An Enhanced Synchronization Approach for RFID Private Authentication

Qingsong Yao, Yong Qi, Jizhong Zhao, Jinsong Han
School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, China
qsyao@yahoo.cn, {qiy, zjz, hjs.jason}@mail.xjtu.edu.cn

Abstract

Radio Frequency Identification (RFID) technologies are on their highway to pervasive usage. However privacy protection is still an important problem since RFID tags attached to items are so cost constrained. Privacy preserving authentication approaches are proposed to authenticate tags without private information leaking. Previously designed approaches based on synchronization seeks O(1) complexity. While these synchronization based methods are efficient in normal case, they have weak points when desynchronized. When maliciously scanned, information stored in tag and reader goes farther and farther away from each other. An adversary can utilize this point to track a tag. We propose an Enhanced Synchronization aPproach for RFID private authentication, ESP, to solve this problem. ESP can eliminate the problem caused by Desynchronization Attack and help detecting Replay Attack. Analysis shows that ESP enhances privacy protection while still maintaining the authentication efficiency.

1. Introduction

Radio Frequency Identification (RFID) technologies are on their highway to pervasive usage [1, 2]. It can enable automatic identification wirelessly. However privacy protection is still an important problem since RFID tags attached to items are so cost constrained to use complex safety measures. Many approaches [6] are proposed to protect information on tags. These private authentication approaches are paying more and more attention to privacy protecting. Usually RFID systems use the challenge-response scheme to authenticate tags. Typically the reader and each tag share the function(s) and distinct key(s). Due to the limited storage and computation ability, the functions are usually cryptographic hash functions and the keys are symmetric. The connection between the reader and the backend database which manages the keys is secure.

When authenticating a tag, the reader first queries the tag. The tag replies with a secret message which is then checked by the reader to determine if the tag is legal. Because the secret message is generated by the tag using its distinct key(s), the reader can search in backend database to find the matching key(s) and authenticate the tag. We briefly use *T* to denote tag and *R* or reader to denote the combination of reader and backend database in the following.

Protocols for RFID private authentication can be roughly classified into two groups, tree-based and non-tree based.

Tree-based approaches usually store keys in a virtual tree and use virtual nodes to hold keys. Walk down the virtual tree, usually called key tree, from root to a leaf node can help find a corresponding tag. The way walk down is to verify the keys stored on such a path. Search complexity and compute complexity for such a tree-based structure are usually $O(\log N)$, where N is the number of tags in the system. The tags are structurally organized, which introduces efficiency as well as flaw. The tree-based approaches usually are vulnerable to *Compromising Attack*. Tags in RFID system can be easily obtained by an adversary, who may then do compromising actions to obtain keys stored in them. As each pair of tags share at least one key in the tree structure, leaking keys in tag(s) will incur information leaking of uncompromised tags. Typically, *Compromising Attack* can help an adversary to track tags [4].

Non-tree-based protocols usually organize keys in linear style. In linear style tags share no key such that compromising some tags will not leak information in uncompromised ones. However linear structure causes $O(N)$ complexity for searching a key, which is not efficient enough for RFID systems with huge number of tags. A special kind of non-tree-based protocol based on synchronizing state between reader and tags

This work was supported in part by China 863 Program under grants No. 2009AA011903 and 2006AA01Z101, China 973 Program under grant No. 2006CB30300, NSFC under grant No. 60873262, Shaanxi ISTC under grant No. 2008KW-02, Doctoral Fund under grant No. 20060698018, IBM Joint Project under grant No. JLP200906008-1.

978-1-4244-5113-5/09/$25.00 ©2009

can avoid the brute-force key search, they are called the synchronization approaches. In synchronization approaches, a tag maintains a counter corresponding to each query and the valid reader roughly knows the current value of the counter. The reader pre-computes a table of tag output values for key search. During authentication, the tag outputs with the values of its key and counter, and the reader searches in pre-computed table to find the corresponding key. Search complexity for synchronization based approaches is usually $O(1)$. But they suffer from *Desynchronization Attack* [5]. As tags are designed to automatically respond to interrogations, a tag can be maliciously scanned. In *Desynchronization Attack* when continuously scanned by the adversary, information stored in a tag varies more and more from the version stored in valid reader. The adversary can launch such *Desynchronization Attack* easily by just keep querying a tag. When the difference between tag side and reader side is big enough, the tag can be recognized. An adversary can utilize this point to track a tag.

In this work, we propose an Enhanced Synchronization aPproach for RFID private authentication, ESP, to eliminate problems caused by *Desynchronization Attack*.

In ESP each tag maintains an inside counter which increases upon each query. The counter is kept inside the tag and never sent out in plain text and the adversary is not aware of the value of the counter, so that the adversary cannot track a tag by observing the counter. The counter increases upon each query, so that if a reply is generated using an older counter, the reader can find it and launch relative warnings. What's more, ESP still remains the $O(1)$ authentication efficiency of synchronization approaches.

Our contribution:

ESP focus on the weak points of traditional synchronization approaches. The contributions of this work are twofold.

First, we find that synchronization approaches which send synchronization messages, usually the counter, in plain text is obviously vulnerable to tracking. And tag acts different when their counter comes to a pre-defined *threshold* is the other reason. We then eliminate plain text counters and keep the same behavior of tag all the time in ESP.

Second, we find that no conversation relative information is used to generate the replies. So that synchronization approaches are vulnerable to *Replay Attacks*. We then record a counter and the used keys, so that if a reply is found using a past counter or key, it must be a replayed one.

As ESP pre-computes a table for looking up outputs of tags, it remains the authentication efficiency of synchronization approaches. In the table, the reader keeps some pre-computed values according to different counter values for each tag. The cost of each tag is $O(1)$, considering storage and computing complexity. For the reader side, it takes $O(1)$ complexity for searching a tag output in the table, considering pre-computing can be done offline. It takes $O(range \bullet N)$ storage for the pre-computed table, where *range* is the number of output values kept for each tag. It's $O(N)$ as the number of stored outputs for each tag can be a pre-defined value.

Analysis shows that ESP enhances protection against both tracking and *Replay Attacks* for synchronization based RFID private authentication and still remains the efficient authentication character.

The rest of this paper is organized as follows. In Section 2 we present the security requirements ESP focuses on. In Section 3 we discuss the existing works. In Section 4 we present the ESP design. In Section 5 we analyze the security and performance of ESP. In Section 6, we conclude the paper and present the future work.

2. Security requirements

In RFID private authentication, following security requirements [3, 4, and 22] are considered fundamental.

Confidential. Any private information should not be leaked in the process of authentication.

Untraceability. Tags should not be able to track because of messages sent during the process of authentication.

Cloning resistance. Legal readers cannot be fooled by faked or impersonated tags. Replaying response from a tag in a later authentication conversation, which is called *Replay Attack*, should not be feasible.

Forward secrecy. Knowing current key should not help to reveal information in previously sent messages.

Compromising resistance. Compromising some tags and thus obtaining their keys will have minimal impact on uncompromised tags.

In traditional synchronization approaches, confidential and compromising resistance characters are fulfilled. In ESP we mainly focus on untraceability and resistance to *Replay Attack*. We will show the ability of ESP of fulfilling them as well as forward secrecy in Section 5.

3. Related work

Protocols [6-9] for RFID private authentication can be roughly classified into two groups, tree-based and non-tree based.

Tree-based approaches [3, 4, and 17] usually store keys in a virtual tree and use virtual nodes to hold keys. Some approaches using similar structure components [18, 23] are also included in this group. Walk down the virtual tree, usually called key tree, from root to a leaf node can help find a corresponding tag. The way walk down is to verify the keys stored on such a path. Search complexity and compute complexity for such a tree-based structure are usually $O(\log N)$, where N is the number of tags in the system. The tags are structurally organized, which introduces efficiency as well as flaw. The tree-based approaches usually are vulnerable to *Compromising Attack*. Tags in RFID system can be easily obtained by an adversary, who may then do compromising actions to obtain keys stored in them. As each pair of tags share at least one key in the tree structure, leaking keys in tag(s) will incur information leaking of uncompromised tags. Typically, *Compromising Attack* can help an adversary to track tags [4].

Non-tree-based protocols [5, 10] usually organize keys in linear style. In linear style tags share no key such that compromising some tags will not leak information in uncompromised ones. However linear structure causes $O(N)$ complexity for searching a key, which is not efficient enough for RFID systems with huge number of tags. A trade-off approach [12] is proposed to achieve the $O(N^{2/3})$ search complexity, but it's still not satisfying.

A special kind of non-tree-based protocol based on synchronizing state between reader and tags can avoid the brute-force key search, they are called the synchronization approaches. In synchronization approaches, a tag maintains a counter corresponding to each query and the valid reader roughly knows the current value of the counter. The reader pre-computes a table of tag output values for key search. During authentication, the tag outputs with the values of its key and counter, and the reader searches in pre-computed table to find the corresponding key. Search complexity for synchronization based approaches are usually $O(1)$. But they suffer from *Desynchronization Attack* [5]. As tags are designed to automatically respond to interrogations, a tag can be maliciously scanned. In *Desynchronization Attack* when continuously scanned by the adversary, information stored in a tag varies more and more from the version stored in valid reader. This provides a character for the adversary to track the tags. The adversary can launch such *Desynchronization Attack* easily by just keep querying a tag. When the difference between tag side and reader side is big enough, the tag can be recognized. An adversary can utilize this point to track a tag. In [11] Ohkubo et al. limit the number of outputs from each tag to a const value. In [13], Henrici et al. let the reader to check a counter for each tag which records the successful authentication times. In [14], Juels propose a method to authenticate tags by looping through a window. When maliciously scanned many times, counters in tags using these approaches would be distinctly large to be recognized for tracking. In [15], two conversation nonces are used to resist against *Replay Attack*, but it loss the efficient character.

Other works mainly consider physical or implementary factors. In [19], Lim et al. mask the communication channel by RF waves, which incurs inconvenience for tag owners to carry a jamming device. In [20], Molnar et al. propose a new method to provide the ability of authentication delegation. In [21], Tan et al. propose a server-free scheme. However, these approaches do not provide an efficient search method.

4. ESP protocol

In this Section, we present the design of ESP. The goal is to provide untraceability and resistance to *Replay Attack* while remaining the efficiency of the Synchronization approaches. ESP comprises of three components, *System initialization*, *Authentication process*, and *Key-updating process*.

4.1 System initialization

Each tag T_j in T has a one-way hash function $h(\cdot)$, a current key K_j, a pseudo random number generator (*PRNG*), a counter $counter_j$ and a system parameter *range*. The *PRNG* generates random numbers with the same length of the output from $h(\cdot)$. The counter records the uncompleted times since last successful authentication. Initially the value of $counter_j$ is 0.

R holds the same hash function $h(\cdot)$, *PRNG* and the *range* as the tags hold. R also holds a hash table *Table* consisting of $(h(K_j, counter_i), K_j)$ for each $1 \le j \le N$ and $0 \le i \le range$.

4.2 Authentication process

The identification process comprises four phases, relative to the three message rounds between tag and reader, as illustrated in Fig. 1.

1) The reader R sends out a query message to a tag T_j, comprising a request and a random number r_1 generated by *PRNG*.

2) Upon the query, the tag first generates a random number r_2 by *PRNG*. Then it checks if the failure time $counter_j$ is larger than *range*. If not, it means the tag has not been crashingly attacked yet. For these normal tags, a respond message component I is generated as $h(K_j, counter_j)$. If $counter_j$ is larger than *range* I is generated as $h(0, K_j, r_1, r_2)$. Both r_2 and I are sent to R as the response. Then $counter_j$ is increased by 1.

3) When R receives the response from T_j, it starts to determine whether it is a legal tag. A search in *Table* is first launched, to find a $h(K_j', counter_i)$ which equals to I. If there exists such a K_j', then R can determine the tag is T_j relative to K_j'. If no such K_j' exists, the tag should not be a normal tag, either crashingly attacked or illegal. Then R computes $h(0, K_j', r_1, r_2)$ for each K_j'. If there exists a K_j, T_j should be the tag being identified. If still can not find such a K_j' the tag is determined as an illegal one. For a legal tag R computes a message component O as $h(1, K_j, r_1, r_2)$, while for a illegal one O is generated by *PRNG*. At last, R sends O to the tag as an authentication message for the reader. If the reader determines that the tag is a legal one, it launches key-updating process.

4) The tag receives O, and checks that if O is the outcome of $h(1, K_j, r_1, r_2)$. If it is, R is authenticated as a legal reader. Then $counter_j$ is set to 0 and the tag launches key-updating process.

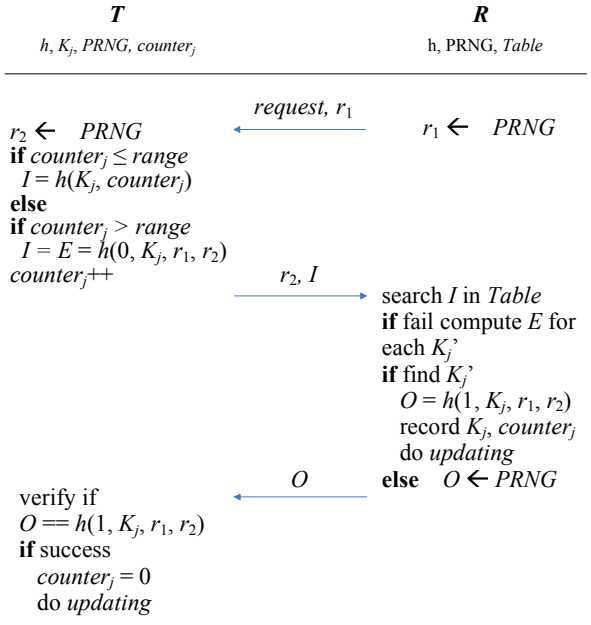

Figure 1. The ESP Protocol

In short, for a not crashingly attacked tag ESP offers fast identification by using a pre-stored table. For a badly attacked one ESP maintains the functionality.

4.3 Key-updating process

After a successful authentication, the reader and tag start the key-updating process, as mentioned above. The **Algorithm 1** illustrated this process.

Algorithm 1: *Updating*

R : $K_j' \leftarrow h(2, K_j, r_1, r_2)$
delete K_j
for $i = 0$ to *range*
compute $h(K_j', i)$
insert $h(K_j', i)$
delete $h(K_j, i)$

T_j : $counter_j = 0$
$K_j' \leftarrow h(2, K_j, r_1, r_2)$
delete K_j

Figure 2. Updating Algorithm

The updating process of ESP has two things to do. The first is to do key-updating in both sides to provide forward privacy for tags. The second is to update the pre-stored *Table* in R and set the counter in the tag to 0. When the key of a tag is updated, the hash values should be updated the same time to keep fast in the succeed authentications.

5. Analysis

In this Section we give an analysis on our protocol, considering both the security assurance of ESP. We first show ESP provides the general security requirements proposed in Section 2. We then show ESP in defending against *Desynchronization Attack*. At last we analyze the performance of ESP.

Confidential: In ESP, all private information is always hashed before being sent out. The function $h(\cdot)$ is a one way hash function, thus an adversary can hardly know any private information other than break the hash function in brute force.

Cloning resistance: In ESP, we maintain in the reader side a counter and all used keys for each tag. If the adversary query the tag and replay the responses later, it will be found using an old key or old counter with current key. Thus it is ESP can help detect *Replay Attacks*.

Forward secrecy: The adversary might obtain the tag's key after successfully compromising it. If the key has never been updated, the adversary can use the key to recover the messages sent before. Some existing works suffer from such attacks due to the lack of key-updating [3, 20]. In ESP, a tag will update its key after each successful mutual authentication. The key-updating process of ESP computes the hash values of old keys, and allocates those values as the new keys. In this process, it is negligible for attackers to successfully deduce the previous keys from current ones. Therefore, ESP guarantees the forward privacy for tags.

Untraceability: In traditional synchronization methods, synchronization messages are usually sent in plain text which makes the tag distinguishable. An adversary can simply interrogate the tag and observe the responses to track it. And for those methods which keep pre-computed hash values, when *Desynchronization Attack* forces the tag to go beyond the range, either random numbers or identical messages are sent, the former makes the functionality unavailable while the latter enables tracking.

In ESP a tag sends out responses on each query, consisting r_2 and I where r_2 is generated as a random nonce. Before *range* is met, I varies each time utilizing secrete key K_j and embedded $counter_j$ as inputs of hash function $h(\cdot)$. Because the adversary has no information of K_j and r_2 in advance, it cannot separate one tag from another from the hash values, saying no tracking.

When the embedded counter goes beyond *range*, the form of the responses is kept the same as when *range* is not met. So it's not applicable to separate a tag by desynchronization attacking it to force the embedded counter going beyond *range*. In this case I varies each time utilizing secrete key K_j and two random nonces as inputs of hash function $h(\cdot)$. Similar to analysis above, the adversary cannot do tracking.

5.1 *Desynchronization Attack* resistance

The case for *Desynchronization Attack* is little complicated, as it has close relation to forward secrecy and tracking. Forward secrecy needs updating, or else information in tag stay the same as time passes will arise the risk of tracking.

The *Desynchronization Attack* can be launched between successful authentications. If the tag launches updating under each query after the first response from tag as in [11], the adversary can interrogate a tag for desynchronization easily. After many times, the state of the tag and its record managed by the reader will be desynchronized to a great degree and thus prohibitively costly to be authenticated.

Some approaches, such as [13] chose to use a counter number or other plain synchronizing message to avoid *Desynchronization Attacks*, but these messages bring in new weak point for tracking. An adversary can simply query a tag to observe its regular responses for recognizing it and further tracking it.

The key updating process of ESP does not take place between two successful authentications but after successful mutual authentications to avoid desynchronization of the keys. Only the embedded $counter_j$ is updated under each query, and it is never sent in plain text. In this way the response before *range* is met varies on each query to avoid tracking.

Note that the adversary may interrupt a tag when the tag is receiving the third round message, and hence make the keys stored in the tag and the reader temporally desynchronized. To deal with this problem, we can store an old key in the backend database until the next successful authentication, which needs to store two keys for a tag. If the fast authentication for the old key is needed, the hash values of this old key should also be stored, which cause a double storage cost.

5.2 Performance

Both tag side storage and computing complexity for ESP are $O(1)$. ESP needs to pre-compute and store hash values for each tag for facilitating fast authentication. Thus the reader side storage complexity for ESP is $O(N)$, considering *range* is a const, so is the reader side pre-computing cost. Noticing that the pre-computing can be done offline, the computing cost we care is the amount during authentication.

The reader side search efficiency for a tag which is not crashingly attacked is $O(1)$, while for a crashingly attacked tag is $O(N)$. We define *cc* as sum of storage and computing complexity at reader side. Then the compound complexity at reader side for ESP is $O(N)$, which is no bad than other approaches. Considering the numbers of RFID tags in practical systems are huge, the ratio that a tag is crashingly attacked is small. What's more, a crashingly attacked tag can still be successfully authenticated and become a normal one for fast authentication next time. The traditional synchronization methods, though efficient in both tag side and reader side, will either lose functionality or break privacy of the tag owners when tags are crashingly attacked.

6. Conclusion and future work

In this work we present an enhanced synchronization approach, ESP. In ESP a tag keeps the

same behavior all the time to obtain resistance to *Desynchronization Attack* and store history information to detect *Replay Attack*. ESP also provides forward secrecy while remains the authentication efficiency of traditional synchronization approaches. Analysis shows that ESP enhances synchronization approach in both the security and privacy.

Our future work will focus on reducing the order of storage and seeking for an optimal time-space trade-off for real deployment.

References

[1] P. De, K. Basu and S. K. Das, "An Ubiquitous Architectural Framework and Protocol for Object Tracking using RFID Tags", in Proceedings of ACM MobiQuitous Networking Conference, 2004.

[2] M. Mamei, F. Zambonelli, "Pervasive Pheromone-based Interactions with RFID Tags", *ACM Transactions on Autonomous and Adaptive Systems*, Vol. 2, Iss. 2, June 2007.

[3] T. Dimitriou, "A Secure and Efficient RFID Protocol that could make Big Brother (partially) Obsolete", in Proceedings of IEEE PerCom, 2006.

[4] L. Lu, J. Han, L. Hu, Y. Liu, and L. M. Ni, "Dynamic Key-Updating: Privacy-Preserving Authentication for RFID Systems", in Proceedings of IEEE PerCom, 2007.

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

[6] G. Avoine, "Bibliography on Security and Privacy in RFID Systems", the Information Security Group of UCL (Belgium), available online at http://www.avoine.net/rfid/, 2009.

[7] M. Lehtonen, T. Staake, F. Michahelles, and E. Fleisch, "From Identification to Authentication – A Review of RFID Product Authentication Techniques", in Proceedings of Workshop on RFID Security (RFIDSec), 2006.

[8] G. Avoine, "Cryptography in Radio Frequency Identification and Fair Exchange Protocols", Ph.D. thesis, EPFL, Lausanne, Switzerland, 2005.

[9] A. Juels, "RFID Security and Privacy: a Research Survey", *Selected Areas in Communication, IEEE Journal on*, Vol. 24, No. 2, 2006, pp. 381-394.

[10] S. Weis, S. Sarma, R. Rivest and D. Engels, "Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems", in Proceedings of SPC, 2003.

[11] M. Ohkubo, K. Suzuki, and S. Kinoshita, "Efficient Hash-Chain based RFID Privacy Protection Scheme", in Proceedings of UbiComp, Workshop Privacy: Current Status and Future Directions, 2004.

[12] G. Avoine, E. Dysli, and P. Oechslin, "Reducing Time Complexity in RFID Systems", in Proceedings of SAC, 2005.

[13] D. Henrici and P. Müller, "Hash-based Enhancement of Location Privacy for Radio-Frequency Identification Devices Using Varying Identifiers," in Proceedings of Pervasive Computing and Communications Workshops, 2004.

[14] A. Juels, "Minimalist Cryptography for Low-Cost RFID Tags", in Proceedings of SCN, 2004.

[15] T. Dimitriou, "A Lightweight RFID Protocol to Protect Against Traceability and Cloning Attacks", in Proceedings of SecureComm, 2005.

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

[17] D. Molnar and D. Wagner, "Privacy and Security in Library RFID: Issues, Practices, and Architectures", in Proceedings of ACM CCS, 2004.

[18] W. Wang, Y. Li, L. Hu, L. Lu, "Storage-Awareness: RFID Private Authentication based on Sparse Tree", in Proceedings of Security, Privacy and Trust in Pervasive and Ubiquitous Computing, 2007.

[19] T. L. Lim, T. Li, and S. L. Yeo , "Randomized Bit Encoding for Stronger Backward Channel Protection", in Proceedings of IEEE PerCom, 2008.

[20] D. Molnar, A. Soppera, and D. Wagner, "A Scalable, Delegatable Pseudonym Protocol Enabling Owner-ship Transfer of RFID Tags", in Proceedings of SAC, 2005.

[21] C. Tan, B. Sheng, and Q. Li, "Serverless Search and Authentication Protocols for RFID", in Proceedings of IEEE PerCom, 2007.

[22] Q. Yao, Y. Qi, J. Han, J. Zhao, X. Li, and Y. Liu, "Randomizing RFID Private Authentication", in Proceedings of IEEE PerCom, 2009.

[23] L. Lu, J. Han, and Y. Liu, "ACTION: Breaking the Privacy Barrier for RFID Systems ", in Proceedings of IEEE INFOCOM, 2009.