information and services *anytime* and *anywhere*, but also *anything*. Adding this characteristic to the area of pervasive computing creates a new virtual-space where objects will gain autonomy and context awareness while being able to interact in full interoperability. Without doubt, this technological evolution, encapsulates immense potential to both users and developers.

One of the cornerstones of pervasive computing is Radio Frequency Identification technology (RFID) because it enables unique identification of any entity in the context of pervasive space. Initially seen as a replacement for barcodes, RFID technology has been deployed successfully in many cases, such as access control, baggage identification, automotive systems, parcel tracking, library check out and check in, logistics and supply chain. However, the wide deployment of this technology is often subject to various kinds of threats and attacks and therefore raises serious confidentiality concerns. Note that in security parlance, confidentiality equals privacy + authenticity. In this paper we propose a mutual authentication protocol for low cost RFID tags and readers. Our scheme enhances existing work in the field and guaranties stronger security between reader and tag communication. Also, it is secure against Denial of Service (DoS) and replay attacks and provides forward secrecy. Last but not least, when compared to similar proposals, our protocol performs better by utilizing the resources of the backend server more effectively.

The remainder of the paper is structured as follows. Next section presents an overview of RFID technology and identifies some security threats commonly met in RFID systems. A definition of the problem in question is given in Section III. Section IV presents and constructively argues on previous work in the same field, while section V provides a security analysis of the proposed protocol. Last section draws a conclusion and gives pointers to future work.

II. RFID Technology And Security Threats

An RFID system is usually comprised of: (a) the RFID tag, which contains a digital number associated with the physical object that it is attached to, and (b) the RFID reader which is connected to a backend database. The reader is also equipped with an antenna, a transceiver and a processor that broadcasts a radio signal in order to query the tag and read its contents. According to their energy resources and computational capabilities RFID tags are distinguished into passive and active. Passive tags, unlike active ones, do not have an internal source of energy and therefore they have a smaller size and computational resources. The maximum reading distance of a tag varies from a few centimeters to approximately ten meters. Also, its cost (e.g., for tags like EPCglobal [1] Class-1 Gen-2 passive tags) is about 13 cents per tag [2] and is expected to decrease to 5 cents within the next few years [3, 4].

A fundamental requirement of pervasive systems in general, is the ability to uniquely identify things and entities. By satisfying this requirement RFID technology brings along with the benefits of pervasiveness the user's expectation of maintaining his confidentiality under any circumstances. With their wide deployment, low cost tags have unfortunately been object of various kinds of attacks, this way raising serious concerns. In [5] the authors classified the threats and possible attacks against RFID systems in 4 different layers. Based on that classification we summarize the possible attacks that can be launched against RFID systems as following:

- Physical Layer: in this layer the adversary launches attacks by taking advantage of the fact that has immediate access to a tag or by exploiting the security holes of the RFID wireless communications. Such category of attacks includes: (a) physical removal of the tag from the associated product, (b) destroying the tag, e.g., by means of exposing it to extreme environmental conditions or static electricity, (c) using electromagnetic jamming in order to temporarily prevent communication with readers and (d) taking advantage of the RFID KILL command to make the tag permanently inoperable.
- Network - Transport Layer: in this layer the adversary launches attacks on the way RFID systems are communicating and packets are transferred. This sort of attacks include replication of a valid tag, tag spoofing, impersonation of a valid reader, eavesdropping on tag - reader communication and finally attacking systems of the enterprise backbone (database servers, networking devices etc).
- Application Layer: in this layer the adversary aims to the exchanged application data as the case may be. Such attacks include modification of the information contained in the tag's memory, accessing the contents of tag without being authorized, and malicious code injection to the writable memory of a tag in order to harm middleware applications (e.g., by exploiting SQL injection attacks).
- Strategic Layer: This layer includes attacks such as social engineering and competitive espionage. More generic attacks that could capitalize on carelessly designed components, practices or policies are also classified in this category.

III. Related Work and Problem Statement

RFID technology and its applications is an ongoing research topic. Due to the openness of the wireless medium and several other restrictions stemming from the miniaturization of tags, particular effort is put into developing secure RFID protocols and mechanisms, especially for sensitive applications. Unfortunately, most of the existing work is focusing on expensive RFID tags that afford more computational resources and are able to perform more complex cryptographic operations, such as hash functions, symmetric encryption, and in some special cases, public key cryptography. On the other hand, low cost tags that are expected to be widely adopted in the near future, lack the processing power needed for such operations. As Juels summarizes in [6] privacy and authentication requirements are satisfied with techniques such as: (a) tag killing or tag sleeping, (b) renaming, (c) proxying, (d) distance measurement, (e) blocking, and (f) verification via PINs. Although in most cases these techniques are competent they cannot be considered as a panacea for any possible threat. For instance, tag killing renders a tag incapable of responding to reader queries after it receives a specific bit sequence. This of course is an effective way to enforce user confidentiality, but it eliminates all the benefits of RFID tags from that point on.

For that reason, several secure communication schemes have been proposed in the literature so far which usually focus on the protection of tags against network-transport layer threats. Duc et al. in [7] propose a mutual authentication scheme appropriate for low cost tags. Although the authors claim that the aforementioned scheme offers implicit reader-to-tag authentication, explicit tag-to-reader authentication, traffic encryption and privacy protection against tracking, Chien and Chen [8] revealed that under certain circumstances is weak. More specifically, the protocol cannot resist DoS attacks if the synchronization between the tag and reader is lost. To achieve that, an attacker could simply intercept one of the End Session messages. The protocol is also weak against replay attacks. When the synchronization of the keys is lost a bogus tag could replay the old data to disguise as a legitimate one. Also, the protocol cannot provide forward secrecy if a tag is compromised.

Also, Chien and Chen [8] propose an extension to the previous scheme in order to overhaul some of its weaknesses. Specifically, the authors claim that their enhanced scheme achieves mutual authentication (of both the tag and reader), privacy protection, resistance against DoS attacks (due to the double set of keys), and forward secrecy. Nonetheless, we can point out two weaknesses of this protocol:

1. It is possible for an attacker to launch a replay attack in order to retrieve the DATA related to a specific tag. The window of opportunity for this attack to be fruitful is only after one successful authentication at maximum. To achieve that, the attacker would simply eavesdrop on a normal reader-to-tag communication and store the values M_1, N_1, N_2. Then, by exploiting his false reader equipment, forwards the message to the server. The server would send the DATA message to the reader because the equation $M_1 \oplus K_{old} = CRC(EPC \| N_1 \| N_2)$ is valid.
2. The protocol is very demanding in resources on the server side because it requires that the server would check if $M_1 \oplus K_{old} = CRC(EPC_x \| N1 \| N2)$ or $M_1 \oplus K_{new} = CRC(EPC_x \| N1 \| N2)$ for every single record in its database.

Taking into account the vulnerabilities of the protocols discussed above we propose a novel scheme in order to provide secure reader-to-tag communication and shield against DoS and replay attacks. Also, our scheme increases the overall system performance by utilizing the backend server' resources better. In the next section we elaborate on our proposal and provide a theoretical comparison with similar solutions in the field.

IV. Framework Description

A. Assumptions and Prerequisites

Our scheme is fully compatible with the EPCGlobal Class-1 Gen-2 standard [9]. The EPCGlobal Class-1 Gen-2 passive tags can afford very limited resources and computational power. The only operations available that can be used for cryptographic purposes are the PRNG, 16-bit CRC [10] and XOR functions. We rely on the XOR operation for ciphering sensitive data such as secret keys and EPC identifiers during transmission. Since passive tags are vulnerable to physical tampering, the secret shared keys must be updated frequently. The proposed scheme utilizes the same PRNG function with the same seed for both the tag and the reader in order to mutually update their keys rather than exchange new ones. This update must be done in a synchronous way at both ends. We also make use of CRC as a lightweight substitute of the cryptographic hash functions, although it is common sense that it is not cryptographically strong.

Due to the open nature of the wireless medium we assume that all bidirectional communication between a tag and a reader is susceptible to various attacks. That is, any malicious entity is able to eavesdrop, alter, drop or insert packets. We also assume that the communication between the reader and the backend server is secured by well accredited Internet mechanisms like the SSL or IPSec protocols. Moreover, physical tampering of tags is considered feasible. Once an attacker has acquired a tag we assume that he is also able to acquire any of the protocol parameters such as the secret keys.

Briefly, the proposed protocol is based on the two previously described schemes [7, 8] and introduces the use of sequence numbers (SQN) and group keys to provide protection against replay attacks and increase the overall system performance, respectively. All notations used for describing our protocol are presented in table I.

TABLE I
NOTATIONS USED IN THE PROPOSED PROTOCOL

Notation	Meaning
T	RFID tag
R	RFID reader
S	Backend Server
EPC	Electronic Product Code
PNRG()	Pseudo Random Number Generators Function
CRC()	Cyclic Redundancy Check Function
$\oplus$	Exclusive OR (XOR) Function
||	Concatenation Function
AK_i	Authentication Key for i-th Session
CK_i	Cipher Key for the i-th Session
SQN	Sequence Number of the Message
A:	Action of entity A
A → B	Communication from A to B
X ← Y	Value assignment command
GK	Group Key

B. Mutual Authentication Scheme

There are two phases in our proposed scheme; the initialization as well as the main phase. During the initialization or deployment phase the server randomly selects and stores in each tag (T_x) the following five values: (1) an authentication key AK_{x0}, (2) a cipher key CK_{x0}, (3) a sequence number SQN_{xtag}, (4) a group key GK, and (5) the Electronic Product Code (EPC_x) for the specific product. The server also associates each EPC with a number of relative values and stores them in the local database. These are: (1) AK_{xold}, (2) AK_{xnew}, (3) CK_{xold}, (4) CK_{xnew}, (5) $SQN_{xserver}$, (6) GK, and (7) EPC_x.

The AKx is a temporary identifier of the tag, so that EPCx will not have to be used during the first steps of the protocol. The CKx is used for encrypting EPCx during transmission. Both of these keys are updated after each successful main phase. AKx and CKx must be synchronized at the server so that successful verifications can take place. If an attacker drops specific messages so that synchronization of either of these two values is lost a DoS attack will occur. As Chien and Chen pointed out in [8] old and new values should be maintained in the database for both AKx and CKx so that the system will resist to this type of attacks. The GK is common for all tags belonging to the same group. In the simplest case, all the tags administered by the same server belong to the same group, but as we will explain later on that is not an optimal practice for security reasons. The GK is updated after a specific time interval administered by the server. The purpose of the SQN value is twofold; firstly provides a mechanism for ensuring message freshness, and secondly a means to detect replays. When a Query Request message reaches the tag, then the SQN is increased sequentially to its next value. On the other hand, the value of the SQN stored on the server is only updated after the end of a successful tag authentication procedure.

During the main phase the following steps take place:

1. $R \rightarrow T_x$: [Query Request]: Using this message the reader initiates the mutual authentication procedure with the tag.
2. $T_x \rightarrow R \rightarrow S$: [$M_1$]: The tag locally generates a random nonce r, increases its SQN_{xtag} by 1 and then computes $M_1 = (AK_{xi} \| SQN_{xtag} \| r) \oplus GK$ and sends this message to the reader, which in turn forwards it to the server. Upon reception the server computes $M_1 \oplus GK$ in order to acquire the values of $AK_{xi,}$ SQN_{xtag}, and r. Note that the length of each of these fields in bits is static. After that, the server checks if a matching record AK_{xold} or AK_{xnew} exists in the database. If true, then the server retrieves the rest of the values stored for this tag. If $SQN_{xserver} < SQN_{xtag}$ then the authentication of the tag is considered successful. In any other case the procedure fails silently and the $SQN_{xserver}$ for the corresponding tag retains its value.

3. $S \rightarrow R$: [M_2, DATA]: The server computes M_2 = CRC(EPC_x ∥ r) ⊕ CK_{xold} or M_2 = CRC(EPC_x ∥ r) ⊕ CK_{xnew} depending on weather it was AK_{xold} or AK_{xnew}. This situation satisfies the verification test taken during the previous step. The server also updates the values of all the corresponding keys, i.e., $AK_{xold} \leftarrow AK_{xnew}$, $CK_{xold} \leftarrow CK_{xnew}$, $AK_{xnew} \leftarrow$ PRNG(AK_{xnew}), $CK_{xnew} \leftarrow$ PRNG(CK_{xnew}) and $SQN_{xserver}$ is set to the same value as SQN_{xtag} ($SQN_{xserver} \leftarrow SQN_{xtag}$). The server forwards the message M_2 to the reader along with all relative information about the tag.

4. $R \rightarrow T_x$: [M_2]: The tag receives the M_2 message, computes M_2 ⊕ CK_{xi} in order to retrieve the values of EPC and r, and verifies them against the local ones. If equal, the reader is authenticated to the tag and the tag can update its keys, that is, $AK_{x(i+1)} \leftarrow$ PRNG(AK_{xi}) and $CK_{x(i+1)} \leftarrow$ PRNG(CK_{xi}). Figure 1 depicts the mutual authentication procedure described above.

V. Security Analysis

The proposed protocol provides privacy protection of the identity of the tag, i.e., its EPC number, implicit tag-to-reader and reader-to-tag authentication, protection against DoS and replay attacks, and forward secrecy. The tag never transmits its EPC. On the contrary, it only sends its authentication key (scrambled with the group key), which acts as a temporary indexer of the tag in the server's database. For each session the tag sends a unique M_1 because AK_{xi} has been updated, SQN has increased and a different r is concatenated. The server transmits the EPC only after it has been masked (XORed) with the cipher key and only after the tag has been authenticated. We also stress here that the M_2 message is different for each session. Protection against DoS attacks is also provided, because the server maintains both previous and current values of the keys (AK_{xold}, AK_{xnew}, CK_{xold}, CK_{xnew}) for each tag. Of course, an attacker could easily intercept the message M_2 on the way to the tag thus causing the server to update its keys while the tag would stop the procedure before updating its own. The result would be loss of synchronization of the keys and both the tag and server would not be able to authenticate each other anymore. However, by checking if AK_{xold} or AK_{xnew} is active during the first steps of the main phase, the tag and server automatically re-synchronize their keys, if necessary. As already mentioned, the SQN is a mechanism to protect against replay attacks. An attacker cannot store and replay old M_1 values because the server would easily detect that the SQN contained in M_1 is stale and therefore it will immediately drop the received message. On the other hand, the attacker cannot replay an old M_2 value because the r contained in M_1 and M_2 will be different and the tag will ignore the message.

Our protocol could be considered weak in the following two cases:

(a) The attacker has compromised a tag that belongs to a certain group and also has somehow derived all the tag members of the same group. Upon acquiring the GK the attacker has also knowledge of the AK and SQN of each tag belonging to that group. More specifically, the attacker would simply have to compute M_1 ⊕ GK to derive them. Having all these values the attacker would be able to authenticate fake tags and also track legitimate tags. Even in this case he is not able to acquire the EPC of the communicating tag(s) since he is not aware of the CK.

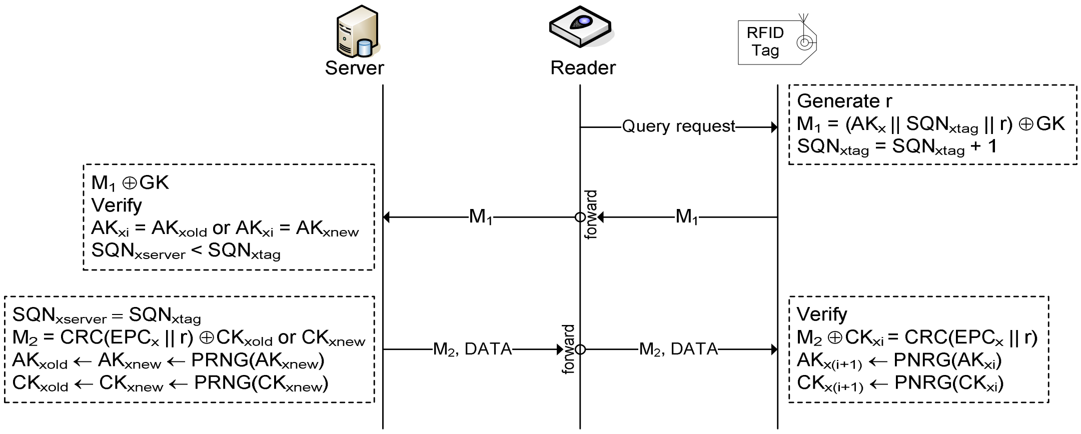
(b) Our protocol could also be considered weak against tracking of specific tags between successive authentications. An attacker would simply have to create Query Request messages and store the answer M_1 coming from the tag. Since the procedure would never be completed and keys would not be updated, the same AK_i will be used for all subsequent M_1 (referred as $M_1^{'}$) messages that the tag would send to the reader. By simply computing $M_1 \oplus M_1'$ the attacker would be able to distinguish a specific tag as the result will always start with as many zeros as the size of AK and end with as many zeros as the size of GK. Even so, the attacker is not in place to derive the EPC of any tag.

It is also stressed that the proposed protocol protects only against network / transport layer DoS. That is, it cannot cope with physical layer electromagnetic jamming, Death-by-Retry style attacks or packet dropping. Such an attack is very difficult, if not impossible, to counteract and remains out of the scope of this paper.

Of course, the use of the group key mechanism has a major impact on the performance of the overall system and more specifically the server workload. The utilization of a single group key would completely eliminate the need for the server to iteratively check if various equations hold for all the tag records inside its database. Note that this must be done in order to derive the identity of the communicating tag for each single session. Nevertheless, this would lead to a serious security risk for the whole group as described above. In many cases it might be desirable to make use of multiple group keys. So, let us consider the following example: a library uses RFID tags for indexing 10^6 books. During the initialization phase, the server could categorize those items in 100 different groups at random and share a group key with the items of the corresponding group. During the main phase of the authentication procedure, the server would have to iteratively test different group keys in order to derive the identity of the communicating tag, but it still would be much more efficient than the Chien and Chen's protocol [8]. That is, 10^2 iterations at maximum against 10^6 iterations required in the works described in [7, 8]. So, when efficiency matters, our protocol has complexity O(g), where g is the number of groups, against O(t) in [7, 8], where t is the number of tags. From an attacker point of view, a truly random categorization of items to groups (for instance unrelated books located around the library premises) would make it extremely difficult for any aggressor to infer all the items that belong in the same group and thus capitalize on the knowledge gained from a single compromised tag. Table II presents a comparison of the security capabilities of various authentication schemes for RFID systems found in literature so far. Note that most of the schemes cited are not Gen-2 compliant which means that have more security capabilities.

TABLE II. COMPARISON OF SIMILAR RFID AUTHENTICATION SCHEMES

Scheme	GEN-2 Compliant	Privacy	Anonymity	Resistant to Replay Attacks	Resistant to DoS Attacks	Supports Forward Secrecy
Weis et al. [11]	X	X	X	X	√	X
Ohkubo et al. [12]	X	√	√	X	√	√
Henrici-Muller [13]	X	√	X	X	√	X
Rhee et al. [14]	X	√	√	√	√	X
Molnar-Wagner[15]	X	√	√	√	√	X
Yang et al.[16,17]	X	√	X	√	√	X
Karthikeyan-Nesterenko[18]	√	√	X	X	X	X
Duc et al. [7]	√	√	√	X	X	X
Chien-Chen[8]	√	√	X	X	√	√
This work	√	√	X	√	√	√



Server
Reader
RFID Tag
Generate r
$M_1 = (AK_x \,||\, SQN_{xtag} \,||\, r) \oplus GK$
$SQN_{xtag} = SQN_{xtag} + 1$
Query request
$M_1 \oplus GK$
Verify
$AK_{xi} = AK_{xold}$ or $AK_{xi} = AK_{xnew}$
$SQN_{xserver} < SQN_{xtag}$
M_1
forward
M_1
$SQN_{xserver} = SQN_{xtag}$
$M_2 = CRC(EPC_x \,||\, r) \oplus CK_{xold}$ or CK_{xnew}
$AK_{xold} \leftarrow AK_{xnew} \leftarrow PRNG(AK_{xnew})$
$CK_{xold} \leftarrow CK_{xnew} \leftarrow PRNG(CK_{xnew})$
M_2, DATA
forward
M_2, DATA
Verify
$M_2 \oplus CK_{xi} = CRC(EPC_x \,||\, r)$
$AK_{x(i+1)} \leftarrow PNRG(AK_{xi})$
$CK_{x(i+1)} \leftarrow PNRG(CK_{xi})$

Figure 1. The proposed scheme

VI. CONCLUSIONS AND FUTURE WORK

Low cost tags are anticipated to have a great impact on RFID applications because of their unique advantages and their minimalistic nature. This paper presented a mutual authentication protocol for low cost RFID equipment, such as the EPCGlobal Class-1 Gen-2. Our scheme improves the works described in [7, 8] by enabling readers and tags to communicate securely. Apart from confidentiality, our protocol provides resistance against DoS and replay attacks as well as forward secrecy. In terms of performance in the backend server, our mechanism can be much less resource demanding when compared to similar proposals. It is our intension to rectify the protocol in order to provide intractability to their owners, and since tags are prone to physical tampering, a better mechanism for assuring tag group security when a tag member is tampered should be provided.

REFERENCES

[1] EPCglobal, http://www.EPCglobalinc.org.

[2] R. Want, “An introduction to the RFID Technology”, IEEE Pervacive Computing, Vol. 5(1), pp. 25-33, March 2006.

[3] RFID Journal, http://www.rfidjournal.com.

[4] S.E. Sarma, S.A. Weis, and D.W. Engels. “RFID systems, security and privacy implications” , Technical Report MIT-AUOTOID-WH-014, AutoID Center, MIT, 2002.

[5] A. Mitrokotsa, M. R. Rieback and A.S. Tanenbaum, “Classification of RFID Attacks”, Int'l Workshop on RFID Technology, pp. 73-86, 2008.

[6] Ari Juels, “RFID Security and Privacy: A Research Survey”, Journal of Selected Areas in Communication (J-SAC), 24(2), p.p., 381-395 Sept. 2006.

[7] D. N. Duc, J. Park, H. Lee, K. Kim, “Enhancing Security of EPCglobal Gen-2 RFID Tag against Traceability and Cloning” Symposium on Cryptography and Information Security (SCIS), Hiroshima Japan, 2006.

[8] H. Chien, C. Chen, “Mutual authentication protocol for RFID conforming to EPC Class 1 Generation 2 standards”, Computer Standards & Interfaces , Vol. 29(2), pp. 254-259, Elsevier 2007.

[9] EPCglobal Class-1 Gen-2 UHF air interface specification, http://www.epcglobalinc.org/standards/uhfc1g2/uhfc1g2_1_2_0-standard-20080511.pdf.

[10] Peterson, W.W., Brown, D.T., “Cyclic Codes for Error Detection”, Proceedings of the IRE, Vol. 49(1), p.p., 228 – 235, Jan. 1961.

[11] S.A. Weis, S.E. Sarma, R.L. Rivest, D.W. Engels, “Security and privacy aspects of low-cost radio frequency identification systems”, proceedings of the 1st Int’l conference on Security in Pervasive Computing (SPC), LNCS 2802, pp. 201-212, 2003.

[12] M. Ohkubo, K. Suzki, S. Kinoshita, “Cryptographic approach to ‘privacy friendly’ tags”, RFID Privacy Workshop, 2003.

[13] A.D. Henrici, P. Muller, “Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers”, Proceedings of PerSec ’04 at IEEE PerCom, pp. 149-153, 2004.

[14] K. Rhee, J. Kwak, S. Kim, D. Won, “Challenge-response based RFID authentication protocol for distributed database environment” Int’l Conference on Security in Pervasive Computing (SPC), pp. 70-84, 2005.

[15] D. Molnar, D. Wagner, “Privacy and security in library RFID: issues practices, and architectures”, Conforence on Computer and Communications Security (CCS), 2004.

[16] J. Yang, J. Park, H. Lee, K. Ren, K. Kim, “Mutual authentication protocol for low-cost RFID, Handout of the Encrypt Workshop on RFID and Lightweight Crypto, 2005.

[17] J. Yang, K. Ren, K. Kim, “Security and privacy on authentication protocol for low-cost radio” , The 2005 Symposium on Cryptography and Information Security, 2005.

[18] S. Kathikeyan, M. Nesterenko, “RFID security without expensive crypography” , Proceedings of the 3rd ACM Workshop on Security of Ad-Hoc and Sensor Networks, pp. 63-67, 2005.