Journal of Applied Mathematics and Physics, 2015, 3, 1-9
Published Online January 2015 in SciRes. http://www.scirp.org/journal/jamp
http://dx.doi.org/10.4236/jamp.2015.31001

RFID Anti-Counterfeiting for Retailing Systems

Duy-Thinh Tran, Sung Je Hong

Department of Computer Science and Engineering Pohang University of Science and Technology (POSTECH), Pohang, Republic of Korea
Email: tranthinh89@postech.ac.kr, sjhong@postech.ac.kr

Received December 2014

Abstract

Counterfeiting is one of the most serious problems in the consumer market. One promising approach for anti-counterfeiting is to attach a low-cost Radio-frequency Identification (RFID) tag to the product authentication. In this paper, we propose an RFID system for detecting counterfeiting products. This RFID system consists of the tag authentication protocol and the database correction protocol. We use the tag authentication protocol for authenticating tags without revealing their sensitive information. This protocol also allows the customer to freely inquire the tag. To prevent the widespread of the counterfeit products, we use the tag status information along with tag identity information. Meanwhile, the database correction protocol guarantees the correctness of the tag status. Our anti-counterfeiting system is the first work considering the seller who plays an important role in the consumer product supply chain. Finally, we show that anti-counterfeiting system is quite secure against counterfeiting and the tag authentication protocol is lightweight enough to be implemented in RFID-based applications.

Keywords

Secure RFID System, Authentication, Identification, Cryptographic One-Way Function, Counterfeiting, Anti-Counterfeiting, Lightweight

1. Introduction

Counterfeiting product is one of the serious problems that most product manufacturers and customers have to confront. The intent to make counterfeiting products is to take advantage of the original value of the genuine products. According to the report of International Chamber of Commerce (ICC), the global market loss due to counterfeiting products reaches $1.7 trillion by 2015 [1]. As a result, several countermeasure solutions have been proposed such as the barcodes, the hologram stickers, and the Radio-Frequency Identification (RFID) tags.

Among the aforementioned countermeasures, RFID technology is the most viable solution for product anti-counterfeiting due to the difficulty for the adversary to access to the sensitive information, e.g., the tag identity and the tag authenticity, of the RFID tag. Typically, each tag is equipped with an authentication protocol which provides more rigorous access control to the tag information. The verifier has to successfully pass the authentication

How to cite this paper: Tran, D.-T. and Hong, S.J. (2015) RFID Anti-Counterfeiting for Retailing Systems. *Journal of Applied Mathematics and Physics*, **2**, 1-9. http://dx.doi.org/10.4236/jamp.2015.31001

procedure to get this information. Thus, the RFID tag with an authentication protocol is a viable and proactive solution for anti-counterfeiting. Currently, the tag-side authentication protocol [2] [3] and the mutual authentication protocol [4]-[7] have been proposed for anti-counterfeiting.

Obviously, the latter protocol is more secure than the former protocol as it requires both sides of the protocol - the server and the tag-to get authenticated. However, we argue that the mutual authentication protocol is inappropriate for use in product retailing systems. First, since the mutual authentication protocol takes time to authenticate the server, to update and to synchronize the secret shared between the server and the tag [4]-[7], it increases the computation and the communication cost whereas most of the tags used in consumer products has weak computational resources. Second, the update and synchronization of the secret shared between the tag and the server can be suffered from the de-synchronization problem [4]. Finally, if the server authenticity must be taken into account, the tag might intentionally refute the authentication request from the verifier by abusing the server authenticity. Instead, customers should be able to verify any tag without hindrance from the tag. Thus, a secure tag-side authentication protocol where tags have to answers their proof of authenticity for any request is sufficient for use in the retailing system.

Additionally, the current proposed RFID authentication systems for anti-counterfeiting do not consider the role of the product seller. These systems consist of three parties: the RFID tag, the reader, and the server [8]-[10]. Meanwhile, the sellers/retailers always exists in the retailing system. Hence, in this paper, our anti-counterfeiting system considers the role of the seller. In the retailing system, a seller who distributes products can be either an authorized entity or an unauthorized one. If a seller is honest (authorized), he can support the product anti-counterfeiting systems. In contrast, if the seller is dishonest (unauthorized), he may conspire with the adversary in selling counterfeit products along with the genuine ones in his shop. Thus, to improve the anti-counterfeiting capability, we need a mechanism in which the seller has to support the anti-counterfeiting whilst refraining him from being dishonest. The contributions of our paper are summarized as follows:

- We propose an RFID-based anti-counterfeiting system consisting of two protocols: the tag authentication protocol and the database correction protocol. The tag authentication protocol increases the usability for the customers by allowing them to authenticate RFID tags without the need of authenticating the reader and the server. Meanwhile, the database correction protocol helps the seller and the server to update the tag status in the server database periodically.
- The proposed anti-counterfeiting system is secure against various attacks such as RFID tag counterfeit, seller impersonation, server impersonation and the database spoiling attack. Further, the proposed system not only allows the customer to detect the counterfeit tags, but also reduces the market loss due to counterfeit products.

The remainder of this paper is organized as follows. Section 2 discusses several related work. Section 3 describes the RFID-based anti-counterfeiting model and the both tag authentication protocol and database correction protocol in detail. Section 4 discusses the function F which involves in the tag authentication protocol. Section 5 analyzes the security of the anti-counterfeiting system. Section 6 analyzes the efficiency and the usability of our system. Finally, Section 7 concludes our paper.

2. Related Work

In recent years, researchers have been proposed numerous RFID-based systems for solving the counterfeiting problem [4] [5] [7] [11]. Their RFID systems consist of three parties: the server, the reader, and the RFID tag. They consider the reader and the server as an integrated entity. They assume that the communication channel between them are secure.

Although the mutual authentication protocols allow the two parties to authenticate each other, they also require the RFID tag to do more computational tasks such as random number generation and tag identity number update [6] [9] [12]. Paradoxically, most of the RFID tags belong to low-cost passive class, which means they have limited hardware resources and need the power supplied from the reader's radio energy.

Additionally, the mutual authentication protocols [9] [13]-[15] usually need to update and synchronize the secret information shared between the tag and the server database after each authentication session. The adversary can exploit this task to destroy the functionality of the tag by de-synchronizing the common information shared between the server database and the tag [4] [16]-[18]. As a result, we only need tag-side authentication RFID system which can prevent counterfeiting problem while allowing the customer (the reader) to freely inquire the tag

without the need to update the common information shared between the tag and the server database.

In terms of tag-side authentication, several schemes have been proposed [2] [3]. However, several limitations makes them inappropriate for use in the retailing system. Typically, the OSK protocol [2] suffers from the de-synchronization problem due to the update operation between the tag and the server database after each authentication session. Although this problem has been solved in the work of Godor *et al.* [6], their protocol is mutual authentication category, and hence it is not suitable for the context of retailing system. Our work can be considered as a variant of the Feldhofer *et al.* scheme [3] which is ISO/IEC 9798-2 unilateral authentication standard. However, this scheme uses AES encryption for providing the proof of identity. Note that AES encryption primitive is not compatible for lightweight application, especially for RFID tag [19]. Therefore, this scheme is not practical in the retailing system.

3. The Proposed Anti-Counterfeiting System

3.1. System Model

The RFID tag, the reader, the server, and the seller are the four parties of the RFID anti-counterfeiting system. Each RFID tag, attached on a product, stores the unique t_id and the corresponding secret$S$. The reader is a device used by the customer to verify if a product is genuine. A reader might be a smart phone with the authentication protocol downloaded from the product manufacturer. The product manufacturer (the tag issuer) maintains a database of the tags. The entities of the database are the tag identification number t_id, the secret S, the tag status t_status, and the seller name s_name. Each tag has unique t_id and the corresponding secret$S$.The tag status t_status is either $unsold$ or $sold$. When issuing a tag, the product manufacturer assigns t_status to $unsold$ in the server database. The server uses the database to authenticate and to maintain the status of tags. Whenever a product is shipped to a seller, the information of the tag is created to the database as shown in **Table 1**.

The tag authentication protocol and the database correction protocol are the two protocols in the RFID anti-counterfeiting system. Through the tag authentication protocol, the sever verifies if a specific product is genuine. Then the sever notifies the result (either $valid$ or $invalid$) to the reader. In this protocol, the server checks two things. One is to check whether the inquired product has been already sold out. If this is the case, the server sends the message $invalid$ to the reader. Another one is to check whether the tag stores the secret S as same as the one stored in the server database. If both secrets do not match, the server also sends the message $invalid$. The message $invalid$ implies that the product is a fake. The server sends the message $valid$ to the reader only when the product is unsold and the tag's secret is matched as well.

In this RFID anti-counterfeiting system, the server always updates the database by changing the tag status $sold$ for any inquired product. By doing this, the system prevents selling fake products with the same tag identification number more than once. However, a customer may visit a shop checking the genuineness of several products and leaves without buying anything. In this case, the database needs to be corrected by changing the tag status from $sold$ to $unsold$. This can be done by the database correction protocol. The server finds the seller of an inquired product from the database and sends him the tag identification number. If the product has been actually sold out, the seller does not need to do anything. Otherwise, the seller sends the message to the sever that the product is not sold out. If this is the case, the server updates the database by changing the tag status back to $unsold$. **Table 2** shows the notations used in the protocol.

3.2. The Tag Authentication Protocol

The players of the RFID tag authentication protocol are the server, the reader, and the tag. The purpose of this protocol is to verify if a product is genuine. The reader initiates the protocol by sending a query to the tag. The query consists of two numbers, a tag identification number t_id and a random number R_1. The t_id can be

Table 1. Server database.

t_id	*S*	*t_status*	*s_name*
0 × 000A	0xACB56	*sold*	Bob
0 × 000B	0xDF56E	*unsold*	Alice
…	…	…	…

Table 2. Notations

Notations	Interpretation
t_id	Tag identity number
s_name	Seller name
S	Secret shared by the tag and the server
t_status	$sold$ / $unsold$
Mu / Mr	Server's public key/private key
Su / Sr	Seller's public key/private key

found from the sticker on the product. Then the tag checks if the received t_id matches with its own tag identification number. If so, the tag computes the response $X = F(t\_id, R_1, S)$ and sends X to the reader. If not, the tag does not respond and the protocol terminates. Here the function F is a one-way function, which is described in Section 4. If the reader receives the response X from the tag, it generates another random number R_2. And then the reader sends $E_{Mu}(t\_id \| X \| R_1 \| R_2)$, which is the encryption of four numbers, t_id, X, R_1 and R_2 by the server's public key Mu. By this encryption, only the server knows the second random number R_2. After receiving the encrypted message $E_{Mu}(t\_id \| X \| R_1 \| R_2)$ from the reader, the server decrypts it using his private key Mr. The server checks if the database has the tag information corresponding to t_id. If not, the server does nothing and the protocol terminates. If so, then the server checks the tag status. If the tag status is $sold$, the server sends the message $(invalid, R_2)$. If the tag status is $unsold$, the server computes $Y = F(t\_id, R_1, S)$ and check if $X = Y$. If so, the server sends the message $(valid, R_2)$, and changes the tag status to $sold$. Otherwise, the server sends the message $(invalid, R_2)$.

Protocol 1. Tag authentication protocol

SUMMARY: Authenticate the tag

1. Setup

1.1 The tag and the server share the common secret S

1.2 The reader gets the server's public key Mu from the product manufacturer.

1.3 The reader gets the t_id from the tag sticker.

1.4 The function F is a one-way function with three inputs: t_id, random R_1, and S.

2. Protocol messages. The protocol involves four messages

The reader → the tag: t_id, R_1 (1)

The reader ← the tag: $X = F(t\_id, R_1, S)$ (2)

The reader → the server: $E_{Mu}(t\_id \| X \| R_1 \| R_2)$ (3)

The reader ← the server: $valid, R_2$ (4)

The reader → the server: $invalid, R_2$ (5)

3. Protocol actions. The tag is authenticated as follows

3.1 The reader chooses a random R_1, and sends (1) to the tag.

3.2 If t_id in (1) matches with the tag's own t_id, the tag computes $X = F(t\_id, R_1, S)$, and sends X to the reader. Otherwise, the tag terminates the protocol.

3.3 The reader generates a random R_2, encrypts t_id, X, R_1, R_2 with Mu, and sends (3) to the server.

3.4 The server decrypts (3) with its private key Mr, and finds the tuple $\{S, s\_id, t\_status\}$ corresponding to t_id. If $t\_status = sold$, the server sends (5) to the reader. Otherwise, the server computes $Y = F(t\_id, R_1, S)$, and checks $X = Y$. If so, the server sets $t\_status = sold$, and sends (4) to the reader. Otherwise, the server sends (5) to the reader. Finally, the server terminates the protocol.

3.3. The Database Correction Protocol

The seller and the server are the two players of the database correction protocol. The server uses this protocol to request the seller report the current status of the tag inquired by the reader. After the inquiry from the reader, the server changes the tag status t_status from $unsold$ to $sold$ in the database. If the customer buys the product with this tag, the tag status $t\_status = sold$ is correct, and the server does not need to update the database. However, if that product is not sold out, t_status should be changed back to $unsold$. To do this, the server has to correct $t\_status = unsold$ in its database before the next authentication session. For increasing the security of the protocol, the server and the seller use the public key infrastructure to exchange their messages. We assume that their public keys Mu and Su are distributed by a trusted certificate authority (CA).

For requesting the seller to report the current status of the inquired tag, the server sends the seller the message $E_{Su}(t\_id \| R_3)$,where t_id is the identity number of the inquired tag, and R_3 is a random number. After receiving $E_{Su}(t\_id \| R_3)$, the seller decrypts it using his private key Sr to get t_id and R_3. If the product with t_id is not sold, the seller responds the server the message $E_{Mu}(t\_id \| R_3 \| unsold)$. Otherwise, the seller responds with the message $E_{Mu}(t\_id \| R_3 \| sold)$.Once receiving $E_{Mu}(t\_id \| R_3 \| sold)$ or $E_{Mu}(t\_id \| R_3 \| unsold)$, the server decrypts it, and check received R_3 with the original version (the R_3 generated by the server). If they are matched, the server updates the database. Otherwise, the server terminates the protocol without updating the database. The use of R_3 guarantees that the message $E_{Mu}(t\_id \| R_3 \| sold)$ or $E_{Mu}(t\_id \| R_3 \| unsold)$, is sent from the legitimate seller. The database correction protocol is described follows.

Remark 1. By changing t_status to $sold$ for all of the inquired tags, the adversary-who can make a large number of counterfeit tags (products) with the same t_id -could only sell at most one counterfeit product before the server changes the tag status t_status to $sold$. Once a product has $sold$ status, this fact discourages the customer's willingness to buy this product. Thus, the customer might not buy it, or buy it with a price significantly cheaper than the $unsold$ product. Note that the adversary only gets benefit from counterfeit products when he sella large number of them. Hence, it is not worth for him to sell only one fake product. Therefore, our anti-counterfeiting system reduces the market loss significantly in the case when the secret S is disclosed to the adversary.

Protocol 2. Database correction protocol

SUMMARY: Update the tag status t_status in the server database after each tag authentication session

---

1. Setup

1.1 Both the server and the seller share their public keys Mu and Su, respectively, through a trusted CA.

1.2 This protocol occurs after an unsold genuine tag is authenticated. The server needs to know if the product attaching this tag is actually sold.

1.3 t_id is the identity number of the tag of which the server needs to update the tag status t_status.

2. Protocol messages. The protocol involves two messages

The server→ the seller: $E_{Su}(t\_id \| R_3)$ (1)

The server← the seller: $E_{Mu}(t\_id \| R_3 \| sold)$ (2)

The server← the seller: $E_{Mu}(t\_id \| R_3 \| unsold)$ (3)

3. Protocol actions. The server updates its database as follows

3.1 The server chooses a random R_3, encrypt the $t\_id \| R_3$ with the seller's public key Su, and sends (1) to the seller.

3.2 The seller decrypts (1) with his private key Sr. If the product having this t_id was sold, the seller sends (2) to the server. Otherwise, the seller sends (3) to the server.

3.3 If the server receives (2), he decrypts it with his private key Mr and checks R_3 value with the R_3 version (1). If they match, the server updates t_status corresponding to t_id.

Remark 2. The seller involves in the RFID system for updating the tag status t_status from $unsold$ to $sold$ after a tag is inquired by the customer. As we explain in *Remark* 1, the customer is unwilling to buy a product having $sold$ tag status even though it is a genuine one. However, the server always changes t_status to $sold$ every time a tag is inquired by a customer by using a reader. Hence, we need the seller to change t_status back to $unsold$ —via the database correction protocol (Protocol 2)—when the product with this tag is still in the shop

(Note that the reader can only check the product's status). As well, the sold fake product can be identified immediately when the server notifies the legitimate seller of product through the database correction protocol.

4. Function *F*

The successful probability of figuring out the inputs of F from the output must be negligible. This is the most important requirement for F. Specifically, the tag uses the output X of $F(t\_id, R_1, S)$ to prove his knowledge about the secret S in the tag authentication protocol. If the tag is legitimate, *i.e.*, it has the correct secret S corresponding to t_id, the tag can compute the $F(t\_id, R_1, S)$ accurately. However, because the adversary can eavesdrop X in Protocol 1 (step 3.2), it must be impossible for the adversary to figure out S from X. Further, for authenticating the tag, the server only needs to verify the correctness of X instead of getting back S from X. F must be lightweight enough for use in low-cost RFID tags as well. Specifically, the number of logic gates (GE) used for implementing F must be less than 2000 GEs, which is the hardware budget for the security function in the RFID tag [8] [20]. Therefore, F must be lightweight and secure one-way function. With this principal, we can choose an appropriate hash function having a decent collision and pre-image resistant level for our RFID system.

Recently, there are numerous lightweight hash function such as PHOTON [21], QUARK [22] and SPONGENT [23]. Among these lightweight hash functions, we choose SPONGENT-128 (128-bit output) as it requires the smallest number of GEs for implementation while providing decent collision and pre-image resistance level for RFID-based applications. Specifically, SPONGENT-128, PHOTON -128, and QUARK-128 require 1122, 1379, and 1060 GEs, respectively. Further, SPONGENT-128 provides 120-bit collision resistance and 64-bit pre-image security, which is strong enough for RFID-based applications.

To adapt the SPONGENT-128 hash function in our RFID system, t_id is assigned as the initial value of F while the random number R_1 is concatenated with the secret S before being processed by F. The bit-size of S should be smaller than the memory size of the tag, but must be long enough to prevent the brute-force search attack. As the memory of the popular low-cost tag is up to 512 bits [24] [25], we choose the size of S is at least 128 bits, which satisfies both the memory size and the security requirement.

5. Security Analysis

5.1. Adversarial Model

The adversary will exploit the weaknesses of the RFID system to achieve malicious goals. In [8], the authors classify adversaries based on their goals, level of interference, and available resources. In our model, we assume that there are two major goals of the potential adversary: 1) to counterfeit tags by stealing the secret information of the tags; 2) to corrupt the system functionality by attacking the server database. Additionally, the adversary can be a dishonest seller, who wants to sell counterfeit products along with the genuine ones. Depending on the specific goal of the adversary, the damage of the RFID system is different. If goal (1) is accomplished, the tags of the RFID system will be suffered from being counterfeited, thus, a large number of fake products will be produced. Meanwhile, if goal (2) is successful, the server functionality, and the tag status will be corrupted, and hence, the RFID system cannot provide its authentication service for the honest customers and the honest sellers.

5.2. RFID Tag Counterfeit

To counterfeit an RFID tag, the adversary must know the secret S corresponding to the tag number t_id. During authentication session, the reader and the tag transfer t_id, R_1 and X, the output of F. Therefore, the adversary can use brute-force search technique to figure out S from t_id, R_1 and X. Specifically, the adversary tries to search for the whole value space of S. Recall that S is at least 128-bit length (Section 4), thus this length satisfies the key-size requirement according to the report on key lengths from ECRYPT II [26] and from NIST [27] [28]. Therefore, it is impossible for the adversary to do brute-force key search to find out S.

Because F operates as a hash function, the adversary can get S by using the collision or the pre-image attacks, which are popular attacks on hash function [29]. Typically, in the collision attack, the adversary tries to find two distinct inputs S and S' such that $F(t\_id, R_1, S) = F(t\_id, R_1, S')$. In the pre-image attack, given X produced by F, the adversary tries to find any S such that $F(t\_id, R_1, S)$. However, F is a cryptographic one-way function which has decent collision resistance and pre-image security level, as shown in Section 4.Thus,

it is difficult for the adversary to find out an arbitrary S value different from the original S value inside the RFID tag, which can cause F to generate the same X value. In other words, the probability for the adversary to find out S from X, t_id, and R_1 is negligible.

5.3. Server Impersonation

To sell fake products, a dishonest seller must deceive a customer's reader through the tag authentication protocol. To do this, he needs to make a fake server which generates the valid message $(valid, R_2)$ for the reader's inquiry. Here, R_2 is the random number chosen by the reader. This R_2 is encrypted by the legitimate server's public key Mu, and then sent to the server. Hence, the seller cannot figure out R_2 because he does not know the server's private key Mr. This means that his fake server cannot generate the valid message $(valid, R_2)$.

5.4. Seller Impersonation

An adversary may impersonate the legitimate seller. His goal is to corrupt the server's database by keeping the tag status of the sold product as $unsold$. If this is the case, the impersonated seller (the adversary) can sell several counterfeit products with the same tag number t_id. The only possible way is to send the message $E_{Mu}(t\_id \| R_3 \| unsold)$ for the sold product with t_id through the database correction protocol. Here R_3 is the random number chosen by the server. This R_3 is encrypted by the legitimate seller's public key Su, and then sent to the seller. If the impersonated seller does not have the correct seller's private key Sr, he cannot figure out R_3. Therefore, the impersonated seller cannot generate the valid message $E_{Mu}(t\_id \| R_3 \| unsold)$. Additionally, whenever the server receives the message $E_{Mu}(t\_id \| R_3 \| unsold)$, the server can identify the seller based on the seller's name s_name in the database. If the seller is an illegitimate agency, the server just ignores this wrong message. Otherwise, if the seller is a legitimate agency, the manufacturer can accuse him for this database corruption attempt later. The seller may lose the dealership. In other words, a legitimate agency will not do this wrong doing.

5.5. Database Spoiling Attack

Since the server always assigns t_status to $sold$ after the tag authentication protocol, the adversary who impersonates as a customer can exploit this fact to spoil the server database by requesting the server to authenticate a large number of genuine and unsold tags. Therefore, the honest seller cannot sell the products attaching these tags anymore. However, the seller can continue to sell these unsold products (these inquired tags) by requesting the server to correct t_status in the database through the database correction protocol.

5.6. Denial of Service Attack

Because anyone can freely request the server to authenticate the tag, the adversary can exploit this characteristic to conduct the Denial-of-Service (DoS) attack. However, we can efficiently mitigate this problem by asking the reader to solve the CAPTCHA puzzle [30] for each time the reader inquires the server. Specifically, between step 3.3 and 3.4 in the tag authentication protocol (Protocol 1), the server asks the reader to solve a CAPTCHA puzzle. Unless the reader solve this puzzle correctly, the server dose not proceed step 3.4. This additional procedure prevents the reader-which is controlled by the adversary-from automatically and continuously inquiring the server.

6. Protocol Efficiency and Customer Usability Analysis

6.1. Protocol Efficiency Analysis

In the tag authentication protocol, function F is the main operation which the tag has to handle. Following the choice in Section 4, the F function requires 1060 GEs, which satisfies the hardware resource constraints for the low-cost RFID tag [8] [20]. In terms of number of operations, the tag has to handle one F operation; the reader has to handle one random number generation and one encryption operation; and the server has to handle one search operation, one F operation, and one decryption operation. Additionally, the database correction protocol only requires one encryption operation and one random number generation for the server and one decryption operation for the seller. As both the server and the seller have enough computational power to handle the pub-

lic-key encryption, the practicality of the system is guaranteed.

6.2. Customer Usability Analysis

Our proposed RFID system increases the usability for the customer as he can freely request the server to authenticate the tag without needing to identify himself to the server. The customer only needs to get the server's public key and send the tag identification number printed on the product for authentication. Further, the customer can use any device that can communicate with the tag and handle the public-key encryption scheme to communicate with the server as a reader.

7. Conclusion

We have proposed an RFID anti-counterfeiting system, which is secure against the RFID tag counterfeit, the server impersonation, the seller impersonation, and the database spoiling attack. Our system not only can detect the counterfeit tag, but also reduces the market loss due to the counterfeit problem. Next, we strengthen the anti-counterfeiting capability of our system by changing the database update permission to the seller who is identified and authenticated by the server, instead of the reader. Consequently, our system improves the usability for the customer by removing the reader-side (the server-side) authentication, thus the customer can freely inquire the tag and the server in the tag authentication protocol. Finally, our system is practical as the tag only has to handle the one-way function F, which is compatible with the low-cost RFID tag.

Acknowledgements

This work was supported by National Research Foundation of Korea under Grant NRF-2013R1A1A2016723.

References

[1] International Chamber of Commerce (ICC) (2011) Estimating the Global Economic and Social Impacts of Counterfeiting and Piracy. http://www.iccwbo.org/data/documents/bascap/global-impacts-study|full-report/

[2] Ohkubo, M., Suzuki, K. and Kinoshita, S. (2003) Cryptographic Approach to "Privacy-Friendly" Tags. *Proceeding of RFID Privacy Workshop*, 2003.

[3] Feldhofer, M., Dominikus, S. and Wolkerstorfer, J. (2004) Strong Authentication for RFID Systems Using the AES Algorithm. *Proceeding of Cryptographic Hardware and Embedded Systems—CHES* 2004, Vol. 3156, LNCS, 357-370.

[4] Kulseng, L., Yu, Z., Wei, Y. and Guan, Y. (2010) Lightweight Mutual Authentication and Ownership Transfer for RFID Systems. *Proceedings of the* 29*th INFOCOM*, NJ, USA, 251-255.

[5] Peris-Lopez, P., Hernandez-Castro, J.C., Tapiador, J.M.E. and Ribagorda, A. (2009) Advances in Ultralightweight Cryptography for Low-Cost RFID Tags: Gossamer Protocol. *Proceeding of Workshop of Information Security Applications*, LNCS, Vol. 5379, Springer, 56-68.

[6] Godor, G. and Imre, S. (2012) Hash-Based Mutual Authentication Protocol for Low-Cost RFID Systems. *Proceeding of Information and Communication Technologies*, LNCS, Vol. 7479, Springer, 76-87.

[7] Lee, Y.S., Kim, T.Y. and Lee, H.J. (2012) Mutual Authentication Protocol for Enhanced RFID Security and Anti-counterfeiting. *Proceeding of* 26*th Advanced Information Networking and Applications Workshops* (*WAINA*), 2012, 558-563.

[8] Cole, P.H. and Ranasinghe, D.C. (2008) Networked RFID Systems and Lightweight Cryptography: Raising Barriers to Product Counterfeiting. 1st Edition.

[9] Niu, B., Zhu, X. and Li, H. (2013) An Ultralightweight and Privacy-Preserving Authentication Protocol for Mobile RFID Systems. *Proceeding of IEEE Wireless Communications and Networking Conference* (*WCNC*), 2013.

[10] Chen, M., Chen, S. and Xiao, Q. (2014) Pandaka: A Lightweight Cipher for RFID Systems. *Proceeding of INFOCOM* 2014.

[11] Tian, Y., Chen, G. and Li, J. (2012) A New Ultralightweight RFID Authentication Protocol with Permutation. *IEEE Communications Letters*, **16**, 702-705. http://dx.doi.org/10.1109/LCOMM.2012.031212.120237

[12] Morshed, M.M., Atkins, A. and Yu, H. (2011) An Efficient and Secure Authentication Protocol for RFID Systems. *Proceeding of* 17*th International Conference on Automation and Computing* (*ICAC*), 2011, 51-56.

[13] Chang, Y.-F., Lin, S.-C. and Chang, P.-Y. (2011) A Location-Privacy-Protected RFID Authentication Scheme. *Proceeding of IEEE International Conference on Communications* (*ICC*), 2011, 1-4.

[14] Doss, R., Sundaresan, S. and Zhou, W. (2013) A Practical Quadratic Residues Based Scheme for Authentication and Privacy in Mobile RFID Systems. *AdHoc Networks*, **1**, 83-96.

[15] Yeh, T.-C., Wang, Y.-J., Kuo, T.-C. and Wang, S.-S. (2010) Securing RFID Systems Conforming To EPCClass 1 Generation 2 Standard. *Expert Systems with Applications*, **37**, 7678-7683. http://dx.doi.org/10.1016/j.eswa.2010.04.074

[16] Ahmadian, Z., Salmasizadeh, M. and Aref, M.R. (2013) Desynchronization Attack on RAPP Ultralightweight Authentication Protocol. *Information Processing Letters*, **113**, 205-209. http://dx.doi.org/10.1016/j.ipl.2013.01.003

[17] Avoine, G. and Carpent, X. (2013) Yet Another Ultralightweight Authentication Protocol That Is Broken. *Security and Privacy Issues*, Vol. 7739 of LNCS.

[18] Bagheri, N., Safkhani, M., Peris-Lopez, P. and Tapiador, J.E. (2014) Weaknesses in a New Ultralightweight RFID Authentication Protocol with Permutation RAPP. *Security and Communication Networks*, **7**, 945-949.

[19] Juels, A. and Weis, S.A. (2005) Authenticating Pervasive Devices with Human Protocols. *Proceedings of the* 25*th Annual International Conference on Advances in Cryptology*, *CRYPTO*, 293-308.

[20] Batina, L., Mentens, N., Sakiyama, K., Preneel, B. and Verbauwhede, I. (2006) Low-Cost Elliptic Curve Cryptography for Wireless Sensor Networks. *Security and Privacy in Ad-Hoc and Sensor Networks*, **4357**, 6-17. http://dx.doi.org/10.1007/11964254_3

[21] Guo, J., Peyrin, T. and Poschmann, A. (2011) The PHOTON Family of Lightweight Hash Functions. *Proceedings of the* 31*st Annual Conference on Advances in Cryptology*, *CRYPTO*, 222-239.

[22] Aumasson, J.-P., Henzen, L., Meier, W. and Naya-Plasencia, M. (2013) Quark: A Lightweight Hash. *Cryptology*, **26**, 313-339. http://dx.doi.org/10.1007/s00145-012-9125-6

[23] Bogdanov, A., Knezevic, M., Leander, G., Toz, D., Varici, K. and Verbauwhede, I. (2011) SPONGENT: A Lightweight Hash Function. *Proceedings of the* 13*th CHES*, 312-325.

[24] (2011) SRI512 RFID Tag Datasheet. http://www.st.com/st-web-ui/static/active/en/resource/technical/document/datasheet/CD00152477.pdf

[25] (2013) SL2S1412, SL2S1512, and SL2S1612 RFID Tag Datasheet. http://www.nxp.com/documents/data_sheet/SL2S1412_SL2S1512_SL2S1612.pdf

[26] (2012) ECRYPT II Yearly Report on Algorithms and Keysizes. http://www.ecrypt.eu.org/documents/D.SPA.20.pdf

[27] (2011) Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Lengths. http://dl.acm.org/citation.cfm?id=2206216

[28] (2012) Recommendation for Key Management Part 1: General (Rev. 3). http://dl.acm.org/citation.cfm?id=2206273

[29] Rogaway, P. and Shrimpton, T. (2004) Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. *Proceeding of Fast Software Encryption*, *FSE*, Vol. 3017 of LNCS, 371-388.

[30] Bursztein, E., Martin, M. and Mitchell, J. (2011) Text-Based Captcha Strengths and Weaknesses. *Proceedings of the* 18*th ACM Conference on Computer and Communications Security*, *CCS*, New York, 2011, 125-138.