World Academy of Science, Engineering and Technology
International Journal of Information and Communication Engineering
Vol:2, No:6, 2008

The Tag Authentication Scheme using Self-Shrinking Generator on RFID System

HangRok Lee, and DoWon Hong

Abstract**—**Since communications between tag and reader in RFID system are by radio, anyone can access the tag and obtain its any information. And a tag always replies with the same ID so that it is hard to distinguish between a real and a fake tag. Thus, there are many security problems in today's RFID System. Firstly, unauthorized reader can easily read the ID information of any Tag. Secondly, Adversary can easily cheat the legitimate reader using the collected Tag ID information, such as the any legitimate Tag. These security problems can be typically solved by encryption of messages transmitted between Tag and Reader and by authentication for Tag.

In this paper, to solve these security problems on RFID system, we propose the Tag Authentication Scheme based on self shrinking generator (SSG). SSG Algorithm using in our scheme is proposed by W.Meier and O.Staffelbach in EUROCRYPT'94. This Algorithm is organized that only one LFSR and selection logic in order to generate random stream. Thus it is optimized to implement the hardware logic on devices with extremely limited resource, and the output generating from SSG at each time do role as random stream so that it is allow our to design the light-weight authentication scheme with security against some network attacks. Therefore, we propose the novel tag authentication scheme which use SSG to encrypt the Tag-ID transmitted from tag to reader and achieve authentication of tag.

Keywords**—**RFID system, RFID security, self shrinking generatior, authentication, protocol.

I. Introduction

THE typical Radio Frequency Identification (RFID) system consists of a tag and a reader. The reader sends a RF signal for identification information to the tag. The tag responds with the respective information. The tag and reader communicate over an RF channel. RFID systems can be classified according to the tags being used whether the tags are active or passive. Active tags have an on-board power source. Passive tags, however, obtain their operating power from electromagnetic wave generated by the reader without a battery. They are smaller, lighter, and less expensive than active tags. Both active and passive tags have limited memory capacity and restricted computing power to keep the cost manageable.

Compared to bar code system, the RFID system enables to collect and process the data effectively. It is possible to be operated in non line of sight without manual controls for RF communication. It can also read hundreds of tags at the same time or read selective tags, as determined by identification data residing on the tag. Owing to these advantages of the RFID system, it is already widely used in access control system, animal tracking, toll system, electronic payment and so on. Furthermore it will seem to replace the bar code system over all consumer goods in near future. The goods tagged make it possible to trace exactly the supply chain of them and easy to verify the originality of them. So it is expected to be used dramatically.

But some researchers and protectors of individual privacy asserted that there are the serious security problems of using the RFID system. For example, an ability of easy acquisition tag's data cause of a reader cause able to trace the consumer's position by tracing the tag of goods. Products belonging to someone can be itemized without the notice of him, and its results can be abused. Also with access control systems, malicious reader can collect tag data and forges the tag using the data collected, which make it possible to enter into forcibly. These problems come from permission of unauthorized access to information. If a tag contains important data in its memory, then this unauthorized access will cause very serious security problems.

These problems mentioned upper can be solved with encryption and authentication of communication data between the reader and the tag. However especially passive tags have very restricted resource to thousands of memory and logic gates, and it is not enough to perform general standardized crypto algorithms or secure protocols. In recent, the result of implementing of AES symmetric key crypto algorithm for passive RFID authentication is presented in CHES 04[1]. It is implemented with 3,596 gates and it needs 1,106 operation clocks for one encryption of 128-bit data.

In this paper, we proposed the simply authentication protocol for passive tag using self-shrinking generator (SSG), which is designed by W. Meier and O. Staffelbach in Eurocrypto 94. Our authentication protocol does not need an additional pseudo random generator because the output stream of the SSG can be used for pseudo random generator. It is secure for basic network attacks such as eavesdropping and reply attack because tag ID is encrypted with SSG and the reader authenticate it on communication between the tag and reader. It also uses only one LFSR logic and selection logic, so it can be implemented by very light-weight system.

This paper is organized as follows. In section 2, we introduce briefly the related works and in section 3, system model for

H. R. Lee is with Electronics and Telecommunications Research Institute (ETRI), 161 Gajeong-Dong Yuseong-Gu, DaeJeon, 305-350, Korea (phone: +82-42-860-1269; fax: +82-42-860-5611; e-mail : neogauss@etri.re.kr).

D. W. Hong is with Electronics and Telecommunications Research Institute (ETRI), 161 Gajeong-Dong Yuseong-Gu, DaeJeon, 305-350, Korea (phone: +82-42-860-1327; fax: +82-42-860-5611; e-mail: dwhong@etri.re.kr).

protocol design and assumptions are presented. In section 4, we consider the thread model and security requirement on RFID system. In section 5, SSG is explained and we describe main the tag authentication scheme using the SSG in section 6. And, in section 7, we analyze about our scheme, and finally, concluding remarks and future works are presented in section 8.

II. Related Works

Recently, several research results to enhance the privacy of RFID system are proposed. In [4], Juels, Livest, and Szyldo developed the blocker tag as a way of protecting consumer privacy. The blocker tag can disrupt all RFID tag reading in its vicinity. The RFID system typically relies on three-walking or ALOHA protocol as distinguishing techniques, and the blocker tag broadcasts both '0' and '1' for a response, which creates a collision in the broadcast bit within three-walking or ALOHA protocol. Therefore a reader can't both read and distinguish the tag ID when the blocker tag is activated. If blocker tag is used with privacy zoning method, then it can block selectively subsets of ID codes. The blocker tag, however, offers only privacy without providing other authentication functions.

Juels suggested minimalist cryptography for authentication and privacy in RFID tags in [2]. In this paper an RFID tag stores a short list of random identifiers or pseudonyms, which is known by authorized verifiers to be equivalent. Each time the tag communicates with a reader, it emits the next pseudonym in the list and used that pseudonym as one-time pad. Using different pseudonyms disturbs unauthorized reading and tracking about the tag. Also Jeuls suggested a throttling mechanism to prevent a rogue reader collecting many pseudonyms at short intervals. The throttling mechanism forces the tag to respond after a temporary delay. Because an authorized reader knows pseudonyms in tag, mutual authorization can be achieved by communicating with these pseudonyms between tag and reader. After successful mutual authentication, the reader generates new pseudonyms and transmits them to tag. Then, the tag updates the pseudonyms by XORing them with old pseudonyms. However, if a malicious attacker collects *m* number of data used for updating pseudonyms each session, then this system may be broken.

In [5], Weis, Sarma, Rivest, and Engel suggested the hash-lock scheme. They used hash function to control read-access of a tag and defined the two states of a tag. The first state is "locked state", in which a tag responds with only meta-ID to a reader, and the second state is "unlocked state", in which a tag can transmit its ID responding to a reader. In the scheme suggested, a reader hashes a random key data; the hashed data is used for meta-ID of a tag, and saves it in the data base. Then, the reader transmits the meta-ID to tag. The tag received the meta-ID saves it in its memory and switches the state to the locked state. To resolve the locked state of tag into unlocked state, the reader transmits a query that it wants to unlock the tag and then the tag responds with its meta-ID. The reader received the meta-ID brings the tag ID and key associating with the meta-ID and transmits again the key to the tag. The tag hashes the key received and compares the result with meta-ID saved in its memory whether two values are equal or not. If two data are equal then it switches the state to unlocked state and transmits its ID data. This scheme, however, has handicaps that an attacker can eavesdrop on last phase to acquire the tag ID and it is difficult to implement a hash function over passive tag owing to its hardware complexity.

Juels and Pappu considered inserting an RFID tag into Euro banknotes in [6]. This scheme was proposed as a tracking mechanism for law enforcement agencies by European Central Bank. They suggested the privacy protection scheme that tag contains the ciphertext of the serial numbers of Euro banknotes and only authorized readers can read and re-encrypt it. This ciphertext is re-encrypted compulsorily by special computational device in a shop. The crypto algorithms used for re-encryption is the modified ElGamal crypto system, which is able to re-encrypt without a public key. Because Juels' scheme assumes a single-verify entity, it can't be applied to general commercial systems or multi-verifier systems.

III. System Model and Assumption

Before presenting details of our tag authentication scheme, we will describe an RFID system applied to our scheme and some assumptions.

The RFID system consists of many different tags and a reader and a trusted backend database server, which contains information about tags and communicates with the reader. We assume that the tag is the passive tag, as mentioned above, which obtains its operating power from electromagnetic wave generated by the reader without a battery, and has rewritable memory with hundreds of bit and hardware with thousands of logic gates for implementing a light-weight crypto algorithm. We also suppose that the communication between the reader and the database system is secure through the existing authentication and secure protocols such as PKI, SSL, IPSec, and TLS.

At the first stage, a reader does not have any information about a tag but have only access authority to be able to read the tag data. When a Reader accesses a database system and brings tag data to itself, this access authority is used. The Self-Shrinking Generator, which we will call SSG, and addition module which is the core of our authentication scheme is implemented at tag. SSG and hash algorithm is implemented at reader, and database only has a SSG algorithm.

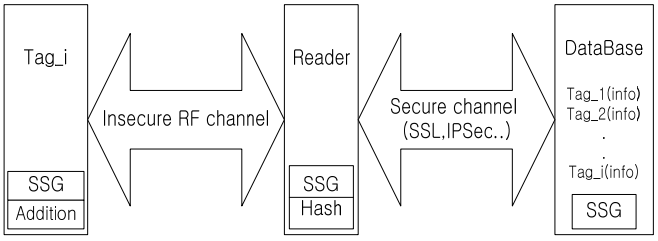

Fig. 1 Secure RFID System Model

Here, the above defined database does not contain the tag ID and does not deal with directly the tag ID. Since currently developing Information Service(IS) or Object Name Service(ONS) servers deal with the 1~10 million of Tag ID information at a time, so imposition of security operations on the IS or the ONS system is seems to be very difficult. And also, there are several RFID systems that do not require the security function. Hence, to easily add and remove security function in general RFID system, the above Secure RFID System is independent with currently existing RFID System. In other words, the Secure RFID System assists the existing RFID system to easily add security ability. The union between the existing RFID system and the Secure RFID system are as following.

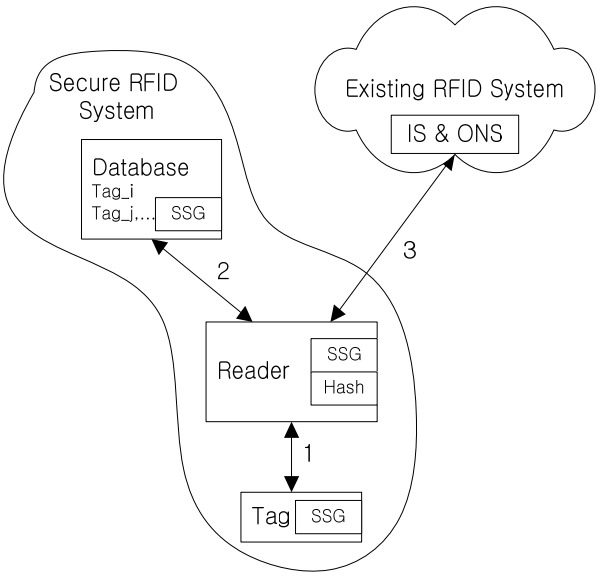

Fig. 2 Graphical Depiction of the union between Secure RFID System and Existing RFID System

IV. Threat Models and Security Requirements

We now describe threat models and security requirements. Firstly, we consider the threat models.

To design the secure protocol in RFID system, the following attack models must be assumed. And the designed protocol should satisfy the security properties against the below attack models. But, here we do not consider a physical attack like tempering attack.

- **Replay attack**: This is that the adversary can eavesdrop the response message from tag and reader, and the adversary retransmits the eavesdropped message to the reader at some later time. So the adversary wishes to pretend as if it is a legitimate RFID tag without information of the Tag's secret.
- **Forgery attack:** This implies that the adversary can forgery any response message from tag to cheat the legitimate reader.
- **Man in the middle attack:** Between the reader and the tag, the adversary can impersonate as a legitimate Reader and get the information from Tag, and she can impersonate as the legitimate Tag responding to Reader. Thus, the adversary cheats both sides as if she were the legitimate parts.

And another aim of attack is to produce a response to a challenge between tag and reader. Such a success of the adversary might be achieved with or without recovering the secret data shared by tag and reader. So our protocol's aim is to protect the unique ID of tag and secure against the specified threat models. To protect the above described attacks, we should consider the following security requirements.

- **Data Confidentiality:** Confidentiality means keeping information secret from unauthorized parties. It is typically achieved with encryption. In Case of RFID system, the Tag's ID must be kept secure to guarantee tag privacy. So, even if the data transmitted during interacting tag and reader are eavesdropped by the adversary, it must be meaningless to the adversary.
- **Data Integrity:** If an adversary modifies a data from a legitimate Tag while the data is in transit, the Reader should be able to detect this modifying. To detect this modifying, a message transmitted from tag must be included the message authentication code or message hash value.
- **Tag Privacy:** Although the *meta_ID* or *pseudonym* of Tag is known, the adversary must be unable to map from *meta_ID* or pseudonym to unique Tag's ID. Thus, *meta_ID* or *pseudonym* of tag must be randomly generated without related to the Tag's ID.
- **User Privacy:** Although the adversary acquires the data stored in the tag, he must be unable to trace the data back through past events in which the tag was involved. Namely, the adversary must do not trace to movement of tag owner by tracking the ID of tag.

Our scheme supports the confidentiality and integrity of tag ID and the tag privacy, but user privacy is not supported in our scheme. To support user privacy, tag has to always reply random value and not fixed value. It is not difficult to generate random value at every time in tag. But in database, there is hard problem which search the correct tag ID associated with random value transmitted from tag. For instance, as shown in [7] and [8], the most of protocols supporting user privacy require to an amount of computational complexity in database or require to additional cryptographic operations in the tag.

So we think that it would be very impractical to support user privacy using the symmetric cryptographic algorithm in RFID system. Therefore, we have concentrated on designing a scheme which satisfies the rest security requirements except for user privacy in order to compatibility with today's RFID system.

V. Description of Self-Shrinking Generator (SSG)

Self-Shrinking Generator is a well-known keystream generator proposed by Meier, Staffelbach in EUROCRYPT'94. It is the modified version of shrinking generator [6]. SSG uses only one LFSR, which generate m-sequence. And, as shown in Fig. 3, it requires selection logic.

SSG runs as follow. Among the generating stream of LFSR in SSG, even bits are used for the choice bits and odd bits are used for output bits of SSG. For instance, Let A-LFSR be maximum length LFSRs generating the *m*-sequence $(a_i)_{i\geq 0}$. And Let $a_0, a_1, a_2, a_3, ..., a_{2i}, a_{2i+1}, ...$ be streams generated from A-LFSR. For any output pair (a_{2i}, a_{2i+1}) of A-LFSR, SSG outputs a_{2i+1} if $a_{2i} = 1$. Otherwise, it discards a_{2i+1}. And SSG has not revealed any weakness for recent attack. Thus, it has very small size and sufficient security. So, we think that SSG is the optimized stream cipher algorithm for hardware implementation in any devices with extremely limited resource.

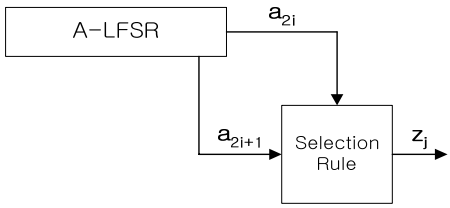

Fig. 3 Structure of Self Shrinking Generator

VI. Tag Authentication Scheme Using SSG

In this section, we describe our tag authentication scheme. In the proposed tag authentication scheme, any Tag_i should previously contain the values as follows.

- *meta-ID*(or *pseudonym*)
- *key_i* : internal secret value of LFSR in SSG.
- *counter_T* : index of output bits of SSG
- *ID* : Tag's unique identification information
- *h_ID* : hash value of *ID*

Initially reader only has access-authority information for specific tag. But this paper does not mention what the detail access-authority information for tag. The database should have the three values about each tag as follows. (i.e,. $Tag_i(meta\_ID_i, key\_i, count\_D_i)$)

- *meta_ID*
- *key_i*
- *counter_D*

Definitions of message and symbol using each flow of the proposed scheme are as follows.

- *counter_T* : This value indicate the index of output bit of SSG and monotonously increase. This is used to synchronize the secret value (*key_i*) shared between tag and database. Initially, value of *counter_T* is set to zero.
- *counter_D* : This value is contained at the database and do a same role as the *counter_T* which indicate the index of output bit of SSG. This is used to synchronize the secret value (*key_i*) shared between tag and database. Initially, value of *counter_D* is set to zero.
- *meta_ID* : This is the randomly chosen pseudonym for tag. Although an adversary known *meta_ID* value, he cannot directly know the tag information corresponding to *meta_ID*.
- *key_i* : This is the secret value shared between Tag_i and Database. It is loaded in LFSR of SSG both Database and Tag_i. After generating each output of SSG, then *key_i* value is updated. So, the updated *key_i* value will be different from the prior *key_i* value. Note that the *key_i* existed in Tag_i is always updated for every authentication session, without relation to the success of authentication process. But the the *key_i* existed in Database is only updated when succeed the authentication process. So it needs the synchronization process between Database and Tag_i.
- *KS* : This is the random output stream of SSG. This is used to encrypt the *ID* and *h_ID*.
- *rand* : This is a random value generated by reader, and it is used to prevent the reply attack. The bit length of *rand* is the same bit length of *ID*||*h_ID*
- *query_1* : This is an initial message from reader to tag. It notifies the starting of authentication process.
- + : This is addition modulo 2^n. Here, the *n* is a bit length of *ID*||*h_ID*.
- ^ : Bitwise XOR operation(⊕)

The overall protocol is shown in Fig. 4. As assumed in section 3, the dotted bracket parts indicate that communication channel between reader and database is secure. We describe the proposed protocol according to the sequence of step number and, in section 7; we discuss the security properties that are achieved during the execution of each step. The detailed procedures for each step are as follows.

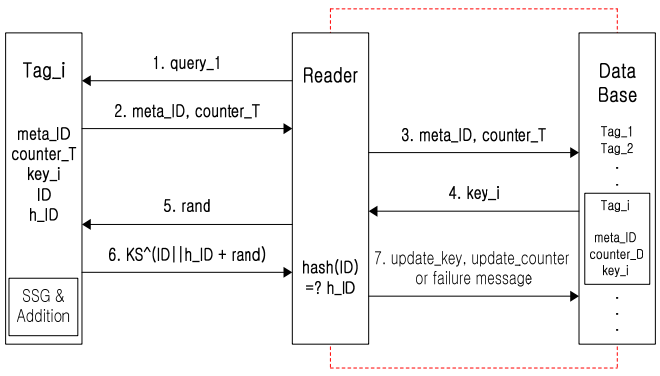

Fig. 4 Tag Authentication Scheme using SSG algorithm

1. Firstly, Reader sends *query_1* message to Tag_i

2. If Tag_i receives *query_1*, then Tag_i sends own's *meta_ID* and *counter_T* to Reader.

3. Reader transmits the received data from Tag_i to Database.

4. After Database receives *meta_ID* and *counter_T* from Reader, then Database starts to search the *counter_D* and *key_i* in internal storage with the received *meta_ID*. If Database will have found the *counter_D* and *key_i* of Tag_i corresponding to *meta_ID*, then Database compares the *counter_D* with the *counter_T*. If the *counter_D* is small than the *counter_T*, in order to synchronize the shared *key_i* with Tag_i, then Database runs SSG with the finded *key_i* until generate the output stream as same as $|counter\_T - counter\_D|$. When *counter_T* and *counter_D* are the same value, Database stops operation of SSG and sends the synchronized *key_i* in LFSR of SSG to Reader.

5. Reader receives the *key_i* from Database. Then, Reader generates (*rand*) random value through the implemented hash algorithm and sends *rand* to Tag_i.

6. Tag_i receives *rand* from Reader, then Tag_i concatanate *ID* and *h_ID* and calculate ($ID||h\_ID + rand$) to use the addition module. And Tag_i generate a *KS* of output stream of SSG with its *key_i* as same as bit length of ($ID||h\_ID + rand$). Then Tag_i calculates $KS \oplus (ID||h\_ID + rand)$ and sends this value to Reader.

7. Reader receives $KS \oplus (ID||h\_ID + rand)$ from Tag_i. And Reader generates a *KS′* of output stream of SSG with the received *key_i* from Database in step 4 as same as bit length of $KS \oplus (ID||h\_ID + rand)$. Reader calculates $KS \oplus KS \oplus (ID||h\_ID + rand)$ with the *KS′* to eliminate the *KS*, then obtains ($ID||h\_ID + rand$). Reader calculates $(ID||h\_ID + rand) - rand$, and obtains *ID*||*h_ID* for Tag authentication. Then, Reader calculates $h\_ID' = H(ID)$ to use hash algorithm with the extracted *ID* from *ID*||*h_ID*. Finally Reader verifies whether the extracted *ID* is valid or not by comparing the extracted *h_ID* with the calculated *h_ID′*. If $h\_ID = h\_ID'$, then Readers send *update_key* and *update_counter* message to Database. Otherwise, Reader sends *failure message* to Database.

8. Finally, if Database receives *update_key* and *update_counter* message, then updates the stored *key_i* and *counter_T* of Tag_i in internal storage of Database. Otherwise, Database does not update information of Tag_i.

VII. Analysis of Our Scheme

We evaluate our scheme in the viewpoint of the security requirement.

Security in our scheme is based on security of SSG algorithm. Namely, the point of view of SSG, an adversary cannot recover the secret value in LFSR of SSG only to use the gathered data in step 6. Also, after the SSG algorithm has been run, the secret value in LFSR of SSG is updated. So, the SSG algorithm always generate pseudo random bit stream at each time. Thus, the adversary cannot predict a posterior output stream of SSG. In other words, because of these properties of SSG in our scheme, although the adversary eavesdrops all communications prior between tag and reader, the adversary cannot derive $(KS \oplus (ID||h\_ID + rand))_{new}$ from $(KS \oplus (ID||h\_ID + rand))_{old}$. The advantages of our scheme are as follows.

Firstly, our scheme guarantees the secure communication between tag and reader. At every authentication session, we use the fresh random value *KS*. This *KS* is randomly generated during every authentication session, since the *key_i* used for generating *KS* is updated for every step 4 attempt. And tag *ID* and *h_ID* are encrypted by one-time pad form with random stream *KS*. Therefore, our scheme is secure against the passive eavesdropping attack.

Secondly, our scheme prevents the replay attack. In step 5, a legitimate Reader always generates unique random value ($rand_i$) only usable in *i*-th session and sends this ($rand_i$) to Tag. Also Tag calculates $(KS \oplus (ID||h\_ID + rand_i))_i$ with this $rand_i$ only usable in *i*-th session and sends it to Reader. Even though an adversary has eavesdropped the *meta_ID*, $counter\_T_i$ and $(KS \oplus (ID||h\_ID + rand_i))_i$ transmitted from tag in the *i*-th session and ,afterward, send these data to Reader at *j*-th session as if he is a legitimate tag, the adversary cannot succeed in cheating the reader. In the *j*-th session, the legitimate reader will generate the new $rand_j$ which differs from $rand_i$ and sends $rand_j$ to tag and will expect to receive the $KS \oplus (ID||h\_ID + rand_j)$ calculated with $rand_j$. If the adversary transmits the eavesdropped $KS \oplus (ID||h\_ID + rand_i)$ to the reader in *j*-th session, then the reader will extract *ID*||*h_ID* from $KS \oplus (ID||h\_ID + rand_j)_j$ with $rand_j$ and will verify the extracted *ID*||*h_ID*. On account of verification process for *ID*||*h_ID*, the reader will become aware that it is the wrong data. So the adversary will fail to cheat the reader such as itself a legitimate tag. Therefore, our scheme is secure against the replay attack.

Thirdly, our scheme guarantees the integrity of tag *ID*. Because a tag contains (*h_ID*) hash value of tag *ID* and transmits the encrypted *h_ID* together with the *ID*. Thus, our scheme can prevent the malicious message modifying attack through verification process for *ID*. And if a reader keeps the information of tags used at prior authentication process in internal storage, the reader will achieve the tag authentication to use the stored tag information, without interaction of the 3,4 step with database. This advantage reduces the computation complexity at the database. So, it implies that our scheme can prevent the DoS attack about the database.

The security of $KS \oplus (ID||h\_ID + rand)$ is as follows. Assume that an adversary can modify *rand* value to one bit at a time and transmit this modified rand (*rand'*) to a tag. And also, the tag calculates $KS \oplus (ID||h\_ID + rand')$ with *rand'* transmitted from the adversary, and replies $KS \oplus (ID||h\_ID + rand')$ to the adversary. Then, the adversary cannot obtain any partial information of $ID||h\_ID$ from $KS \oplus (ID||h\_ID + rand')$. Because, whenever the adversary sends *rand'* to the tag, the tag creates a randomly *KS* and calculates $KS \oplus (ID||h\_ID + rand')$ with the random *KS*.

The proof of this is as follows. We consider about the 0-th bit which don't need to calculate the addition of carry bit. Let r_0 denote the 0-th bit of *rand'*, where the LSB is r_0, and d_0, k_0, x_0 denote the 0-th bit of $ID||h\_ID$ and *KS* and $KS \oplus (ID||h\_ID + rand')$ respectively. If r_0 is 1 and $x_0 = k_0 \oplus (d_0 + r_0)$ is 1, then d_0 and k_0 can be 0 and 0 or 1 and 1, with $\frac{1}{2}$ probability respectively. If r_0 is 1 and x_0 is 0, then d_0 and k_0 can be 1 and 0 or 0 and 1, with $\frac{1}{2}$ probability respectively. In such a case that r_0 and x_0 become 0 and 0 or 1 and 1, above similar too. Furthermore, the tag will calculate x_0 with the randomly generated k_0 at every query time so that the adversary cannot eliminate k_0 from x_0 with the prior values. Thus, although the adversary transmits the modified r_0 differ from r_0 used in prior query to the tag, he cannot exactly determine what are the d_0 and k_0.

Contrary, suppose that a tag uses fixed *KS* to calculate $KS \oplus (ID||h\_ID + rand)$ at every query, then an adversary is feasible to easily obtain $ID||h\_ID$ information. Assume that *rand'* and *rand''* are only different the 0-th bit. Let y and z denote $KS \oplus (ID||h\_ID + rand')$ and $KS \oplus (ID||h\_ID + rand'')$, respectively. And let y_i and z_i denote the i-th bit of y and z, respectively. If (d_0) the 0-th bit of $ID||h\_ID$ is 0, then y_0 and z_0 become different value and y_1 and z_1 are same value. This is because the carry bit is not occurred from the $d_0 + k_0$. If d_0 is 1, then y_1 and z_1 become different value according to a carry bit. Consequently, d_0 is determined by state of y_1 and z_1. Therefore, if the adversary will have only to sequentially compare $KS \oplus (ID||h\_ID + rand')$ with $KS \oplus (ID||h\_ID + rand'')$ starting from LSB, then the adversary can easily obtain the entire $ID||h\_ID$ information.

The efficiency of the proposed scheme is as follows. When we suppose the 128-bit security, the hardware size needed to implement SSG using 128bit m-LFSR are roughly 1,057 logic gates and 32bit-Adder are 378 logic gates. So the total logic gates needed to implement our scheme in tag are 1435. This hardware size is very small than another authentication protocols using hash algorithm. Because, hardware size needed to implement the SHA1 of most popular hash algorithms are the minimum 10,000~20,000 logic gates [11].

In the case of performance, because of the characteristic of SSG generating 1 bit per 4 operation-clock, operation clock needed to generating 128 bit-KS output at SSG are 512(128*4), and operation clock required to achieve addition with 128 bit ($ID||h\_ID$) and 128bit *rand* are 4. Therefore, total operation clock needed to calculate $KS \oplus (ID||h\_ID + rand)$ are roughly 517 operation clocks. Also, based on 128 bit security, Tag is needed to the 32 byte read-only memory for 16 byte $ID||h\_ID$ and 16 byte *meta_ID*, and the 32 byte rewritable memory for 16 byte *counter_T* and 16 byte *key_i*. Thus, we will predict that a tag can create the 128 bit response in roughly 517 operation clocks using 1435 logic gates and 64 byte memory. Therefore, the proposed scheme very suits the resource-constrained RFID-tag environment.

VIII. Conclusion

In this paper, we have described the tag authentication scheme using SSG algorithm.

The proposed scheme requires the synchronization process of secret data shared between tag and database with counter and transmits the encrypted tag *ID* together with its authentication data. So, basically this scheme supports the authentication for tag *ID* and is secure against replay attack, eavesdropping attack, malicious message modifying attack, and additionally is secure against the denial of service attack.

Furthermore, SSG algorithm using in the proposed scheme is optimized to implementation of Hardware, and the proposed scheme only requires the 64byte memory for the *meta_ID*, *counter_T*, *key*, *ID* and *h_ID* so that our scheme is very feasible and practical for resource-constrained tag such as passive tag.

Finally, since the proposed scheme place great emphasis on the practicality and compatibility with today's existing RFID system, it seems to able to adapt to the applications of many RFID systems which do not need user privacy.

References

[1] Martin Feldhofer, Sandra Dominikus, Johannes Wolkerstorfer. "Strong Authention for RFID Systems Using the AES Algorithm", CHES2004, LNCS 3156, pp.357-370.

[2] Ari Juels. "Minimalist Cryptography for Low-Cost RFID Tags" 2003, In submission.

[3] A. Juels, R.L. Rivest, M. Szydlo. "The Block Tag : Selective blocking of RFID tags for consumer privacy" 8th ACM Conference of Computer and Communications Security, pp 103-111.

[4] K. Finkenzeller. "RFID-Handbook", Carl Hanser Verlag Munchen, 2nd edition 2003.

[5] S. Weis, S.Sarma, R.Rivest, and D.Engel, "Security and privacy aspects of low-cost radio frequency identification systems", In Proceedings of the 1th International Conference on Security in Pervasive Computing 2003.

[6] W. Meier, S. Staffelbach, "Self-Shrinking Generator" Advances in Cryptology EUROCRYPT'94, volume 950 of LNCS, pages 205-214.

[7] M. Ohkubo, K. Suzuki, and S. Kinoshita, "Cryptographic Approach to Privacy-Friendly Tags", RFID Privacy Workshop 2003, MIT, MA, USA, Nov. 2003.

[8] D. Henrici and P. Muller, "Hash-based Enhancement of Location Privacy for Radio-Frequency Identification Devices using Varying Identifiers", PerSec'04 at IEEE PerCom, pp.149-153, Mar.2004.

[9] D. Molnar, D. Wagner, "Security and privacy in library RFID : Issues, practices, and architectures", ACM CCS 2004.

[10] Stephen A. Weis, Sanjay E. Sarma, Ronald L.Rivest, and Daniel W. Engels, "Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems" In Security in Pervasive Computing, LNCS 2802, pp 201-212, 2004.

[11] Yu Ming-yan, Zhou Tong, Wang Jin-xiang, Ye Yi-zheng, "An Efficient ASIC Implementation Of SHA-1 Engine For TPM", IEEE Asia-Pacific Conference on Circuits and Systems, December 6-9, 2004 , pp 873-876.