TELKOMNIKA, Vol.13, No.3, September 2015, pp. 1054~1061
ISSN: 1693-6930, accredited A by DIKTI, Decree No: 58/DIKTI/Kep/2013
DOI: 10.12928/TELKOMNIKA.v13i3.1440

SLRV: An RFID Mutual Authentication Protocol Conforming to EPC Generation-2 Standard

Mu'awya Naser[1]*, Ismat Aldmour[2], Rahmat Budiarto[2], Pedro Peris-Lopez[3]
[1]Khalifa City Womens College, Higher College of Technology HCT, Abu Dhabi, UAE
[2]College of Computer Science and Information Technology, Albaha University,
P.O. Box 1998 Albaha, Kingdom of Saudi Arabia
[3]Computer Security Lab (COSEC), Computer Science Department, Carlos III University of Madrid, Spain
*Corresponding author, e-mail: muawya.aldalaien@hct.ac.ae

Abstract

Having done an analysis on the security vulnerabilities of Radio Frequency Identification (RFID) through a desynchronization and an impersonation attacks, it is revealed that the secret information (i.e.: secret key and static identifier) shared between the tag and the reader is unnecessary. To overcome the vulnerability, this paper introduces Shelled Lightweight Random Value (SLRV) protocol; a mutual authentication protocol with high-security potentials conforming to electronic product code (EPC) Class-1 Generation-2 Tags, based on lightweight and standard cryptography on the tag's and reader's side, respectively. SLRV prunes de-synchronization attacks where the updating of internal values is only executed on the tag's side and is a condition to a successful mutual authentication. Results of security analysis of SLRV, and comparison with existing protocols, are presented.

Keywords*: lightweight RFID, EPC Class-1 Gen-2, mutual authentication protocol, security analysis*

1. Introduction

Radio Frequency IDentification (RFID) is a technology highly demanded in numerous applications and domains and therefore is under a continuous and rapid development [1-4]. Securing RFID tags against security threats is considered the main obstacle facing the widespread adoption of RFID technology [5-11], where hundreds of RFID protocols have been proposed and focused on providing a secure contact between readers and tags over the insecure radio channel. Nevertheless, due to the limitations of tags in terms of circuitry (gate equivalents), storage, and power consumption, the design of an efficient and secure mutual authentication protocol presents an immense challenge. Designing security protocols is even more challenging for low-cost technologies such as the lightweight RFID security protocolswhereby the tags imposes stronger hardware and memory limitations. Among the set of risks linked to RFID technology, privacy and de-synchronization are the most challenging as the majority of designed protocols fail to offer protection against these threats.

RFID tags compliant with EPC Class-1 Generation-2 (Gen-2 in short) are based on transponders with limited resources. In detail, Gen-2 tags only support a 16-bit pseudo-random number generator (PRNG), a 16-bit cyclic redundancy check code (CRC), and bitwise operations such as XOR, AND, and OR [12].

Several protocols were proposed with the aim of securing Gen-2 tags. Unfortunately the majority of these protocols failed either to fulfill Gen-2 requirements or to satisfy the claimed security properties. For instance, [13] presented a protocol using a PIN password to securize the communication. This protocol suffers from several attacks as the ones mentioned in [14] and [15]. First, it was vulnerable to a de-synchronization attack as a consequence of the weak updating mechanism of the secret keys and shared values. Secondly, it does not offer protection against replay attacks and a passive attacker can reuse tokens from previous sessions. Thirdly, it was susceptible to a traceability attack since tags respond with the same value every time – in this attack, the attacker has to intercept the updating message and the tag would respond with a constant value.

Received December 26, 2014; Revised April 20, 2015; Accepted May 11, 2015

Yeh et al.'s protocol [16] aims to secure EPC Class-1 Gen-2 standard. Similar to many previously proposed protocols, it can be categorized under the class of lightweight mutual authentication protocols, following the classification proposed in [17]. In this category, it is assumed that tags can generate a random number but they do not have the computational resources to support on-board hash function. On the other hand, and similar to other lightweight RFID authentication protocols, Yeh et al.'s scheme is designed with a new parameter representing a database index value.

1.1. Vulnerability of Yeh et al.'s Protocol

Naser et al. [18] showed how the protocol is vulnerable against de-synchronization and impersonation attacks. The attacks can be conducted by a malicious reader, which mainly forwards message and does simple modifications exploiting the weaknesses of the bitwise XOR operations.

1.2. De-synchronization Attack

Yeh et al.'s protocol was designed using two sets of authentication and access keys to combat DoS attack, which causes a de-synchronization state between the tag and the server. The authors in [16] criticized the fact that its predecessor scheme (i.e., Chien and Huang's protocol [14]) updated the key values (K_{old} and P_{old}) on every successful mutual authentication session at the database side. Motivated by this, Yeh et al. proposed to add a validation criterion for this updating mechanism to solve the de-synchronization attack, which Chien and Chen's protocol suffer, and is based on the usage of the new values of D, E, and C_i. Nevertheless, despite these validation tokens, we, in this paper, show how replay attacks can de-synchronize the protocol. The used adversary (malicious reader) has to be able to interrupt and forward messages only, and it does not need to have the capability to communicate with the database. This adversary will execute two session procedures in one session. That is, both communication sessions are executed almost in parallel but with only a slight difference in time:

In the $(i+1)^{th}$ authentication session, the malicious reader will intercept the last message from the database and throw away M_2 message to keep the tag using the same index value C_{i+1}. At the same time, the database will update its local parameters, specifically C_{old} would be C_{new}, and C_{i+1}, and its C_{new} would be C_{i+2}.

In a slightly posterior session (almost a parallel session), the malicious reader will resend a new Message 3. However, instead of containing (V, M1, D, C_i, E, N_R), it will send (V, M1, D⊕RND, C_i, E⊕RND, N_R), which will allow the database to understand that it is a new session. These values (i.e., V, M1, D⊕RND, C_i, E⊕RND, and NR) will facilitate the tag to be authenticated by the database because N_R continues to represent the same values from the eavesdropped session. N_T will become NT⊕RND, which is correctly used in D and E messages. Due to modified Message 3 sent by the reader, the database will update its C_{new} value based on the C_x (in this case, X=old) from C_{i+2} to C_{i+3}. At the same time, the malicious reader will forward the stored M2 message to the tag, causing the tag to update its values from (K_{i+1}, P_{i+1} and C_{i+1}) to (K_{i+2}, P_{i+2} and C_{i+2}).

At this step the tag will store C_{i+2} as index value, and the database will keep the values C_{i+1} and C_{i+3}. Therefore, the tag and the database lost its synchronization and this is permanent. In fact, the tag can never be identified because the search index stored into its memory is different from the two indices (old and new) stored in the database.

1.3. Impersonation Attack

Tag impersonation attack is conducted by a dishonest reader. The key points of this attack are based on the use of N_T nonce in both D and E tokens and the abusive use of the bitwise XOR operations. Bitwise operations like XOR are linear functions, which are vulnerable to active and passive attacks. The proposed attack is sketched below:

a) $(i + 1)^{th}$ authentication phase

(1) R → Tag_x: N_R

(2) Tag → R: M1, D, C_i, E

- M1= PRNG (EPC_S⊕N_R)⊕K_i
- D = N_T⊕K_i
- E = N_T⊕PRNG(C_i⊕K_i)

(3) R → DB: V, M1, D, C_i, E, N_R

(4) DB → R: M2, Info
(5) R → Tag_x: Attack

The attack can be performed using two methods. The first is by preventing the reader from forwarding any messages to the tag. Alternatively, the adversary can interrupt the last message and send a fraudulent message containing an incorrect value of M2. At this point, the targeted tag is isolated and the malicious reader can replace and impersonate the original tag by computing simple bitwise XOR operations as described in the following.

b) $(i + n)^{th}$ authentication phase (n>2)

Basically the fraudulent reader simulates that the tag always incorrectly receives the message M2. Therefore, the updating phase is not run in the tag and previous M1 message is valid. M1, D, E, N_R, and V are the picked values of a previous legitimate session. After the reception of M2, the reader block this message and simulates the tag incorrectly received M2. After that, the fraudulent reader sends M1, D⊕RND, E⊕RND, N_R, V, where RND represents an arbitrary random value. The tag is authenticated since M1 is legitimate. The random number N_T' associated to this session is the bitwise XOR between N_T and RND. We sketch the process below:

DB→R: M2, Info Fake R →DB: M1, D⊕RND, E⊕RND, N_R, V

The proposed attack can be executed indefinitely as the original scheme does not assume any threshold for the number of times the M2 message can be interrupted, altered, or incorrectly received.

2. Research Method

In an accumulative effort to enhance the security of Yeh et al.'s protocol, we propose two possible solutions for the desynchronization threat. The first is the creation of two extra fields for each tag's record in the database as a short memory for random numbers (N_R, N_T) generated in the $(i+1)^{th}$ session. The short memory using these two extra fields (N_{R_last}, N_{T_last}) indicates that they will be overwritten at every successful mutual authentication $(i+2)^{th}$ session. The second solution involves the modification of the formulas for D and E values, executed by the tag, as not to be able to misuse or manipulate these during transaction (e.g., by using a non-rectangular function such as a rotation function). However, these proposed solutions are not completely effective; other attacks can be configured based on the original protocol plot design, which depends on updating values of common secrets between tag and backend database.

Motivated by the abovementioned disadvantage in enhancing Yeh et al.'s protocol, a new protocol was developed utilizing the strengths and addressing the weaknesses of existing protocols, particularly the vulnerabilities in the said protocol (See Figure 1). The goal in creating the new protocol was to produce a lightweight one with higher security level and lower computational power requirements for the EPC Class-1 Generation-2 standard for RFID tags. The protocol presents transactions of the original stored data combined with random values, and encapsulated in shell values capable of transporting hidden data between the tag and the reader without compromising or revealing this data. Based on its characteristics, the proposed protocol is named as Shelled Lightweight Randomized Value (SLRV), which focuses on securing the channel between the tag and the reader, the reader and the database, and vice-versa. We introduce the following notations for the protocol:

a) Ψ : encryption value holding the EPC_S
b) K_e : encryption key with fixed value for all tags stored in the database
c) N_i : random number generated by the database at every $(i)^{th}$ session
d) R_{temp} : temporary value calculated by the database at every $(i+1)^{th}$ session
e) K1, K2: two distinct secret keys with fixed values for each tag stored in the tag and the corresponding record in the database We recommend that these two secret keys (K_1, K_2) be changed from time to time to maintain a higher security level.

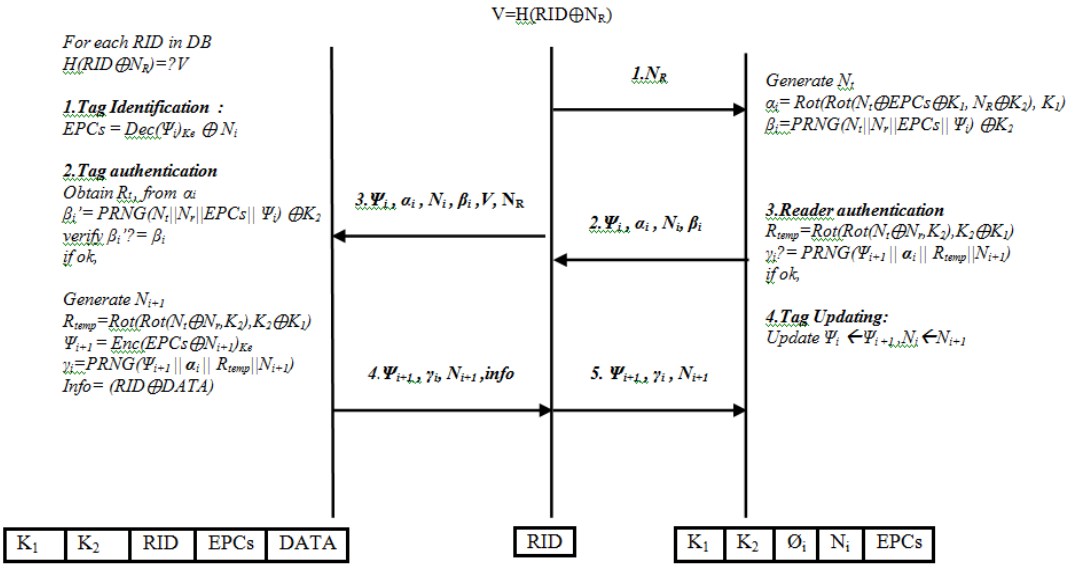

Figure 1. Shelled Lightweight Randomized Value (SLRV) protocol

The protocol consists of the following phases:

2.1. Initialization Phase

For each tag, denoted as Tag_x, the manufacturer randomly generates two secret keys and EPC_S values and stores these in the tag in a corresponding record in the database identified by EPC_S. The manufacturer likewise generates an encryption key K_e and stores it in the database, generates a random number and stores it in the tag as N_i, and generates an encryption value using the formula $\Psi=ENC(EPC_S \oplus N_i)_{Ke}$ and stores it in the tag as Ψ.

In the (i + 1)th tag authentication phase

1) R →Tag_x: N_R

The reader generates a nonce random N_R to the tag as a challenge. Upon receiving N_R, the tag generates a random number N_T to be used in the following formulas:

a) $\alpha_i=Rot(Rot(N_T \oplus EPC_S \oplus K_1)N_R \oplus K_2),K_1$

b) $\beta_i=PRNG(N_T||N_R||EPC_S||\Psi_i) \oplus K_2$

Where α_i is used to hide N_T and β_i is used to check the message integrity by the database.

2) Tag →R: Ψ_i, α_i, β_i and N_i

When the reader receives the message, it will compute V value using $V=H(RID \oplus N_R)$, which is the hashed value of the reader's ID (RID) XORed with N_R. The reader then forwards it with contents of Messages 1 and 2 to the backend database for the purpose of tag identification.

3) R → DB: Ψ_i , α_i , N_i , β_i ,V, and N_R

Once Message 3 is received, the database performs the following operations sequentially, where each is conditioned to the success of its predecessor operation or else will abort the session:

2.2. Reader Authentication Phase

The database iteratively picks up each stored RID and computes $H(RID \oplus N_R)$ to authenticate the reader based on the value of V.For each RID in DB test the following formula: $H(RID \oplus N_R)$=? V

2.3. Tag Identification Phase

The database extracts EPC_S value using the following formula:

$EPCs=Dec(\Psi_i)k_e \oplus N_i$

The reader uses this formula to extract EPCs by decrypting Ψ_i using K_e and N_i, and locks up the tag's corresponding secret keys in the database. Subsequently, it starts the mutual authentication phase only if the EPCs are verified.

2.4. Tag Authentication Phase

The database uses the values it acquired (EPC_S, K_1, K_2) and resolves A_i shell by inversing the function A_i=ROT(ROT($N_T \oplus EPC_S \oplus K_1$, N_R^K_2), $N_R \oplus K_1$) to extract the value of the random number N_T. Next, the reader checks the integrity of the message by verifying B_i value using the following formula:

$$PRNG(N_T||N_R||EPCs||\Psi_i)\oplus K_2 =? \beta_i$$

2.5. Tag updating Phase

The database authenticates the tag, creates a new random value (N_{i+1}), and uses K_1, K_2, and N_T to calculate R_{temp} using the following formula:

$$R_{temp}=Rot(Rot(N_t \oplus N_R,K_2),K_2 \oplus K_1)$$

The database then creates new tag parameters for the values (Ψ_{i+1} , γ_i, and Info) sequentially using the following:

a) $\Psi_{i+1} = Enc(EPC_S \oplus N_{i+1})K_e$
b) $\gamma_i = PRNG(\Psi_{i+1} || \boldsymbol{\alpha}_i || R_{temp} || N_{i+1})$
c) Info= (RID⊕DATA)

Where γ_i is used to check the message integrity by the database.

1) DB→R:Ψ_{i+1} , γ_i, N_{i+1} , and Info

When the reader receives Message 4, it obtains DATA from the info field by inversing the formula DATA=info⊕RID using the RID stored in it. It forwards Ψ_{i+1}, γ_{i+1}, N_{i+1} to the tag.

2) R →Tag: Ψ_{i+1}, γ_{i+1}, N_{i+1}

When Message 5 is delivered, the tag recalculates N_{temp} using the following formula:

$$R_{temp}=Rot(Rot(N_T \oplus N_R,K_2),K_2 \oplus K_1)$$

Next, the tag checks the integrity of the message by verifying G_i value using the following formula:

$$PRNG(\Psi_{i+1} \oplus A_i \oplus R_{temp}) =? \gamma_i$$

If γ_i value verification failed, the tag presumes manipulation in the message and therefore aborts the session. Otherwise, the tag completes mutual authentication, authenticates the database, and concludes the session in the final phase (Tag Updating) by updating its values and overwriting the old values as the following:

a) $\Psi_i=\Psi_{i+1}$
b) $N_i=N_{i+1}$

As illustrated in Figure 1, two shells (α and γ) are generated in every session. This makes the embedded values difficult to predict, and these values would be useless if obtained after the session is terminated. Furthermore, there are three verification tokens — V, β, and γ — that allow the system to terminate unsuccessful session in four positions: EPCs lock up in reader authentication, tag identification, tag authentication, and tag updating. These tokens start a new session in another timeframe.

3. Security Analysis

We conducted security analysis against the most relevant threats discussed in previous literature. Analysis was conducted by investigating each attack and its requirements and properties in the following categories:

User data confidentiality: Secret keys K_1 and K_2 are carefully hidden inside α, β, and γ. In every new session, the keys are mixed with two different random numbers N_T and N_R.

Moreover, if any of the sub-messages in α, β, or γ was broken, tag identity will remain anonymous to the adversary. This is because the tag EPCs was XORed with a random number and subsequently encrypted using a secret key that exists only in the reader's database. Therefore, the tag's identity can be recognized only by legitimate readers.

Tag anonymity: Sub-messages are updated in every session's transaction, and tag-reader-database messages are mixed with random numbers. As a result, the adversary is unable to recognize the tag's location or trace it unless the adversary continues to interrupt the communication between the same tag and any legitimate reader; this leads to the transmission of the same message values of Ψ_i and N_i every time. This scenario was not considered of any considerable value and had been ignored in most previous studies in the domain since the tag was unable to randomize itself due to the limited recourse. An in-depth analysis of all these scenarios has been given in detail in [19].

Mutual authentication and data integrity: Our mutual authentication protocol can be performed only between legitimate readers and legitimate tags owing to the sub-messages α, β, and γ; these are generated using the common secret keys K_1 and K_2, which are only held in the tag and backend database and not communicated in plain values over an open channel. In addition, verifying the values of β and γ composed by the tag and the database, respectively, provides strong data integrity validation.

Forward security: It is not possible for an adversary to infer any data patterns from past communications among the tag, reader, and database. This is because any previous data sent in one session will have no meaning in any subsequent sessions; each message is based on a random number that is checked for integrity for the session it was created in. Therefore, the integrity check will recognize that the value is not created during the same session, and it consequently will terminate the session unsuccessfully. Moreover, Keys K_1 and K_2 are not dropped. However, they are difficult to obtain and can be changed frequently, rendering this attack quite impossible. Assuming the tag is somehow compromised; there remain several unknown data variables in the server, such as K_e.

Resistance to replay attacks: An adversary may eavesdrop on any of the exchanged messages. However, it would not be useful to send it back to either the database or the tag. This is because each message is based on random numbers that are changed in every successful authentication session. Accordingly, a replay attack can be detected immediately once the message is received by either the tag or the database.

Data-update-confirmation and desynchronization: Majority of recent authentication protocols require updating the secret keys' values between the tag and reader. Cases where transmitted data had been modified or even interrupted lead to desynchronization. A desynchronization attack is the first vulnerability that commonly appears in all current protocols. In our protocol, the tag does not require updating of its local data in other entities. Moreover, even if any of these messages are modified or interrupted, any modification can be discovered easily when the values of V, β_i, and γ_i are verified. Interruptions will not make any difference because tag data will be updated only after receiving and verifying the last message. Thus, the reader is never affected and always obtains the original EPCs for every new session.

Resistance to man-in-the-middle attacks and disclosure attacks: Man-in-the-middle attacks can not affect SLRV protocol since all exchanged messages are verified and all modifications can be simply detected. Similarly, in a disclosure attack when an attacker makes changes in any message sent from database to tag or vice versa, SLRV protocol will detect any alteration and ignore the message.

A significant aspect of SLRV is that it is based on classical cryptography primitives on the database server's side. At the same time, the protocol is based on lightweight cryptography on the tag's side. More precisely, the protocol uses a combination of triangular and non-triangular functions. Non-triangular functions use a double-rotation function instead of the simple XOR function to obtain a greater diffusion effect and combat cryptanalysis of the protocol [12, 19]. Utilizing computational capabilities on the database server's side for using classical cryptography primitives and using a triangular and non-triangular functions collectively provide a higher security and protect against all known kinds of disclosure attacks that other protocols fail to defend against. Additionally, a meaningless message cannot affect the tag or reader, but only results in ending the current session unsuccessfully, enabling a new session to begin in another timeframe.

Database loading: Finally, to cover all possible threats to SLRV, an adversary can perform database loading attacks by modifying any of the values in the message forwarded from the tag. This will either result in performing excessive EPCs lock-up processes in the database for invalid EPCs when manipulating the Ψ_i or N_i values, or in the verification of a manipulated PRNG value(s). However, this attack will not produce a significant effect because the SLRV uses a binary search algorithm for EPCs lockup, which is moderately fast where the lockup complexity is *O*(*log n*). Furthermore, the database in SLRV maintains the assumption that all values are fixed once added, EPCs are serialized, and data are indexed. Therefore, it results in a complexity value of O(1) for EPCs lockup, which minimizes the effect of DoS attacks.

Comparison with related protocols, such as Juels Protocol, Duc Protocol, etc., is summarized in Table 1. SLRV covers well all aspects of security being considered from confidentiality to the database loading aspect.

Table 2. Comparison of Lightweight Authentication Protocol

	Confidentiality	Anonymity	Authentication	Forward Security	Replay Attacks	Desynchronization and DoS	MIMA	DB Loading
Juels protocol	o	x	o	x	x	x	x	x
Duc et al.	o	o	x	x	x	x	x	x
Lies et al.	x	x	x	x	x	x	x	x
Sun and Ting	x	o	o	o	o	x	o	x
Karthikeyan and Nestenko	o	x	x	x	x	x	x	x
Chien and Chen	o	o	x	x	x	x	x	x
Yeh et al.	o	o	x	x	x	x	x	o
SLRV	o	o	o	o	o	o	o	o

4. Conclusion

We have proposed SLRV as a new lightweight authentication protocol capable of providing transactions of shelled values able to transport encapsulated encrypted private data between the tag, the reader, and the database without compromising the data. This guarantees privacy and anonymity of the tags' holder. The main advantage offered by this protocol is that each session is considered an atom entity where no data from previous sessions are stored after session termination. In addition, no data values can be changed on the tag's side until all transactions have been executed and validated successfully, ensuring data integrity on the RFID tag, reader, and backend database entities at all times. Comparing to previous protocols in the lightweight RFID field, the proposed protocol (SLRV) covers all aspects of security.

Additionally, we urge protocol designers to check their protocols against compatibility with standards carefully (e.g., EPC-C1G2 or ISO/IEC 18006-C), bearing in mind that the design of a secure and efficient RFID authentication protocol is not a simple issue but a complicated challenge that requires in many cases a trade-off between objectives.

References

[1] Kim MC, Kim CO, Hong SR, Kwon IH. Forward-Backward Analysis of RFID-Enabled Supply Chain Using Fuzzy Cognitive Map and Genetic Algorithm. *Journal of Expert Systems with Applications*. 2008; 35(3): 1166-1176.

[2] Sun Q, Zhang H, Mo L. Dual Reader Wireless Protocols For Dense Active RFID Identification. *International Journal of Communication Systems*. 2011; 24(11): 1431-1444.

[3] Cho K, Pack SH, Kwon TY, Choi YH. An Extensible and Ubiquitous RFID Management Framework Over Next Generation Network. *International Journal of Communication Systems*. 2009; 23(9-10): 1093-1110.

[4] Chen YN, Fang F, Ding DH, Zhu XH, Yang YK. Organic RFID Based on Traceability System of Rice Supply Chain. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2014; 12(5): 3769-3776.
[5] Deng M, Zhu W. Desynchronization Attacks on RFID Security Protocols. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2013; 11(2): 681-688.
[6] Luo H, Liu R, Wang Y, Chen J. Security Evaluation for RFID System: Security Evaluation Index Architecture and Evaluation Model. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2014; 12(6): 4557-4562.
[7] Weis SA, Sarma SA, Rivest RL, Engels DW. Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems. In: Hutter D, Müller G, Stephan W, Ullman M. *Editors*. Security in Pervasive Computing. Berlin Heidelberg: Springer; 2004: 2802, 201-212.
[8] Juels, A. RFID security and privacy: A Research Survey. *IEEE Journal on Selected Areas in Communications*, 2006: 24(2), 381-394.
[9] Lim, C. H., Korkishko, T. mCryptontion a Lightweight Block Cipher For Security of Low-Cost RFID Tags and Sensors. In: Song J-S., Kwon, T-Y., Yung. M. Editors. Information Security Applications. Berlin Heidelberg: Springer; 2006: 243-258.
[10] Li JS, Liu KH. A Hidden Mutual Authentication Protocol for Low Cost RFID Tags. *International Journal of Communication Systems*. 2011; 24(9): 1196-1211.
[11] Peris-Lopez P, Hernandez-Castro JC, Tapiador JME, Ribagorda A. Advances in Ultralightweight Cryptography for Low-Cost RFID Tags: Gossamer Protocol. In: Chun KI, Sohn K, Yung M. *Editors*. Information Security Applications. Berlin Heidelberg: Springer; 2009: 56-68.
[12] Chien HY. SASI: A New Ultralightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity. *IEEE Transactions on Dependable and Secure Computing*. 2007; 4(4): 337-340.
[13] Duc DN, Lee HR, Kim KJ. *Enhancing Security of Epcglobal Gen-2 Rfid Tag against Traceability and Cloning*. In: Cole PH, Ranasinghe DC. *Editors*. Networked RFID Systems and Lightweight Cryptography. Berlin Heidelberg: Springer; 2008: 269-277.
[14] Chien HY, Huang CW. Security of Ultra-Lightweight RFID Authentication Protocols and Its Improvements. *ACM SIGOPS Operating Systems Review*. 2007; 41(4): 83-86.
[15] Sun HM, Ting WC, Wang KH. On the Security of Chien's Ultra-Lightweight RFID Authentication Protocol. *IEEE Transactions on Dependable and Secure Computing*. 2009; 8(2): 315-317.
[16] Yeh TC, Wang YJ, Kuo TC, Wang SS. Securing RFID Systems Conforming to EPC Class 1 Generation 2 Standard. *Journal of Expert Systems with Applications*. 2010; 37(4): 7678-7683.
[17] EPC global. Class 1 Generation 2 UHF Air Interface Protocol Standard "Gen 2" Version 1.2.0. 2008. Available on EPCglobal website: http://www.epcglobalinc.org/standards/.
[18] Naser M, Aldmour I, Budiarto R, Peris-Lopes P. *Vulnerability Analysis of a Mutual Authentication Protocol Conforming to EPC Class-1 Generation-2 Standard*. Proceedings of the 1st International Conference on Electrical Engineering, Computer Scince and Informatics (EECSI). Yogyakarta. 2014: 173-176.
[19] Avoine G, Oechslin P. RFID traceability: A multilayer problem. *Journal of Financial Cryptography and Data Security*. 2005; 3570: 125-140.