Group-ID based RFID Mutual Authentication

Yongsu PARK[1] , Younho LEE[2*]
[1]*Division of Computer Science and Engineering, Hanyang University, Korea*
[2]*IT Management Programme, SeoulTech, Korea*
yongsu@hanyang.ac.kr, younholee@seoultech.ac.kr

***Abstract*—For passive type RFID tags, EPCglobal Class 1 Generation-2 Revision is used widely as a de facto standard. As it was designed for low cost, it is quite vulnerable to security issues, such as privacy concerns. This paper[1] presents a new RFID mutual authentication protocol, which is designed to be configured on EPC Gen2 platform and to meet various security requirements while providing efficiency using PRNG (Pseudo Random Number Generator). Group-ID is used to minimize the authentication time. Security analysis of the proposed protocol is discussed.**

***Index Terms*—Authentication, Communication System Security, Identity Management Systems, RFID Tags, Security.**

I. Introduction

The RFID (Radio-frequency identification) system, which exchanges data between tags and readers via a wireless radio frequency band, is being used actively in many areas [5-7], such as the supply chain, production management, etc. As a currently popular standard of the RFID systems, EPCglobal class 1 generation 2 is used most commonly [1]. EPCglobal class 1 generation 2 standards focus on minimizing the production cost so they provide the minimal security features for the low-cost production of RFID tags.

One of the most important security features for RFID is to hide the tag's identity (ID) for anonymity, e.g., if a tag ID is exposed to anyone, it can be replicated and tracked through by scanning wherever the owner goes.

The EPCglobal Class 1 Generation-2 protocol, a de facto standard protocol, is considered to be vulnerable for security and privacy. Secure standard cryptographic primitives, such as AES or SHA-1 can be used to cope with such vulnerabilities. However, these measures increase the manufacturing cost of RFID tags considerably. This paper proposes a new mutual authentication protocol, which can be considered a variant of the EPCglobal Class 1 Generation-2 protocol since it relies on only PRNG of the standard. Moreover, Group-ID is introduced to minimize the authentication time.

This paper is organized as follows. Section 2 explains the RFID security threats and security requirements and Section 3 describes the proposed protocol. Section 4 shows the security analysis results and Section 5 concludes the paper.

II. Threats and Security requirements of the RFID Protocol

This section describes the security threats that can occur during protocol execution in a RFID system and the security requirements to counteract them. The following is a list of security threats that can occur when the protocol runs:

Eavesdropping - Because communication between the tag and reader is wireless, anyone can obtain important information of tags/readers.

Traffic analysis – The predictable response of the tag provides information that the tag and tag owner's identity can be connected.

Spoofing/impersonation attack - After collecting the communication histories, the attacker can masquerade as legitimate tags/readers.

Tracking - RFID tags that receive the query message from the reader are always supposed to respond to any message. If the tag's response is fixed or predictable to the attacker, it can cause privacy issues.

Replay attack – RFID tags may be vulnerable to a replay attack that reuses authentication information, which was captured in previous transactions.

Offline man-in-the-middle attack - An attacker can interfere with the communications between the tag and reader, and can also exchange messages (possibly modified).

De-synchronization attack - The synchronization of the state between the tag and its reader needs to be made consistent in spite of interruption, intervention, de-synchronization attacks.

The following information is the security requirements to defend against security threats:

Mutual authentication – To defend against spoofing, the tags and readers must authenticate each other.

Untraceability - The tags must not be traced such that the attacker should not be able to collect the data related to the tag itself or information stored in the tag.

Availability – It should defend against a de-synchronization attack.

Resistance to various attacks - The defense should make spoofing more difficult for both sides of the tag and reader, and must not allow interruption or interception attacks. It should be able to resist a replay attack and offline-man-in-the-middle attack.

Forward Security [2] - Once the secret in the tag is stolen, all the previous activities can be traced by searching the past logs. The forward security ensures that the latest memory in the tag does not give a hint to guess the previous outputs. Therefore, the previous activities can be protected from tampering.

Backward Security [3] - The backward security protects future tag interrogations from traffic analysis (correlation) attacks, in which the adversary uses the information leaked by the tags to find their inner state.

*Corresponding author: Younho Lee

This work was supported by the IT R&D program of MSIP/KEIT.[No.10047212, Development of homomorphic encryption supporting arithmetics on ciphertexts of size less than 1kB and its applications]

Digital Object Identifier 10.4316/AECE.2013.04002

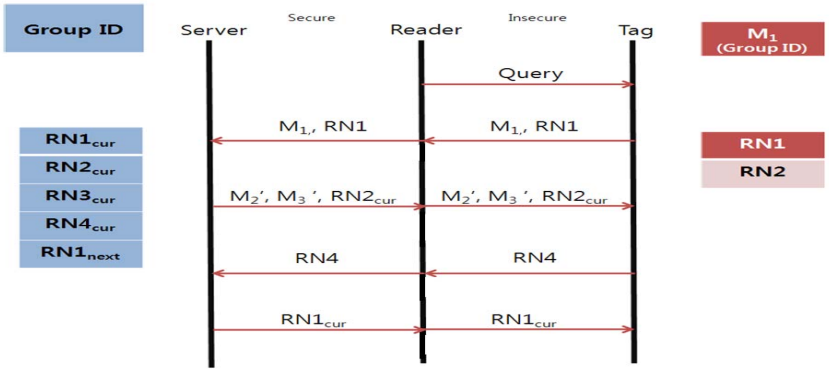

Figure 1. The overview of the proposed protocol.

III. PROPOSED PROTOCOL

This section describes the structure and operation process of the proposed protocol. The proposed protocol was designed to use only XOR (Exclusive OR) and an insecure lightweight hash function to provide security, which is called PRNG (Pseudo Random Number Generator). The RFID reader and server were assumed to be connected to each other and were protected to be safe. Whenever tag authentication is performed, the tag identification information and authentication information are randomized. Synchronization of these values is also performed simultaneously. A lightweight hash function provides weak safety. Therefore, our scheme use the method that the reader always transmits a secure random number (RND) to refresh the hash function's internal status.

The components of the proposed protocol are as follows:

① Tag's components: M_1, RN_1, RN_2, State, K.

M_1: obfuscated group ID.

RN1, RN2: Random numbers used for to change the tag's response continuously to provide untraceability.

State: internal state values of PRNG.

K: the key value used to refresh the state values. An access password can be used for K.

② The reader's components: G_{old}, G_{new}, S_{key}, $RN1_{cur}$, $RN2_{cur}$, $RN3_{cur}$, $RN1_{next}$, State, K.

G_{id} (=G_{old}): (old) Group ID that has not yet been updated.

G_{new}: newly updated Group ID.

$RN1_{cur}$, $RN2_{cur}$, $RN3_{cur}$: random numbers generated from PRNG.

$RN1_{next}$: random number generated from the updated PRNG.

RND: securely generated random number.

State: internal state values of PRNG.

K: key value used for refreshing state values. Access password of EPCglobal-Class1-Gen2-type tags can be used for K.

Figure 1 presents an overview of the proposed protocol. The detailed procedure is explained as follows.

STEP 1: R (reader) → T (tag): Query transmission.

STEP 2: T → R: After receiving the Query message, the tag sends stored values (M_1=G_{id} ⊕ S_{key}, RN1) to R. M_1 is used to identify the group ID and RN1 is used to identify the tag in the group.

STEP 3: R → T: R computes M_1 ⊕ S_{key} to obtain the group ID, $G_{id,}$ and then searches the tag information in the group. T is identified successfully if R finds $RN1_{cur}$ or $RN1_{next}$ in the database. These two cases are explained as follows.

RN1=$RN1_{cur}$: R checks whether the group ID is the old or new value. If G_{id} = G_{old}, G_{new}:= a new random value. Otherwise (G_{id}=G_{new}), G_{old}:= G_{new}, G_{new}:= a new random value. R then generates M_2', M_1', and M_3' as follows: M_2' = RND ⊕ $RN3_{cur}$, M_1' = G_{new} ⊕ S_{key}, and M_3' = M_1' ⊕ $RN3_{cur}$. Finally, R sends M_2', M_3', $RN2_{cur}$ to T.

RN1=$RN1_{next}$: R updates $RN1_{cur}$ as $RN1_{next}$. Then, R refreshes $RN2_{cur}$, $RN3_{cur}$, and $RN4_{cur}$ from PRNG. In addition, R refreshes the State value (states of PRNG) using RND, K and State: refresh(RND, K, State). R then checks whether the group ID is the old or new value. If G_{id} = G_{old}, G_{old}:= G_{new} and G_{new}:= a new random value from PRNG. Otherwise (G_{id}=G_{new}), G_{new} = a new random value from PRNG. R then generates M_2', M_1', and M_3' as follows: M_2' = RND ⊕ $RN3_{cur}$, M_1' = G_{new} ⊕ S_{key}, and M_3' = M_1' ⊕ $RN3_{cur}$. Finally, R sends M_2', M_3', and $RN2_{cur}$ to T.

STEP 4: T → R: After receiving M_2', M_3' and $RN2_{cur}$, T saves the State to State_copy in the DRAM and generates RN2 from PRNG. If RN2 is not equal to $RN2_{cur}$, communication terminates with authentication failure and T restores the State value from the State_copy. Otherwise, if RN2 = $RN2_{cur}$, T successfully authenticates R. In this case, T generates RN3 from PRNG and calculates RND=M_2' ⊕ RN3 and stores M_1=M_3' ⊕ RN3. T generates RN4 from PRNG. T refreshes the State value of PRNG from refresh (RND, K, State). T generates RN1 from PRNG and stores it in the non-volatile memory. Finally, T sends RN4 to R.

STEP 5: R → T: When R receives RN4 from T, it compares RN4 with $RN4_{cur}$. If they are identical, R successfully authenticates T. R then refreshes the State value of PRNG from refresh(RND, K, State) and generates $RN1_{next}$, copies $RN1_{next}$ to $RN1_{cur}$ and generates $RN2_{cur}$, $RN3_{cur}$, and $RN4_{cur}$ from PRNG. For resilience of the protocol, R generates/stores $RN1_{next}$ from additionally refreshed PRNG. Finally, R sends $RN1_{cur}$ to T. If T receives this, T compares it with RN1. If they are identical to each other, T successfully authenticates R.

Note that in each communication, G_{id}, is encrypted with the securely generated random number, S_{key}, which enhances the security of the protocol.

Example 1: We provide an example to help understand our protocol. Suppose that T stores the following information: M_1=0x4125, RN1=0x1111. Assume that T's information is stored in S/R as follows: G_{id}=0x0064, S_{key}=0x4141, $RN1_{cur}$=0x1111, $RN2_{cur}$=0x2222, $RN3_{cur}$ =0x3333, $RN4_{cur}$=0x4444, $RN1_{next}$=0x7777. Suppose that

RND is set to 0x4a7e. After STEP 1, in STEP 2, T sends M_1=0x4125 and RN1=0x1111 to R. In STEP 3, R finds T's G_{id}=0x0064 by computing $M_1 \oplus S_{key}$ = 0x4125 ⊕ 0x4141 =0x0064. R finds T's information in the database with the group G_{id} because $RN1_{cur}$=RN1=0x1111. R generates a new Group ID G_{new} = 0x0065, computes $M_1' = G_{new} \oplus S_{key}$ = (0x0065 ⊕ 0x4141)=0x4124, $M_2' = RND \oplus RN3_{cur}$ = 0x4a7e ⊕ 0x3333 = 0x794d, and $M_3' = M_1' \oplus RN3_{cur}$= 0x4124 ⊕ 0x3333 = 0x7217. Finally, R sends M_2', M_3', and $RN2_{cur}$ to T. In STEP 4, T generates RN2=0x2222 from PRNG and authenticates R by checking that RN2=$RN2_{cur}$. Then, T generates RN3=0x3333 from PRNG, computes $RND = M_2' \oplus RN3$= 0x794d ⊕ 0x3333 = 0x4a7e, stores $M_1 = M_3' \oplus RN3$=0x7217 ⊕ 0x3333=0x4124. T generates RN4=0x4444 from PRNG and refreshes the State value in PRNG by executing refresh(RND=0x4a7e, K, State), where State is current state value of PRNG, K is the access password in the EPCglobal-Gen2-type standard tag. Then, T generates a new RN1=0x7777 from PRNG and stores it in the non-volatile memory. Finally, T sends RN4=0x4444 to R. In STEP 5, R authenticates T by checking RN4=$RN4_{cur}$=0x4444. R then refreshes the state of PRNG using refresh() and generates $RN1_{cur}$ (=0x7777), $RN2_{cur}$ (=0x8888), $RN3_{cur}$, and $RN4_{cur}$. R stores $RN1_{next}$ =0xcccc which was from additionally refreshed PRNG. R sends $RN1_{cur}$=0x7777 it to T. After T receives this, T successfully authenticates R because $RN1_{cur}$=RN1=0x7777.

In the above example, for better understanding we intentionally chose the weak PRNG which produces easily predictable random numbers from 0x1111 to 0xcccc. However, PRNG is one of the major cryptographic primitives of EPCglobal Gen2 type tags and the standard document [1] specifies that PRNG should meet the following conditions for security.

- The probability of any RN16, 16-bit number drawn from PRNG, should meet the following condition: for any *i* ($0 \le i \le 2^{16}-1$), $0.18/2^{16} < Prob(RN16 = i) < 1.25/2^{16}$.
- For a tag population of up to 10,000 tags, the probability that any of two or more tags generate the same RN16s should be less than 0.1%.
- An RN16 drawn from a tag's PRNG 10 ms after the end of transaction shall not be predictable with a probability greater than 0.025% if the outcomes of prior draws from PRNG are known under the same condition.

Note that this protocol uses G_{id} (group ID) to reduce the search space in R. Without G_{id}, for every transaction, R should find $RN1_{cur}$ or $RN1_{next}$ in the entire content of the database. Generally R/S should support a large number of tags, which implies a slow search speed. To minimize the search speed, initially tags are grouped and each group has a group id, G_{id}. R computes $M_1 \oplus S_{key}$ to obtain the group ID and then searches the tag information in the group. The size of G_{id} is 16 bits and the search space can be reduced up to $1/(2^{16})$=1/65535.

IV. Security and Performance Analysis

First, the security of this protocol was analyzed in terms of mutual authentication, untraceability, resistance to the replay attack, resistance to impersonation attack, and resistance to an offline man-in-the-middle attack, forward/backward security, and resistance to a de-synchronization attack.

Mutual authentication - M_1 is used to identify the group in the reader, and RN1 is used to identify the tag in the reader's database. R compares RN4 with $RN4_{cur}$. If they are identical, R successfully authenticates the tag. The tag compares $RN1_{cur}$ with RN1. If they are identical, the tag successfully authenticates the reader R.

Untraceability - For every communication, internal states of the tag and the reader are updated, which implies all the messages (except for the 1st message) are different for every transaction. This means that in the communication, the group id is encrypted with the securely generated random number S_{key}. In addition, even if M_1 and $RN2_{cur}$ are exposed to an attacker, if it cannot transmit RN3 and $RN1_{cur}$, mutual authentication fails and the transaction aborts.

Resistance to the replay attack - To make the replay attack available, the attacker should be able to use the previous history of the communication, which is collected by eavesdropping. On the other hand, all the communication messages of the proposed protocol were randomized using PRNG (pseudo-random number generator) for every time the communication between the tag and reader was initiated. Therefore, the replay attack is impossible.

Resistance to the impersonation attack - The proposed protocol updates the state of PRNG for attackers to make it extremely difficult to guess the communication message values, which indicates resistance to an impersonation attack.

The resistance to the offline Man-in-the-middle attack –The attackers can eavesdrop and reuse the second and third message. On the other hand, they cannot reuse the 4th and 5th messages because every time, the tag and reader update the internal state of the PRNG and use it with the next communication.

The forward/backward security - In the proposed protocol, the tag performs the update process of the internal status of PRNG using the secure random number (RND) that is received from the reader. After the update process, it is difficult to guess the previous/next communication.

The resistance to the de-synchronization attack - The following describes the resistance of de-synchronization attacks for blocking each message in the protocol, as follows:

The case in which the attacker blocks the exchange of the second message: In this case, T and R use the same internal state for the next communication and the state is synchronized.

The case in which the attacker blocks the exchange of the third message of the protocol: R has the old and new values of the group IDs, which prevents de-synchronization attacks.

The case in which the attacker blocks the exchange of the fourth message of the protocol: For the normal case, where there is no de-synchronization attack, R matches RN1 with $RN1_{cur}$. For the de-synchronization attack, R matches RN1 with $RN1_{next}$ and continues the communication.

The case in which the attacker blocks the fifth message: Both T and R already updated the internal status of PRNG, and there is no de-synchronization.

For estimating efficiency, the proposed protocol is compared with previous work. Tables I and II show the additional memory requirements in a flyweight RFID

Protocol [3] and the proposed protocol. In the proposed scheme, compared with [3], the flash memory size is reduced to 48-bits and the RAM size is reduced to 16-bits.

Moreover, the number of gates required to implement PRNG is much smaller than AES or SHA-1. If PRNG is implemented by LAMED on the tag, the proposed protocol requires only 1,566 gates. On the other hand, if the AES is implemented on the tag, the number of gates required is at least 3,595 [4].

Recently, a large number of research achievements have been made with respect to RFID mutual authentication [2-25]. Table 3 compares the security functionalities with the other recently released lightweight protocols, which shows that the proposed protocol has all major security features.

TABLE I. MEMORY REQUIREMENTS OF THE FLYWEIGHT PROTOCOL [3].

Flash Memory			RAM		
81 bits	RN16	16 bits * 2	48 bits	RN16	16 bits * 3
	Key	48 bits			
	Flag	1 bits			

TABLE II. MEMORY REQUIREMENTS OF THE PROPOSED PROTOCOL.

Flash memory			RAM		
64 bits	RN16	16 bits * 3	16 bits	RN16	16 bits*1
	Key	16 bits			
	Flag	0 bits			

V. CONCLUSION

A secure and efficient mutual authentication protocol was designed using a pseudo random number generator for passive tags. The resistance to various security threats is analyzed. The proposed protocol uses less memory than the Flyweight Protocol and the tag identification time is smaller. Furthermore, due to use of the standard PRNG, the manufacturing cost of proposed scheme is lower than that of other RFID authentication schemes. If PRNG is implemented by LAMED on the tag, the proposed protocol requires 1,566 gates. On the other hand, if the AES is implemented on the tag, the number of gates required is at least 3,595.

REFERENCES

[1] EPCGlobal Class 1 Generation 2 Revison, available at http://www.epcglobalinc.org/standards/uhfc1g2.

[2] M. Ohkubo, K. Suzuki and S. Kinoshita, "Cryptographic Approach to Privacy-friendly Tags", in Proc. *RFID Privacy Workshop*, MIT, 2003.

[3] Burmester, M., and Munilla, J. "A Flyweight RFID Authentication Protocol", *Proc. the 5th Workshop on RFID Security*, 2009.

[4] M. O'Neill, "Low-cost SHA-1 Hash Function Architecture for RFID Tags", *Proc. RFIDSec'08*, 2008.

[5] R. John Robles and T. Kim, "A Review on Security in Smart Home Development", Advanced Science Letters, vol. 15, pp. 13-22, 2010.

[6] A. Irshad, W. Noshairwan, M. Shafiq, S. Khurram, E. Irshad, M. Usman, "Security Enhancement in MANET Authentication by checking the CRL status of Servers", Advanced Science Letters, vol. 1, pp. 91-98, 2008.

[7] N. Omer, F. Elssied, O. Ibrahim, A. Ali A.alaziz and A. Yousif, "Review Paper: Security in E-government Using Fuzzy Methods", Advanced Science Letters, vol. 37, pp. 99-112, 2011.

[8] J. Cho, S. Yeo, Sung Kwon Kim, "Securing against brute-force attack: A hash-based RFID mutual authentication protocol using a secret value", Computer Communications vol. 34, no. 3, pp. 391–397, 2011.

[9] T. Li, "Vulnerability Analysis of EMAP-An Efficient RFID Mutual Authentication Protocol", *Proc. ARES 2007*, pp. 238-245, 2007.

[10] H. Chien, C. Chen, "Mutual authentication protocol for RFID conforming to EPC Class 1 Generation 2 standards", Computer Standards & Interfaces, vol. 29, no. 2, Pages 254–259, 2007.

[11] P. Peris-Lopez, J. Cesar Hernandez-Castro, J. M. Estevez-Tapiador, A. Ribagorda, "EMAP: An Efficient Mutual-Authentication Protocol for Low-Cost RFID Tags", *Proc. OTM 2006 Workshop*, LNCS vol. 4277, pp. 352-361, 2006.

[12] P. Peris-Lopez, J. Cesar Hernandez-Castro, J. M. Estevez-Tapiador, A. Ribagorda, "M^{2}AP: A Minimalist Mutual-Authentication Protocol for Low-Cost RFID Tags", *Ubiquitous Intelligence and Computing*, LNCS vol. 4159, pp. 912-923, 2006.

[13] S. Kang, D. Lee, I. Lee, "A study on secure RFID mutual authentication scheme in pervasive computing environment", Computer Communications, vol. 31, no. 18, pp. 4248–4254, 2008.

[14] R. Paise, S. Vaudenay, "Mutual authentication in RFID: security and privacy", in *Proc. of ASIACCS '08*, pp. 292-299, 2008.

[15] H. Chien, "SASI: A New Ultralightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity", IEEE Trans. on Dependable and Secure Computing, vol. 4, no. 4, pp. 337-340, 2007.

[16] Y. Lee, Y. Park, "A New Privacy-preserving Path Authentication Scheme using RFID for Supply Chain Management", Advances in Electrical and Computer Engineering, vol. 13, no. 1, pp. 23-26, 2013.

[17] T. Li, W. Luo, Z. Mo, and S. Chen, "Privacy-preserving RFID Authentication based on Cryptographical Encoding" *Proc. IEEE INFOCOM'12*, pp. 2174-2182, 2012.

[18] Q. Yao, Y. Qi, J. Han, J. Zhao, X. Li, and Y. Liu, "Randomizing RFID Private Authentication," *Proc. IEEE PERCOM*, 2009.

[19] T. Dimitriou, "A Secure and Efficient RFID Protocol that could make Big Brother (partially) Obsolete," *Proc. IEEE PERCOM*, 2006.

[20] L. Lu, J. Han, L. Hu, Y. Liu, and L. Ni, "Dynamic Key-Updating: Privacy-Preserving Authentication for RFID Systems," *Proc. IEEE PERCOM*, 2007.

[21] L. Lu, J. Han, R. Xiao, and Y. Liu, "ACTION: Breaking the Privacy Barrier for RFID Systems," *Proc. of IEEE INFOCOM*, 2009.

[22] L. Lu, Y. Liu, and X. Li, "Refresh: Weak Privacy Model for RFID Systems," *Proc. of IEEE INFOCOM*, 2010.

[23] M. Ohkubo, K. Suzuki, and S. Kinoshita, "Efficient Hash-Chain based RFID Privacy Protection Scheme", *Proc.* Of *UbiComp Workshop Privacy*, 2004.

[24] T. Dimitriou, "A Lightweight RFID Protocol to Protect Against Traceability and Cloning Attacks", in *Proc.* Of *SecureComm*, 2005.

[25] D. Henrici, P Müller, "Providing Security and Privacy in RFID Systems Using Triggered Hash Chains", *Proc.* of *IEEE PerCom*, 2008.

TABLE III. COMPARISON OF THE PRESENT PROTOCOL WITH THE OTHER RECENTLY RELEASED LIGHTWEIGHT PROTOCOLS [5-7].

	Flyweight [2009]	Quinling-Yonghua [2008]	Choi-Lim [2008]	Lightweight Privacy Preserving~ [2010]	Proposed Protocol
Mutual Authentication	O	O	O	O	O
Resistance to Replay Attack	O	O	O	O	O
Resistance to De-synchronization	O	Don't Care (ID update X)	Don't Care (ID update X)	O	O
Resistance to offline man-in-the middle attack	O	O	O	O	O
Forward/Backward Security	O	X	X	△ (Forward only)	O
Untraceability (Tag's response change for Query)	△ After Mutual Authentication	O	O	O	O
Tag's power failure	X	X	X	X	O