Special English Edition of Journal of KIISC, Vol. 18 No.6 (B), December 2008

Improving Varying-Pseudonym-Based RFID Authentication Protocols to Resist Denial-of-Service Attacks*

(Invited paper)

Hung-Yu Chien, [1] *Tzong-Chen Wu* [2]
[1]National Chi Nan University, PuLi, Taiwan, ROC.
[2] National Taiwan University of Science and Technology, Taiwan, R.O.C.

Abstract

Applying Varying Pseudonym (VP) to design of Radio Frequency Identification (RFID) authentication protocol outperforms the other existing approaches in several respects. However, this approach is prone to the well-known denial-of-service (DOS) attack. In this paper, we examine the de-synchronization problems of VP-based RFID authentication protocols, and propose effective solutions to eliminate such weaknesses. We shall show that the proposed solutions indeed improve the security for these protocols, and moreover, these solutions require O(1) computational cost for identifying a tag and O(1) key space on the tag. These excellent performances make them very attractive to many RFID applications.

Key words : authentication, RFID, varying pseudonym, traceability, denial of service

I. Introduction

Due to the properties of low cost, reading without physical contact and writable storage, Radio Frequency Identification (RFID) tags, have many practical applications in our daily life, and further, several new potential applications are under development [27]. However, the potential wide deployment of RFIDs also incurs many security concerns, such as privacy, access control, identity protection, un-traceability, etc [3, 29]. It is interesting that anonymity and un-traceability are desirable or even necessary security requirements in many RFID applications, because identifying or tracing specific tags would reveal the identity, the location and the movement of the tagged object and even the person who carries the object.

From the security point of view, we do not discriminate the term *server* and the term *reader* when we describe the protocols in the following sections, since the server and the reader are usually resource-abundant devices on which conventional well-studied security protocols can be effectively implemented. Here, we only concentrate on those RFID authentication protocols that considered or claimed to achieve the requirements of anonymity and un-traceability. Those protocols like [4, 10, 12, 20, 26, 29, 30] that did not consider or did not well protect the properties of anonymity and un-traceability are excluded from the following discussion.

In this paper, we first classify the previous RFID authentication protocols based on their rationales, and then discuss their performance properties. After that, we examine the security weaknesses inherent in some previous protocols based on varying-pseudonym (VP), and finally propose our improvements. The proposed improvements not only enforce the security weakness but also have excellent performance.

The rest of this paper is organized as follows. In Section 2, we classify the previous RFID authentication schemes according to their rationales. In Section 3, we give brief review of some VP-based RFID authentication protocols, and point out their vulnerability against the Denial-of-Service (DOS) attacks. Our countermeasures to the DOS attacks are given in Section 4. In Section 5, we analyze the security and evaluate the performance of the proposed solutions. Finally, conclusion is given in Section 6.

II. Classification of RFID authentication protocols Tables

Based on the rationale used by an RFID

Manuscript received September 30, 2008.
Manuscript revised November 5, 2008.
1) Hung-Yu Chien, National Chi Nan University, redfish6@ms45.hinet.net
2) Tzong-Chen Wu, Taiwan University of Science and Technology, tcwu@cs.ntust.edu.tw
*This paper is the extended version of our previous work presented in [6].

authentication protocol to identify a tag while protecting the anonymity, we may classify such protocols into the following categories, as depicted in Fig. 1. Note that we only focus on the techniques to identify tags while preserving the anonymity, without covering the details of the protocols.

Simple challenge-response approach. In this approach, each tag T_i shares a distinct key k_i with the server or the reader. When a reader R probes a tag T_i by sending a random value N_R as a challenge, the tag T_i responds with $h(k_i, N_R)$, where $h()$ denotes a secure one-way function that can output commitment on its inputs while protecting the undisclosed input k_i. Upon receiving the response $h(k_i, N_R)$, the server verifies $h(k_j, N_R)$ for each potential tag T_j in its database to check whether there is a matched one. This approach allows the server to identify a tag without disclosing the identity to eavesdroppers. Each tag just keeps one secret key, but the server needs to perform the computation for each potential tag to identify the tag. So, the tag's storage space is O(1) but the computational cost for identifying a tag is O(n), where n the number of possible tags. The previous protocols proposed in [7, 9, 13, 14, 19, 22, 28, 31, 32] adopt this approach.

Tree-walk approach. In this approach, the tags are organized as a tree, where each leaf node in the tree denotes one tag and each edge in the tree is associated with a key. Fig. 1(b) shows a simple example. In the example, the tag T_1 holds the key $K1$ and $K3$, and the tag T_2 holds the keys $K1$ and $K4$. When a reader probes the tag T_2 by sending a challenge N_R, the tag T_2 responds with $\{h(K_1, N_R), h(K_4, N_R)\}$ on which the server can perform the so-called depth-first-search to identify the tag. This approach requires O($\log n$) key space on each tag and O($\log n$) computational cost for identifying a tag. However, the required key space is a serious burden on low-cost tags. One more serious problem of this approach is that once a tag is compromised, other tags that share the same keys on the same key path could be partially traced. The more the number of keys one tag T_i shares with the compromised tag T_j, the more probability the tag T_i could be identified and traced. The protocols proposed in [1, 19] adopt this approach.

Hash chains approach. One well-known work of this approach is Ohkubo *et al.*'s protocol [21]. In this approach, the server and each tag T_x share a distinct hash seed s_{1_x} initially. For each query request, the tag T_x updates $s_{i+1\_x} = h(s_{i\_x})$ for $i \geq 1$ and responds with $a_{i\_x} = g(s_{i\_x})$, where $h()$ and $g()$ are two different hash functions. This approach achieves the forward secrecy property. That is, if a tag is compromised some day in the future, then the past communications from the same tag can not be traced. However, Ohkubo et al.'s original version cannot resist the replay attack [1] and has the problem of poor scalability [2, 3]. In Ohkubo et al.'s protocol, the computational cost for identifying a tag is O(nm), where n is the number of potential tags and m is the maximum length of the hash chain. Lately, Avoine et al. [1, 2] proposed an improvement to overcome the replay attack inherent in the Ohkubo et al.'s original version. However, their improvement reduces the time complexity at the cost of extra memory required.

Varying Pseudonym (VP) approach. In this approach, each tag synchronizes its varying identifier and its internal state with the server. The varying identifier is called pseudonym in [5, 8, 17, 18, 23, 24, 25] and is called metaID in [11, 15, 16]. Here, we all refer to them the pseudonyms. Upon receiving a challenge request, a tag responds with the current pseudonym and the commitment on the challenge and the secret internal state. The server verifies the tag based on the commitment. During the authentication stage, the tag and the server respectively update their pseudonyms and their internal state. In this approach, the pseudonym not only protects the anonymity of the tag but also facilitates the server to identify the tag in its database with O(1) computational complexity, because the server can directly use the pseudonym to locate the corresponding entry in its database and perform necessary computations for this matched entry only. Furthermore, each tag only needs constant quantity of internal values, i.e., O(1) key space. These excellent features make the VP-based approach more attractive than the other ones. However, due to the synchronization requirement, the VP-based protocols are prone to the de-synchronization attacks (or the Denial-of-Service (DOS) attacks) [7, 8, 17, 18], if an adversary can manipulate the communications to let the tag and the server be out of synchronization. The protocols proposed in [5, 8, 11, 15, 16, 17, 18, 23, 24, 25] adopt this approach. It should be noted that some challenge-response-based protocols like [2, 7, 9] also synchronize the state between the tags and the server. But these protocols do not send a varying pseudonym to facilitate the server to perform fast identification, and therefore, we do not count them in the VP approach.

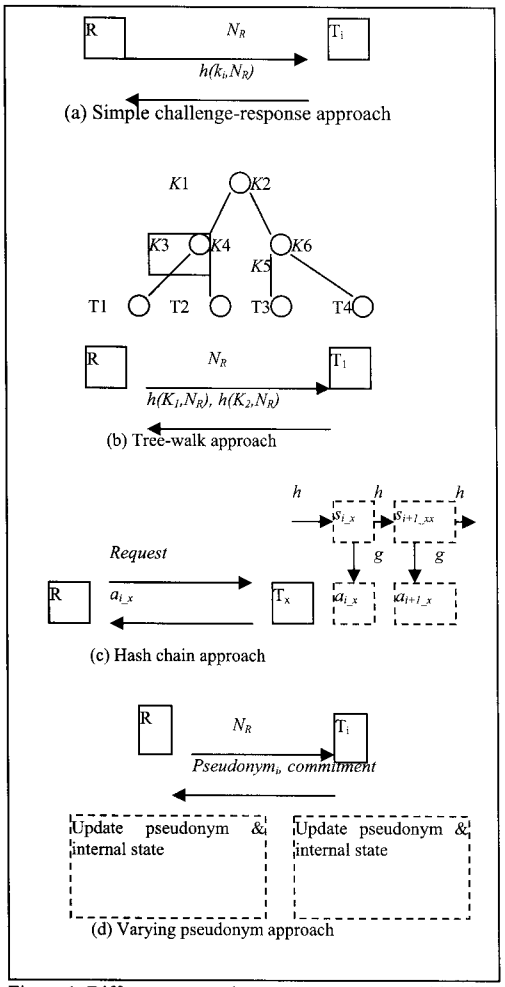

Figure 1. Different approaches to protect RFID tag identity

III. Discussions of previous VP-based protocols

In this section, we take several previous VP-based protocols as examples to examine their vulnerability to the DOS attacks.

A. SRAC Protocol

Lee and Verbauwhede [15] proposed a VP-based RFID authentication protocol called the SRAC protocol. The SRAC protocol keeps two sets of records for each tag to resist the DOS attack. However, this strategy does not work.

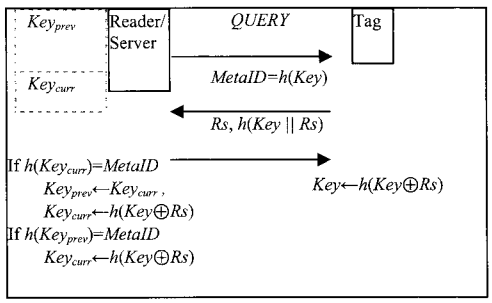

Figure 2. SRAC protocol

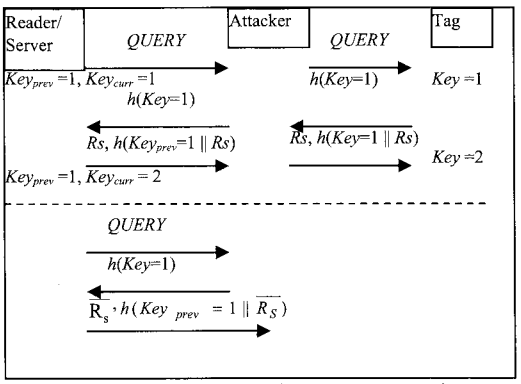

Figure 3. DOS attack on SRAC protocol

The SRAC protocol is depicted in Fig. 2. Initially, each tag shares one secret key *Key* with the server, while the server keeps two sets of keys for each tag: Key_{prev} is for the previous one and Key_{curr} is for the potential next one. Upon receiving the query from the reader or the server, the tag computes *MetaID*=*h*(*Key*), which is regarded as pseudonym, and sends it to the server. The server uses the *MetaID* to search its database. If the *MataID* matches one record in its database, then the server generates a random number *Rs* and checks whether *h*(*Key*⊕*Rs*) collides with any existing one. If so, the server re-generates a new *Rs* until *h*(*Key*⊕*Rs*) is unique. Note that *h*(*Key*⊕*Rs*) would be used as the next potential key. After that, the server updates the local state as specified in Fig. 2, and sends {*Rs*, *h*(*Key* || *Rs*)} to the tag. Finally, the tag uses {*Rs*, *h*(*Key* || *Rs*)} to verify the reader and then updates its local state as in Fig. 2 if the verification process succeeds.

Ideally, if an adversary can jam the communications all the time, then the service is, of course, out of function. However, such kind of attack can be easily detected and fixed in practice. We,

therefore, exclude such kind of impractical attacks from our discussions. Lee and Verbauwhede [15] had noticed a practical attack: An attacker might intercept the 3rd message in Fig. 2 such that the tag and the server would be out of synchronization if the server only keeps one key for each tag. This practical attack would let a legitimate tag no longer be accessed. Therefore, the SRAC protocol arranges the server keeps two sets of keys for each tag. For this reason, such attack does not de-synchronize the state, because the server still can use the previous key Key_{prev} to authenticate the tag. However, we find out that the SRAC protocol still cannot resist a sophisticated attack as shown in Fig. 3. .

DOS attack on SRAC protocol : A potential DOS attack on the SRAC protocol is plotted as follows. Initially, we assume that the tag has Key=1 and the server keeps two sets of keys, Key_{prev} =1 and Key_{curr} =1, for the tag. The attack consists of two stages. In the first stage, the attacker eavesdrops the communications between the tag and the server. In this stage, the server generates the random number Rs, and we assume that the value of $h((Key=1)\oplus Rs)$ to be 2. So, after the first stage, the tag would update Key=2, and the server has Key_{prev} =1 and Key_{curr} = 2. In the second stage, the attacker pretends to be the tag and replays the $MetaID=h(Key=1)$, where the server will find a matched record with $h(Key_{prev}=1)$ in its database and generates a new random number $\overline{R_S}$. Now, the server would have Key_{prev} =1 and Key_{curr} = $h((Key=1)\oplus \overline{R_S})$. At this time, Key_{curr} is supposed to be updated as 3 in our example of Fig. 3, but the tag still holds the old key Key=2. So, the DOS attack succeeds and the tag can on longer be accessed. This attack exploits the weakness of lacking of *message freshness* checking in the SRAC protocol. Such attack can also apply to the RFID ownership transfer protocol [22], due to the same reason.

B. LCAP Protocol

Lee *et al.* [16] proposed another VP-based RFID authentication protocol called the LCAP protocol. Like the SRAC protocol, the LCAP protocol keeps two sets of records for each tag. Fortunately, the LCAP protocol further generates a new random number in every authentication request to deter the replay attack stated above. However, a more sophisticated DOS attack still can de-synchronize the tag and the server.

The LCAP protocol is depicted in Fig. 4. The server maintains two sets of records with respect to each tag: one set is the last matched record and the other is the potential next one. Each set contains ID, $h(ID)$, TD and $DATA$, where ID denotes tag's varying identity, $h(ID)$ takes as tag's *metaID*, TD is a pointer to the other record of the same tag (i.e., TD_{prev} points to TD_{curr}, and TD_{curr} points to TD_{prev}), and $DATA$ denotes the other information of the tag. Upon receiving a request challenge r, the tag responds with $HaID=h(ID)$ and $h_L(ID||\ r)$, where $h_L(ID||\ r)$ denotes the left half substring of $h(ID||\ r)$. Based on $HaID$, the server first searches its *current* database to see whether a matched record exists; otherwise, it searches the *previous* database. If a record is found, the server uses the matched record to check whether $h_L(ID||\ r)$ equals the received one. If the verification succeeds and the matched one is in the *current* database, the server copies the *current* record to the

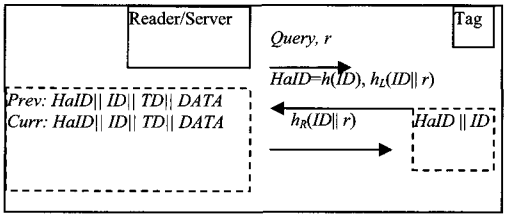

Figure 4. LCAP protocol

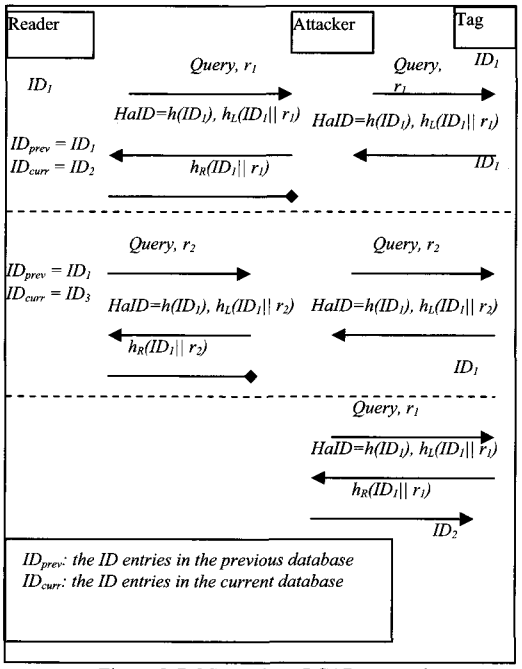

Figure 5. DOS attack on LCAP protocol

previous one, and updates $HaID=h(ID\oplus r)$ and $ID=ID\oplus r$ in the current record. Otherwise, if the matched one is in the *previous* database, the server directly updates the current ones as $HaID=h(ID\oplus r)$ and $ID=ID\oplus r$. After that, the server returns $h_R(ID||\ r)$, on which the tag can verify the reader/the server and decides whether or not to update its local values as $HaID=h(ID\oplus r)$ and $ID=ID\oplus r$.

Obviously, the replay attack plotted in Figure 3 does not work on the LCAP protocol, because each response in Step 2 should depend on the new challenge r. However, a deliberated attack like the one plotted in Fig. 5 can still de-synchronize the server and the tag.

DOS attack on LCAP protocol : The attack consists of three stages. In the first stage, the attacker just intercepts the message in Step 3 to let the server updates its records as $ID_{prev} = ID_1$ and $ID_{curr} = ID_1\oplus r_1$ (we denotes it as ID_2) but the tag still has ID_1. In the second stage, the reader sends a new challenge r_2 and the tags responds with $HaID=h(ID_1)$ and $h_L(ID_1||\ r_2)$. In this stage, the attacker also intercepts the message in Step 3 to let the server updates records as $ID_{prev} = ID_1$ and $ID_{curr} = ID_1\oplus r_2$ (we denotes it as ID_3) but the tag still has ID_1. In the third stage, the attacker impersonates the reader and replays the request eavesdropped in the first stage to let the tag locally updates its ID as $ID = ID_1\oplus r_1$ which should be ID_2. Now the tag and the server are out of synchronization and the DOS attack succeeds. Again, this attack exploits the weakness of lacking of *message freshness* checking in Step 3 in the LCAP protocol.

C. Henrici-MÄuller's protocol

In order to defeat the DOS attacks plotted in Fig. 3 and Fig. 5, the protocol proposed by Henrici and MÄuller [11] utilizes a more deliberated technique of using a blinded counter to defeat such attacks.

The Henrici-MÄuller's protocol is depicted in Fig. 6. Initially, the tag stores *ID*, *DB-ID*, *TID*, and *LST* in its memory, where *ID* denotes the varying pseudonym (VP), *DB-ID* denotes the corresponding database for this tag, *TID* denotes the tag's current counter which will be incremented after each query, *LST* denotes the counter value of the last successful authentication. Meanwhile, the database keeps *HID, ID, TID, LST, AE,* and *DATA* for each tag, where *ID* denotes the VP, *HID* equals $h(ID)$, *TID* denotes the last received counter, *AE* points to the other record entry of the same tag if such an record exists, and *DATA* points to the tag-related information.

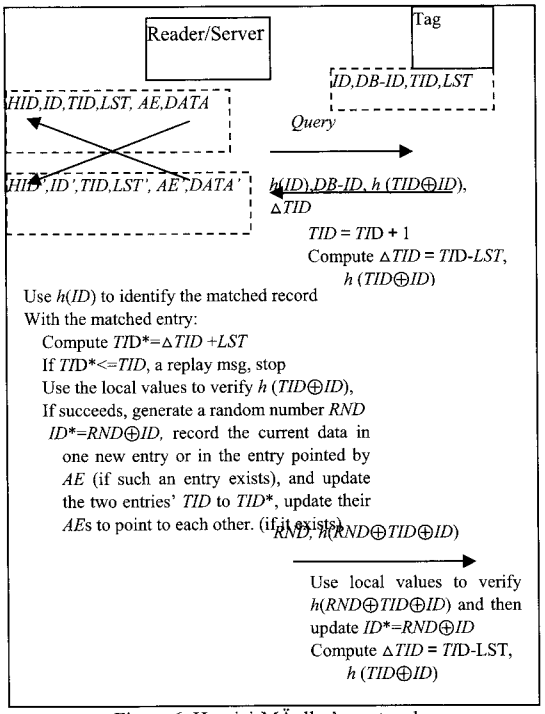

Figure 6. Henrici-MÄuller's protocol

For initializing each tag, *ID* is selected as a random number, *TID* and *LST* are another random number with the same value, *AE* points to null. Upon receiving a query, the tag increments its *TID*, computes $\Delta TID = TID\text{-}LST$, and sends out the data in Step 2. The server then uses $h(ID)$ to identify the matched record, and uses ΔTID and its local *LST* to compute $TID^*=\Delta TI + LST$. If $TID^*\leq TID$, then this could be a replay message and the server stops the protocol. Otherwise, the server verifies $h\ (TID\oplus ID)$. After the success of verification, the server generates a random number *RND* and computes $ID^*=RND\oplus ID$. Finally, the server updates the two entries such that one entry records the new data and the other entry records the current matched data. Meanwhile, both entries' *TID*s are updated to *TID**, and the two *AE*s are pointed to each other.

Yang *et al.* [31] had pointed out that, in Henrici-MÄuller's protocol, a tag is traceable between two consecutive successful authentications. Avonine and Oechslin [3] also had shown another tracing problem which is not limited to the time period between two successful authentications. Note that Henrici-MÄuller's protocol can effectively withstand our DOS attacks plotted in previous section. However, a

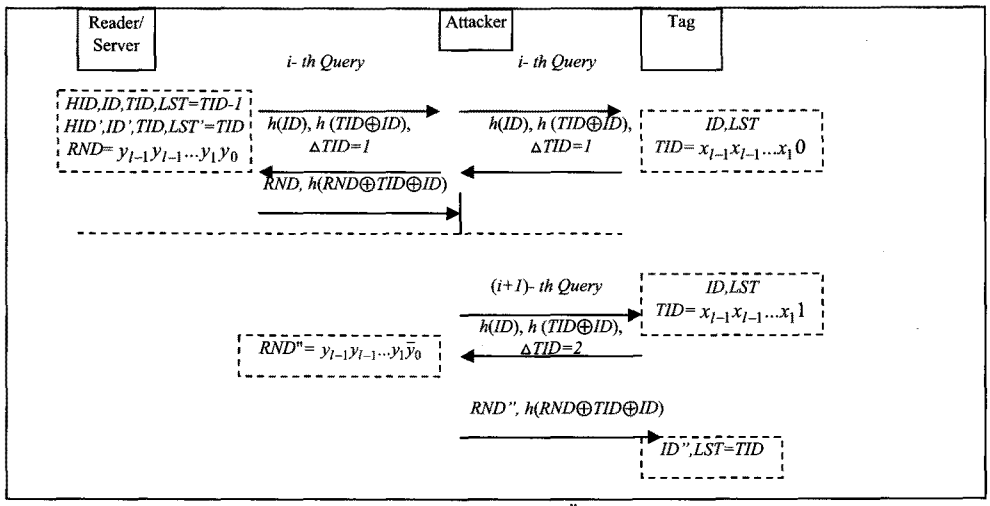

Figure 7. DOS attack on Henrici-MÄuller's protocol

more sophisticated attack can still de-synchronize the state as depicted in Fig. 7.

DOS attack on the Henrici-MÄuller's protocol : A possible DOS attack on Henrici-MÄuller's protocol is launched as follows. Recall that the tag increments its local counter *TID* for each query. The last bit of *TID* is either 0 or 1 (each case is of probability 1/2). Assume that the *TID* of the *i*th query is of the form ' $x_{l-1}x_{l-1}\ldots x_1 0$ ', where *l* denotes the bit length of the counter and $x_{j,1\le j\le l-1}$ (either 1 or 0) denotes the *j*th bit. This assumption holds with a probability of 1/2. That is, the last bit of *TID* for the *i*th query is 0. So, the next counter *TID*+1 for the (*i*+1)-th query will be of the form ' $x_{l-1}x_{l-1}\ldots x_1 1$ '. Our attack is sated in the following and depicted as Figure 7.

In Fig. 7, we only describe the attack for the normal case (ΔTID=1). Note that our attack still works even if ΔTID>1. The attack consists of two stages. In the first stage, the attacker intercepts the 3rd message $\{RND= y_{l-1}y_{l-1}\cdots y_1y_0,\ h(RND\oplus TID\oplus ID)\}$ between the server and the tag, where $y_{j,0\le j\le l-1}$ denotes the *j*th bit of RND. In the *i*-th query, the attacker eavesdrops the communications and intercepts the 3rd message such that the server will successfully authenticate the tag and has two entries for the tag $\{(HID, ID),(HID', ID')\}$, where *ID* is the current matched pseudonym, and *ID*' is the potential next one. Note that the tag still holds the old one *ID*. In the second stage, the attacker pretends to be the reader and sends the (*i*+1)-th query. At this time, the tag increments the counter *TID*, i.e., *TID* will be changed from $x_{l-1}x_{l-1}\ldots x_1 0$ to $x_{l-1}x_{l-1}\ldots x_1 1$. The attacker intercepts the 2nd message from the tag, inverts the last bit of *RND*, denoted as $RND"= y_{l-1}y_{l-1}\cdots y_1\bar{y}_0$, and sends $\{RND", h(RND\oplus TID\oplus ID)\}$ as the server's response to the tag, where the $h(RND\oplus TID\oplus ID)$ is the eavesdropped in the first stage. It is easy to check that the tag will accept the fake message $\{RND", h(RND\oplus TID\oplus ID)\}$ and updates its pseudonym $ID"= RND"\oplus ID$ accordingly, because of $h(RND"\oplus(TID+1)\oplus ID) = h(\ y_{l-1}y_{l-1}\cdots y_1\bar{y}_0 \oplus x_{l-1}x_{l-1}\ldots x_1 1 \oplus ID) = h(\ y_{l-1}y_{l-1}\cdots y_1y_0 \oplus x_{l-1}x_{l-1}\ldots x_1 0 \oplus ID\ ) = h(RND\oplus TID\oplus ID)$. Now the tag's pseudonym *ID"* is in-consistent with the two entries (*ID*, *ID*') in the database. The DOS attack succeeds, if the last bit of *TID* in the first stage is 0. The successful probability for such attack is 1/2 per each attack launched.

The attack exploits the weakness of no strong integrity protection provided in Henrici-MÄuller's protocol. The same weakness can also be found in Kim *et al.*'s protocol [14]. Recently, Peris-Lopez *et al.* [23, 24, 25] proposed a series of ultra-lightweight RFID authentication protocols for very low-cost tags, where only simple bit-wise operations like AND, OR, XOR are supported. However, due to lacking of strong integrity protection provided, these protocols still face the de-synchronization attacks plotted by Li and Deng [17], Li and Wang [18], and Chien and Huang [8].

IV. VP-based RFID authentication protocols resistant to DOS attack

Based on the analysis in Section 3, we highlight that strong mechanisms for mutual authentication and integrity protection are necessary rationales for resisting various DOS attacks. In this section, we shall propose some practical solutions against the DOS attacks stated in the previous sections. Our solutions preserve the excellent merits of the VP approach, and all of them require O(1) computational complexity for identifying a tag and O(1) key space per tag. The countermeasures against possible DOS attacks are given in the following.

A. VP Protocol 1

This solution is modified from Ohkubo et al.'s hash chain-based scheme. This solution not only conquers the weaknesses of the replay attack and the tracing problem in the original scheme. Assume that two one-way hash functions, $h()$ and $g()$, and a random number generator are ported on the tags. Initially, each tag shares two random seeds namely as s_0 and $g_0 = g(s_0)$ with the server. Denote $s_i = h(s_{i-1})$, for $i \geq 1$, as the hashing chain that will be used as the varying shared key between the tag and the server. Denote $g_i = g(s_i)$, for $i \geq 0$, as the VPs. The server maintains two sets of (s_i, g_i) for each tag: one is the last successfully matched set and the other is the potential next one. Initially, the two sets all contain $(s_0, g_0 = g(s_0))$. In the following, R denotes the reader/the server, T denotes the tag, $R \rightarrow T$ denotes the message is from R to T, and N_R / N_T respectively denotes the random nonce from R and T. The following protocol describes the authentication process where the tag's current state is $(s_i, g_i = g(s_i))$ and the server's state for the tag is $\{(s_{i-1}, g_{i-1}), (s_i, g_i)\}$.

1. $R \rightarrow T$: *Query*, N_R
2. $T \rightarrow R$: $g_i, N_T, h(s_i \| N_R \| N_T)$

Upon receiving the query, the tag generates a new random number N_T, and sends $\{g_i, N_T, h(s_i \| N_R \| N_T)\}$ to the server.

3. $R \rightarrow T$: $g(s_i \| N_R \| N_T)$

Upon receiving the data in Step 2, the server first uses the pseudonym g_i to identify the tag and uses the matched record in its database to verify the received data, $h(s_i \| N_R \| N_T)$. If the verification succeeds, the server updates its database by keeping one set of (s_i, g_i) and the other is (s_{i+1}, g_{i+1}). After that, the server returns $g(s_i \| N_R \| N_T)$ to the tag who will use its local data to verify $g(s_i \| N_R \| N_T)$. Finally, the server updates its state to (s_{i+1}, g_{i+1}) if the verification succeeds.

From the protocol description, it is noticed that even an attacker could intercept the message in Step 3, the tag and the server can still authenticate each other using the set (s_i, g_i). Of course, an attacker can trace the tag between two successful authentications. However, such kind of tracing does not produce significant practical value.

B. VP Protocol 2

This solution assumes that only one hash function $h()$ and one random number generator are supported on the tag. Initially, each tag shares one pseudonym *IDS* and one secret key K with the server. For each tag, the server maintains two records: one is the last matched set and the other is the potential next one. Initially, the two sets all contain the same values. The protocol is described as follows.

1. $R \rightarrow T$: *Query*, N_R
2. $T \rightarrow R$: $IDS, h(K \| N_R) \oplus N_T, h(K \| N_R \| N_T)$

Upon receiving the query, the tag generates a new random number N_T, and sends $\{IDS, h(K \| N_R) \oplus N_T), h(K \| N_R \| N_T)\}$ to the server, where $h(K \| N_R \| N_T)$ is regarded as the message authentication code of (K, N_R, N_T).

3. $R \rightarrow T$: $h(IDS'' \| K'' \| K)$

After receiving the data in Step 2, the server first uses *IDS* for identifying the matched records in its database, uses the K in the matched record for retrieving N_T from $h(K \| N_R) \oplus N_T$, and finally uses the retrieved value N_T and the local K for verifying the message authentication code $h(K \| N_R \| N_T)$. If the verification succeeds, i.e., the tag is successfully authenticated, and the server computes a new $IDS'' = h(IDS \| N_T \| K)$ and a new $K'' = h(K \| IDS \| N_T)$, and updates the two records for this tag in its database such that one record keeps the potential next values (*IDS''*, *K''*) and the other keeps the current matched one (*IDS*, *K*). The server finally sends the tag the data $h(IDS'' \| K'' \| K)$ as a confirmation of the message in Step 2 and of the new data (*IDS''*, *K''*). Upon receiving the data in Step 3, the tag uses its local values (IDS, N_T, K) to compute

the new $IDS = h(IDS \| N_T \| K)$ and the new $K = h(K \| IDS \| N_T)$, uses the new values to verify the received $h(IDS'' \| K'' \| K)$. If the verification succeeds, it uses the new values to replace the old ones.

The above protocols assume that one-way hash function could be effectively implemented on the tags. However, some low-cost RFIDs (like EPC Gen2 tag) still cannot afford the porting of one-way hash functions, and some very low cost tags even cannot support random number generator. Notice that even if we assume two tags share the same pseudonym, the probability that the two tags also share the same key is negligible so that the protocol can differentiate and authenticate the two tags.

V. Security analysis and performance evaluation

In this section, we analyze the security of the proposed solutions, and evaluate their performance.

A. Security Analysis

Some possible but potential attacks and security requirements (anonymity, un-traceability, and forward secrecy) on our proposed solutions are discussed as follows:

Mutual authentication/resistance to replay attacks: In our solutions, both the tag and the server will generate new random numbers, and the designated receiver should generate its response based on the secret key and the new challenge. It is easy to check that the proposed solutions can achieve mutual authentication and resist replay attacks.

Anonymity/un-traceability: The VP protocol 1 is modified from Ohkubo *et al.*'s hash chain-based scheme. Upon a query, the tag will respond with a new pseudonym $g_i = g(s_i)$ and s_i is updated as $s_i = h(s_{i-1})$ such that attackers can not identify the tag and cannot trace the tag, since each pseudonym seems random to the attackers. In the VP protocol 2, the pseudonym is updated consequently followed by the secret key K and the blinded random number N_T. Therefore, an attacker cannot identify the tag and cannot trace the tag. Of course, an attacker can trace a tag between two successful authentications. However, such kind of tracing could have little practical value.

Forward secrecy: Forward secrecy requires that even if a tag is compromised some day later, the past communications from the same tag cannot be linked. Solution 1 inherits this property from the extra one-way hash $g_i = g(s_i)$ on the hash chain $s_i = h(s_{i-1})$. The inspiration of using two hash functions was first introduced by Ohkubo *et al.* [4]. The solution 2 also preserves forward secrecy. Even if a tag is compromised some day later, the attacker can get the values of the current (*IDS*, *K*). However, since the computation of the new *IDS* and the *K* are respectively defined as $IDS'' = h(IDS \| N_T \| K)$ and $K'' = h(K \| IDS \| N_T)$, the attacker cannot deduce the previous *IDS* and the previous *K* from the equations. Therefore, the attacker cannot trace the past communications, based on the current *IDS* and *K*. The forward secrecy is preserved.

Resistance to DOS attacks: Our proposed solutions provide strong authentication mechanisms between the server and the tag. Also, they provide strong integrity protection on the communication messages. Therefore, an attacker cannot de-synchronize the state by modifying the messages without being detected. Actually, an attacker can intercept the 3rd message in the two solutions to deter the tag from updating its state. However, since the server always keeps two sets of records for each tag: one is the potential next one and the other is the last matched one. So, the server can still authenticate the tag under such condition.

B. Performance Evaluation

In our proposed VP protocol 1, the tag is required to support two hash functions and one random number generator. In our proposed VP protocol 2, the tag only is required to support one hash function. To identify a tag, both protocols only need O(1) computation, because the server directly uses the pseudonym to identify the potential tag in its database and performs necessary matched entry only. Our solutions only require three interactions to achieve mutual authentication. The features of distinct approaches are summarized in Table 1. Because the difference of the numbers of steps among distinct approaches is not significant, and the number of steps highly depends on the design of the protocol but not the approach adopted, we, therefore, do not count the number of steps (the communication complexity) in Table 1. Table 2 summarizes the properties of some well-known VP-based protocols. From Table 2, one can see that our proposed protocols out-perform the other VP-based schemes in the security respect and preserve the low storage requirement and high computational performance of the VP approach.

Table 1. Summaries of features of distinct approaches

Distinct approaches	Computational cost for identifying a tag	Key storage on tag	Compromised one tag will disclose the privacy of other tags
VP-based approach	O(1)	O(1)	No
Simple challenge-response	O(n)	O(1)	No
Tree-walk approach	O($\log n$)	O($\log n$)	Yes
Hash-chain approach	O(nm)	O(1)	No

Note: n denotes the number of tags, m denotes the maximum length of the hash chain

Table 2. Comparisons among various VP-based protocols

Distinct schemes	Computational cost for identifying a tag	Key storage on tag	Number of steps	Resist DOS	Resist key disclosure	Resist tracing	Resist replay attack
Our schemes	O(1)	O(1)	3	Yes	Yes	Yes	Yes
SRAC [27]	O(1)	O(1)	3	No	Yes	Yes	Yes
LCAP [28]	O(1)	O(1)	3	No	Yes	Yes	Yes
Henrici-MÄuller [8]	O(1)	O(1)	3	No	Yes	No	Yes
Karthikeyan-Nesterenko [12]	O(1)	O(1)	3	No	Yes	No	No
Peris-Lopez et al. [20-22]	O(1)	O(1)	4	No	No	No	Yes

VI. Conclusion

In this paper, we have discussed the features of different approaches to anonymously authenticate RFIDs and have shown the possible and potential DOS attacks on some well-known previously proposed VP-based protocols. We also have proposed two VP-based protocols to enforce the security weaknesses inherent in the previous work. Our solutions require only O(1) computational cost to identify a tag, and need only O(1) key space on the tag. These excellent performances make them very attractive to many RFID applications.

References

[1] Avoine, G., Dysli, E., and Oechslin, P.: 'Reducing time complexity in RFID systems'. Proc. th 12th Annual Workshop on Selected Areas in Cryptography (SAC), LNCS 3897, Springer, 2006, pp. 291-306.

[2] Avoine,G., and Oechslin, P.: 'A scalable and provably secure hash-based RFID protocol'. Proc. IEEE PerCom, 2005, pp. 110-114.

[3] Avoine, G., and Oechslin, P.: 'RFID traceability: a multi-layer problem'. Proc. Financial Cryptography 2005, LNCS 3570, Springer, pp. 125-140.

[4] Bringer, J., Chabanne, H. and Dottax, E.: 'HB++: A Lightweight Authentication Protocol Secure against Some Attacks'. IEEE International Conference on Pervasive Service, Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing – SecPerU, 2006.

[5] Chien, H.-Y.: 'SASI: A New Ultra-Lightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity'. IEEE Transactions on Dependable and Secure Computing, 2007, 4, (4).

[6] Chien, H.-Y.: 'DOS attacks on Varying Pseudonyms-Based RFID Authentication Protocols'. Proc. of IEEE APSCC 2008, Yilan, Taiwan, Dec. 9-12.

[7] Chien, H.-Y., and Chen, C.-H.: 'Mutual Authentication Protocol for RFID Conforming to EPC Class 1 Generation 2 Standards'. Computers Standards & Interfaces, 2007, 29, (2), pp 254-259.

[8] Chien, H.-Y., Huang, C.-W.: 'Security of Ultra-Lightweight RFID Authentication Protocols and Its Improvements'. ACM Operating System Reviews, 2007, 41, (2), pp. 83-86.
[9] Duc, D. N., Park, J., Lee, H., and Kim, K.: 'Enhancing Security of EPCglobal Gen-2 RFID Tag against Traceability and Cloning'. The 2006 Symposium on Cryptography and Information Security, 2006.
[10] Gilbert, H., Robshaw, M., and Sibert, H.: 'An Active Attack against HB+-A Provably Secure Lightweight Authentication Protocol'. Cryptology ePrint Archive, Report 2005/237, 2005.
[11] Henrici, A. D., and MÄuller, P.: 'Hash-based Enhancement of Location Privacy for Radio-Frequency Identification Devices using Varying Identifiers'. Proc. of IEEE PerCom 2004, pp.149-153.
[12] Juels, A., and Weis, S. A.: 'Authenticating pervasive devices with human protocols'. Crypto'05, LNCS 3126, Springer, pp.293-308.
[13] Karthikeyan, S., and Nesterenko, M.: 'RFID security without extensive cryptography'. Proc. of the 3rd ACM workshop on Security of ad hoc and sensor networks, 2005, pp. 63-67.
[14] Kim, J., Choi, D., Kim, I., and Kim, H.: 'Product authentication service of consumer's mobile RFID device'. ISEC'06, 2006, pp. 1-6.
[15] Lee, Y. K., and Verbauwhede, I.: 'Secure and Low-cost RFID Authentication Protocols'. Adaptive Wireless Networks－AWiN, November 2005.
[16] Lee, S. M., Hwang, Y. J., Lee, D. H., and Lim, J. I.: 'Efficient Authentication for Low-Cost RFID Systems'. International Conference on Computational Science and its Applications - ICCSA 2005, May 2005.
[17] Li, T., and Deng, R. H.: 'Vulnerability Analysis of EMAP-An Efficient RFID Mutual Authentication Protocol'. The Second International Conference on Availability, Reliability and Security (AReS 2007), Vienna, 2007.
[18] Li, T., and Wang, G.: 'Security Analysis of Two Ultra-Lightweight RFID Authentication Protocols'. IFIP SEC 2007, May 2007.
[19] Molnar, D., and Wagner, D. 'Privacy and security in library RFID: Issues, practices, and architectures'. Proc. Conference on Computer and Communications Security – CCS'04, 2004, pp. 210–219.
[20] Munilla, J., and Peinado, A.: 'HB-MP: a further step in the HB-family of lightweight authentication protocols'. Computer Networks, 2007, doi:10.1016/ j.comnet.2007.01.011.
[21] Ohkubo, M., Suzuki, K., and Kinoshita, S.: Cryptographic approach to 'Privacy-friendly' tag'. RFID Privacy workshop, MIT, USA, 2003.
[22] Osaka, K., Takagi, T., Yamazaki, K., and Takahashi, O.: An efficient and secure RFID security method with ownership transfer'. Proc. of International Conference on Computational Intelligence and Security 2006, LNCS 9743, Springer, pp. 1090-1095.
[23] Peris-Lopez, P., Hernandez-Castro, J. C., Estevez-Tapiador, J. M., and Ribagorda, A.: 'EMAP: An Efficient Mutual Authentication Protocol for Low-cost RFID Tags'. OTM Federated Conferences and Workshop: IS Workshop, November 2006.
[24] Peris-Lopez, P., Hernandez-Castro, J. C., Estevez-Tapiador, J. M., and Ribagorda, A.: 'M2AP: A Minimalist Mutual-Authentication Protocol for Low-cost RFID Tags'. Proc. of International Conference on Ubiquitous Intelligence and Computing UIC'06, LNCS 4159, Springer, pp. 912-923.
[25] Peris-Lopez, P., Hernandez-Castro, J. C., Estevez-Tapiador, J. M., and Ribagorda, A.: 'LMAP: A Real Lightweight Mutual Authentication Protocol for Low-cost RFID tags'. Proc. of 2nd Workshop on RFID Security, July 2006.
[26] Piramuthu, S.: 'HB and Related Lightweight Authentication Protocols for Secure RFID Tag/Reader Authentication'. CollECTeR Europe Conference, June 2006.
[27] RFID Journal, http://www.rfidjournal.com/.
[28] Rhee, K., Kwak, J., Kim, S., and Won, D.: 'Challenge-response based RFID authentication protocol for distributed database environment'. International Conference on Security in Pervasive Computing – SPC 2005, pp. 70–84.
[29] Weis, S. A.: 'Security and Privacy in Radio-Frequency Identification Devices'. Masters Thesis MIT, 2003.
[30] Weis, S. A., Sarma, S. E., Rivest, R. L., and Engels, D. W.: 'Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems'. Proc. of the First Security in Pervasive Computing, 2003, LNCS2802, Springer, pp.201-212.
[31] Yang, J., Park, Lee, J., Ren, H., K., and Kim, K.: 'Mutual authentication protocol for low-cost RFID'. Handout of the Ecrypt Workshop on RFID and Lightweight Crypto, 2005.
[32] Yang, J., Ren, K. and Kim, K.: 'Security and privacy on authentication protocol for low-cost radio'. The 2005 Symposium on Cryptography and Information Security, 2005.

Authors

Hung-Yu Chien received the B.S. degree in Computer Science from NCTU, Taiwan, 1988, the M.S. degree in Computer and Information Engineering from NTU, Taiwan, 1990, and the doctoral degree in applied mathematics at NCHU 2002. He was an assistant researcher at TL, MOTC, Taiwan, during 1992-1995. He was an associate professor of ChaoYang University of Technology during 2003~2006/08. Now he is a professor of National Chi Nan University, a member of the Chinese Association for Information Security, an IEEE member, and an ACM member. His research interests include cryptography, networking and network security.

Tzong-Chen Wu received the B.S. degree in the Department of Information Engineering from National Taiwan University (Taiwan) in 1983, the M.S. degree in the Department of Applied Mathematics from National Chung Hsing University (Taiwan) in 1989, and the Ph.D. in the Department of Computer Science and Information Engineering from National Chiao Tung University (Taiwan) in 1992. From August 1992 to January 1997, he has been the associate professor at the Department of Information Management, National Taiwan University of Science and Technology (NTUST, Taiwan Tech.). Since February 1997, he has been the professor at the Department of Information Management, NTUST, and chaired the Department of Information Management from 1999 to 2003. He is the members of IEEE, ACM, and the Chinese Cryptology and Information Security Association (CCISA). Now, he serves as the Dean of School of Management (NTUST), President of CCISA, one of the Steering Committees of Asiacrypt from Taiwanese side, and the Director of TWISC@NTUST. His research interests include cryptography, data security, network security, and data engineering.